



## Identifying Traffic with Access Lists

---

This chapter describes how to identify traffic with access lists. Access lists are used in a variety of features. If your feature uses Modular Policy Framework, you can use an access list to identify traffic within a traffic class map. For more information on Modular Policy Framework, see [Chapter 18, “Using Modular Policy Framework.”](#) This chapter includes the following topics:

- [Access List Overview, page 10-1](#)
- [Adding an Extended Access List, page 10-6](#)
- [Adding an EtherType Access List, page 10-8](#)
- [Adding a Standard Access List, page 10-10](#)
- [Simplifying Access Lists with Object Grouping, page 10-10](#)
- [Adding Remarks to Access Lists, page 10-17](#)
- [Scheduling Extended Access List Activation, page 10-18](#)
- [Logging Access List Activity, page 10-19](#)

For information about IPv6 access lists, see the [“Configuring IPv6 Access Lists”](#) section on [page 9-5](#).

### Access List Overview

Access lists are made up of one or more Access Control Entries. An ACE is a single entry in an access list that specifies a permit or deny rule, and is applied to a protocol, a source and destination IP address or network, and optionally the source and destination ports.

This section includes the following topics:

- [Access List Types, page 10-2](#)
- [Access Control Entry Order, page 10-2](#)
- [Access List Implicit Deny, page 10-3](#)
- [IP Addresses Used for Access Lists When You Use NAT, page 10-3](#)
- [Access List Commitment, page 10-5](#)
- [Maximum Number of ACEs, page 10-5](#)

## Access List Types

Table 10-1 lists the types of access lists and some common uses for them.

**Table 10-1** Access List Types and Common Uses

Access List Use	Access List Type	Description
Control network access for IP traffic (routed and transparent mode)	Extended	The FWSM does not allow any traffic unless it is explicitly permitted by an extended access list.  <b>Note</b> To access the FWSM interface for management access, you do not also need an access list allowing the host IP address. You only need to configure management access according to <a href="#">Chapter 21, “Configuring Management Access.”</a>
Identify traffic for AAA rules	Extended	AAA rules use access lists to identify traffic.
Control network access for IP traffic for a given user	Extended, downloaded from a AAA server per user	You can configure the RADIUS server to download a dynamic access list to be applied to the user, or the server can send the name of an access list that you already configured on the FWSM.
Identify addresses for NAT (policy NAT and NAT exemption)	Extended	Policy NAT lets you identify local traffic for address translation by specifying the source and destination addresses in an extended access list.
Establish VPN access	Extended	You can use an extended access list in VPN commands.
Identify traffic in a traffic class map for Modular Policy	Extended EtherType	Access lists can be used to identify traffic in a class map, which is used for features that support Modular Policy Framework. Features that support Modular Policy Framework include TCP and general connection settings, and inspection.
For transparent firewall mode, control network access for non-IP traffic	EtherType	You can configure an access list that controls traffic based on its EtherType.
Identify OSPF route redistribution	Standard	Standard access lists include only the destination address. You can use a standard access list to control the redistribution of OSPF routes.

## Access Control Entry Order

An access list is made up of one or more Access Control Entries. Depending on the access list type, you can specify the source and destination addresses, the protocol, the ports (for TCP or UDP), the ICMP type (for ICMP), or the EtherType.

Each ACE that you enter for a given access list name is appended to the end of the access list unless you specify the line number in the ACE (extended access lists only).

The order of ACEs is important. When the FWSM decides whether to forward or drop a packet, the FWSM tests the packet against each ACE in the order in which the entries are listed. After a match is found, no more ACEs are checked. For example, if you create an ACE at the beginning of an access list that explicitly permits all traffic, no further statements are ever checked.

You can disable an ACE by making it inactive.

## Access List Implicit Deny

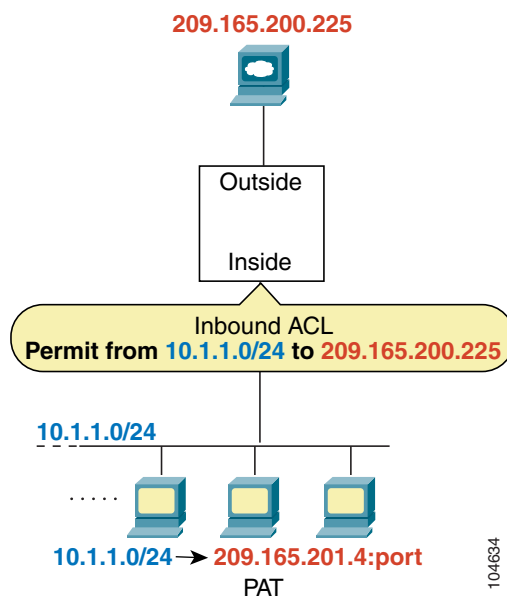
Access lists have an implicit deny at the end of the list, so unless you explicitly permit it, traffic cannot pass. For example, if you want to allow all users to access a network through the FWSM except for particular addresses, then you need to deny the particular addresses and then permit all others.

## IP Addresses Used for Access Lists When You Use NAT

When you use NAT, the IP addresses you specify for an access list depend on the interface to which the access list is attached; you need to use addresses that are valid on the network connected to the interface. This guideline applies for both inbound and outbound access groups: the direction does not determine the address used, only the interface does.

For example, you want to apply an access list to the inbound direction of the inside interface. You configure the FWSM to perform NAT on the inside source addresses when they access outside addresses. Because the access list is applied to the inside interface, the source addresses are the original untranslated addresses. Because the outside addresses are not translated, the destination address used in the access list is the real address (see [Figure 10-1](#)).

**Figure 10-1** IP Addresses in Access Lists: NAT Used for Source Addresses

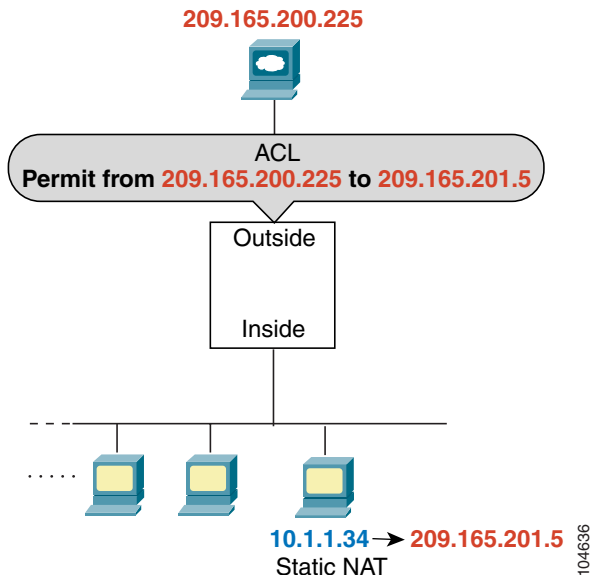


See the following commands for this example:

```
hostname(config)# access-list INSIDE extended permit ip 10.1.1.0 255.255.255.0 host
209.165.200.225
hostname(config)# access-group INSIDE in interface inside
```

If you want to allow an outside host to access an inside host, you can apply an inbound access list on the outside interface. You need to specify the translated address of the inside host in the access list because that address is the address that can be used on the outside network (see [Figure 10-2](#)).

**Figure 10-2** IP Addresses in Access Lists: NAT used for Destination Addresses

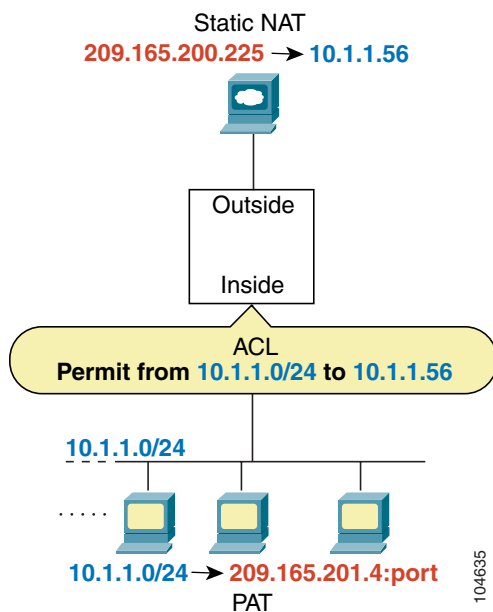


See the following commands for this example:

```
hostname(config)# access-list OUTSIDE extended permit ip host 209.165.200.225 host
209.165.201.5
hostname(config)# access-group OUTSIDE in interface outside
```

If you perform NAT on both interfaces, keep in mind the addresses that are visible to a given interface. In [Figure 10-3](#), an outside server uses static NAT so that a translated address appears on the inside network.

**Figure 10-3** IP Addresses in Access Lists: NAT used for Source and Destination Addresses



See the following commands for this example:

```
hostname(config)# access-list INSIDE extended permit ip 10.1.1.0 255.255.255.0 host
10.1.1.56
hostname(config)# access-group INSIDE in interface inside
```

## Access List Commitment

When you add an ACE to an access list, the FWSM activates the access list by committing it to the network processors. The FWSM waits a short period of time after you last entered an **access-list** command and then commits the access list. If you enter an ACE after the commitment starts, the FWSM aborts the commitment, and recommits the access list after a short waiting period. The FWSM displays a message similar to the following after it commits the access list:

```
Access Rules Download Complete: Memory Utilization: < 1%
```

Large access lists of approximately 60 K ACEs can take 3 to 4 minutes to commit, depending on the size.



### Note

To keep this message from displaying after every access list change and subsequent committal to the network processor, enter the **np acl-notify disable** command. This command is local and not saved in the startup configuration, so it does not replicate to the peer through failover, and you must re-enter the command after each reload.

For information about exceeding memory limits, see the [“Maximum Number of ACEs”](#) section.

## Maximum Number of ACEs

The FWSM supports a maximum of 80 K rules for the entire system in single mode, and 142 K rules for multiple mode. Rules include ACEs, ACEs used for policy NAT, filters, AAA, ICMP, Telnet, SSH, HTTP, and established rules.

Some access lists use more memory than others, and these include access lists that use large port number ranges or overlapping networks (for example one ACE specifies 10.0.0.0/8 and another specifies 10.1.1.0/24, resulting in ACEs with overlapping networks). Depending on the type of access list, the actual limit the system can support will be less than 80 K (single mode) or 142 K (multiple mode).

If you use object groups in ACEs, the number of actual ACEs that you enter is fewer, but the number of *expanded* ACEs is the same as without object groups, and expanded ACEs count towards the system limit. To view the number of expanded ACEs in an access list, enter the **show access-list** command.

When you add an ACE, and the FWSM commits the access list, the console displays the memory used in a message similar to the following:

```
Access Rules Download Complete: Memory Utilization: < 1%
```

If you exceed the memory limitations, you receive an error message and a system message (106024), and all the access lists that were added in this commitment are removed from the configuration. Only the set of access lists that were successfully committed in the previous commitment are used. For example, if you paste 1000 ACEs at the prompt, and the last ACE exceeds the memory limitations, all 1000 ACEs are rejected.

## Adding an Extended Access List

This section describes how to add an extended access list, and includes the following sections:

- [Extended Access List Overview, page 10-6](#)
- [Adding an Extended ACE, page 10-7](#)

### Extended Access List Overview

An extended access list is made up of one or more ACEs, in which you can specify the line number to insert the ACE, source and destination addresses, and, depending on the ACE type, the protocol, the ports (for TCP or UDP), or the ICMP type (for ICMP). You can identify all of these parameters within the **access-list** command, or you can use object groups for each parameter. This section describes how to identify the parameters within the command. To use object groups, see the “[Simplifying Access Lists with Object Grouping](#)” section on page 10-10.

For information about logging options that you can add to the end of the ACE, see the “[Logging Access List Activity](#)” section on page 10-19. For information about time range options, see “[Scheduling Extended Access List Activation](#)” section on page 10-18.

For TCP and UDP connections, you do not need an access list to allow returning traffic, because the FWSM allows all returning traffic for established, bidirectional connections. For connectionless protocols such as ICMP, however, the FWSM establishes unidirectional sessions, so you either need access lists to allow ICMP in both directions (by applying access lists to the source and destination interfaces), or you need to enable the ICMP inspection engine. The ICMP inspection engine treats ICMP sessions as bidirectional connections.

You can apply only one access list of each type (extended and EtherType) to each direction of an interface. You can apply the same access lists on multiple interfaces. See [Chapter 11, “Permitting or Denying Network Access,”](#) for more information about applying an access list to an interface.



#### Note

If you change the access list configuration, and you do not want to wait for existing connections to time out before the new access list information is used, you can clear the connections using the **clear local-host** command.

### Allowing Special IP Traffic through the Transparent Firewall

In routed firewall mode, some types of IP traffic are blocked even if you allow them in an access list, including unsupported dynamic routing protocols and DHCP (unless you configure DHCP relay). Transparent firewall mode can allow any IP traffic through. Because these special types of traffic are connectionless, you need to apply an extended access list to both interfaces, so returning traffic is allowed through.

[Table 10-2](#) lists common traffic types that you can allow through the transparent firewall.

**Table 10-2** *Transparent Firewall Special Traffic*

Traffic Type	Protocol or Port	Notes
BGP	TCP port 179	—
DHCP	UDP ports 67 and 68	If you enable the DHCP server, then the FWSM does not pass DHCP packets.

**Table 10-2** Transparent Firewall Special Traffic (continued)

Traffic Type	Protocol or Port	Notes
EIGRP	Protocol 88	—
OSPF	Protocol 89	—
Multicast streams	The UDP ports vary depending on the application.	Multicast streams are always destined to a Class D address (224.0.0.0 to 239.x.x.x).
RIP (v1 or v2)	UDP port 520	—

## Adding an Extended ACE

When you enter the **access-list** command for a given access list name, the ACE is added to the end of the access list unless you specify the **line** number.

To add an ACE, enter the following command:

```
hostname(config)# access-list access_list_name [line line_number] [extended]
{deny | permit} protocol source_address mask [operator port] dest_address mask
[operator port | icmp_type] [inactive]
```



**Tip**

Enter the access list name in upper case letters so the name is easy to see in the configuration. You might want to name the access list for the interface (for example, **INSIDE**), or for the purpose for which it is created (for example, **NO\_NAT** or **VPN**).

Typically, you identify the **ip** keyword for the protocol, but other protocols are accepted. For a list of protocol names, see the “[Protocols and Applications](#)” section on page D-11.

Enter the **host** keyword before the IP address to specify a single address. In this case, do not enter a mask. Enter the **any** keyword instead of the address and mask to specify any address.

You can specify the source and destination ports only for the **tcp** or **udp** protocols. For a list of permitted keywords and well-known port assignments, see the “[TCP and UDP Ports](#)” section on page D-12. DNS, Discard, Echo, Ident, NTP, RPC, SUNRPC, and Talk each require one definition for TCP and one for UDP. TACACS+ requires one definition for port 49 on TCP.

Use an *operator* to match port numbers used by the source or destination. The permitted operators are as follows:

- **lt**—less than
- **gt**—greater than
- **eq**—equal to
- **neq**—not equal to
- **range**—an inclusive range of values. When you use this operator, specify two port numbers, for example:

```
range 100 200
```

You can specify the ICMP type only for the **icmp** protocol. Because ICMP is a connectionless protocol, you either need access lists to allow ICMP in both directions (by applying access lists to the source and destination interfaces), or you need to enable the ICMP inspection engine (see the “[Adding an ICMP](#)”

[Type Object Group](#)” section on page 10-13). The ICMP inspection engine treats ICMP sessions as stateful connections. To control ping, specify **echo-reply (0)** (FWSM to host) or **echo (8)** (host to FWSM). See the [“Adding an ICMP Type Object Group”](#) section on page 10-13 for a list of ICMP types.

When you specify a network mask, the method is different from the Cisco IOS software **access-list** command. The FWSM uses a network mask (for example, 255.255.255.0 for a Class C mask). The Cisco IOS mask uses wildcard bits (for example, 0.0.0.255).

To make an ACE inactive, use the **inactive** keyword. To reenable it, enter the entire ACE without the **inactive** keyword. This feature lets you keep a record of an inactive ACE in your configuration to make reenabling easier.

See the following examples:

The following access list allows all hosts (on the interface to which you apply the access list) to go through the FWSM:

```
hostname(config)# access-list ACL_IN extended permit ip any any
```

The following sample access list prevents hosts on 192.168.1.0/24 from accessing the 209.165.201.0/27 network. All other addresses are permitted.

```
hostname(config)# access-list ACL_IN extended deny tcp 192.168.1.0 255.255.255.0
209.165.201.0 255.255.255.224
hostname(config)# access-list ACL_IN extended permit ip any any
```

If you want to restrict access to only some hosts, then enter a limited permit ACE. By default, all other traffic is denied unless explicitly permitted.

```
hostname(config)# access-list ACL_IN extended permit ip 192.168.1.0 255.255.255.0
209.165.201.0 255.255.255.224
```

The following access list restricts all hosts (on the interface to which you apply the access list) from accessing a website at address 209.165.201.29. All other traffic is allowed.

```
hostname(config)# access-list ACL_IN extended deny tcp any host 209.165.201.29 eq www
hostname(config)# access-list ACL_IN extended permit ip any any
```

## Adding an EtherType Access List

### Transparent firewall mode only

This section describes how to add an EtherType access list, and includes the following sections:

- [EtherType Access List Overview, page 10-8](#)
- [Adding an Extended ACE, page 10-7](#)

## EtherType Access List Overview

An EtherType access list is made up of one or more ACEs that specify an EtherType. An EtherType ACE controls any EtherType identified by a 16-bit hexadecimal number. EtherType ACEs do not affect IPv4 or ARP packets. EtherType access lists support Ethernet V2 frames. 802.3-formatted frames are not handled by the access list because they use a length field as opposed to a type field. BPDUs, which are handled by the access list, are the only exception: they are SNAP-encapsulated, and the FWSM is designed to specifically handle BPDUs.

The FWSM receives trunk port (Cisco proprietary) BPDUs because FWSM ports are trunk ports. Trunk BPDUs have VLAN information inside the payload, so the FWSM modifies the payload with the outgoing VLAN if you allow BPDUs. If you use failover, you must allow BPDUs on both interfaces with an EtherType access list to avoid bridging loops.

Because EtherTypes are connectionless, you need to apply the access list to both interfaces if you want traffic to pass in both directions.

If you allow MPLS, ensure that LDP and TDP TCP connections are established through the FWSM by configuring both MPLS routers connected to the FWSM to use the IP address on the router interface connected to the FWSM as the router-id for LDP or TDP sessions. (LDP and TDP allow MPLS routers to negotiate the labels (addresses) used to forward packets.)

On Cisco IOS routers, enter the appropriate command for your protocol, LDP or TDP. The *interface* is the interface connected to the FWSM.

```
hostname(config)# mpls ldp router-id interface force
```

Or

```
hostname(config)# tag-switching tdp router-id interface force
```

You can apply only one access list of each type (extended and EtherType) to each direction of an interface. You can also apply the same access lists on multiple interfaces.

## Adding an EtherType ACE

To add an EtherType ACE, enter the following command:

```
hostname(config)# access-list access_list_name ethertype {permit | deny} {ipx | bpdu |
mpls-unicast | mpls-multicast | any | hex_number}
```

The *hex\_number* is any EtherType that can be identified by a 16-bit hexadecimal number greater than or equal to 0x600. See RFC 1700, “Assigned Numbers,” at <http://www.ietf.org/rfc/rfc1700.txt> for a list of EtherTypes.

When you enter the **access-list** command for a given access list name, the ACE is added to the end of the access list.



**Tip**

Enter the *access\_list\_name* in upper case letters so the name is easy to see in the configuration. You might want to name the access list for the interface (for example, INSIDE), or for the purpose (for example, MPLS or IPX).

For example, the following sample access list allows common EtherTypes originating on the inside interface:

```
hostname(config)# access-list ETHER ethertype permit ipx
hostname(config)# access-list ETHER ethertype permit bpdu
hostname(config)# access-list ETHER ethertype permit mpls-unicast
hostname(config)# access-group ETHER in interface inside
```

The following access list allows some EtherTypes through the FWSM, but denies IPX:

```
hostname(config)# access-list ETHER ethertype deny ipx
hostname(config)# access-list ETHER ethertype permit 0x1234
hostname(config)# access-list ETHER ethertype permit bpdu
hostname(config)# access-list ETHER ethertype permit mpls-unicast
hostname(config)# access-group ETHER in interface inside
hostname(config)# access-group ETHER in interface outside
```

The following access list denies traffic with EtherType 0x1256, but allows all others on both interfaces:

```
hostname(config)# access-list nonIP ethertype deny 1256
hostname(config)# access-list nonIP ethertype permit any
hostname(config)# access-group ETHER in interface inside
hostname(config)# access-group ETHER in interface outside
```

## Adding a Standard Access List

Standard access lists are used in some commands to identify the destination IP addresses only. For example, you use a standard access list to identify the destination addresses of OSPF routes for use in a route map for OSPF redistribution. Standard access lists cannot be applied to interfaces to control traffic.

The following command adds a standard ACE. To add another ACE at the end of the access list, enter another **access-list** command specifying the same access list name.

To add an ACE, enter the following command:

```
hostname(config)# access-list access_list_name standard {deny | permit} {any | ip_address mask}
```

The following sample access list identifies routes to 192.168.1.0/24:

```
hostname(config)# access-list OSPF standard permit 192.168.1.0 255.255.255.0
```

## Simplifying Access Lists with Object Grouping

This section describes how to use object grouping to simplify access list creation and maintenance. This section includes the following topics:

- [How Object Grouping Works, page 10-10](#)
- [Adding Object Groups, page 10-11](#)
- [Nesting Object Groups, page 10-14](#)
- [Displaying Object Groups, page 10-17](#)
- [Removing Object Groups, page 10-17](#)
- [Using Object Groups with an Access List, page 10-16](#)

## How Object Grouping Works

By grouping like-objects together, you can use the object group in an ACE instead of having to enter an ACE for each object separately. You can create the following types of object groups:

- Protocol
- Network
- Service
- ICMP type

For example, consider the following three object groups:

- MyServices—Includes the TCP and UDP port numbers of the service requests that are allowed access to the internal network

- **TrustedHosts**—Includes the host and network addresses allowed access to the greatest range of services and servers
- **PublicServers**—Includes the host addresses of servers to which the greatest access is provided

After creating these groups, you could use a single ACE to allow trusted hosts to make specific service requests to a group of public servers.

You can also nest object groups in other object groups.

**Note**

The ACE system limit applies to expanded access lists. If you use object groups in ACEs, the number of actual ACEs that you enter is fewer, but the number of expanded ACEs is the same as without object groups. In many cases, object groups create more ACEs than if you added them manually, because creating ACEs manually leads you to summarize addresses more than an object group does. To view the number of expanded ACEs in an access list, enter the **show access-list** command.

## Adding Object Groups

This section describes how to add object groups, and includes the following topics:

- [Adding a Protocol Object Group, page 10-11](#)
- [Adding a Network Object Group, page 10-12](#)
- [Adding a Service Object Group, page 10-12](#)
- [Adding an ICMP Type Object Group, page 10-13](#)

## Adding a Protocol Object Group

To add or change a protocol object group, perform the following steps. After you add the group, you can add more objects as required by following this procedure again for the same group name and specifying additional objects. You do not need to reenter existing objects; the commands you already set remain in place unless you remove them with the **no** form of the command.

To add a protocol group, perform the following steps:

---

**Step 1** To add a protocol group, enter the following command:

```
hostname(config)# object-group protocol grp_id
```

The *grp\_id* is a text string up to 64 characters in length.

The prompt changes to protocol configuration mode.

**Step 2** (Optional) To add a description, enter the following command:

```
hostname(config-protocol)# description text
```

The description can be up to 200 characters.

**Step 3** To define the protocols in the group, enter the following command for each protocol:

```
hostname(config-protocol)# protocol-object protocol
```

The *protocol* is the numeric identifier of the specific IP protocol (1 to 254) or a keyword identifier (for example, **icmp**, **tcp**, or **udp**). To include all IP protocols, use the keyword **ip**. For a list of protocols you can specify, see the “[Protocols and Applications](#)” section on page D-11.

For example, to create a protocol group for TCP, UDP, and ICMP, enter the following commands:

```
hostname(config)# object-group protocol tcp_udp_icmp
hostname(config-protocol)# protocol-object tcp
hostname(config-protocol)# protocol-object udp
hostname(config-protocol)# protocol-object icmp
```

## Adding a Network Object Group

To add or change a network object group, perform the following steps. After you add the group, you can add more objects as required by following this procedure again for the same group name and specifying additional objects. You do not need to reenter existing objects; the commands you already set remain in place unless you remove them with the **no** form of the command.



**Note** A network object group supports IPv4 and IPv6 addresses, depending on the type of access list. For more information about IPv6 access lists, see “[Configuring IPv6 Access Lists](#)” section on page 9-5.

To add a network group, perform the following steps:

**Step 1** To add a network group, enter the following command:

```
hostname(config)# object-group network grp_id
```

The *grp\_id* is a text string up to 64 characters in length.

The prompt changes to network configuration mode.

**Step 2** (Optional) To add a description, enter the following command:

```
hostname(config-network)# description text
```

The description can be up to 200 characters.

**Step 3** To define the networks in the group, enter the following command for each network or address:

```
hostname(config-network)# network-object {host ip_address | ip_address mask}
```

For example, to create network group that includes the IP addresses of three administrators, enter the following commands:

```
hostname(config)# object-group network admins
hostname(config-network)# description Administrator Addresses
hostname(config-network)# network-object host 10.1.1.4
hostname(config-network)# network-object host 10.1.1.78
hostname(config-network)# network-object host 10.1.1.34
```

## Adding a Service Object Group

To add or change a service object group, perform the following steps. After you add the group, you can add more objects as required by following this procedure again for the same group name and specifying additional objects. You do not need to reenter existing objects; the commands you already set remain in place unless you remove them with the **no** form of the command.

To add a service group, perform the following steps:

**Step 1** To add a service group, enter the following command:

```
hostname(config)# object-group service grp_id {tcp | udp | tcp-udp}
```

The *grp\_id* is a text string up to 64 characters in length.

Specify the protocol for the services (ports) you want to add, either **tcp**, **udp**, or **tcp-udp** keywords. Enter **tcp-udp** keyword if your service uses both TCP and UDP with the same port number, for example, DNS (port 53).

The prompt changes to service configuration mode.

**Step 2** (Optional) To add a description, enter the following command:

```
hostname(config-service)# description text
```

The description can be up to 200 characters.

**Step 3** To define the ports in the group, enter the following command for each port or range of ports:

```
hostname(config-service)# port-object {eq port | range begin_port end_port}
```

For a list of permitted keywords and well-known port assignments, see the [“Protocols and Applications” section on page D-11](#).

For example, to create service groups that include DNS (TCP/UDP), LDAP (TCP), and RADIUS (UDP), enter the following commands:

```
hostname(config)# object-group service services1 tcp-udp
hostname(config-service)# description DNS Group
hostname(config-service)# port-object eq domain

hostname(config-service)# object-group service services2 udp
hostname(config-service)# description RADIUS Group
hostname(config-service)# port-object eq radius
hostname(config-service)# port-object eq radius-acct

hostname(config-service)# object-group service services3 tcp
hostname(config-service)# description LDAP Group
hostname(config-service)# port-object eq ldap
```

## Adding an ICMP Type Object Group

To add or change an ICMP type object group, perform the following steps. After you add the group, you can add more objects as required by following this procedure again for the same group name and specifying additional objects. You do not need to reenter existing objects; the commands you already set remain in place unless you remove them with the **no** form of the command.

To add an ICMP type group, perform the following steps:

**Step 1** To add an ICMP type group, enter the following command:

```
hostname(config)# object-group icmp-type grp_id
```

The *grp\_id* is a text string up to 64 characters in length.

The prompt changes to ICMP type configuration mode.

**Step 2** (Optional) To add a description, enter the following command:

```
hostname(config-icmp-type)# description text
```

The description can be up to 200 characters.

**Step 3** To define the ICMP types in the group, enter the following command for each type:

```
hostname(config-icmp-type)# icmp-object icmp_type
```

See the “ICMP Types” section on page D-15 for a list of ICMP types.

For example, to create an ICMP type group that includes echo-reply and echo (for controlling ping), enter the following commands:

```
hostname(config)# object-group icmp-type ping
hostname(config-service)# description Ping Group
hostname(config-icmp-type)# icmp-object echo
hostname(config-icmp-type)# icmp-object echo-reply
```

## Nesting Object Groups

To nest an object group within another object group of the same type, first create the group that you want to nest according to the “Adding Object Groups” section on page 10-11. Then perform the following steps:

**Step 1** To add or edit an object group under which you want to nest another object group, enter the following command:

```
hostname(config)# object-group {{protocol | network | icmp-type} grp_id | service grp_id
{tcp | udp | tcp-udp}}
```

**Step 2** To add the specified group under the object group you specified in Step 1, enter the following command:

```
hostname(config-group_type)# group-object grp_id
```

The nested group must be of the same type.

You can mix and match nested group objects and regular objects within an object group.

For example, you create network object groups for privileged users from various departments:

```
hostname(config)# object-group network eng
hostname(config-network)# network-object host 10.1.1.5
hostname(config-network)# network-object host 10.1.1.9
hostname(config-network)# network-object host 10.1.1.89

hostname(config-network)# object-group network hr
hostname(config-network)# network-object host 10.1.2.8
hostname(config-network)# network-object host 10.1.2.12

hostname(config-network)# object-group network finance
```

```
hostname(config-network) # network-object host 10.1.4.89
hostname(config-network) # network-object host 10.1.4.100
```

You then nest all three groups together as follows:

```
hostname(config)# object-group network admin
hostname(config-network)# group-object eng
hostname(config-network)# group-object hr
hostname(config-network)# group-object finance
```

You only need to specify the admin object group in your ACE as follows:

```
hostname(config)# access-list ACL_IN extended permit ip object-group admin host
209.165.201.29
```

## Using Object Groups with an Access List

To use object groups in an access list, replace the normal protocol (*protocol*), network (*source\_address\_mask*, and so on), service (*operator port*), or ICMP type (*icmp\_type*) parameter with **object-group grp\_id** parameter.

For example, to use object groups for all available parameters in the **access-list {tcp | udp}** command, enter the following command:

```
hostname(config)# access-list access_list_name [line line_number] [extended] {deny |
permit} {tcp | udp} object-group nw_grp_id [object-group svc_grp_id] object-group
nw_grp_id [object-group svc_grp_id]
```

You do not have to use object groups for all parameters; for example, you can use an object group for the source address, but identify the destination address with an address and mask.

The following normal access list that does not use object groups restricts several hosts on the inside network from accessing several web servers. All other traffic is allowed.

```
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.4 host 209.165.201.29
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.78 host 209.165.201.29
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.89 host 209.165.201.29
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.4 host 209.165.201.16
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.78 host 209.165.201.16
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.89 host 209.165.201.16
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.4 host 209.165.201.78
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.78 host 209.165.201.78
eq www
hostname(config)# access-list ACL_IN extended deny tcp host 10.1.1.89 host 209.165.201.78
eq www
hostname(config)# access-list ACL_IN extended permit ip any any
hostname(config)# access-group ACL_IN in interface inside
```

If you make two network object groups, one for the inside hosts, and one for the web servers, then the configuration can be simplified and can be easily modified to add more hosts:

```
hostname(config)# object-group network denied
hostname(config-network)# network-object host 10.1.1.4
hostname(config-network)# network-object host 10.1.1.78
hostname(config-network)# network-object host 10.1.1.89

hostname(config-network)# object-group network web
hostname(config-network)# network-object host 209.165.201.29
```

```

hostname(config-network)# network-object host 209.165.201.16
hostname(config-network)# network-object host 209.165.201.78

hostname(config-network)# access-list ACL_IN extended deny tcp object-group denied
object-group web eq www
hostname(config)# access-list ACL_IN extended permit ip any any
hostname(config)# access-group ACL_IN in interface inside

```

## Displaying Object Groups

To display a list of the currently configured object groups, enter the following command:

```
hostname(config)# show object-group [protocol | network | service | icmp-type | id grp_id]
```

If you enter the command without any parameters, the system displays all configured object groups.

The following is sample output from the **show object-group** command:

```

hostname# show object-group
object-group network ftp_servers
  description: This is a group of FTP servers
  network-object host 209.165.201.3
  network-object host 209.165.201.4
object-group network TrustedHosts
  network-object host 209.165.201.1
  network-object 192.168.1.0 255.255.255.0
group-object ftp_servers

```

## Removing Object Groups

To remove an object group, enter one of the following commands.



### Note

---

You cannot remove an object group or make an object group empty if it is used in an access list.

---

- To remove a specific object group, enter the following command:

```
hostname(config)# no object-group grp_id
```

- To remove all object groups of the specified type, enter the following command:

```
hostname(config)# clear object-group [protocol | network | services | icmp-type]
```

If you do not enter a type, all object groups are removed.

## Adding Remarks to Access Lists

You can include remarks about entries in any access list, including extended, EtherType, and standard access lists. The remarks make the access list easier to understand.

To add a remark to an access list, enter the following command:

```
hostname(config)# access-list access_list_name [line line_number] remark text
```

When you enter the **access-list remark** command for a given access list name, the remark is added to the end of the access list unless you specify the **line** number.

If you delete an access list using the **clear configure access-list** *access\_list\_name* command, then all the remarks are also removed.

The text can be up to 100 characters in length. You can enter leading spaces at the beginning of the text. Trailing spaces are ignored.

For example, you can add remarks before each ACE, and the remark appears in the access list in this location. Entering a dash (-) at the beginning of the remark helps set it apart from ACEs.

```
hostname(config)# access-list OUT remark - this is the inside admin address
hostname(config)# access-list OUT extended permit ip host 209.168.200.3 any
hostname(config)# access-list OUT remark - this is the hr admin address
hostname(config)# access-list OUT extended permit ip host 209.168.200.4 any
```

## Scheduling Extended Access List Activation

You can schedule each ACE to be activated at specific times of the day and week by applying a time range to the ACE. This section includes the following topics:

- [Adding a Time Range, page 10-18](#)
- [Applying the Time Range to an ACE, page 10-19](#)

### Adding a Time Range

To add a time range to implement a time-based access list, perform the following steps:

**Step 1** Identify the time-range name by entering the following command:

```
hostname(config)# time-range name
```

**Step 2** Specify the time range as either a recurring time range or an absolute time range.



**Note** Users could experience a delay of approximately 80 to 100 seconds after the specified end time for the ACL to become inactive. For example, if the specified end time is 3:50, because the end time is inclusive, the command is picked up anywhere between 3:51:00 and 3:51:59. After the command is picked up, the security appliance finishes any currently running task and then services the command to deactivate the ACL.

Multiple periodic entries are allowed per **time-range** command. If a **time-range** command has both **absolute** and **periodic** values specified, then the **periodic** commands are evaluated only after the **absolute** start time is reached, and are not further evaluated after the **absolute** end time is reached.

- Recurring time range:

```
hostname(config-time-range)# periodic days-of-the-week time to [days-of-the-week] time
```

You can specify the following values for *days-of-the-week*:

- **monday, tuesday, wednesday, thursday, friday, saturday, and sunday.**
- **daily**
- **weekdays**
- **weekend**

The *time* is in the format *hh:mm*. For example, 8:00 is 8:00 a.m. and 20:00 is 8:00 p.m.

- Absolute time range:

```
hostname(config-time-range)# absolute start time date [end time date]
```

The *time* is in the format *hh:mm*. For example, 8:00 is 8:00 a.m. and 20:00 is 8:00 p.m.

The *date* is in the format *day month year*; for example, **1 january 2006**.

The following is an example of an absolute time range beginning at 8:00 a.m. on January 1, 2006. Because no end time and date are specified, the time range is in effect indefinitely.

```
hostname(config)# time-range for2006
hostname(config-time-range)# absolute start 8:00 1 january 2006
```

The following is an example of a weekly periodic time range from 8:00 a.m. to 6:00 p.m. on weekdays.:

```
hostname(config)# time-range workinghours
hostname(config-time-range)# periodic weekdays 8:00 to 18:00
```

## Applying the Time Range to an ACE

To apply the time range to an ACE, enter the following command:

```
hostname(config)# access-list access_list_name [extended] {deny | permit}...[time-range name]
```

See the “Adding an Extended Access List” section on page 10-6 for complete **access-list** command syntax.



### Note

If you also enable logging for the ACE, use the **log** keyword before the **time-range** keyword. If you disable the ACE using the **inactive** keyword, use the **inactive** keyword as the last keyword.

The following example binds an access list named “Sales” to a time range named “New\_York\_Minute.”

```
hostname(config)# access-list Sales line 1 extended deny tcp host 209.165.200.225 host 209.165.201.1 time-range New_York_Minute
```

## Logging Access List Activity

This section describes how to configure access list logging for extended access lists and Webtype access lists.

This section includes the following topics:

- [Access List Logging Overview, page 10-20](#)
- [Configuring Logging for an ACE, page 10-21](#)
- [Managing Deny Flows, page 10-22](#)

## Access List Logging Overview

By default, when traffic is denied by an extended ACE, the FWSM generates system message 106023 for each denied packet, in the following form:

```
%XXX-106023: Deny protocol src [interface_name:source_address/source_port] dst
interface_name:dest_address/dest_port [type {string}, code {code}] by access_group acl_id
```

If the FWSM is attacked, the number of system messages for denied packets can be very large. We recommend that you instead enable logging using system message 106100, which provides statistics for each ACE and lets you limit the number of system messages produced. Alternatively, you can disable all logging.



### Note

Only ACEs in the access list generate logging messages; the implicit deny at the end of the access list does not generate a message. If you want all denied traffic to generate messages, add the implicit ACE manually to the end of the access list, as follows.

```
hostname(config)# access-list TEST deny ip any any log
```

The **log** options at the end of the extended **access-list** command lets you to set the following behavior:

- Enable message 106100 instead of message 106023
- Disable all logging
- Return to the default logging using message 106023

System message 106100 is in the following form:

```
%XXX-n-106100: access-list acl_id {permitted | denied} protocol
interface_name/source_address(source_port) -> interface_name/dest_address(dest_port)
hit-cnt number ((first hit | number-second interval))
```

When you enable logging for message 106100, if a packet matches an ACE, the FWSM creates a flow entry to track the number of packets received within a specific interval. The FWSM generates a system message at the first hit and at the end of each interval, identifying the total number of hits during the interval. At the end of each interval, the FWSM resets the hit count to 0. If no packets match the ACE during an interval, the FWSM deletes the flow entry.



### Note

An ACL only denies SYN packets, so if another type of packet comes in, that packet will not show up in the access-list hit counters. TCP packet types other than SYN packets (including RST, SYN-ACK, ACK, PSH, and FIN) are dropped by the FWSM before they can be dropped by an access list. Only SYN packets can create a session in the Adaptive Security Algorithm, so only SYN packets are assessed by the access list.

A flow is defined by the source and destination IP addresses, protocols, and ports. Because the source port might differ for a new connection between the same two hosts, you might not see the same flow increment because a new flow was created for the connection.

Permitted packets that belong to established connections do not need to be checked against access lists; only the initial packet is logged and included in the hit count. For connectionless protocols, such as ICMP, all packets are logged even if they are permitted, and all denied packets are logged.

See the *Catalyst 6500 Series Switch and Cisco 7600 Series Router Firewall Services Module Logging Configuration and System Log Messages* for detailed information about this system message.

## Configuring Logging for an ACE

To configure logging for an ACE, see the following information about the **log** option:

```
hostname(config)# access-list access_list_name [extended] {deny | permit}...[log [[level]  
[interval secs] | disable | default]]
```

See the “Adding an Extended Access List” section on page 10-6 for complete **access-list** command syntax.



### Note

If you also enable a time range for the ACE, use the **log** keyword before the **time-range** keyword. If you disable the ACE using the **inactive** keyword, use the **inactive** keyword as the last keyword.

If you enter the **log** option without any arguments, you enable system log message 106100 at the default level (6) and for the default interval (300 seconds). See the following options:

- *level*—A severity level between 0 and 7. The default is 6.
- **interval** *secs*—The time interval in seconds between system messages, from 1 to 600. The default is 300. This value is also used as the timeout value for deleting an inactive flow.
- **disable**—Disables all access list logging.
- **default**—Enables logging to message 106023. This setting is the same as having no **log** option.

For example, you configure the following access list:

```
hostname(config)# access-list outside-acl permit ip host 1.1.1.1 any log 7 interval 600  
hostname(config)# access-list outside-acl permit ip host 2.2.2.2 any  
hostname(config)# access-list outside-acl deny ip any any log 2  
hostname(config)# access-group outside-acl in interface outside
```

When a packet is permitted by the first ACE of **outside-acl**, the FWSM generates the following system message:

```
%PIX-7-106100: access-list outside-acl permitted tcp outside/1.1.1.1(12345) ->  
inside/192.168.1.1(1357) hit-cnt 1 (first hit)
```

Although 20 additional packets for this connection arrive on the outside interface, the traffic does not have to be checked against the access list, and the hit count does not increase.

If one more connection by the same host is initiated within the specified 10 minute interval (and the source and destination ports remain the same), then the hit count is incremented by 1 and the following message is displayed at the end of the 10 minute interval:

```
%PIX-7-106100: access-list outside-acl permitted tcp outside/1.1.1.1(12345)->  
inside/192.168.1.1(1357) hit-cnt 2 (600-second interval)
```

When a packet is denied by the third ACE, then the FWSM generates the following system message:

```
%PIX-2-106100: access-list outside-acl denied ip outside/3.3.3.3(12345) ->  
inside/192.168.1.1(1357) hit-cnt 1 (first hit)
```

20 additional attempts within a 5 minute interval (the default) result in the following message at the end of 5 minutes:

```
%PIX-2-106100: access-list outside-acl denied ip outside/3.3.3.3(12345) ->  
inside/192.168.1.1(1357) hit-cnt 21 (300-second interval)
```

## Managing Deny Flows

When you enable logging for message 106100, if a packet matches an ACE, the FWSM creates a flow entry to track the number of packets received within a specific interval. The FWSM has a maximum of 64 K logging flows for ACEs. A large number of flows can exist concurrently at any point of time. To prevent unlimited consumption of memory and CPU resources, the FWSM places a limit on the number of concurrent *deny* flows; the limit is placed only on deny flows (and not permit flows) because they can indicate an attack. When the limit is reached, the FWSM does not create a new deny flow for logging until the existing flows expire.

For example, if someone initiates a DoS attack, the FWSM can create a large number of deny flows in a short period of time. Restricting the number of deny flows prevents unlimited consumption of memory and CPU resources.

When you reach the maximum number of deny flows, the FWSM issues system message 106100:

```
%XXX-1-106101: The number of ACL log deny-flows has reached limit (number).
```

To configure the maximum number of deny flows and to set the interval between deny flow alert messages (106101), enter the following commands:

- To set the maximum number of deny flows permitted per context before the FWSM stops logging, enter the following command:

```
hostname(config)# access-list deny-flow-max number
```

The *number* is between 1 and 4096. 4096 is the default.

- To set the amount of time between system messages (number 106101) that identify that the maximum number of deny flows was reached, enter the following command:

```
hostname(config)# access-list alert-interval secs
```

The *seconds* are between 1 and 3600. 300 is the default.