Configure IP ACLs

This chapter describes how to configure IP access control lists (ACLs) on Cisco NX-OS devices.

Unless otherwise specified, the term IP ACL refers to IPv4 and IPv6 ACLs.

About ACLs

An ACL is an ordered set of rules that you can use to filter traffic. Each rule specifies a set of conditions that a packet must satisfy to match the rule. When the device determines that an ACL applies to a packet, it tests the packet against the conditions of all rules. The first matching rule determines whether the packet is permitted or denied. If there is no match, the device applies the applicable implicit rule. The device continues processing packets that are permitted and drops packets that are denied.

You can use ACLs to protect networks and specific hosts from unnecessary or unwanted traffic. For example, you could use ACLs to disallow HTTP traffic from a high-security network to the Internet. You could also use ACLs to allow HTTP traffic but only to specific sites, using the IP address of the site to identify it in an IP ACL.

ACL Types and Applications

The device supports the following types of ACLs for security traffic filtering:

IPv4 ACLs
The device applies IPv4 ACLs only to IPv4 traffic.
IPv6 ACLs
The device applies IPv6 ACLs only to IPv6 traffic.
MAC ACLs
The device applies MAC ACLs only to non-IP traffic.

IP and MAC ACLs have the following types of applications:

Port ACL
Filters Layer 2 traffic
MAC ACL with UDF-based match
Filters MAC ACLs with UDF-based match
Router ACL
Filters Layer 3 traffic
VLAN ACL
Filters VLAN traffic
VTY ACL
Filters virtual teletype (VTY) traffic

This table summarizes the applications for security ACLs.

Table 1: Security ACL Applications

Application	Supported Interfaces	Types of ACLs Supported
Port ACL	• Layer 2 interfaces • Layer 2 Ethernet port-channel interfaces When a port ACL is applied to a trunk port, the ACL filters traffic on all VLANs on the trunk port.	• IPv4 ACLs • IPv4 ACLs with UDF-based match • IPv6 ACLs • IPv6 ACLs with UDF-based match • MAC ACLs • MAC ACLs with UDF-based match
Router ACL	• VLAN interfaces • Physical Layer 3 interfaces • Layer 3 Ethernet subinterfaces • Layer 3 Ethernet port-channel interfaces • Management interfaces **Note** You must enable VLAN interfaces globally before you can configure a VLAN interface.	• IPv4 ACLs • IPv6 ACLs **Note** MAC ACLs are supported on Layer 3 interfaces only if you enable MAC packet classification.
VLAN ACL	• VLANs	• IPv4 ACLs • IPv6 ACLs • MAC ACLs

Application	Supported Interfaces	Types of ACLs Supported
VTY ACL	• VTYs	• IPv4 ACLs • IPv6 ACLs

Related Topics

About VLAN ACLs
About MAC ACLs

Order of ACL Application

When the device processes a packet, it determines the forwarding path of the packet. The path determines which ACLs that the device applies to the traffic. The device applies the ACLs in the following order:

1. Port ACL
2. Ingress VACL
3. Ingress router ACL
4. Ingress VTY ACL
5. Egress VTY ACL
6. Egress router ACL
7. Egress VACL

If the packet is bridged within the ingress VLAN, the device does not apply router ACLs.

Figure 1: Order of ACL Application

The following figure shows the order in which the device applies ACLs.

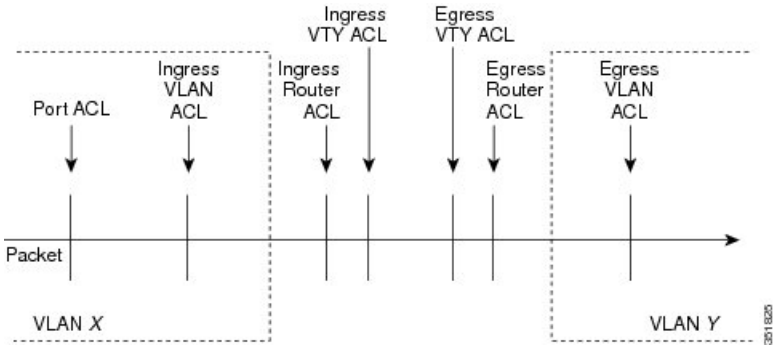
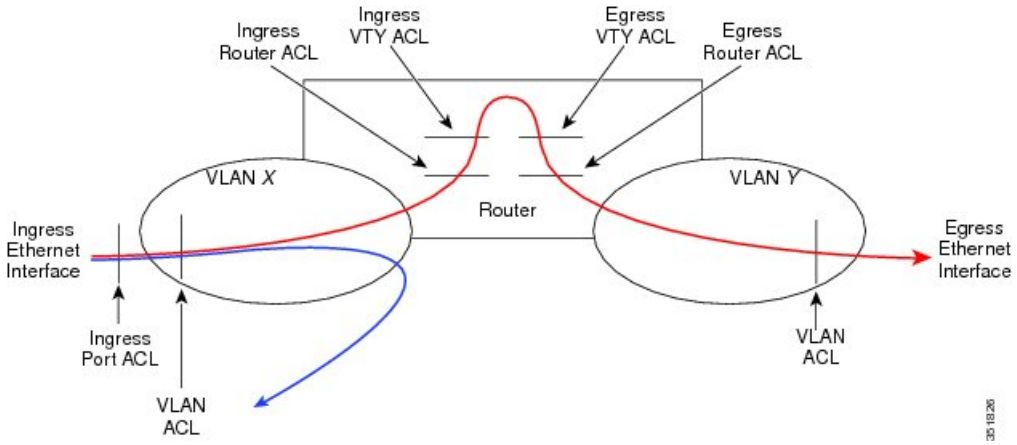


Figure 2: ACLs and Packet Flow

The following figure shows where the device applies ACLs, depending upon the type of ACL. The red path indicates a packet sent to a destination on a different interface than its source. The blue path indicates a packet that is bridged within its VLAN.

The device applies only the applicable ACLs. For example, if the ingress port is a Layer 2 port and the traffic is on a VLAN that is a VLAN interface, a port ACL and a router ACL both can apply. In addition, if a VACL is applied to the VLAN, the device applies that ACL too.



About Rules

Rules are what you create, modify, and remove when you configure how an ACL filters network traffic. Rules appear in the running configuration. When you apply an ACL to an interface or change a rule within an ACL that is already applied to an interface, the supervisor module creates ACL entries from the rules in the running configuration and sends those ACL entries to the applicable I/O module. Depending upon how you configure the ACL, there may be more ACL entries than rules, especially if you implement policy-based ACLs by using object groups when you configure rules.

You can create rules in access-list configuration mode by using the **permit** or **deny** command. The device allows traffic that matches the criteria in a permit rule and blocks traffic that matches the criteria in a deny rule. You have many options for configuring the criteria that traffic must meet in order to match the rule.

This section describes some of the options that you can use when you configure a rule.

Protocols for IP ACLs and MAC ACLs

IPv4, IPv6, and MAC ACLs allow you to identify traffic by protocol. For your convenience, you can specify some protocols by name. For example, in an IPv4 or IPv6 ACL, you can specify ICMP by name.

You can specify any protocol by number. In MAC ACLs, you can specify protocols by the EtherType number of the protocol, which is a hexadecimal number. For example, you can use 0x0800 to specify IP traffic in a MAC ACL rule.

In IPv4 and IPv6 ACLs, you can specify protocols by the integer that represents the Internet protocol number.

Source and Destination

In each rule, you specify the source and the destination of the traffic that matches the rule. You can specify both the source and destination as a specific host, a network or group of hosts, or any host. How you specify the source and destination depends on whether you are configuring IPv4 ACLs, IPv6 ACLs, or MAC ACLs.

Implicit Rules for IP and MAC ACLs

IP and MAC ACLs have implicit rules, which means that although these rules do not appear in the running configuration, the device applies them to traffic when no other rules in an ACL match. When you configure the device to maintain per-rule statistics for an ACL, the device does not maintain statistics for implicit rules.

All IPv4 ACLs include the following implicit rule:

```
deny ip any any
```

This implicit rule ensures that the device denies unmatched IP traffic.

All IPv6 ACLs include the following implicit rule:

```
deny ipv6 any any
```

This implicit rule ensures that the device denies unmatched IPv6 traffic.

Note

- IPv6 Neighbor Discovery packets (Router Solicitation, and Router Advertisement) will not be permitted due to the implicit **deny ipv6 any any** rule of an IPv6 ACL.
- You must add the following rules explicitly to allow IPv6 Neighbor Discovery packets in the Cisco Nexus 93180YC-EX, Nexus 93180YC-FX, Nexus 93240YC-FX2, Nexus 93360YC-FX2, Nexus 9336C-FX2, Nexus 9336C-FX2-E, Nexus 93180YC-FX3, N9K-C9316D-GX, N9K-C93600CD-GX, Nexus 9364C-GX, N9K-C9332D-GX2B, Nexus 9364C and Nexus 9332C platform switches:
  - **permit icmp any any router-advertisement**
  - **permit icmp any any router-solicitation**
- Neighbor Solicitation (NS) and Neighbor Advertisement (NA) messages do not match under the implicit rule. The following commands are required to match the NS or NA IPv6 traffic.
  - **permit/deny icmp any any nd-na**
  - **permit/deny icmp any any nd-ns**

All MAC ACLs include the following implicit rule:

```
deny any any protocol
```

This implicit rule ensures that the device denies the unmatched traffic, regardless of the protocol specified in the Layer 2 header of the traffic.

Additional Filtering Options

You can identify traffic by using additional options. These options differ by ACL type. The following list includes most but not all additional filtering options:

- IPv4 ACLs support the following additional filtering options:
  - Layer 4 protocol
  - TCP and UDP ports
  - ICMP types and codes
  - IGMP types
  - Precedence level
  - Differentiated Services Code Point (DSCP) value
  - TCP packets with the ACK, FIN, PSH, RST, SYN, or URG bit set
  - Established TCP connections

- Packet length

- IPv6 ACLs support the following additional filtering options:
  - Layer 4 protocol
  - Encapsulating Security Payload
  - Payload Compression Protocol
  - Stream Control Transmission Protocol (SCTP)
  - SCTP, TCP, and UDP ports
  - ICMP types and codes
  - DSCP value
  - TCP packets with the ACK, FIN, PSH, RST, SYN, or URG bit set
  - Established TCP connections
  - Packet length

- MAC ACLs support the following additional filtering options:
  - Layer 3 protocol (Ethertype)
  - VLAN ID
  - Class of Service (CoS)

- Beginning Cisco NX-OS Release 9.2(4), IPv4 ACLs and IPv6 in Cisco Nexus 9500 platform switches with N9K-X96136YC-R, N9K-X9636C-R, and N9K-X9636C-RX line cards and N9K-C9504-FM-R fabric module support the following additional filtering options:
  - TCP packets with the ACK, FIN, PSH, RST, SYN, or URG bit set
  - Established TCP connections

Note

- TCP flag options are correctly processed by Netstack rather than the kernel (KStack), due to the kernel's lack of support for TCP flags. Additionally, the following syslog message is generated:

  ```
  <HOSTNAME> %NPACL-2-IPT_WARNING: npacl [<#>] WARNING: Mgmt ACL: <ACL>
  Seq:<Seq#> has ACL option: tcp-flags that is not supported in kernel
  stack. Hence that option is not added in its filter rule.
  ```

- The **tcp-flags-mask** option is not supported.

Sequence Numbers

The device supports sequence numbers for rules. Every rule that you enter receives a sequence number, either assigned by you or assigned automatically by the device. Sequence numbers simplify the following ACL tasks:

Adding new rules between existing rules

By specifying the sequence number, you specify where in the ACL a new rule should be positioned. For example, if you need to insert a rule between rules numbered 100 and 110, you could assign a sequence number of 105 to the new rule.

Removing a rule

Without using a sequence number, removing a rule requires that you enter the whole rule, as follows:

```
switch(config-acl)# no permit tcp 10.0.0.0/8 any
```

However, if the same rule had a sequence number of 101, removing the rule requires only the following command:

```
switch(config-acl)# no 101
```

Moving a rule

With sequence numbers, if you need to move a rule to a different position within an ACL, you can add a second instance of the rule using the sequence number that positions it correctly, and then you can remove the original instance of the rule. This action allows you to move the rule without disrupting traffic.

If you enter a rule without a sequence number, the device adds the rule to the end of the ACL and assigns a sequence number that is 10 greater than the sequence number of the preceding rule to the rule. For example, if the last rule in an ACL has a sequence number of 225 and you add a rule without a sequence number, the device assigns the sequence number 235 to the new rule.

In addition, Cisco NX-OS allows you to reassign sequence numbers to rules in an ACL. Resequencing is useful when an ACL has rules numbered contiguously, such as 100 and 101, and you need to insert one or more rules between those rules.

Logical Operators and Logical Operation Units

IP ACL rules for TCP and UDP traffic can use logical operators to filter traffic based on port numbers. Cisco NX-OS supports logical operators in only the ingress direction.

The device stores operator-operand couples in registers called logical operator units (LOUs). The LOU usage for each type of operator is as follows:

eq
Is never stored in an LOU

gt
Uses 1 LOU

lt
Uses 1 LOU

neq
Uses 1 LOU

range
Uses 1 LOU

Note For range operators, LOU threshold configuration is used to control how the port range is expanded when configuring an ACL entry. If you want to use the LOU operator when the number of the ACL rules exceed the configured threshold value, run the following command: **hardware access-list lou resource threshold** <*x*>, wherein <*x*> denotes the number of ACL rules to be used before the LOU threshold is reached. The range value for <*x*> is 1 to 50, and the default value for LOU threshold is 5.

ACL Logging

The ACL logging feature monitors ACL flows and logs statistics.

A flow is defined by the source interface, protocol, source IP address, source port, destination IP address, and destination port values. The statistics maintained for a flow include the number of forwarded packets (for each flow that matches the permit conditions of the ACL entry) and dropped packets (for each flow that matches the deny conditions of the ACL entry).

Time Ranges

You can use time ranges to control when an ACL rule is in effect. For example, if the device determines that a particular ACL applies to traffic arriving on an interface, and a rule in the ACL uses a time range that is not in effect, the device does not compare the traffic to that rule. The device evaluates time ranges based on its clock.

When you apply an ACL that uses time ranges, the device updates the affected I/O module whenever a time range referenced in the ACL starts or ends. Updates that are initiated by time ranges occur on a best-effort priority. If the device is especially busy when a time range causes an update, the device may delay the update by up to a few seconds.

IPv4, IPv6, and MAC ACLs support time ranges. When the device applies an ACL to traffic, the rules in effect are as follows:

- All rules without a time range specified
- Rules with a time range that includes the second when the device applies the ACL to traffic

The device supports named, reusable time ranges, which allows you to configure a time range once and specify it by name when you configure many ACL rules. Time range names have a maximum length of 64 alphanumeric characters.

A time range contains one or more rules. The two types of rules are as follows:

Absolute

A rule with a specific start date and time, specific end date and time, both, or neither. The following items describe how the presence or absence of a start or end date and time affect whether an absolute time range rule is active:

- Start and end date and time both specified—The time range rule is active when the current time is later than the start date and time and earlier than the end date and time.
- Start date and time specified with no end date and time—The time range rule is active when the current time is later than the start date and time.

- No start date and time with end date and time specified—The time range rule is active when the current time is earlier than the end date and time.
- No start or end date and time specified—The time range rule is always active.

For example, you could prepare your network to allow access to a new subnet by specifying a time range that allows access beginning at midnight of the day that you plan to place the subnet online. You can use that time range in ACL rules that apply to the subnet. After the start time and date have passed, the device automatically begins applying the rules that use this time range when it applies the ACLs that contain the rules.

Periodic

A rule that is active one or more times per week. For example, you could use a periodic time range to allow access to a lab subnet only during work hours on weekdays. The device automatically applies ACL rules that use this time range only when the range is active and when it applies the ACLs that contain the rules.

Note The order of rules in a time range does not affect how a device evaluates whether a time range is active. Cisco NX-OS includes sequence numbers in time ranges to make editing the time range easier.

Time ranges also allow you to include remarks, which you can use to insert comments into a time range. Remarks have a maximum length of 100 alphanumeric characters.

The device determines whether a time range is active as follows:

- The time range contains one or more absolute rules—The time range is active if the current time is within one or more absolute rules.
- The time range contains one or more periodic rules—The time range is active if the current time is within one or more periodic rules.
- The time range contains both absolute and periodic rules—The time range is active if the current time is within one or more absolute rules and within one or more periodic rules.

When a time range contains both absolute and periodic rules, the periodic rules can only be active when at least one absolute rule is active.

Policy-Based ACLs

The device supports policy-based ACLs (PBACLs), which allow you to apply access control policies across object groups. An object group is a group of IP addresses or a group of TCP or UDP ports. When you create a rule, you specify the object groups rather than specifying IP addresses or ports.

Using object groups when you configure IPv4 or IPv6 ACLs can help reduce the complexity of updating ACLs when you need to add or remove addresses or ports from the source or destination of rules. For example, if three rules reference the same IP address group object, you can add an IP address to the object instead of changing all three rules.

PBACLs do not reduce the resources required by an ACL when you apply it to an interface. When you apply a PBACL or update a PBACL that is already applied, the device expands each rule that refers to object groups into one ACL entry per object within the group. If a rule specifies the source and destination both with object

groups, the number of ACL entries created on the I/O module when you apply the PBACL is equal to the number of objects in the source group multiplied by the number of objects in the destination group.

The following object group types apply to port, router, policy-based routing (PBR), and VLAN ACLs:

IPv4 Address Object Groups

Can be used with IPv4 ACL rules to specify source or destination addresses. When you use the **permit** or **deny** command to configure a rule, the **addrgroup** keyword allows you to specify an object group for the source or destination.

IPv6 Address Object Groups

Can be used with IPv6 ACL rules to specify source or destination addresses. When you use the **permit** or **deny** command to configure a rule, the **addrgroup** keyword allows you to specify an object group for the source or destination.

Protocol Port Object Groups

Can be used with IPv4 and IPv6 TCP and UDP rules to specify source or destination ports. When you use the **permit** or **deny** command to configure a rule, the **portgroup** keyword allows you to specify an object group for the source or destination.

Note Policy-based routing (PBR) ACLs do not support deny access control entries (ACEs) or **deny** commands to configure a rule.

Kernel Stack ACL

The Kernel Stack ACL is a common CLI infrastructure to configure ACLs for management of inband and outband components.

The Kernel Stack ACL uses NX-OS ACL CLI to secure management applications on management and front panel ports. Configuring a single ACL must be able to secure all management applications on NX-OS.

Kernel Stack ACL is the component that fixes the manual intervention of the user and automatically programs iptable entries when the ACL is applied to mgmt0 interface.

The following is an example for configuring Kernel Stack ACL:

```
swtich# conf t
Enter configuration commands, one per line. End with CNTL/Z.
switch(config)# ip access-list kacl1
switch(config-acl)# statistics per-entry
switch(config-acl)# 10 deny tcp any any eq 443
switch(config-acl)# 20 permit ip any any
switch(config-acl)# end
switch#

switch(config-if)# interface mgmt0
switch(config-if)#   ip access-group acl1 in
switch(config-if)#   ipv6 traffic-filter acl6 in
switch(config-if)#

switch# sh ip access-lists kacl1
IP access list kacl1
statistics per-entry
10 deny tcp any any eq 443 [match=136]
```

```
20 permit ip any any [match=44952]
switch(config)#
```

The following is the Kernel Stack filtering for iptables entries based on the configuration:

```
bash-4.4# ip netns exec management iptables -L -n -v --line-numbers
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
num pkts bytes target prot opt in out source destination
1 9 576 DROP tcp -- * * 0.0.0.0/0 0.0.0.0/0 tcp dpt:443
2 0 0 ACCEPT all -- * * 0.0.0.0/0 0.0.0.0/0
3 0 0 DROP all -- * * 0.0.0.0/0 0.0.0.0/0

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
num pkts bytes target prot opt in out source destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
num pkts bytes target prot opt in out source destination
bash-4.4#
```

The following are the limitations for the Kernel Stack ACL support:

- This feature is supported only on mgmt0 interface and not on other inband interfaces.
- Five tuples (protocol, source-ip, destination-ip, source-port, and destination-port) of the ACL entry are programmed in the iptables. Rest of the options provided in the ACL entry are not programmed in the iptables and throws a warning syslog in such instances.

  For example, "WARNING: Some ACL options are not supported in kstack. Only partial rule will be installed".
- If the device user has host bash access, then the user can manually update the iptables. This update could potentially corrupt the iptable rules for which they are programmed.
- The verified maximum number of ACEs is 100 for IPv4 traffic and an additional 100 for IPv6 traffic. Throughput may be impacted if more than this scale is applied.

Statistics and ACLs

The device can maintain global statistics for each rule that you configure in IPv4, IPv6, and MAC ACLs. If an ACL is applied to multiple interfaces, the maintained rule statistics are the sum of packet matches (hits) on all the interfaces on which that ACL is applied.

Note The device does not support interface-level ACL statistics.

For each ACL that you configure, you can specify whether the device maintains statistics for that ACL, which allows you to turn ACL statistics on or off as needed to monitor traffic filtered by an ACL or to help troubleshoot the configuration of an ACL.

The device does not maintain statistics for implicit rules in an ACL. For example, the device does not maintain a count of packets that match the implicit **deny ip any any** rule at the end of all IPv4 ACLs. If you want to maintain statistics for implicit rules, you must explicitly configure the ACL with rules that are identical to the implicit rules.

Related Topics

Monitoring and Clearing IP ACL Statistics, on page 63
Implicit Rules for IP and MAC ACLs, on page 4

Atomic ACL Updates

By default, when a supervisor module of a Cisco Nexus 9000 Series device updates an I/O module with changes to an ACL, it performs an atomic ACL update. An atomic update does not disrupt traffic that the updated ACL applies to; however, an atomic update requires that an I/O module that receives an ACL update has enough available resources to store each updated ACL entry in addition to all pre-existing entries in the affected ACL. After the update occurs, the additional resources used for the update are freed. If the I/O module lacks the required resources, the device generates an error message and the ACL update to the I/O module fails.

If an I/O module lacks the resources required for an atomic update, you can disable atomic updates by using the **no hardware access-list update atomic** command; however, during the brief time required for the device to remove the preexisting ACL and implement the updated ACL, traffic that the ACL applies to is dropped by default.

If you want to permit all traffic that an ACL applies to while it receives a nonatomic update, use the **hardware access-list update default-result permit** command.

This example shows how to disable atomic updates to ACLs:

```
switch# config t
switch(config)# no hardware access-list update atomic
```

This example shows how to permit affected traffic during a nonatomic ACL update:

```
switch# config t
switch(config)# hardware access-list update default-result permit
```

This example shows how to revert to the atomic update method:

```
switch# config t
switch(config)# no hardware access-list update default-result permit
switch(config)# hardware access-list update atomic
```

Session Manager Support for IP ACLs

Session Manager supports the configuration of IP and MAC ACLs. This feature allows you to verify ACL configuration and confirm that the resources required by the configuration are available prior to committing them to the running configuration.

ACL TCAM Regions

You can change the size of the ACL ternary content addressable memory (TCAM) regions in the hardware.

On Cisco Nexus 9300 and 9500 Series switches and Cisco Nexus 3164Q, 31128PQ, 3232C, and 3264Q switches, the egress TCAM size is 1K, divided into four 256 entries. On Cisco Nexus NFE2-enabled devices (such as the Cisco Nexus 3232C and 3264Q switches), the ingress TCAM size is 6K, divided into twelve 512 slices. Three slices are in one group. On other Cisco Nexus 9300 and 9500 Series switches and the 3164Q and 31128PQ switches, the ingress TCAM size is 4K, divided into eight 256 slices and four 512 slices. A slice is the unit of allocation. A slice can be allocated to one region only. For example, a 512-size slice cannot be used to configure two features of size 256 each. Similarly, a 256-size slice cannot be used to configure two features of size 128 each. The IPv4 TCAM regions are single wide. The IPv6, QoS, MAC, CoPP, and system

TCAM regions are double wide and consume double the physical TCAM entries. For example, a logical region size of 256 entries actually consumes 512 physical TCAM entries.

You can create IPv6, port ACLs, VLAN ACLs, and router ACLs, and you can match IPv6 and MAC addresses for QoS. However, Cisco NX-OS cannot support all of them simultaneously. You must remove or reduce the size of the existing TCAM regions (TCAM carving) to enable the IPv6, MAC, or other desired TCAM regions. For every TCAM region configuration command, the system evaluates if the new change can be fit in the TCAM. If not, it reports an error, and the command is rejected. You must remove or reduce the size of existing TCAM regions to make room for new requirements.

On Cisco Nexus 9200 Series switches, the egress TCAM size is 2K, and the ingress TCAM size is 4K. The concepts of TCAM slices and single- and double-wide regions do not apply to these switches. For example, the ing-ifacl region can host IPv4, IPv6, or MAC type entries. IPv4 and MAC types occupy one TCAM entry whereas IPv6 types occupy two TCAM entries.

For N9K-X9636C-RX, when PACL uses external TCAM region, the internal TCAM needs to take 2K for ifacl and the ingress RACL-IPv4 can take upto 2044. Additional four entries are required when egress PACL external TCAM region is used.

ACL TCAM region sizes have the following guidelines and limitations:

- To enable RACL or PACL on existing TCAM regions, you must carve the TCAM region beyond 12, 288.
- On Cisco Nexus 9300 Series switches, the X9536PQ, X9564PX, and X9564TX line cards are used to enforce the QoS classification policies applied on 40G ports. It has 768 TCAM entries available for carving in 256-entry granularity. These region names are prefixed with "ns-".
- For the X9536PQ, X9564PX, and X9564TX line cards, only the IPv6 TCAM regions consume double-wide entries. The rest of the TCAM regions consume single-wide entries.
- When a VACL region is configured, it is configured with the same size in both the ingress and egress directions. If the region size cannot fit in either direction, the configuration is rejected.
- On Cisco Nexus 9200 Series switches, the ing-sup region occupies a minimum size of 512 entries, and the egr-sup region occupies a minimum size of 256 entries. These regions cannot be configured to lesser values. Any region size can be carved with a value only in multiples of 256 entries (with the exception of the span region, which can be carved only in multiples of 512 entries).
- RACL v6, CoPP, and multicast have default TCAM sizes and these TCAM sizes must be non-zero on the following Cisco Nexus 9504 and Cisco Nexus 9508 line cards to avoid line card failure during reload:
  - N9K-X96136YC-R
  - N9K-X9636C-RX
  - N9K-X9636Q-R
  - N9K-X9636C-R
- When the egress RACL is beyond 4K, the TCAM carving configuration has to be ingress RACL (RACL) + egress RACL (e-racl) summing to 20480. See the following TCAM carving example:

```
hardware access-list tcam region ifacl 0
hardware access-list tcam region ipv6-ifacl 0
hardware access-list tcam region mac-ifacl 0
hardware access-list tcam region racl 0
hardware access-list tcam region ipv6-racl 0
hardware access-list tcam region span 0
```

```
hardware access-list tcam region redirect_v4 0
hardware access-list tcam region redirect_v6 0
hardware access-list tcam region e-racl 20480
```

- You can partially use IPv6 RACL with IPv6 IFCAL. This is applicable to Cisco Nexus N9K-C9508 and N9K-C9504 with N9K-X96136YC-R, N9K-X9636C-R, N9K-X9636Q-R, and N9K-X9636C-RX line cards.
- The N9K-X9636C-R and N9K-X9636Q-R line cards support a maximum TCAM region size of 12K. If you configure a greater number, the TCAM region is set to 12K.
- The N9K-X96136YC-R and N9K-X9636C-R line cards support egress RACL of 2K.
- The N9K-X9636C-RX line card supports a TCAM region size beyond 12K. If you configure the RACL IPv4 TCAM region to 100K, the TCAM region is set to 12K for the N9K-X9636C-R and N9K-X9636Q-R line cards and to 100K for the N9K-X9636C-RX line card, provided you have set all of the other TCAM regions and made space for the N9K-X9636C-R and N9K-X9636Q-R line cards to accommodate 12K.
- Beginning with Cisco NX-OS Release 10.2(2)F, The N9K-X9636C-R and N9K-X9636Q-R line cards support a maximum TCAM region size of 20K. If you configure a greater number, the TCAM region is re-set to 20K.
- In addition to the internal TCAM, an external TCAM of 128K is available on the N9K-X9636C-RX line card.

The following table summarizes the regions that need to be configured for a given feature to work. The region sizes should be selected based on the scale requirements of a given feature.

Table 2: Features per ACL TCAM Region

Feature Name	Region Name
Port ACL	ifacl: For IPv4 port ACLs ifacl-udf: For UDFs on IPv4 port ACLs ing-ifacl: For ingress IPv4, IPv6, and MAC port ACLs ing-ifacl: For ingress IPv4, IPv6, MAC port ACLs, and MAC port ACLs with UDF ipv6-ifacl: For IPv6 port ACLs mac-ifacl: For MAC port ACLs

Feature Name	Region Name
Port QoS (QoS classification policy applied on Layer 2 ports or port channels)	qos, qos-lite, rp-qos, rp-qos-lite, ns-qos, e-qos, or e-qos-lite: For classifying IPv4 packets ing-l2-qos: For classifying ingress Layer 2 packets ipv6-qos, rp-ipv6-qos, ns-ipv6-qos, or e-ipv6-qos: For classifying IPv6 packets mac-qos, rp-mac-qos, ns-mac-qos, or e-mac-qos: For classifying non-IP packets **Note** For traffic that needs to be classified on 40G ports on Cisco Nexus 9300 Series switches, you must carve the qos regions and the corresponding ns-*qos regions.
VACL	vacl: For IPv4 packets ipv6-vacl: For IPv6 packets mac-vacl: For non-IP packets
VLAN QoS (QoS classification policy applied on a VLAN)	vqos or ns-vqos: For classifying IPv4 packets ipv6-vqos or ns-ipv6-vqos: For classifying IPv6 packets ing-l3-vlan-qos: For classifying ingress Layer 3, VLAN, and SVI QoS packets mac-vqos or ns-mac-vqos: For classifying non-IP packets **Note** For traffic that needs to be classified on 40G ports on Cisco Nexus 9300 Series switches, you must carve the qos regions and the corresponding ns-*qos regions.
RACL	egr-racl: For egress IPv4 and IPv6 RACLs e-racl: For egress IPv4 RACLs e-ipv6-racl: For egress IPv6 RACLs ing-racl: For ingress IPv4 and IPv6 RACLs racl: For IPv4 RACLs racl-lite: For IPv4 RACLs racl-udf: For UDFs on IPv4 RACLs ipv6-racl: For IPv6 RACLs

Feature Name	Region Name
Layer 3 QoS (QoS classification policy applied on Layer 3 ports or port channels)	l3qos, l3qos-lite, or ns-l3qos: For classifying IPv4 packets ipv6-l3qos or ns-ipv6-l3qos: For classifying IPv6 packets **Note** For traffic that needs to be classified on 40G ports on Cisco Nexus 9300 Series switches, you must carve qos regions and the corresponding ns-*qos regions.
VLAN source or VLAN filter SPAN (for Cisco Nexus 9500 or 9300 Series switches) Rx SPAN on 40G ports (for Cisco Nexus 9300 Series switches only)	span
SPAN filters	ifacl: For filtering IPv4 traffic on Layer 2 (switch port) source interfaces. ifacl-udf: For UDFs on IPv4 port ACLs ipv6-ifacl: For filtering IPv6 traffic on Layer 2 (switch port) source interfaces. mac-ifacl: For filtering Layer 2 traffic on Layer 2 (switch port) source interfaces. racl-udf: For UDFs on IPv4 RACLs vacl: For filtering IPv4 traffic on VLAN sources. ipv6-vacl: For filtering IPv6 traffic on VLAN sources. mac-vacl: For filtering Layer 2 traffic on VLAN sources. racl: For filtering IPv4 traffic on Layer 3 interfaces. ipv6-racl: For filtering IPv6 traffic on Layer 3 interfaces. ing-l2-span-filter: For filtering ingress Layer 2 SPAN traffic ing-l3-span-filter: For filtering ingress Layer 3 and VLAN SPAN traffic
SVI counters **Note** This region enables the packet counters for Layer 3 SVI interfaces.	svi

Feature Name	Region Name
BFD, DHCP relay, or DHCPv6 relay	redirect **Note** BFD uses the ing-sup region while DHCPv4 relay, DHCPv4 snooping, and DHCPv4 client use the ing-redirect region.
CoPP	copp **Note** The region size cannot be 0.
System-managed ACLs	system **Note** The region size cannot be changed.
vPC convergence **Note** This region boosts the convergence times when a vPC link goes down and traffic needs to be redirected to the peer link.	vpc-convergence **Note** Setting this region size to 0 might affect the convergence times of vPC link failures.
Fabric extender (FEX)	fex-ifacl, fex-ipv6-ifacl, fex-ipv6-qos, fex-mac-ifacl, fex-mac-qos, fex-qos, fex-qos-lite
Dynamic ARP inspection (DAI)	arp-ether
IP source guard (IPSG)	ipsg
Multicast PIM Bidir	mcast_bidir
Static MPLS	mpls
Network address translation (NAT)	nat
NetFlow	ing-netflow
OpenFlow	openflow
sFlow	sflow
Supervisor modules	egr-sup: Egress supervisor ing-sup: Ingress supervisor

Related Topics

Configuring ACL TCAM Region Sizes, on page 33
Configuring TCAM Carving, on page 43

Maximum Label Sizes Supported for ACL Types

Cisco NX-OS switches support the following label sizes for the corresponding ACL types:

Table 3: ACL Types and Maximum Label Sizes

ACL Types	Direction	Label	Label Type
RACL/PBR/VACL/ L3-VLAN QoS/L3-VLAN SPAN ACL	Ingress	62	BD
PACL/L2 QoS/L2 SPAN ACL	Ingress	62[1]	IF
RACL/VACL/L3-VLAN QoS	Egress	254	BD
L2 QoS	Egress	31	IF

[1] The label size can be increased to 62 when you enter the **hardware access-list tcam label ing-ifacl 6** command and reload the switch.

Beginning with Cisco NX-OS Release 9.3(6), the **hardware access-list tcam label ing-ifacl 6** command is introduced and is applicable only for Cisco Nexus 9300-FX platform switches.

Beginning with Cisco NX-OS Release 10.1(2), the **hardware access-list tcam label ing-ifacl 6** command is also supported on Cisco Nexus 9300-FX2 platform switches.

Prerequisites for IP ACLs

IP ACLs have the following prerequisites:

- You must be familiar with IP addressing and protocols to configure IP ACLs.
- You must be familiar with the interface types that you want to configure with ACLs.

Guidelines and Limitations for IP ACLs

IP ACLs have the following configuration guidelines and limitations:

Note For more information about the Cisco Nexus 9000 series platform switches that support various features spanning from release 7.0(3)I7(1) to the current release, refer to Nexus Switch Platform Support Matrix.

- Beginning with Cisco NX-OS Release 10.2(1)F, Egress PACL is supported on the Cisco Nexus 9364D-GX2A, and 9332D-GX2B switches.
- If you configure egress PACL and egress VACL on the same interface, only egress VACL is enabled.
- We recommend that you perform ACL configuration using the Session Manager. This feature allows you to verify ACL configuration and confirm that the resources that are required by the configuration are available before committing them to the running configuration. This recommendation is especially

useful for ACLs that include more than 1000 rules. For more information about Session Manager, see the *Cisco Nexus 9000 Series NX-OS System Management Configuration Guide*.

- Configuring a IPv4 PACL in the range of 12K to 64K is supported on Cisco Nexus 9500 Series switches with -RX line cards.
- Duplicate ACL entries with different sequence numbers are allowed in the configuration. However, these duplicate entries are not programmed in the hardware access-list.
- Only 62 unique ACLs can be configured. Each ACL takes one label. If the same ACL is configured on multiple interfaces, the same label is shared. If each ACL has unique entries, the ACL labels are not shared, and the label limit is 62. This is not applicable to Cisco Nexus 9500 Series switches.
- Usually, ACL processing for IP packets occurs on the I/O modules, which use hardware that accelerates ACL processing. In some circumstances, processing occurs on the supervisor module, which can result in slower ACL processing, especially during processing that involves an ACL with many rules. Management interface traffic is always processed on the supervisor module. If IP packets in any of the following categories are exiting a Layer 3 interface, they are sent to the supervisor module for processing:
  - Packets that fail the Layer 3 maximum transmission unit check and therefore require fragmenting.
  - IPv4 packets that have IP options (other IP packet header fields following the destination address field).
  - IPv6 packets that have extended IPv6 header fields.

  Rate limiters prevent redirected packets from overwhelming the supervisor module.
- When you apply an ACL that uses time ranges, the device updates the ACL entries whenever a time range that is referenced in an ACL entry starts or ends. Updates that are initiated by time ranges occur on a best-effort priority. If the device is especially busy when a time range causes an update, the device may delay the update by up to a few seconds.
- To apply an IP ACL to a VLAN interface, you must have enabled VLAN interfaces globally. For more information about VLAN interfaces, see the *Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide*.
- The VTY ACL feature restricts all traffic for all VTY lines. You cannot specify different traffic restrictions for different VTY lines. Any router ACL can be configured as a VTY ACL.
- An egress VTY ACL (an IP ACL applied to the VTY line in the outbound direction) prevents the switch from copying files using a file transfer protocol (TFTP, FTP, SCP, SFTP, etc.) unless the file transfer protocol is explicitly permitted within the egress VTY ACL.
- When you apply an undefined ACL to an interface, the system treats the ACL as empty and permits all traffic.
- IP tunnels do not support ACLs or QoS policies.
- The following guidelines apply to ACLs for VXLANs:
  - Ingress port ACLs applied on a Layer 2 port for traffic in the access to a network direction (Layer 2 to Layer 3 encapsulation path) are supported on the inner payload.
  - We recommend using port ACLs on the access side to filter out traffic entering the overlay network.
  - Ingress router ACLs applied on an uplink Layer 3 interface matching on the inner or outer payload in the network to access direction (Layer 3 to Layer 2 decapsulation path) are not supported.

    - Egress router ACLs applied on an uplink Layer 3 interface matching on the inner or outer payload in the access to a network direction (encapsulation path) are not supported.

- Cisco Nexus 9300 and 9500 Series switches, and Cisco Nexus 9200 and 9300-EX Series switches have the following limitations for ACL options that can be used on VXLAN traffic:
    - Does not support egress port ACLs applied on a Layer 2 port for traffic in the network to access direction (decapsulation path).
    - Supports ingress VACLs applied on a VLAN for traffic in the access to a network direction (encapsulation path).
    - Supports egress VACLs applied on a VLAN for traffic in the network to access direction (decapsulation path).
    - Supports ingress RACLs applied on a tenant or server facing SVI for traffic in the access to network direction (encapsulation path).
    - Supports egress RACLs applied on a tenant or server facing SVI for traffic in the network to access direction (decapsulation path).
- IPv6 ACL logging is not supported for egress PACL.
- IPv4 ACL logging in the egress direction is not supported.
- ACL logging for VACLs is not supported.
- ACL logging applies to port ACLs configured by the **ip port access-group** command and to router ACLs configured by the **ip access-group** command only.
- The total number of IPv4 ACL flows is limited to a user-defined maximum value to prevent DoS attacks. If this limit is reached, no new logs are created until an existing flow finishes.
- The number of syslog entries that are generated by IPv4 ACL logging is limited by the configured logging level of the ACL logging process. If the number of syslog entries exceeds this limit, the logging facility might drop some logging messages. Therefore, IPv4 ACL logging should not be used as a billing tool or as an accurate source of the number of matches to an ACL.
- Egress router ACLs are not supported on Cisco Nexus 9300 Series switch uplink ports.
- For Network Forwarding Engine (NFE)-enabled switches, ingress RACLs matching the outer header of the tunnel interface are not supported.
- If the same QoS policy and ACL are applied to multiple interfaces, the label is shared only when the QoS policy is applied with the no-stats option.
- The switch hardware does not support range checks (Layer 4 operations) in the egress TCAM. Therefore, ACL and QoS policies with a Layer 4 operations-based classification must be expanded to multiple entries in the egress TCAM.

  The switch hardware supports only up to 16 Layer 4 operands. Make sure to consider this limitation for egress TCAM space planning. For more information see the Logical Operators and Logical Operation Units, on page 7 section.
- For Cisco Nexus X96136YC-R, X9636C-RX, X9636C-RX, and X9636Q-R line cards, run the **hardware profile acl-eg-ext module all** command before applying **eg-racl-v6** configuration on a SVI or port object on an EoR switch.

- TCAM resources are shared in the following scenarios:
  - When a routed ACL is applied to multiple switched virtual interfaces (SVIs) in the ingress direction.
  - When a routed ACL is applied to multiple layer 2 interfaces in the ingress or egress direction.
- TCAM resources are not shared in the following scenarios:
  - VACL (VLAN ACL) is applied to multiple VLANs.
  - Routed ACL is applied to multiple SVIs in the egress direction.
- Access-lists based on HTTP methods are not supported on the Cisco Nexus 9200, 9300-EX, 9300-FX, 9300-FX2, 9300-FXP, and 9300-GX platform switches and the 9500 switches with the X9700-EX, and X9700-FX line cards. For all these switches, you must use UDF-based ACLs.
- HTTP methods are not supported on FEX ports.
- The following guidelines and limitations apply to Cisco Nexus 9200 and 9300-EX Series switches:
  - Egress MAC ACLs are not supported.
  - Egress RACLs are not supported on an interface if the packet matches the outer header of the tunnel interface on the device where the tunnel is originating the traffic.
  - Ingress RACLs matching the outer header of the tunnel interface are not supported.
  - IP length-based matches are not supported.
  - All ACL-based features cannot be enabled at the same time.
  - 16 Layer 4 operations are supported.
  - Layer 4 operations are not supported on egress TCAM regions.
  - The MAC compression table size is 4096 + 512 overflow TCAM.
  - An overlap of MAC addresses and MAC masks is rejected.
  - The ACL log rate limiter does not have any hardware counters for transmitted or dropped packets.
  - The ACL log rate limiter is implemented at the per-TCAM entry level (instead of using aggregated rate limiting), and the default is 1 pps.
  - The Network Address Translation (NAT) exception counters are zero.
  - Only PACL redirects are supported for TAP aggregation. VACL redirects are not supported.
  - Only three of the following four features can be supported at a time: DHCPv4 snooping or relay, DHCPv6 relay, ARP snooping, VXLAN. The first three configured features take effect, but the fourth one will fail because all three bridge domain label bits are already in use.
  - RACLs cannot match on packets with multicast MAC destination addresses.
- The following guidelines and limitations apply to Cisco Nexus 9364C-GX, Cisco Nexus 9316D-GX, and Cisco Nexus 93600CD-GX series switches:
  - The MAC compression table size is 4096 + 512 overflow TCAM.
  - An overlap of MAC addresses and MAC masks is rejected.

- Cisco Nexus 9504 and Cisco Nexus 9508 switches with -R line cards do not support the following TCAM:
  - All FEX related TCAM
  - All xxx-lite related TCAM region
  - Ranger related TCAM
  - All FCoE related TCAM
- TCAM carving configuration of the ing-netflow region can be performed on -FX line cards. -EX line cards have a default ing-netflow region TCAM carving of 1024 and cannot be configured otherwise. For ports on the -EX and -FX line cards, the suggested maximum for the ing-netflow region is 1024.
- On the Cisco Nexus 9200 and 9300-EX platform switches, router ACL with the ACL log option will not take into effect as the sup-redirect ACLs have higher priority for the traffic that is destined to SUP.
- On the Cisco Nexus 9300-GX platform switches, dot1q VLAN with ACL redirect supports only the VLAN IDs from 1 to 511.

  If PACL redirect or TapAgg is configured, the **switchport access vlan** *vlan-id* command supports only the vlan IDs from 1 to 511.
- For traffic destined to the FHRP VIP and ingressing on FHRP standby which matches an ACL log enabled ACE designed to permit the traffic, the Cisco Nexus 9000 Series switch drops this packet.
- For Cisco Nexus 3172TQ, 3172TQ-XL, 36180YC-R, and 3636C-R switches, when there is a SVI and subinterface matching the same VLAN tag, the traffic that gets routed out through a subinterface gets dropped if the access-list is configured on that SVI. This is due to an ASIC limitation and egress router ACL on L3 subinterfaces is not supported due to this limitation.
- Cisco Nexus 9364D-GX2A, and 9332D-GX2B switches do not support the following on egress router ACL:
  - UDF to support ICMP Type Match.
  - ACL log-on egress
  - Egress IPv4 router ACL with additional filter option tcp/udp ports with lt/gt
  - Egress IPv4 router ACL with additional filter option tcp/udp ports with neq
  - Egress IPv4 router ACL with extra filter option tcp/udp ports with range
  - Egress IPv4 router ACL with a flag
  - Egress router ACL on an external TCAM
  - Egress PACL support
  - Statistics support
  - Label sharing
- Cisco Nexus 9500 platform switches with -R and -RX line cards have the following guidelines:
  - Atomic ACL update is supported for all the ingress ACL features except for the Multihop BFD and CoPP features.
  - Atomic ACL update is not supported for the egress ACL features.

- Label sharing is supported only for the same policy on different interfaces within the same ASIC.
- In Cisco NX-OS Release 9.2(3), ACL statistics are supported for the following:
  - PACL - IPv4 (including system ACL for both, internal, and external TCAM)
  - Router ACL - IPv4 (internal TCAM for both, ingress RACL-IPv4 and egress RACL-IPv4)
  - Only 2K counters are supported in the egress.
- ACL statistics are not supported for the following:
  - BFD
  - DHCP - IPv4 and IPv6
  - PACL-MAC
  - PACL- IPv6
  - PBR - IPv4 and IPv6
  - RACL-IPv6
  - RACL-IPv4 when using an external TCAM

- ACL label sharing works on Cisco Nexus 9300-EX/FX/FX2/FX3/GX/GX2 and 9300C platform switches with following limitations:
  - ACL statistics are disabled by default. However, statistics are enabled by default only for QoS policies.
  - ACL target (port / VLAN / SVI etc.,) must be on the same slice, and port.
  - Additionally, label space is shared with following features:
    - Ingress RACL, PBR and Ingress L3 QoS
    - Ingress PACL, Ingress L2 QoS
    - Egress RACL, Egress QoS

Note For label sharing to work, ensure that the same set of features are supported on interfaces.

- When you enable the counters for the ACL TCAM entries using the hardware profile acl-stats module *xx* command, the input discard field in the show interface is always zero. This limitation is applicable only to the Cisco Nexus 9500 platform switches with -R and -RX line cards.
- Cisco Nexus 9500 platform switches with -R and -RX line cards do not support the following:
  - Egress atomic updates
  - Egress router ACL on external TCAM
  - Egress router ACL with UDF

- Router ACL v6 counters for both egress and ingress
- Egress and ingress router ACL IPv6 with l4 ops
- Egress router ACL on subinterface
- Egress and ingress router ACL with IPv6 ICMP Type and Code
- IPv6 ingress router ACL with tcp-flag
- IPv4 router ACL with extra option

- In Cisco NX-OS Release 9.3(3), egress IPv4 RACLs support the following on Cisco Nexus 9504 and 9508 switches with -R and -RX line cards:
  - TCP flags
  - ICMP Type and Code
  - ACL logs

- IPv6 Egress ACL support the following on Cisco Nexus 9504 and 9508 switches with -R and -RX line cards:
  - Layer 4 Protocol
  - TCP flags
  - Fragment
  - ACL logs for IPv4
  - IPv6 header fields

  The following limitations are applicable for the IPv6 egress ACL:
  - Port groups and Layer 4 Operations are not supported. The ranges expand to multiple ACE entries for eg-racl-ipv6.
  - Address group defined host is not supported.
  - Counters are not supported.
  - Egress IPv6 RACL is not supported on sub-interfaces and external TCAM.
  - Atomic updates are not supported.
  - VXLAN is not supported when acl-eg-ext is enabled.

- PACL redirects are supported on Cisco Nexus 9300-GX switches. The following limitations are applicable:
  - To support PACL redirects, you must run the **mode tap-agg** command on the ingress tap interface.
  - To support the MPLS strip feature, the **mpls strip** and the **hardware acl tap-agg** commands must be configured and the switch reloaded.
  - For double tag VLAN, the range of the second VLAN is 2-510.
  - MPLS strip with dot1q VLAN is not supported.

  - The redirect port carries the tag if the incoming packet is tagged, even when the redirect port is configured as an access port.
  - TapAgg redirect is not supported for deny ACE.

- In Cisco NX-OS Release 10.1(2), PACL redirect feature is not supported in mixed mode on Cisco Nexus X9736C-FX, X9788TC-FX, and X97160YC-EX line cards.
- Egress ACL does not support traffic that is destined to the IP address of the second VLAN in inter-VLAN routing flow.
- In Cisco Nexus 9300-EX/FX/FX2/FX3/GX platform switches and 93180YC-FX switches, RACLs cannot match on packets with multicast MAC destination addresses on Layer-3 interfaces. Use the **ignore routable** command when you configure the ACL to remove the routable qualifier. However, when you add ignore-routable to a RACL and apply on SVI, RACL will match with the bridged packets too.
- The Get operation provides incomplete data/no sequence number when wildcard bits are in A.B.C.D format. This is a known behavior. The Open Config model does not have srcPrefixMask/dstPrefixMask. Also, no meaningful value can be returned for prefix length because it is not possible to convert the mask to prefix length for non-contiguous mask.
- The ing-sup region occupies a minimum size of 512 entries, and the egr-sup region occupies a minimum size of 256 entries. These regions cannot be configured to lesser values. Any region size can be carved with a value only in multiples of 256 entries (with the exception of the span region, which can be carved only in multiples of 512 entries).
- Beginning with Cisco NX-OS Release 9.3(9), the Layer 3 subinterface egress router ACL feature is supported on Cisco Nexus 9300-EX, 9300-FX, and 9300-FX2 platform switches.
- Beginning with Cisco NX-OS Release 10.2(3)F, the Layer 3 subinterface egress router ACL feature is supported on Cisco Nexus 9300 Series platform switches.
- For egress RACL V6 region, you need to configure **hw profile mdb-balanced-exem**.
- From Cisco NX-OS Release 10.2(2)F, the egress PACL feature is supported on egress router ACL on Cisco Nexus 9300-GX platform switches and 93108TC-FX3P, and 93180YC-FX3 switches.
- Beginning with Cisco NX-OS Release 10.2(3)F, the egress filtering on subinterfaces feature supports Layer 3 subinterface egress router ACL on Cisco Nexus 9300-FX/FX2/FX3/GX/GX2 platform switches.
- Beginning with Cisco NX-OS Release 10.2(3)F, the increase ACL LOU threshold feature supports configurable LOU threshold limit for ACL configuration on Cisco Nexus 9500-R platform switches.
- Deny ACE in MAC ACL or PACL (Port ACL) with redirect option is not supported on Cisco Nexus 9000 Series switches.

Default Settings for IP ACLs

This table lists the default settings for IP ACL parameters.

Table 4: Default IP ACL Parameters

Parameters	Default
IP ACLs	No IP ACLs exist by default
IP ACL entries	1024
ACL rules	Implicit rules apply to all ACLs
Object groups	No object groups exist by default
Time ranges	No time ranges exist by default

Related Topics

Implicit Rules for IP and MAC ACLs, on page 4

Configuring IP ACLs

Creating an IP ACL

You can create an IPv4 ACL or IPv6 ACL on the device and add rules to it.

Before you begin

We recommend that you perform the ACL configuration using the Session Manager. This feature allows you to verify the ACL configuration and confirm that the resources that are required by the configuration are available before committing them to the running configuration. This feature is especially useful for ACLs that include more than about 1000 rules.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	Enter one of the following commands: • **ip access-list** *name* • **ipv6 access-list** *name*	Creates the IP ACL and enters IP ACL configuration mode. The *name* argument can be up to 64 characters. **Notice**

	Command or Action	Purpose

	Example:
	`switch(config)# ip access-list acl-01`
	`switch(config-acl)#`
		The names **dynamic**, **expanded**, and **summary** are reserved for system-defined access lists. Do not use these names for user-defined ACLs, as this can cause conflicts when displaying or verifying your configuration.

Step 3	(Optional) **fragments** {**permit-all** | **deny-all**} **Example:** `switch(config-acl)# fragments permit-all`	Optimizes fragment handling for noninitial fragments. When a device applies to traffic an ACL that contains the **fragments** command, the **fragments** command only matches noninitial fragments that do not match any explicit **permit** or **deny** commands in the ACL.
Step 4	[*sequence-number*] {**permit** | **deny**} *protocol* {*source-ip-prefix* | *source-ip-mask*} {*destination-ip-prefix* | *destination-ip-mask*} **Example:** `switch(config-acl)# 10 permit ipv6 1::1 2::2 3::3 4::4`	Creates a rule in the IP ACL. You can create many rules. The *sequence-number* argument can be a whole number between 1 and 4294967295. The **permit** and **deny** commands support many ways of identifying traffic. For IPv4 and IPv6 access lists, you can specify a source and destination IPv4 or IPv6 prefix, which matches only on the first contiguous bits, or you can specify a source and destination IPv4 or IPv6 wildcard mask, which matches on any bit in the address. IPv6 wildcard masks are supported for Cisco Nexus 9200, 9300-EX, and 9300-FX/FX2/FXP switches and the Cisco Nexus 9364C switch.
Step 5	(Optional) **statistics per-entry** **Example:** `switch(config-acl)# statistics per-entry`	Specifies that the device maintains global statistics for packets that match the rules in the ACL. **Note** Beginning Cisco NX-OS Release 9.2(3), ACL statistics is supported on Cisco Nexus 9500 platform switches with -R line cards. This is a mandatory step if you are using the Cisco Nexus 9500 platform switches.
Step 6	**hardware profile acl-stats module** *xx* **Example:** `switch(config-acl)# hardware profile acl-stats module 10`	Enables counters for the ACL TCAM entries on both, the internal and external TCAM. **Note** This command is applicable only for Cisco Nexus 9500 platform switches with -R and -RX line cards and Cisco Nexus 3636C-R and 36180YC-R switches. VLAN and SVI

	Command or Action	Purpose
		statistics are lost when you enable the counters.
Step 7	**reload module** *xx* **Example:** `switch(config)# reload module 10`	Reloads the switch. **Note** For the Cisco Nexus 9500 platform switches, this command is optional and only those module (s) where the**hardware profile ac-stats** is applied must be reloaded.
Step 8	**ignore routeable** **Example:** `switch(config)# ignore routeable`	Enables the filtering of multicast traffic on Cisco Nexus 9300-EX and 9300-FX platform switches.
Step 9	(Optional) Enter one of the following commands: • **show ip access-lists** *name* • **show ipv6 access-lists** *name* **Example:** `switch(config-acl)# show ip access-lists acl-01`	Displays the IP ACL configuration.
Step 10	(Optional) **copy running-config startup-config** **Example:** `switch(config-acl)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Changing an IP ACL

You can add and remove rules in an existing IPv4 or IPv6 ACL, but you cannot change existing rules. Instead, to change a rule, you can remove it and recreate it with the desired changes.

If you need to add more rules between existing rules than the current sequence numbering allows, you can use the **resequence** command to reassign sequence numbers.

Before you begin

We recommend that you perform ACL configuration using the Session Manager. This feature allows you to verify ACL configuration and confirm that the resources required by the configuration are available prior to committing them to the running configuration. This feature is especially useful for ACLs that include more than about 1000 rules.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	Enter one of the following commands:
	• **ip access-list** *name*
	• **ipv6 access-list** *name*
	Example:
	`switch(config)# ip access-list acl-01`
	`switch(config-acl)#`
		Enters IP ACL configuration mode for the ACL that you specify by name.

Step 3	(Optional) [*sequence-number*] {**permit** | **deny**} *protocol source destination* **Example:** `switch(config-acl)# 100 permit ip 192.168.2.0/24 any`	Creates a rule in the IP ACL. Using a sequence number allows you to specify a position for the rule in the ACL. Without a sequence number, the rule is added to the end of the rules. The *sequence-number* argument can be a whole number between 1 and 4294967295. The **permit** and **deny** commands support many ways of identifying traffic.
Step 4	(Optional) [**no**] **fragments** {**permit-all** | **deny-all**} **Example:** `switch(config-acl)# fragments permit-all`	Optimizes fragment handling for noninitial fragments. When a device applies to traffic an ACL that contains the **fragments** command, the **fragments** command only matches noninitial fragments that do not match any explicit **permit** or **deny** commands in the ACL. The **no** option removes fragment-handling optimization.
Step 5	(Optional) **no** {*sequence-number* | {**permit** | **deny**} *protocol source destination*} **Example:** `switch(config-acl)# no 80`	Removes the rule that you specified from the IP ACL. The **permit** and **deny** commands support many ways of identifying traffic.
Step 6	(Optional) [**no**] **statistics per-entry** **Example:** `switch(config-acl)# statistics per-entry`	Specifies that the device maintains global statistics for packets that match the rules in the ACL. The **no** option stops the device from maintaining global statistics for the ACL.
Step 7	(Optional) Enter one of the following commands: • **show ip access-lists** *name*	Displays the IP ACL configuration.

	Command or Action	Purpose
	• **show ipv6 access-lists** *name* **Example:** `switch(config-acl)# show ip access-lists acl-01`	
Step 8	(Optional) **copy running-config startup-config** **Example:** `switch(config-acl)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Related Topics

Changing Sequence Numbers in an IP ACL, on page 31

Creating a VTY ACL

You can configure a VTY ACL to control access to all IPv4 or IPv6 traffic over all VTY lines in the ingress or egress direction.

Before you begin

Set identical restrictions on all the virtual terminal lines because a user can connect to any of them.

We recommend that you perform ACL configuration using the Session Manager. This feature allows you to verify ACL configuration and confirm that the resources required by the configuration are available prior to committing them to the running configuration, which is especially useful for ACLs that include more than about 1000 rules.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	{**ip** | **ipv6**} **access-list** *name* **Example:** `switch(config)# ip access-list vtyacl`	Creates an ACL and enters IP access list configuration mode for that ACL. The maximum length for the *name* argument is 64 characters.
Step 3	{**permit** | **deny**} *protocol source destination* [**log**] [**time-range** *time*] **Example:** `switch(config-ip-acl)# permit tcp any any`	Creates an ACL rule that permits TCP traffic from and to the specified sources.

	Command or Action	Purpose

Step 4	**exit**
	Example:
	`switch(config-ip-acl)# exit`
	`switch(config)#`
		Exits IP access list configuration mode.

Step 5	**line vty**
	Example:
	`switch(config)# line vty`
	`switch(config-line)#`
		Specifies the virtual terminal and enters line configuration mode.

Step 6	{**ip** | **ipv6**} **access-class** *name* {**in** | **out**} **Example:** `switch(config-line)# ip access-class vtyacl out`	Restricts incoming or outgoing connections to and from all VTY lines using the specified ACL. The maximum length for the *name* argument is 64 characters.
Step 7	(Optional) **show** {**ip** | **ipv6**} **access-lists** **Example:** `switch# show ip access-lists`	Displays the configured ACLs, including any VTY ACLs.
Step 8	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Changing Sequence Numbers in an IP ACL

You can change all the sequence numbers assigned to the rules in an IP ACL.

Before you begin

We recommend that you perform ACL configuration using the Session Manager. This feature allows you to verify ACL configuration and confirm that the resources required by the configuration are available prior to committing them to the running configuration. This feature is especially useful for ACLs that include more than about 1000 rules.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**resequence** {**ip** | **ipv6**} **access-list** *name starting-sequence-number increment*	Assigns sequence numbers to the rules contained in the ACL, where the first rule

	Command or Action	Purpose
	Example: `switch(config)# resequence access-list ip acl-01 100 10`	receives the starting sequence number that you specify. Each subsequent rule receives a number larger than the preceding rule. The difference in numbers is determined by the increment that you specify. The *starting-sequence-number* argument and the *increment* argument can be a whole number between 1 and 4294967295.
Step 3	(Optional) **show ip access-lists** *name* **Example:** `switch(config)# show ip access-lists acl-01`	Displays the IP ACL configuration.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Removing an IP ACL

You can remove an IP ACL from the device.

Before you begin

Ensure that you know whether the ACL is applied to an interface. The device allows you to remove ACLs that are currently applied. Removing an ACL does not affect the configuration of interfaces where you have applied the ACL. Instead, the device considers the removed ACL to be empty. Use the **show ip access-lists** command or the **show ipv6 access-lists** command with the summary keyword to find the interfaces that an IP ACL is configured on.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	Enter one of the following commands: • **no ip access-list** *name* • **no ipv6 access-list** *name* **Example:** `switch(config)# no ip access-list acl-01`	Removes the IP ACL that you specified by name from the running configuration.

	Command or Action	Purpose
Step 3	(Optional) Enter one of the following commands: • **show ip access-lists** *name* **summary** • **show ipv6 access-lists** *name* **summary** **Example:** `switch(config)# show ip access-lists acl-01 summary`	Displays the IP ACL configuration. If the ACL remains applied to an interface, the command lists the interfaces.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Configuring ACL TCAM Region Sizes

You can change the size of the ACL ternary content addressable memory (TCAM) regions in the hardware. After TCAM carving, for the TCAM to qualify, you must save the configuration and reload the switch. If the switch has a faulty module, saving the configuration will take a longer time.

You can use this procedure for all Cisco Nexus 9200, 9300, and 9500 Series switches and the Cisco Nexus 3164Q, 31128PQ, 3232C, and 3264Q switches, except for NFE2-enabled devices (such as the X9432C-S 100G line card and the C9508-FM-S fabric module), which must use TCAM templates to configure ACL TCAM region sizes. For more information on using TCAM templates, see "Using Templates to Configure ACL TCAM Region Sizes."

Note

- Once you apply a template (using Using Templates to Configure ACL TCAM Region Sizes, on page 42), the **hardware access-list tcam region** command in this section will not work. You must uncommit the template in order to use the command.
- The **hardware access-list tcam region** command for the Multicast PIM Bidir feature is applicable only to the Broadcom-based Cisco Nexus 9000 Series switches.
- For information on configuring QoS TCAM carving, see the *Cisco Nexus 9000 Series NX-OS Quality of Service Configuration Guide*.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal switch(config)#`	Enters global configuration mode.

	Command or Action	Purpose
Step 2	[**no**] **hardware access-list tcam region** *region tcam-size* **Example:** `switch(config)# hardware access-list tcam region mpls 256`	Changes the ACL TCAM region size. These are the available regions: • **n9k-arp-acl**—Configures the rate limit for arp packets entering an interface on their way to the CPU. You will have to set this rate limit per interface to ensure that arp packets conform to the configured rate. • **arp-ether**—Configures the size of the ARP/Layer 2 Ethertype TCAM region. • **copp**—Configures the size of the CoPP TCAM region. • **e-flow**—Configures the size of the egress flow counters TCAM region. • **egr-copp**—Configures the size of the egress CoPP TCAM region. • **egr-racl**—Configures the size of the egress IPv4 or IPv6 router ACL (RACL) TCAM region. • **egr-sup**—Configures the size of the egress supervisor TCAM region. • **e-ipv6-qos**—Configures the size of the IPv6 egress QoS TCAM region. • **e-ipv6-racl**—Configures the size of the IPv6 egress router ACL (ERACL) TCAM region. • **e-mac-qos**—Configures the size of the egress MAC QoS TCAM region. • **e-qos**—Configures the size of the IPv4 egress QoS TCAM region. • **e-qos-lite**—Configures the size of the IPv4 egress QoS lite TCAM region. • **e-racl**—Configures the size of the IPv4 egress router ACL (ERACL) TCAM region. • **fex-ifacl**—Configures the size of the FEX IPv4 port ACL TCAM region. • **fex-ipv6-ifacl**—Configures the size of the FEX IPv6 port ACL TCAM region. • **fex-ipv6-qos**—Configures the size of the FEX IPv6 port QoS TCAM region.

	Command or Action	Purpose
		• **fex-mac-ifacl**—Configures the size of the FEX MAC port ACL TCAM region. • **fex-mac-qos**—Configures the size of the FEX MAC port QoS TCAM region. • **fex-qos**—Configures the size of the FEX IPv4 port QoS TCAM region. • **fex-qos-lite**—Configures the size of the FEX IPv4 port QoS lite TCAM region. • **fhs**—Configures the size of the fhs TCAM region. You can configure TCAM for the fhs region on the Cisco Nexus 9300 and 9500 Series switches. • **flow**—Configures the size of the ingress flow counters TCAM region. • **ifacl**—Configures the size of the IPv4 port ACL TCAM region. • **ifacl-udf**—Configures the size of the IPv4 port ACL user-defined field (UDF) TCAM region. • **ing-ifacl**—Configures the size of the ingress IPv4, IPv6, or MAC port ACL TCAM region. **Note** You can attach user-defined fields (UDFs) to the ing-ifacl TCAM region to configure UDF-based IPv4 or IPv6 port ACLs. UDF-based IPv6 port ACLs. For more information and configuration instructions, see Configuring UDF-Based Port ACLs, on page 50. • **ing-l2qos**—Configures the size of the ingress Layer 2 QoS TCAM region. • **ing-l2-span-filter**—Configures the size of the ingress Layer 2 SPAN filter TCAM region. • **ing-l3-span-filter**—Configures the size of the ingress Layer 3 and VLAN SPAN filter TCAM region. • **ing-l3-vlan-qos**—Configures the size of the ingress Layer 3, VLAN, and SVI QoS TCAM region.

	Command or Action	Purpose
		• **ing-netflow**—Configures the size of the NetFlow TCAM region. • **ing-racl**—Configures the size of the IPv4 or IPv6 ingress router ACL (RACL) TCAM region. • **ing-redirect**—Configures the size of the redirect TCAM region for DHCPv4 relay, DHCPv4 snooping, and DHCPv4 client. • **ing-sup**—Configures the size of the ingress supervisor TCAM region. • **ipsg**—Configures the size of the IP source guard SMAC-IP binding TCAM region. • **ipv6-ifacl**—Configures the size of the IPv6 port ACL TCAM region. • **ipv6-l3qos**—Configures the size of the IPv6 Layer 3 QoS TCAM region. • **ipv6-qos**—Configures the size of the IPv6 port QoS TCAM region. • **ipv6-racl**—Configures the size of the IPv6 RACL TCAM region. • **ipv6-vacl**—Configures the size of the IPv6 VACL TCAM region. • **ipv6-vqos**—Configures the size of the IPv6 VLAN QoS TCAM region. • **l3qos**—Configures the size of the IPv4 Layer 3 QoS TCAM region. • **l3qos-lite**—Configures the size of the IPv4 Layer 3 QoS lite TCAM region. • **mac-ifacl**—Configures the size of the MAC port ACL TCAM region. • **mac-l3qos**—Configures the size of the MAC Layer 3 QoS TCAM region. • **mac-qos**—Configures the size of the MAC port QoS TCAM region. • **mac-vacl**—Configures the size of the MAC VACL TCAM region. • **mac-vqos**—Configures the size of the MAC VLAN QoS TCAM region.

	Command or Action	Purpose
		• **mcast_bidir**—Configures the size of the multicast PIM Bidir TCAM region. • **mpls**—Configures the size of the static MPLS TCAM region. • **nat**—Configures the size of the network address translation (NAT) TCAM region. • **ns-ipv6-l3qos**—Configures the size of the IPv6 Layer 3 QoS TCAM region for the X9536PQ, X9564PX, and X9564TX line cards and the M12PQ generic expansion module (GEM). • **ns-ipv6-qos**—Configures the size of the IPv6 port QoS TCAM region for the X9536PQ, X9564PX, and X9564TX line cards and the M12PQ generic expansion module (GEM). • **ns-ipv6-vqos**—Configures the size of the IPv6 VLAN QoS TCAM region for the X9536PQ, X9564PX, and X9564TX line cards and the M12PQ generic expansion module (GEM). • **ns-l3qos**—Configures the size of the IPv4 Layer 3 QoS TCAM region for the X9536PQ, X9564PX, and X9564TX line cards and the M12PQ generic expansion module (GEM). • **ns-mac-l3qos**—Configures the size of the MAC Layer 3 QoS TCAM region for the X9536PQ, X9564PX, and X9564TX line cards and the M12PQ generic expansion module (GEM). • **ns-mac-qos**—Configures the size of the MAC port QoS TCAM region for the X9536PQ, X9564PX, and X9564TX line cards and the M12PQ generic expansion module (GEM). • **ns-mac-vqos**—Configures the size of the MAC VLAN QoS TCAM region for the X9536PQ, X9564PX, and X9564TX line cards and the M12PQ generic expansion module (GEM). • **ns-qos**—Configures the size of the IPv4 port QoS TCAM region for the X9536PQ,

	Command or Action	Purpose
		X9564PX, and X9564TX line cards and the M12PQ generic expansion module (GEM). • **ns-vqos**—Configures the size of the IPv4 VLAN QoS TCAM region for the X9536PQ, X9564PX, and X9564TX line cards and the M12PQ generic expansion module (GEM). • **openflow**—Configures the size of the OpenFlow TCAM region. • **qos**—Configures the size of the IPv4 port QoS TCAM region. • **qos-lite**—Configures the size of the IPv4 port QoS lite TCAM region. • **racl**—Configures the size of the IPv4 router ACL (RACL) TCAM region. • **racl-lite**—Configures the size of the IPv4 router ACL (RACL) lite TCAM region. • **racl-udf**—Configures the size of the IPv4 router ACL (RACL) user-defined field (UDF) TCAM region. • **redirect**—Configures the size of the redirect TCAM region. • **redirect-tunnel**—Configures the size of the redirect-tunnel TCAM region, which is used for BFD over VXLAN. **Note** This command is supported only if the TP_SERVICES_PKG license is installed. • **rp-ipv6-qos**—Configures the size of the IPv6 port QoS TCAM region for the 100G 9408PC line card and the 100G M4PC generic expansion module (GEM). • **rp-mac-qos**—Configures the size of the MAC port QoS TCAM region for the 100G 9408PC line card and the 100G M4PC generic expansion module (GEM). • **rp-qos**—Configures the size of the IPv4 port QoS TCAM region for the 100G 9408PC line card and the 100G M4PC generic expansion module (GEM).

	Command or Action	Purpose
		• **rp-qos-lite**—Configures the size of the IPv4 port QoS lite TCAM region for the 100G 9408PC line card and the 100G M4PC generic expansion module (GEM). • **sflow**—Configures the size of the sFlow TCAM region. • **span**—Configures the size of the SPAN TCAM region. • **svi**—Configures the size of the ingress SVI counters TCAM region. • **vacl**—Configures the size of the IPv4 VACL TCAM region. • **vpc-convergence**—Configures the size of the vPC convergence TCAM region. • **vqos**—Configures the size of the IPv4 VLAN QoS TCAM region. • **vqos-lite**—Configures the size of the IPv4 VLAN QoS lite TCAM region. • *tcam-size*—TCAM size. The size has to a multiple of 256. If the size is more than 256, it has to be multiple of 512. For FHS, the range is from 0-4096. You can use the **no** form of this command to revert to the default TCAM region size. **Note** You can attach IPv4 user-defined fields (UDFs) to the racl, ifacl, and vacl TCAM regions using the **hardware access-list tcam region** {**racl** | **ifacl** | **vacl**} **qualify udf** *udf-names* command to configure IPv4 UDF-based SPAN or ERSPAN.You can attach IPv6 UDFs to the ing-l2-span-filter and ing-l3-span-filter TCAM regions using the **hardware access-list tcam region** {**ing-ifacl** | **ing-l2-span-filter** | **ing-l3-span-filter**} **qualify v6udf** *v6udf-names* commands to configure IPv6 UDF-based ERSPAN. For more information and configuration instructions, see the latest *Cisco Nexus 9000 Series NX-OS System Management Configuration Guide*.

	Command or Action	Purpose
Step 3	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.
Step 4	(Optional) **show hardware access-list tcam region** **Example:** `switch(config)# show hardware access-list tcam region`	Displays the TCAM sizes that will be applicable on the next reload of the device.
Step 5	**reload** **Example:** `switch(config)# reload`	Reloads the device. **Note** The new size values are effective only after you enter **copy running-config startup-config** + **reload** or reload all line card modules.

Example

The following example shows how to change the size of the n9k-arp-acl TCAM region on a Cisco Nexus NFE-enabled device:

```
switch(config)#hardware access-list tcam region n9k-arp-acl 256switch(config)#copy r s
switch(config)# reload
Configuring storm-control-cpu:
switch (config)# interface ethernet 1/10switch
switch (config-if)# storm-control-cpu arp rate 150
switch (config)# show access-list storm-control-cpu arp-stats interface ethernet 1/10

slot  1
```

The following example shows how to change the size of the RACL TCAM region on a Cisco Nexus 9500 Series switch:

```
switch(config)# hardware access-list tcam region racl 256
[SUCCESS] New tcam size will be applicable only at boot time.
You need to 'copy run start' and 'reload'
switch(config)# copy running-config startup-config
switch(config)# reload
WARNING: This command will reboot the system
Do you want to continue? (y/n) [n] y
```

This example shows how to display the TCAM region sizes to verify your changes:

```
switch(config)# show hardware access-list tcam region
         TCAM Region Sizes:

                              IPV4 PACL [ifacl] size =    512
                          IPV6 PACL [ipv6-ifacl] size =    0
                            MAC PACL [mac-ifacl] size =    0
                             IPV4 Port QoS [qos] size =    256
                         IPV6 Port QoS [ipv6-qos] size =    0
                           MAC Port QoS [mac-qos] size =    0
```

```
                FEX IPV4 PACL [fex-ifacl] size =    0
              FEX IPV6 PACL [fex-ipv6-ifacl] size =    0
                FEX MAC PACL [fex-mac-ifacl] size =    0
                 FEX IPV4 Port QoS [fex-qos] size =    0
             FEX IPV6 Port QoS [fex-ipv6-qos] size =    0
               FEX MAC Port QoS [fex-mac-qos] size =    0
                           IPV4 VACL [vacl] size =    512
                       IPV6 VACL [ipv6-vacl] size =    0
                         MAC VACL [mac-vacl] size =    0
                        IPV4 VLAN QoS [vqos] size =    0
                    IPV6 VLAN QoS [ipv6-vqos] size =    0
                      MAC VLAN QoS [mac-vqos] size =    0
                           IPV4 RACL [racl] size =    512
                       IPV6 RACL [ipv6-racl] size =    0
                IPV4 Port QoS Lite [qos-lite] size =    0
          FEX IPV4 Port QoS Lite [fex-qos-lite] size =    0
               IPV4 VLAN QoS Lite [vqos-lite] size =    0
                IPV4 L3 QoS Lite [l3qos-lite] size =    0
                      Egress IPV4 QoS [e-qos] size =    0
                 Egress IPV6 QoS [e-ipv6-qos] size =    0
                   Egress MAC QoS [e-mac-qos] size =    0
                       Egress IPV4 VACL [vacl] size =    512
                 Egress IPV6 VACL [ipv6-vacl] size =    0
                   Egress MAC VACL [mac-vacl] size =    0
                    Egress IPV4 RACL [e-racl] size =    256
               Egress IPV6 RACL [e-ipv6-racl] size =    0
            Egress IPV4 QoS Lite [e-qos-lite] size =    0
                           IPV4 L3 QoS [l3qos] size =    0
                     IPV6 L3 QoS [ipv6-l3qos] size =    0
                       MAC L3 QoS [mac-l3qos] size =    0
                                Ingress System size =  256
                                 Egress System size =  256
                                   SPAN [span] size =  256
                             Ingress COPP [copp] size =  256
                  Ingress Flow Counters [flow] size =    0
                    Egress Flow Counters [e-flow] size = 0
                    Ingress SVI Counters [svi] size =    0
                           Redirect [redirect] size =  512
                     NS IPV4 Port QoS [ns-qos] size =  256
                NS IPV6 Port QoS [ns-ipv6-qos] size =    0
                  NS MAC Port QoS [ns-mac-qos] size =    0
                    NS IPV4 VLAN QoS [ns-vqos] size =  256
               NS IPV6 VLAN QoS [ns-ipv6-vqos] size =    0
                 NS MAC VLAN QoS [ns-mac-vqos] size =    0
                     NS IPV4 L3 QoS [ns-l3qos] size =  256
                NS IPV6 L3 QoS [ns-ipv6-l3qos] size =    0
                  NS MAC L3 QoS [ns-mac-l3qos] size =    0
             VPC Convergence [vpc-convergence] size =  256
                IPSG SMAC-IP bind table [ipsg] size =    0
             Ingress ARP-Ether ACL [arp-ether] size =    0
          ranger+ IPV4 QoS Lite [rp-qos-lite] size =    0
                    ranger+ IPV4 QoS [rp-qos] size =  256
                ranger+ IPV6 QoS [rp-ipv6-qos] size =  256
                  ranger+ MAC QoS [rp-mac-qos] size =  256
                                NAT ACL[nat] size =    0
                                    Mpls ACL size =    0
                           Ingress IPv4 N3K QoS size =    0
                           Ingress IPv6 N3K QoS size =    0
                                    MOD RSVD size =    0
                               sFlow ACL [sflow] size =    0
                                 mcast bidir ACL size =    0
                                    Openflow size =    0
```

This example shows how to revert to the default RACL TCAM region size:

```
switch(config)# no hardware profile tcam region racl 512
[SUCCESS] New tcam size will be applicable only at boot time.
You need to 'copy run start' and 'reload'
switch(config)# copy running-config startup-config
switch(config)# reload
WARNING: This command will reboot the system
Do you want to continue? (y/n) [n] y
```

Using Templates to Configure ACL TCAM Region Sizes

You can use create and apply custom templates to configure ACL TCAM region sizes.

For all Cisco Nexus 9200, 9300, and 9500 Series switches, you can use this procedure or the Configuring ACL TCAM Region Sizes procedure to configure ACL TCAM region sizes. However, NFE2-enabled devices (such as the X9432C-S 100G line card and the C9508-FM-S fabric module) do not support the **hardware access-list tcam region** command and must use a template to configure the ACL TCAM region size.

Note

- Once you apply a TCAM template, the **hardware access-list tcam region** command will not work. You must uncommit the template in order to use the command.
- For information on configuring QoS TCAM carving, see the *Cisco Nexus 9000 Series NX-OS Quality of Service Configuration Guide*.
- The TCAM profile template is not supported on the C9508-FM-S fabric module.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	[**no**] **hardware profile tcam resource template** *template-name* **ref-template** {**nfe** | **nfe2** | {**l2-l3** | **l3**}}
	Example:
	`switch(config)# hardware profile tcam resource template SR_MPLS_CARVE ref-template nfe2`
	`switch(config-tcam-temp)#`
		Creates a template for configuring ACL TCAM region sizes.
		nfe—The default TCAM template for Network Forwarding Engine (NFE)-enabled Cisco Nexus 9300 and 9500 Series.
		nfe2—The default TCAM template for NFE2-enabled Cisco Nexus 9500 Series devices.
		l2-l3—The default TCAM template for Layer 2 and Layer 3 configurations.

	Command or Action	Purpose
		l3—The default TCAM template for Layer 3 configurations.
Step 3	(Optional) *region tcam-size* **Example:** `switch(config-tcam-temp)# mpls 256`	Adds any desired TCAM regions and their sizes to the template. Enter this command for each region you want to add to the template. For the list of available regions, see Configuring ACL TCAM Region Sizes.

Step 4	**exit**
	Example:
	`switch(config-tcam-temp)# exit`
	`switch(config#)`
		Exits the TCAM template configuration mode.

Step 5	[**no**] **hardware profile tcam resource service-template** *template-name* **Example:** `switch(config)# hardware profile tcam resource service-template SR_MPLS_CARVE`	Applies the custom template to all line cards and fabric modules.
Step 6	(Optional) **show hardware access-list tcam template** {**all** | **nfe** | **nfe2** | **l2-l3** | **l3** | *template-name*} **Example:** `switch(config)# show hardware access-list tcam template SR_MPLS_CARVE`	Displays the configuration for all TCAM templates or for a specific template.
Step 7	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.
Step 8	**reload** **Example:** `switch(config)# reload`	Reloads the device. **Note** The configuration is effective only after you enter **copy running-config startup-config** + **reload**.

Configuring TCAM Carving

The default TCAM region configuration varies by platform and does not accommodate all TCAM regions. To enable any desired regions, you must decrease the TCAM size of one region and then increase the TCAM size for the desired region.

Note For information on configuring QoS TCAM carving, see the *Cisco Nexus 9000 Series NX-OS Quality of Service Configuration Guide*.

The following tables list the default sizes for the ingress and egress TCAM regions on different platforms.

Table 5: Default TCAM Region Configuration (Ingress) - For Cisco Nexus 9500 Series Switches

Region Name	Size	Width	Total Size
IPv4 RACL	1536	1	1536
IPv4 Layer 3 QoS	256	2	512
SPAN	256	1	256
CoPP	256	2	512
System	256	2	512
Redirect	256	1	256
vPC convergence	512	1	512
			4K

Table 6: Default TCAM Region Configuration (Egress) - For Cisco Nexus 9500 Series Switches

Region Name	Size	Width	Total Size
IPv4 RACL	768	1	768
System	256	1	256
			1K

Table 7: Default TCAM Size - For Cisco Nexus 9504 and 9508 Platform switches

Region	Size
MAC PACL [mac-ifacl]	1952
IPV6 Port QoS [ipv6-qos]	256
PV6 L3 QoS [ipv6-l3qos]	256
SPAN [span]	96
Ingress CoPP [copp]	128
Redirect IPv4	2048
Redirect IPv6	2048

Table 8: Default TCAM Region Configuration (Ingress) - For Cisco Nexus 9300-FX Series Switches

Region Name	Size	Width	Total Size
IPv4 RACL	2304	1	2304
Layer 2 QoS	256	1	256
Layer 3/VLAN QoS	512	1	512

Region Name	Size	Width	Total Size
System	512	1	512
Layer 2 SPAN filter	256	1	256
Layer 3 SPAN filter	256	1	256
SPAN	512	1	512
NetFlow/Analytics filter	512	1	512
			5K

Table 9: Default TCAM Region Configuration (Egress) - For Cisco Nexus 9300-FX Series Switches

Region Name	Size	Width	Total Size
IPv4 RACL	1792	1	1792
System	256	1	256
			2K

Table 10: Default TCAM Region Configuration (Ingress) - For Cisco Nexus 9300-EX Series Switches

Region Name	Size	Width	Total Size
IPv4 RACL	1792	1	1792
Layer 2 QoS	256	1	256
Layer 3/VLAN QoS	512	1	512
System	512	1	512
Layer 2 SPAN ACL	256	1	256
Layer 3/VLAN SPAN ACL	256	1	256
SPAN	512	1	512
			4K

Table 11: Default TCAM Region Configuration (Egress) - For Cisco Nexus 9300-EX Series Switches

Region Name	Size	Width	Total Size
IPv4 RACL	1792	1	1792
System	256	1	256
			2K

Table 12: Default TCAM Region Configuration (Ingress) - For Cisco Nexus 9300 Series Switches

Region Name	Size	Width	Total Size
IPv4 port ACL	512	1	512

Region Name	Size	Width	Total Size
IPv4 port QoS	256	2	512
IPv4 VACL	512	1	512
IPv4 RACL	512	1	512
SPAN	256	1	256
CoPP	256	2	512
IPv4 port QoS for ACI leaf line card	256	1	256
IPv4 VLAN QoS for ACI leaf line card	256	1	256
IPv4 Layer 3 QoS for ACI leaf line card	256	1	256
System	256	2	512
Redirect	512	1	512
vPC convergence	256	1	256
			4K

Table 13: Default TCAM Region Configuration (Egress) - For Cisco Nexus 9300 Series Switches

Region Name	Size	Width	Total Size
IPv4 VACL	512	1	512
IPv4 RACL	256	1	256
System	256	1	256
			1K

Table 14: Default TCAM Region Configuration (Ingress) - For Layer 2-to-Layer 3 Configurations on Cisco Nexus 9200 Series Switches

Region Name	Size	Width	Total Size
Ingress NAT	0	1	0
Ingress port ACL	256	1	256
Ingress VACL	256	1	256
Ingress RACL	1536	1	1536
Ingress Layer 2 QoS	256	1	256
Ingress Layer 3 VLAN QoS	256	1	256
Ingress supervisor	512	1	512
Ingress Layer 2 ACL SPAN	256	1	256

Region Name	Size	Width	Total Size
Ingress Layer 3 ACL SPAN	256	1	256
Port-based SPAN	512	1	512
			4096

Table 15: Default TCAM Region Configuration (Egress) - For Layer 2-to-Layer 3 Configurations on Cisco Nexus 9200 Series Switches

Region Name	Size	Width	Total Size
Egress VACL	256	1	256
Egress RACL	1536	1	1536
Egress supervisor	256	1	256
			2048

Table 16: Default TCAM Region Configuration (Ingress) - For Layer 3 Configurations on Cisco Nexus 9200 Series Switches

Region Name	Size	Width	Total Size
Ingress NAT	0	1	0
Ingress port ACL	0	1	0
Ingress VACL	0	1	0
Ingress RACL	1792	1	1792
Ingress Layer 2 QoS	256	1	256
Ingress Layer 3 VLAN QoS	512	1	512
Ingress supervisor	512	1	512
Ingress Layer 2 ACL SPAN	256	1	256
Ingress Layer 3 ACL SPAN	256	1	256
Port-based SPAN	512	1	512
			4096

Table 17: Default TCAM Region Configuration (Egress) - For Layer 3 Configurations on Cisco Nexus 9200 Series Switches

Region Name	Size	Width	Total Size
Egress VACL	0	1	0
Egress RACL	1792	1	1792
Egress supervisor	256	1	256
			2048

The following example sets the IPv6 RACL TCAM size to 256 on a Cisco Nexus 9500 Series switch. An IPv6 RACL of size 256 takes 512 entries because IPv6 is double wide.

Note Follow a similar procedure to modify the TCAM settings for a different region or to modify the TCAM settings on a different device.

To set the size of the ingress IPv6 RACL TCAM region on a Cisco Nexus 9500 Series switch, perform one of two options.

Option #1

Reduce the ingress IPv4 RACL by 1024 entries (1536 - 1024 = 512) and add an ingress IPv6 RACL with 512 entries—This option is preferred.

```
switch(config)# hardware access-list tcam region racl 512
Warning: Please reload the linecard for the configuration to take effect
switch(config)# hardware access-list tcam region ipv6-racl 256
Warning: Please reload the linecard for the configuration to take effect
```

Table 18: Updated TCAM Region Configuration After Reducing the IPv4 RACL (Ingress)

Region Name	Size	Width	Total Size
IPv4 RACL	1024	1	1024
IPv6 RACL	256	2	1024[2]
IPv4 Layer 3 QoS	256	2	512
SPAN	256	1	256
CoPP	256	2	512
System	256	2	512
Redirect	256	1	256
vPC convergence	512	1	512
			4K

[2] 2 x 512 entry slices are allocated due to the non-availability of 256 entry slices.

Option #2

Remove IPv4 Layer 3 QoS by reducing its size to 0 and add an ingress IPv6 RACL—This option is available if you are not using IPv4 Layer 3 QoS.

```
switch(config)# hardware access-list tcam region l3qos 0
Warning: Please reload the linecard for the configuration to take effect
switch(config)# hardware access-list tcam region ipv6-racl 256
Warning: Please reload the linecard for the configuration to take effect
```

Table 19: Updated TCAM Region Configuration After Removing Layer 3 QoS (Ingress)

Region Name	Size	Width	Total Size
IPv4 RACL	1536	1	1536

Region Name	Size	Width	Total Size
IPv6 RACL	256	2	512
IPv4 Layer 3 QoS	0	2	0
SPAN	256	1	256
CoPP	256	2	512
System	256	2	512
Redirect	256	1	256
vPC convergence	512	1	512
			4K

To enable an egress IPv6 RACL of size 256, reduce the egress IPv4 RACL to 256 and add the egress IPv6 RACL:

```
switch(config)# hardware access-list tcam region e-racl 256
Warning: Please reload the linecard for the configuration to take effect
switch(config)# hardware access-list tcam region e-ipv6-racl 256
Warning: Please reload the linecard for the configuration to take effect
```

Table 20: Default TCAM Region Configuration After Reducing the IPv4 RACL (Egress)

Region Name	Size	Width	Total Size
IPv4 RACL	256	1	256
IPv6 RACL	256	2	512
System	256	1	256
			1K

Note Each IPv6 ACL is limited to 1,000 ACEs. This applies to all IPv6 ACLs (RACL, QoS or SPAN filter). No such limitation applies for IPv4 ACL.

After you adjust the TCAM region sizes, enter the **show hardware access-list tcam region** command to display the TCAM sizes that will be applicable on the next reload of the device.

Attention To keep all modules synchronized, you must reload all line card modules or enter **copy running-config startup-config** + **reload** to reload the device. Multiple TCAM region configurations require only a single reload. You can wait until you complete all of your TCAM region configurations before you reload the device.

Depending on the configuration, you might exceed the TCAM size or run out of slices.

If you exceed the 4K ingress limit for all TCAM regions when you configure a TCAM region, the following message appears:

```
ERROR: Aggregate TCAM region configuration exceeded the available Ingress TCAM space. Please
 re-configure.
```

If you exceed the number of slices, the following message appears:

```
ERROR: Aggregate TCAM region configuration exceeded the available Ingress TCAM slices.
Please re-configure.
```

If you exceed the 1K egress limit for all TCAM regions when you configure a TCAM region, the following message appears:

```
ERROR: Aggregate TCAM region configuration exceeded the available Egress TCAM space. Please
 re-configure.
```

If TCAM for a particular feature is not configured and you try to apply a feature that requires TCAM carving, the following message appears:

```
 ERROR: Module x returned status: TCAM region is not configured. Please configure TCAM
region and retry the command.
```

Note The default redirect TCAM region size of 256 might not be sufficient if you are running many BFD or DHCP relay sessions. To accommodate more BFD or DHCP relay sessions, you might need to increase the TCAM size to 512 or greater.

Note "e-racl" tcam region size can be maximum of 16K when we have at least one "N9K-X9624D-R2" line card on a N9K-C9508 (Fretta) system.

Related Topics

Configuring ACL TCAM Region Sizes, on page 33

Configuring UDF-Based Port ACLs

You can configure UDF-based port ACLs for Cisco Nexus 9200, 9300, and 9300-EX Series switches. This feature enables the device to match on user-defined fields (UDFs) and to apply the matching packets to an IPv4 port ACL.

You can configure UDF-based port IPv6 ACLs for Cisco Nexus 9300-EX switches. This feature enables the device to match on the new UDFs and to apply the matching packets to an IPv6 port ACL.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**udf** *udf-name offset-base offset length* **Example:** `switch(config)# udf pktoff10 packet-start 10 2`	Defines the UDF as follows: • *udf-name*—Specifies the name of the UDF. You can enter up to 16 alphanumeric characters for the name.

	Command or Action	Purpose
	Example: `switch(config)# udf pktoff10 header outer l3 20 2`	• *offset-base*—Specifies the UDF offset base as follows, where **header** is the packet header to consider for the offset: {**packet-start** | **header** {**outer** | **inner** {**l3** | **l4**}}}. • *offset*—Specifies the number of bytes offset from the offset base. To match the first byte from the offset base (Layer 3/Layer 4 header), configure the offset as 0. • *length*—Specifies the number of bytes from the offset. Only 1 or 2 bytes are supported. To match additional bytes, you must define multiple UDFs. You can define multiple UDFs, but Cisco recommends defining only required UDFs.
Step 3	**hardware access-list tcam region ing-ifacl qualify** {**udf** *udf-name* | **v6udf** *v6udf-name*} **Example:** `switch(config)# hardware access-list tcam region ing-ifacl qualify udf pktoff10`	Attaches the UDFs to the ing-ifacl TCAM region, which applies to IPv4 or IPv6 port ACLs. The number of UDFs that can be attached to a TCAM region varies by platform. You can attach up to 2 UDFs for Cisco Nexus 9200 switches, up to 8 UDFs for Cisco Nexus 9300 switches, and up to 18 UDFs for IPv4 port ACLs or 7 UDFs for IPv6 port ACLs for Cisco Nexus 9300-EX switches. **Note** When the UDF qualifier is added, the TCAM region goes from single wide to double wide. Make sure enough free space is available; otherwise, this command will be rejected. If necessary, you can reduce the TCAM space from unused regions and then re-enter this command. For more information, see Configuring ACL TCAM Region Sizes. **Note** The **no** form of this command detaches the UDFs from the TCAM region and returns the region to single wide.
Step 4	Required: **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

	Command or Action	Purpose
Step 5	Required: **reload** **Example:** `switch(config)# reload`	Reloads the device. **Note** Your UDF configuration is effective only after you enter **copy running-config startup-config** + **reload**.

Step 6	**ip access-list** *udf-acl*
	Example:
	`switch(config)# ip access-list udfacl`
	`switch(config-acl)#`
		Creates an IPv4 access control list (ACL) and enters IP access list configuration mode.

Step 7	Enter one of the following commands:
	• **permit udf** *udf-name value mask*
	• **permit ip** *source destination* **udf** *udf-name value mask*
	Example:
	`switch(config-acl)# permit udf pktoff10 0x1234 0xffff`
	Example:
	`switch(config-acl)# permit ip any any udf pktoff10 0x1234 0xffff`
		Configures the ACL to match only on UDFs (example 1) or to match on UDFs along with the current access control entries (ACEs) for the outer packet fields (example 2). The range for the *value* and *mask* arguments is from 0x0 to 0xffff.
		A single ACL can have ACEs with and without UDFs together. Each ACE can have different UDF fields to match, or all ACEs can match for the same list of UDFs.

Step 8	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Applying an IP ACL as a Router ACL

You can apply an IPv4 or IPv6 ACL to any of the following types of interfaces:

- Physical Layer 3 interfaces and subinterfaces
- Layer 3 Ethernet port-channel interfaces
- VLAN interfaces
- Management interfaces

ACLs applied to these interface types are considered router ACLs.

Note Egress router ACLs are not supported on Cisco Nexus 9300 Series switch uplink ports.

Before you begin

Ensure that the ACL you want to apply exists and that it is configured to filter traffic in the manner that you need for this application.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	Enter one of the following commands:
	• **interface ethernet** *slot*/*port*[**.** *number*]
	• **interface port-channel** *channel-number*
	• **interface vlan** *vlan-id*
	• **interface mgmt** *port*
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
	`switch(config)# interface ethernet 2/3.1`
	`switch(config-if)#`
		Enters configuration mode for the interface type that you specified.

Step 3	(Optional) **encapsulation dot1q 21**
	Example:
	`switch(config-if)# encapsulation dot1q 21`
	`switch(config-if)#`
		Note
		This command is required only for Layer 3 subinterfaces.

Step 4	Enter one of the following commands: • **ip access-group** *access-list* {**in** | **out**} • **ipv6 traffic-filter** *access-list* {**in** | **out**} **Example:** `switch(config-if)# ip access-group acl1 in`	Applies an IPv4 or IPv6 ACL to the Layer 3 interface and subinterfaces for traffic flowing in the direction specified. You can apply one router ACL per direction.
Step 5	**ip access-list match-local-traffic** **Example:** `switch(config-if)# ip access-list match-local-traffic`	Lists the matching traffic which is generated locally. It does not affect transit traffic through the switch.
Step 6	(Optional) **show running-config aclmgr** **Example:** `switch(config-if)# show running-config aclmgr`	Displays the ACL configuration.

	Command or Action	Purpose
Step 7	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Related Topics

Creating an IP ACL, on page 26

Applying an IP ACL as a Port ACL

You can apply an IPv4 or IPv6 ACL to a Layer 2 interface, which can be a physical port or a port channel. ACLs applied to these interface types are considered port ACLs.

Note If the interface is configured with the **mac packet-classify** command, you cannot apply an IP port ACL to the interface until you remove the **mac packet-classify** command from the interface configuration.

Before you begin

Ensure that the ACL you want to apply exists and that it is configured to filter traffic in the manner that you need for this application.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	Enter one of the following commands:
	• **interface ethernet** *slot/port*
	• **interface port-channel** *channel-number*
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
		Enters configuration mode for the interface type that you specified.

Step 3	Enter one of the following commands: • **ip port access-group** *access-list* **in** • **ipv6 port traffic-filter** *access-list* **in** **Example:**	Applies an IPv4 or IPv6 ACL to the interface or port channel. Only inbound filtering is supported with port ACLs. You can apply one port ACL to an interface.

	Command or Action	Purpose
	`switch(config-if)# ip port access-group acl-l2-marketing-group in`	
Step 4	(Optional) **show running-config aclmgr** **Example:** `switch(config-if)# show running-config aclmgr`	Displays the ACL configuration.
Step 5	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Related Topics

Creating an IP ACL, on page 26
Enabling or Disabling MAC Packet Classification

Applying an IP ACL as a VACL

You can apply an IP ACL as a VACL.

Related Topics

Configuring VACLs

Configuring ACL Logging

To configure the ACL logging process, you first create the access list, then enable filtering of traffic on an interface using the specified ACL, and finally configure the ACL logging process parameters.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**ip access-list** *name*
	Example:
	`switch(config)# ip access-list logging-test`
	`switch(config-acl)#`
		Creates an IPv4 ACL and enters IP ACL configuration mode. The *name* argument can be up to 64 characters.

Step 3	{**permit** | **deny**} **ip** *source-address destination-address* **log**	Creates an ACL rule that permits or denies IPv4 traffic matching its conditions. To enable the system to generate an informational

	Command or Action	Purpose
	Example: `switch(config-acl)# permit ip any 10.30.30.0/24 log`	logging message about each packet that matches the rule, you must include the **log** keyword. The *source-address* and *destination-address* arguments can be the IP address with a network wildcard, the IP address and variable-length subnet mask, the host address, or **any** to designate any address.

Step 4	**exit**
	Example:
	`switch(config-acl)# exit`
	`switch(config)#`
		Updates the configuration and exits IP ACL configuration mode.

Step 5	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 1/1`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 6	**ip access-group** *name* **in** **Example:** `switch(config-if)# ip access-group logging-test in`	Enables the filtering of IPv4 traffic on an interface using the specified ACL. You can apply an ACL to inbound traffic.

Step 7	**exit**
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Updates the configuration and exits interface configuration mode.

Step 8	**logging ip access-list cache interval** *interval* **Example:** `switch(config)# logging ip access-list cache interval 490`	Configures the log-update interval (in seconds) for the ACL logging process. The default value is 300 seconds. The range is from 5 to 86400 seconds.
Step 9	**logging ip access-list cache entries** *number-of-flows* **Example:** `switch(config)# logging ip access-list cache entries 8001`	Specifies the maximum number of flows to be monitored by the ACL logging process. The default value is 8000. The range of values supported is from 0 to 1048576.
Step 10	**logging ip access-list cache threshold** *threshold* **Example:** `switch(config)# logging ip access-list cache threshold 490`	If the specified number of packets is logged before the expiry of the alert interval, the system generates a syslog message.

	Command or Action	Purpose
Step 11	**logging ip access-list detailed** **Example:** `switch(config)# logging ip access-list detailed`	Enables the following information to be displayed in the output of the **show logging ip access-list cache** command: the access control entry (ACE) sequence number, ACE action, ACL name, ACL direction, ACL filter type, and ACL applied interface.
Step 12	**hardware rate-limiter access-list-log** *packets* **Example:** `switch(config)# hardware rate-limiter access-list-log 200`	Configures rate limits in packets per second for packets copied to the supervisor module for ACL logging. The range is from 0 to 30000.
Step 13	**acllog match-log-level** *severity-level* **Example:** `switch(config)# acllog match-log-level 5`	Specifies the minimum severity level to log ACL matches. The default is 6 (informational). The range is from 0 (emergency) to 7 (debugging).
Step 14	(Optional) **show logging ip access-list cache** [**detail**] **Example:** `switch(config)# show logging ip access-list cache`	Displays information on the active logged flows, such as source IP and destination IP addresses, source port and destination port information, source interfaces. No other information of active flows will be displayed specifically all the unsupported options. If you entered the **logging ip access-list detailed** command, the output also includes the following information: the access control entry (ACE) sequence number, ACE action, ACL name, ACL direction, ACL filter type, and ACL applied interface.

Configuring ACLs Using HTTP Methods to Redirect Requests

You can configure ACLs to intercept and redirect specific HTTP methods to a server that is connected to a specific port.

The following HTTP methods can be redirected:

- connect
- delete
- get
- head
- post
- put
- trace

Before you begin

Enable the double-wide TCAM for the IFACL region using the **hardware access-list tcam region ifacl 512 double-wide** command. This command applies to the global configuration. Reload the switch for this configuration to take into effect.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**ip access-list** *name*
	Example:
	`switch(config)# ip access-list acl-01`
	`switch(config-acl)#`
		Creates the IP ACL and enters IP ACL configuration mode. The *name* argument can be up to 64 characters.

Step 3	[*sequence-number*] **permit** *protocol source destination* **http-method** *method* [**tcp-option-length** *length*] [**redirect** *interface*] **Example:** `switch(config-acl)# permit tcp 1.1.1.1/32 any http-method get`	Configures the ACL to redirect specific HTTP methods to a server. The following HTTP methods are supported: • connect—Matches HTTP packets with the CONNECT method [0x434f4e4e] • delete—Matches HTTP packets with the DELETE method [0x44454c45] • get—Matches HTTP packets with the GET method [0x47455420] • head—Matches HTTP packets with the HEAD method [0x48454144] • post—Matches HTTP packets with the POST method [0x504f5354] • put—Matches HTTP packets with the PUT method [0x50555420] • trace—Matches HTTP packets with the TRACE method [0x54524143] The **tcp-option-length** option specifies the length of the TCP options header in the packets. You can configure up to four TCP option lengths (in multiples of four bytes) in the access control entries (ACEs). The *length* range is from 0 to 40. If you do not configure this option, the length is specified as 0, and only packets without the TCP options header can match the

	Command or Action	Purpose
		ACE. This option allows the HTTP method to be matched even on packets that have a variable-length TCP options header. The **redirect** option redirects an HTTP method to a server that is connected to a specific port. The HTTP redirect feature does not work on Layer 3 ports.
Step 4	(Optional) **show ip access-lists** *name* **Example:** `switch(config-acl)# show ip access-lists acl-01`	Displays the IP ACL configuration.
Step 5	(Optional) **show run interface** *interface slot/port* **Example:** `switch(config-acl)# show run interface ethernet 2/2`	Displays the interface configuration.

Example

The following example specifies a length for the TCP options header in the packets and redirects the post HTTP method to a server that is connected to port channel 4001:

```
switch(config)# ip access-list http-redirect-acl
switch(config-acl)# 10 permit tcp any any http-method get tcp-option-length 4 redirect
port-channel4001
switch(config-acl)# 20 permit tcp any any http-method post redirect port-channel4001
switch(config-acl)# statistics per-entry
switch(config)# interface Ethernet 1/33
switch(config-if)# ip port access-group http-redirect-acl in
```

Configuring an ACL for IPv6 Extension Headers

This procedure applies only to the following devices:

- Cisco Nexus 9504 and 9508 modular chassis with these line cards: N9K-X9636C-R, N9K-X9636Q-R, N9K-X9636C-RX, and N9K-X96136YC-R
- Cisco Nexus 3600 Platform Switches (N3K-C36180YC-R and N3K-C3636C-R)

Beginning with Cisco NX-OS Release 9.3(7), if you configure an IPv6 ACL on the devices listed here, you must include a new rule for the disposition of IPv6 packets that include extension headers. For more information about IPv6 extension headers, see "Simplified IPv6 Packet Header" in NX-OS Release 9.3(x) or later of the *Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide*.

Note The permit or deny rule that you choose in this procedure is applied to any IPv6 packet with at least one extension header regardless of any other ACL rule that matches the packet's other fields.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**ipv6 access-list** *name*
	Example:
	`switch(config)# ipv6 access-list acl-01`
	`switch(config-acl)#`
		Creates the IPv6 ACL and enters ACL configuration mode.

Step 3	**extension-header** {**permit-all** | **deny-all**}
	Example:
	`switch(config-acl)# extension-header permit-all`
	`switch(config-acl)#`
		Choose the desired action for matched packets:
		• **permit-all** — Any IPv6 packet with at least one extension header is permitted.
		• **deny-all** — Any IPv6 packet with at least one extension header is dropped.

Verifying the IP ACL Configuration

To display IP ACL configuration information, perform one of the following tasks.

Command	Purpose
show hardware access-list tcam region	Displays the TCAM sizes that will be applicable on the next reload of the device.

Command	Purpose
show hardware access-list tcam template {**all** | **nfe** | **nfe2** | **l2-l3** | **l3** | *template-name*}	Displays the configuration for all TCAM templates or for a specific template. **nfe**—The default TCAM template for Network Forwarding Engine (NFE)-enabled Cisco Nexus 9300 and 9500 Series devices. **nfe2**—The default TCAM template for NFE2-enabled Cisco Nexus 9500 devices. **l2-l3**—The default TCAM template for Layer 2 and Layer 3 configurations. **l3**—The default TCAM template for Layer 3 configurations.
show ip access-lists	Displays the IPv4 ACL configuration.
show ipv6 access-lists	Displays the IPv6 ACL configuration.
show logging ip access-list cache [**detail**]	Displays information on the active logged flows, such as source IP and destination IP addresses, source port and destination port information, and source interfaces. No other information of active flows will be displayed specifically all the unsupported options. If you entered the **logging ip access-list detailed** command, the output also includes the following information: the access control entry (ACE) sequence number, ACE action, ACL name, ACL direction, ACL filter type, and ACL applied interface.
show logging ip access-list status	Displays the deny maximum flow count, the current effective log interval, and the current effective threshold value.
show running-config acllog	Displays the ACL log running configuration.

Command	Purpose
show running-config aclmgr [**all**]	Displays the ACL running configuration, including the IP ACL configuration and the interfaces to which IP ACLs are applied. **Note** This command displays the user-configured ACLs in the running configuration. The **all** option displays both the default (CoPP-configured) and user-configured ACLs in the running configuration.
show startup-config acllog	Displays the ACL log startup configuration.
show startup-config aclmgr [**all**]	Displays the ACL startup configuration. **Note** This command displays the user-configured ACLs in the startup configuration. The **all** option displays both the default (CoPP-configured) and user-configured ACLs in the startup configuration.

show hardware access-list interface *ethernet X/Y* **input entries detail**	Displays the hardware ACL interface input entries' detail.
	Note
	On platforms other than 9500-R, even when the entry is expanded, the display shows the range as x y.
	Sample output for 9500-R:
	`permit tcp 100.1.1.0/24 eq 10006 100.1.1.0/24 eq 0x4e24/fffe [0]`
	Sample output for 9300-FX3S:
	`permit tcp 100.1.1.0/24 eq 10006 100.1.1.0/24 range 20004 20005 routeable 0x1 [0]`

Monitoring and Clearing IP ACL Statistics

To monitor or clear IP ACL statistics, use one of the commands in this table.

Command	Purpose
show ip access-lists	Displays the IPv4 ACL configuration. If the IPv4 ACL includes the **statistics per-entry** command, the **show ip access-lists** command output includes the number of packets that have matched each rule.
show ipv6 access-lists	Displays IPv6 ACL configuration. If the IPv6 ACL includes the **statistics per-entry** command, then the **show ipv6 access-lists** command output includes the number of packets that have matched each rule.
clear ip access-list counters	Clears statistics for all IPv4 ACLs or for a specific IPv4 ACL.
clear ipv6 access-list counters	Clears statistics for all IPv6 ACLs or for a specific IPv6 ACL.

Configuration Examples for IP ACLs

The following example shows how to create an IPv4 ACL named acl-01 and apply it as a port ACL to Ethernet interface 2/1, which is a Layer 2 interface:

```
ip access-list acl-01
  permit ip 192.168.2.0/24 any
interface ethernet 2/1
  ip port access-group acl-01 in
```

The following example shows how to create an IPv6 ACL named acl-120 and apply it as a router ACL to Ethernet interface 2/3, which is a Layer 3 interface:

```
ipv6 access-list acl-120
  permit tcp 2001:0db8:85a3::/48 2001:0db8:be03:2112::/64
  permit udp 2001:0db8:85a3::/48 2001:0db8:be03:2112::/64
  permit tcp 2001:0db8:69f2::/48 2001:0db8:be03:2112::/64
  permit udp 2001:0db8:69f2::/48 2001:0db8:be03:2112::/64
interface ethernet 2/3
  ipv6 traffic-filter acl-120 in
```

The following example shows how to create a VTY ACL named single-source and apply it on input IP traffic over the VTY line. This ACL allows all TCP traffic through and drops all other IP traffic:

```
ip access-list single-source
  permit tcp 192.168.7.5/24 any
  exit
  line vty
  ip access-class single-source in
  show ip access-lists
```

The following example shows how to configure IPv4 ACL logging:

```
switch# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
switch(config)# ip access-list logging-test
```

```
switch(config-acl)# permit ip any 2001:DB8:1::1/64 log
switch(config-acl)# exit
switch(config)# interface ethernet 1/1
switch(config-if)# ip access-group logging-test in
switch(config-if)# exit
switch(config)# logging ip access-list cache interval 400
switch(config)# logging ip access-list cache entries 100
switch(config)# logging ip access-list cache threshold 900
switch(config)# hardware rate-limiter access-list-log 200
switch(config)# acllog match-log-level 5
```

The following example shows how to configure a UDF-based port ACL:

```
switch# configure terminal
switch(config)# hardware access-list tcam region ing-ifacl 256
switch(config)# udf pktoff10 packet-start 10 2
switch(config)# udf pktoff20 packet-start 10 1
switch(config)# hardware access-list tcam region ing-ifacl qualify udf pktoff10 pktoff20

switch# configure terminal
switch(config)# ip access-list udfacl
switch(config-acl)# statistics per-entry
switch(config-acl)# 10 permit ip any any udf pktoff10 0x1234 0xffff

switch# configure terminal
switch(config)# interface Ethernet1/1
switch(config-if)# ip port access-group udfacl in
switch(config-if)# switchport
switch(config-if)# no shutdown
```

About System ACLs

You can configure system ACLs on Cisco Nexus 9500 Series switches with -R and -RX line cards. With system ACLs, you can now configure a Layer 2 port ACL (PACL) on all the ports with the same access-list in the switch. Configuring system ACLs reduces the TCAM usage and also brings down the time and memory usage while the policy is being applied or modified.

See the following guidelines and limitations for configuring system ACLs:

- The system PACL is supported for Layer 2 interface only.
- Up to 10K ACEs are supported with all other basic features for the switch to come up on Cisco Nexus 9500 Series switches with -R line cards. The hardware capacity on Cisco Nexus 9500 Series switches with -RX line cards is 64K ACEs.
- You can also configure system ACLs on Cisco Nexus 3600 platform switches with N3K-C3636C-R and N3K-C36180YC-R line cards.
- Configuring IPv4 PACL TCAM region (ifacl) with anything more than the total physical TCAM capacity of -R line cards of 12k results in the power down of -R line cards only.
- ACE statistics are not yet supported for the system ACLs.
- IPv6 is not yet supported in the system ACLs.
- System ACLs are not supported on the breakout port.

- For quality of service, ACL, or TCAM carving configuration on Cisco Nexus Series switches with -R series line cards, see the Cisco Nexus 3600 NX-OS Quality of Service Configuration Guide, Release 7.x for more information.
- The non-atomic update either drops or permits all the traffic. By default, the non-atomic update drops all the traffic until the ACL update completes. The non-atomic ACL update behavior can be controlled using the **hardware access-list update default-result permit** CLI command. This CLI works only for physical ports. See the following example:

```
hardware access-list update default-result permit     => #Allows all the traffic during
 ACL updates. There may be < 10secs traffic drop.
no hardware access-list update default-result permit   => #This is the default behavior.
 It denies all the traffic during ACL updates.
```

- In Cisco NX-OS Release 9.2(2) and earlier releases, although the atomic ACL update is not supported on Cisco Nexus -R series line cards, the non-atomic update **hardware access-list update default-result** is supported on the Cisco Nexus -R series line cards.

Carving a TCAM Region

Before configuring the system ACLs, carve the TCAM region first. Note that for configuring the ACLs less than 1k, you do not need to carve the TCAM region. See the Configuring ACL TCAM Region Sizes, on page 33 section for more information.

Note Beginning with Cisco NX-OS Release 7.0(3)F3(4) or a later release, you can configure PACL IPv4, RACL IPv4, and RACL IPv6 beyond 12k.

Configuring System ACLs

After an IPv4 ACL is created, configure the system ACL.

Before you begin

Create an IPv4 ACL on the device. See Creating an IP ACL, on page 26 for more information.

Procedure

	Command or Action	Purpose
Step 1	**config t**	Enters the configuration mode.
Step 2	**system acl**	Configures the system ACL.
Step 3	**ip port access-group** <*pacl name*> **in**	Applies a Layer 2 PACL to the interface. Only inbound filtering is supported with port ACLs. You can apply one port ACL to an interface.

Configuration and Show Command Examples for the System ACLs

See the following configuration examples for the system ACL show commands.

Configuring system PACL with 1K scale [using default TCAM]

See the following example for configuring system PACL with 1K scale [Using default TCAM].

Step 1: Create PACL.

```
config t
ip access-list PACL-DNA
    10 permit ip 1.1.1.1/32 any
    20 permit tcp 3.0.0.0/8 255.0.0.0 eq 1500
    25 deny udp any any eq 500
    26 deny tcp any eq 490 any
    …·· ….

    1000 deny any any
```

Step 2: Apply PACL into system level.

```
configuration terminal
system acl
    ip port access-group PACL-DNA in
```

To validate the system ACLs that are configured on the switch, use the **sh run aclmgr | sec system** command:

```
switch# sh run aclmgr | sec system
system acl
 ip port access-group test in
switch#
```

To validate the PACLs that are configured on the switch, use the **sh ip access-lists <name> [summary]** command:

```
switch# sh ip access-lists test

IP access list test
        10 deny udp any any eq 27
        20 permit ip 1.1.1.1/32 100.100.100.100/32
        30 permit ip 1.2.1.1/32 100.100.100.100/32
        40 permit ip 1.3.1.1/32 100.100.100.100/32
        50 permit ip 1.4.1.1/32 100.100.100.100/32
        60 permit ip 1.5.1.1/32 100.100.100.100/32
        70 permit ip 1.6.1.1/32 100.100.100.100/32
        80 permit ip 1.7.1.1/32 100.100.100.100/32
        90 permit ip 1.8.1.1/32 100.100.100.100/32

switch# sh ip access-lists test summary
IPV4 ACL test
        Total ACEs Configured: 12279
        Configured on interfaces:
        Active on interfaces:
                - ingress
                - ingress
```

```
switch#
```

To validate PACL IPv4 (ifacl) TCAM region size, use the **show hardware access-list tcam region** command:

```
switch# show hardware access-list tcam region
*********************************WARNING*********************************
*****************The output shows NFE tcam region info*****************
***Please refer to 'show hardware access-list tcam template' for NFE2***
*************************************************************************
                              IPV4 PACL [ifacl] size = 12280
                           IPV6 PACL [ipv6-ifacl] size =    0
                             MAC PACL [mac-ifacl] size =    0
                              IPV4 Port QoS [qos] size =  640
                           IPV6 Port QoS [ipv6-qos] size =  256
                             MAC Port QoS [mac-qos] size =    0
                         FEX IPV4 PACL [fex-ifacl] size =    0
                      FEX IPV6 PACL [fex-ipv6-ifacl] size =    0
                        FEX MAC PACL [fex-mac-ifacl] size =    0
                         FEX IPV4 Port QoS [fex-qos] size =    0
                   FEX IPV6 Port QoS [fex-ipv6-qos] size =    0
                     FEX MAC Port QoS [fex-mac-qos] size =    0
                                  IPV4 VACL [vacl] size =    0
                            IPV6 VACL [ipv6-vacl] size =    0
                              MAC VACL [mac-vacl] size =    0
                             IPV4 VLAN QoS [vqos] size =    0
                          IPV6 VLAN QoS [ipv6-vqos] size =    0
                            MAC VLAN QoS [mac-vqos] size =    0
                                  IPV4 RACL [racl] size =    0
                            IPV6 RACL [ipv6-racl] size =  128
                     IPV4 Port QoS Lite [qos-lite] size =    0
               FEX IPV4 Port QoS Lite [fex-qos-lite] size =    0
                      IPV4 VLAN QoS Lite [vqos-lite] size =    0
                       IPV4 L3 QoS Lite [l3qos-lite] size =    0
                             Egress IPV4 QoS [e-qos] size =    0
                          Egress IPV6 QoS [e-ipv6-qos] size =    0
                            Egress MAC QoS [e-mac-qos] size =    0
                               Egress IPV4 VACL [vacl] size =    0
                          Egress IPV6 VACL [ipv6-vacl] size =    0
                            Egress MAC VACL [mac-vacl] size =    0
                             Egress IPV4 RACL [e-racl] size =    0
                        Egress IPV6 RACL [e-ipv6-racl] size =    0
                    Egress IPV4 QoS Lite [e-qos-lite] size =    0
                              IPV4 L3 QoS [l3qos] size =  640
                           IPV6 L3 QoS [ipv6-l3qos] size =  256
                             MAC L3 QoS [mac-l3qos] size =    0
                                      Ingress System size =    0
                                       Egress System size =    0
                                         SPAN [span] size =   96
                                  Ingress COPP [copp] size =  128
                           Ingress Flow Counters [flow] size =    0
switch#
```

To view ACL related tech support information, use the **show tech-support aclmgr** and **show tech-support aclqos** commands.

```
show tech-support aclmgr
show tech-support aclqos
```

Configuring Object Groups

You can use object groups to specify source and destination addresses and protocol ports in IPv4 ACL and IPv6 ACL rules.

Session Manager Support for Object Groups

Session Manager supports the configuration of object groups. This feature allows you to create a configuration session and verify your object group configuration changes prior to committing them to the running configuration. For more information about Session Manager, see the *Cisco Nexus 9000 Series NX-OS System Management Configuration Guide*.

Creating and Changing an IPv4 Address Object Group

You can create and change an IPv4 address group object.

Note Beginning Cisco Nexus Release 7.0(3)I5(2), the **no host IPv4-address** command is not supported. With the DME support, deletion without the no sequence command is not supported.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**object-group ip address** *name*
	Example:
	`switch(config)# object-group ip address ipv4-addr-group-13`
	`switch(config-ipaddr-ogroup)#`
		Creates the IPv4 address object group and enters IPv4 address object-group configuration mode.

Step 3	Enter one of the following commands: • [*sequence-number*] **host** *IPv4-address* • [*sequence-number*] *IPv4-address*/*prefix-len* • [*sequence-number*] *IPv4-address network-wildcard* **Example:** `switch(config-ipaddr-ogroup)# host 10.99.32.6`	Creates an entry in the object group. For each entry that you want to create, use the **host** command and specify a single host, or omit the **host** command to specify a network of hosts. You can specify a prefix length for an IPv4 object group, which matches only on the first contiguous bits, or you can specify a wildcard mask, which matches on any bit in the address.

	Command or Action	Purpose
Step 4	Enter one of the following commands: • **no** [*sequence-number*] • **no host** *IPv4-address* • **no** *IPv4-address*/*prefix-len* • **no** *IPv4-address network-wildcard* **Example:** `switch(config-ipaddr-ogroup)# no host 10.99.32.6`	Removes an entry in the object group. For each entry that you want to remove from the object group, use the **no** form of the **host** command.
Step 5	(Optional) **show object-group** *name* **Example:** `switch(config-ipaddr-ogroup)# show object-group ipv4-addr-group-13`	Displays the object group configuration.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-ipaddr-ogroup)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Creating and Changing an IPv6 Address Object Group

You can create and change an IPv6 address group object.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**object-group ipv6 address** *name*
	Example:
	`switch(config)# object-group ipv6 address ipv6-addr-group-A7`
	`switch(config-ipv6addr-ogroup)#`
		Creates the IPv6 address object group and enters IPv6 address object-group configuration mode.

Step 3	Enter one of the following commands: • [*sequence-number*] **host** *IPv6-address* • [*sequence-number*] *IPv6-address*/*prefix-len* • [*sequence-number*] *IPv6-address network-wildcard*	Creates an entry in the object group. For each entry that you want to create, use the **host** command and specify a single host, or omit the **host** command to specify a network of hosts. You can specify a prefix length for an IPv6 object group, which matches only on the first contiguous bits, or you can specify a wildcard

	Command or Action	Purpose

	Example:
	`switch(config-ipv6addr-ogroup)# host 2001:db8:0:3ab0::1`
	Example:
	`switch(config-ipv6addr-ogroup)# 10 1::1 2::2`
		mask, which matches on any bit in the address. IPv6 wildcard masks are supported for Cisco Nexus 9200, 9300-EX, and 9300-FX/FX2/FXP switches and the Cisco Nexus 9364C switch.

Step 4	Enter one of the following commands: • **no** *sequence-number* • **no host** *IPv6-address* • **no** *IPv6-address*/*prefix-len* • **no** *IPv6-address network-wildcard* **Example:** `switch(config-ipv6addr-ogroup)# no host 2001:db8:0:3ab0::1`	Removes an entry from the object group. For each entry that you want to remove from the object group, use the **no** form of the **host** command.
Step 5	(Optional) **show object-group** *name* **Example:** `switch(config-ipv6addr-ogroup)# show object-group ipv6-addr-group-A7`	Displays the object group configuration.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-ipv6addr-ogroup)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Creating and Changing a Protocol Port Object Group

You can create and change a protocol port object group.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**object-group ip port** *name*
	Example:
	`switch(config)# object-group ip port NYC-datacenter-ports`
	`switch(config-port-ogroup)#`
		Creates the protocol port object group and enters port object-group configuration mode.

	Command or Action	Purpose
Step 3	[*sequence-number*] *operator port-number* [*port-number*] **Example:** `switch(config-port-ogroup)# eq 80`	Creates an entry in the object group. For each entry that you want to create, use one of the following operator commands: • **eq**—Matches only the port number that you specify. • **gt**—Matches port numbers that are greater than (and not equal to) the port number that you specify. • **lt**—Matches port numbers that are less than (and not equal to) the port number that you specify. • **neq**—Matches all port numbers except for the port number that you specify. • **range**—Matches the range of port numbers between and including the two port numbers that you specify. **Note** The **range** command is the only operator command that requires two *port-number* arguments.
Step 4	**no** {*sequence-number* | *operator port-number* [*port-number*]} **Example:** `switch(config-port-ogroup)# no eq 80`	Removes an entry from the object group. For each entry that you want to remove, use the **no** form of the applicable operator command.
Step 5	(Optional) **show object-group** *name* **Example:** `switch(config-port-ogroup)# show object-group NYC-datacenter-ports`	Displays the object group configuration.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-port-ogroup)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Removing an Object Group

You can remove an IPv4 address object group, an IPv6 address object group, or a protocol port object group.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**no object-group** {**ip address** | **ipv6 address** | **ip port**} *name* **Example:** `switch(config)# no object-group ip address ipv4-addr-group-A7`	Removes the specified object group.
Step 3	(Optional) **show object-group** **Example:** `switch(config)# show object-group`	Displays all object groups. The removed object group should not appear.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Verifying the Object-Group Configuration

To display object-group configuration information, enter one of the following commands:

Command	Purpose
show object-group	Displays the object-group configuration.
show {**ip** | **ipv6**} **access-lists** *name* [**expanded**]	Displays expanded statistics for the ACL configuration.
show running-config aclmgr	Displays the ACL configuration, including object groups.

Configuring Time-Ranges

Session Manager Support for Time-Ranges

Session Manager supports the configuration of time ranges. This feature allows you to create a configuration session and verify your time-range configuration changes prior to committing them to the running configuration. For more information about Session Manager, see the *Cisco Nexus 9000 Series NX-OS System Management Configuration Guide*.

Creating a Time-Range

You can create a time range on the device and add rules to it.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**time-range** *name*
	Example:
	`switch(config)# time-range workday-daytime`
	`switch(config-time-range)#`
		Creates the time range and enters time-range configuration mode.

Step 3	(Optional) [*sequence-number*] **periodic** *weekday time* **to** [*weekday*] *time* **Example:** `switch(config-time-range)# periodic monday 00:00:00 to friday 23:59:59`	Creates a periodic rule that is in effect for one or more contiguous days between and including the specified start and end days and times.
Step 4	(Optional) [*sequence-number*] **periodic** *list-of-weekdays time* **to** *time* **Example:** `switch(config-time-range)# periodic weekdays 06:00:00 to 20:00:00`	Creates a periodic rule that is in effect on the days specified by the *list-of-weekdays* argument between and including the specified start and end times. The following keywords are also valid values for the *list-of-weekdays* argument: • **daily** —All days of the week. • **weekdays** —Monday through Friday. • **weekend** —Saturday through Sunday.
Step 5	(Optional) [*sequence-number*] **absolute start** *time date* [**end** *time date*] **Example:** `switch(config-time-range)# absolute start 1:00 15 march 2013`	Creates an absolute rule that is in effect beginning at the time and date specified after the **start** keyword. If you omit the **end** keyword, the rule is always in effect after the start time and date have passed.
Step 6	(Optional) [*sequence-number*] **absolute** [start *time date*] **end** *time date* **Example:** `switch(config-time-range)# absolute end 23:59:59 31 may 2013`	Creates an absolute rule that is in effect until the time and date specified after the **end** keyword. If you omit the **start** keyword, the rule is always in effect until the end time and date have passed.

	Command or Action	Purpose
Step 7	(Optional) **show time-range** *name* **Example:** `switch(config-time-range)# show time-range workday-daytime`	Displays the time-range configuration.
Step 8	(Optional) **copy running-config startup-config** **Example:** `switch(config-time-range)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Changing a Time-Range

You can add and remove rules in an existing time range. You cannot change existing rules. Instead, to change a rule, you can remove it and recreate it with the desired changes.

If you need to add more rules between existing rules than the current sequence numbering allows, you can use the **resequence** command to reassign sequence numbers.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**time-range** *name*
	Example:
	`switch(config)# time-range workday-daytime`
	`switch(config-time-range)#`
		Enters time-range configuration mode for the specified time range.

Step 3	(Optional) [*sequence-number*] **periodic** *weekday time* **to** [*weekday*] *time* **Example:** `switch(config-time-range)# periodic monday 00:00:00 to friday 23:59:59`	Creates a periodic rule that is in effect for one or more contiguous days between and including the specified start and end days and times.
Step 4	(Optional) [*sequence-number*] **periodic** *list-of-weekdays time* **to** *time* **Example:** `switch(config-time-range)# 100 periodic weekdays 05:00:00 to 22:00:00`	Creates a periodic rule that is in effect on the days specified by the *list-of-weekdays* argument between and including the specified start and end times. The following keywords are also valid values for the *list-of-weekdays* argument: • **daily** —All days of the week.

	Command or Action	Purpose
		• **weekdays** —Monday through Friday. • **weekend** —Saturday through Sunday.
Step 5	(Optional) [*sequence-number*] **absolute start** *time date* [**end** *time date*] **Example:** `switch(config-time-range)# absolute start 1:00 15 march 2013`	Creates an absolute rule that is in effect beginning at the time and date specified after the **start** keyword. If you omit the **end** keyword, the rule is always in effect after the start time and date have passed.
Step 6	(Optional) [*sequence-number*] **absolute** [**start** *time date*] **end** *time date* **Example:** `switch(config-time-range)# absolute end 23:59:59 31 may 2013`	Creates an absolute rule that is in effect until the time and date specified after the **end** keyword. If you omit the **start** keyword, the rule is always in effect until the end time and date have passed.
Step 7	(Optional) **no** {*sequence-number* | **periodic** *arguments . . .* | **absolute** *arguments. . .*} **Example:** `switch(config-time-range)# no 80`	Removes the specified rule from the time range.
Step 8	(Optional) **show time-range** *name* **Example:** `switch(config-time-range)# show time-range workday-daytime`	Displays the time-range configuration.
Step 9	(Optional) **copy running-config startup-config** **Example:** `switch(config-time-range)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Related Topics

Changing Sequence Numbers in a Time Range, on page 76

Removing a Time-Range

You can remove a time range from the device.

Before you begin

Ensure that you know whether the time range is used in any ACL rules. The device allows you to remove time ranges that are used in ACL rules. Removing a time range that is in use in an ACL rule does not affect the configuration of interfaces where you have applied the ACL. Instead, the device considers the ACL rule using the removed time range to be empty.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**no time-range** *name* **Example:** `switch(config)# no time-range daily-workhours`	Removes the time range that you specified by name.
Step 3	(Optional) **show time-range** **Example:** `switch(config-time-range)# show time-range`	Displays the configuration for all time ranges. The removed time range should not appear.
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Changing Sequence Numbers in a Time Range

You can change all the sequence numbers assigned to rules in a time range.

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**resequence time-range** *name starting-sequence-number increment*
	Example:
	`switch(config)# resequence time-range daily-workhours 100 10`
	`switch(config)#`
		Assigns sequence numbers to the rules contained in the time range, where the first rule receives the starting sequence number that you specify. Each subsequent rule receives a number larger than the preceding rule. The difference in numbers is determined by the increment that you specify.

Step 3	(Optional) **show time-range** *name* **Example:**	Displays the time-range configuration.

	Command or Action	Purpose
	`switch(config)# show time-range daily-workhours`	
Step 4	(Optional) **copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Verifying the Time-Range Configuration

To display time-range configuration information, perform one of the following tasks.

Command	Purpose
show time-range	Displays the time-range configuration.
show running-config aclmgr	Displays ACL configuration, including all time ranges.

Additional References for IP ACLs

Related Documents

Related Topic	Document Title
TAP aggregation	Configuring TAP Aggregation and MPLS Stripping