Security Configuration Guide: Access Control Lists, Cisco IOS XE Release 3E

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

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

CONTENTS

CHAPTER 1

Commented IP Access List Entries

The Commented IP Access List Entries feature allows you to include comments or remarks about **deny** or **permit** conditions in any IP access list. These remarks make access lists easier for network administrators to understand. Each remark is limited to 100 characters in length.

This module provides information about the Commented IP Access List Entries feature.

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 www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Commented IP Access List Entries

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.

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
```

How to Configure Commented IP Access List Entries

Writing Remarks in a Named or Numbered Access List

You can use a named or numbered access list configuration. You must apply the access list to an interface or terminal line after the access list is created for the configuration to work.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list** {**standard** | **extended**} {*name* | *number*}
4. **remark** *remark*
5. **deny** *protocol* **host** *host-address* **any eq** *port*
6. **end**

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** {**standard** | **extended**} {*name* | *number*}
	Example:
	`Device(config)# ip access-list extended telnetting`
		Identifies the access list by a name or number and enters extended named access list configuration mode.

Step 4	**remark** *remark* **Example:** `Device(config-ext-nacl)# remark Do not allow host1 subnet to telnet out`	Adds a remark for an entry in a named IP access list. • The remark indicates the purpose of the **permit** or **deny** statement.
Step 5	**deny** *protocol* **host** *host-address* **any eq** *port* **Example:** `Device(config-ext-nacl)# deny tcp host 172.16.2.88 any eq telnet`	Sets conditions in a named IP access list that denies packets.

	Command or Action	Purpose

Step 6	**end**
	Example:
	`Device(config-ext-nacl)# end`
		Exits extended named access list configuration mode and enters privileged EXEC mode.

Configuration Examples for Commented IP Access List Entries

Example: Writing Remarks in an IP Access List

```
Device# configure terminal
Device(config)# ip access-list extended telnetting
Device(config-ext-nacl)# remark Do not allow host1 subnet to telnet out
Device(config-ext-nacl)# deny tcp host 172.16.2.88 any eq telnet
Device(config-ext-nacl)# end
```

Additional References for Commented IP Access List Entries

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Security commands	• 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

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

Feature Information for Commented IP Access List Entries

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 1: Feature Information for Commented IP Access List Entries

Feature Name	Releases	Feature Information
Commented IP Access List Entries	Cisco IOS XE Release 3.5E Cisco IOS XE 3.6E	The Commented IP Access List Entries feature allows you to include comments or remarks about **deny** or **permit** conditions in any IP access list. These remarks make access lists easier for network administrators to understand. Each remark is limited to 100 characters in length. In Cisco IOS XE Release 3.2SE and Cisco IOS XE Release 3.6E, support was added for the Cisco Catalyst 3850 Series Switches. The following command was introduced or modified: **remark**.

CHAPTER 2

Configuring an FQDN ACL

This document describes how to configure an access control lists (ACL) using a fully qualified domain name (FQDN). The Configuring an FQDN ACL feature allows you to configure and apply an ACL to a wireless session based on the domain name system (DNS). The domain names are resolved to IP addresses, the IP addresses are given to the client as part of the DNS response, and the FQDN is then mapped to an ACL based on the IP address.

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 www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Configuring FQDN ACL

The Configuring FQDN ACL feature is supported only on IPv4 wireless sessions.

Information About Configuring an FQDN ACL

Configuring an FQDN ACL

When access control lists (ACLs) are configured using a fully qualified domain name (FQDN), ACLs can be applied based on the destination domain name. The destination domain name is then resolved to an IP address, which is provided to the client as a part of the DNS response.

Guest users can log in using web authentication with a parameter map that consists of an FQDN ACL name.

Before you configure an FQDN ACL, complete the following tasks:

- Configure an IP access list.
- Configure an IP domain name list.
- Map an FQDN ACL with a domain name.

You can apply an access list to a specific domain by configuring the RADIUS server to send the **fqdn-acl-name** AAA attribute to the controller. The operating system checks for the passthrough domain list and its mapping, and permits the FQDN. The FQDN ACL allows clients to access only configured domains without authentication.

Note By default, an IP access list name is configured with the same name as the pass-through domain name. To override the default name, you can use the **access-session passthrou-access-group** *access-group-name* **passthrou-domain-list** *domain-list-name* command in global configuration mode.

How to Configure FQDN ACL

Configuring an IP Access List

SUMMARY STEPS

1. **configure terminal**
2. **ip access-list extended** *name*
3. **permit ip any any**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `# configure terminal`	Enters global configuration mode.
Step 2	**ip access-list extended** *name* **Example:** `(config)# ip access-list extended ABC`	Creates the IP access list.
Step 3	**permit ip any any** **Example:** `(config-ext-nacl)# permit ip any any`	Specifies the domains to be allowed for the wireless client. The domains are specified in the domain name list.

Step 4	**end**
	Example:
	`(config)# end`
		Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.

Configuring a Domain Name List

You can configure a domain name list that contains a list of domain names that are allowed for DNS snooping by the access point. The DNS domain list name string must be identical to the extended access list name.

SUMMARY STEPS

1. **configure terminal**
2. **passthrou-domain-list** *name*
3. **match** *word*
4. **end**

DETAILED STEPS

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

	Command or Action	Purpose

Step 2	**passthrou-domain-list** *name*
	Example:
	`(config)# passthrou-domain-list abc`
	`(config-fqdn-acl-domains)#`
		Configures a passthrough domain name list.

Step 3	**match** *word*
	Example:
	`(config-fqdn-acl-domains)# match play.google.com`
	`(config-fqdn-acl-domains)# match www.yahoo.com`
		Configures a passthrough domain list. Adds a list of websites that the client is allowed to query for access without first being required to be authenticated through the RADIUS server.

Step 4	**end**
	Example:
	`(config)# end`
		Returns to privileged EXEC mode. Alternatively, you can also press **Ctrl-Z** to exit global configuration mode.

Mapping the FQDN ACL with a Domain Name

SUMMARY STEPS

1. **configure terminal**
2. **access-session passthrou-access-group** *access-group-name* **passthrou-domain-list** *domain-list-name*
3. **parameter-map type webauth** *domain-list-name* and **login-auth-bypass fqdn-acl-name** *acl-name* **domain-name** *domain-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `# configure terminal`	Enters global configuration mode.
Step 2	**access-session passthrou-access-group** *access-group-name* **passthrou-domain-list** *domain-list-name* **Example:** `(config)# access-session passthrou-access-group abc passthrou-domain-list abc`	Maps the FQDN ACL AAA attribute name with the domain name list. Use this command when configuring central web authentication.

	Command or Action	Purpose

Step 3	**parameter-map type webauth** *domain-list-name* and **login-auth-bypass fqdn-acl-name** *acl-name* **domain-name** *domain-name*
	Example:
	`(config)# parameter-map type webauth abc`
	`(config-params-parameter-map)# login-auth-bypass fqdn-acl-name abc domain-name abc`
		Maps an FQDN ACL name with the domain name list. Use the command when configuring local authentication on the controller.
		The RADIUS server can be configured to return an FQDN ACL name as part of the authenticated user profile. The controller dynamically applies the FQDN ACL to the user if the FQDN ACL is defined on the controller.

Monitoring an FQDN ACL

The following commands can be used to monitor FQDN ACLs.

Command	Purpose
show access-session interface *interface-name* **details**	Displays the FQDN ACL information configured on the interface.
show access-session fqdn fqdn-maps	Displays the FQDN ACL mapped to the domain name list.
show access-session fqdn list-domain *domain-name*	Displays the domain names.
show access-session fqdn passthru-domain-list	Displays the domains that are configured.

Configuration Examples for an FQDN ACL

Examples: FQDN ACL Configuration

This example shows how to create IP access list:

```
# config terminal
(config)# ip access-list extended abc
(config-ext-nacl)# permit ip any any
(config-ext-nacl)# end
# show ip access-list abc
```

This example shows how to configure domain name list:

```
# config terminal
(config)# passthrou-domain-list abc
(config-fqdn-acl-domains)# match play.google.com
(config-fqdn-acl-domains)# end
# show access-session fqdn fqdn-maps
```

This example shows how to map FQDN ACL with domain name using central web authentication:

```
# config terminal
(config)# access-session passthrou-access-group abc passthrou-domain-list abc
(config)# end
# show access-session interface vlan 20
```

This example shows how to map FQDN ACL with domain name using local authentication:

```
# config terminal
(config)# parameter-map type webauth abc
(config-params-parameter-map)# login-auth-bypass fqdn-acl-name abc domain-name abc
(config-params-parameter-map)# end
# show access-session fqdn fqdn-maps
```

Additional References for Configuring FQDN ACL

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Security commands	• 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
ACL configuration guide	*Security Configuration Guide: Access Control Lists*

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

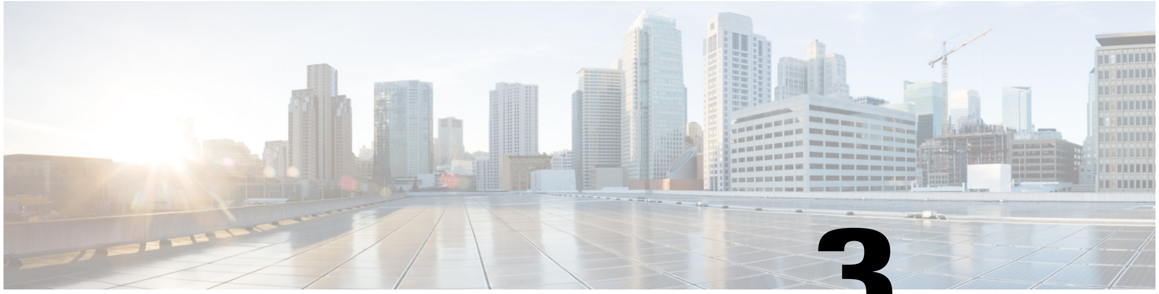
Feature Information for Configuring FQDN ACL

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 2: Feature Information for Configuring FQDN ACL

Feature Name	Releases	Feature Information
Configuring an FQDN ACL	Cisco IOS XE 3.6E	The Configuring an FQDN ACL feature allows you to configure and apply an access control lists (ACL) to a wireless session based on the domain name system (DNS). The domain names are resolved to IP addresses, where the IP addresses are given to the client as part of the DNS response; the FQDN is then mapped to an ACL based on the IP address. In Cisco IOS XE Release 3.6E, this feature is supported on Cisco Catalyst 3850 Series Switches. The following commands were introduced or modified: **access session passthrou access group**, **login-auth-bypass**, **parameter-map type webauth global**, **pass throu domain list name**, **show access-session fqdn**.



CHAPTER 3

IP Access List Entry Sequence Numbering

Users can apply sequence numbers to **permit** or **deny** statements and also reorder, add, or remove such statements from a named IP access list. This feature makes revising IP access lists much easier. Prior to this feature, users could add access list entries to the end of an access list only; therefore needing to add statements anywhere except the end required reconfiguring the access list entirely.

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 www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for IP Access List Entry Sequence Numbering

- This feature does not support dynamic, reflexive, or firewall access lists.
- This feature does not support old-style numbered access lists, which existed before named access lists. Keep in mind that you can name an access list with a number, so numbers are allowed when they are entered in the standard or extended named access list (NACL) configuration mode.

Information About IP Access List Entry Sequence Numbering

Purpose of IP Access Lists

Access lists perform packet filtering to control which packets move through the network and where. Such control can help limit network traffic and restrict the access of users and devices to the network. Access lists have many uses, and therefore many commands accept a reference to an access list in their command syntax. Access lists can be used to do the following:

- Filter incoming packets on an interface.
- Filter outgoing packets on an interface.
- Restrict the contents of routing updates.
- Limit debug output based on an address or protocol.
- Control virtual terminal line access.
- Identify or classify traffic for advanced features, such as congestion avoidance, congestion management, and priority and custom queuing.
- Trigger dial-on-demand routing (DDR) calls.

How an IP Access List Works

An access list is a sequential list consisting of at least one **permit** statement and possibly one or more **deny** statements that apply to IP addresses and possibly upper-layer IP protocols. The access list has a name by which it is referenced. Many software commands accept an access list as part of their syntax.

An access list can be configured and named, but it is not in effect until the access list is referenced by a command that accepts an access list. Multiple commands can reference the same access list. An access list can control traffic arriving at the device or leaving the device, but not traffic originating at the device.

IP Access List Process and Rules

- The software tests the source or destination address or the protocol of each packet being filtered against the conditions in the access list, one condition (**permit** or **deny** statement) at a time.
- If a packet does not match an access list statement, the packet is then tested against the next statement in the list.
- If a packet and an access list statement match, the rest of the statements in the list are skipped and the packet is permitted or denied as specified in the matched statement. The first entry that the packet matches determines whether the software permits or denies the packet. That is, after the first match, no subsequent entries are considered.
- If the access list denies the address or protocol, the software discards the packet and returns an ICMP Host Unreachable message.

- If no conditions match, the software drops the packet. This is because each access list ends with an unwritten or implicit **deny** statement. That is, if the packet has not been permitted by the time it was tested against each statement, it is denied.
- The access list must contain at least one **permit** statement or else all packets are denied.
- Because the software stops testing conditions after the first match, the order of the conditions is critical. The same **permit** or **deny** statements specified in a different order could result in a packet being passed under one circumstance and denied in another circumstance.
- If an access list is referenced by name in a command, but the access list does not exist, all packets pass.
- Only one access list per interface, per protocol, per direction is allowed.
- Inbound access lists process packets arriving at the device. Incoming packets are processed before being routed to an outbound interface. An inbound access list is efficient because it saves the overhead of routing lookups if the packet is to be discarded because it is denied by the filtering tests. If the packet is permitted by the tests, it is then processed for routing. For inbound lists, **permit** means continue to process the packet after receiving it on an inbound interface; **deny** means discard the packet.
- Outbound access lists process packets before they leave the device. Incoming packets are routed to the outbound interface and then processed through the outbound access list. For outbound lists, **permit** means send it to the output buffer; **deny** means discard the packet.

Helpful Hints for Creating IP Access Lists

- Create the access list before applying it to an interface. An interface with an empty access list applied to it permits all traffic.
- Another reason to configure an access list before applying it is because if you applied 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.
- Because the software stops testing conditions after it encounters the first match (to either a **permit** or **deny** statement), you will reduce processing time and resources if you put the statements that packets are most likely to match at the beginning of the access list. Place more frequently occurring conditions before less frequent conditions.
- Organize your access list so that more specific references in a network or subnet appear before more general ones.
- In order to make the purpose of individual statements more easily understood at a glance, you can write a helpful remark before or after any statement.

Source and Destination Addresses

Source and destination address fields in an IP packet are two typical fields on which to base an access list. Specify source addresses to control the packets being sent from certain networking devices or hosts. Specify destination addresses to control the packets being sent to certain networking devices or hosts.

Wildcard Mask and Implicit Wildcard Mask

When comparing the address bits in an access list entry to a packet being submitted to the access list, address filtering uses wildcard masking to determine whether to check or ignore the corresponding IP address bits. By carefully setting wildcard masks, an administrator can select 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.
- A wildcard mask bit 1 means ignore that corresponding bit value.

If you do not supply a wildcard mask with a source or destination address in an access list statement, the software assumes a default wildcard mask of 0.0.0.0.

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

Transport Layer Information

You can filter packets based on transport layer information, such as whether the packet is a TCP, UDP, Internet Control Message Protocol (ICMP) or Internet Group Management Protocol (IGMP) packet.

IP Access List Entry Sequence Numbering

Benefits

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 a user wanted to insert an entry (statement) 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 a user adds a new entry, the user chooses 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.

Sequence Numbering Behavior

- For backward compatibility with previous releases, if entries with no sequence numbers are applied, the first entry is assigned a sequence number of 10, and successive entries are incremented by 10. The maximum sequence number is 2147483647. If the generated sequence number exceeds this maximum number, the following message is displayed:

```
Exceeded maximum sequence number.
```

- If you enter an entry without a sequence number, it is assigned a sequence number that is 10 greater than the last sequence number in that access list and is placed at the end of the list.
- If you enter an entry that matches an already existing entry (except for the sequence number), then no changes are made.
- If you enter a sequence number that is already present, the following error message is generated:

```
Duplicate sequence number.
```

- If a new access list is entered from global configuration mode, then sequence numbers for that access list are generated automatically.
- Distributed support is provided so that the sequence numbers of entries in the Route Processor (RP) and line card (LC) are always synchronized.
- Sequence numbers are not nvgened. That is, the sequence numbers themselves are not saved. In the event that the system is reloaded, the configured sequence numbers revert to the default sequence starting number and increment from that number. The function is provided for backward compatibility with software releases that do not support sequence numbering.
- The IP Access List Entry Sequence Numbering feature works with named standard and extended IP access lists. Because the name of an access list can be designated as a number, numbers are acceptable.

How to Use Sequence Numbers in an IP Access List

Sequencing Access-List Entries and Revising the Access List

This task shows how to assign sequence numbers to entries in a named IP access list and how to add or delete an entry to or from an access list. It is assumed a user wants to revise an access list. The context of this task is the following:

- A user need not resequence access lists for no reason; resequencing in general is optional. The resequencing step in this task is shown as required because that is one purpose of this feature and this task demonstrates the feature.
- Step 5 happens to be a **permit** statement and Step 6 happens to be a **deny** statement, but they need not be in that order.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list resequence** *access-list-name starting-sequence-number increment*
4. **ip access-list** {**standard**| **extended**} *access-list-name*
5. Do one of the following:
   - *sequence-number* **permit** *source source-wildcard*
   - *sequence-number* **permit** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*][**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]
6. Do one of the following:
   - *sequence-number* **deny** *source source-wildcard*
   - *sequence-number* **deny** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*][**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]
7. Repeat Step 5 and/or Step 6 as necessary, adding statements by sequence number where you planned. Use the **no** *sequence-number* command to delete an entry.
8. **end**
9. **show ip access-lists** *access-list-name*

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 resequence** *access-list-name starting-sequence-number increment* **Example:** `Device(config)# ip access-list resequence kmd1 100 15`	Resequences the specified IP access list using the starting sequence number and the increment of sequence numbers. • This example resequences an access list named kmd1. The starting sequence number is 100 and the increment is 15.
Step 4	**ip access-list** {**standard**| **extended**} *access-list-name*	Specifies the IP access list by name and enters named access list configuration mode.

	Command or Action	Purpose

	Example:
	`Device(config)# ip access-list standard kmd1`
	`Device(config)# ip access-list extended kmd1`
		• If you specify **standard**, make sure you subsequently specify **permit** and/or **deny** statements using the standard access list syntax.
		• If you specify **extended**, make sure you subsequently specify **permit** and/or **deny** statements using the extended access list syntax.

Step 5	Do one of the following: • *sequence-number* **permit** *source source-wildcard* • *sequence-number* **permit** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*][**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**] **Example:** `Device(config-std-nacl)# 105 permit 10.5.5.5 0.0.0 255`	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. • See the permit (IP) command for additional command syntax to permit upper layer protocols (ICMP, IGMP, TCP, and UDP). • Use the **no** *sequence-number* command to delete an entry. • As the prompt indicates, this access list was a standard access list. If you had specified **extended** in Step 4, the prompt for this step would be Device(config-ext-nacl) and you would use the extended **permit** command syntax.
Step 6	Do one of the following: • *sequence-number* **deny** *source source-wildcard* • *sequence-number* **deny** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*][**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**] **Example:** `Device(config-std-nacl)# 105 deny 10.6.6.7 0.0.0 255`	(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. • See the deny (IP) command for additional command syntax to permit upper layer protocols (ICMP, IGMP, TCP, and UDP). • Use the **no** *sequence-number* command to delete an entry. • As the prompt indicates, this access list was a standard access list. If you had specified **extended** in Step 4, the prompt for this step would be Device(config-ext-nacl) and you would use the extended **deny** command syntax.
Step 7	Repeat Step 5 and/or Step 6 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 8	**end**
	Example:
	`Device(config-std-nacl)# end`
		(Optional) Exits the configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose

Step 9	**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 see that the access list includes the new entry.
		`Device#` **show ip access-lists kmd1**
		`Standard IP access list kmd1`
		`100 permit 10.4.4.0, wildcard bits 0.0.0.255`
		`105 permit 10.5.5.0, wildcard bits 0.0.0.255`
		`115 permit 10.0.0.0, wildcard bits 0.0.0.255`
		`130 permit 10.5.5.0, wildcard bits 0.0.0.255`
		`145 permit 10.0.0.0, wildcard bits 0.0.0.255`

What to Do Next

If your access list is not already applied to an interface or line or otherwise referenced, apply the access list. Refer to the "Configuring IP Services" chapter of the *Cisco IOS IP Configuration Guide* for information about how to apply an IP access list.

Configuration Examples for IP Access List Entry Sequence Numbering

Example: Resequencing Entries in an Access List

The following example shows access list resequencing. The starting value is 1, and increment value is 2. The subsequent entries are ordered based on the increment values that users provide, and the range is from 1 to 2147483647.

When an entry with no sequence number is entered, by default it has a sequence number of 10 more than the last entry in the access list.

```
Device# show access-list 150
Extended IP access list 150
    10 permit ip host 10.3.3.3 host 172.16.5.34
    20 permit icmp any any
    30 permit tcp any host 10.3.3.3
    40 permit ip host 10.4.4.4 any
    50 Dynamic test permit ip any any
    60 permit ip host 172.16.2.2 host 10.3.3.12
    70 permit ip host 10.3.3.3 any log
    80 permit tcp host 10.3.3.3 host 10.1.2.2
    90 permit ip host 10.3.3.3 any
    100 permit ip any any
Device(config)# ip access-list extended 150
Device(config)# ip access-list resequence 150 1 2
Device(config)# end
Device# show access-list 150
Extended IP access list 150
    1 permit ip host 10.3.3.3 host 172.16.5.34
    3 permit icmp any any
    10 permit tcp any any eq 22 log
    7 permit ip host 10.4.4.4 any
    9 Dynamic test permit ip any any
    11 permit ip host 172.16.2.2 host 10.3.3.12
    13 permit ip host 10.3.3.3 any log
    15 permit tcp host 10.3.3.3 host 10.1.2.2
    17 permit ip host 10.3.3.3 any
    19 permit ip any any
```

Example: Adding Entries with Sequence Numbers

In the following example, a new entry is added to a specified access list:

```
Device# show ip access-list
Standard IP access list tryon
2 permit 10.4.4.2, wildcard bits 0.0.255.255
5 permit 10.0.0.44, wildcard bits 0.0.0.255
10 permit 10.0.0.1, wildcard bits 0.0.0.255
20 permit 10.0.0.2, wildcard bits 0.0.0.255
Device(config)# ip access-list standard tryon
Device(config-std-nacl)# 15 permit 10.5.5.5 0.0.0.255
Device# show ip access-list
Standard IP access list tryon
2 permit 10.4.0.0, wildcard bits 0.0.255.255
5 permit 10.0.0.0, wildcard bits 0.0.0.255
10 permit 10.0.0.0, wildcard bits 0.0.0.255
15 permit 10.5.5.0, wildcard bits 0.0.0.255
20 permit 10.0.0.0, wildcard bits 0.0.0.255
```

Example: Entry without Sequence Number

The following example shows how an entry with no specified sequence number is added to the end of an access list. When an entry is added without a sequence number, it is automatically given a sequence number that puts it at the end of the access list. Because the default increment is 10, the entry will have a sequence number 10 higher than the last entry in the existing access list.

```
Device(config)# ip access-list standard 1
Device(config-std-nacl)# permit 1.1.1.1 0.0.0.255
Device(config-std-nacl)# permit 2.2.2.2 0.0.0.255
Device(config-std-nacl)# permit 3.3.3.3 0.0.0.255
Device# show access-list
Standard IP access list 1
10 permit 0.0.0.0, wildcard bits 0.0.0.255
20 permit 0.0.0.0, wildcard bits 0.0.0.255
30 permit 0.0.0.0, wildcard bits 0.0.0.255
Device(config)# ip access-list standard 1
Device(config-std-nacl)# permit 4.4.4.4 0.0.0.255
Device(config-std-nacl)# end
Device# show access-list
Standard IP access list 1
10 permit 0.0.0.0, wildcard bits 0.0.0.255
20 permit 0.0.0.0, wildcard bits 0.0.0.255
```

```
30 permit 0.0.0.0, wildcard bits 0.0.0.255
40 permit 0.4.0.0, wildcard bits 0.0.0.255
```

Additional References for IP Access List Entry Sequence Numbering

The following sections provide references related to IP access lists.

Related Documents

Related Topic	Document Title
Configuring IP access lists	"Creating an IP Access List and Applying It to an Interface"
Cisco IOS commands	Cisco IOS Master Command List, All Releases
IP access list commands	• 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

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. 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. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for IP Access List Entry Sequence Numbering

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 3: Feature Information for IP Access List Entry Sequence Numbering

Feature Name	Releases	Feature Information
IP Access List Entry Sequence Numbering	Cisco IOS XE 3.6E	Users can apply sequence numbers to **permit** or **deny** statements and also reorder, add, or remove such statements from a named IP access list. This feature makes revising IP access lists much easier. Prior to this feature, users could add access list entries to the end of an access list only; therefore needing to add statements anywhere except the end required reconfiguring the access list entirely. In , Cisco IOS XE 3.6E, support was added for the Cisco Catalyst 3850 Series Switches. The following commands were introduced or modified: **deny (IP)**, **ip access-list resequence deny (IP), permit (IP)**.

CHAPTER 4

ACL Support for Filtering IP Options

The ACL Support for Filtering IP Options feature describes how to use an IP access list to filter IP packets that contain IP options to prevent devices from becoming saturated with spurious packets.

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 www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for ACL Support for Filtering IP Options

Before you configure the ACL Support for Filtering IP Options feature, you must understand the concepts of the IP access lists.

Information About ACL Support for Filtering IP Options

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.

How to Configure ACL Support for Filtering IP Options

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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list extended** *access-list-name*
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**]
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**]
6. Repeat Step 4 or Step 5 as necessary.
7. **end**
8. **show ip access-lists** *access-list-name*

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** *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**]	(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.

	Command or Action	Purpose
	Example: `Device(config-ext-nacl)# deny ip any any option traceroute`	- 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.

Configuration Examples for ACL Support for Filtering IP Options

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)
```

Additional References for ACL Support for Filtering IP Options

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Security commands	• 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
Overview information about access lists	"IP Access List Overview"

Table 4: Standards and RFCs

Standards/RFCs	Title
RFC 791	*Internet Protocol*
RFC 793	*Transmission Control Protocol*
RFC 1393	*Traceroute Using an IP Option*

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

Feature Information for ACL Support for Filtering IP Options

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 5: Feature Information for ACL Support for Filtering IP Options

Feature Name	Releases	Feature Information
ACL Support for Filtering IP Options	Cisco IOS XE 3.6E	The ACL Support for Filtering IP Options feature describes how to use an IP access list to filter IP packets that contain IP options to prevent devices from becoming saturated with spurious packets. In Cisco IOS XE Release 3.6E, this feature is supported on Cisco Catalyst 3850 Series Switches.

CHAPTER 5

Creating an IP Access List to Filter TCP Flags

This module documents the ACL TCP Flags Filtering feature and describes how to use an IP access list to filter IP packets that contain TCP flags. 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

The ACL TCP Flags Filtering feature provides a flexible mechanism for filtering on TCP flags. Before this feature, an incoming packet was matched if any TCP flag in the packet matched a flag specified in the access control entry (ACE). This behavior allowed for a security loophole, because packets with all flags set could get past the access control list (ACL).

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 www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Creating an IP Access List to Filter TCP Flags

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 TCP Flags

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 6: 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.

TCP Flag	Purpose
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.

How to Create an IP Access List to Filter TCP Flags

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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list extended** *access-list-name*
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**]
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**]
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.
7. **end**
8. **show ip access-lists** *access-list-name*

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** *access-list-name* **Example:** `Device(config)# ip access-list extended kmd1`	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.

	Command or Action	Purpose
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.

Configuration Examples for Configuring an IP Access List to Filter TCP Flags

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:

```
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
```

Additional References for Creating an IP Access List to Filter TCP Flags

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Security Commands	• 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
Order of access list entries	"Refining an IP Access List"
Access list entries based on time of day or week	"Refining an IP Access List”
Packets with noninitial fragments	"Refining an IP Access List”
Filtering on IP Options, TCP flags, noncontiguous ports, or TTL values	“Creating an IP Access List to Filter IP Options, TCP Flags, Noncontiguous Ports, or TTL Values”
Access to virtual terminal lines	"Controlling Access to a Virtual Terminal Line”
Routing updates and policy routing	“Configuring Routing Protocol-Independent Features” modules in the *Cisco IOS IP Routing Protocols Configuration Guide*
Traffic identification or classification for features such as congestion avoidance, congestion management, and priority queuing	“Regulating Packet Flow on a Per-Interface Basis--Using Generic Traffic Shaping” module in the *Quality of Service Solutions Configuration Guide*

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

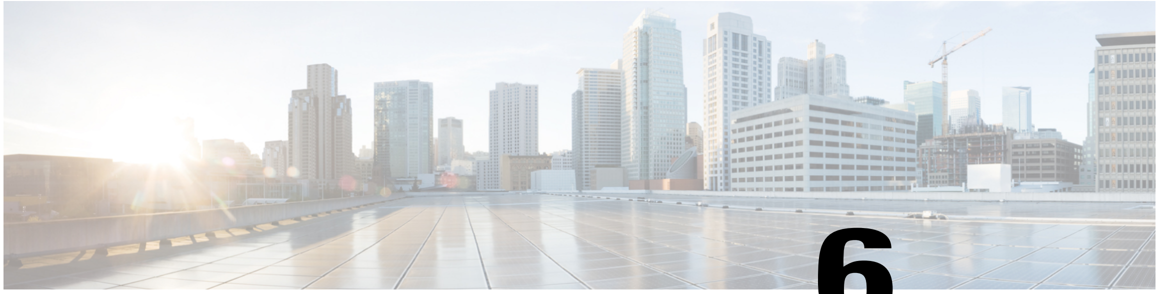
Feature Information for Creating an IP Access List to Filter TCP Flags

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 7: Feature Information for Creating an IP Access List to Filter TCP Flags

Feature Name	Releases	Feature Configuration Information
ACL TCP Flags Filtering	Cisco IOS XE RElease 3.6E	This feature provides a flexible mechanism for filtering on TCP flags. 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. In Cisco IOS XE Release 3.6E, this feature is supported on Cisco Catalyst 3850 Series Switches.



CHAPTER 6

Named ACL Support for Noncontiguous Ports on an Access Control Entry

The Named ACL Support for Noncontiguous Ports on an Access Control Entry feature allows you to specify noncontiguous ports in a single access control entry, which greatly reduces the number of entries required in an access control list when several entries have the same source address, destination address, and protocol, but differ only in the ports.

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 www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for Named ACL Support for Noncontiguous Ports on an Access Control Entry

Before you configure the ACL Support for Filtering IP Options feature, you must understand the concepts of the IP access lists.

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

Information About Named ACL Support for Noncontiguous Ports on an Access Control Entry

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 Configure Named ACL Support for Noncontiguous Ports on an Access Control 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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list extended** *access-list-name*
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**]
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**]
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.
7. **end**
8. **show ip access-lists** *access-list-name*

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** *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.

	Command or Action	Purpose
		• 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`	(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 kmd1`	(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.

SUMMARY STEPS

1. **enable**
2. **show ip access-lists** *access-list-name*
3. **configure terminal**
4. **ip access-list extended** *access-list-name*
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**]
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**]
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.
8. **end**
9. **show ip access-lists** *access-list-name*

DETAILED STEPS

	Command or Action	Purpose
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**]	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.

	Command or Action	Purpose
	Example: `Device(config-ext-nacl)# no 10`	• 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.

Configuration Examples for Named ACL Support for Noncontiguous Ports on an Access Control Entry

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 for Named ACL Support for Noncontiguous Ports on an Access Control Entry

Related Documents

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

Related Topic	Document Title
Overview information about access lists	"IP Access List Overview"

Table 8: Standards and RFCs

Standards/RFCs	Title
RFC 791	*Internet Protocol*
RFC 793	*Transmission Control Protocol*

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

Feature Information for Named ACL Support for Noncontiguous Ports on an Access Control Entry

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 9: Feature Information for ACL Support for Filtering IP Options

Feature Name	Releases	Feature Information
Named ACL Support for Noncontiguous Ports on an Access Control Entry	Cisco IOS XE Release 3.6E	The Named ACL Support for Noncontiguous Ports on an Access Control Entry feature allows you to specify noncontiguous ports in a single access control entry, which greatly reduces the number of entries required in an access control list when several entries have the same source address, destination address, and protocol, but differ only in the ports.

CHAPTER 7

Object Groups for ACLs

The Object Groups for ACLs feature lets you classify users, devices, or protocols into groups and apply those groups to access control lists (ACLs) to create access control policies for those groups. This feature lets you use object groups instead of individual IP addresses, protocols, and ports, which are used in conventional ACLs. This feature allows multiple access control entries (ACEs), but now you can use each ACE to allow an entire group of users to access a group of servers or services or to deny them from doing so.

In large networks, the number of ACLs can be large (hundreds of lines) and difficult to configure and manage, especially if the ACLs frequently change. Object group-based ACLs are smaller, more readable, and easier to configure and manage than conventional ACLs, simplifying static and dynamic ACL deployments for large user access environments on Cisco IOS routers.

Cisco IOS Firewall benefits from object groups, because they simplify policy creation (for example, group A has access to group A services).

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 www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Object Groups for ACLs

- You can use object groups only in extended named and numbered ACLs.
- Object group-based ACLs support only IPv4 addresses.
- Object group-based ACLs support only Layer 3 interfaces (such as routed interfaces and VLAN interfaces). Object group-based ACLs do not support Layer 2 features such as VLAN ACLs (VACLs) or port ACLs (PACLs).
- Object group-based ACLs are not supported with IPsec.
- The highest number of object group-based ACEs supported in an ACL is 2048.

Information About Object Groups for ACLs

You can configure conventional ACEs and ACEs that refer to object groups in the same ACL.

You can use object group-based ACLs with quality of service (QoS) match criteria, Cisco IOS Firewall, Dynamic Host Configuration Protocol (DHCP), and any other features that use extended ACLs. In addition, you can use object group-based ACLs with multicast traffic.

When there are many inbound and outbound packets, using object group-based ACLs increases performance when compared to conventional ACLs. Also, in large configurations, this feature reduces the storage needed in NVRAM, because using object groups in ACEs means that you do not need to define an individual ACE for every address and protocol pairing.

Object Groups

An object group can contain a single object (such as a single IP address, network, or subnet) or multiple objects (such as a combination of multiple IP addresses, networks, or subnets).

A typical access control entry (ACE) allows a group of users to have access only to a specific group of servers. In an object group-based access control list (ACL), you can create a single ACE that uses an object group name instead of creating many ACEs (which requires each ACE to have a different IP address). A similar object group (such as a protocol port group) can be extended to provide access only to a set of applications for a user group. ACEs can have object groups for the source only, destination only, none, or both.

You can use object groups to separate the ownership of the components of an ACE. For example, each department in an organization controls its group membership, and the administrator owns the ACE itself to control which departments can contact one another.

You can use object groups in features that use Cisco Policy Language (CPL) class maps.

This feature supports two types of object groups for grouping ACL parameters: network object groups and service object groups. Use these object groups to group IP addresses, protocols, protocol services (ports), and Internet Control Message Protocol (ICMP) types.

Objects Allowed in Network Object Groups

A network object group is a group of any of the following objects:

- Host IP addresses
- Network address of group members
- Nested object groups

Objects Allowed in Service Object Groups

A service object group is a group of any of the following objects:

- Source and destination protocol ports (such as Telnet or Simple Network Management Protocol [SNMP])
- Internet Control Message Protocol (ICMP) types (such as echo, echo-reply, or host-unreachable)
- Top-level protocols (such as Encapsulating Security Payload [ESP], TCP, or UDP)
- Other service object groups

ACLs Based on Object Groups

All features that use or reference conventional access control lists (ACLs) are compatible with object-group-based ACLs, and the feature interactions for conventional ACLs are the same with object-group-based ACLs. This feature extends the conventional ACLs to support object-group-based ACLs and also adds new keywords and the source and destination addresses and ports.

You can add, delete, or change objects in an object group membership list dynamically (without deleting and redefining the object group). Also, you can add, delete, or change objects in an object group membership list without redefining the ACL access control entry (ACE) that uses the object group. You can add objects to groups, delete them from groups, and then ensure that changes are correctly functioning within the object-group-based ACL without reapplying the ACL to the interface.

You can configure an object-group-based ACL multiple times with a source group only, a destination group only, or both source and destination groups.

You cannot delete an object group that is used within an ACL or a class-based policy language (CPL) policy.

How to Configure Object Groups for ACLs

To configure object groups for ACLs, you first create one or more object groups. These can be any combination of network object groups (groups that contain objects such as, host addresses and network addresses) or service object groups (which use operators such as **lt**, **eq**, **gt**, **neq**, and **range** with port numbers). Then, you create access control entries (ACEs) that apply a policy (such as **permit** or **deny**) to those object groups.

Creating a Network Object Group

A network object group that contains a single object (such as a single IP address, a hostname, another network object group, or a subnet) or nested objects (multiple network object groups can be defined in single network object group), is with a network object-group-based ACL to create access control policies for the objects.

Perform this task to create a network object group.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **object-group network** *object-group-name*
4. **description** *description-text*
5. **host** {*host-address* | *host-name*}
6. *network-address* {*/nn* | *network-mask*}
7. **group-object** *nested-object-group-name*
8. Repeat the steps until you have specified objects on which you want to base your object group.
9. **end**

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	**object-group network** *object-group-name* **Example:** `Device(config)# object-group network my-network-object-group`	Defines the object group name and enters network object-group configuration mode.
Step 4	**description** *description-text* **Example:** `Device(config-network-group)# description test engineers`	(Optional) Specifies a description of the object group. • You can use up to 200 characters.

Step 5	**host** {*host-address* | *host-name*}
	Example:
	`Device(config-network-group)# host 209.165.200.237`
		(Optional) Specifies the IP address or name of a host.
		• If you specify a host address, you must use an IPv4 address.

Step 6	*network-address* {*/nn* | *network-mask*}	(Optional) Specifies a subnet object.

	Command or Action	Purpose
	Example: `Device(config-network-group)# 209.165.200.241 255.255.255.224`	• You must specify an IPv4 address for the network address. The default network mask is 255.255.255.255.
Step 7	**group-object** *nested-object-group-name* **Example:** `Device(config-network-group)# group-object my-nested-object-group`	(Optional) Specifies a nested (child) object group to be included in the current (parent) object group. • The type of child object group must match that of the parent (for example, if you are creating a network object group, you must specify another network object group as the child). • You can use duplicated objects in an object group only via nesting of group objects. For example, if object 1 is in both group A and group B, you can define a group C that includes both A and B. However, you cannot include a group object that causes the group hierarchy to become circular (for example, you cannot include group A in group B and then also include group B in group A). • You can use an unlimited number of levels of nested object groups (however, a maximum of two levels is recommended).
Step 8	Repeat the steps until you have specified objects on which you want to base your object group.	—

Step 9	**end**
	Example:
	`Device(config-network-group)# end`
		Exits network object-group configuration mode and returns to privileged EXEC mode.

Creating a Service Object Group

Use a service object group to specify TCP and/or UDP ports or port ranges. When the service object group is associated with an access control list (ACL), this service object-group-based ACL can control access to ports.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **object-group service** *object-group-name*
4. **description** *description-text*
5. *protocol*
6. {**tcp** | **udp** | **tcp-udp**} [**source** {{[**eq**] | **lt** | **gt**} *port1* | **range** *port1 port2*}] [{[**eq**] | **lt** | **gt**} *port1* | **range** *port1 port2*]
7. **icmp** *icmp-type*
8. **group-object** *nested-object-group-name*
9. Repeat the steps to specify the objects on which you want to base your object group.
10. **end**

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	**object-group service** *object-group-name* **Example:** `Device(config)# object-group service my-service-object-group`	Defines an object group name and enters service object-group configuration mode.
Step 4	**description** *description-text* **Example:** `Device(config-service-group)# description test engineers`	(Optional) Specifies a description of the object group. • You can use up to 200 characters.
Step 5	*protocol* **Example:** `Device(config-service-group)# ahp`	(Optional) Specifies an IP protocol number or name.

	Command or Action	Purpose
Step 6	{**tcp** | **udp** | **tcp-udp**} [**source** {{[**eq**] | **lt** | **gt**} *port1* | **range** *port1 port2*}] [{[**eq**] | **lt** | **gt**} *port1* | **range** *port1 port2*] **Example:** `Device(config-service-group)# tcp-udp range 2000 2005`	(Optional) Specifies TCP, UDP, or both.
Step 7	**icmp** *icmp-type* **Example:** `Device(config-service-group)# icmp conversion-error`	(Optional) Specifies the decimal number or name of an Internet Control Message Protocol (ICMP) type.
Step 8	**group-object** *nested-object-group-name* **Example:** `Device(config-service-group)# group-object my-nested-object-group`	(Optional) Specifies a nested (child) object group to be included in the current (parent) object group. • The type of child object group must match that of the parent (for example, if you are creating a network object group, you must specify another network object group as the child). • You can use duplicated objects in an object group only via nesting of group objects. For example, if object 1 is in both group A and group B, you can define a group C that includes both A and B. However, you cannot include a group object that causes the group hierarchy to become circular (for example, you cannot include group A in group B and then also include group B in group A). • You can use an unlimited number of levels of nested object groups (however, a maximum of two levels is recommended).
Step 9	Repeat the steps to specify the objects on which you want to base your object group.	—

Step 10	**end**
	Example:
	`Device(config-service-group)# end`
		Exits service object-group configuration mode and returns to privileged EXEC mode.

Creating an Object-Group-Based ACL

When creating an object-group-based access control list (ACL), configure an ACL that references one or more object groups. As with conventional ACLs, you can associate the same access policy with one or more interfaces.

You can define multiple access control entries (ACEs) that reference object groups within the same object-group-based ACL. You can also reuse a specific object group in multiple ACEs.

Perform this task to create an object-group-based ACL.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list extended** *access-list-name*
4. **remark** *remark*
5. **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**]
6. **remark** *remark*
7. **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**]
8. Repeat the steps to specify the fields and values on which you want to base your access list.
9. **end**

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** *access-list-name* **Example:** `Device(config)# ip access-list extended nomarketing`	Defines an extended IP access list using a name and enters extended access-list configuration mode.
Step 4	**remark** *remark* **Example:** `Device(config-ext-nacl)# remark protect server by denying access from the Marketing network`	(Optional) Adds a comment about the configured 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 Marketing network access to the interface.

	Command or Action	Purpose
Step 5	**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**] **Example:** `Device(config-ext-nacl)# deny ip 209.165.200.244 255.255.255.224 host 209.165.200.245 log`	(Optional) Denies any packet that matches all conditions specified in the statement. • Optionally use the **object-group** *service-object-group-name* keyword and argument as a substitute for the *protocol.* argument • Optionally use the **object-group** *source-network-object-group-name* keyword and argument as a substitute for the *source source-wildcard.* arguments • Optionally use the **object-group** *destination-network-object-group-name* keyword and argument as a substitute for the *destination destination-wildcard.* arguments • If the *source-wildcard* or *destination-wildcard*is omitted, a wildcard mask of 0.0.0.0 is assumed, which matches all bits of the source or destination address, respectively. • Optionally use the **any** keyword as a substitute for the *source source-wildcard* or *destination destination-wildcard* to specify the address and wildcard of 0.0.0.0 255.255.255.255. • Optionally use the **host** *source* keyword and argument to indicate a source and source wildcard of *source* 0.0.0.0 or the **host** *destination* keyword and argument to indicate a destination and destination wildcard of *destination* 0.0.0.0. • In this example, packets from all sources are denied access to the destination network 209.165.200.244. 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.
Step 6	**remark** *remark* **Example:** `Device(config-ext-nacl)# remark allow TCP from any source to any destination`	(Optional) Adds a comment about the configured access list entry. • A remark can precede or follow an access list entry.
Step 7	**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**]	Permits any packet that matches all conditions specified in the statement. • Every access list needs at least one permit statement. • Optionally use the **object-group** *service-object-group-name* keyword and argument as a substitute for the *protocol.* • Optionally use the **object-group** *source-network-object-group-name* keyword and argument as a substitute for the *source source-wildcard.*

	Command or Action	Purpose
	Example: `Device(config-ext-nacl)# permit tcp any any`	• Optionally use the **object-group** *destination-network-object-group-name* keyword and argument as a substitute for the *destination destination-wildcard.* • If *source-wildcard* or *destination-wildcard* is omitted, a wildcard mask of 0.0.0.0 is assumed, which matches on all bits of the source or destination address, respectively. • Optionally use the **any**keyword as a substitute for the *source source-wildcard* or *destination destination-wildcard* 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.
Step 8	Repeat the steps to specify the fields and values 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-ext-nacl)# end`
		Exits extended access-list configuration mode and returns to privileged EXEC mode.

Applying an Object Group-Based ACL to an Interface

Use the **ip access-group** command to apply an object group-based ACL to an interface. An object group-based access control list (ACL) can be used to control traffic on the interface it is applied to.

Perform this task to apply an object group-based ACL to an interface.

SUMMARY STEPS

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

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	**interface** *type number* **Example:** `Device(config)# interface vlan 100`	Specifies the interface and enters interface configuration mode.

Step 4	**ip access-group** {*access-list-name* | *access-list-number*} {**in** | **out**}
	Example:
	`Device(config-if)# ip access-group my-ogacl-policy in`
		Applies the ACL to the interface and specifies whether to filter inbound or outbound packets.

Step 5	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and returns to privileged EXEC mode.

Verifying Object Groups for ACLs

SUMMARY STEPS

1. **enable**
2. **show object-group** [*object-group-name*]
3. **show ip access-list** [*access-list-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. - Enter your password if prompted.
Step 2	**show object-group** [*object-group-name*] **Example:** `Device# show object-group my-object-group`	Displays the configuration in the named or numbered object group (or in all object groups if no name is entered).
Step 3	**show ip access-list** [*access-list-name*] **Example:** `Device# show ip access-list my-ogacl-policy`	Displays the contents of the named or numbered access list or object group-based ACL (or for all access lists and object group-based ACLs if no name is entered).

Configuration Examples for Object Groups for ACLs

Example: Creating a Network Object Group

The following example shows how to create a network object group named my-network-object-group, which contains two hosts and a subnet as objects:

```
Device> enable
Device# configure terminal
Device(config)# object-group network my-network-object-group
Device(config-network-group)# description test engineers
Device(config-network-group)# host 209.165.200.237
Device(config-network-group)# host 209.165.200.238

Device(config-network-group)# 209.165.200.241 255.255.255.224
Device(config-network-group)# end
```

The following example shows how to create a network object group named my-company-network, which contains two hosts, a subnet, and an existing object group (child) named my-nested-object-group as objects:

```
Device> enable
Device# configure terminal
Device(config)# object-group network my-company-network
Device(config-network-group)# host host1
Device(config-network-group)# host 209.165.200.242
Device(config-network-group)# 209.165.200.225 255.255.255.224
Device(config-network-group)# group-object my-nested-object-group
Device(config-network-group)# end
```

Example: Creating a Service Object Group

The following example shows how to create a service object group named my-service-object-group, which contains several ICMP, TCP, UDP, and TCP-UDP protocols and an existing object group named my-nested-object-group as objects:

```
Device> enable
Device# configure terminal
Device(config)# object-group service my-service-object-group
Device(config-service-group)# icmp echo
Device(config-service-group)# tcp smtp
Device(config-service-group)# tcp telnet
Device(config-service-group)# tcp source range 1 65535 telnet
Device(config-service-group)# tcp source 2000 ftp
Device(config-service-group)# udp domain
Device(config-service-group)# tcp-udp range 2000 2005
Device(config-service-group)# group-object my-nested-object-group
Device(config-service-group)# end
```

Example: Creating an Object Group-Based ACL

The following example shows how to create an object-group-based ACL that permits packets from the users in my-network-object-group if the protocol ports match the ports specified in my-service-object-group:

```
Device> enable
Device# configure terminal
Device(config)# ip access-list extended my-ogacl-policy
Device(config-ext-nacl)# permit object-group my-service-object-group object-group
my-network-object-group any
Device(config-ext-nacl)# deny tcp any any
Device(config-ext-nacl)# end
```

Example Applying an Object Group-Based ACL to an Interface

The following example shows how to apply an object group-based ACL to an interface. In this example, an object group-based ACL named my-ogacl-policy is applied to VLAN interface 100:

```
Device> enable
Device# configure terminal
Device(config)# interface vlan 100
Device(config-if)# ip access-group my-ogacl-policy in
Device(config-if)# end
```

Example: Verifying Object Groups for ACLs

The following example shows how to display all object groups:

```
Device# show object-group

Network object group auth-proxy-acl-deny-dest
 host 209.165.200.235
Service object group auth-proxy-acl-deny-services
 tcp eq www
 tcp eq 443
Network object group auth-proxy-acl-permit-dest
 209.165.200.226 255.255.255.224
 209.165.200.227 255.255.255.224
 209.165.200.228 255.255.255.224
 209.165.200.229 255.255.255.224
 209.165.200.246 255.255.255.224
 209.165.200.230 255.255.255.224
 209.165.200.231 255.255.255.224
 209.165.200.232 255.255.255.224
 209.165.200.233 255.255.255.224
 209.165.200.234 255.255.255.224
Service object group auth-proxy-acl-permit-services
 tcp eq www
 tcp eq 443
```

The following example shows how to display information about specific object-group-based ACLs:

```
Device# show ip access-list my-ogacl-policy

Extended IP access list my-ogacl-policy
10 permit object-group eng_service any any
```

Additional References for Object Groups for ACLs

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Security commands	• 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
ACL configuration guide	*Security Configuration Guide: Access Control Lists*

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

Feature Information for Object Groups for ACLs

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 10: Feature Information for Object Groups for ACLs

Feature Name	Releases	Feature Information
Object Groups for ACLs	12.4(20)T	The Object Groups for ACLs feature lets you classify users, devices, or protocols into groups and apply them to access control lists (ACLs) to create access control policies for those groups. This feature lets you use object groups instead of individual IP addresses, protocols, and ports, which are used in conventional ACLs. This feature allows multiple access control entries (ACEs), but now you can use each ACE to allow an entire group of users to access a group of servers or services or to deny them from doing so. The following commands were introduced or modified: **deny**, **ip access-group**, **ip access-list**, **object-group network**, **object-group service**, **permit**, **show ip access-list**, **show object-group**.

CHAPTER 8

IP Named Access Control Lists

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.

The IP Named Access Control Lists feature gives network administrators the option of using names to identify their access lists.

This module describes IP named access lists and how to configure them.

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 www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About IP Named Access Control Lists

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
```

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.

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.

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.

Note Outbound access list is not supported in Cisco ASR 900 RSP3 Module.

- 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.
- Because the software stops testing conditions after it encounters the first match (to either a **permit** or **deny** statement), you will reduce processing time and resources if you put the statements that packets are most likely to match at the beginning of the access list. Place more frequently occurring conditions before less frequent conditions.
- Organize your access list so that more specific references in a network or subnet appear before more general ones.

- 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.

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.

Note Outbound access list is not supported in Cisco ASR 900 RSP3 Module.

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.

How to Configure IP Named Access Control Lists

Creating an IP Named Access List

You can create an IP named access list to filter source addresses and destination addresses or a combination of addresses and other IP fields. Named access lists allow you to identify your access lists with an intuitive name.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list extended** *name*
4. **remark** *remark*
5. **deny** *protocol* [*source source-wildcard*] {**any** | **host** {*address* | *name*} {*destination* [*destination-wildcard*] {**any** | **host** {*address* | *name*} [**log**]
6. **remark** *remark*
7. **permit** *protocol* [*source source-wildcard*] {**any** | **host** {*address* | *name*} {*destination* [*destination-wildcard*] {**any** | **host** {*address* | *name*} [**log**]
8. Repeat Steps 4 through 7 to specify more statements for your access list.
9. **end**
10. **show ip access-lists**

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** *name* **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.

	Command or Action	Purpose
Step 4	**remark** *remark* **Example:** `Device(config-ext-nacl)# remark protect server by denying sales access to the acl1 network`	(Optional) Adds a description for an access list statement. • A remark can precede or follow an IP access list entry. • In this example, the **remark** command reminds the network administrator that the **deny** command configured in Step 5 denies the Sales network access to the interface.
Step 5	**deny** *protocol* [*source source-wildcard*] {**any** | **host** {*address* | *name*} {*destination* [*destination-wildcard*] {**any** | **host** {*address* | *name*} [**log**] **Example:** `Device(config-ext-nacl)# deny ip 192.0.2.0 0.0.255.255 host 192.0.2.10 log`	(Optional) Denies all packets that match all conditions specified by the remark.
Step 6	**remark** *remark* **Example:** `Device(config-ext-nacl)# remark allow TCP from any source to any destination`	(Optional) Adds a description for an access list statement. • A remark can precede or follow an IP access list entry.
Step 7	**permit** *protocol* [*source source-wildcard*] {**any** | **host** {*address* | *name*} {*destination* [*destination-wildcard*] {**any** | **host** {*address* | *name*} [**log**] **Example:** `Device(config-ext-nacl)# permit tcp any any`	Permits all packets that match all conditions specified by the statement.
Step 8	Repeat Steps 4 through 7 to specify more statements for your access list.	**Note** All source addresses that are not specifically permitted by a statement are denied by an implicit deny statement at the end of the access list.

Step 9	**end**
	Example:
	`Device(config-ext-nacl)# end`
		Exits extended named access list configuration mode and returns to privileged EXEC mode.

Step 10	**show ip access-lists** **Example:** `Device# show ip access-lists`	Displays the contents of all current IP access lists.

Example:

The following is sample output from the **show ip access-lists** command:

```
Device# show ip access-lists acl1
```

```
Extended IP access list acl1
   permit tcp any 192.0.2.0 255.255.255.255 eq telnet
   deny tcp any any
   deny udp any 192.0.2.0 255.255.255.255 lt 1024
   deny ip any any log
```

Applying an Access List to an 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. **end**

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	**interface** *type number* **Example:** `Device(config)# interface Gigabitethernet 1/0/2`	Specifies an interface and enters interface configuration mode.

Step 4	**ip access-group** {*access-list-number* | *access-list-name*} {**in** | **out**}
	Example:
	`Device(config-if)# ip access-group acl1 in`
		Applies the specified access list to the inbound interface.
		• To filter source addresses, apply the access list to the inbound interface.

Step 5	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and returns to privileged EXEC mode.

Configuration Examples for IP Named Access Control Lists

Example: Creating an IP Named Access Control List

```
Device# configure terminal
Device(config)# ip access-list extended acl1
Device(config-ext-nacl)# remark protect server by denying sales access to the acl1 network
Device(config-ext-nacl)# deny ip 192.0.2.0 0.0.255.255 host 192.0.2.10 log
Device(config-ext-nacl)# remark allow TCP from any source to any destination
Device(config-ext-nacl)# permit tcp any any
```

Example: Applying the Access List to an Interface

```
Device# configure terminal
Device(config)# interface Gigabitethernet 1/0/2
Device(config-if)# ip access-group acl1 in
```

Additional References for IP Named Access Control Lists

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Security commands	• 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

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

Feature Information for IP Named Access Control Lists

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 11: Feature Information for IP Named Access Control Lists

Feature Name	Releases	Feature Information
IP Named Access Control Lists	Cisco IOS XE Release 3.6E	Access control lists (ACLs) perform packet filtering to control the movement of packets through a network. Packet filtering provides security by limiting 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. In Cisco IOS XE Release 3.6E, this feature is supported on Cisco Catalyst 3850 Series Switches.

CHAPTER 9

IPv4 ACL Chaining Support

ACL Chaining, also known as Multi-Access Control List, allows you to split access control lists (ACLs). This module describes how with the IPv4 ACL Chaining Support feature, you can explicitly split ACLs into common and user-specific ACLs and bind both ACLs to a target for traffic filtering on a device. In this way, the common ACLs in Ternary Content Addressable Memory (TCAM) are shared by multiple targets, thereby reducing the resource usage.

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 www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for IPv4 ACL Chaining Support

- A single access control List (ACL) cannot be used for both common and regular ACLs for the same target in the same direction.
- ACL chaining applies to only security ACLs. It is not supported for feature policies, such as Quality of Service (QoS), Firewall Services Module (FW) and Policy Based Routing (PBR).

- Per-target statistics are not supported for common ACLs.

Information About IPv4 ACL Chaining Support

ACL Chaining Overview

The packet filter process supports only a single Access control list (ACL) to be applied per direction and per protocol on an interface. This leads to manageability and scalability issues if there are common ACL entries needed on many interfaces. Duplicate Access control entries (ACEs) are configured for all those interfaces, and any modification to the common ACEs needs to be performed for all ACLs.

A typical ACL on the edge box for an Internet Service Provider (ISP) has two sets of ACEs:

- Common ISP specific ACEs
- Customer/interface specific ACEs

The purpose of these address blocks is to deny access to ISP's protected infrastructure networks and anti-spoofing protection by allowing only customer source address blocks. This results in configuring unique ACL per interface and most of the ACEs being common across all ACLs on a device. ACL provisioning and modification is very cumbersome, hence, any changes to the ACE impacts every target.

IPv4 ACL Chaining Support

IPv4 ACL Chaining Support allows you to split the Access control list (ACL) into common and customer-specific ACLs and attach both ACLs to a common session. In this way, only one copy of the common ACL is attached to Ternary Content Addressable Memory (TCAM) and shared by all users, thereby making it easier to maintain the common ACEs.

The IPv4 ACL Chaining feature allows two IPV4 ACLs to be active on an interface per direction:

- Common
- Regular
- Common and Regular

Note If you configure both common and regular ACLs on an interface, the common ACL is considered over a regular ACL.

How to Configure IPv4 ACL Chaining Support

ACL chaining is supported by extending the **ip traffic filter** command.

The **ip traffic filter** command is not additive. When you use this command, it replaces earlier instances of the command.

For more information, refer to the *IPv6 ACL Chaining with a Common ACL* section in the Security Configuration Guide: Access Control Lists Configuration Guide.

Configuring an Interface to Accept Common ACL

Perform this task to configure the interface to accept a common Access control list (ACL) along with an interface-specific ACL:

SUMMARY STEPS

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

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	**interface** *type number*}
	Example:
	`Device(config)# interface gigabitethernet 0/0/0`
		Configures an interface (in this case a gigabitethernet interface) and enters the interface configuration mode.

Step 4	**ip access-group** {**common** {*common-access-list-name* {*regular-access-list* | **acl**}}{**in** | **out**}}
	Example:
	`Device(config)# ipv4 access-group common acl-p acl1 in`
		Configures the interface to accept a common ACL along with the interface-specific ACL.

Step 5	**end**
	Example:
	`Device(config-if)# end`
		(Optional) Exits the configuration mode and returns to privileged EXEC mode.

Configuration Examples for IPv4 ACL Chaining Support

This section provides configuration examples of Common Access Control List (ACL).

Example: Configuring an Interface to Accept a Common ACL

This example shows how to replace an Access Control List (ACL) configured on the interface without explicitly deleting the ACL:

```
interface gigabitethernet 0/0/0
ipv4 access-group common  C_acl ACL1 in
end
replace interface acl ACL1 by ACL2
interface gigabitethernet 0/0/0
ipv4 access-group common  C_acl ACL2 in
end
```

This example shows how common ACL cannot be replaced on interfaces without deleting it explicitly from the interface:

```
interface gigabitethernet 0/0/0
ipv4 access-group common  C_acl1 ACL1 in
end
change the common acl to C_acl2
interface gigabitethernet 0/0/0
no ipv4 access-group common  C_acl1  ACL1 in
end
interface gigabitethernet 0/0/0
ipv4 access-group common  C_acl2  ACL1 in
end
```

Note When reconfiguring a common ACL, you must ensure that no other interface on the line card is attached to the common ACL.

Note If both common ACL and interface ACL are attached to an interface and only one of the above is reconfigured on the interface, then the other is removed automatically.

This example shows how the interface ACL is removed:

```
interface gigabitethernet 0/0/0
ipv4 access-group common C_acl1 ACL1 in
end
```

Additional References for IPv4 ACL Chaining Support

Related Documents

Related Topic	Document Title
IPv6 ACL Chaining Support	Security Configuration Guide: Access Control Lists, Cisco IOS XE Release 3S

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Security commands	• 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

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. 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. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for IPv4 ACL Chaining Support

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 12: Feature Information for IPv4 ACL Chaining Support

Feature Name	Releases	Feature Information
IPv4 ACL Chaining Support	Cisco IOS XE Release 3.11S Cisco IOS XE Release 3.6E	The IPv4 ACL Chaining Support feature describes how you can explicitly split Access control lists (ACLs) into common and user-specific ACLs and bind both ACLs to a session for traffic filtering on a device. In this way, the common ACLs in Ternary Content Addressable Memory (TCAM) are shared by multiple targets, thereby reducing the resource usage. The following commands were introduced or modified: **ip access-group command**.

CHAPTER 10

IPv6 ACL Chaining with a Common ACL

ACL Chaining, also known as Multi-Access Control List (ACL), allows you to split ACLs. This document describes how with the IPv6 ACL Chaining Support feature, you can explicitly split ACLs into common and user-specific ACLs and bind both ACLs to a target for traffic filtering on a device. In this way, the common ACLs in Ternary Content Addressable Memory (TCAM) are shared by multiple targets, thereby reducing the resource usage.

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 www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About IPv6 ACL Chaining with a Common ACL

ACL Chaining Overview

The packet filter process supports only a single Access control list (ACL) to be applied per direction and per protocol on an interface. This leads to manageability and scalability issues if there are common ACL entries needed on many interfaces. Duplicate Access control entries (ACEs) are configured for all those interfaces, and any modification to the common ACEs needs to be performed for all ACLs.

A typical ACL on the edge box for an Internet Service Provider (ISP) has two sets of ACEs:

- Common ISP specific ACEs
- Customer/interface specific ACEs

The purpose of these address blocks is to deny access to ISP's protected infrastructure networks and anti-spoofing protection by allowing only customer source address blocks. This results in configuring unique ACL per interface and most of the ACEs being common across all ACLs on a device. ACL provisioning and modification is very cumbersome, hence, any changes to the ACE impacts every target.

IPv6 ACL Chaining with a Common ACL

With IPv6 ACL Chaining, you can configure a traffic filter with the following:

- Common ACL
- Specific ACL
- Common and Specific ACL

Each Access control list (ACL) is matched in a sequence. For example, if you have specified both the ACLs - a common and a specific ACL, the packet is first matched against the common ACL; if a match is not found, it is then matched against the specific ACL.

Note Any IPv6 ACL may be configured on a traffic filter as a common or specific ACL. However, the same ACL cannot be specified on the same traffic filter as both common and specific.

How to Configure IPv6 ACL Chaining with a Common ACL

Before You Begin

IPv6 ACL chaining is configured on an interface using an extension of the existing IPv6 traffic-filter command: **ipv6 traffic-filter** [**common** *common-acl*] [*specific-acl*] [**in** | **out**]

Note You may choose to configure either of the following:

- Only a common ACL. For example: **ipv6 traffic-filter common** *common-acl*
- Only a specific ACL. For example: **ipv6 traffic-filter** *common-acl*
- Both ACLs. For example: **ipv6 traffic-filter common** *common-acl specific-acl*

The ipv6 traffic-filter command is not additive. When you use the command, it replaces earlier instances of the command. For example, the command sequence: **ipv6 traffic-filter** [**common** *common-acl*] [*specific-acl*] **in** **ipv6 traffic-filter** [*specific-acl*] **in** binds a common ACL to the traffic filter, removes the common ACL and then binds a specific ACL.

Configuring the IPv6 ACL to an Interface

Perform this task to configure the interface to accept a common access control list (ACL) along with an interface-specific ACL:

SUMMARY STEPS

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

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	**interface** *type number*}
	Example:
	`Device(config)# interface gigabitethernet 0/0/0`
		Specifies the interface type and number, and enters interface configuration mode.

Step 4	**ipv6 traffic filter** {*common-access-list-name* {**in** | **out**}}
	Example:
	`Device(config)# ipv6 traffic-filter outbound out`
		Applies the specified IPv6 access list to the interface specified in the previous step.

Step 5	**end**
	Example:
	`Device(config-if)# end`
		(Optional) Exits the configuration mode and returns to privileged EXEC mode.

Configuration Examples for IPv6 ACL Chaining with a Common ACL

You may configure the following combinations in no particular order:

- A common ACL, for example: **ipv6 traffic-filter common** *common-acl* **in**
- A specific ACL, for example: **ipv6 traffic-filter** *specific-acl* **in**
- Both ACLs, for example: **ipv6 traffic-filter common** *common-acl specific-acl* **in**

Example: Configuring an Interface to Accept a Common ACL

This example shows how to replace an access control list (ACL) configured on the interface without explicitly deleting the ACL:

```
interface gigabitethernet 0/0/0
ipv6 access-group common  C_acl ACL1 in
end
replace interface acl ACL1 by ACL2
interface gigabitethernet 0/0/0
ipv6 access-group common  C_acl ACL2 in
end
```

This example shows how to delete a common ACL from an interface. A common ACL cannot be replaced on interfaces without deleting it explicitly from the interface.

```
interface gigabitethernet 0/0/0
ipv6 access-group common  C_acl1 ACL1 in
end
change the common acl to C_acl2
interface gigabitethernet 0/0/0
no ipv6 access-group common  C_acl1  ACL1 in
end
interface gigabitethernet 0/0/0
ipv6 access-group common  C_acl2  ACL1 in
end
```

Note When reconfiguring a common ACL, you must ensure that no other interface on the line card is attached to the common ACL.

Note If both common ACL and interface ACL are attached to an interface and only one of the above is reconfigured on the interface, then the other is removed automatically.

This example shows how to remove the interface ACL:

```
interface gigabitethernet 0/0/0
ipv6 access-group common C_acl1 ACL1 in
end
```

Additional References for IPv6 ACL Chaining with a Common ACL

Related Documents

Related Topic	Document Title
IPv4 ACL Chaining Support	Security Configuration Guide: Access Control Lists, Cisco IOS XE Release 3S
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Security commands	• 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

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. 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. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for IPv6 ACL Chaining with a Common ACL

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 13: Feature Information for IPv6 ACL Chaining with a Common ACL

Feature Name	Releases	Feature Information
IPv6 ACL Chaining with a Common ACL	Cisco IOS XE Release 3.11S Cisco IOS XE Release 3.6E	The ACL Chaining feature, also known as Multi-ACLs, allows you to explicitly split IPv6 traffic filter access control lists (ACLs) into common and per-session ACLs. In this way, the common access control entries (ACEs) that are used reduces resource usage of each ACL entry per session in the Ternary Content Addressable Memory (TCAM). The following commands were introduced or modified: **ip access-group common**.

CHAPTER 11

Standard IP Access List Logging

The Standard IP Access List Logging feature provides the ability to log messages about packets that are permitted or denied by a standard IP access list. Any packet that matches the access list logs an information message about the packet at the device console.

This module provides information about standard IP access list logging.

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 www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Standard IP Access List Logging

IP access list logging is supported only for routed interfaces or router access control lists (ACLs).

Information About Standard IP Access List Logging

Standard IP Access List Logging

The Standard IP Access List Logging feature provides the ability to log messages about packets that are permitted or denied by a standard IP access list. Any packet that matches the access list causes an information log message about the packet to be sent to the device console. The log level of messages that are printed to the device console is controlled by the **logging console** command.

The first packet that the access list inspects triggers the access list to log a message at the device console. Subsequent packets are collected over 5-minute intervals before they are displayed or logged. Log messages include information about the access list number, the source IP address of packets, the number of packets from the same source that were permitted or denied in the previous 5-minute interval, and whether a packet was permitted or denied. You can also monitor the number of packets that are permitted or denied by a particular access list, including the source address of each packet.

How to Configure Standard IP Access List Logging

Creating a Standard IP Access List Using Numbers

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **access-list** *access-list-number* {**deny** | **permit**} **host** *address* [**log**]
4. **access-list** *access-list-number* {**deny** | **permit**} **any** [**log**]
5. **interface** *type number*
6. **ip access-group** *access-list-number* {**in** | **out**}
7. **end**

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.

	Command or Action	Purpose
Step 3	**access-list** *access-list-number* {**deny** | **permit**} **host** *address* [**log**] **Example:** `Device(config)# access-list 1 permit host 10.1.1.1 log`	Defines a standard numbered IP access list using a source address and wildcard, and configures the logging of informational messages about packets that match the access list entry at the device console.
Step 4	**access-list** *access-list-number* {**deny** | **permit**} **any** [**log**] **Example:** `Device(config)# access-list 1 permit any log`	Defines a standard numbered IP access list by using an abbreviation for the source and source mask 0.0.0.0 255.255.255.255.
Step 5	**interface** *type number* **Example:** `Device(config)# interface Gigabitethernet 0/0`	Configures an interface and enters interface configuration mode.

Step 6	**ip access-group** *access-list-number* {**in** | **out**}
	Example:
	`Device(config-if)# ip access-group 1 in`
		Applies the specified numbered access list to the incoming or outgoing interface.
		• When you filter based on source addresses, you typically apply the access list to an incoming interface.

Step 7	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and enters privileged EXEC mode.

Creating a Standard IP Access List Using Names

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list standard** *name*
4. {**deny** | **permit**} {**host** *address* | **any**} **log**
5. **exit**
6. **interface** *type number*
7. **ip access-group** *access-list-name* {**in** | **out**}
8. **end**

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 standard** *name* **Example:** `Device(config)# ip access-list standard acl1`	Defines a standard IP access list and enters standard named access list configuration mode.
Step 4	{**deny** | **permit**} {**host** *address* | **any**} **log** **Example:** `Device(config-std-nacl)# permit host 10.1.1.1 log`	Sets conditions in a named IP access list that will deny packets from entering a network or permit packets to enter a network, and configures the logging of informational messages about packets that match the access list entry at the device console.
Step 5	**exit** **Example:** `Device(config-std-nacl)# exit`	Exits standard named access list configuration mode and enters global configuration mode.
Step 6	**interface** *type number* **Example:** `Device(config)# interface Gigabitethernet 0/0`	Configures an interface and enters interface configuration mode.

Step 7	**ip access-group** *access-list-name* {**in** | **out**}
	Example:
	`Device(config-if)# ip access-group acl1 in`
		Applies the specified access list to the incoming or outgoing interface.
		• When you filter based on source addresses, you typically apply the access list to an incoming interface.

Step 8	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and enters privileged EXEC mode.

Configuration Examples for Standard IP Access List Logging

Example: Creating a Standard IP Access List Using Numbers

```
Device# configure terminal
Device(config)# access-list 1 permit host 10.1.1.1 log
Device(config)# access-list 1 permit any log
Device(config)# interface Gigabitethernet 0/0
Device(config-if)# ip access-group 1 in
```

Example: Creating a Standard IP Access List Using Names

```
Device# configure terminal
Device(config)# ip access-list standard acl1
Device(config-std-nacl)# permit host 10.1.1.1 log
Device(config-std-nacl)# exit
Device(config)# interface Gigabitethernet 0/0
Device(config-if)# ip access-group acl1 in
```

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
```

Additional References for Standard IP Access List Logging

Related Documents

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

Related Topic	Document Title
Security commands	• 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

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

Feature Information for Standard IP Access List Logging

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 14: Feature Information for Standard IP Access List Logging

Feature Name	Releases	Feature Information
Standard IP Access List Logging	Cisco IOS XE Release 3.6E	The Standard IP Access List Logging feature provides the ability to log messages about packets that are permitted or denied by a standard IP access list. Any packet that matches the access list logs an information message about the packet at the device console. In Cisco IOS XE Release 3.6E, this feature is supported on Cisco Catalyst 3850 Series Switches.

CHAPTER 12

IPv6 Services—Standard Access Control Lists

Access lists determine the type of traffic that is blocked or forwarded at device interfaces. Access control lists (ACLs) allow the filtering of inbound and outbound traffic at interfaces based on source and destination addresses.

This module provides information about standard IPv6 ACLs.

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 www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About IPv6 Services--Standard 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 Services--Standard Access Control Lists

Configuring IPv6 Services—Standard Access Control Lists

IPv6 access control lists (ACLs) do not contain implicit permit rules.

The IPv6 neighbor discovery process uses the IPv6 network-layer service; therefore, you must configure IPv6 ACLs to allow IPv6 neighbor discovery packets to be sent and received on an interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 access-list** *access-list-name*
4. **permit** *source-ipv6-prefix/prefix-length* **host**
5. **deny** *protocol source-ipv6-prefix/prefix-length* **eq telnet any**
6. **exit**
7. **interface** *type number*
8. **ipv6 traffic-filter** *access-list-name* {**in** | **out**}
9. **end**
10. **show ipv6 access-list** [*access-list-name*]

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	**ipv6 access-list** *access-list-name* **Example:** `Device(config)# ipv6 access-list ipv6acl`	Defines an IPv6 ACL and enters IPv6 access list configuration mode. • IPv6 ACL names cannot contain a space or quotation mark or begin with a numeral.
Step 4	**permit** *source-ipv6-prefix/prefix-length* **host** **Example:** `Device(config-ipv6-acl)# permit 2001:DB8:1::1/32 any`	Specifies permit conditions for the IPv6 access list.
Step 5	**deny** *protocol source-ipv6-prefix/prefix-length* **eq telnet any** **Example:** `Device(config-ipv6-acl)# deny tcp 2001:DB8:0300:0201::/32 eq telnet any`	Specifies deny conditions for the IPv6 access list.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Device(config-ipv6-acl)# exit`	Exits IPv6 access list configuration mode and enters global configuration mode.
Step 7	**interface** *type number* **Example:** `Device(config)# interface Gigabitethernet 1/0/2`	Configures an interface and enters interface configuration mode.

Step 8	**ipv6 traffic-filter** *access-list-name* {**in** | **out**}
	Example:
	`Device(config-if)# ipv6 traffic-filter ipv6acl out`
		Applies the specified IPv6 access list to the interface configured in Step 7.

Step 9	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and enters privileged EXEC mode.

Step 10	**show ipv6 access-list** [*access-list-name*] **Example:** `Device# show ipv6 access-list`	Displays the contents of all current IPv6 access lists.

Example:

The following is sample output from the **show ipv6 access-list** command:

```
Device# show ipv6 access-list

IPv6 access list ipv6acl
    permit tcp any any eq bgp reflect tcptraffic (8 matches) sequence 10
    permit tcp any any eq telnet reflect tcptraffic (15 matches) sequence 20
    permit udp any any reflect udptraffic sequence 30

IPv6 access list tcptraffic (reflexive) (per-user)
    permit tcp host 2001:DB8:1::32 eq bgp host 2001:DB8:2::32 eq 11000 timeout 300 (time
left 243) sequence 1
    permit tcp host 2001:DB8:1::32 eq telnet host 2001:DB8:2::32 eq 11001 timeout 300 (time
 left 296) sequence 2

IPv6 access list outbound
    evaluate udptraffic
    evaluate tcptraffic
```

Configuration Examples for IPv6 Services--Standard Access Control Lists

Example: Configuring IPv6 Services—Standard Access Control Lists

```
Device# configure terminal
Device(config)# ipv6 access-list ipv6acl
Device(config-ipv6-acl)# permit 2001:DB8:1::1/32 any
Device(config-ipv6-acl)# deny tcp 2001:DB8:0300:0201::/32 eq telnet any
Device(config-ipv6-acl)# exit
Device(config)# interface Gigabitethernet 1/0/2
Device(config-if)# ipv6 traffic-filter ipv6acl out
Device(config-if)# end
```

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 OUTBOUND
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
```

Additional References for IPv6 Services—Standard Access Control Lists

Related Documents

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

Related Topic	Document Title
Security commands	• 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
IPv6 Commands	Cisco IOS IPv6 Command Reference

Standards and RFCs

Standard/RFC	Title
RFC 2474	Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers

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

Feature Information for IPv6 Services—Standard Access Control Lists

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to . An account on Cisco.com is not required.

Table 15: Feature Information for IPv6 Services—Standard Access Control Lists

Feature Name	Releases	Feature Information
IPv6 Services—Standard Access Control Lists	Cisco IOS XE 3.6E	Access lists determine the type of traffic that is blocked or forwarded at device interfaces. Access control lists (ACLs) allow the filtering of inbound and outbound traffic based on source and destination addresses at interfaces. Standard IPv6 ACLs support traffic filtering based on IPv6 option headers and optional, upper-layer protocol type information. In Cisco IOS XE Release 3.6E, this feature is supported on Cisco Catalyst 3850 Series Switches. The following commands were introduced or modified: **ipv6 access-list**, **show ipv6 access-list**.