

IP Access List Entry Sequence Numbering

The IP Access List Entry Sequence Numbering feature allows you to apply sequence numbers to **permit** or **deny** statements as well as reorder, add, or remove such statements from a named IP access list. The IP Access List Entry Sequence Numbering feature makes revising IP access lists much easier. Prior to this feature, you could add access list entries to the end of an access list only; therefore, needing to add statements anywhere except at the end of a named IP access list required reconfiguring the entire access list.

- Restrictions for IP Access List Entry Sequence Numbering, on page 1
- Information About IP Access List Entry Sequence Numbering, on page 1
- How to Use Sequence Numbers in an IP Access List, on page 5
- Configuration Examples for IP Access List Entry Sequence Numbering, on page 9
- Additional References, on page 10
- Feature Information for IP Access List Entry Sequence Numbering, on page 12

Restrictions for IP Access List Entry Sequence Numbering

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

Information About IP Access List Entry Sequence Numbering

Purpose of IP Access Lists

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

- Filter incoming packets on an interface.
- Filter outgoing packets on an interface.
- Restrict the contents of routing updates.

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

How an IP Access List Works

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

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

IP Access List Process and Rules

- The software tests the source or destination address or the protocol of each packet being filtered against the conditions in the access list, one condition (**permit** or **deny** statement) at a time.
- If a packet does not match an access list statement, the packet is then tested against the next statement in the list.
- If a packet and an access list statement match, the rest of the statements in the list are skipped and the packet is permitted or denied as specified in the matched statement. The first entry that the packet matches determines whether the software permits or denies the packet. That is, after the first match, no subsequent entries are considered.
- If the access list denies the address or protocol, the software discards the packet and returns an Internet Control Message Protocol (ICMP) Host Unreachable message.
- If no conditions match, the packet is dropped. This is because each access list ends with an unwritten or implicit **deny** statement. That is, if the packet has not been permitted by the time it was tested against each statement, it is denied.
- Because the software stops testing conditions after the first match, the order of the conditions is critical. The same **permit** or **deny** statements specified in a different order could result in a packet being passed under one circumstance and denied in another circumstance.
- If an access list is referenced by name in a command, but the access list does not exist, all packets pass.
- Only one access list per interface, per protocol, per direction is allowed.
- Inbound access lists process packets arriving at the device. Incoming packets are processed before being routed to an outbound interface. An inbound access list is efficient because it saves the overhead of routing lookups if the packet is to be discarded because it is denied by the filtering tests. If the packet is permitted by the tests, it is then processed for routing. For inbound lists, **permit** means continue to process the packet after receiving it on an inbound interface; **deny** means discard the packet.

- Outbound access lists process packets before they leave the device. Incoming packets are routed to the outbound interface and then processed through the outbound access list. For outbound lists, **permit** means send it to the output buffer; **deny** means discard the packet.

Helpful Hints for Creating IP Access Lists

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

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

- Before you add new ACL statements, provide time to the parser to clean up the deletion.

Source and Destination Addresses

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

Wildcard Mask and Implicit Wildcard Mask

When comparing the address bits in an access list entry to a packet being submitted to the access list, address filtering uses wildcard masking to determine whether to check or ignore the corresponding IP address bits. By carefully setting wildcard masks, an administrator can select one or more IP addresses for permit or deny tests.

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

- A wildcard mask bit 0 means check the corresponding bit value.
- A wildcard mask bit 1 means ignore that corresponding bit value.

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

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

Transport Layer Information

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

Benefits IP Access List Entry Sequence Numbering

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

The IP Access List Entry Sequence Numbering feature allows you to add sequence numbers to access list entries and resequence them. When you add a new entry, you can choose the sequence number so that the entry is in a desired position in the access list. If necessary, entries currently in the access list can be resequenced (reordered) to create room to insert the new entry.

Sequence Numbering Behavior

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

```
Exceeded maximum sequence number.
```

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

```
Duplicate sequence number.
```

- If a new access list is entered from global configuration mode, then sequence numbers for that access list are generated automatically.
- Entries that contain a fully qualified 32-bit host address are hashed instead of linked. And entries that define a sub-net are maintained in a linked list that is sorted by the sequence number for speed of ACL classification. When a packet is matched against a standard ACL, the source address is hashed and matched against the hash table. If no match is found, it then searches the linked list for a possible match.
- Distributed support is provided so that the sequence numbers of entries in the Route Processor (RP) and line card (LC) are always synchronized.
- Sequence numbers are not nvgened. That is, the sequence numbers themselves are not saved. In the event that the system is reloaded, the configured sequence numbers revert to the default sequence starting number and increment from that number. The function is provided for backward compatibility with software releases that do not support sequence numbering.
- The IP Access List Entry Sequence Numbering feature works with named standard and extended IP access lists. Because the name of an access list can be designated as a number, numbers are acceptable.

How to Use Sequence Numbers in an IP Access List

Sequencing Access-List Entries and Revising the Access List

This task shows how to assign sequence numbers to entries in a named IP access list and how to add or delete an entry to or from an access list. When completing this task, keep the following points in mind:

- Resequencing the access list entries is optional. The resequencing step in this task is shown as required because that is one purpose of this feature and this task demonstrates that functionality.
- In the following procedure, the **permit** command is shown in Step 5 and the **deny** command is shown in Step 6. However, that order can be reversed. Use the order that suits the need of your configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip access-list resequence** *access-list-name starting-sequence-number increment*
4. **ip access-list** {**standard**| **extended**} *access-list-name*
5. Do one of the following:
   - *sequence-number* **permit** *source source-wildcard*
   - *sequence-number* **permit** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*][**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]
6. Do one of the following:
   - *sequence-number* **deny** *source source-wildcard*
   - *sequence-number* **deny** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*][**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]
7. Do one of the following:
   - *sequence-number* **permit** *source source-wildcard*
   - *sequence-number* **permit** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*][**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]
8. Do one of the following:
   - *sequence-number* **deny** *source source-wildcard*
   - *sequence-number* **deny** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*][**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**]
9. Repeat Step 5 and/or Step 6 to add sequence number statements, as applicable.
10. **end**
11. **show ip access-lists** *access-list-name*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip access-list resequence** *access-list-name starting-sequence-number increment* **Example:** `Device(config)# ip access-list resequence kmd1 100 15`	Resequences the specified IP access list using the starting sequence number and the increment of sequence numbers.
Step 4	**ip access-list** {**standard**| **extended**} *access-list-name* **Example:** `Device(config)# ip access-list standard kmd1`	Specifies the IP access list by name and enters named access list configuration mode. • If you specify **standard**, make sure you subsequently specify **permit** and/or **deny** statements using the standard access list syntax. • If you specify **extended**, make sure you subsequently specify **permit** and/or **deny** statements using the extended access list syntax.
Step 5	Do one of the following: • *sequence-number* **permit** *source source-wildcard* • *sequence-number* **permit** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*][**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**] **Example:** `Device(config-std-nacl)# 105 permit 10.5.5.5 0.0.0 255`	Specifies a permit statement in named IP access list mode. • This access list happens to use a **permit** statement first, but a **deny** statement could appear first, depending on the order of statements you need. • As the prompt indicates, this access list was a standard access list. If you had specified **extended** in Step 4, the prompt for this step would be Device(config-ext-nacl) and you would use the extended **permit** command syntax.
Step 6	Do one of the following: • *sequence-number* **deny** *source source-wildcard* • *sequence-number* **deny** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*][**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**] **Example:** `Device(config-std-nacl)# 105 deny 10.6.6.7 0.0.0 255`	(Optional) Specifies a deny statement in named IP access list mode. • This access list uses a **permit** statement first, but a **deny** statement could appear first, depending on the order of statements you need. • As the prompt indicates, this access list was a standard access list. If you had specified **extended** in Step 4, the prompt for this step would be Device(config-ext-nacl) and you would use the extended **deny** command syntax.
Step 7	Do one of the following: • *sequence-number* **permit** *source source-wildcard* • *sequence-number* **permit** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*][**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**] **Example:**	Specifies a permit statement in named IP access list mode. • This access list happens to use a **permit**statement first, but a **deny** statement could appear first, depending on the order of statements you need. • See the permit (IP) command for additional command syntax to permit upper layer protocols (ICMP, IGMP, TCP, and UDP).

	Command or Action	Purpose
	`Device(config-ext-nacl)# 150 permit tcp any any log`	• Use the **no** *sequence-number* command to delete an entry.
Step 8	Do one of the following: • *sequence-number* **deny** *source source-wildcard* • *sequence-number* **deny** *protocol source source-wildcard destination destination-wildcard* [**precedence** *precedence*][**tos** *tos*] [**log**] [**time-range** *time-range-name*] [**fragments**] **Example:** `Device(config-ext-nacl)# 150 deny tcp any any log`	(Optional) Specifies a deny statement in named IP access list mode. • This access list happens to use a **permit**statement first, but a **deny** statement could appear first, depending on the order of statements you need. • See the deny (IP) command for additional command syntax to permit upper layer protocols (ICMP, IGMP, TCP, and UDP). • Use the **no** *sequence-number* command to delete an entry.
Step 9	Repeat Step 5 and/or Step 6 to add sequence number statements, as applicable.	Allows you to revise the access list.
Step 10	**end** **Example:** `Device(config-std-nacl)# end`	(Optional) Exits the configuration mode and returns to privileged EXEC mode.
Step 11	**show ip access-lists** *access-list-name* **Example:** `Device# show ip access-lists kmd1`	(Optional) Displays the contents of the IP access list.

Examples

Review the output of the **show ip access-lists** command to see that the access list includes the new entries:

```
Device# show ip access-lists kmd1

Standard IP access list kmd1
100 permit 10.4.4.0, wildcard bits 0.0.0.255
105 permit 10.5.5.0, wildcard bits 0.0.0.255
115 permit 10.0.0.0, wildcard bits 0.0.0.255
130 permit 10.5.5.0, wildcard bits 0.0.0.255
145 permit 10.0.0.0, wildcard bits 0.0.0.255
```

Configuration Examples for IP Access List Entry Sequence Numbering

Example: Resequencing Entries in an Access List

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

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

```
Device# show access-list 150

Extended IP access list 150
    10 permit ip host 10.3.3.3 host 172.16.5.34
    20 permit icmp any any
    30 permit tcp any host 10.3.3.3
    40 permit ip host 10.4.4.4 any
    50 Dynamic test permit ip any any
    60 permit ip host 172.16.2.2 host 10.3.3.12
    70 permit ip host 10.3.3.3 any log
    80 permit tcp host 10.3.3.3 host 10.1.2.2
    90 permit ip host 10.3.3.3 any
    100 permit ip any any

Device(config)# ip access-list extended 150
Device(config)# ip access-list resequence 150 1 2
Device(config)# exit

Device# show access-list 150

Extended IP access list 150
    1 permit ip host 10.3.3.3 host 172.16.5.34
    3 permit icmp any any
    10 permit tcp any any eq 22 log
    5 permit tcp any host 10.3.3.3
    7 permit ip host 10.4.4.4 any
    9 Dynamic test permit ip any any
    11 permit ip host 172.16.2.2 host 10.3.3.12
    13 permit ip host 10.3.3.3 any log
    15 permit tcp host 10.3.3.3 host 10.1.2.2
    17 permit ip host 10.3.3.3 any
    19 permit ip any any
```

Example: Adding Entries with Sequence Numbers

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

```
Device# show ip access-list

Standard IP access list tryon
2 permit 10.4.4.2, wildcard bits 0.0.255.255
5 permit 10.0.0.44, wildcard bits 0.0.0.255
10 permit 10.0.0.1, wildcard bits 0.0.0.255
```

```
20 permit 10.0.0.2, wildcard bits 0.0.0.255

Device(config)# ip access-list standard tryon
Device(config-std-nacl)# 15 permit 10.5.5.5 0.0.0.255
Device(config-std-nacl)# exit
Device(config)# exit
Device# show ip access-list

Standard IP access list tryon
2 permit 10.4.0.0, wildcard bits 0.0.255.255
5 permit 10.0.0.0, wildcard bits 0.0.0.255
10 permit 10.0.0.0, wildcard bits 0.0.0.255
15 permit 10.5.5.0, wildcard bits 0.0.0.255
20 permit 10.0.0.0, wildcard bits 0.0.0.255
```

Example: Entry Without Sequence Number

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

```
Device(config)# ip access-list standard 1
Device(config-std-nacl)# permit 10.1.1.1 0.0.0.255
Device(config-std-nacl)# permit 10.2.2.2 0.0.0.255
Device(config-std-nacl)# permit 10.3.3.3 0.0.0.255
Device(config-std-nacl)## exit
Device# show access-list

Standard IP access list 1
10 permit 0.0.0.0, wildcard bits 0.0.0.255
20 permit 0.0.0.0, wildcard bits 0.0.0.255
30 permit 0.0.0.0, wildcard bits 0.0.0.255

Device(config)# ip access-list standard 1
Device(config-std-nacl)# permit 10.4.4.4 0.0.0.255
Device(config-std-nacl)# end
Device(config-std-nacl)## exit
Device# show access-list

Standard IP access list 1
10 permit 0.0.0.0, wildcard bits 0.0.0.255
20 permit 0.0.0.0, wildcard bits 0.0.0.255
30 permit 0.0.0.0, wildcard bits 0.0.0.255
40 permit 0.0.0.0, wildcard bits 0.0.0.255
```

Additional References

Related Documents

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

Related Topic	Document Title
Security commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Security Command Reference*
Secure Shell	Configuring Secure Shell and Secure Shell Version 2 Support feature modules.
Configuring authentication and authorization	Configuring Authentication , Configuring Authorization , and Configuring Accounting feature modules.

Standards

Standards	Title
None	--

MIBs

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

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	--

Technical Assistance

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

Feature Information for IP Access List Entry Sequence Numbering

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

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

Table 1: Feature Information for IP Access List Entry Sequence Numbering

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