



Security Configuration Guide: Access Control Lists, Cisco IOS XE Release 3S (ASR 900 Series)

Americas Headquarters

Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
<http://www.cisco.com>
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883



CONTENTS

CHAPTER 1

IP Access List Overview 1

- Information About IP Access Lists 1
 - Benefits of IP Access Lists 1
 - Restrictions for Access Control Lists 2
 - Border Routers and Firewall Routers Should Use Access Lists 2
 - Definition of an Access List 3
 - Access List Rules 3
 - Helpful Hints for Creating IP Access Lists 4
 - Named or Numbered Access Lists 5
 - Standard or Extended Access Lists 5
 - IP Packet Fields You Can Filter to Control Access 6
 - Wildcard Mask for Addresses in an Access List 6
 - Access List Sequence Numbers 7
 - Access List Logging 7
 - Alternative to Access List Logging 8
 - Additional IP Access List Features 8
 - Where to Apply an Access List 8
- Additional References 9

CHAPTER 2

Creating an IP Access List and Applying It to an Interface 11

- Finding Feature Information 11
- Restrictions for Creating an IP Access List and Applying It to an Interface 12
- Information About Creating an IP Access List and Applying It to an Interface 12
 - Helpful Hints for Creating IP Access Lists 12
 - Access List Remarks 13
 - Additional IP Access List Features 13

How to Create an IP Access List and Apply It to an Interface	13
Creating a Standard Access List to Filter on Source Address	14
Creating a Named Access List to Filter on Source Address	14
Creating a Numbered Access List to Filter on Source Address	16
Creating an Extended Access List	17
Creating a Named Extended Access List	18
Creating a Numbered Extended Access List	20
Applying an Access List to a Physical Interface	22
Configuration Examples for Creating an IP Access List and Applying It to a Physical Interface	23
Example: Filtering on Host Source Address	23
Example: Filtering on Subnet Source Address	23
Example: Filtering on Source and Destination Addresses and IP Protocols	24
Example: Filtering on Source Addresses Using a Numbered Access List	24
Example: Preventing Telnet Access to a Subnet	24
Example: Filtering on TCP and ICMP Using Port Numbers	25
Example: Allowing SMTP E-mail and Established TCP Connections	25
Example: Preventing Access to the Web by Filtering on Port Name	25
Example: Filtering on Source Address and Logging the Packets	26
Example: Limiting Debug Output	26
Apply Access Control List on Management Interface	27
Restrictions	27
Configure ACL on Management Interface	27
Verification of ACL Configuration on Management Interface	27
Additional References Creating an IP Access List and Applying It to an Interface	28
<hr/>	
CHAPTER 3	Creating an IP Access List to Filter IP Options, TCP Flags, Noncontiguous Ports
	31
Finding Feature Information	31
Prerequisites for Creating an IP Access List to Filter IP Options TCP Flags Noncontiguous Ports	31
Information About Creating an IP Access List to Filter IP Options, TCP Flags, Noncontiguous Ports	32
IP Options	32
Benefits of Filtering IP Options	32
Benefits of Filtering on TCP Flags	32
TCP Flags	33

Benefits of Using the Named ACL Support for Noncontiguous Ports on an Access Control Entry Feature	33
How to Create an IP Access List to Filter IP Options TCP Flags Noncontiguous Ports	34
Filtering Packets That Contain IP Options	34
What to Do Next	35
Filtering Packets That Contain TCP Flags	35
Configuring an Access Control Entry with Noncontiguous Ports	38
Consolidating Access List Entries with Noncontiguous Ports into One Access List Entry	39
What To Do Next	41
Configuration Examples for Filtering IP Options, TCP Flags, Noncontiguous Ports	41
Example: Filtering Packets That Contain IP Options	41
Example: Filtering Packets That Contain TCP Flags	41
Example: Creating an Access List Entry with Noncontiguous Ports	42
Example: Consolidating Some Existing Access List Entries into One Access List Entry with Noncontiguous Ports	42
Additional References	43

CHAPTER 4
IPv6 Access Control Lists 45

Finding Feature Information	45
Restrictions for IPv6 Access Control Lists	45
Restrictions - Dual Stack (IPv4 + IPv6 ACL on an Interface)	46
Information About IPv6 Access Control Lists	47
Access Control Lists for IPv6 Traffic Filtering	47
IPv6 Packet Inspection	47
Access Class Filtering in IPv6	47
How to Configure IPv6 Access Control Lists	47
Configuring IPv6 Traffic Filtering	47
Creating and Configuring an IPv6 ACL for Traffic Filtering	47
Applying the IPv6 ACL to an Interface	49
Controlling Access to a vty	50
Creating an IPv6 ACL to Provide Access Class Filtering	50
Applying an IPv6 ACL to the Virtual Terminal Line	51
Configuration Examples for IPv6 Access Control Lists	52
Example: Verifying IPv6 ACL Configuration	52

Example: Creating and Applying an IPv6 ACL	52
Example: Controlling Access to a vty	53
Use Case or Deployment Scenarios	53
Additional References	54

CHAPTER 5

MAC Access Control Lists	55
Finding Feature Information	55
Prerequisites for MAC Access Control Lists	55
Restrictions for MAC Access Control Lists	55
Information About MAC Access Control Lists	56
MAC Access Control Lists	56
How to Configure MAC Access Control Lists	56
Configuring ACL	56
Applying ACL on Cross-Connect	57
Verifying MAC Access Control Lists	60
Configuration Examples for MAC Access Control Lists	60
MAC ACL Configuration	60
Additional References for MAC Access Control Lists	60



CHAPTER 1

IP Access List Overview

Access control lists (ACLs) perform packet filtering to control which packets move through a network and to where. The packet filtering provides security by helping to limit the network traffic, restrict the access of users and devices to a network, and prevent the traffic from leaving a network. IP access lists reduce the chance of spoofing and denial-of-service attacks and allow dynamic, temporary user-access through a firewall.

IP access lists can also be used for purposes other than security, such as bandwidth control, restrict the content of routing updates, redistribute routes, trigger dial-on-demand (DDR) calls, limit debug output, and identify or classify traffic for quality of service (QoS) features. This module provides an overview of IP access lists.

- [Information About IP Access Lists, on page 1](#)
- [Additional References, on page 9](#)

Information About IP Access Lists

Benefits of IP Access Lists

Access control lists (ACLs) perform packet filtering to control the flow of packets through a network. Packet filtering can restrict the access of users and devices to a network, providing a measure of security. Access lists can save network resources by reducing traffic. The benefits of using access lists are as follows:

- Authenticate incoming rsh and rcp requests—Access lists can simplify the identification of local users, remote hosts, and remote users in an authentication database that is configured to control access to a device. The authentication database enables Cisco software to receive incoming remote shell (rsh) and remote copy (rcp) protocol requests.
- Block unwanted traffic or users—Access lists can filter incoming or outgoing packets on an interface, thereby controlling access to a network based on source addresses, destination addresses, or user authentication. You can also use access lists to determine the types of traffic that are forwarded or blocked at device interfaces. For example, you can use access lists to permit e-mail traffic to be routed through a network and to block all Telnet traffic from entering the network.
- Control access to vty—Access lists on an inbound vty (Telnet) can control who can access the lines to a device. Access lists on an outbound vty can control the destinations that the lines from a device can reach.
- Identify or classify traffic for QoS features—Access lists provide congestion avoidance by setting the IP precedence for Weighted Random Early Detection (WRED) and committed access rate (CAR). Access

lists also provide congestion management for class-based weighted fair queueing (CBWFQ), priority queueing, and custom queueing.

- Limit debug command output—Access lists can limit debug output based on an IP address or a protocol.
- Provide bandwidth control—Access lists on a slow link can prevent excess traffic on a network.
- Provide NAT control—Access lists can control which addresses are translated by Network Address Translation (NAT).
- Reduce the chance of DoS attacks—Access lists reduce the chance of denial-of-service (DoS) attacks. Specify IP source addresses to control traffic from hosts, networks, or users from accessing your network. Configure the TCP Intercept feature to can prevent servers from being flooded with requests for connection.
- Restrict the content of routing updates—Access lists can control routing updates that are sent, received, or redistributed in networks.
- Trigger dial-on-demand calls—Access lists can enforce dial and disconnect criteria.

Restrictions for Access Control Lists

- The **deny ip any any** command does not deny the packets that are not fragmented first when it is preceded by the `permit tcp any any port-number` or `permit udp any any port-number` command.

Example:

```
permit tcp any any eq 3000
deny ip any any fragment
```

TCP stream with port number 4000 is not denied for the packets that are not fragmented first. ACE works fine for the first fragment as it has TCP port information.

Border Routers and Firewall Routers Should Use Access Lists

There are many reasons to configure access lists; for example, you can use access lists to restrict contents of routing updates or to provide traffic flow control. One of the most important reasons to configure access lists is to provide a basic level of security for your network by controlling access to it. If you do not configure access lists on your router, all packets passing through the router could be allowed onto all parts of your network.

An access list can allow one host to access a part of your network and prevent another host from accessing the same area. In the figure below, by applying an appropriate access list to the interfaces of the router, Host A is allowed to access the Human Resources network and Host B is prevented from accessing the Human Resources network.

Access lists should be used in firewall routers, which are often positioned between your internal network and an external network such as the Internet. You can also use access lists on a router positioned between two parts of your network, to control traffic entering or exiting a specific part of your internal network.

To provide some security benefits of access lists, you should at least configure access lists on border routers--routers located at the edges of your networks. Such an access list provides a basic buffer from the outside network or from a less controlled area of your own network into a more sensitive area of your network. On these border routers, you should configure access lists for each network protocol configured on the router interfaces. You can configure access lists so that inbound traffic or outbound traffic or both are filtered on an interface.

Access lists are defined on a per-protocol basis. In other words, you should define access lists for every protocol enabled on an interface if you want to control traffic flow for that protocol.

Definition of an Access List

Access control lists (ACLs) perform packet filtering to control the movement of packets through a network. Packet filtering provides security by limiting the access of traffic into a network, restricting user and device access to a network, and preventing traffic from leaving a network. IP access lists reduce the chance of spoofing and denial-of-service attacks, and allow dynamic, temporary user-access through a firewall.

IP access lists can also be used for purposes other than security, such as to control bandwidth, restrict the content of routing updates, redistribute routes, trigger dial-on-demand (DDR) calls, limit debug output, and identify or classify traffic for quality of service (QoS) features.

An access list is a sequential list that consists of at least one **permit** statement and possibly one or more **deny** statements. In the case of IP access lists, these statements can apply to IP addresses, upper-layer IP protocols, or other fields in IP packets.

Access lists are identified and referenced by a name or a number. Access lists act as packet filters, filtering packets based on the criteria defined in each access list.

After you configure an access list, for the access list to take effect, you must either apply the access list to an interface (by using the **ip access-group** command), a vty (by using the **access-class** command), or reference the access list by any command that accepts an access list. Multiple commands can reference the same access list.

In the following configuration, an IP access list named `branchoffices` is configured on Fast Ethernet interface 0/1/0 and applied to incoming packets. Networks other than the ones specified by the source address and mask pair cannot access Fast Ethernet interface 0/1/0. The destinations for packets coming from sources on network 172.16.7.0 are unrestricted. The destination for packets coming from sources on network 172.16.2.0 must be 172.31.5.4.

```
ip access-list extended branchoffices
 10 permit 172.16.7.0 0.0.0.3 any
 20 permit 172.16.2.0 0.0.0.255 host 172.31.5.4
!
interface fastethernet 0/1/0
 ip access-group branchoffices in
```

Access List Rules

The following rules apply to access lists:

- Only one access list per interface, per protocol, and per direction is allowed.
- An access list must contain at least one **permit** statement or all packets are denied entry into the network.
- The order in which access list conditions or match criteria are configured is important. While deciding whether to forward or block a packet, Cisco software tests the packet against each criteria statement in the order in which these statements are created. After a match is found, no more criteria statements are checked. The same **permit** or **deny** statements specified in a different order can result in a packet being passed under one circumstance and denied in another circumstance.
- If an access list is referenced by a name, but the access list does not exist, all packets pass. An interface or command with an empty access list applied to it permits all traffic into the network.

- Standard access lists and extended access lists cannot have the same name.
- Inbound access lists process packets before the packets are routed to an outbound interface. Inbound access lists that have filtering criteria that deny packet access to a network saves the overhead of routing lookup. Packets that are permitted access to a network based on the configured filtering criteria are processed for routing. For inbound access lists, when you configure a **permit** statement, packets are processed after they are received, and when you configure a **deny** statement, packets are discarded.
- Outbound access lists process packets before they leave the device. Incoming packets are routed to the outbound interface and then processed by the outbound access list. For outbound access lists, when you configure a **permit** statement, packets are sent to the output buffer, and when you configure a **deny** statement, packets are discarded.
- An access list can control traffic arriving at a device or leaving a device, but not traffic originating at a device.

Helpful Hints for Creating IP Access Lists

The following tips will help you avoid unintended consequences and help you create more efficient, useful access lists.

- Create the access list before applying it to an interface (or elsewhere), because if you apply a nonexistent access list to an interface and then proceed to configure the access list, the first statement is put into effect, and the implicit **deny** statement that follows could cause you immediate access problems.
- Another reason to configure an access list before applying it is because an interface with an empty access list applied to it permits all traffic.
- All access lists need at least one **permit** statement; otherwise, all packets are denied and no traffic passes.
- Use the statement **permit any any** if you want to allow all other packets not already denied. Using the statement **permit any any** in effect avoids denying all other packets with the implicit deny statement at the end of an access list. Do not make your first access list entry **permit any any** because all traffic will get through; no packets will reach the subsequent testing. In fact, once you specify **permit any any**, all traffic not already denied will get through.
- Although all access lists end with an implicit **deny** statement, we recommend use of an explicit **deny** statement (for example, **deny ip any any**). On most platforms, you can display the count of packets denied by issuing the **show access-list** command, thus finding out more information about who your access list is disallowing. Only packets denied by explicit **deny** statements are counted, which is why the explicit **deny** statement will yield more complete data for you.
- While you are creating an access list or after it is created, you might want to delete an entry.
 - You cannot delete an entry from a numbered access list; trying to do so will delete the entire access list. If you need to delete an entry, you need to delete the entire access list and start over.
 - You can delete an entry from a named access list. Use the **no permit** or **no deny** command to delete the appropriate entry.
- In order to make the purpose of individual statements more scannable and easily understood at a glance, you can write a helpful remark before or after any statement by using the **remark** command.
- If you want to deny access to a particular host or network and find out if someone from that network or host is attempting to gain access, include the **log** keyword with the corresponding **deny** statement so that the packets denied from that source are logged for you.

- This hint applies to the placement of your access list. When trying to save resources, remember that an inbound access list applies the filter conditions before the routing table lookup. An outbound access list applies the filter conditions after the routing table lookup.

Named or Numbered Access Lists

All access lists must be identified by a name or a number. Named access lists are more convenient than numbered access lists because you can specify a meaningful name that is easier to remember and associate with a task. You can reorder statements in or add statements to a named access list.

Named access lists support the following features that are not supported by numbered access lists:

- IP options filtering
- Noncontiguous ports
- TCP flag filtering
- Deleting of entries with the **no permit** or **no deny** command



Note Not all commands that accept a numbered access list will accept a named access list. For example, vty uses only numbered access lists.

Standard or Extended Access Lists

All access lists are either standard or extended access lists. If you only intend to filter on a source address, the simpler standard access list is sufficient. For filtering on anything other than a source address, an extended access list is necessary.

- Named access lists are specified as standard or extended based on the keyword **standard** or **extended** in the **ip access-list** command syntax.
- Numbered access lists are specified as standard or extended based on their number in the **access-list** command syntax. Standard IP access lists are numbered 1 to 99 or 1300 to 1999; extended IP access lists are numbered 100 to 199 or 2000 to 2699. The range of standard IP access lists was initially only 1 to 99, and was subsequently expanded with the range 1300 to 1999 (the intervening numbers were assigned to other protocols). The extended access list range was similarly expanded.



Note Starting from Cisco IOS XE 16.9.4, use the **ip access-list** command to configure object-group based numbered ACL.

Standard Access Lists

Standard IP access lists test only source addresses of packets (except for two exceptions). Because standard access lists test source addresses, they are very efficient at blocking traffic close to a destination. There are two exceptions when the address in a standard access list is not a source address:

- On outbound VTY access lists, when someone is trying to telnet, the address in the access list entry is used as a destination address rather than a source address.
- When filtering routes, you are filtering the network being advertised to you rather than a source address.

Extended Access Lists

Extended access lists are good for blocking traffic anywhere. Extended access lists test source and destination addresses and other IP packet data, such as protocols, TCP or UDP port numbers, type of service (ToS), precedence, TCP flags, and IP options. Extended access lists can also provide capabilities that standard access lists cannot, such as the following:

- Filtering IP Options
- Filtering TCP flags
- Filtering noninitial fragments of packets (see the module “[Refining an IP Access List](#)”)



Note Packets that are subject to an extended access list will not be autonomous switched.

IP Packet Fields You Can Filter to Control Access

You can use an extended access list to filter on any of the following fields in an IP packet. Source address and destination address are the two most frequently specified fields on which to base an access list:

- Source address--Specifies a source address to control packets coming from certain networking devices or hosts.
- Destination address--Specifies a destination address to control packets being sent to certain networking devices or hosts.
- Protocol--Specifies an IP protocol indicated by the keyword **eigrp**, **gre**, **icmp**, **igmp**, **ip**, **ipinip**, **nos**, **ospf**, **tcp**, or **udp**, or indicated by an integer in the range from 0 to 255 (representing an Internet protocol). If you specify a transport layer protocol (**icmp**, **igmp**, **tcp**, or **udp**), the command has a specific syntax.
 - Ports and non-contiguous ports--Specifies TCP or UDP ports by a port name or port number. The port numbers can be noncontiguous port numbers. Port numbers can be useful to filter Telnet traffic or HTTP traffic, for example.
 - TCP flags--Specifies that packets match any flag or all flags set in TCP packets. Filtering on specific TCP flags can help prevent false synchronization packets.
- IP options--Specifies IP options; one reason to filter on IP options is to prevent routers from being saturated with spurious packets containing them.

Wildcard Mask for Addresses in an Access List

Address filtering uses wildcard masking to indicate to the software whether to check or ignore corresponding IP address bits when comparing the address bits in an access list entry to a packet being submitted to the access list. By carefully setting wildcard masks, you can specify one or more IP addresses for permit or deny tests.

Wildcard masking for IP address bits uses the number 1 and the number 0 to specify how the software treats the corresponding IP address bits. A wildcard mask is sometimes referred to as an inverted mask because a 1 and 0 mean the opposite of what they mean in a subnet (network) mask.

- A wildcard mask bit 0 means check the corresponding bit value; they must match.
- A wildcard mask bit 1 means ignore that corresponding bit value; they need not match.

If you do not supply a wildcard mask with a source or destination address in an access list statement, the software assumes an implicit wildcard mask of 0.0.0.0, meaning all values must match.

Unlike subnet masks, which require contiguous bits indicating network and subnet to be ones, wildcard masks allow noncontiguous bits in the mask.

The table below shows examples of IP addresses and masks from an access list, along with the corresponding addresses that are considered a match.

Table 1: Sample IP Addresses, Wildcard Masks, and Match Results

Address	Wildcard Mask	Match Results
0.0.0.0	255.255.255.255	All addresses will match the access list conditions.
172.18.0.0/16	0.0.255.255	Network 172.18.0.0
172.18.5.2/16	0.0.0.0	Only host 172.18.5.2 matches
172.18.8.0	0.0.0.7	Only subnet 172.18.8.0/29 matches
172.18.8.8	0.0.0.7	Only subnet 172.18.8.8/29 matches
172.18.8.15	0.0.0.3	Only subnet 172.18.8.15/30 matches
10.1.2.0	0.0.252.255 (noncontiguous bits in mask)	Matches any even-numbered network in the range of 10.1.2.0 to 10.1.254.0

Access List Sequence Numbers

The ability to apply sequence numbers to IP access list entries simplifies access list changes. Prior to the IP Access List Entry Sequence Numbering feature, there was no way to specify the position of an entry within an access list. If you wanted to insert an entry in the middle of an existing list, all of the entries after the desired position had to be removed, then the new entry was added, and then all the removed entries had to be reentered. This method was cumbersome and error prone.

This feature allows users to add sequence numbers to access list entries and resequence them. When you add a new entry, you specify the sequence number so that it is in a desired position in the access list. If necessary, entries currently in the access list can be resequenced to create room to insert the new entry.

Access List Logging

The Cisco IOS software can provide logging messages about packets permitted or denied by a single standard or extended IP access list entry. That is, any packet that matches the entry will cause an informational logging message about the packet to be sent to the console. The level of messages logged to the console is controlled by the **logging console** global configuration command.

The first packet that triggers the access list entry causes an immediate logging message, and subsequent packets are collected over 5-minute intervals before they are displayed or logged. The logging message includes the access list number, whether the packet was permitted or denied, the source IP address of the packet, and the number of packets from that source permitted or denied in the prior 5-minute interval.

However, you can use the **ip access-list log-update** command to set the number of packets that, when match an access list (and are permitted or denied), cause the system to generate a log message. You might want to do this to receive log messages more frequently than at 5-minute intervals.



Caution If you set the *number-of-matches* argument to 1, a log message is sent right away, rather than caching it; every packet that matches an access list causes a log message. A setting of 1 is not recommended because the volume of log messages could overwhelm the system.

Even if you use the **ip access-list log-update** command, the 5-minute timer remains in effect, so each cache is emptied at the end of 5 minutes, regardless of the count of messages in each cache. Regardless of when the log message is sent, the cache is flushed and the count reset to 0 for that message the same way it is when a threshold is not specified.



Note The logging facility might drop some logging message packets if there are too many to be handled or if there is more than one logging message to be handled in 1 second. This behavior prevents the router from crashing due to too many logging packets. Therefore, the logging facility should not be used as a billing tool or an accurate source of the number of matches to an access list.

Alternative to Access List Logging

Packets matching an entry in an ACL with a log option are process switched. It is not recommended to use the log option on ACLs, but rather use NetFlow export and match on a destination interface of Null0. This is done in the CEF path. The destination interface of Null0 is set for any packet that is dropped by the ACL.

Additional IP Access List Features

Beyond the basic steps to create a standard or extended access list, you can enhance your access lists as mentioned below. Each of these methods is described completely in the module entitled “Refining an Access List.”

- You can impose dates and times when **permit** or **deny** statements in an extended access list are in effect, making your access list more granular and specific to an absolute or periodic time period.
- After you create a named access list, you might want to add entries or change the order of the entries, known as resequencing an access list.
- You can achieve finer granularity when filtering packets by filtering on noninitial fragments of packets.

Where to Apply an Access List

You can apply access lists to the inbound or outbound interfaces of a device. Applying an access list to an inbound interface controls the traffic that enters the interface and applying an access list to an outbound interface controls the traffic that exits the interface.

When software receives a packet at the inbound interface, the software checks the packet against the statements that are configured for the access list. If the access list permits packets, the software processes the packet. Applying access lists to filter incoming packets can save device resources because filtered packets are discarded before entering the device.

Access lists on outbound interfaces filter packets that are transmitted (sent) out of the interface. You can use the TCP Access Control List (ACL) Splitting feature of the Rate-Based Satellite Control Protocol (RBSCP) on the outbound interface to control the type of packets that are subject to TCP acknowledgment (ACK) splitting on an outbound interface.

You can reference an access list by using a **debug** command to limit the amount of debug logs. For example, based on the filtering or matching criteria of the access list, debug logs can be limited to source or destination addresses or protocols.

You can use access lists to control routing updates, dial-on-demand (DDR), and quality of service (QoS) features.

Additional References

Related Documents

Related Topic	Document Title
IP access list commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	Cisco IOS IP Addressing Services Command Reference
Filtering on source address, destination address, or protocol	Creating an IP Access List and Applying It to an Interface” module
Filtering on IP Options, TCP flags, noncontiguous ports, or TTL	Creating an IP Access List to Filter IP Options, TCP Flags, or Noncontiguous Ports module

Standards

Standards & RFCs	Title
None	—

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html



CHAPTER 2

Creating an IP Access List and Applying It to an Interface

IP access lists provide many benefits for securing a network and achieving nonsecurity goals, such as determining quality of service (QoS) factors or limiting **debug** command output. This module describes how to create standard, extended, named, and numbered IP access lists. An access list can be referenced by a name or a number. Standard access lists filter on only the source address in IP packets. Extended access lists can filter on source address, destination address, and other fields in an IP packet.

After you create an access list, you must apply it to something in order for it to have any effect. This module describes how to apply an access list to an interface. However, there are many other uses for access lists, which are mentioned in this module and described in other modules and in other configuration guides for various technologies.

- [Finding Feature Information, on page 11](#)
- [Restrictions for Creating an IP Access List and Applying It to an Interface, on page 12](#)
- [Information About Creating an IP Access List and Applying It to an Interface, on page 12](#)
- [How to Create an IP Access List and Apply It to an Interface, on page 13](#)
- [Configuration Examples for Creating an IP Access List and Applying It to a Physical Interface, on page 23](#)
- [Apply Access Control List on Management Interface, on page 27](#)
- [Additional References Creating an IP Access List and Applying It to an Interface, on page 28](#)

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see [Bug Search Tool](#) and the release notes for your platform and 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 feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to <https://cfng.cisco.com/>. An account on Cisco.com is not required.

Restrictions for Creating an IP Access List and Applying It to an Interface

The following restrictions apply when configuring IPv4 and IPv6 access control lists (ACLs)

- Application control engine (ACE)-specific counters are not supported.
- Layer 3 IPv4 and IPv6 ACLs are not supported on the same interface.
- MAC ACLs are not supported on Ethernet flow points (EFPs) or trunk EFP interfaces to which Layer 3 IPv4 or IPv6 ACLs are applied.
- IPv4 and IPv6 ACLs are not currently supported on EFP interfaces. IPv4 and IPv6 ACLs are supported on physical interfaces, bridge-domain interfaces, and port-channel interfaces.
- Layer 4 port-range functionality expands into Ternary Content-Addressable Memory (TCAM). IPv4 ACL scale is limited to 1K TCAM, Layer 2 ACL scale is limited to 1K TCAM entries.
- Object-groups are not supported with IP ACLs.
- The command **any options** is not supported.
- Starting with Cisco IOS XE Cupertino Release 17.7.1, ACLs are supported on management interface, Gigabit 0.

Information About Creating an IP Access List and Applying It to an Interface

Helpful Hints for Creating IP Access Lists

- Create the access list before applying it to an interface (or elsewhere), because if you apply a nonexistent access list to an interface and then proceed to configure the access list, the first statement is put into effect, and the implicit **deny** statement that follows could cause immediate access problems.
- Another reason to configure an access list before applying it is because an interface with an empty access list applied to it permits all traffic.
- All access lists need at least one **permit** statement; otherwise, all packets are denied and no traffic passes.
- Organize your access list so that more specific references in a network or subnet appear before more general ones.
- A packet will match the first ACE in the ACL. Thus, a **permit ip any any** will match all packets, ignoring all subsequent ACES.
- Although all access lists end with an implicit **deny** statement, we recommend use of an explicit **deny** statement (for example, **deny ip any any**). On most platforms, you can display the count of packets denied by issuing the **show access-list** command, thus finding out more information about who your

access list is disallowing. Only packets denied by explicit **deny** statements are counted, which is why the explicit **deny** statement will yield more complete data for you.

- While you are creating an access list or after it is created, you might want to delete an entry. You can delete an entry from a named access list. Use the **no permit** or **no deny** command to delete the appropriate entry.
- In order to make the purpose of individual statements more scannable and easily understood at a glance, you can write a helpful remark before or after any statement by using the **remark** command.
- If you want to deny access to a particular host or network and find out if someone from that network or host is attempting to gain access, include the **log** keyword with the corresponding **deny** statement so that the packets denied from that source are logged for you.
- This hint applies to the placement of your access list. When trying to save resources, remember that an inbound access list applies the filter conditions before the routing table lookup. An outbound access list applies the filter conditions after the routing table lookup.

Access List Remarks

You can include comments or remarks about entries in any IP access list. An access list remark is an optional remark before or after an access list entry that describes the entry so that you do not have to interpret the purpose of the entry. Each remark is limited to 100 characters in length.

The remark can go before or after a **permit** or **deny** statement. Be consistent about where you add remarks. Users may be confused if some remarks precede the associated **permit** or **deny** statements and some remarks follow the associated statements.

The following is an example of a remark that describes function of the subsequent **deny** statement:

```
ip access-list extended telnetting
remark Do not allow host1 subnet to telnet out
deny tcp host 172.16.2.88 any eq telnet
```

Additional IP Access List Features

Beyond the basic steps to create a standard or extended access list, you can enhance your access lists as mentioned below. Each of these methods is described completely in the *Refining an IP Access List module*.

- You can impose dates and times when **permit** or **deny** statements in an extended access list are in effect, making your access list more granular and specific to an absolute or periodic time period.
- After you create a named or numbered access list, you might want to add entries or change the order of the entries, which are known as resequencing an access list.
- You can achieve finer granularity when filtering packets by filtering on noninitial fragments of packets.

How to Create an IP Access List and Apply It to an Interface

This section describes the general ways to create a standard or extended access list using either a name or a number. Access lists are very flexible; the tasks simply illustrate one **permit** command and one **deny** command

to provide you the command syntax of each. Only you can determine how many **permit** and **deny** commands you need and their order.



Note The first two tasks in this module create an access list; you must apply the access list in order for it to function. If you want to apply the access list to an interface, perform the task “Applying the Access List to an Interface”.

Creating a Standard Access List to Filter on Source Address

If you want to filter on source address only, a standard access list is simple and sufficient. There are two alternative types of standard access list: named and numbered. Named access lists allow you to identify your access lists with a more intuitive name rather than a number, and they also support more features than numbered access lists.

Creating a Named Access List to Filter on Source Address

Use a standard, named access list if you need to filter on source address only. This task illustrates one **permit** statement and one **deny** statement, but the actual statements you use and their order depend on what you want to filter or allow. Define your **permit** and **deny** statements in the order that achieves your filtering goals.

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **ip access-list standard name**

Example:

```
Device(config)# ip access-list standard R&D
```

Defines a standard IP access list using a name and enters standard named access list configuration mode.

Step 4 **remark remark**

Example:

```
Device(config-std-nacl)# remark deny Sales network
```

(Optional) Adds a user-friendly comment about an access list entry.

- A remark can precede or follow an access list entry.
- In this example, the remark reminds the network administrator that the subsequent entry denies the Sales network access to the interface (assuming this access list is later applied to an interface).

Step 5 **deny** {*source* [*source-wildcard*] | **any**} [**log**]**Example:**

```
Device(config-std-nacl)# deny 172.16.0.0 0.0.255.255 log
```

(Optional) Denies the specified source based on a source address and wildcard mask.

- If the *source-wildcard* is omitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source address.
- Optionally use the keyword **any** as a substitute for the *source source-wildcard* to specify the source and source wildcard of 0.0.0.0 255.255.255.255.
- In this example, all hosts on network 172.16.0.0 are denied passing the access list.
- Because this example explicitly denies a source address and the **log** keyword is specified, any packets from that source are logged when they are denied. This is a way to be notified that someone on a network or host is trying to gain access.

Step 6 **remark** *remark***Example:**

```
Device(config-std-nacl)# remark Give access to Tester's host
```

(Optional) Adds a user-friendly comment about an access list entry.

- A remark can precede or follow an access list entry.
- This remark reminds the network administrator that the subsequent entry allows the Tester's host access to the interface.

Step 7 **permit** {*source* [*source-wildcard*] | **any**} [**log**]**Example:**

```
Device(config-std-nacl)# permit 172.18.5.22 0.0.0.0
```

Permits the specified source based on a source address and wildcard mask.

- Every access list needs at least one **permit** statement; it need not be the first entry.
- If the *source-wildcard* is omitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source address.
- Optionally use the keyword **any** as a substitute for the *source source-wildcard* to specify the source and source wildcard of 0.0.0.0 255.255.255.255.
- In this example, host 172.18.5.22 is allowed to pass the access list.

Step 8 Repeat some combination of Steps 4 through 7 until you have specified the sources on which you want to base your access list.

Remember that all sources not specifically permitted are denied by an implicit **deny** statement at the end of the access list.

Step 9 **end****Example:**

```
Device(config-std-nacl)# end
```

Exits standard named access list configuration mode and enters privileged EXEC mode.

Step 10 **show ip access-list****Example:**

```
Device# show ip access-list
```

(Optional) Displays the contents of all current IP access lists.

Creating a Numbered Access List to Filter on Source Address

Configure a standard, numbered access list if you need to filter on source address only and you prefer not to use a named access list.

IP standard access lists are numbered 1 to 99 or 1300 to 1999. This task illustrates one **permit** statement and one **deny** statement, but the actual statements you use and their order depend on what you want to filter or allow. Define your **permit** and **deny** statements in the order that achieves your filtering goals.

Step 1 **enable****Example:**

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal****Example:**

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **access-list *access-list-number* permit {*source* [*source-wildcard*] | any} [log]****Example:**

```
Device(config)# access-list 1 permit 172.16.5.22 0.0.0.0
```

Permits the specified source based on a source address and wildcard mask.

- Every access list needs at least one permit statement; it need not be the first entry.

- Standard IP access lists are numbered 1 to 99 or 1300 to 1999.
- If the source-wildcard is omitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source address.
- Optionally use the keyword **any** as a substitute for the source source-wildcard to specify the source and source wildcard of 0.0.0.0 255.255.255.255.
- In this example, host 172.16.5.22 is allowed to pass the access list.

Step 4 **access-list** *access-list-number* **deny** {*source* [*source-wildcard*] | **any**} [**log**]

Example:

```
Device(config)# access-list 1 deny 172.16.7.34 0.0.0.0
```

Denies the specified source based on a source address and wildcard mask.

- If the *source-wildcard* is omitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source address.
- Optionally use the abbreviation **any** as a substitute for the *source source-wildcard* to specify the source and source wildcard of 0.0.0.0 255.255.255.255.
- In this example, host 172.16.7.34 is denied passing the access list.

Step 5 Repeat some combination of Steps 3 through 6 until you have specified the sources on which you want to base your access list.

Remember that all sources not specifically permitted are denied by an implicit **deny** statement at the end of the access list.

Step 6 **end**

Example:

```
Device(config)# end
```

Exits global configuration mode and enters privileged EXEC mode.

Step 7 **show ip access-list**

Example:

```
Device# show ip access-list
```

(Optional) Displays the contents of all current IP access lists.

Creating an Extended Access List

If you want to filter on anything other than source address, you need to create an extended access list. There are two alternative types of extended access list: named and numbered. Named access lists allow you to identify your access lists with a more intuitive name rather than a number, and they also support more features.

For details on how to filter something other than source or destination address, see the syntax descriptions in the command reference documentation.

Creating a Named Extended Access List

Create a named extended access list if you want to filter the source and destination address or filter a combination of addresses and other IP fields.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list extended** *name*
4. **deny** *protocol source* [*source-wildcard*] *destination* [*destination-wildcard*] [**option** *option-name*] [**precedence** *precedence*] [**tos** *tos*] [**established**] [**log** | **log-input**] [**time-range** *time-range-name*] [**fragments**]
5. **permit** *protocol source* [*source-wildcard*] *destination* [*destination-wildcard*] [**option** *option-name*] [**precedence** *precedence*] [**tos** *tos*] [**established**] [**log** | **log-input**] [**time-range** *time-range-name*] [**fragments**]
6. Repeat some combination of Steps 4 through 7 until you have specified the fields and values on which you want to base your access list.
7. **end**
8. **show ip access-list**

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable Example: Device> enable	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	configure terminal Example: Device# configure terminal	Enters global configuration mode.
Step 3	ip access-list extended <i>name</i> Example: Device(config)# ip access-list extended acl1	Defines an extended IP access list using a name and enters extended named access list configuration mode.
Step 4	deny <i>protocol source</i> [<i>source-wildcard</i>] <i>destination</i> [<i>destination-wildcard</i>] [option <i>option-name</i>] [precedence <i>precedence</i>] [tos <i>tos</i>] [established] [log log-input] [time-range <i>time-range-name</i>] [fragments] Example:	(Optional) Denies any packet that matches all of the conditions specified in the statement. • If the <i>source-wildcard</i> or <i>destination-wildcard</i> is omitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source or destination address, respectively.

	Command or Action	Purpose
	<pre>Device(config-ext-nacl)# deny ip 172.18.0.0 0.0.255.255 host 172.16.40.10 log</pre>	<ul style="list-style-type: none"> Optionally use the keyword any as a substitute for the <i>source source-wildcard</i> or <i>destination destination-wildcard</i> to specify the address and wildcard of 0.0.0.0 255.255.255.255. Optionally use the keyword host <i>source</i> to indicate a source and source wildcard of <i>source</i> 0.0.0.0 or the abbreviation host <i>destination</i> to indicate a destination and destination wildcard of <i>destination</i> 0.0.0.0. In this example, packets from all sources are denied access to the destination network 172.18.0.0. Logging messages about packets permitted or denied by the access list are sent to the facility configured by the logging facility command (for example, console, terminal, or syslog). That is, any packet that matches the access list will cause an informational logging message about the packet to be sent to the configured facility. The level of messages logged to the console is controlled by the logging console command.
<p>Step 5</p>	<p>permit <i>protocol source [source-wildcard] destination [destination-wildcard] [option option-name] [precedence precedence] [tos tos] [established] [log log-input] [time-range time-range-name] [fragments]</i></p> <p>Example:</p> <pre>Device(config-ext-nacl)# permit tcp any any</pre>	<p>Permits any packet that matches all of the conditions specified in the statement.</p> <ul style="list-style-type: none"> Every access list needs at least one permit statement. If the <i>source-wildcard</i> or <i>destination-wildcard</i> is omitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source or destination address, respectively. Optionally use the keyword any as a substitute for the <i>source source-wildcard</i> or <i>destination destination-wildcard</i> to specify the address and wildcard of 0.0.0.0 255.255.255.255. In this example, TCP packets are allowed from any source to any destination. Use the log-input keyword to include input interface, source MAC address, or virtual circuit in the logging output.
<p>Step 6</p>	<p>Repeat some combination of Steps 4 through 7 until you have specified the fields and values on which you want to base your access list.</p>	<p>Remember that all sources not specifically permitted are denied by an implicit deny statement at the end of the access list.</p>
<p>Step 7</p>	<p>end</p> <p>Example:</p> <pre>Device(config-ext-nacl)# end</pre>	<p>Exits standard named access list configuration mode and enters privileged EXEC mode.</p>

	Command or Action	Purpose
Step 8	show ip access-list Example: Device# show ip access-list	(Optional) Displays the contents of all current IP access lists.

Creating a Numbered Extended Access List

Create a numbered extended access list if you want to filter on source and destination address, or a combination of addresses and other IP fields, and you prefer not to use a name. Extended IP access lists are numbered 100 to 199 or 2000 to 2699.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **access-list access-list-number remark remark**
4. **access-list access-list-number permit protocol {source [source-wildcard] | any} {destination [destination-wildcard] | any} [precedence precedence] [tos tos] [established] [log | log-input] [time-range time-range-name] [fragments]**
5. **access-list access-list-number remark remark**
6. **access-list access-list-number deny protocol {source [source-wildcard] | any} {destination [destination-wildcard] | any} [precedence precedence] [tos tos] [established] [log | log-input] [time-range time-range-name] [fragments]**
7. Repeat some combination of Steps 3 through 6 until you have specified the fields and values on which you want to base your access list.
8. **end**
9. **show ip access-list**

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable Example: Device> enable	Enables privileged EXEC mode. <ul style="list-style-type: none"> • Enter your password if prompted.
Step 2	configure terminal Example: Device# configure terminal	Enters global configuration mode.
Step 3	access-list access-list-number remark remark Example: Device(config)# access-list 107 remark allow Telnet packets from any source to network 172.69.0.0 (headquarters)	(Optional) Adds a user-friendly comment about an access list entry. <ul style="list-style-type: none"> • A remark of up to 100 characters can precede or follow an access list entry.

	Command or Action	Purpose
Step 4	<p>access-list <i>access-list-number</i> permit <i>protocol</i> {<i>source</i> [<i>source-wildcard</i>] any} {<i>destination</i> [<i>destination-wildcard</i>] any} [precedence <i>precedence</i>] [tos <i>tos</i>] [established] [log log-input] [time-range <i>time-range-name</i>] [fragments]</p> <p>Example:</p> <pre>Device(config)# access-list 107 permit tcp any 172.69.0.0 0.0.255.255 eq telnet</pre>	<p>Permits any packet that matches all of the conditions specified in the statement.</p> <ul style="list-style-type: none"> • Every access list needs at least one permit statement; it need not be the first entry. • Extended IP access lists are numbered 100 to 199 or 2000 to 2699. • If the <i>source-wildcard</i> or <i>destination-wildcard</i> is omitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source or destination address, respectively. • Optionally use the keyword any as a substitute for the <i>source source-wildcard</i> or <i>destination destination-wildcard</i> to specify the address and wildcard of 0.0.0.0 255.255.255.255. • TCP and other protocols have additional syntax available. See the access-list command in the command reference for complete syntax.
Step 5	<p>access-list <i>access-list-number</i> remark <i>remark</i></p> <p>Example:</p> <pre>Device(config)# access-list 107 remark deny all other TCP packets</pre>	<p>(Optional) Adds a user-friendly comment about an access list entry.</p> <ul style="list-style-type: none"> • A remark of up to 100 characters can precede or follow an access list entry.
Step 6	<p>access-list <i>access-list-number</i> deny <i>protocol</i> {<i>source</i> [<i>source-wildcard</i>] any} {<i>destination</i> [<i>destination-wildcard</i>] any} [precedence <i>precedence</i>] [tos <i>tos</i>] [established] [log log-input] [time-range <i>time-range-name</i>] [fragments]</p> <p>Example:</p> <pre>Device(config)# access-list 107 deny tcp any any</pre>	<p>Denies any packet that matches all of the conditions specified in the statement.</p> <ul style="list-style-type: none"> • If the <i>source-wildcard</i> or <i>destination-wildcard</i> is omitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source or destination address, respectively. • Optionally use the keyword any as a substitute for the <i>source source-wildcard</i> or <i>destination destination-wildcard</i> to specify the address and wildcard of 0.0.0.0 255.255.255.255.
Step 7	<p>Repeat some combination of Steps 3 through 6 until you have specified the fields and values on which you want to base your access list.</p>	<p>Remember that all sources not specifically permitted are denied by an implicit deny statement at the end of the access list.</p>
Step 8	<p>end</p> <p>Example:</p> <pre>Device(config)# end</pre>	<p>Exits global configuration mode and enters privileged EXEC mode.</p>

	Command or Action	Purpose
Step 9	show ip access-list Example: Device# show ip access-list	(Optional) Displays the contents of all current IP access lists.

Applying an Access List to a Physical Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip access-group** {*access-list-number* | *access-list-name*} {**in** | **out**}
5. **ip access-list extended** *acl-name* *acl-number*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable Example: Device> enable	Enables privileged EXEC mode. <ul style="list-style-type: none"> • Enter your password if prompted.
Step 2	configure terminal Example: Device# configure terminal	Enters global configuration mode.
Step 3	interface <i>type number</i> Example:	Specifies an interface and enters interface configuration mode.
Step 4	ip access-group { <i>access-list-number</i> <i>access-list-name</i> } { in out } Example: Device(config-if)# ip access-group acl1 in	Applies the specified access list to the inbound interface. <ul style="list-style-type: none"> • To filter source addresses, apply the access list to the inbound interface.
Step 5	ip access-list extended <i>acl-name</i> <i>acl-number</i> Example:	Enables extended ACL configuration mode. The CLI enters the extended ACL configuration mode in which all subsequent commands apply to the current extended access list. Enables extended ACL configuration mode. The CLI enters the extended ACL configuration mode in which all subsequent commands apply to the current extended access list.

	Command or Action	Purpose
		<ul style="list-style-type: none"> • Access list to which all commands entered from ACL configuration mode apply, using an alphanumeric string of up to 30 characters, beginning with a letter. • Access list to which all commands entered from access list configuration mode apply, using a numeric identifier. For extended access lists, the valid range is 100 to 199.
Step 6	end Example: Device(config-if)# end	Exits interface configuration mode and returns to privileged EXEC mode.

Configuration Examples for Creating an IP Access List and Applying It to a Physical Interface

Example: Filtering on Host Source Address

In the following example, the workstation belonging to user1 is allowed access to gigabitethernet 0/0/0, and the workstation belonging to user2 is not allowed access:

```
interface gigabitethernet 0/0/0
 ip access-group workstations in
 !
 ip access-list standard workstations
  remark Permit only user1 workstation through
  permit 172.16.2.88
  remark Do not allow user2 workstation through
  deny 172.16.3.13
```

Example: Filtering on Subnet Source Address

In the following example, the user1 subnet is not allowed access to gigabitethernet interface 0/0/0, but the Main subnet is allowed access:

```
interface gigabitethernet 0/0/0
 ip access-group prevention in
 !
 ip access-list standard prevention
  remark Do not allow user1 subnet through
  deny 172.22.0.0 0.0.255.255
  remark Allow Main subnet
  permit 172.25.0.0 0.0.255.255
```

Example: Filtering on Source and Destination Addresses and IP Protocols

The following configuration example shows an interface with two access lists, one applied to outgoing packets and one applied to incoming packets. The standard access list named Internet-filter filters outgoing packets on source address. The only packets allowed out the interface must be from source 172.16.3.4.

The extended access list named marketing-group filters incoming packets. The access list permits Telnet packets from any source to network 172.26.0.0 and denies all other TCP packets. It permits any ICMP packets. It denies UDP packets from any source to network 172.26.0.0 on port numbers less than 1024. Finally, the access list denies all other IP packets and performs logging of packets passed or denied by that entry.

```
interface gigabitethernet 0/0/0
 ip address 172.20.5.1 255.255.255.0
 ip access-group Internet-filter out
 ip access-group marketing-group in
!
ip access-list standard Internet-filter
 permit 172.16.3.4
ip access-list extended marketing-group
 permit tcp any 172.26.0.0 0.0.255.255 eq telnet
 deny tcp any any
 permit icmp any any
 deny udp any 172.26.0.0 0.0.255.255 lt 1024
 deny ip any any
```

Example: Filtering on Source Addresses Using a Numbered Access List

In the following example, network 10.0.0.0 is a Class A network whose second octet specifies a subnet; that is, its subnet mask is 255.255.0.0. The third and fourth octets of a network 10.0.0.0 address specify a particular host. Using access list 2, the Cisco IOS XE software would accept one address on subnet 48 and reject all others on that subnet. The last line of the list shows that the software would accept addresses on all other network 10.0.0.0 subnets.

```
interface gigabitethernet 0/0/0
 ip access-group 2 in
!
access-list 2 permit 10.48.0.3
access-list 2 deny 10.48.0.0 0.0.255.255
access-list 2 permit 10.0.0.0 0.255.255.255
```

Example: Preventing Telnet Access to a Subnet

In the following example, the user1 subnet is not allowed to telnet out of gigabitethernet interface 0/0/0:

```
interface gigabitethernet 0/0/0
 ip access-group telnetting out
!
ip access-list extended telnetting
 remark Do not allow user1 subnet to telnet out
 deny tcp 172.20.0.0 0.0.255.255 any eq telnet
 remark Allow Top subnet to telnet out
 permit tcp 172.33.0.0 0.0.255.255 any eq telnet
```

Example: Filtering on TCP and ICMP Using Port Numbers

In the following example, the first line of the extended access list named `acl1` permits any incoming TCP connections with destination ports greater than 1023. The second line permits incoming TCP connections to the Simple Mail Transfer Protocol (SMTP) port of host 172.28.1.2. The last line permits incoming ICMP messages for error feedback.

```
interface gigabitethernet 0/0/0
 ip access-group acl1 in
!
ip access-list extended acl1
 permit tcp any 172.28.0.0 0.0.255.255 gt 1023
 permit tcp any host 172.28.1.2 eq 25
 permit icmp any 172.28.0.0 255.255.255.255
```

Example: Allowing SMTP E-mail and Established TCP Connections

Suppose you have a network connected to the Internet, and you want any host on an Ethernet to be able to form TCP connections to any host on the Internet. However, you do not want IP hosts to be able to form TCP connections to hosts on the gigabitethernet except to the mail (SMTP) port of a dedicated mail host.

SMTP uses TCP port 25 on one end of the connection and a random port number on the other end. The same two port numbers are used throughout the life of the connection. Mail packets coming in from the Internet will have a destination port of 25. Outbound packets will have the port numbers reversed. The fact that the secure system behind the router always will accept mail connections on port 25 is what makes possible separate control of incoming and outgoing services. The access list can be configured on either the outbound or inbound interface.

In the following example, the gigabitethernet network is a Class B network with the address 172.18.0.0, and the address of the mail host is 172.18.1.2. The **established** keyword is used only for the TCP protocol to indicate an established connection. A match occurs if the TCP datagram has the ACK or RST bits set, which indicate that the packet belongs to an existing connection.

```
interface gigabitethernet 0/0/0
 ip access-group 102 in
!
access-list 102 permit tcp any 172.18.0.0 0.0.255.255 established
access-list 102 permit tcp any host 172.18.1.2 eq 25
```

Example: Preventing Access to the Web by Filtering on Port Name

In the following example, the `w1` and `w2` workstations are not allowed web access; other hosts on network 172.20.0.0 are allowed web access:

```
interface gigabitethernet0/0/0
 ip access-group no-web out
!
ip access-list extended no-web
 remark Do not allow w1 to browse the web
 deny host 172.20.3.85 any eq http
 remark Do not allow w2 to browse the web
 deny host 172.20.3.13 any eq http
 remark Allow others on our network to browse the web
 permit 172.20.0.0 0.0.255.255 any eq http
```

Example: Filtering on Source Address and Logging the Packets

The following example defines access lists 1 and 2, both of which have logging enabled:

```
interface gigabitethernet 0/0/0
 ip address 172.16.1.1 255.0.0.0
 ip access-group 1 in

!
access-list 1 permit 172.25.0.0 0.0.255.255 log
access-list 1 deny 172.30.0.0 0.0.255.255 log
!
access-list 2 permit 172.27.3.4 log
access-list 2 deny 172.17.0.0 0.0.255.255 log
```

If the interface receives 10 packets from 172.25.7.7 and 14 packets from 172.17.23.21, the first log will look like the following:

```
list 1 permit 172.25.7.7 1 packet
list 2 deny 172.17.23.21 1 packet
```

Five minutes later, the console will receive the following log:

```
list 1 permit 172.25.7.7 9 packets
list 2 deny 172.17.23.21 13 packets
```

Example: Limiting Debug Output

The following sample configuration uses an access list to limit the **debug** command output. Limiting the **debug** output restricts the volume of data to what you are interested in, saving you time and resources.

```
Device(config)# ip access-list acl1
Device(config-std-nacl)# remark Displays only advertisements for LDP peer in acl1
Device(config-std-nacl)# permit host 10.0.0.44

Device# debug mpls ldp advertisements peer-acl acl1

tagcon: peer 10.0.0.44:0 (pp 0x60E105BC): advertise 172.17.0.33
tagcon: peer 10.0.0.44:0 (pp 0x60E105BC): advertise 172.16.0.31
tagcon: peer 10.0.0.44:0 (pp 0x60E105BC): advertise 172.22.0.33
tagcon: peer 10.0.0.44:0 (pp 0x60E105BC): advertise 192.168.0.1
tagcon: peer 10.0.0.44:0 (pp 0x60E105BC): advertise 192.168.0.3
tagcon: peer 10.0.0.44:0 (pp 0x60E105BC): advertise 192.168.1.33
```

Apply Access Control List on Management Interface

Table 2: Feature History

Feature Name	Release Information	Feature Description
Apply Access Control List on Management Interface	Cisco IOS XE Cupertino 17.7.1	This feature enables Access Control Lists (ACLs) to be applied on management interface, Gigabit 0. The ACL configuration on the management interface helps to block ICMP traffic and thus prevents the Denial-of-Service (DoS) attacks.

Prior to Cisco IOS XE Cupertino Release 17.7.1, Access Control Lists (ACLs) were only supported on physical interfaces.

Starting with Cisco IOS XE Cupertino Release 17.7.1, ACLs are also supported on management interface, Gigabit 0.

Restrictions

- Egress ACL is not supported on management interface.
- Logging is not supported.
- Numbered ACL stats is not supported.

Configure ACL on Management Interface

Create an ACL

```
ip access-list mgmt
Extended IP access list mgmt
5 deny ip any any
10 permit icmp any any (4294967316 matches)
40 permit tcp any any eq telnet
```

Apply ACL on Management Interface

```
interface GigabitEthernet0
vrf forwarding Mgmt-intf
ip address 7.32.19.140 255.255.0.0
ip access-group mgmt intf
negotiation auto
```

Verification of ACL Configuration on Management Interface

Use the `show ip access-lists name` command to verify ACL configuration on management interface.

```
Router#show ip access-lists test
Extended IP access list test
```

```
10 deny ip host 10.126.154.73 any
20 deny ip any any (67485 matches)
```

Additional References Creating an IP Access List and Applying It to an Interface

Related Documents

Related Topic	Document Title
Security commands	<ul style="list-style-type: none"> • Cisco IOS Security Command Reference: Commands A to C • Cisco IOS Security Command Reference: Commands D to L • Cisco IOS Security Command Reference: Commands M to R • Cisco IOS Security Command Reference: Commands S to Z
<ul style="list-style-type: none"> • Order of access list entries • Access list entries based on time of day or week • Packets with noninitial fragments 	Refining an IP Access List
Filtering on IP options, TCP flags, or noncontiguous ports	Creating an IP Access List for Filtering
Controlling logging-related parameters	Understanding Access Control List Logging!

Standards and RFCs

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

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html



CHAPTER 3

Creating an IP Access List to Filter IP Options, TCP Flags, Noncontiguous Ports

This module describes how to use an IP access list to filter IP packets that contain certain IP Options, TCP flags, noncontiguous ports.

- [Finding Feature Information](#), on page 31
- [Prerequisites for Creating an IP Access List to Filter IP Options TCP Flags Noncontiguous Ports](#) , on page 31
- [Information About Creating an IP Access List to Filter IP Options, TCP Flags, Noncontiguous Ports](#) , on page 32
- [How to Create an IP Access List to Filter IP Options TCP Flags Noncontiguous Ports](#) , on page 34
- [Configuration Examples for Filtering IP Options, TCP Flags, Noncontiguous Ports](#) , on page 41
- [Additional References](#), on page 43

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see [Bug Search Tool](#) and the release notes for your platform and 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 feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to <https://cfng.cisco.com/>. An account on Cisco.com is not required.

Prerequisites for Creating an IP Access List to Filter IP Options TCP Flags Noncontiguous Ports

Before you perform any of the tasks in this module, you should be familiar with the information in the following modules:

- “IP Access List Overview”
- “Creating an IP Access List and Applying It to an Interface”

Information About Creating an IP Access List to Filter IP Options, TCP Flags, Noncontiguous Ports

IP Options

IP uses four key mechanisms in providing its service: Type of Service, Time to Live, Options, and Header Checksum.

The Options, commonly referred to as IP Options, provide for control functions that are required in some situations but unnecessary for the most common communications. IP Options include provisions for time stamps, security, and special routing.

IP Options may or may not appear in datagrams. They must be implemented by all IP modules (host and gateways). What is optional is their transmission in any particular datagram, not their implementation. In some environments the security option may be required in all datagrams.

The option field is variable in length. There may be zero or more options. IP Options can have one of two formats:

- Format 1: A single octet of option-type.
- Format 2: An option-type octet, an option-length octet, and the actual option-data octets.

The option-length octet counts the option-type octet, the option-length octet, and the option-data octets.

The option-type octet is viewed as having three fields: a 1-bit copied flag, a 2-bit option class, and a 5-bit option number. These fields form an 8-bit value for the option type field. IP Options are commonly referred to by their 8-bit value.

For a complete list and description of IP Options, refer to RFC 791, *Internet Protocol* at the following URL: <http://www.faqs.org/rfcs/rfc791.html>

Benefits of Filtering IP Options

- Filtering of packets that contain IP Options from the network relieves downstream devices and hosts of the load from options packets.
- This feature also minimizes load to the Route Processor (RP) for packets with IP Options that require RP processing on distributed systems. Previously, the packets were always routed to or processed by the RP CPU. Filtering the packets prevents them from impacting the RP.

Benefits of Filtering on TCP Flags

The ACL TCP Flags Filtering feature provides a flexible mechanism for filtering on TCP flags. Previously, an incoming packet was matched as long as any TCP flag in the packet matched a flag specified in the access control entry (ACE). This behavior allows for a security loophole, because packets with all flags set could get past the access control list (ACL). The ACL TCP Flags Filtering feature allows you to select any combination of flags on which to filter. The ability to match on a flag set and on a flag not set gives you a greater degree of control for filtering on TCP flags, thus enhancing security.

Because TCP packets can be sent as false synchronization packets that can be accepted by a listening port, it is recommended that administrators of firewall devices set up some filtering rules to drop false TCP packets.

The ACEs that make up an access list can be configured to detect and drop unauthorized TCP packets by allowing only the packets that have a very specific group of TCP flags set or not set. The ACL TCP Flags Filtering feature provides a greater degree of packet-filtering control in the following ways:

- You can select any desired combination of TCP flags on which to filter TCP packets.
- You can configure ACEs to allow matching on a flag that is set, as well as on a flag that is not set.

TCP Flags

The table below lists the TCP flags, which are further described in RFC 793, *Transmission Control Protocol*.

Table 3: TCP Flags

TCP Flag	Purpose
ACK	Acknowledge flag—Indicates that the acknowledgment field of a segment specifies the next sequence number the sender of this segment is expecting to receive.
FIN	Finish flag—Used to clear connections.
PSH	Push flag—Indicates the data in the call should be immediately pushed through to the receiving user.
RST	Reset flag—Indicates that the receiver should delete the connection without further interaction.
SYN	Synchronize flag—Used to establish connections.
URG	Urgent flag—Indicates that the urgent field is meaningful and must be added to the segment sequence number.

Benefits of Using the Named ACL Support for Noncontiguous Ports on an Access Control Entry Feature

This feature greatly reduces the number of access control entries (ACEs) required in an access control list to handle multiple entries for the same source address, destination address, and protocol. If you maintain large numbers of ACEs, use this feature to consolidate existing groups of access list entries wherever it is possible and when you create new access list entries. When you configure access list entries with noncontiguous ports, you will have fewer access list entries to maintain.

How to Create an IP Access List to Filter IP Options TCP Flags Noncontiguous Ports

Filtering Packets That Contain IP Options

Complete these steps to configure an access list to filter packets that contain IP options and to verify that the access list has been configured correctly.



Note

- The ACL Support for Filtering IP Options feature can be used only with named, extended ACLs.
- Resource Reservation Protocol (RSVP) Multiprotocol Label Switching Traffic Engineering (MPLS TE), Internet Group Management Protocol Version 2 (IGMPV2), and other protocols that use IP options packets may not function in drop or ignore mode if this feature is configured.
- On most Cisco devices, a packet with IP options is not switched in hardware, but requires control plane software processing (primarily because there is a need to process the options and rewrite the IP header), so all IP packets with IP options will be filtered and switched in software.

Step 1 enable

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 configure terminal

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 ip access-list extended *access-list-name*

Example:

```
Device(config)# ip access-list extended mylist1
```

Specifies the IP access list by name and enters named access list configuration mode.

Step 4 [*sequence-number*] deny *protocol source source-wildcard destination destination-wildcard* [**option** *option-value*] [**precedence** *precedence*] [**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]

Example:

```
Device(config-ext-nacl)# deny ip any any option traceroute
```

(Optional) Specifies a **deny** statement in named IP access list mode.

- This access list happens to use a **deny** statement first, but a **permit** statement could appear first, depending on the order of statements you need.

- Use the **option** keyword and *option-value* argument to filter packets that contain a particular IP Option.
- In this example, any packet that contains the traceroute IP option will be filtered out.
- Use the **no sequence-number** form of this command to delete an entry.

Step 5 `[sequence-number] permit protocol source source-wildcard destination destination-wildcard [option option-value] [precedence precedence] [tos tos] [log] [time-range time-range-name] [fragments]`

Example:

```
Device(config-ext-nacl)# permit ip any any option security
```

Specifies a **permit** statement in named IP access list mode.

- In this example, any packet (not already filtered) that contains the security IP option will be permitted.
- Use the **no sequence-number** form of this command to delete an entry.

Step 6 Repeat Step 4 or Step 5 as necessary.

Allows you to revise the access list.

Step 7 **end**

Example:

```
Device(config-ext-nacl)# end
```

(Optional) Exits named access list configuration mode and returns to privileged EXEC mode.

Step 8 `show ip access-lists access-list-name`

Example:

```
Device# show ip access-lists mylist1
```

(Optional) Displays the contents of the IP access list.

What to Do Next

Apply the access list to an interface or reference it from a command that accepts an access list.



Note To effectively eliminate all packets that contain IP Options, we recommend that you configure the global **ip options drop** command.

Filtering Packets That Contain TCP Flags

This task configures an access list to filter packets that contain TCP flags and verifies that the access list has been configured correctly.

**Note**

- TCP flag filtering can be used only with named, extended ACLs.
- The ACL TCP Flags Filtering feature is supported only for Cisco ACLs.
- Previously, the following command-line interface (CLI) format could be used to configure a TCP flag-checking mechanism:

permit tcp any any rst The following format that represents the same ACE can now be used: **permit tcp any any match-any +rst** Both the CLI formats are accepted; however, if the new keywords **match-all** or **match-any** are chosen, they must be followed by the new flags that are prefixed with “+” or “-”. It is advisable to use only the old format or the new format in a single ACL. You cannot mix and match the old and new CLI formats.

**Caution**

If a device having ACEs with the new syntax format is reloaded with a previous version of the Cisco software that does not support the ACL TCP Flags Filtering feature, the ACEs will not be applied, leading to possible security loopholes.

Step 1 **enable****Example:**

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal****Example:**

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **ip access-list extended *access-list-name*****Example:**

```
Device(config)# ip access-list extended kmdl
```

Specifies the IP access list by name and enters named access list configuration mode.

Step 4 [*sequence-number*] **permit tcp** *source source-wildcard* [*operator* [*port*]] *destination destination-wildcard* [*operator* [*port*]] [**established** | **match-any** | **match-all**] {+ | -} [*flag-name*] [**precedence** *precedence*] [**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]**Example:**

```
Device(config-ext-nacl)# permit tcp any any match-any +rst
```

Specifies a **permit** statement in named IP access list mode.

- This access list happens to use a **permit** statement first, but a **deny** statement could appear first, depending on the order of statements you need.
- Use the TCP command syntax of the **permit** command.
- Any packet with the RST TCP header flag set will be matched and allowed to pass the named access list `kmd1` in Step 3.

Step 5 `[sequence-number] deny tcp source source-wildcard [operator [port]] destination destination-wildcard [operator [port]] [established] {match-any | match-all} {+ | -} flag-name [precedence precedence] [tos tos] [log] [time-range time-range-name] [fragments]`

Example:

```
Device(config-ext-nacl)# deny tcp any any match-all -ack -fin
```

(Optional) Specifies a **deny** statement in named IP access list mode.

- This access list happens to use a **permit** statement first, but a **deny** statement could appear first, depending on the order of statements you need.
- Use the TCP command syntax of the **deny** command.
- Any packet that does not have the ACK flag set, and also does not have the FIN flag set, will not be allowed to pass the named access list `kmd1` in Step 3.
- See the **deny**(IP) command for additional command syntax to permit upper-layer protocols (ICMP, IGMP, TCP, and UDP).

Step 6 Repeat Step 4 or Step 5 as necessary, adding statements by sequence number where you planned. Use the **no sequence-number** command to delete an entry.

Allows you to revise the access list.

Step 7 **end**

Example:

```
Device(config-ext-nacl)# end
```

(Optional) Exits the configuration mode and returns to privileged EXEC mode.

Step 8 **show ip access-lists** *access-list-name*

Example:

```
Device# show ip access-lists kmd1
```

(Optional) Displays the contents of the IP access list.

- Review the output to confirm that the access list includes the new entry.

Configuring an Access Control Entry with Noncontiguous Ports

Perform this task to create access list entries that use noncontiguous TCP or UDP port numbers. Although this task uses TCP ports, you could use the UDP syntax of the **permit** and **deny** commands to filter noncontiguous UDP ports.

Although this task uses a **permit** command first, use the **permit** and **deny** commands in the order that achieves your filtering goals.



Note The ACL—Named ACL Support for Noncontiguous Ports on an Access Control Entry feature can be used only with named, extended ACLs.

Step 1 enable

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 configure terminal

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 ip access-list extended *access-list-name*

Example:

```
Device(config)# ip access-list extended acl-extd-1
```

Specifies the IP access list by name and enters named access list configuration mode.

Step 4 [*sequence-number*] **permit tcp** *source source-wildcard* [*operator port* [*port*]] *destination destination-wildcard* [*operator* [*port*]] [**established** {**match-any** | **match-all**} {+ | -} *flag-name*] [**precedence** *precedence*] [**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]

Example:

```
Device(config-ext-nacl)# permit tcp any eq telnet ftp any eq 450 679
```

Specifies a **permit** statement in named IP access list configuration mode.

- Operators include **lt** (less than), **gt** (greater than), **eq** (equal), **neq** (not equal), and **range** (inclusive range).
- If the operator is positioned after the source and source-wildcard arguments, it must match the source port. If the operator is positioned after the destination and destination-wildcard arguments, it must match the destination port.
- The **range** operator requires two port numbers. You can configure up to 10 ports after the **eq** and **neq** operators. All other operators require one port number.
- To filter UDP ports, use the UDP syntax of this command.

Step 5 `[sequence-number] deny tcp source source-wildcard [operator port [port]] destination destination-wildcard [operator [port]] [established {match-any | match-all} {+ | -} flag-name] [precedence precedence] [tos tos] [log] [time-range time-range-name] [fragments]`

Example:

```
Device(config-ext-nacl)# deny tcp any neq 45 565 632 any
```

(Optional) Specifies a **deny** statement in named access list configuration mode.

- Operators include **lt** (less than), **gt** (greater than), **eq** (equal), **neq** (not equal), and **range** (inclusive range).
- If the *operator* is positioned after the *source* and *source-wildcard* arguments, it must match the source port. If the *operator* is positioned after the *destination* and *destination-wildcard* arguments, it must match the destination port.
- The **range** operator requires two port numbers. You can configure up to 10 ports after the **eq** and **neq** operators. All other operators require one port number.
- To filter UDP ports, use the UDP syntax of this command.

Step 6 Repeat Step 4 or Step 5 as necessary, adding statements by sequence number where you planned. Use the **no sequence-number** command to delete an entry.

Allows you to revise the access list.

Step 7 **end**

Example:

```
Device(config-ext-nacl)# end
```

(Optional) Exits named access list configuration mode and returns to privileged EXEC mode.

Step 8 **show ip access-lists** *access-list-name*

Example:

```
Device# show ip access-lists kmdl
```

(Optional) Displays the contents of the access list.

Consolidating Access List Entries with Noncontiguous Ports into One Access List Entry

Perform this task to consolidate a group of access list entries with noncontiguous ports into one access list entry.

Although this task uses TCP ports, you could use the UDP syntax of the **permit** and **deny** commands to filter noncontiguous UDP ports.

Although this task uses a **permit** command first, use the **permit** and **deny** commands in the order that achieves your filtering goals.

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 `show ip access-lists` *access-list-name*

Example:

```
Device# show ip access-lists mylist1
```

(Optional) Displays the contents of the IP access list.

- Review the output to see if you can consolidate any access list entries.

Step 3 `configure terminal`

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 4 `ip access-list extended` *access-list-name*

Example:

```
Device(config)# ip access-list extended mylist1
```

Specifies the IP access list by name and enters named access list configuration mode.

Step 5 `no` [*sequence-number*] **permit** *protocol source source-wildcard destination destination-wildcard* [**option** *option-name*] [**precedence** *precedence*] [**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]

Example:

```
Device(config-ext-nacl)# no 10
```

Removes the redundant access list entry that can be consolidated.

- Repeat this step to remove entries to be consolidated because only the port numbers differ.
- After this step is repeated to remove the access list entries 20, 30, and 40, for example, those entries are removed because they will be consolidated into one **permit** statement.
- If a *sequence-number* is specified, the rest of the command syntax is optional.

Step 6 [*sequence-number*] **permit** *protocol source source-wildcard* [*operator port[port]*] *destination destination-wildcard* [*operator port[port]*] [**option** *option-name*] [**precedence** *precedence*] [**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]

Example:

```
Device(config-ext-nacl)# permit tcp any neq 45 565 632 any eq 23 45 34 43
```

Specifies a **permit** statement in named access list configuration mode.

- In this instance, a group of access list entries with noncontiguous ports was consolidated into one **permit** statement.
- You can configure up to 10 ports after the **eq** and **neq** operators.

Step 7 Repeat Steps 5 and 6 as necessary, adding **permit** or **deny** statements to consolidate access list entries where possible. Use the `no` *sequence-number* command to delete an entry.

Allows you to revise the access list.

Step 8 **end**

Example:

```
Device(config-std-nacl)# end
```

(Optional) Exits named access list configuration mode and returns to privileged EXEC mode.

Step 9 **show ip access-lists** *access-list-name*

Example:

```
Device# show ip access-lists mylist1
```

(Optional) Displays the contents of the access list.

What To Do Next

Apply the access list to an interface or reference it from a command that accepts an access list.

Configuration Examples for Filtering IP Options, TCP Flags, Noncontiguous Ports

Example: Filtering Packets That Contain IP Options

The following example shows an extended access list named mylist2 that contains access list entries (ACEs) that are configured to permit TCP packets only if they contain the IP Options that are specified in the ACEs:

```
ip access-list extended mylist2
 10 permit ip any any option eool
 20 permit ip any any option record-route
 30 permit ip any any option zsu
 40 permit ip any any option mtup
```

The **show access-list** command has been entered to show how many packets were matched and therefore permitted:

```
Device# show ip access-list mylist2
Extended IP access list test
10 permit ip any any option eool (1 match)
20 permit ip any any option record-route (1 match)
30 permit ip any any option zsu (1 match)
40 permit ip any any option mtup (1 match)
```

Example: Filtering Packets That Contain TCP Flags

The following access list allows TCP packets only if the TCP flags ACK and SYN are set and the FIN flag is not set:

Example: Creating an Access List Entry with Noncontiguous Ports

```
ip access-list extended aaa
 permit tcp any any match-all +ack +syn -fin
end
```

The **show access-list** command has been entered to display the ACL:

```
Device# show access-list aaa

Extended IP access list aaa
 10 permit tcp any any match-all +ack +syn -fin
```

Example: Creating an Access List Entry with Noncontiguous Ports

The following access list entry can be created because up to ten ports can be entered after the **eq** and **neq** operators:

```
ip access-list extended aaa
 permit tcp any eq telnet ftp any eq 23 45 34
end
```

Enter the **show access-lists** command to display the newly created access list entry.

```
Device# show access-lists aaa

Extended IP access list aaa
 10 permit tcp any eq telnet ftp any eq 23 45 34
```

Example: Consolidating Some Existing Access List Entries into One Access List Entry with Noncontiguous Ports

The **show access-lists** command is used to display a group of access list entries for the access list named abc:

```
Device# show access-lists abc
Extended IP access list abc
 10 permit tcp any eq telnet any eq 450
 20 permit tcp any eq telnet any eq 679
 30 permit tcp any eq ftp any eq 450
 40 permit tcp any eq ftp any eq 679
```

Because the entries are all for the same **permit** statement and simply show different ports, they can be consolidated into one new access list entry. The following example shows the removal of the redundant access list entries and the creation of a new access list entry that consolidates the previously displayed group of access list entries:

```
ip access-list extended abc
 no 10
 no 20
 no 30
 no 40
 permit tcp any eq telnet ftp any eq 450 679
end
```

When the **show access-lists** command is reentered, the consolidated access list entry is displayed:

```
Device# show access-lists abc
Extended IP access list abc
 10 permit tcp any eq telnet ftp any eq 450 679
```

Additional References

Related Documents

Related Topic	Document Title
Security commands	<i>Cisco IOS Security Command Reference</i>
Configuring the device to drop or ignore packets containing IP Options by using the no ip options command.	<i>ACL IP Options Selective Drop</i>
Overview information about access lists.	<i>IP Access List Overview</i>
Information about creating an IP access list and applying it to an interface	<i>Creating an IP Access List and Applying It to an Interface</i>
QoS commands	<i>Cisco IOS Quality of Service Solutions Command Reference</i>

RFCs

RFC	Title
RFC 791	<i>Internet Protocol</i> http://www.faqs.org/rfcs/rfc791.html
RFC 793	<i>Transmission Control Protocol</i>
RFC 1393	<i>Traceroute Using an IP Option</i>

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html



CHAPTER 4

IPv6 Access Control Lists

Access lists determine what traffic is blocked and what traffic is forwarded at device interfaces and allow filtering of traffic based on source and destination addresses, and inbound and outbound traffic to a specific interface. Standard IPv6 ACL functionality was extended to support traffic filtering based on IPv6 option headers and optional, upper-layer protocol type information for finer granularity of control. Standard IPv6 ACL functionality was extended to support traffic filtering based on IPv6 option headers and optional, upper-layer protocol type information for finer granularity of control.

This module describes how to configure IPv6 traffic filtering and to control access to virtual terminal lines.

- [Finding Feature Information, on page 45](#)
- [Restrictions for IPv6 Access Control Lists, on page 45](#)
- [Information About IPv6 Access Control Lists, on page 47](#)
- [How to Configure IPv6 Access Control Lists, on page 47](#)
- [Configuration Examples for IPv6 Access Control Lists, on page 52](#)
- [Use Case or Deployment Scenarios, on page 53](#)
- [Additional References, on page 54](#)

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see [Bug Search Tool](#) and the release notes for your platform and 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 feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to <https://cfnng.cisco.com/>. An account on Cisco.com is not required.

Restrictions for IPv6 Access Control Lists

- IPv4 / IPv6 ACL is supported only at the access Layer 3 or bridge domain interfaces. It is not supported on the MPLS enabled core interfaces.
- IPv6 ACL is *not* supported on port-channel member link.
- IPv4 or IPv6 is *not* supported on label interface.
- MAC ACL supports only non-IP packets.

- MAC ACL counters are *not* supported.
- MAC ACL is *not* supported on trunk EFP or port channel EFP.
- TCAMS are programmed for ACL configured on both physical interface and BDI.
- A total of 512 unique and combination of IPv4 and IPv6 ACLs can be configured.
- Filtering based on extension header types is *not* supported except for filtering based on fragmentation.
- Filtering based on IPv6 Protocols and L4 ports will *not* work for IPv6 traffic carrying extension headers.
- Filtering based on TCP Control Flags is *not* supported.
- Egress IPv6 ACL is *not* supported.
- All NDP packets (ICMPv6 type 133-137) are allowed by default.
- IPv6 ACL is *not* supported on Link Local Addresses (LL).



Note IPv6 packets with LL address as SA is permitted by default.

- Per ACE counters are *not* supported.
- Re-sequencing of ACE's for an IPv6 ACL is *not* supported.
- Starting Cisco IOS XE Release 3.18SP, IPv6 ACLs are supported on the RSP3 module.
- Although the hardware is not programmed for TCAM exhaustion, ACL stats get incremented for control packets on the interface.

Restrictions - Dual Stack (IPv4 + IPv6 ACL on an Interface)

- When a unique combination of IPv4 ACL and IPv6 ACL are configured on the same interface for the first time, there is a delay in filtering when the first ACL applied (in the sequence) is reattached.
- If an interface has both IPv4 ACL and IPv6 ACL configured, there is a delay in filtering when one unique ACL is removed and the other unique ACL is attached. For example, IPv4 ACL is x and IPv6 ACL is y on an interface. When x is removed and y is reattached there is a delay in filtering.
- A single IPv6 ACL attached to an interface or multiple interfaces works as a unique ACL when compared to a combination of IPv6 ACL and IPv4 ACL applied to one or more interfaces.
- The ACL attachment is rejected and may result in TCAM overflow, when you attach an IPv6 ACL where an IPv4 ACL was attached and the reverse.
- The IPv6 ACL attachment, for the first time, to an interface is rejected and may result in TCAM overflow when there is no IPv4 ACL attached.
- If a IPv6 ACL is detached from an interface, which also has IPv4 ACL attached to it may result in TCAM overflow of the IPv4 ACL and detach of both ACLs from that interface.

Information About IPv6 Access Control Lists

Access Control Lists for IPv6 Traffic Filtering

The standard ACL functionality in IPv6 is similar to standard ACLs in IPv4. Access lists determine what traffic is blocked and what traffic is forwarded at device interfaces and allow filtering based on source and destination addresses, inbound and outbound to a specific interface. Each access list has an implicit deny statement at the end. IPv6 ACLs are defined and their deny and permit conditions are set using the **ipv6 access-list** command with the **deny** and **permit** keywords in global configuration mode.

IPv6 extended ACLs augments standard IPv6 ACL functionality to support traffic filtering based on IPv6 option headers and optional, upper-layer protocol type information for finer granularity of control (functionality similar to extended ACLs in IPv4).

IPv6 Packet Inspection

The following header fields are used for IPv6 inspection: traffic class, flow label, payload length, next header, hop limit, and source or destination IP address. For further information on and descriptions of the IPv6 header fields, see RFC 2474.

Access Class Filtering in IPv6

Filtering incoming and outgoing connections to and from the device based on an IPv6 ACL is performed using the **ipv6 access-class** command in line configuration mode. The **ipv6 access-class** command is similar to the **access-class** command, except the IPv6 ACLs are defined by a name. If the IPv6 ACL is applied to inbound traffic, the source address in the ACL is matched against the incoming connection source address and the destination address in the ACL is matched against the local device address on the interface. If the IPv6 ACL is applied to outbound traffic, the source address in the ACL is matched against the local device address on the interface and the destination address in the ACL is matched against the outgoing connection source address. We recommend that identical restrictions are set on all the virtual terminal lines because a user can attempt to connect to any of them.

How to Configure IPv6 Access Control Lists

Configuring IPv6 Traffic Filtering

Creating and Configuring an IPv6 ACL for Traffic Filtering



Note IPv6 ACLs on the Cisco ASR 1000 platform do not contain implicit permit rules. The IPv6 neighbor discovery process uses the IPv6 network-layer service; therefore, to enable IPv6 neighbor discovery, you must add IPv6 ACLs to allow IPv6 neighbor discovery packets to be sent and received on an interface. In IPv4, the Address Resolution Protocol (ARP), which is equivalent to the IPv6 neighbor discovery process, uses a separate data-link-layer protocol; therefore, by default IPv4 ACLs implicitly allow ARP packets to be sent and received on an interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. Do one of the following:
 - **permit protocol** {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address*} [*operator* [*port-number*]] {*destination-ipv6-prefix / prefix-length* | **any** | **host** *destination-ipv6-address*} [**operator** [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
 - **deny protocol** {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address*} [*operator* [*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address*} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable Example: Device> enable	Enables privileged EXEC mode. <ul style="list-style-type: none"> • Enter your password if prompted.
Step 2	configure terminal Example: Device# configure terminal	Enters global configuration mode.
Step 3	ipv6 access-list <i>access-list-name</i> Example: Device(config)# ipv6 access-list inbound	Defines an IPv6 ACL, and enters IPv6 access list configuration mode. <ul style="list-style-type: none"> • The <i>access-list name</i> argument specifies the name of the IPv6 ACL. IPv6 ACL names cannot contain a space or quotation mark, or begin with a numeral.
Step 4	Do one of the following: <ul style="list-style-type: none"> • permit protocol {<i>source-ipv6-prefix/prefix-length</i> any host <i>source-ipv6-address</i>} [<i>operator</i> [<i>port-number</i>]] {<i>destination-ipv6-prefix / prefix-length</i> any host <i>destination-ipv6-address</i>} [operator [<i>port-number</i>]] [dest-option-type [<i>doh-number</i> <i>doh-type</i>]] [dscp <i>value</i>] [flow-label <i>value</i>] [fragments] [log] [log-input] [mobility] [mobility-type [<i>mh-number</i> <i>mh-type</i>]] [routing] [routing-type <i>routing-number</i>] [sequence <i>value</i>] [time-range <i>name</i>] • deny protocol {<i>source-ipv6-prefix/prefix-length</i> any host <i>source-ipv6-address</i>} [<i>operator</i> 	Specifies permit or deny conditions for an IPv6 ACL.

	Command or Action	Purpose
	<p><i>port-number</i>] {<i>destination-ipv6-prefix/prefix-length</i> any host <i>destination-ipv6-address</i>} [<i>operator</i> [<i>port-number</i>]] [dest-option-type [<i>doh-number</i> <i>doh-type</i>]] [dscp <i>value</i>] [flow-label <i>value</i>] [fragments] [log] [log-input] [mobility] [mobility-type [<i>mh-number</i> <i>mh-type</i>]] [routing] [routing-type <i>routing-number</i>] [sequence <i>value</i>] [time-range <i>name</i>] [undetermined-transport</p> <p>Example:</p> <pre>Device(config-ipv6-acl)# permit tcp 2001:DB8:0300:0201::/32 eq telnet any</pre> <p>Example:</p> <pre>Device(config-ipv6-acl)# deny tcp host 2001:DB8:1::1 any log-input</pre>	

Applying the IPv6 ACL to an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 traffic-filter** *access-list-name* {**in**| **out**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	<p>enable</p> <p>Example:</p> <pre>Device> enable</pre>	<p>Enables privileged EXEC mode.</p> <ul style="list-style-type: none"> • Enter your password if prompted.
Step 2	<p>configure terminal</p> <p>Example:</p> <pre>Device# configure terminal</pre>	<p>Enters global configuration mode.</p>
Step 3	<p>interface <i>type number</i></p> <p>Example:</p> <pre>Device(config)# interface gigabitethernet 0/0/0</pre>	<p>Specifies the interface type and number, and enters interface configuration mode.</p>
Step 4	<p>ipv6 traffic-filter <i>access-list-name</i> {in out}</p> <p>Example:</p>	<p>Applies the specified IPv6 access list to the interface specified in the previous step.</p>

	Command or Action	Purpose
	Device(config-if)# ipv6 traffic-filter inbound in	

Controlling Access to a vty

Creating an IPv6 ACL to Provide Access Class Filtering

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. Do one of the following:
 - **permit protocol** {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address*} [*operator* [*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address*} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*]
 - **deny protocol** {*source-ipv6-prefix/prefix-length* | **any** | **host** *source-ipv6-address*} [*operator* [*port-number*]] {*destination-ipv6-prefix/prefix-length* | **any** | **host** *destination-ipv6-address*} [*operator* [*port-number*]] [**dest-option-type** [*doh-number* | *doh-type*]] [**dscp** *value*] [**flow-label** *value*] [**fragments**] [**log**] [**log-input**] [**mobility**] [**mobility-type** [*mh-number* | *mh-type*]] [**routing**] [**routing-type** *routing-number*] [**sequence** *value*] [**time-range** *name*] [**undetermined-transport**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable Example: Device> enable	Enables privileged EXEC mode. <ul style="list-style-type: none"> • Enter your password if prompted.
Step 2	configure terminal Example: Device# configure terminal	Enters global configuration mode.
Step 3	ipv6 access-list <i>access-list-name</i> Example: Device(config)# ipv6 access-list cisco	Defines an IPv6 ACL, and enters IPv6 access list configuration mode.
Step 4	Do one of the following: <ul style="list-style-type: none"> • permit protocol {<i>source-ipv6-prefix/prefix-length</i> any host <i>source-ipv6-address</i>} [<i>operator</i> 	Specifies permit or deny conditions for an IPv6 ACL.

	Command or Action	Purpose
	<p>[<i>port-number</i>] {<i>destination-ipv6-prefix / prefix-length</i> any host <i>destination-ipv6-address</i>} [<i>operator</i> [<i>port-number</i>]] [dest-option-type [<i>doh-number</i> <i>doh-type</i>]] [dscp <i>value</i>] [flow-label <i>value</i>] [fragments] [log] [log-input] [mobility] [mobility-type [<i>mh-number</i> <i>mh-type</i>]] [routing] [routing-type <i>routing-number</i>] [sequence <i>value</i>] [time-range <i>name</i>]</p> <p>• deny <i>protocol</i> {<i>source-ipv6-prefix/prefix-length</i> any host <i>source-ipv6-address</i>} [<i>operator</i> <i>port-number</i>]] {<i>destination-ipv6-prefix/prefix-length</i> any host <i>destination-ipv6-address</i>} [<i>operator</i> [<i>port-number</i>]] [dest-option-type [<i>doh-number</i> <i>doh-type</i>]] [dscp <i>value</i>] [flow-label <i>value</i>] [fragments] [log] [log-input] [mobility] [mobility-type [<i>mh-number</i> <i>mh-type</i>]] [routing] [routing-type <i>routing-number</i>] [sequence <i>value</i>] [time-range <i>name</i>] [undetermined-transport]</p> <p>Example:</p> <pre>Device(config-ipv6-acl)# permit ipv6 host 2001:DB8:0:4::32 any</pre> <p>Example:</p> <pre>Device(config-ipv6-acl)# deny ipv6 host 2001:DB8:0:6::6 any</pre>	

Applying an IPv6 ACL to the Virtual Terminal Line

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **line** [**aux** | **console** | **tty** | **vty**] *line-number*[*ending-line-number*]
4. **ipv6 access-class** *ipv6-access-list-name* {**in** | **out**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	<p>enable</p> <p>Example:</p> <pre>Device> enable</pre>	<p>Enables privileged EXEC mode.</p> <ul style="list-style-type: none"> • Enter your password if prompted.
Step 2	<p>configure terminal</p> <p>Example:</p> <pre>Device# configure terminal</pre>	<p>Enters global configuration mode.</p>

	Command or Action	Purpose
Step 3	line [aux console tty vty] <i>line-number</i> [ending-line-number] Example: Device(config)# line vty 0 4	Identifies a specific line for configuration and enters line configuration mode. <ul style="list-style-type: none"> In this example, the vtty keyword is used to specify the virtual terminal lines for remote console access.
Step 4	ipv6 access-class <i>ipv6-access-list-name</i> {in out} Example: Device(config-line)# ipv6 access-class cisco in	Filters incoming and outgoing connections to and from the device based on an IPv6 ACL.

Configuration Examples for IPv6 Access Control Lists

Example: Verifying IPv6 ACL Configuration

In this example, the **show ipv6 access-list** command is used to verify that IPv6 ACLs are configured correctly:

```
Device> show ipv6 access-list
```

```
IPv6 access list inbound
  permit tcp any any eq bgp (8 matches) sequence 10
  permit tcp any any eq telnet (15 matches) sequence 20
  permit udp any any sequence 30
```

```
IPv6 access list Virtual-Access2.1#427819008151 (per-user)
  permit tcp host 2001:DB8:1::32 eq bgp host 2001:DB8:2::32 eq 11000 sequence 1
  permit tcp host 2001:DB8:1::32 eq telnet host 2001:DB8:2::32 eq 11001 sequence 2
```

Example: Creating and Applying an IPv6 ACL

The following example shows how to restrict HTTP access to certain hours during the day and log any activity outside of the permitted hours:

```
Device# configure terminal
Device(config)# time-range lunchtime
Device(config-time-range)# periodic weekdays 12:00 to 13:00
Device(config-time-range)# exit
Device(config)# ipv6 access-list INBOUND
Device(config-ipv6-acl)# permit tcp any any eq www time-range lunchtime
Device(config-ipv6-acl)# deny tcp any any eq www log-input
Device(config-ipv6-acl)# permit tcp 2001:DB8::/32 any
Device(config-ipv6-acl)# permit udp 2001:DB8::/32 any
Device(config-ipv6-acl)# end
```

Example: Controlling Access to a vty

In the following example, incoming connections to the virtual terminal lines 0 to 4 are filtered based on the IPv6 access list named acl1:

```
ipv6 access-list acl1
 permit ipv6 host 2001:DB8:0:4::2/32 any
!
line vty 0 4
 ipv6 access-class acl1 in
```

Use Case or Deployment Scenarios

Creating and Applying a IPv6 Access List

```
Device(config)#ipv6 access-list test
Device(config-ipv6-acl)#deny ipv6 10:10:10:10::/64 20:20:20:20::/64 log-input
Device(config-ipv6-acl)#permit ipv6 any any log
Device(config-ipv6-acl)#exit
Device(config)#int gig 0/0/5
Device(config-if)#ipv6 traffic-filter test in
Device(config-if)#end

Device# show ipv6 access-list
IPv6 access list test
  deny ipv6 10:10:10:10::/64 20:20:20:20::/64 log-input (100 matches) sequence 10
  permit ipv6 any any log sequence 20

Device# show ipv6 access-list test
IPv6 access list test
  deny ipv6 10:10:10:10::/64 20:20:20:20::/64 log-input (100 matches) sequence 10
  permit ipv6 any any log sequence 20

Device#show platform hardware pp active feature acl ipv6 test
ACL Name: test

Independent TCAM Consumption: 3
Unique Combination Count: 1
Total TCAM Consumption: 3
Ingress Label: 1

Ingress Bind Point Details:

Interface: GigabitEthernet0/0/5 :
IPv4 ACL: None (Label: 0)
ACL Handle: 1

Device#show platform hardware pp active feature acl l3-acl aclhandles
Details of RSP3 L3 ACL Handles
```

ACL Handle	Ref Count	IPv4 ACL IPv6 ACL Label	IPv4 ACL Label	IPv6 ACL
1	1	None	0	test

```
Device#show platform hardware pp active feature acl l3-acl resource-summary
RSP3 L3 ACL Resource Summary
```

Type	Total	Used	Free
ACL Handles	512	1	511
IPv4 ACL TCAM	1000	16	984
IPv6 ACL TCAM	128	7	121

Additional References

Related Documents

Related Topic	Document Title
IP access list commands	<i>Cisco IOS Security Command Reference</i>
Configuring IP access lists	<i>Creating an IP Access List and Applying It to an Interface</i>

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html



CHAPTER 5

MAC Access Control Lists

This chapter describes how to configure MAC access control lists (ACLs) on a Cisco router. It contains the following sections:

- [Finding Feature Information, on page 55](#)
- [Prerequisites for MAC Access Control Lists, on page 55](#)
- [Restrictions for MAC Access Control Lists, on page 55](#)
- [Information About MAC Access Control Lists, on page 56](#)
- [How to Configure MAC Access Control Lists, on page 56](#)
- [Configuration Examples for MAC Access Control Lists, on page 60](#)
- [Additional References for MAC Access Control Lists, on page 60](#)

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see [Bug Search Tool](#) and the release notes for your platform and 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 feature information table.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to <https://cfng.cisco.com/>. An account on Cisco.com is not required.

Prerequisites for MAC Access Control Lists

- You must be familiar with MAC addressing and non-IP protocols to configure MAC ACLs.

Restrictions for MAC Access Control Lists

- MAC ACL is not supported on:
 - Trunk EFP or trunk port
 - Egress
 - Interface

- MAC ACL is not supported for IP packets.
- MAC ACL counters are not supported.
- MAC ACLs are not supported on port, routed interface, and BDI.
- ACL and QoS cannot be applied on the same interface, EFP and bridge domain interface (BDI).
- Outbound MAC ACL is not supported on the Cisco RSP3 Module.
- MAC ACL does not deny broadcast packets on the EFP on the RSP3 module. To deny broadcast packets do any of the following:
 - Use storm control feature to restrict the broadcast packets.
 - Create a policy with the ACL (permit broadcast), and apply it to the interface with the 64K policer, to rate limit and allow only minimal amount of broadcast packets.

Information About MAC Access Control Lists

MAC Access Control Lists

MAC ACLs are ACLs that filter traffic using information in the Layer 2 header of each packet. You can use ACLs to control which hosts can access different parts of a network or to decide which types of traffic are forwarded or blocked at the router interfaces. MAC ACL is supported on EFP and Cross-Connect.

How to Configure MAC Access Control Lists

Configuring ACL

To configure ACL, perform the steps below.

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

Enters global configuration mode.

Step 3 **mac access-list extended *name***

Example:

```
Router(config)# mac access-list ext macext2
```

Creates an extended MAC access control list (ACL) and define its access control entries (ACEs).

- *name*—Name of the ACL to which the entry belongs.

Step 4 `{permit | deny} {any | host src-MAC-addr} {any | host dst-MAC-addr}`

Example:

```
Router(config-ext-macl)# deny any any
```

Permits or denies Layer 2 traffic to be forwarded if the conditions are matched.

- **permit**—Permits Layer 2 traffic to be forwarded if the conditions are matched.
- **deny**—Denies Layer 2 traffic to be forwarded if the conditions are matched.
- **any**—Keyword to deny any source or destination MAC address.
- **host *src-MAC-addr***—Defines a host MAC address. MAC address-based subnets are not allowed.
- **host *dst-MAC-addr***—Defines a destination MAC address. MAC address-based subnets are not allowed.

Step 5 `end`

Example:

```
Router(config-ext-macl)# end
```

Returns to privileged EXEC mode.

Applying ACL on Cross-Connect

To apply ACL on Cross-Connect, perform the steps below.

SUMMARY STEPS

1. `enable`
2. `configure terminal`
3. `interface gigabitethernet slot/subslot/port`
4. `service instance id ethernet`
5. `description string`
6. `encapsulation dot1q sp-vlan-id second-dot1q { ce-vlan-id | any }`
7. `rewrite ingress tag {pop {1 | 2}} [symmetric]`
8. `mac access-group access-list-name in`
9. `xconnect peer-router-id vcid encapsulation mpls`
10. `mtu mtu-value`
11. `exit`
12. `service instance id ethernet`
13. `encapsulation dot1q vlan-id`
14. `mac access-group access-list-name in`
15. `bridge-domain bridge-id`
16. `end`

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable Example: Router> enable	Enables privileged EXEC mode. <ul style="list-style-type: none">• Enter your password if prompted.
Step 2	configure terminal Example: Router# configure terminal	Enters global configuration mode.
Step 3	interface gigabitethernet slot/subslot/port Example: Router(config)# interface gigabitEthernet 0/2/1	Specifies an interface and enters interface configuration mode. <ul style="list-style-type: none">• <i>slot/subslot/port</i>—The location of the interface.
Step 4	service instance id ethernet Example: Router(config-if)# service instance 1120 ethernet	Configures an Ethernet service instance on an interface and enters service instance configuration mode. <ul style="list-style-type: none">• <i>id</i>—Specifies the EFP identifier, an integer from 1 to 4000.
Step 5	description string Example: Router(config-if-srv)# description sBROBA 41CHE Shared VLAN highest priority	Adds a description about an interface to help you remember its function. <ul style="list-style-type: none">• <i>string</i>—Adds a description (up to 240 characters) for an interface.
Step 6	encapsulation dot1q sp-vlan-id second-dot1q { ce-vlan-id any } Example: Router(config-if-srv)# encapsulation dot1q 141 second-dot1q 120	Enables stacked VLAN (802.1Q-in-Q) processing of customer VLAN traffic on an Ethernet subinterface. <ul style="list-style-type: none">• <i>sp-vlan-id</i>—Virtual LAN identifier of the unique service-provider VLAN used in 802.1Q-in-Q encapsulation of Ethernet traffic from the VLANs of a customer. The valid values are from 1 to 4095.• <i>ce-vlan-id</i>—Virtual LAN identifier of a customer VLAN encapsulated with the service-provider VLAN ID specified by <i>sp-vlan-id</i> in stacked VLAN (802.1Q-in-Q) processing. The valid values are from 1 to 4095. This argument is the inner VLAN tag in 802.1Q-in-Q headers.• any—Configures stacked VLAN processing for all customer VLAN IDs encapsulated with the specified service-provider VLAN ID that are not specified in a separate encapsulation dot1q second-dot1q command on another subinterface.
Step 7	rewrite ingress tag {pop {1 2}} [symmetric] Example: Router(config-if-srv)# rewrite ingress tag pop 2 symmetric	Specifies the encapsulation adjustment that is to be performed on the frame ingress to the service instance. <ul style="list-style-type: none">• pop {1 2}—One or two tags are removed from the packet. This command can be combined with a push (pop N and subsequent push vlan-id).

	Command or Action	Purpose
		<ul style="list-style-type: none"> • symmetric—(Optional) A rewrite operation is applied on both ingress and egress. The operation on egress is the inverse operation as ingress.
Step 8	mac access-group <i>access-list-name</i> in Example: <pre>Router(config-if-srv)# mac access-group macext2 in</pre>	To use a MAC access control list (ACL) to control inbound traffic on an Ethernet service instance. <ul style="list-style-type: none"> • <i>access-list-name</i>—Name of a MAC ACL to apply to an interface or subinterface (as specified by the mac access-list extended command). • in—Filters on inbound packets.
Step 9	xconnect <i>peer-router-id</i> <i>vcid</i> encapsulation mpls Example: <pre>Router(config-if-srv)# xconnect 77.77.77.77 1120 encapsulation mpls</pre>	Binds the attachment circuit to a pseudowire VC. The syntax for this command is the same as for all other Layer 2 transports. Enters cross-connect configuration mode. <ul style="list-style-type: none"> • <i>peer-router-id</i>—IP address of the remote provider edge (PE) peer router. • <i>vcid</i>—32-bit identifier to assign to the pseudowire.
Step 10	mtu <i>mtu-value</i> Example: <pre>Router(config-if-xconn)# mtu 1500</pre>	Specifies the MTU for the VC. <ul style="list-style-type: none"> • <i>mtu-value</i>—Specifies the value of the MTU.
Step 11	exit Example: <pre>Router(config-if-xconn)# exit</pre>	Exits cross-connect configuration mode.
Step 12	service instance <i>id</i> ethernet Example: <pre>Router(config-if)# service instance 1000 ethernet</pre>	Configures an Ethernet service instance on an interface and enters service instance configuration mode. <ul style="list-style-type: none"> • <i>id</i>—Specifies the EFP identifier, an integer from 1 to 4000.
Step 13	encapsulation dot1q <i>vlan-id</i> Example: <pre>Router(config-if-srv)# encapsulation dot1q 141 second-dot1q 120</pre>	To define the matching criteria to map 802.1Q frames ingress on an interface to the appropriate service instance. <ul style="list-style-type: none"> • <i>vlan-id</i>—VLAN ID, integer in the range 1 to 4094.
Step 14	mac access-group <i>access-list-name</i> in Example: <pre>Router(config-if-srv)# mac access-group macext2 in</pre>	To use a MAC access control list (ACL) to control inbound traffic on an Ethernet service instance. <ul style="list-style-type: none"> • <i>access-list-name</i>—Name of a MAC ACL to apply to an interface or subinterface (as specified by the mac access-list extended command). • in—Filters on inbound packets.
Step 15	bridge-domain <i>bridge-id</i> Example: <pre>Router(config-if-srv)# bridge-domain 1000</pre>	Binds a service instance or a MAC tunnel to a bridge domain instance. <ul style="list-style-type: none"> • <i>bridge-id</i>—Numerical identifier for the bridge domain instance. The range is an integer from 1 to 4000.

	Command or Action	Purpose
Step 16	end Example: Router(config-if-srv)# end	Exits service instance mode and returns to privileged EXEC mode.

Verifying MAC Access Control Lists

To verify the MAC ACL configuration, use the following **show** command.

- **show access-lists** *name*—Displays information about the named access list.

```
Router# show access-list macext4
```

```
Extended MAC access list macext4
  permit any host 0000.0000.0009
  permit any host 0000.0000.0010
  permit any host 0000.0000.0011
  permit any host 0000.0000.0012
```

Configuration Examples for MAC Access Control Lists

MAC ACL Configuration

Example: Allowing Specified Source or Destination MAC Addresses

```
(config)#mac access-list extended macext5
(config-ext-macl)#permit any host 0000.0000.0009
(config-ext-macl)#permit any host 0000.0000.0010
(config-ext-macl)#permit any host 0000.0000.0011
(config-ext-macl)#permit any host 0000.0000.0012
```

Example: Allowing any Source or Destination MAC Address

```
(config)#mac access-list extended macext9
(config-ext-macl)#permit any any
```

Additional References for MAC Access Control Lists

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases

Standards and RFCs

Standard/RFC	Title
Standard	—

MIBs

MIB	MIBs Link
• CCMB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
<p>The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies.</p> <p>To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds.</p> <p>Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.</p>	http://www.cisco.com/cisco/web/support/index.html

