Creating an IP Access List and Applying It to an Interface

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

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

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

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

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

Before you create or apply an IP access list, you should understand the concepts in the "IP Access List Overview" module. You should also have IP running in your network.

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

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

- Application control engine (ACE)-specific counters are not supported.
- Layer 3 IPv4 and IPv6 ACLs together are not supported on same EVC; only one of the ACL (either IPv4 and IPv6) is supported.
- MAC ACLs are not supported on Ethernet flow points (EFPs) or trunk EFP interfaces to which Layer 3 IPv4 or IPv6 ACLs are applied.
- A maximum of 500 ACEs per ACL are supported.
- IPv4 and IPv6 ACLs are not supported on the GigabitEthernet0 out-of-band management interface and EFP interfaces. IPv4 and IPv6 ACLs are supported on physical interfaces, bridge-domain interfaces, and port-channel interfaces.
- Object-groups are not supported on IP ACLs.
- **Permit ip any any** counters are not supported.
- The command **any-options** is not supoprted. For example: In the **60 deny ip any any-option** command any-option, is not supported.
- If the first statement is **deny permit**, you must add statement for the **permit** traffic.
- ICMP based filter, applied to ingress packets on BDI, filters injected, punted and incoming packets. For example, when a ping request is intiated from the device, an ACE, like **permit icmp any any**, filters the:
  - packets generated from the device
  - packets punted to the CPU
  - echo-replies that are incoming on this BDI.

  The ACL logs counter displays thrice the number of packets that were used to ping. To view packet-count, use the **show platform hardware pp active acl ipv4** *name ip access-list* **statistics** to view the packet count.
- The **show platform hardware pp active acl ipv4** *number 120 stats* fails to provide any output; but this command works fine with named ACL.
- IPv4 and IPv6 ACLs are not supported on TE Tunnel interfaces.

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

Helpful Hints for Creating IP Access Lists

The following tips will help you avoid unintended consequences and help you create more efficient 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.

Access List Remarks

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

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

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

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

Additional IP Access List Features

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

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

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

This section describes the general ways to create a standard or extended access list using either a name or a number. Access lists are very flexible; the tasks simply illustrate one **permit** command and one **deny** command to provide you the command syntax of each. Only you can determine how many **permit** and **deny** commands you need and their order.

Note The first two tasks in this module create an access list; you must apply the access list in order for it to function. If you want to apply the access list to an interface, perform the task "Applying the Access List to an Interface". If you don’t intend to apply the access list to an interface, see the "Where to Go Next" for pointers to modules that describe other ways to apply access lists.

Creating a Standard Access List to Filter on Source Address

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

Creating a Named Access List to Filter on Source Address

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

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

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

Example:

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

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

Step 4 **remark** *remark*

Example:

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

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

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

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

Example:

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

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

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

Step 6 **remark** *remark*

Example:

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

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

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

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

Example:

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

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

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

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

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

Step 9 **end**

Example:

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

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

Step 10 **show ip access-list**

Example:

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

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

Creating a Numbered Access List to Filter on Source Address

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

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

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **access-list** *access-list-number* **remark** *remark*

Example:

```
Device(config)# access-list 1 remark Give access to user1
```

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

- A remark of up to 100 characters can precede or follow an access list entry.

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

Example:

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

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

- Every access list needs at least one permit statement; it need not be the first entry.
- Standard IP access lists are numbered 1 to 99 or 1300 to 1999.
- If the source-wildcard is omitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source address.
- Optionally use the keyword any as a substitute for the source source-wildcard to specify the source and source wildcard of 0.0.0.0 255.255.255.255.
- In this example, host 172.16.5.22 is allowed to pass the access list.

Step 5 **access-list** *access-list-number* **remark** *remark*

Example:

```
Device(config)# access-list 1 remark Don’t give access to user2 and log any attempts
```

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

- A remark of up to 100 characters can precede or follow an access list entry.

Step 6 **access-list** *access-list-number* **deny** {*source* [*source-wildcard*] | **any**} [**log**]

Example:

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

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

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

Step 7 Repeat some combination of Steps 3 through 6 until you have specified the sources on which you want to base your access list.

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

Step 8 **end**

Example:

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

Exits global configuration mode and enters privileged EXEC mode.

Step 9 **show ip access-list**

Example:

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

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

Creating an Extended Access List

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

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

Creating a Named Extended Access List

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

Procedure

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

Enters global configuration mode.

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

Example:

```
Router(config)# ip access-list extended nomarketing
```

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

Step 4 **remark** *remark*

Example:

```
Router(config-ext-nacl)# remark protect server by denying access from the Marketing network
```

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

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

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:

```
Router(config-ext-nacl)# deny ip 172.18.0.0 0.0.255.255 host 172.16.40.10 log
```

(Optional) Denies any packet that matches all of the conditions specified in the statement.

- If the *source-wildcard* or *destination-wildcard*isomitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source or destination address, respectively.
- Optionally use the keyword **any** as a substitute for the *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 keyword **host** *source* to indicate a source and source wildcard of *source* 0.0.0.0 or the abbreviation **host** *destination*to indicate a destination and destination wildcard of *destination* 0.0.0.0.
- In this example, packets from the source network 172.18.0.0 are denied access to host 172.16.40.10. 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:

```
Router(config-ext-nacl)# remark allow TCP from any source to any destination
```

(Optional) Adds a user-friendly comment about an 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**]

Example:

```
Router(config-ext-nacl)# permit tcp any any
```

Permits any packet that matches all of the conditions specified in the statement.

- Every access list needs at least one permit statement.
- If the *source-wildcard* or *destination-wildcard*isomitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source or destination address, respectively.
- Optionally use the keyword **any** as a substitute for the *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.
- The **log-input** keyword can be configured, but it is not supported, and will not work as expected.

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

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:

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

Ends configuration mode and brings the system to privileged EXEC mode.

Step 10 **show ip access-list**

Example:

```
Router# show ip access-list
```

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

Creating a Numbered Extended Access List

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

Procedure

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

Enters global configuration mode.

Step 3 **access-list** *access-list-number* **remark** *remark*

Example:

```
Router(config)# access-list 107 remark allow Telnet packets from any source to network
172.69.0.0 (headquarters)
```

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

- A remark of up to 100 characters can precede or follow an access list entry.

Step 4 **access-list** *access-list-number* **permit** *protocol* {*source* [*source-wildcard*] | **any**} {*destination* [*destination-wildcard*] | **any**} [**precedence** *precedence*] [**tos** *tos*] [**established**] [**log** | **log-input**] [**time-range** *time-range-name*] [**fragments**]

Example:

```
Router(config)# access-list 107 permit tcp any 172.69.0.0 0.0.255.255 eq telnet
```

Permits any packet that matches all of the conditions specified in the statement.

- Every access list needs at least one **permit** statement; it need not be the first entry.
- Extended IP access lists are numbered 100 to 199 or 2000 to 2699.
- If the *source-wildcard* or *destination-wildcard*isomitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source or destination address, respectively.
- Optionally use the keyword **any** as a substitute for the *source source-wildcard*or *destination destination-wildcard*to specify the address and wildcard of 0.0.0.0 255.255.255.255.
- TCP and other protocols have additional syntax available. See the **access-list** command in the command reference for complete syntax.

Step 5 **access-list** *access-list-number* **remark** *remark*

Example:

```
Router(config)# access-list 107 remark deny all other TCP packets
```

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

- A remark of up to 100 characters can precede or follow an access list entry.

Step 6 **access-list** *access-list-number* **deny** *protocol* {*source* [*source-wildcard*] | **any**} {*destination* [*destination-wildcard*] | **any**} [**precedence** *precedence*] [**tos** *tos*] [**established**] [**log** | **log-input**] [**time-range** *time-range-name*] [**fragments**]

Example:

```
Router(config)# access-list 107 deny tcp any any
```

Denies any packet that matches all of the conditions specified in the statement.

- If the *source-wildcard* or *destination-wildcard*isomitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source or destination address, respectively.
- Optionally use the keyword **any** as a substitute for the *source source-wildcard*or *destination destination-wildcard*to specify the address and wildcard of 0.0.0.0 255.255.255.255.

Step 7 Repeat some combination of Steps 3 through 6 until you have specified the fields and values on which you want to base your access list.

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

Step 8 **end**

Example:

```
Router(config)# end
```

Ends configuration mode and brings the system to privileged EXEC mode.

Step 9 **show ip access-list**

Example:

```
Router# show ip access-list
```

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

Applying the Access List to an Interface

Perform this task to apply an access list to an interface.

Procedure

Step 1 **enable**

Example:

```
Router> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Router# configure terminal
```

Enters global configuration mode.

Step 3 **interface** *type slot /subslot /port*

Example:

```
Router(config)# interface GigabitEthernet 0/0/1
```

Specifies an interface and enters interface configuration mode.

Step 4 **ip access-group** {*access-list-number* | *access-list-name*} {**in** | **out**}

Example:

```
Router(config-if)# ip access-group noncorp in
```

Applies the specified access list to the incoming or outgoing interface.

- When you are filtering on source addresses, you typically apply the access list to an incoming interface.
- Filtering on source addresses is most efficient when applied near the destination.

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

Example: Filtering on Source Address (Hosts)

In the following example, the workstation belonging to Jones is allowed access to Gigabitethernet interface 0/0/2 and the workstation belonging to Smith is not allowed access:

```
interface Gigabitethernet 0/0/2

 ip access-group workstations in
!
ip access-list standard workstations
 remark Permit only Jones workstation through
 permit 172.16.2.88
 remark Do not allow Smith workstation through
 deny 172.16.3.13
```

Example: Filtering on Source Address (Subnet)

In the following example, the Jones subnet is not allowed access to Gigabitethernet interface 0/0/2, but the Main subnet is allowed access:

```
interface Gigabitethernet 0/0/0

 ip access-group prevention in
!
ip access-list standard prevention
 remark Do not allow Jones subnet through
 deny 172.22.0.0 0.0.255.255
 remark Allow Main subnet
 permit 172.25.0.0 0.0.255.255
```

Example: Filtering on Source Address Destination Address and IP Protocols

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

The extended access list named marketing_group filters incoming packets. The access list permits Telnet packets from any source to network 172.26.0.0 and denies all other TCP packets. It permits any ICMP packets. It denies UDP packets from any source to network 172.26.0 0 on port numbers less than 1024. Finally, the access list denies all other IP packets and performs logging of packets passed or denied by that entry.

```
interface Gigabitethernet 0/0/5

 ip address 172.20.5.1 255.255.255.0
 ip access-group Internet_filter out
 ip access-group marketing_group in
!
ip access-list standard Internet_filter
 permit 172.16.3.4
ip access-list extended marketing_group
 permit tcp any 172.26.0.0 0.0.255.255 eq telnet
 deny tcp any any
 permit icmp any any
 deny udp any 172.26.0.0 0.0.255.255 lt 1024
 deny ip any any
```

Example: Filtering on Source Address (Host and Subnets) Using a Numbered Access List

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

```
interface Gigabitethernet 0/0/3

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

Example: Preventing Telnet Access to a Subnet

In the following example, the Jones subnet is not allowed to Telnet out Gigabitethernet interface 0/0/2:

```
interface Gigabitethernet 0/0/2

 ip access-group telnetting out
!
ip access-list extended telnetting
 remark Do not allow Jones subnet to telnet out
 deny tcp 172.20.0.0 0.0.255.255 any eq telnet
```

```
 remark Allow Top subnet to telnet out
 permit tcp 172.33.0.0 0.0.255.255 any eq telnet
```

Example: Filtering on TCP and ICMP Using Port Numbers

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

```
interface Gigabitethernet 0/0/2

 ip access-group goodports in
!
ip access-list extended goodports
 permit tcp any 172.28.0.0 0.0.255.255 gt 1023
 permit tcp any host 172.28.1.2 eq 25
 permit icmp any 172.28.0.0 255.255.255.255
```

Example: Allowing SMTP (E-mail) and Established TCP Connections

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

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

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

```
interface Gigabitethernet 0/0/2

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

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

In the following example, the Winter and Smith workstations are not allowed web access; other hosts on network 172.20.0.0 are allowed web access:

```
interface Gigabitethernet 0/0/2

 ip access-group no_web out
!
ip access-list extended no_web
```

```
 remark Do not allow Winter to browse the web
 deny host 172.20.3.85 any eq http
 remark Do not allow Smith to browse the web
 deny host 172.20.3.13 any eq http
 remark Allow others on our network to browse the web
 permit 172.20.0.0 0.0.255.255 any eq http
```

Example: Filtering on Source Address and Logging the Packets Permitted and Denied

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

```
interface Gigabitethernet 0/0/0
 ip address 172.16.1.1 255.0.0.0
 ip access-group 1 in
 ip access-group 2 out
!
access-list 1 permit 172.25.0.0 0.0.255.255 log
access-list 1 deny 172.30.0.0 0.0.255.255 log
!
access-list 2 permit 172.27.3.4 log
access-list 2 deny 172.17.0.0 0.0.255.255 log
```

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

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

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

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

Example: Limiting Debug Output

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

```
Device(config)# ip access-list standard acl1

!Displays only advertisements for LDP peer in acl1
Device(config-std-nacl)# permit host 10.0.0.44

Device# debug mpls ldp advertisements peer-acl acl1

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

Apply Access Control List on Management Interface

Table 1: Feature History

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

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

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

Restrictions

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

Configure ACL on Management Interface

Create an ACL

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

Apply ACL on Management Interface

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

Verification of ACL Configuration on Management Interface

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

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

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

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/mcl/allreleasemcl/all-book.html

Standards and RFCs

Standard/RFC	Title
No specific Standards and RFCs are supported by the features in this document.	—

MIBs

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

Technical Assistance

Description	Link
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/cisco/web/support/index.html

Feature Information for Creating an IP Access List and Applying It to an Interface

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

Table 2: Feature Information for Creating an IP Access List and Applying It to an Interface

Feature Name	Releases	Feature Configuration Information
Creating an IP Access List and Applying It to an Interface	Cisco IOS XE Release 3.14.0S	This feature was introduced on the Cisco ASR 920 Series Aggregation Services Router (ASR-920-12CZ-A, ASR-920-12CZ-D, ASR-920-4SZ-A, ASR-920-4SZ-D, ASR-920-10SZ-PD, ASR-920-24SZ-IM, ASR-920-24SZ-M, ASR-920-24TZ-M) .