ACL Support for Filtering on TTL Value

Customers may use extended IP access lists (named or numbered) to filter packets based on their time-to-live (TTL) value, from 0 to 255. This filtering enhances a customer’s control over which packets reach a router.

History for the ACL Support for Filtering on TTL Value Feature

Release	Modification
12.4(2)T	This feature was introduced.

Finding Support Information for Platforms and Cisco IOS Software Images

Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support. Access Cisco Feature Navigator at http://www.cisco.com/go/fn. You must have an account on Cisco.com. If you do not have an account or have forgotten your username or password, click **Cancel** at the login dialog box and follow the instructions that appear.

Contents

Restrictions for ACL Support for Filtering on TTL Value

- This feature does not support turbo access lists.

Cisco Systems

Corporate Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

- When the access list specifies the operation EQ or NEQ, routers running Cisco IOS Release 12.2S can have that access list specify up to ten TTL values. However, for Release 12.0S, only one TTL value can be specified.

Information About ACL Support for Filtering on TTL Value

Before you configure an access list that filters on TTL, you should understand the following concepts:

- How Filtering on TTL Works, page 2
- Benefits of Filtering on TTL, page 3

How Filtering on TTL Works

IP extended named and numbered access lists may filter on the TTL value of packets arriving at or leaving an interface. Packets with any possible TTL values 0 through 255 may be permitted or denied (filtered). Like filtering on other fields, such as source or destination address, the **ip access-group** command specifies **in** or **out**, which makes the access list ingress or egress and applies it to incoming or outgoing packets, respectively. The TTL value is checked in conjunction with the specified protocol, application, and any other settings in the access list entry, and all conditions must be met.

Special Handling for Packets with TTL or 0 or 1 Arriving on Ingress Interface

The software switching paths [distributed Cisco Express Forwarding (dCEF), CEF, fast switching, and process switching] will usually permit or discard the packets based on the access list statements. However, when the TTL value of packets arriving on an *ingress* interface have a TTL of 0 or 1, special handling is required. The packets with a TTL of 0 or 1 get sent to the process level before the ingress access list is checked in CEF, dCEF, or fast switching paths. The ingress access list is applied to packets with TTL values 2 through 255 and a permit or deny decision is made.

Packets with a TTL value of 0 or 1 are sent to the process level because they will never be forwarded out of the device; the process level must check whether each packet is destined for the router or not and whether an Internet Control Message Protocol (ICMP) TTL Expire message needs to be sent back or not. This means that even if an ACL with TTL value 0 or 1 filtering is configured on the ingress interface with the intention to drop packets with a TTL of 0 or 1, the dropping of the packets will not happen in the faster paths. It will instead happen in the process level when the process applies the ACL. This is also true for hardware switching platforms. Packets with TTL 0 or 1 are sent to the process level of the route processor (RP) or Multilayer Switch Feature Card (MSFC).

On egress interfaces, access list filtering on TTL work just like other access list features. The check will happen in the fastest switching path enabled in the device. This is because the faster switching paths handle all the TTL values (0-255) equally on the egress interface.

Control Plane Policing for Filtering TTL Values 0 and 1

The special behavior for packets with a TTL of 0 or 1 results in higher CPU usage for the device. If you are filtering on TTL value 0 or 1, you should use control plane policing (CPP) to protect the CPU from being overwhelmed. In order to leverage CPP, you must configure an access list especially for filtering TTL values 0 and 1 and apply the access list through CPP. This access list will be a separate access list from any interface access lists. Because CPP works for the entire system, not just on individual interfaces, you would need to configure only one such special access list for the entire device. This task is described in the section "Enabling Control Plane Policing to Filter on TTL Values 0 and 1" section on page 4.

Benefits of Filtering on TTL

- Filtering on TTL provides a way to control which packets are allowed to reach the router or prevented from reaching the router. By looking at your network layout, you can choose whether to accept or deny packets from a certain router based on how many hops away it is. For example, in a small network, you can deny packets from a location more than three hops away. Filtering on TTL allows you to validate if the traffic originated from a neighboring device, as follows. You can accept only packets that reach you in one hop, for example, by accepting only packets with a TTL of one less than the initial TTL value of a particular protocol.
- Many control plane protocols communicate only with their neighbors, but receive packets from everyone. By applying to receiving routers an access list that filters on TTL, you can block unwanted packets.
- The Cisco IOS software sends all packets with a TTL of 0 or 1 to the process level to be processed. The device must then send an ICMP TTL expire message to the source. By filtering packets that have a TTL of 0 through 2, you can reduce the load on the process level.

How to Filter Packets Based on TTL Value

Because access lists are very flexible, it is not possible to define only one combination of **permit** and **deny** commands to filter packets based on the TTL value. These tasks illustrate just one example that achieves TTL filtering. Configure the appropriate **permit** and **deny** statements that will accomplish your filtering plan.

- Filtering Packets Based on TTL Value, page 3
- Enabling Control Plane Policing to Filter on TTL Values 0 and 1, page 4

Filtering Packets Based on TTL Value

Perform steps similar to the steps in this task to filter packets based on their TTL value. Configure the appropriate **permit** and **deny** statements that will accomplish your filtering plan.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list extended** *access-list-name*
4. [*sequence-number*] **permit** *protocol source source-wildcard destination destination-wildcard* [**option** *option-name*] [**precedence** *precedence*] [**tos** *tos*] [**ttl** *operator value*] [**log**] [**time-range** *time-range-name*] [**fragments**]
5. Continue to add **permit** or **deny** statements to achieve the filtering you want.
6. **exit**
7. **interface** *type number*
8. **ip access-group** *access-list-name* {**in** | **out**}

DETAILED STEPS

	Command or Action	Purpose
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** *access-list-name* **Example:** `Router(config)# ip access-list extended ttlfilter`	Defines an IP access list by name. - An access list that filters on TTL value must be an extended access list.
Step 4	[*sequence-number*] **permit** *protocol source source-wildcard destination destination-wildcard* [**option** *option-name*] [**precedence** *precedence*] [**tos** *tos*] [**ttl** *operator value*] [**log**] [**time-range** *time-range-name*] [**fragments**] **Example:** `Router(config-ext-nacl)# permit ip host 172.16.1.1 any ttl lt 2`	Sets conditions to allow a packet to pass a named IP access list. - Every access list must have at least one **permit** statement. - This example permits packets from source 172.16.1.1 to any destination with a TTL value less than 2.
Step 5	Continue to add **permit** or **deny** statements to achieve the filtering you want.	—
Step 6	**exit** **Example:** `Router(config-ext-nacl)# exit`	Exits any configuration mode to the next highest mode in the CLI mode hierarchy.
Step 7	**interface** *type number* **Example:** `Router(config)# interface ethernet 0`	Configures an interface type and enters interface configuration mode.

Step 8	**ip access-group** *access-list-name* {**in** | **out**}
	Example:
	`Router(config-if)# ip access-group ttlfilter in`
		Applies the access list to an interface.

Enabling Control Plane Policing to Filter on TTL Values 0 and 1

Perform this task if you want to filter IP packets based on a TTL value of 0 or 1 and you want to protect the CPU from being overwhelmed. This task configures an access list for classification on TTL 0 and 1, configures Modular QoS CLI (MQC), and applies the policy map to the control plane. Any packets that pass the access list are dropped. This special access list is separate from any interface access lists.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list extended** *access-list-name*
4. [*sequence-number*] **permit** *protocol source source-wildcard destination destination-wildcard* **ttl** *operator value*
5. Continue to add **permit** or **deny** statements to achieve the filtering you want.
6. **exit**
7. **class-map** *class-map-name* [**match-all** | **match-any**]
8. **match access-group** {*access-group* | **name** *access-group-name*}
9. **exit**
10. **policy-map** *policy-map-name*
11. **class** {*class-name* | **class-default**}
12. **drop**
13. **exit**
14. **exit**
15. **control-plane**
16. **service-policy** {**input** | **output**} *policy-map-name*

DETAILED STEPS

	Command or Action	Purpose

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`** *`access-list-name`*
	Example:
	`Router(config)# ip access-list extended ttlfilter`
		Defines an IP access list by name.
		• An access list that filters on a TTL value must be an extended access list.

Step 4	[*`sequence-number`*] **`permit`** *`protocol source source-wildcard destination destination-wildcard`* **`ttl`** *`operator`* `value` **Example:** `Router(config-ext-nacl)# permit ip host 172.16.1.1 any ttl lt 2`	Sets conditions to allow a packet to pass a named IP access list. • Every access list must have at least one **permit** statement. • This example permits packets from source 172.16.1.1 to any destination with a TTL value less than 2.

	Command or Action	Purpose
Step 5	Continue to add **permit** or **deny** statements to achieve the filtering you want.	The packets that pass the access list will be dropped.

Step 6	`exit`
	Example:
	`Router(config-ext-nacl)# exit`
		Exits any configuration mode to the next highest mode in the CLI mode hierarchy.

Step 7	**`class-map`** *`class-map-name`* [**`match-all`** | **`match-any`**] **Example:** `Router(config)# class-map acl-filtering`	Creates a class map to be used for matching packets to a specified class.

Step 8	**`match access-group`** {*`access-group`* | **`name`** *`access-group-name`*}
	Example:
	`Router(config-cmap)# match access-group name ttlfilter`
		Configures the match criteria for a class map on the basis of the specified access control list.

Step 9	`exit`
	Example:
	`Router(config-cmap)# exit`
		Exits any configuration mode to the next highest mode in the CLI mode hierarchy.

Step 10	**`policy-map`** *`policy-map-name`*
	Example:
	`Router(config)# policy-map acl-filter`
		Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.

Step 11	**`class`** {*`class-name`* | **`class-default`**}
	Example:
	`Router(config-pmap)# class acl-filter-class`
		Specifies the name of the class whose policy you want to create or change or to specify the default class (commonly known as the class-default class) before you configure its policy.

Step 12	`drop`
	Example:
	`Router(config-pmap-c)# drop`
		Configures a traffic class to discard packets belonging to a specific class.

Step 13	`exit`
	Example:
	`Router(config-pmap-c)# exit`
		Exits any configuration mode to the next highest mode in the CLI mode hierarchy.

Step 14	`exit`
	Example:
	`Router(config-pmap)# exit`
		Exits any configuration mode to the next highest mode in the CLI mode hierarchy.

	Command or Action	Purpose
Step 15	**control-plane** **Example:** `Router(config)# control-plane`	Associates or modifies attributes or parameters that are associated with the control plane of the device.
Step 16	**service-policy** {**input** | **output**} *policy-map-name* **Example:** `Router(config-cp)# service-policy input acl-filter`	Attaches a policy map to a control plane for aggregate control plane services.

Configuration Examples for Filtering on TTL Value

This section contains the following configuration examples:

- Filtering on TTL Value: Example, page 7
- Control Plane Policing to Filter on TTL Values 0 and 1: Example, page 7

Filtering on TTL Value: Example

The following access list filters IP packets containing type of service (ToS) level 3 with TTL values 10 and 20. It also filters IP packets with a TTL greater than 154 and applies that rule to noninitial fragments. It permits IP packets with a precedence level of flash and a TTL not equal to 1, and it sends log messages about such packets to the console. All other packets are denied.

```
ip access-list extended incomingfilter
 deny ip any any tos 3 ttl eq 10 20
 deny ip any any ttl gt 154 fragments
 permit ip any any precedence flash ttl neq 1 log
!
interface ethernet 0
ip access-group incomingfilter in
```

Control Plane Policing to Filter on TTL Values 0 and 1: Example

The following example configures a traffic class called acl-filter-class for use in a policy map called acl-filter. An access list permits IP packets from any source having a TTL of 0 or 1. Any packets matching the access list are dropped. The policy map is attached to the control plane.

```
ip access-list extended ttlfilter
 permit ip any any ttl eq 0 1
class-map acl-filter-class
 match access-group name ttlfilter
policy-map acl-filter
 class acl-filter-class
 drop
control-plane
 service-policy input acl-filter
```

Additional References

The following sections provide references related to ACL Support for Filtering on TTL Value.

Related Documents

Related Topic	Document Title
Configuring IP access lists	*Cisco IOS IP Application Services Configuration Guide*, Release 12.4
QoS commands	*Cisco IOS Quality of Service Solutions Command Reference*, Release 12.4

Standards

Standard	Title
None	—

MIBs

MIB	MIBs Link
None	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

RFCs

RFC	Title
None	—

Technical Assistance

Description	Link
Technical Assistance Center (TAC) home page, containing 30,000 pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/public/support/tac/home.shtml

Command Reference

This section documents modified commands.

- **deny (IP)**
- **permit (IP)**

deny (IP)

To set conditions in a named IP access list that will deny packets, use the **deny** command in access list configuration mode. To remove a deny condition from an access list, use the **no** form of this command.

[*sequence-number*] **deny** *source* [*source-wildcard*]

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

no *sequence-number*

no deny *source* [*source-wildcard*]

no deny *protocol source source-wildcard destination destination-wildcard*

Internet Control Message Protocol (ICMP)

[*sequence-number*] **deny icmp** *source source-wildcard destination destination-wildcard* [*icmp-type* [*icmp-code*] | *icmp-message*] [**precedence** *precedence*] [**tos** *tos*] [**ttl** *operator value*] [**log**] [**time-range** *time-range-name*] [**fragments**]

Internet Group Management Protocol (IGMP)

[*sequence-number*] **deny igmp** *source source-wildcard destination destination-wildcard* [*igmp-type*] [**precedence** *precedence*] [**tos** *tos*] [**ttl** *operator value*] [**log**] [**time-range** *time-range-name*] [**fragments**]

Transmission Control Protocol (TCP)

[*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*] [**ttl** *operator value*] [**log**] [**time-range** *time-range-name*] [**fragments**]

User Datagram Protocol (UDP)

[*sequence-number*] **deny udp** *source source-wildcard* [*operator port* [*port*]] *destination destination-wildcard* [*operator* [*port*]] [**precedence** *precedence*] [**tos** *tos*] [**ttl** *operator value*] [**log**] [**time-range** *time-range-name*] [**fragments**]

Syntax Description

sequence-number	(Optional) Sequence number assigned to the deny statement. The sequence number causes the system to insert the statement in that numbered position in the access list.
source	Number of the network or host from which the packet is being sent. There are three alternative ways to specify the source: • Use a 32-bit quantity in four-part dotted-decimal format. • Use the **any** keyword as an abbreviation for a *source* and *source-wildcard* of 0.0.0.0 255.255.255.255. • Use **host** *source* as an abbreviation for a *source* and *source-wildcard* of *source* 0.0.0.0.
source-wildcard	Wildcard bits to be applied to the source. There are three alternative ways to specify the source wildcard: • Use a 32-bit quantity in four-part dotted-decimal format. Place 1s in the bit positions that you want to ignore. • Use the **any** keyword as an abbreviation for a *source* and *source-wildcard* of 0.0.0.0 255.255.255.255. • Use **host** *source* as an abbreviation for a *source* and *source-wildcard* of *source* 0.0.0.0.
protocol	Name or number of an Internet protocol. The *protocol* argument can be one of the keywords **eigrp**, **gre**, **icmp**, **igmp**, **ip**, **ipinip**, **nos**, **ospf**, **tcp**, or **udp**, or an integer in the range from 0 to 255 representing an Internet protocol number. To match any Internet protocol (including ICMP, TCP, and UDP), use the **ip** keyword. **Note** When the **icmp**, **igmp**, **tcp,** and **udp** keywords are entered, they must be followed with the specific command syntax that is shown for the ICMP, IGMP, TCP, and UDP forms of the **deny** command.
icmp	Denies only ICMP packets. When you enter the **icmp** keyword, you must use the specific command syntax shown for the ICMP form of the **deny** command.
igmp	Denies only IGMP packets. When you enter the **igmp** keyword, you must use the specific command syntax shown for the IGMP form of the **deny** command.
tcp	Denies only TCP packets. When you enter the **tcp** keyword, you must use the specific command syntax shown for the TCP form of the **deny** command.
udp	Denies only UDP packets. When you enter the **udp** keyword, you must use the specific command syntax shown for the UDP form of the **deny** command.
destination	Number of the network or host to which the packet is being sent. There are three alternative ways to specify the destination: • Use a 32-bit quantity in four-part dotted-decimal format. • Use the **any** keyword as an abbreviation for the *destination* and *destination-wildcard* of 0.0.0.0 255.255.255.255. • Use **host** *destination* as an abbreviation for a *destination* and *destination-wildcard* of *destination* 0.0.0.0.

destination-wildcard	Wildcard bits to be applied to the destination. There are three alternative ways to specify the destination wildcard: • Use a 32-bit quantity in four-part dotted-decimal format. Place 1s in the bit positions that you want to ignore. • Use the **any** keyword as an abbreviation for a *destination* and *destination-wildcard* of 0.0.0.0 255.255.255.255. • Use **host** *destination* as an abbreviation for a *destination* and *destination-wildcard* of *destination* 0.0.0.0.
option *option-name*	(Optional) Packets can be filtered by IP Options, as specified by a number from 0 to 255 or by the corresponding IP Option name, as listed in Table 1 in the “Usage Guidelines” section.
precedence *precedence*	(Optional) Packets can be filtered by precedence level, as specified by a number from 0 to 7 or by a name.
tos *tos*	(Optional) Packets can be filtered by type of service (ToS) level, as specified by a number from 0 to 15, or by a name as listed in the “Usage Guidelines” section of the **access-list** (IP extended) command.
ttl *operator value*	(Optional) Compares the TTL value in the packet to the TTL value specified in this **deny** statement. • The *operator* can be **lt** (less than), **gt** (greater than), **eq** (equal), **neq** (not equal), or **range** (inclusive range). • The *value* can range from 0 to 255. • If the operator is **range**, specify two values separated by a space. • For Release 12.0S, if the operator is **eq** or **neq**, only one TTL value can be specified. • For all other releases, if the operator is **eq** or **neq**, as many as 10 TTL values can be specified, separated by a space. If the TTL in the packet matches just one of the possibly 10 values, the entry is considered to be matched.
log	(Optional) Causes an informational logging message about the packet that matches the entry to be sent to the console. (The level of messages logged to the console is controlled by the **logging console** command.)
time-range *time-range-name*	(Optional) Name of the time range that applies to this **deny** statement. The name of the time range and its restrictions are specified by the **time-range** and **absolute** or **periodic** commands, respectively.
fragments	(Optional) The access list entry applies to noninitial fragments of packets; the fragment is either permitted or denied accordingly. For more details about the **fragments** keyword, see the “Access List Processing of Fragments” and “Fragments and Policy Routing” sections in the “Usage Guidelines” section.
icmp-type	(Optional) ICMP packets can be filtered by ICMP message type. The type is a number from 0 to 255.
icmp-code	(Optional) ICMP packets that are filtered by ICMP message type can also be filtered by the ICMP message code. The code is a number from 0 to 255.

icmp-message	(Optional) ICMP packets can be filtered by an ICMP message type name or an ICMP message type and code name. The possible names are listed in the "Usage Guidelines" section of the **access-list** (IP extended) command.
igmp-type	(Optional) IGMP packets can be filtered by IGMP message type or message name. A message type is a number from 0 to 15. IGMP message names are listed in the "Usage Guidelines" section of the **access-list** (IP extended) command.
operator	(Optional) Compares source or destination ports. 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. Up to ten port numbers can be entered for the **eq** (equal) and **neq** (not equal) operators. All other operators require one port number.
port	(Optional) The decimal number or name of a TCP or UDP port. A port number is a number from 0 to 65535. TCP and UDP port names are listed in the "Usage Guidelines" section of the **access-list** (IP extended) command. TCP port names can be used only when filtering TCP. UDP port names can be used only when filtering UDP.
established	(Optional) For the TCP protocol only: Indicates an established connection. A match occurs if the TCP datagram has the ACK or RST bit set. The nonmatching case is that of the initial TCP datagram to form a connection. **Note** The **established** keyword can be used only with the old command-line interface (CLI) format. To use the new CLI format, you must use the **match-any** or **match-all** keywords followed by the **+** or **-** keywords and *flag-name* argument.
{**match-any** | **match-all**}	(Optional) For the TCP protocol only: A match occurs if the TCP datagram has certain TCP flags set or not set. You use the **match-any** keyword to allow a match to occur if any of the specified TCP flags are present, or you can use the **match-all** keyword to allow a match to occur only if all of the specified TCP flags are present. You must follow the **match-any** and **match-all** keywords with the **+** or **-** keyword and the *flag-name* argument to match on one or more TCP flags.
{**+** | **-**} *flag-name*	(Optional) For the TCP protocol only: The **+** keyword allows IP packets if their TCP headers contain the TCP flags that are specified by the *flag-name* argument. The **-** keyword filters out IP packets that do not contain the TCP flags specified by the *flag-name* argument. You must follow the **+** and **-** keywords with the *flag-name* argument. TCP flag names can be used only when filtering TCP. Flag names for the TCP flags are as follows: **urg**, **ack**, **psh**, **rst**, **syn**, and **fin**.

Defaults

There are no specific conditions under which a packet is denied passing the named access list.

Command Modes

Access list configuration

Command History

Release	Modification
11.2	This command was introduced.
12.0(1)T	The **time-range** *time-range-name* keyword and argument were added.
12.0(11)	The **fragments** keyword was added.
12.2(13)T	The **igrp** keyword was removed because the IGRP protocol is no longer available in Cisco IOS software.
12.2(14)S	The *sequence-number* argument was added.
12.2(15)T	The *sequence-number* argument was added.
12.3(4)T	The **option** *option-name* keyword and argument were added. The **match-any**, **match-all, +**, and **-** keywords and the *flag-name* argument were added.
12.3(7)T	Command functionality was modified to allow up to ten port numbers to be added after the **eq** and **neq** operators so that an access list entry can be created with noncontiguous ports.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.4(2)T	The **ttl** *operator value* keyword and arguments were added.

Usage Guidelines

Use this command following the **ip access-list** command to specify conditions under which a packet cannot pass the named access list.

The **time-range** keyword allows you to identify a time range by name. The **time-range**, **absolute**, and **periodic** commands specify when this **deny** statement is in effect.

log Keyword

A log message includes the access list number, whether the packet was permitted or denied; the protocol, whether it was TCP, UDP, ICMP, or a number; and, if appropriate, the source and destination addresses and source and destination port numbers. The message is generated for the first packet that matches, and then at 5-minute intervals, including the number of packets permitted or denied in the prior 5-minute interval.

Use the **ip access-list log-update** command to generate logging messages when the number of matches reaches a configurable threshold (rather than waiting for a 5-minute-interval). See the **ip access-list log-update** command for more information.

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

If you enable Cisco Express Forwarding (CEF) and then create an access list that uses the **log** keyword, the packets that match the access list are not CEF-switched. They are fast-switched. Logging disables CEF.

Access List Filtering of IP Options

Access control lists can be used to filter packets with IP Options to prevent routers from being saturated with spurious packets containing IP Options. To see a complete table of all IP Options, including ones currently not in use, refer to the latest Internet Assigned Numbers Authority (IANA) information that is available from its URL: www.iana.org.

Cisco IOS software allows you to filter packets according to whether they contain one or more of the legitimate IP Options by entering either the IP Option value or the corresponding name for the *option-name* argument as shown in Table 1.

Table 1 IP Option Values and Names

IP Option Value or Name	Description
0 to 255	IP Options values.
add-ext	Match packets with Address Extension Option (147).
any-options	Match packets with any IP Option.
com-security	Match packets with Commercial Security Option (134).
dps	Match packets with Dynamic Packet State Option (151).
encode	Match packets with Encode Option (15).
eool	Match packets with End of Options (0).
ext-ip	Match packets with Extended IP Options (145).
ext-security	Match packets with Extended Security Option (133).
finn	Match packets with Experimental Flow Control Option (205).
imitd	Match packets with IMI Traffic Descriptor Option (144).
lsr	Match packets with Loose Source Route Option (131).
mtup	Match packets with MTU Probe Option (11).
mtur	Match packets with MTU Reply Option (12).
no-op	Match packets with No Operation Option (1).
nsapa	Match packets with NSAP Addresses Option (150).
record-route	Match packets with Router Record Route Option (7).
router-alert	Match packets with Router Alert Option (148).
sdb	Match packets with Selective Directed Broadcast Option (149).
security	Match packets with Base Security Option (130).
ssr	Match packets with Strict Source Routing Option (137).
stream-id	Match packets with Stream ID Option (136).
timestamp	Match packets with Time Stamp Option (68).
traceroute	Match packets with Trace Route Option (82).
ump	Match packets with Upstream Multicast Packet Option (152).
visa	Match packets with Experimental Access Control Option (142).
zsu	Match packets with Experimental Measurement Option (10).

Filtering IP Packets Based on TCP Flags

The access list entries that make up an access list can be configured to detect and drop unauthorized TCP packets by allowing only the packets that have very specific groups of TCP flags set or not set. Users can select any desired combination of TCP flags with which to filter TCP packets. Users can configure access list entries in order to allow matching on a flag that is set and on a flag that is not set. Use the **+** and **-** keywords with a flag name to specify that a match is made based on whether a TCP header flag has been set. Use the **match-any** and **match-all** keywords to allow the packet if any or all, respectively, of the flags specified by the **+** or **-** keyword and *flag-name* argument have been set or not set.

Access List Processing of Fragments

The behavior of access list entries regarding the use or lack of use of the **fragments** keyword can be summarized as follows:

If the Access-List Entry Has...	Then...
...no **fragments** keyword (the default behavior), and assuming all of the access-list entry information matches,	For an access list entry that contains only Layer 3 information: • The entry is applied to nonfragmented packets, initial fragments, and noninitial fragments. For an access list entry that contains Layer 3 and Layer 4 information: • The entry is applied to nonfragmented packets and initial fragments. – If the entry is a **permit** statement, then the packet or fragment is permitted. – If the entry is a **deny** statement, then the packet or fragment is denied. • The entry is also applied to noninitial fragments in the following manner. Because noninitial fragments contain only Layer 3 information, only the Layer 3 portion of an access list entry can be applied. If the Layer 3 portion of the access list entry matches, and – If the entry is a **permit** statement, then the noninitial fragment is permitted. – If the entry is a **deny** statement, then the next access list entry is processed. **Note** The **deny** statements are handled differently for noninitial fragments versus nonfragmented or initial fragments.
...the **fragments** keyword, and assuming all of the access-list entry information matches,	The access list entry is applied only to noninitial fragments. The **fragments** keyword cannot be configured for an access list entry that contains any Layer 4 information.

Be aware that you should not add the **fragments** keyword to every access list entry because the first fragment of the IP packet is considered a nonfragment and is treated independently of the subsequent fragments. An initial fragment will not match an access list **permit** or **deny** entry that contains the **fragments** keyword. The packet is compared to the next access list entry, and so on, until it is either permitted or denied by an access list entry that does not contain the **fragments** keyword. Therefore, you may need two access list entries for every **deny** entry. The first **deny** entry of the pair will not include the **fragments** keyword and applies to the initial fragment. The second **deny** entry of the pair will include the **fragments** keyword and applies to the subsequent fragments. In the cases in which there are multiple **deny** access list entries for the same host but with different Layer 4 ports, a single **deny** access list entry with the **fragments** keyword for that host is all that needs to be added. Thus all the fragments of a packet are handled in the same manner by the access list.

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

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

Fragments and Policy Routing

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

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

Creating an Access List Entry with Noncontiguous Ports

For Cisco IOS Release 12.3(7)T and later releases, you can specify noncontiguous ports on the same access control entry, which greatly reduces the number of access list entries required for the same source address, destination address, and protocol. If you maintain large numbers of access list entries, we recommend that you consolidate them when possible by using noncontiguous ports. You can specify up to ten port numbers following the **eq** and **neq** operators.

Examples

The following example sets conditions for a standard access list named Internetfilter:

```
ip access-list standard Internetfilter
 deny 192.168.34.0  0.0.0.255
 permit 172.16.0.0  0.0.255.255
 permit 10.0.0.0  0.255.255.255
! (Note: all other access implicitly denied.)
```

The following example denies HTTP traffic on Monday through Friday from 8:00 a.m. to 6:00 p.m.:

```
time-range no-http
 periodic weekdays 8:00 to 18:00
!
ip access-list extended strict
 deny tcp any any eq http time-range no-http
!
interface ethernet 0
 ip access-group strict in
```

The following example adds an entry with the sequence number 25 to extended IP access list 150:

```
ip access-list extended 150
 25 deny ip host 172.16.3.3 host 192.168.5.34
```

The following example removes the entry with the sequence number 25 from the extended access list example shown above:

```
 no 25
```

The following example sets a deny condition for an extended access list named filter2. The access list entry specifies that a packet cannot pass the named access list if it contains the Strict Source Routing IP Option, which is represented by the IP option value ssr.

```
ip access-list extended filter2
 deny ip any any option ssr
```

The following example sets a deny condition for an extended access list named kmdfilter1. The access list entry specifies that a packet cannot pass the named access list if the RST and FIN TCP flags have been set for that packet:

```
ip access-list extended kmdfilter1
 deny tcp any any match-any +rst +fin
```

The following example shows several **deny** statements that can be consolidated into one access list entry with noncontiguous ports. The **show access-lists** command is entered to display a group of access list entries for the access list named abc.

```
Router# show access-lists abc

Extended IP access list abc
 10 deny tcp any eq telnet any eq 450
 20 deny tcp any eq telnet any eq 679
 30 deny tcp any eq ftp any eq 450
 40 deny tcp any eq ftp any eq 679
```

Because the entries are all for the same **deny** 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
 deny tcp any eq telnet ftp any eq 450 679
```

The following examples shows the creation of the consolidated access list entry:

```
Router# show access-lists abc

Extended IP access list abc
 10 deny tcp any eq telnet ftp any eq 450 679
```

The following access list filters IP packets containing Type of Service (ToS) level 3 with TTL values 10 and 20. It also filters IP packets with a TTL greater than 154 and applies that rule to noninitial fragments. It permits IP packets with a precedence level of flash and a TTL not equal to 1, and sends log messages about such packets to the console. All other packets are denied.

```
ip access-list extended canton
 deny ip any any tos 3 ttl eq 10 20
 deny ip any any ttl gt 154 fragments
 permit ip any any precedence flash ttl neq 1 log
```

Related Commands

Command	Description
absolute	Specifies an absolute time when a time range is in effect.
access-list (IP extended)	Defines an extended IP access list.

Command	Description
access-list (IP standard)	Defines a standard IP access list.
ip access-group	Controls access to an interface.
ip access-list	Defines an IP access list by name.
ip access-list log-update	Sets the threshold number of packets that cause a logging message.
ip access-list resequence	Applies sequence numbers to the access list entries in an access list.
ip options	Drops or ignores IP Options packets that are sent to the router.
logging console	Sends system logging (syslog) messages to all available TTY lines and limits messages based on severity.
match ip address	Distributes any routes that have a destination network number address that is permitted by a standard or extended access list, or performs policy routing on packets.
periodic	Specifies a recurring (weekly) time range for functions that support the time-range feature.
permit (IP)	Sets conditions under which a packet passes a named IP access list.
remark	Writes a helpful comment (remark) for an entry in a named IP access list.
show access-lists	Displays a group of access-list entries.
show ip access-list	Displays the contents of all current IP access lists.
time-range	Specifies when an access list or other feature is in effect.

permit (IP)

To set conditions to allow a packet to pass a named IP access list, use the **permit** command in access list configuration mode. To remove a permit condition from an access list, use the **no** form of this command.

[*sequence-number*] **permit** *source* [*source-wildcard*]

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

no *sequence-number*

no permit *source* [*source-wildcard*]

no permit *protocol source source-wildcard destination destination-wildcard* [**option** *option-name*] [**precedence** *precedence*] [**tos** *tos*] [**ttl** *operator value*] [**log**] [**time-range** *time-range-name*] [**fragments**]

Internet Control Message Protocol (ICMP)

[*sequence-number*] **permit icmp** *source source-wildcard destination destination-wildcard* [*icmp-type* [*icmp-code*] | *icmp-message*] [**precedence** *precedence*] [**tos** *tos*] [**ttl** *operator value*] [**log**] [**time-range** *time-range-name*] [**fragments**]

Internet Group Management Protocol (IGMP)

[*sequence-number*] **permit igmp** *source source-wildcard destination destination-wildcard* [*igmp-type*] [**precedence** *precedence*] [**tos** *tos*] [**ttl** *operator value*] [**log**] [**time-range** *time-range-name*] [**fragments**]

Transmission Control Protocol (TCP)

[*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*] [**ttl** *operator value*] [**log**] [**time-range** *time-range-name*] [**fragments**]

User Datagram Protocol (UDP)

[*sequence-number*] **permit udp** *source source-wildcard* [*operator* [*port*]] *destination destination-wildcard* [*operator* [*port*]] [**precedence** *precedence*] [**tos** *tos*] [**ttl** *operator value*] [**log**] [**time-range** *time-range-name*] [**fragments**]

Syntax Description

sequence-number	(Optional) Sequence number assigned to the permit statement. The sequence number causes the system to insert the statement in that numbered position in the access list.
source	Number of the network or host from which the packet is being sent. There are three alternative ways to specify the source: • Use a 32-bit quantity in four-part dotted-decimal format. • Use the **any** keyword as an abbreviation for a *source* and *source-wildcard* of 0.0.0.0 255.255.255.255. • Use **host** *source* as an abbreviation for a *source* and *source-wildcard* of *source* 0.0.0.0.
source-wildcard	Wildcard bits to be applied to source. There are three alternative ways to specify the source wildcard: • Use a 32-bit quantity in four-part dotted-decimal format. Place 1s in the bit positions that you want to ignore. • Use the **any** keyword as an abbreviation for a *source* and *source-wildcard* of 0.0.0.0 255.255.255.255. • Use **host** *source* as an abbreviation for a *source* and *source-wildcard* of *source* 0.0.0.0.
protocol	Name or number of an Internet protocol. The *protocol* argument can be one of the keywords **eigrp**, **gre**, **icmp**, **igmp**, **ip**, **ipinip**, **nos**, **ospf**, **tcp**, or **udp**, or an integer in the range from 0 to 255 representing an Internet protocol number. To match any Internet protocol (including ICMP, TCP, and UDP), use the **ip** keyword. **Note** When the **icmp**, **igmp**, **tcp,** and **udp** keywords are entered, they must be followed with the specific command syntax that is shown for the ICMP, IGMP, TCP, and UDP forms of the **permit** command.
icmp	Permits only ICMP packets. When you enter the **icmp** keyword, you must use the specific command syntax shown for the ICMP form of the **permit** command.
igmp	Permits only IGMP packets. When you enter the **igmp** keyword, you must use the specific command syntax shown for the IGMP form of the **permit** command.
tcp	Permits only TCP packets. When you enter the **tcp** keyword, you must use the specific command syntax shown for the TCP form of the **permit** command.
udp	Permits only UDP packets. When you enter the **udp** keyword, you must use the specific command syntax shown for the UDP form of the **permit** command.

destination	Number of the network or host to which the packet is being sent. There are three alternative ways to specify the destination: • Use a 32-bit quantity in four-part dotted-decimal format. • Use the **any** keyword as an abbreviation for the *destination* and *destination-wildcard* of 0.0.0.0 255.255.255.255. • Use **host** *destination* as an abbreviation for a *destination* and *destination-wildcard* of *destination* 0.0.0.0.
destination-wildcard	Wildcard bits to be applied to the destination. There are three alternative ways to specify the destination wildcard: • Use a 32-bit quantity in four-part dotted-decimal format. Place 1s in the bit positions that you want to ignore. • Use the **any** keyword as an abbreviation for a *destination* and *destination-wildcard* of 0.0.0.0 255.255.255.255. • Use **host** *destination* as an abbreviation for a *destination* and *destination-wildcard* of *destination* 0.0.0.0.
option *option-name*	(Optional) Packets can be filtered by IP Options, as specified by a number from 0 to 255, or by the corresponding IP Option name, as listed in Table 2 in the “Usage Guidelines” section.
precedence *precedence*	(Optional) Packets can be filtered by precedence level, as specified by a number from 0 to 7 or by a name.
tos *tos*	(Optional) Packets can be filtered by type of service (ToS) level, as specified by a number from 0 to 15, or by a name as listed in the “Usage Guidelines” section of the **access-list** (IP extended) command.
ttl *operator value*	(Optional) Compares the TTL value in the packet to the TTL value specified in this **permit** statement. • The *operator* can be **lt** (less than), **gt** (greater than), **eq** (equal), **neq** (not equal), or **range** (inclusive range). • The *value* can range from 0 to 255. • If the operator is **range**, specify two values separated by a space. • For Release 12.0S, if the operator is **eq** or **neq**, only one TTL value can be specified. • For all other releases, if the operator is **eq** or **neq**, as many as 10 TTL values can be specified, separated by a space.
log	(Optional) Causes an informational logging message about the packet that matches the entry to be sent to the console. (The level of messages logged to the console is controlled by the **logging console** command.)
time-range *time-range-name*	(Optional) Name of the time range that applies to this **permit** statement. The name of the time range and its restrictions are specified by the **time-range** and **absolute** or **periodic** commands, respectively.
fragments	(Optional) The access list entry applies to noninitial fragments of packets; the fragment is either permitted or denied accordingly. For more details about the **fragments** keyword, see the “Access List Processing of Fragments” and “Fragments and Policy Routing” sections in the “Usage Guidelines” section.

icmp-type	(Optional) ICMP packets can be filtered by ICMP message type. The type is a number from 0 to 255.
icmp-code	(Optional) ICMP packets that are filtered by ICMP message type can also be filtered by the ICMP message code. The code is a number from 0 to 255.
icmp-message	(Optional) ICMP packets can be filtered by an ICMP message type name or an ICMP message type and code name. The possible names are listed in the "Usage Guidelines" section of the **access-list** (IP extended) command.
igmp-type	(Optional) IGMP packets can be filtered by IGMP message type or message name. A message type is a number from 0 to 15. IGMP message names are listed in the "Usage Guidelines" section of the **access-list** (IP extended) command.
operator	(Optional) Compares source or destination ports. 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. Up to ten port numbers can be entered for the **eq** (equal) and **neq** (not equal) operators. All other operators require one port number.
port	(Optional) The decimal number or name of a TCP or UDP port. A port number is a number from 0 to 65535. TCP and UDP port names are listed in the "Usage Guidelines" section of the **access-list (IP extended)** command. TCP port names can be used only when filtering TCP. UDP port names can be used only when filtering UDP.
established	(Optional) For the TCP protocol only: Indicates an established connection. A match occurs if the TCP datagram has the ACK or RST bit set. The nonmatching case is that of the initial TCP datagram to form a connection. **Note** The **established** keyword can be used only with the old command-line interface (CLI) format. To use the new CLI format, you must use the **match-any** or **match-all** keywords followed by the **+** or **-** keywords and *flag-name* argument.

{**match-any** | **match-all**}	(Optional) For the TCP protocol only: A match occurs if the TCP datagram has certain TCP flags set or not set. You use the **match-any** keyword to allow a match to occur if any of the specified TCP flags are present, or you can use the **match-all** keyword to allow a match to occur only if all of the specified TCP flags are present. You must follow the **match-any** and **match-all** keywords with the **+** or **-** keyword and the *flag-name* argument to match on one or more TCP flags.
{+ | -} *flag-name*	(Optional) For the TCP protocol only: The **+** keyword matches IP packets if their TCP headers contain the TCP flags that are specified by the *flag-name* argument. The **-** keyword matches IP packets that do not contain the TCP flags specified by the *flag-name* argument. You must follow the **+** and **-** keywords with the *flag-name* argument. TCP flag names can be used only when filtering TCP. Flag names for the TCP flags are as follows: **urg**, **ack**, **psh**, **rst**, **syn**, and **fin**.

Syntax Description

There are no specific conditions under which a packet passes the named access list.

Command Modes

Access list configuration

Command History

Release	Modification
11.2	This command was introduced.
12.0(1)T	The **time-range** *time-range-name* keyword and argument were added.
12.0(11)	The **fragments** keyword was added.
12.2(13)T	The **igrp** keyword was removed because the IGRP protocol is no longer available in Cisco IOS software.
12.2(14)S	The *sequence-number* argument was added.
12.2(15)T	The *sequence-number* argument was added.
12.3(4)T	The **option** *option-name* keyword and argument were added. The **match-any**, **match-all, +** and **-** keywords and the *flag-name* argument were added.
12.3(7)T	Command functionality was modified to allow up to ten port numbers to be added after the **eq** and **neq** operators so that an access list entry can be created with noncontiguous ports.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.4(2)T	The **ttl** *operator value* keyword and arguments were added.

Usage Guidelines

Use this command following the **ip access-list** command to define the conditions under which a packet passes the named access list.

The **time-range** keyword allows you to identify a time range by name. The **time-range**, **absolute**, and **periodic** commands specify when this **permit** statement is in effect.

log Keyword

A log message includes the access list number, whether the packet was permitted or denied; the protocol, whether it was TCP, UDP, ICMP, or a number; and, if appropriate, the source and destination addresses and source and destination port numbers. The message is generated for the first packet that matches, and then at 5-minute intervals, including the number of packets permitted or denied in the prior 5-minute interval.

Use the **ip access-list log-update** command to generate logging messages when the number of matches reaches a configurable threshold (rather than waiting for a 5-minute-interval). See the **ip access-list log-update** command for more information.

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

If you enable Cisco Express Forwarding (CEF) and then create an access list that uses the **log** keyword, the packets that match the access list are not CEF-switched. They are fast-switched. Logging disables CEF.

Access List Filtering of IP Options

Access control lists can be used to filter packets with IP Options to prevent routers from being saturated with spurious packets containing IP Options. To see a complete table of all IP Options, including ones currently not in use, refer to the latest Internet Assigned Numbers Authority (IANA) information that is available from their URL: www.iana.org.

Cisco IOS software allows you to filter packets according to whether they contain one or more of the legitimate IP Options by entering either the IP Option value or the corresponding name for the *option-name* argument as shown in Table 2.

Table 2 IP Option Values and Names

IP Option Value or Name	Description
0 to 255	IP Options values.
add-ext	Match packets with Address Extension Option (147).
any-options	Match packets with any IP Option.
com-security	Match packets with Commercial Security Option (134).
dps	Match packets with Dynamic Packet State Option (151).
encode	Match packets with Encode Option (15).
eool	Match packets with End of Options (0).
ext-ip	Match packets with Extended IP Options (145).
ext-security	Match packets with Extended Security Option (133).
finn	Match packets with Experimental Flow Control Option (205).
imitd	Match packets with IMI Traffic Descriptor Option (144).
lsr	Match packets with Loose Source Route Option (131).
mtup	Match packets with MTU Probe Option (11).
mtur	Match packets with MTU Reply Option (12).
no-op	Match packets with the No Operation Option (1).
nsapa	Match packets with the NSAP Addresses Option (150).

Table 2 IP Option Values and Names (continued)

IP Option Value or Name	Description
record-route	Match packets with Router Record Route Option (7).
router-alert	Match packets with Router Alert Option (148).
sdb	Match packets with Selective Directed Broadcast Option (149).
security	Match packets with Base Security Option (130).
ssr	Match packets with Strict Source Routing Option (137).
stream-id	Match packets with Stream ID Option (136).
timestamp	Match packets with Time Stamp Option (68).
traceroute	Match packets with Trace Route Option (82).
ump	Match packets with Upstream Multicast Packet Option (152).
visa	Match packets with Experimental Access Control Option (142).
zsu	Match packets with Experimental Measurement Option (10).

Filtering IP Packets Based on TCP Flags

The access list entries that make up an access list can be configured to detect and drop unauthorized TCP packets by allowing only the packets that have very specific groups of TCP flags set or not set. Users can select any desired combination of TCP flags with which to filter TCP packets. Users can configure access list entries in order to allow matching on a flag that is set and on a flag that is not set. Use the **+** and **-** keywords with a flag name to specify that a match is made based on whether a TCP header flag has been set. Use the **match-any** and **match-all** keywords to allow the packet if any or all, respectively, of the flags specified by the **+** or **-** keyword and *flag-name* argument have been set or not set.

Access List Processing of Fragments

The behavior of access list entries regarding the use or lack of use of the **fragments** keyword can be summarized as follows:

If the Access-List Entry Has...	Then...
...no **fragments** keyword (the default behavior), and assuming all of the access-list entry information matches,	For an access list entry that contains only Layer 3 information: • The entry is applied to nonfragmented packets, initial fragments, and noninitial fragments. For an access list entry that contains Layer 3 and Layer 4 information: • The entry is applied to nonfragmented packets and initial fragments. – If the entry is a **permit** statement, then the packet or fragment is permitted. – If the entry is a **deny** statement, then the packet or fragment is denied. • The entry is also applied to noninitial fragments in the following manner. Because noninitial fragments contain only Layer 3 information, only the Layer 3 portion of an access list entry can be applied. If the Layer 3 portion of the access list entry matches, and – If the entry is a **permit** statement, then the noninitial fragment is permitted. – If the entry is a **deny** statement, then the next access list entry is processed. **Note** The **deny** statements are handled differently for noninitial fragments versus nonfragmented or initial fragments.
...the **fragments** keyword, and assuming all of the access list entry information matches,	The access list entry is applied only to noninitial fragments. The **fragments** keyword cannot be configured for an access list entry that contains any Layer 4 information.

Be aware that you should not add the **fragments** keyword to every access list entry because the first fragment of the IP packet is considered a nonfragment and is treated independently of the subsequent fragments. An initial fragment will not match an access list **permit** or **deny** entry that contains the **fragments** keyword. The packet is compared to the next access list entry, and so on, until it is either permitted or denied by an access list entry that does not contain the **fragments** keyword. Therefore, you may need two access list entries for every **deny** entry. The first **deny** entry of the pair will not include the **fragments** keyword and applies to the initial fragment. The second **deny** entry of the pair will include the **fragments** keyword and applies to the subsequent fragments. In the cases in which there are multiple **deny** access list entries for the same host but with different Layer 4 ports, a single **deny** access list entry with the **fragments** keyword for that host is all that needs to be added. Thus all the fragments of a packet are handled in the same manner by the access list.

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

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

Fragments and Policy Routing

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

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

Creating an Access List Entry with Noncontiguous Ports

For Cisco IOS Release 12.3(7)T and later releases, you can specify noncontiguous ports on the same access control entry, which greatly reduces the number of access list entries required for the same source address, destination address, and protocol. If you maintain large numbers of access list entries, we recommend that you consolidate them when possible by using noncontiguous ports. You can specify up to ten port numbers following the **eq** and **neq** operators.

Examples

The following example sets conditions for a standard access list named Internetfilter:

```
ip access-list standard Internetfilter
 deny 192.168.34.0  0.0.0.255
 permit 172.16.0.0  0.0.255.255
 permit 10.0.0.0  0.255.255.255
! (Note: all other access implicitly denied).
```

The following example permits Telnet traffic on Mondays, Tuesdays, and Fridays from 9:00 a.m. to 5:00 p.m.:

```
time-range testing
 periodic Monday Tuesday Friday 9:00 to 17:00
!
ip access-list extended legal
 permit tcp any any eq telnet time-range testing
!
interface ethernet0
 ip access-group legal in
```

The following example sets a permit condition for an extended access list named filter2. The access list entry specifies that a packet may pass the named access list only if it contains the NSAP Addresses IP Option, which is represented by the IP Option value nsapa.

```
ip access-list extended filter2
 permit ip any any option nsapa
```

The following example sets a permit condition for an extended access list named kmdfilter1. The access list entry specifies that a packet can pass the named access list only if the RST IP flag has been set for that packet:

```
ip access-list extended kmdfilter1
 permit tcp any any match-any +rst
```

The following example sets a permit condition for an extended access list named kmdfilter1. The access list entry specifies that a packet can pass the named access list only if the RST and FIN TCP flags have been set for that packet:

```
ip access-list extended kmdfilter1
 permit tcp any any match-any +rst +fin
```

The following example shows how to verify the access list by using the **show access-lists** command and then to add an entry to an existing access list:

```
Router# show access-lists

Standard IP access list 1
2 permit 10.0.0.0, wildcard bits 0.0.255.255
5 permit 10.0.0.0, wildcard bits 0.0.255.255
10 permit 10.0.0.0, wildcard bits 0.0.255.255
20 permit 10.0.0.0, wildcard bits 0.0.255.255

ip access-list standard 1
 15 permit 10.0.0.0 0.0.255.255
```

The following examples shows how the entry with the sequence number of 20 is removed from the access list:

```
ip access-list standard 1
 no 20

!Verify that the list has been removed.

Router# show access-lists

Standard IP access list 1
10 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
```

The following examples shows how, if a user tries to enter an entry that is a duplicate of an entry already on the list, no changes occur. The entry that the user is trying to add is a duplicate of the entry already in the access list with a sequence number of 20.

```
Router# show access-lists 101

Extended IP access list 101
    10 permit ip host 10.0.0.0 host 10.5.5.34
    20 permit icmp any any
    30 permit ip host 10.0.0.0 host 10.2.54.2
    40 permit ip host 10.0.0.0 host 10.3.32.3 log

ip access-list extended 101
 100 permit icmp any any

Router# show access-lists 101

Extended IP access list 101
    10 permit ip host 10.3.3.3 host 10.5.5.34
    20 permit icmp any any
    30 permit ip host 10.34.2.2 host 10.2.54.2
    40 permit ip host 10.3.4.31 host 10.3.32.3 log
```

The following example shows what occurs if a user tries to enter a new entry with a sequence number of 20 when an entry with a sequence number of 20 is already in the list. An error message appears, and no change is made to the access list.

```
Router# show access-lists 101

Extended IP access lists 101
    10 permit ip host 10.3.3.3 host 10.5.5.34
    20 permit icmp any any
    30 permit ip host 10.34.2.2 host 10.2.54.2
    40 permit ip host 10.3.4.31 host 10.3.32.3 log
```

```
ip access-lists extended 101
 20 permit udp host 10.1.1.1 host 10.2.2.2

%Duplicate sequence number.

Router# show access-lists 101

Extended IP access lists 101
    10 permit ip host 10.3.3.3 host 10.5.5.34
    20 permit icmp any any
    30 permit ip host 10.34.2.2 host 10.2.54.2
    40 permit ip host 10.3.4.31 host 10.3.32.3 log
```

The following example shows several **permit** statements that can be consolidated into one access list entry with noncontiguous ports. The **show access-lists** command is entered to display a group of access list entries for the access list named aaa.

```
Router# show access-lists aaa

Extended IP access lists aaa
 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 aaa
 no 10
 no 20
 no 30
 no 40
 permit tcp any eq telnet ftp any eq 450 679
```

The following example shows the creation of the consolidated access list entry:

```
Router# show access-lists aaa

Extended IP access list aaa
 10 permit tcp any eq telnet ftp any eq 450 679
```

The following access list filters IP packets containing Type of Service (ToS) level 3 with TTL values 10 and 20. It also filters IP packets with a TTL greater than 154 and applies that rule to noninitial fragments. It permits IP packets with a precedence level of flash and a TTL not equal to 1, and sends log messages about such packets to the console. All other packets are denied.

```
ip access-list extended canton
 deny ip any any tos 3 ttl eq 10 20
 deny ip any any ttl gt 154 fragments
 permit ip any any precedence flash ttl neq 1 log
```

Related Commands

Command	Description
absolute	Specifies an absolute time when a time range is in effect.
access-list (IP extended)	Defines an extended IP access list.

Command	Description
access-list (IP standard)	Defines a standard IP access list.
deny (IP)	Sets conditions under which a packet does not pass a named IP access list.
ip access-group	Controls access to an interface.
ip access-list log-update	Sets the threshold number of packets that cause a logging message.
ip access-list resequence	Applies sequence numbers to the access list entries in an access list.
ip options	Drops or ignores IP Options packets that are sent to the router.
logging console	Sends system logging (syslog) messages to all available TTY lines and limits messages based on severity.
match ip address	Distributes any routes that have a destination network number address that is permitted by a standard or extended access list, or performs policy routing on packets.
periodic	Specifies a recurring (weekly) time range for functions that support the time-range feature.
show access-lists	Displays a group of access-list entries.
show ip access-list	Displays the contents of all current IP access lists.
time-range	Specifies when an access list or other feature is in effect.