Configuring 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.

Note The Cisco NX-OS release that is running on a managed device may not support all documented features or settings. For the latest feature information and caveats, see the documentation and release notes for your platform and software release.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your software release. To find information about the features documented in this module, and to see a list

of the releases in which each feature is supported, see the "New and Changed Information"chapter or the Feature History table in this chapter.

Information 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:

FCoE ACLs
The device applies Fibre Channel over Ethernet (FCoE) ACLs only to Fibre Channel traffic. For more information on FCoE, see the *Cisco NX-OS FCoE Configuration Guide for Cisco Nexus 7000 and Cisco MDS 9500*.

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 by default; however, you can configure Layer 2 interfaces to apply MAC ACLs to all traffic.

Security-group ACLs (SGACLs)
The device applies SGACLs to traffic tagged by Cisco TrustSec.

IP and MAC ACLs have the following types of applications:

Port ACL
Filters Layer 2 traffic

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 • IPv6 ACLs • MAC ACLs
Router ACL	• VLAN interfaces • Physical Layer 3 interfaces • Layer 3 Ethernet subinterfaces • Layer 3 Ethernet port-channel interfaces • Layer 3 Ethernet port-channel subinterfaces • Tunnels • Management interfaces **Note** You must enable VLAN interfaces globally before you can configure a VLAN interface. For more information, see the *Cisco Nexus 7000 Series NX-OS Interfaces Configuration Guide*.	• 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
VTY ACL	• VTYs	• IPv4 ACLs • IPv6 ACLs

Related Topics

MAC Packet Classification
Information About MAC ACLs
Information About VLAN ACLs
SGACLs and SGTs

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. SGACL
6. Egress VTY ACL
7. Egress router ACL
8. 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.

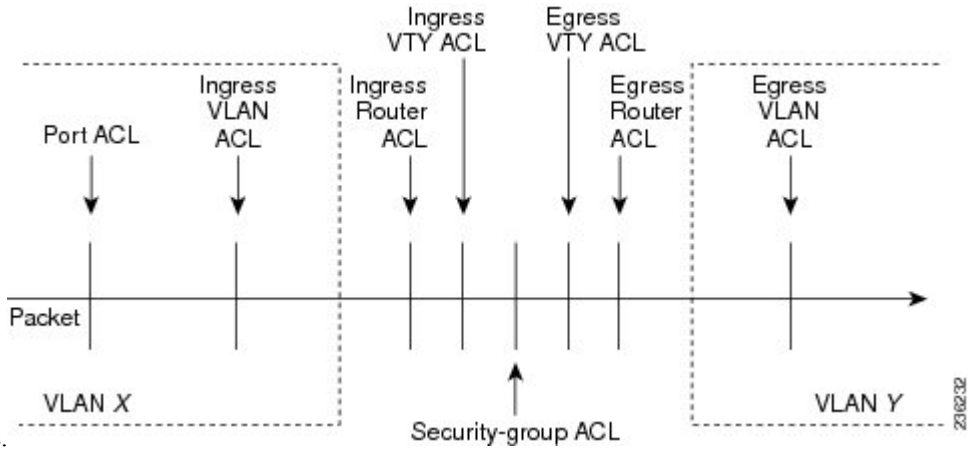
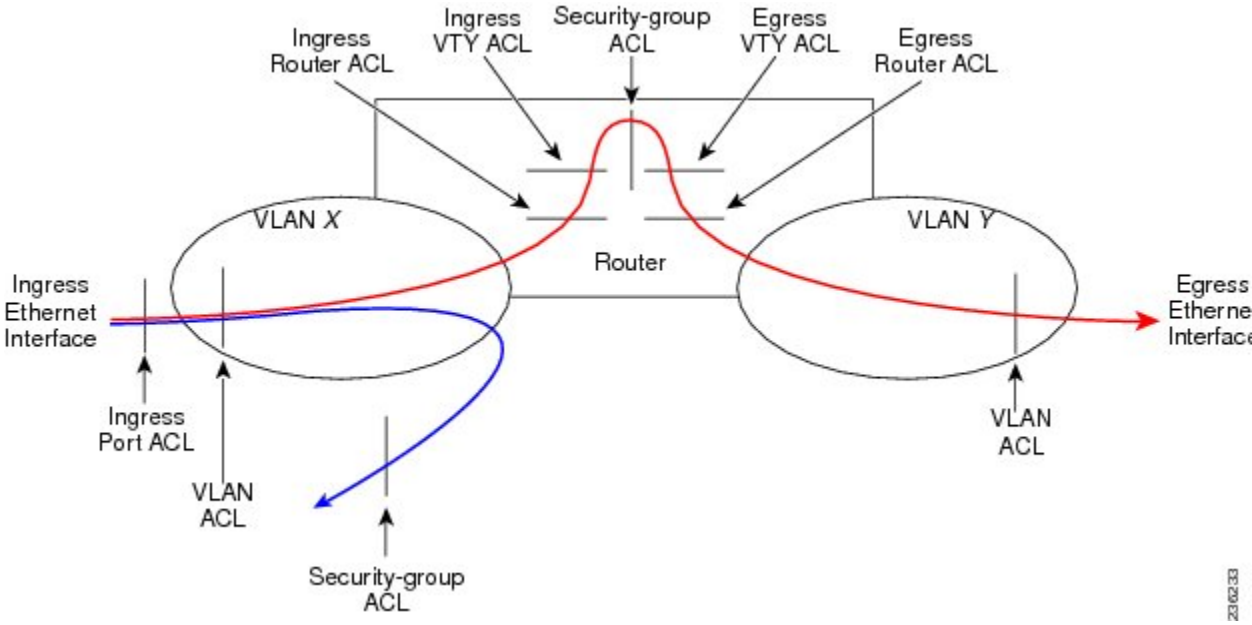


Figure 2: ACLs and Packet Flow

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.



Related Topics

SGACLs and SGTs

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. For information about every option, see the applicable **permit** and **deny** commands in the *Cisco Nexus 7000 Series NX-OS Security Command Reference*.

Protocols for IP 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. For example, you can use 115 to specify Layer 2 Tunneling Protocol (L2TP) traffic.

For a list of the protocols that each type of ACL supports by name, see the applicable **permit** and **deny** commands in the *Cisco Nexus 7000 Series NX-OS Security Command Reference*.

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, IPv6, or MAC ACLs. For information about specifying the source and destination, see the applicable **permit** and **deny** commands in the *Cisco Nexus 7000 Series NX-OS Security Command Reference*.

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 rules:

```
permit icmp any any nd-na
permit icmp any any nd-ns
permit icmp any any router-advertisement
permit icmp any any router-solicitation
deny ipv6 any any
```

Unless you configure an IPv6 ACL with a rule that denies ICMPv6 neighbor discovery messages, the first four rules ensure that the device permits neighbor discovery advertisement and solicitation messages. The fifth rule ensures that the device denies unmatched IPv6 traffic.

Note If you explicitly configure an IPv6 ACL with a **deny ipv6 any any** rule, the implicit permit rules can never permit traffic. If you explicitly configure a **deny ipv6 any any** rule but want to permit ICMPv6 neighbor discovery messages, explicitly configure a rule for all five implicit IPv6 ACL rules.

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
- Authentication Header Protocol
- Enhanced Interior Gateway Routing Protocol (EIGRP)
- Encapsulating Security Payload
- General Routing Encapsulation (GRE)
- KA9Q NOS-compatible IP-over-IP tunneling
- Open Shortest Path First (OSPF)
- Payload Compression Protocol
- Protocol-independent multicast (PIM)
- 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
  - Authentication Header Protocol
  - Encapsulating Security Payload
  - Payload Compression Protocol
  - Stream Control Transmission Protocol (SCTP)
  - SCTP, TCP, and UDP ports
  - ICMP types and codes
  - IGMP types
  - Flow label
  - 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
  - VLAN ID
  - Class of Service (CoS)

For information about all filtering options available in rules, see the applicable **permit** and **deny** commands in the *Cisco Nexus 7000 Series NX-OS Security Command Reference*.

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. The device stores operator-operand couples in registers called logical operator units (LOUs). Cisco Nexus 7000 Series devices support 104 LOUs.

The LOU usage for each type of operator is as follows:

eq
Is never stored in an LOU

gt
Uses 1/2 LOU

lt
Uses 1/2 LOU

neq
Uses 1/2 LOU

range
Uses 1 LOU

The following guidelines determine when the devices store operator-operand couples in LOUs:

- If the operator or operand differs from other operator-operand couples that are used in other rules, the couple is stored in an LOU.

  For example, the operator-operand couples "gt 10" and "gt 11" would be stored separately in half an LOU each. The couples "gt 10" and "lt 10" would also be stored separately.

- Whether the operator-operand couple is applied to a source port or a destination port in the rule affects LOU usage. Identical couples are stored separately when one of the identical couples is applied to a source port and the other couple is applied to a destination port.

  For example, if a rule applies the operator-operand couple "gt 10" to a source port and another rule applies a "gt 10" couple to a destination port, both couples would also be stored in half an LOU, resulting in the use of one whole LOU. Any additional rules using a "gt 10" couple would not result in further LOU usage.

Logging

You can enable the device to create an informational log message for packets that match a rule. The log message contains the following information about the packet:

- Protocol
- Status of whether the packet is a TCP, UDP, or ICMP packet, or if the packet is only a numbered packet.
- Source and destination address
- Source and destination port numbers, if applicable

Access Lists with Fragment Control

As non-initial fragments contain only Layer 3 information, these access-list entries containing only Layer 3 information, can now be applied to non-initial fragments also. The fragment has all the information the system requires to filter, so the access-list entry is applied to the fragments of a packet.

This feature adds the optional **fragments** keyword to the following IP access list commands: **deny (IPv4)**, **permit (IPv4)**, **deny (IPv6)**, **permit (IPv6)**. By specifying the **fragments** keyword in an access-list entry, that particular access-list entry applies only to non-initial fragments of packets; the fragment is either permitted or denied accordingly.

The behavior of access-list entries regarding the presence or absence of the **fragments** keyword can be summarized as follows:

If the Access-List Entry has...	Then...

...no **fragments** keyword and all of the access-list entry information matches	For an access-list entry containing only Layer 3 information:
	• The entry is applied to non-fragmented packets, initial fragments, and non-initial fragments.
	
	For an access-list entry containing Layer 3 and Layer 4 information:
	• The entry is applied to non-fragmented packets and initial fragments.
	  • If the entry matches and is a permit statement, the packet or fragment is permitted.
	  • If the entry matches and is a deny statement, the packet or fragment is denied.
	• The entry is also applied to non-initial fragments in the following manner. Because non-initial fragments contain only Layer 3 information, only the Layer 3 portion of an access-list entry can be applied. If the Layer 3 portion of the access-list entry matches, and
	  • If the entry is a **permit** statement, the non-initial fragment is permitted.
	  • If the entry is a **deny** statement, the next access-list entry is processed.
	
	Note The deny statements are handled differently for non-initial fragments versus non-fragmented or initial fragments.

...the **fragments** keyword and all of the access-list entry information matches	The access-list entry is applied only to non-initial fragments. **Note** The **fragments** keyword cannot be configured for an access-list entry that contains any Layer 4 information.

You should not add the **fragments** keyword to every access-list entry, because the first fragment of the IP packet is considered a non-fragment and is treated independently of the subsequent fragments. Because an initial fragment will not match an access list permit or deny entry that contains the **fragments** keyword, the packet is compared to the next access list entry until it is either permitted or denied by an access list entry that does not contain the **fragments** keyword. Therefore, you may need two access list entries for every deny entry. The first deny entry of the pair will not include the **fragments** keyword, and applies to the initial fragment. The second deny entry of the pair will include the **fragments** keyword and applies to the subsequent

fragments. In the cases where there are multiple deny access list entries for the same host but with different Layer 4 ports, a single deny access-list entry with the **fragments** keyword for that host is all that has to be added. Thus all the fragments of a packet are handled in the same manner by the access list.

Packet fragments of IP datagrams are considered individual packets and each fragment counts individually as a packet in access-list accounting and access-list violation counts.

Note The **fragments** keyword cannot solve all cases involving access lists and IP fragments.

Note Within the scope of ACL processing, Layer 3 information refers to fields located within the IPv4 header; for example, source, destination, protocol. Layer 4 information refers to other data contained beyond the IPv4 header; for example, source and destination ports for TCP or UDP, flags for TCP, type and code for ICMP.

Policy Routing

Fragmentation and the fragment control feature affect policy routing if the policy routing is based on the **match ip address** command and the access list had entries that match on Layer 4 through Layer 7 information. It is possible that noninitial fragments pass the access list and are policy routed, even if the first fragment was not policy routed or the reverse.

By using the **fragments** keyword in access-list entries as described earlier, a better match between the action taken for initial and noninitial fragments can be made and it is more likely policy routing will occur as intended.

Note Filtering with L3 and L4 information can lead to routing or packet loss issues in the network. Perform any one of the following to prevent these issues:

- Modify the route map to allow required L3 information for appropriate UDP ports.
- Check the MTU by verifying the path from source to destination to ensure that the packet is not fragmented.

ACL Capture

You can configure ACL capture in order to selectively monitor traffic on an interface or VLAN.

When you enable the capture option for an ACL rule, packets that match this rule are either forwarded or dropped based on the specified **permit** or **deny** action and may also be copied to an alternate destination port for further analysis.

An ACL rule with the capture option can be applied as follows:

- On a VLAN
- In the ingress direction on all interfaces
- In the egress direction on all Layer 3 interfaces

ACL capture can be used in a variety of scenarios. For example, ACL capture can use ACL rules to identify packets belonging to a tunnel and to send a copy (or capture) of the tunnel packets to a specific destination. ACL capture can also be used to monitor all HTTP traffic on a particular VLAN.

Finally, you can also configure the capture session for the whole ACL rather than configuring it per ACL rule. This configuration applies the capture session to all of the ACL rules.

Related Topics

Enabling or Disabling ACL Capture, on page 32
Configuring an ACL Capture Session, on page 33
Applying an ACL with Capture Session ACEs to an Interface, on page 34
Applying a Whole ACL Capture Session to an Interface, on page 35

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. From Cisco NX-OS Release 8.4(2), the ACL time range name has a maximum length of 256 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, 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.

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 42
Implicit Rules for IP and MAC ACLs, on page 6

Atomic ACL Updates

An atomic ACL update is a hardware operation where both the existing ACL and the updated ACL are programmed in TCAM memory. This is the default mode of operation. The benefit of this update method is that ACL changes are not service impacting. When you make a change to the ACL, the current ACL is already programmed in TCAM. The Cisco Nexus 7000 Series device will then take the current ACL and merge it with the changes to produce ACL prime. ACL prime will also be programmed into TCAM. The Cisco Nexus 7000 Series device will then change the pointer so that ACL prime is associated with the interface. The final step is to delete the old ACL from TCAM. Functionally this means that you can never exceed 50 percent of ACL TCAM resources if you want to use atomic ACL updates. If you exceed 50 percent of ACL resources while atomic ACL update is active, the “ERROR: Tcam will be over used, please turn off atomic update” message is received and the new ACL changes are not applied.

Nonatomic ACL updates are required if you are using more than 50 percent of the ACL TCAM. When this mode is active, the Cisco Nexus 7000 Series device will remove the old ACL from TCAM and replace it with ACL prime as quickly as possible. This allows you to use up to 100 percent of your ACL TCAM but has the disadvantage that it will cause a temporary interruption in service because packets that were permitted by the old ACL will be dropped until ACL prime can be successfully programed into the ACL TCAM.

By default, when a supervisor module of a Cisco Nexus 7000 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.

Note The **hardware access-list update** command is available in the default VDC only but applies to all VDCs.

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
```

Planning for Atomic ACL Updates

To adequately plan for Atomic ACL updates you need to be aware of how many ACE (Access Control Elements) you are using on all of your ACLs on each module. You also need to know how many ACEs your TCAM can support. You can find out your current usage with the **show hardware access-list resource utilization mod** *module-number* command.

```
show hardware access-list resource
utilization mod 3
 INSTANCE 0x0
 -------------
 ACL Hardware Resource Utilization (Mod 3)
 ------------------------------------------
               Used  Free  Percent
```

```
                          Utilization
-----------------------------------------------------
Tcam 0, Bank 0  1   16383  0.01
Tcam 0, Bank 1  2   16382  0.01
Tcam 1, Bank 0  7   16377  0.04
Tcam 1, Bank 1  138 16246  0.84
```

For M-series modules, the ACL TCAM is spread across four banks. On non-XL modules, each bank has 16,000 entries for a total of 64K entries. On XL modules each bank has 32,000 entries for a total of 128,000 entries. Under normal circumstances, a single ACL will only use the resources of a single TCAM bank. In order to enable a single ACL to use resources from all of the banks you need to enable bank pooling with the **hardware access-list resource pooling module** *mod-numbe*r command.

You can verify that bank pooling is enabled with the **show hardware access-list resource pooling** command.

ACL TCAM Bank Mapping

ACL ternary control address memory (TCAM) bank mapping allows TCAM banks to accommodate more feature combinations in a more predictable manner. Features are preclassified into feature groups, which are further predefined into feature classes according to which features are allowed to coexist in a TCAM bank. For example, a port ACL (port ACL) feature and a Layer 2 NetFlow feature are defined as one feature class. These classes are allocated to specific banks. An error message appears if you enable or disable a feature class that is not supported on a specific TCAM bank.

ACL TCAM bank mapping allows you to configure a set of features at the same time and reduces multiple results that can accumulate when feature combinations that cannot coexist are configured on the same TCAM banks. By using this feature, you can optimize space and maximize the utilization of TCAM banks.

Beginning with Cisco NX-OS Release 6.2(10), you can issue the **show hardware access-list** {**input** | **output**} {**interface** | **vlan**} **feature-combo** *features* command to display the bank mapping matrix.

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.

Virtualization Support for IP ACLs

The following information applies to IP and MAC ACLs used in virtual device contexts (VDCs):

- ACLs are unique per VDC. You cannot use an ACL that you created in one VDC in a different VDC.
- Because ACLs are not shared by VDCs, you can reuse ACL names in different VDCs.
- The device does not limit ACLs or rules on a per-VDC basis.
- Configuring atomic ACL updates must be performed in the default VDC but applies to all VDCs.

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:

- When an access control list (ACL) is applied at the ingress of the original packet, it gets the destination index of the actual egress port and has no knowledge of the Encapsulated Remote Switched Port Analyzer (ERSPAN) session's point of egress at that moment. Because the packet does not go through the ACL engine after rewrite, it cannot be matched on ERSPAN packets.
- 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 is especially useful for ACLs that include more than about 1000 rules. For more information about Session Manager, see the *Cisco Nexus 7000 Series NX-OS System Management Configuration Guide*.
- In most cases, 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 a large number of 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 (additional 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.

  Note Prior to Cisco NX-OS Release 4.2(3), ACL logging does not support ACL processing that occurs on the supervisor module.
- When you apply an ACL that uses time ranges, the device updates the ACL entries on the affected I/O modules whenever a time range 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 7000 Series NX-OS Interfaces Configuration Guide*
- The maximum number of supported IP ACL entries is 64,000 for devices without an XL line card and 128,000 for devices with an XL line card.
- If you try to apply too many ACL entries to a non-XL line card, the configuration is rejected.

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.

- ACLs configured for VTYs do not apply to the mgmt0 interface. Mgmt0 ACLs must be applied specifically to the interface.
- The Cisco Nexus 2000 Series Fabric Extender supports the full range of ingress ACLs that are available on its parent Cisco Nexus 7000 Series device. For more information about the Fabric Extender, see the *Configuring the Cisco Nexus 2000 Series Fabric Extender*.
- ACL policies are not supported on the Fabric Extender fabric port channel.
- ACL capture is a hardware-assisted feature and is not supported for the management interface or for control packets originating in the supervisor. It is also not supported for software ACLs such as SNMP community ACLs and VTY ACLs.
- Enabling ACL capture disables ACL logging for all VDCs and the rate limiter for ACL logging.
- Port channels and supervisor in-band ports are not supported as a destination for ACL capture.
- ACL capture session destination interfaces do not support ingress forwarding and ingress MAC learning. If a destination interface is configured with these options, the monitor keeps the ACL capture session down. Use the **show monitor session all** command to see if ingress forwarding and MAC learning are enabled.

Note You can use the **switchport monitor** command to disable ingress forwarding and MAC learning on the interface.

- The source port of the packet and the ACL capture destination port cannot be part of the same packet replication ASIC. If both ports belong to the same ASIC, the packet is not captured. The **show monitor session** command lists all the ports that are attached to the same ASIC as the ACL capture destination port.
- Only one ACL capture session can be active at any given time in the system across VDCs.
- If you configure an ACL capture monitor session before configuring the **hardware access-list capture** command, you must shut down the monitor session and bring it back up in order to start the session.
- When you apply an undefined ACL to an interface, the system treats the ACL as empty and permits all traffic.
- An IPv6 atomic policy update can be disruptive. It may cause disruption when there is an addition, deletion, or modification of an IPv6 source or destination address:
  - Modifying the Layer 4 fields of the IPv6 ACE is not disruptive.
  - Adding an IPv6 address may not always be disruptive, however, it can cause disruption in some cases.
  - There may be disruption if you change the prefix length of an existing entry or add/delete the entry with a new prefix length.

Note An IPv6 atomic policy update is not disruptive for F3 and M3 Series modules.

- Resource pooling and ACL TCAM bank mapping cannot be enabled at the same time.
- You cannot configure the **mac packet-classify** command on shared interfaces.
- M1 Series Modules
  - M1 Series modules support ACL capture.
  - FCoE ACLs are not supported for M1 Series modules.
  - For M1 Series modules, the **mac packet-classify** command enables a MAC ACL for port and VLAN policies.
  - M1 Series modules do not support IP ACLs on port ACL and VACL policies, when the MAC packet classification feature is enabled on the interface. Before you upgrade to Cisco NX-OS Release 6.x or later versions, you need to disable the MAC packet classification feature on M1 Series module and verify whether all the existing functionalities work.
  - M1 Series modules support WCCP.
- M2 Series Modules
  - M2 Series modules support ACL capture.
  - FCoE ACLs are not supported for M2 Series modules.
  - For M2 Series modules, the **mac packet-classify** command enables a MAC ACL for port and VLAN policies.
  - M2 Series modules do not support IP ACLs on port ACL and VACL policies, when the MAC packet classification feature is enabled on the interface. Before you upgrade to Cisco NX-OS Release 6.x or later versions, you need to disable the MAC packet classification feature on M2 Series module and verify whether all the existing functionalities work.
  - M2 Series modules support WCCP.
- F1 Series Modules
  - Each forwarding engine on an F1 Series module supports 1000 ingress ACL entries, with 984 entries available for user configuration. The total number of IP ACL entries for the F1 Series modules is from 1000 to 16,000, depending on which forwarding engines the policies are applied.
  - Each of the 16 forwarding engines in an F1 Series module supports up to 250 IPv6 addresses across multiple ACLs.
  - Each port ACL can support up to four different Layer 4 operations for F1 Series modules.
  - F1 Series modules do not support router ACLs.
  - F1 Series modules do not support ACL logging.
  - F1 Series modules do not support bank chaining.

- F1 Series modules do not support ACL capture.
- FCoE ACLs are supported only for F1 Series modules.
- F1 Series modules do not support WCCP.
- F1 Series modules do not support ACL TCAM bank mapping.
- For F1 Series module proxy-forwarded traffic, ACL classification is matched against the Layer 3 protocols shown in the following table:

Table 2: Protocol Number and Associated Layer 3 Protocol

Protocol Number	Layer 3 Protocol
1	ICMP
2	IGMP
4	IPv4 Encapsulation
6	TCP
17	UDP

Note Layer 3 protocols not listed in the table are classified as protocol number 4 (IPv4 Encapsulation).

- F2 Series Modules
  - Each of the 12 forwarding engines in an F2 Series module has 16,000 total TCAM entries, equally split across two banks. 168 default entries are reserved. Each forwarding engine also has 512 IPv6 compression TCAM entries.
  - F2 Series modules do not support ACL capture.
  - For F2 Series modules, the **log** option in egress ACLs is not supported for multicast packets.
  - If an F2 Series module is shared among different VDCs, any egress ACL that is configured on one VDC is pushed to the other VDCs.
  - F2 Series modules do not support egress WCCP on SVI.
  - For F2 Series modules, the **mac packet-classify** command enables a MAC ACL for port policies but an IPv4 or IPv6 ACL for VLAN policies.
- Two banks can be chained within the same TCAM. However, you cannot chain banks across multiple TCAMs.
- The bank chaining and bank mapping features cannot co-exist.
- You cannot configure port ACL features such as PACL, L2 QOS, and L2 Netflow when you enable the VLAN-VLAN mode for configuring the flexible ACL TCAM bank chaining feature.
- The flexible ACL TCAM bank chaining feature is not supported on the F2 Series modules.

- Enabling the flexible ACL TCAM bank chaining feature on all the modules is not supported.
- F3 Series Module
  - The forwarding engines in an F3 Series module has 16,000 total TCAM entries that are equally split across two banks.
  - F3 Series modules supports ACL capture.
  - F3 Series modules supports FCoE ACLs.
  - For F3 Series modules, the log option in egress ACLs is not supported for multicast packets.
  - If an F3 Series module is shared among different VDCs, any egress ACL that is configured on one VDC is pushed to the other VDCs.
  - For F3 Series modules, the **mac packet-classify** command enables a MAC ACL for port policies but an IPv4 or IPv6 ACL for VLAN policies.
  - Two banks can be chained within the same TCAM. However, you cannot chain banks across multiple TCAMs.
  - The bank chaining and bank mapping features cannot co-exist.
  - You cannot configure port ACL features such as PACL, L2 QOS, and L2 Netflow when you enable the VLAN-VLAN mode for configuring the flexible ACL TCAM bank chaining feature.
  - The flexible ACL TCAM bank chaining feature is supported only on the F3 Series modules. Enabling the flexible ACL TCAM bank chaining feature on all the modules is not supported.

ACLs on VTY lines have the following guidelines and limitations:

- ACLs applied on a VTY line in egress direction filter traffic without any issues. However, ACLs applied on a VTY line in ingress direction will not filter management traffic. For example, FTP, TFTP, or SFP traffic in the return direction, that is, if the FTP connection is initiated from a switch to an external server, ingress ACL on a VTY line will not be used, if ACLs are configured to block or permit this return traffic. Therefore, ACLs should be applied in the egress direction on VTY lines to block the FTP, TFTP, or SCP traffic from the switch.
- It is recommended to use ACLs on management interface as well to secure access to the switch from secured and permitted sources.

Default Settings for IP ACLs

This table lists the default settings for IP ACL parameters.

Table 3: Default IP ACL Parameters

Parameters	Default
IP ACLs	No IP ACLs exist by default
ACL rules	Implicit rules apply to all ACLs
ACL capture	Disabled

Parameters	Default
Object groups	No object groups exist by default
Time ranges	No time ranges exist by default
ACL TCAM bank mapping	Disabled

Related Topics

Implicit Rules for IP and MAC ACLs, on page 6

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 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. For more information about Session Manager, see the *Cisco Nexus 7000 Series NX-OS System Management Configuration Guide*.

SUMMARY STEPS

1. **configure terminal**
2. Enter one of the following commands:
   - **ip access-list** *name*
   - **ipv6 access-list** *name*
3. (Optional) **fragments** {**permit-all** | **deny-all**}
4. [*sequence-number*] {**permit** | **deny**} *protocol source destination*
5. (Optional) **statistics per-entry**
6. (Optional) Enter one of the following commands:
   - **show ip access-lists** *name*
   - **show ipv6 access-lists** *name*
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`switch# configure terminal` `switch(config)#`	

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)#`
		Creates the IP ACL and enters IP ACL configuration mode. The *name* argument can be up to 64 characters. From Cisco NX-OS Release 8.4(2), the name argument can be upto 256 characters.

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 destination* **Example:** `switch(config-acl)# permit ip 192.168.2.0/24 any`	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 more information, see the *Cisco Nexus 7000 Series NX-OS Security Command Reference*.
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.
Step 6	(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 7	(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. For more information about Session Manager, see the *Cisco Nexus 7000 Series NX-OS System Management Configuration Guide*.

SUMMARY STEPS

1. **configure terminal**
2. Enter one of the following commands:
   - **ip access-list** *name*
   - **ipv6 access-list** *name*
3. (Optional) [*sequence-number*] {**permit** | **deny**} *protocol source destination*
4. (Optional) [**no**] **fragments** {**permit-all** | **deny-all**}
5. (Optional) **no** {*sequence-number* | {**permit** | **deny**} *protocol source destination*}
6. (Optional) [**no**] **statistics per-entry**
7. (Optional) Enter one of the following commands:
   - **show ip access-lists** *name*
   - **show ipv6 access-lists** *name*
8. (Optional) **copy running-config startup-config**

DETAILED STEPS

	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. For more information, see the *Cisco*

	Command or Action	Purpose
		Nexus 7000 Series NX-OS System Management Configuration Guide.
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. For more information, see the *Cisco Nexus 7000 Series NX-OS Security Command Reference*.
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* • **show ipv6 access-lists** *name* **Example:** `switch(config-acl)# show ip access-lists acl-01`	Displays the IP ACL configuration.
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 27

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.

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

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. For more information about Session Manager, see the *Cisco Nexus 7000 Series NX-OS System Management Configuration Guide*.

SUMMARY STEPS

1. **configure terminal**
2. {**ip** | **ipv6**} **access-list** *name*
3. {**permit** | **deny**} *protocol source destination* [**log**] [**time-range** *time*]
4. **exit**
5. **line vty**
6. {**ip** | **ipv6**} **access-class** *name* {**in** | **out**}
7. (Optional) **show** {**ip** | **ipv6**} **access-lists**
8. (Optional) **copy running-config startup-config**

DETAILED STEPS

	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.

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.

	Command or Action	Purpose
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. For more information about Session Manager, see the *Cisco Nexus 7000 Series NX-OS System Management Configuration Guide*.

SUMMARY STEPS

1. **configure terminal**
2. **resequence** {**ip** | **ipv6**} **access-list** *name starting-sequence-number increment*
3. (Optional) **show ip access-lists** *name*
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	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* **Example:** `switch(config)# resequence access-list ip acl-01 100 10`	Assigns sequence numbers to the rules contained in the ACL, 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. 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:**	Displays the IP ACL configuration.

	Command or Action	Purpose
	`switch(config)# show ip access-lists acl-01`	
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.

SUMMARY STEPS

1. **configure terminal**
2. Enter one of the following commands:
   - **no ip access-list** *name*
   - **no ipv6 access-list** *name*
3. (Optional) Enter one of the following commands:
   - **show ip access-lists** *name* **summary**
   - **show ipv6 access-lists** *name* **summary**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	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.

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 and subinterfaces
- VLAN interfaces
- Tunnels
- Management interfaces

ACLs applied to these interface types are considered router ACLs.

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.

SUMMARY STEPS

1. switch# **configure terminal**
2. Enter one of the following commands:
   - switch(config)# **interface ethernet** *slot*/*port*[**.** *number*]
   - switch(config)# **interface port-channel** *channel-number*[**.** *number*]
   - switch(config)# **interface tunnel** *tunnel-number*
   - switch(config)# **interface vlan** *vlan-ID*
   - switch(config)# **interface mgmt** *port*
3. Enter one of the following commands:
   - switch(config-if)# **ip access-group** *access-list* {**in** | **out**}
   - switch(config-if)# **ipv6 traffic-filter** *access-list* {**in** | **out**}
4. (Optional) switch(config-if)# **show running-config aclmgr**

5. (Optional) switch(config-if)# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# **configure terminal**	Enters global configuration mode.
Step 2	Enter one of the following commands: • switch(config)# **interface ethernet** *slot*/*port*[**.** *number*] • switch(config)# **interface port-channel** *channel-number*[**.** *number*] • switch(config)# **interface tunnel** *tunnel-number* • switch(config)# **interface vlan** *vlan-ID* • switch(config)# **interface mgmt** *port*	Enters configuration mode for the interface type that you specified.
Step 3	Enter one of the following commands: • switch(config-if)# **ip access-group** *access-list* {**in** | **out**} • switch(config-if)# **ipv6 traffic-filter** *access-list* {**in** | **out**}	Applies an IPv4 or IPv6 ACL to the Layer 3 interface for traffic flowing in the direction specified. You can apply one router ACL per direction.
Step 4	(Optional) switch(config-if)# **show running-config aclmgr**	Displays the ACL configuration.
Step 5	(Optional) switch(config-if)# **copy running-config startup-config**	Copies the running configuration to the startup configuration.

Related Topics

Creating an IP ACL, on page 22

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.

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.

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.

SUMMARY STEPS

1. **configure terminal**
2. Enter one of the following commands:

- **interface ethernet** *slot/port*
- **interface port-channel** *channel-number*

3. Enter one of the following commands:
   - **ip port access-group** *access-list* **in**
   - **ipv6 port traffic-filter** *access-list* **in**
4. (Optional) **show running-config aclmgr**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	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:** `switch(config-if)# ip port access-group acl-l2-marketing-group in`	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.
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 22
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

Enabling or Disabling ACL Capture

Beginning with Cisco NX-OS Release 5.2, you can enable or disable ACL capture in the default VDC.

Before you begin

Ensure that you are in the default VDC.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **hardware access-list capture**
3. (Optional) **show hardware access-list status module** *slot*
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

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

Step 2	[**no**] **hardware access-list capture** **Example:** `switch(config)# hardware access-list capture`	Enables or disables ACL capture on all VDCs. **Note** When you enable ACL capture, a warning message appears to inform you that ACL logging is being disabled for all VDCs. When you disable ACL capture, ACL logging is enabled.
Step 3	(Optional) **show hardware access-list status module** *slot* **Example:** `switch(config)# show hardware access-list status module 2`	Displays the ACL capture 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.

Related Topics

ACL Capture, on page 11

Configuring an ACL Capture Session

Beginning with Cisco NX-OS Release 5.2, you can configure an ACL capture session.

Before you begin

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **monitor session** *session* **type acl-capture**
3. **destination interface** *interface slot/port*
4. **no shut**
5. **exit**
6. (Optional) **show ip access-lists capture session** *session*
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

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

Step 2	**monitor session** *session* **type acl-capture**
	Example:
	`switch(config)# monitor session 2 type acl-capture`
	`switch(config-acl-capture)#`
		Configures an ACL capture session. The range for the *session* argument is from 1 to 48.

Step 3	**destination interface** *interface slot/port*
	Example:
	`switch(config-acl-capture)# destination interface ethernet 2/2`
	`switch#`
		Configures a destination for ACL capture packets.
		Note Only the physical interface can be used for the destination. Port-channel interfaces and supervisor in-band ports are not supported.
		Note You can enter this command multiple times to add multiple destinations.

Step 4	**no shut** **Example:** `switch(config-acl-capture)# no shut`	Brings the ACL capture session administratively up. **Note** The session becomes operationally up only after the monitor confirms that the ACL capture has been enabled in the default VDC.

	Command or Action	Purpose

Step 5	**exit**
	Example:
	`switch(config-acl-capture)# exit`
	`switch(config)#`
		Updates the monitor configuration and exits the ACL capture configuration mode.

Step 6	(Optional) **show ip access-lists capture session** *session* **Example:** `switch(config)# show ip access-lists capture session 2`	Displays the ACL capture session configuration.
Step 7	(Optional) **copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Applying an ACL with Capture Session ACEs to an Interface

You can enable a capture session for an ACL's access control entries (ACEs) and then apply the ACL to an interface.

Before you begin

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **ip access-list** *name*
3. **permit** *protocol source destination* **capture session** *session*
4. **exit**
5. **interface** *interface slot*/*port*
6. **ip access-group** *name* **in**
7. **no shut**
8. (Optional) **show running-config aclmgr**
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	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:**	Creates an access list.

	Command or Action	Purpose
	`switch(config)# ip access-list acl1` `switch(config-acl)#`	
Step 3	**permit** *protocol source destination* **capture session** *session* **Example:** `switch(config-acl)# permit tcp any any capture session 2`	Enables a capture session for the ACL's ACEs. The range for the *session* argument is from 1 to 16.

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

Step 5	**interface** *interface slot/port*
	Example:
	`switch(config)# interface ethernet 7/1`
	`switch(config-if)#`
		Specifies a port and enters interface configuration mode.

Step 6	**ip access-group** *name* **in** **Example:** `switch(config-if)# ip access-group acl1 in`	Applies an ACL with capture session ACEs to the interface.
Step 7	**no shut** **Example:** `switch(config-if)# no shut`	Brings the interface administratively up.
Step 8	(Optional) **show running-config aclmgr** **Example:** `switch(config-if)# show running-config aclmgr`	Displays the ACL configuration and the interfaces to which ACLs are applied.
Step 9	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Applying a Whole ACL Capture Session to an Interface

You can enable a capture session for the whole ACL and then apply the ACL to an interface.

Before you begin

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

SUMMARY STEPS

1. **configure terminal**
2. **ip access-list** *name*

3. **capture session** *session*
4. **exit**
5. **interface** *interface slot*/*port*
6. **ip access-group** *name* **in**
7. **no shut**
8. (Optional) **show running-config aclmgr**
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	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 acl1`
	`switch(config-acl)#`
		Creates an access list.

Step 3	**capture session** *session* **Example:** `switch(config-acl)# capture session 2`	Enables a capture session for the whole ACL. The range for the *session* argument is from 1 to 16.

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

Step 5	**interface** *interface slot*/*port*
	Example:
	`switch(config)# interface ethernet 7/1`
	`switch(config-if)#`
		Specifies a port and enters interface configuration mode.

Step 6	**ip access-group** *name* **in** **Example:** `switch(config-if)# ip access-group acl1 in`	Applies an ACL with the capture session configuration to the interface.
Step 7	**no shut** **Example:** `switch(config-if)# no shut`	Brings the interface administratively up.
Step 8	(Optional) **show running-config aclmgr** **Example:** `switch(config-if)# show running-config aclmgr`	Displays the ACL configuration and the interfaces to which ACLs are applied.

	Command or Action	Purpose
Step 9	(Optional) **copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Configuring Scale ACL

Scale ACL is introduced in Cisco NX-OS Release 8.4(2) and it is supported on M3 modules. This feature support is added only for RACL policies with object-group. This feature helps you to implement large scale configuration of ACL with support of object-group configuration. Both IPv4 and IPv6 RACL is supported. Scale ACL is configured with the key word, **compress**.

SUMMARY STEPS

1. **configure terminal**
2. **[no] hardware access-list compress module** *module-number*
3. **interface** *interface-name number*
4. **[no] ip access-group access-list** {**in** | **out** } **compress**
5. **end**
6. **show ip access-list** *name* **compress**
7. **show hardware access-list compress**
8. **show system internal access-list resource presearch-utilization**
9. **show system internal access-list interface** *interface-name number* **input presearch-entries**
10. **show system internal access-list interface** *interface-name number* **input statistics**

DETAILED STEPS

	Command or Action	Purpose

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

Step 2	**[no] hardware access-list compress module** *module-number* **Example:** `switch(config)# hardware access-list compress module 2`	Configures Scale ACL on a module. Reload the module after configuring the scale ACL.
Step 3	**interface** *interface-name number* **Example:** `switch(config)# interface port-channel 1`	Enters interface configuration mode.

	Command or Action	Purpose
Step 4	**[no] ip access-group access-list** {**in** | **out** } **compress** **Example:** `switch(config-if)# ip access-group test in compress`	Configures access list on an interface and applies the scale ACL. You can apply access-list only when the "statistics per-entry" is enabled.
Step 5	**end** **Example:** `switch(config-if)# end`	Exits interface configuration mode and enters privileged EXEC mode.
Step 6	**show ip access-list** *name* **compress** **Example:** `switch# show ip access-list test compress`	Displays the scale ACL statistics.
Step 7	**show hardware access-list compress** **Example:** `switch# show hardware access-list compress`	Displays the M3 modules on which the compression is enabled.
Step 8	**show system internal access-list resource presearch-utilization** **Example:** `switch# show system internal access-list resource presearch-utilization`	Displays the pre-search TCAM utilization information.
Step 9	**show system internal access-list interface** *interface-name number* **input presearch-entries** **Example:** `switch# show system internal access-list interface port-channel 1 input presearch-entries`	Displays information on the IP programmed in pre-search TCAM for a policy.
Step 10	**show system internal access-list interface** *interface-name number* **input statistics** **Example:** `switch# show system internal access-list interface port-channel 1 input statistics`	Displays information on the TCAM programming for a policy.

Configuration Examples for Scale ACL

The following example shows the M3 module on which the compression is enabled:

```
switch# show hardware access-list compress
+------------+---------------+----------------+
| MODULE_NUM | CONFIG_STATUS | RUNTIME_STATUS |
+------------+---------------+----------------+
| 1 |               No |        Inactive |
+------------+---------------+----------------+
```

The following example displays the ACL statistics:

```
switch# show ip access-lists test compress
IP access list test
statistics per-entry
10 permit ip addrgroup G1 addrgroup G2 fragments log [match=1833318182]
20 permit ip addrgroup G1 addrgroup G3 dscp af21 log [match=1833318182]
30 permit ip addrgroup G1 addrgroup G3 precedence critical log [match=1833318182]
40 permit ip addrgroup G1 addrgroup G2 dscp af11 log [match=1833318181]
50 permit ip addrgroup G1 addrgroup G2 dscp af12 log [match=0]
60 permit ip addrgroup G1 addrgroup G2 dscp af13 log [match=0]
70 permit ip addrgroup G1 addrgroup G2 dscp af22 log [match=0]
80 permit ip addrgroup G1 addrgroup G2 dscp af23 packet-length neq 9010 log [match=0]
```

The following example displays the pre-search TCAM utilization information.

```
switch# show system internal access-list resource presearch-utilization
INSTANCE 0x0
-------------
Presearch-SA ACL Hardware Resource Utilization (Mod 1)
-------------------------------------------
Used Free Percent
Utilization
-----------------------------------------------------
Tcam 0, Bank 0 0 16384 0.00
Tcam 0, Bank 1 0 16384 0.00
Tcam 1, Bank 0 0 16384 0.00
Tcam 1, Bank 1 80 16304 0.49
Presearch-DA ACL Hardware Resource Utilization (Mod 1)
-------------------------------------------
Used Free Percent
Utilization
-----------------------------------------------------
Tcam 0, Bank 0 0 16384 0.00
Tcam 0, Bank 1 0 16384 0.00
Tcam 1, Bank 0 0 16384 0.00
Tcam 1, Bank 1 67 16317 0.41
```

The following example shows how to verify the IP programmed in pre-search TCAM for a policy:

```
switch# show system internal access-list interface port-channel 1 input presearch-entries

INSTANCE 0x0
---------------
Tcam 0 resource usage:
----------------------
Presearch-SA
------------
Label_a = 0x2
Bank 0
------
IPv4 Class
Policies: RACL(test_acl)
Entries:
[Index] Entry [Result]
---------------------
[0000:257042:0000] 1.1.1.1/32 [0x2000000]
[0001:256882:0001] 1.1.1.2/32 [0x2000000]
[0002:2568c2:0002] 1.1.1.3/32 [0x2000000]
[0003:256942:0003] 5.5.5.37/32 [0x2000000]
[0004:256a02:0004] 6.6.6.40/32 [0x2000000]
[0005:256e82:0005] 10.10.10.10/32 [0x2000000]
[0006:256902:0006] 20.20.20.20/32 [0x1000000]
[0007:2569c2:0007] 23.23.23.23/32 [0x1000000]
[0008:256c42:0008] 192.168.1.1/32 [0x3000000]
[0009:256c82:0009] 192.168.1.2/32 [0x3000000]
[000a:256cc2:000a] 192.168.1.3/32 [0x3000000]
```

```
[000b:257502:000b] 192.168.1.4/32 [0x3000000]
Bank 1
------
IPv4 Class
Policies: RACL(test_acl)
Entries:
[Index] Entry [Result]
---------------------
[0000:256842:0000] 1.1.1.1/32 [0x2000000]
[0001:257082:0001] 1.1.1.2/32 [0x2000000]
[0002:2570c2:0002] 1.1.1.3/32 [0x2000000]
[0003:257142:0003] 5.5.5.37/32 [0x2000000]
[0004:257202:0004] 6.6.6.40/32 [0x2000000]
[0005:257682:0005] 10.10.10.10/32 [0x2000000]
[0006:257102:0006] 20.20.20.20/32 [0x1000000]
[0007:2571c2:0007] 23.23.23.23/32 [0x1000000]
[0008:257442:0008] 192.168.1.1/32 [0x3000000]
[0009:257482:0009] 192.168.1.2/32 [0x3000000]
[000a:2574c2:000a] 192.168.1.3/32 [0x3000000]
[000b:256d02:000b] 192.168.1.4/32 [0x3000000]
```

The following example shows how to verify the main TCAM programming for a policy:

```
switch# show system internal access-list interface port-channel 1 input statistics
INSTANCE 0x0
---------------
Tcam 0 resource usage:
----------------------
Label_a = 0x1
Bank 0
------
IPv4 Class
Policies: RACL(test_acl)
Netflow profile: 0
Netflow deny profile: 0
Entries:
[Index] Entry [Stats]
---------------------
[0014:436a2:0000] prec 2 objgrp-permit-routed ip 0x1000000/0x7000000 0x3000000/0x3000000
[3545]
[0015:43722:0001] prec 2 objgrp-permit-routed ip 0x2000000/0x7000000 0x1000000/0x3000000
[0]
[0016:437a2:0002] prec 2 objgrp-permit-routed ip 0x3000000/0x7000000 0x2000000/0x3000000
[0]
[0017:3c222:0003] prec 2 objgrp-permit-routed ip 0x4000000/0x7000000 0x4000000/0x4000000
[0]
[0018:43222:0004] prec 2 deny-routed ip 0x0/0x0 0x0/0x0 [0]
```

Verifying the IP ACL Configuration

To display IP ACL configuration information, perform one of the following tasks. For detailed information about the fields in the output from these commands, see the *Cisco Nexus 7000 Series NX-OS Security Command Reference*.

Command	Purpose
show hardware access-list status module *slot*	Displays the ACL capture configuration.

Command	Purpose
show ip access-lists [**capture session** *session*]	Displays the IPv4 ACL configuration.
show ipv6 access-lists [**capture session** *session*]	Displays the IPv6 ACL configuration.
show system internal access-list feature bank-class map {**ingress** | **egress**} [**module** *module*]	Displays the feature group and class combination tables.
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** Beginning with Cisco NX-OS Release 5.2, 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 aclmgr [**all**]	Displays the ACL startup configuration. **Note** Beginning with Cisco NX-OS Release 5.2, 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.

Note If TCP permits or deny in the ACL, the **ip access-list detailed** command doesn't identify established conditions. The traffic is counted for ACL if other condition matches though a successful TCP connection is not established. Detailed log entries will not be displayed (this is only for the ACL logging and does not include or affect the actual ACL forwarding decision).

Monitoring and Clearing IP ACL Statistics

To monitor or clear IP ACL statistics, use one of the commands in this table. For detailed information about these commands, see the *Cisco Nexus 7000 Series NX-OS Security Command Reference*.

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 enable ACL capture in the default VDC and configure a destination for ACL capture packets:

```
hardware access-list capture
  monitor session 1 type acl-capture
  destination interface ethernet 2/1
  no shut
  exit
  show ip access-lists capture session 1
```

The following example shows how to enable a capture session for an ACL's access control entries (ACEs) and then apply the ACL to an interface:

```
ip access-list acl1
  permit tcp any any capture session 1
  exit
  interface ethernet 1/11
  ip access-group acl1 in
  no shut
  show running-config aclmgr
```

The following example shows how to apply an ACL with capture session access control entries (ACEs) to a VLAN:

```
vlan access-map acl-vlan-first
  match ip address acl-ipv4-first
  match mac address acl-mac-first
  action foward
  statistics per-entry
  vlan filter acl-vlan-first vlan-list 1
  show running-config vlan 1
```

The following example shows how to enable a capture session for the whole ACL and then apply the ACL to an interface:

```
ip access-list acl2
  capture session 2
  exit
  interface ethernet 7/1
  ip access-group acl1 in
  no shut
  show running-config aclmgr
```

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 7000 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.

SUMMARY STEPS

1. **configure terminal**
2. **object-group ip address** *name*
3. Enter one of the following commands:
   - [*sequence-number*] **host** *IPv4-address*
   - [*sequence-number*] *IPv4-address network-wildcard*
   - [*sequence-number*] *IPv4-address*/*prefix-len*
4. Enter one of the following commands:
   - **no** [*sequence-number*]
   - **no host** *IPv4-address*
   - **no** *IPv4-address network-wildcard*
   - **no** *IPv4-address*/*prefix-len*
5. (Optional) **show object-group** *name*
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	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 network-wildcard* • [*sequence-number*] *IPv4-address*/*prefix-len* **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.
Step 4	Enter one of the following commands: • **no** [*sequence-number*] • **no host** *IPv4-address* • **no** *IPv4-address network-wildcard* • **no** *IPv4-address*/*prefix-len* **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.

	Command or Action	Purpose
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.

SUMMARY STEPS

1. **config t**
2. **object-group ipv6 address** *name*
3. Enter one of the following commands:
   - [*sequence-number*] **host** *IPv6-address*
   - [*sequence-number*] *IPv6-address*/*prefix-len*
4. Enter one of the following commands:
   - **no** *sequence-number*
   - **no host** *IPv6-address*
   - **no** *IPv6-address*/*prefix-len*
5. (Optional) **show object-group** *name*
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**config t**
	Example:
	`switch# config t`
	`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.

	Command or Action	Purpose
Step 3	Enter one of the following commands: • [*sequence-number*] **host** *IPv6-address* • [*sequence-number*] *IPv6-address*/*prefix-len* **Example:** `switch(config-ipv6addr-ogroup)# host 2001:db8:0:3ab0::1`	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 specify a network of hosts.
Step 4	Enter one of the following commands: • **no** *sequence-number* • **no host** *IPv6-address* • **no** *IPv6-address*/*prefix-len* **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.

SUMMARY STEPS

1. **configure terminal**
2. **object-group ip port** *name*
3. [*sequence-number*] *operator port-number* [*port-number*]
4. **no** {*sequence-number* | *operator port-number* [*port-number*]}
5. (Optional) **show object-group** *name*
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`switch# configure terminal` `switch(config)#`	

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.

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 the port number that you specify only. • **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 number 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.

SUMMARY STEPS

1. **configure terminal**
2. **no object-group** {**ip address** | **ipv6 address** | **ip port**} *name*
3. (Optional) **show object-group**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	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 object group that you specified.
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, perform one of the following tasks:

Command	Purpose
show object-group	Displays the object-group configuration.
show running-config aclmgr	Displays ACL configuration, including object groups.

For detailed information about the fields in the output from these commands, see the *Cisco Nexus 7000 Series NX-OS Security Command Reference*.

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 7000 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.

Before you begin

Ensure that you are in the correct VDC (or use the **switchto vdc** command). Because ACL names can be repeated in different VDCs, we recommend that you confirm which VDC you are working in.

SUMMARY STEPS

1. **configure terminal**
2. **time-range** *name*
3. (Optional) [*sequence-number*] **periodic** *weekday time* **to** [*weekday*] *time*
4. (Optional) [*sequence-number*] **periodic** *list-of-weekdays time* **to** *time*
5. (Optional) [*sequence-number*] **absolute start** *time date* [**end** *time date*]
6. (Optional) [*sequence-number*] **absolute** [start *time date*] **end** *time date*
7. (Optional) **show time-range** *name*
8. (Optional) **copy running-config startup-config**

DETAILED STEPS

	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:**	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.

	Command or Action	Purpose
	`switch(config-time-range)# periodic monday 00:00:00 to friday 23:59:59`	
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 2008`	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 december 2008`	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) **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.

Before you begin

Ensure that you are in the correct VDC (or use the **switchto vdc** command). Because ACL names can be repeated in different VDCs, we recommend that you confirm which VDC you are working in.

SUMMARY STEPS

1. **configure terminal**
2. **time-range** *name*
3. (Optional) [*sequence-number*] **periodic** *weekday time* **to** [*weekday*] *time*
4. (Optional) [*sequence-number*] **periodic** *list-of-weekdays time* **to** *time*
5. (Optional) [*sequence-number*] **absolute start** *time date* [**end** *time date*]
6. (Optional) [*sequence-number*] **absolute** [**start** *time date*] **end** *time date*
7. (Optional) **no** {*sequence-number* | **periodic** *arguments . . .* | **absolute** *arguments. . .*}
8. (Optional) **show time-range** *name*
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	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. • **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 2008`	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*	Creates an absolute rule that is in effect until the time and date specified after the **end** keyword. If you omit the **start**

	Command or Action	Purpose
	Example: `switch(config-time-range)# absolute end 23:59:59 31 december 2008`	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 53

Removing a Time Range

You can remove a time range from the device.

Before you begin

Ensure that you are in the correct VDC (or use the **switchto vdc** command). Because ACL names can be repeated in different VDCs, we recommend that you confirm which VDC you are working in.

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.

SUMMARY STEPS

1. **configure terminal**
2. **no time-range** *name*
3. (Optional) **show time-range**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	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.

Before you begin

Ensure that you are in the correct VDC (or use the **switchto vdc** command). Because ACL names can be repeated in different VDCs, we recommend that you confirm which VDC you are working in.

SUMMARY STEPS

1. **configure terminal**
2. **resequence time-range** *name starting-sequence-number increment*
3. (Optional) **show time-range** *name*
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	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:**	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

	Command or Action	Purpose
	`switch(config)# resequence time-range daily-workhours 100 10` `switch(config)#`	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:** `switch(config)# show time-range daily-workhours`	Displays the time-range 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.

Verifying the Time-Range Configuration

To display time-range configuration information, perform one of the following tasks. For detailed information about the fields in the output from these commands, see the *Cisco Nexus 7000 Series NX-OS Security Command Reference*.

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
IP ACL commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco Nexus 7000 Series NX-OS Security Command Reference*
Object group commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco Nexus 7000 Series NX-OS Security Command Reference*
Time range commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco Nexus 7000 Series NX-OS Security Command Reference*
SNMP	*Cisco Nexus 7000 Series NX-OS System Management Configuration Guide*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

Feature History for IP ACLs

This table lists the release history for this feature.

Table 4: Feature History for IP ACLs

Feature Name	Releases	Feature Information
Configuring ACLs over M3 modules	7.3(0)DX(1)	Support for M3 modules is introduced.
Flexible ACL TCAM Bank Chaining	7.3(0)D1(1)	Added the support for the flexible ACL TCAM bank chaining feature.
ACL TCAM bank mapping	6.2(10)	Added a command to display the bank-mapping matrix.
IP ACLs	6.2(2)	Added support for ACL TCAM bank mapping.
IP ACLs	6.1(1)	Updated for M2 Series modules.
IP ACLs	6.0(1)	Updated for F2 Series modules.
FCoE ACLs	5.2(1)	Added support for FCoE ACLs on F1 Series modules.
IP ACLs	5.2(1)	Added support for ACL capture on M1 Series modules.
IP ACLs	5.2(1)	Changed the **show running-config aclmgr** and **show startup-config aclmgr** commands to display only the user-configured ACLs (and not also the default CoPP-configured ACLs) in the running and startup configurations.
VTY ACLs	5.1(1)	Added support to control access to traffic received over a VTY line.
IP ACLs	5.0(2)	Added support for up to 128K ACL entries when using an XL line card, provided a scalable services license is installed.
ACL logging	4.2(3)	Added support for logging of packets sent to the supervisor module for ACL processing.

Feature Name	Releases	Feature Information
IP ACLs	4.2(1)	Added support for MAC packet classification on Layer 2 interfaces.