CHAPTER 10

# Controlling Network Access with Access Control Lists

This chapter tells how to control network access through the Firewall Services Module (FWSM) using access control lists (ACLs). You can also use ACLs for other purposes, for example, to identify addresses for NAT, AAA, or OSPF route redistribution. This chapter describes how to create ACLs for these purposes as well as for network access, but this chapter only describes how to *apply* the ACLs for network access. Refer to the NAT, AAA, or IP chapters for information about applying ACLs for these other purposes.

**Note** You use ACLs to control network access in both routed and transparent firewall modes. In transparent mode, you can use both extended ACLs (for Layer 3 traffic) and EtherType ACLs (for Layer 2 traffic).

This chapter contains the following sections:

## Access Control List Overview

ACLs are made up of one or more Access Control Entries (ACEs). An ACE is a single entry in an ACL 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 Control List Types and Uses

This section includes the following topics:

## Access Control List Type Overview

Table 10-1 lists the types of ACLs you can create and how you can use them.

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

| ACL Use | ACL Type | For more information... |
|---|---|---|
| Control network access for IP traffic | Extended | See the "Controlling Network Access for IP Traffic (Extended)" section on page 10-2. |
| Identify traffic for AAA rules | Extended | See the "Identifying Traffic for AAA rules (Extended)" section on page 10-3. |
| Control network access for IP traffic for a given user | Extended, downloaded from a AAA server per user | See the "Controlling Network Access for IP Traffic for a Given User (Extended)" section on page 10-4. |
| Identify addresses for NAT (policy NAT and NAT exemption) | Extended | See the "Identifying Addresses for Policy NAT and NAT Exemption (Extended)" section on page 10-4. |
| Establish VPN management access | Extended | See the "VPN Management Access (Extended)" section on page 10-5. |
| For transparent firewall mode, control network access for non-IP traffic | EtherType | See the "Controlling Network Access for Non-IP Traffic (EtherType)" section on page 10-5. |
| Identify OSPF route redistribution | Standard | See the "Redistributing OSPF Routes (Standard)" section on page 10-6. |

## Controlling Network Access for IP Traffic (Extended)

Extended ACLs control connections based on source address, destination address, protocol, or port. The FWSM does not allow any traffic through unless it is explicitly permitted by an extended ACL. This rule is true for both routed firewall mode and transparent firewall mode.

For TCP and UDP connections, you do not need an ACL to allow returning traffic, because the FWSM allows all returning traffic for established connections. See the "Stateful Inspection Feature" section on page 1-5 for more information. For connectionless protocols such as ICMP, however, you either need

ACLs to allow ICMP in both directions (by applying ACLs to the source and destination interfaces), or you need to enable the ICMP inspection engine (see the "ICMP Inspection Engine" section on page 13-10). The ICMP inspection engine treats ICMP sessions as stateful connections.

You can apply one ACL of each type to each direction of an interface. You can also apply the same ACLs on multiple interfaces.

To control network access for IP traffic, perform the following task:

- Create and apply the ACL according to the "Adding an Extended Access Control List" section on page 10-13.

### Allowing Special Traffic through the Transparent Firewall

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

Table 10-2 lists common traffic types that you can allow through the transparent firewall. See Appendix D, "Addresses, Protocols, and Ports Reference," for more protocols and ports.

*Table 10-2 Transparent Firewall Special Traffic*

| Traffic Type | Protocol or Port | Notes |
|---|---|---|
| BGP[1] | TCP port 179 | — |
| DHCP[2] | UDP ports 67 and 68 | If you enable the DHCP server, then the FWSM does not pass DHCP packets. |
| EIGRP[3] | Protocol 88 | — |
| 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). |
| OSPF | Protocol 89 | — |
| RIP (v1 or v2) | UDP port 520 | — |

[1] Border Gateway Protocol

[2] Dynamic Host Configuration Protocol

[3] Enhanced Interior Gateway Routing Protocol

## Identifying Traffic for AAA rules (Extended)

ACLs can be used with AAA in several ways.

- To identify traffic for network access authorization using a TACACS+ server, perform the following tasks:

  **a.** Add the ACL using the "Adding an Extended Access Control List" section on page 10-13.

  Permit entries in the ACL mark matching traffic for authorization, while deny entries exclude matching traffic from authorization.

  **b.** Apply the ACL using the **aaa authorization match** command in the "Configuring TACACS+ Authorization" section on page 12-22.

- To identify traffic for network access authentication using a TACACS+ or RADIUS server, perform the following tasks:
  a. Add the ACL using the "Adding an Extended Access Control List" section on page 10-13.

     Permit entries in the ACL mark matching traffic for authentication, while deny entries exclude matching traffic from authentication.
  b. Apply the ACL using the **aaa authentication match** command in the "Configuring Authentication for Network Access" section on page 12-20.
- To identify traffic for network access accounting using a TACACS+ or RADIUS server, perform the following tasks:
  a. Add the ACL using the "Adding an Extended Access Control List" section on page 10-13.

     Permit entries in the ACL mark matching traffic for accounting, while deny entries exclude matching traffic from accounting.
  b. Apply the ACL using the **aaa accounting match** command in the "Configuring Accounting for Network Access" section on page 12-25.

## Controlling Network Access for IP Traffic for a Given User (Extended)

When you configure user authentication for network access, you can also choose to configure user authorization that determines the specific access privileges for each user. If you use a RADIUS server, you can configure the RADIUS server to download a dynamic ACL to be applied to the user, or the server can send the name of an ACL that you already configured on the FWSM. See the following tasks for each method.

- For dynamic ACLs, all ACL configuration takes place on the RADIUS server. Perform the following tasks:
  a. Refer to the "Adding an Extended Access Control List" section on page 10-13 for ACL syntax and guidelines.
  b. To create the ACL on the RADIUS server, see the "Configuring the RADIUS Server to Download Per-User Access Control Lists" section on page 12-23.
- For a downloaded ACL name, perform the following tasks:
  a. Configure an extended ACL according to the "Adding an Extended Access Control List" section on page 10-13.

     This extended ACL is not assigned to an interface, but is designed to be applied to one or more users.
  b. Use the ACL name according to the "Configuring the RADIUS Server to Download Per-User Access Control List Names" section on page 12-25.

These per-user ACLs must be as restrictive or more restrictive than an extended ACL that is assigned to the interface. For example, if the ACL assigned to the inside interface allows all users to have only HTTP access to other networks, it would not make sense to configure an authorization ACL for that user to access FTP.

## Identifying Addresses for 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 ACL. You can also optionally specify the source and destination ports. Regular NAT can only consider the local addresses.

NAT exemption statements also use ACLs, but you cannot specify the ports.

To use ACLs with NAT, perform the following tasks:

1. Add the ACL using the "Adding an Extended Access Control List" section on page 10-13. This ACL can contain only permit elements. Specify ports using the **eq** operator.
2. Use the ACL in the **nat** and **static** commands described in the following sections:
   - "Using Dynamic NAT and PAT" section on page 9-16
   - "Using Static NAT" section on page 9-26
   - "Using Static PAT" section on page 9-27
   - "Configuring Static Identity NAT" section on page 9-30
   - "Configuring NAT Exemption" section on page 9-31

## VPN Management Access (Extended)

You can use an extended ACL in VPN commands. See the following tasks for each method.

- To identify hosts allowed to connect to the FWSM over an IPSec site-to-site tunnel, perform the following tasks:
  a. Add the ACL using the "Adding an Extended Access Control List" section on page 10-13.

     Specify the FWSM address as the source address. Specify the remote address(es) for the destination address.
  b. Use the ACL in the **crypto map match address** command according to the "Configuring a Site-to-Site Tunnel" section on page 11-8.
- To identify the traffic that should be tunneled from a VPN client, perform the following tasks:
  a. Add the ACL using the "Adding an Extended Access Control List" section on page 10-13.

     Specify the FWSM address as the source address, and the VPN pool addresses as the destination addresses.
  b. Then use the ACL in the **vpngroup split-tunnel** command according to the "Configuring VPN Client Access" section on page 11-7.

The FWSM only supports IPSec tunnels that terminate on the FWSM and that allow access to the FWSM for management purposes; you cannot terminate a tunnel on the FWSM for traffic that goes through the FWSM to another network.

## Controlling Network Access for Non-IP Traffic (EtherType)

**Transparent firewall mode only**

You can configure an ACL that controls traffic based on its EtherType. The FWSM can control any EtherType identified by a 16-bit hexadecimal number. EtherType ACLs support Ethernet V2 frames. 802.3-formatted frames are not handled by the ACL because they use a length field as opposed to a type field. Bridge protocol data units (BPDUs), which are handled by the ACL, are the only exception: they are SNAP-encapsulated, and the FWSM is designed to specifically handle BPDUs.

To control non-IP traffic, perform the following task:

- Create and apply the ACL according to the "Adding an EtherType Access Control List" section on page 10-16.

## Redistributing OSPF Routes (Standard)

**Single context mode only**

Standard ACLs include only the destination address. You can use a standard ACL with the **route-map** command to control the redistribution of OSPF routes, perform the following tasks:

1. Create the ACL according to the "Adding a Standard Access Control List" section on page 10-17.
2. Create a route map and apply it according to the "Redistributing Routes Between OSPF Processes" section on page 8-6.

# Access Control List Guidelines

See the following guidelines for creating ACLs:

- Access Control Entry Order, page 10-6
- Access Control List Implicit Deny, page 10-6
- Access Control List Commit, page 10-6
- Maximum Number of ACEs, page 10-7
- IP Addresses Used for Access Control Lists When You Use NAT, page 10-7
- Inbound and Outbound Access Control Lists, page 10-10

## Access Control Entry Order

An ACL is made up of one or more Access Control Entries (ACEs). Depending on the ACL 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 ACL name is appended to the end of the ACL.

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 ACL that explicitly permits all traffic, no further statements are ever checked.

## Access Control List Implicit Deny

ACLs 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.

## Access Control List Commit

When you add an ACE to an ACL, the FWSM activates the ACL 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 ACL. This waiting period minimizes the number of times the FWSM commits the ACL. If you enter multiple ACEs within the short waiting period, or paste ACEs at the command prompt, then the FWSM does not commit the ACL until the waiting period has passed and you do not enter more entries. The FWSM displays a message similar to the following after it commits the ACL:

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

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

To manually commit ACLs, see the "Manually Committing Access Control Lists and Rules" section on page 10-24.

For information about exceeding memory limits, see the "Maximum Number of ACEs" section.

## Maximum Number of ACEs

The FWSM supports a maximum of 80K rules for the entire system in single mode, and 142K rules for multiple mode. Rules include ACEs, ACEs used for policy NAT, filters, AAA, ICMP, Telnet, SSH, HTTP, and established rules. See the "Rule Limits" section on page A-5 for the limits for each rule type.

Some ACLs use more memory than others, and these include ACLs 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). Depending on the type of ACL, the actual limit the system can support will be less than 80K (single mode) or 142K (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 ACL, enter the **show access-list** *acl_name* command.

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

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

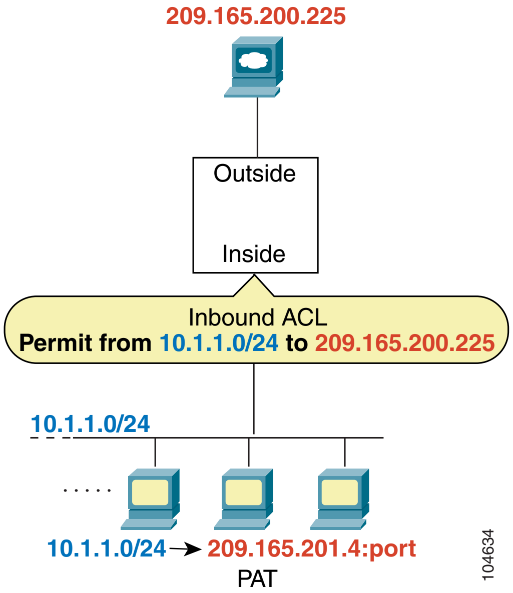
If you exceed the memory limitations, you receive an error message and a system message (106024), and all the ACLs that were added in this compilation are removed from the configuration. Only the set of ACLs that were successfully committed in the previous commitment are used. For example, if you paste 1,000 ACEs at the prompt, and the last ACE exceeds the memory limitations, all 1,000 ACEs are rejected.

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

When you use NAT, the IP addresses you specify for an ACL depend on the interface to which the ACL 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 ACLs: the direction does not determine the address used, only the interface does.

For example, you want to apply an ACL 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 ACL 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 ACL is the real address (see Figure 10-1).

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

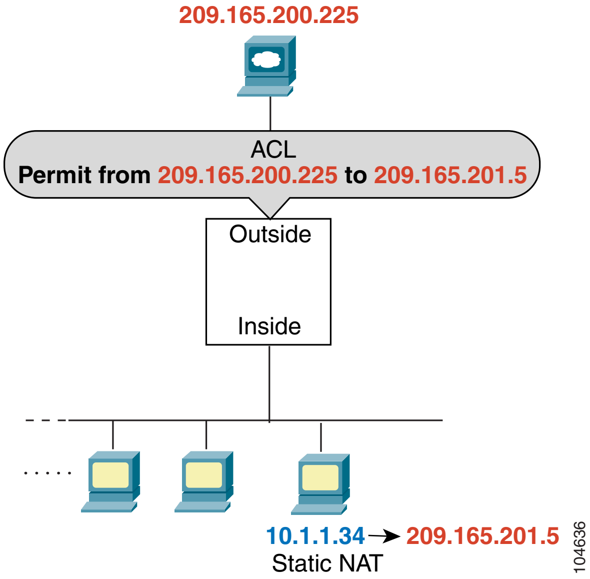

209.165.200.225
Outside
Inside
Inbound ACL
Permit from 10.1.1.0/24 to 209.165.200.225
10.1.1.0/24
.....
10.1.1.0/24→ 209.165.201.4:port
PAT
104634

See the following commands for this example:

```
FWSM/contexta(config)# access-list INSIDE extended permit ip 10.1.1.0 255.255.255.0 host
209.165.200.225
FWSM/contexta(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 ACL on the outside interface. You need to specify the translated address of the inside host in the ACL because that address is the address that can be used on the outside network (see Figure 10-2).

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

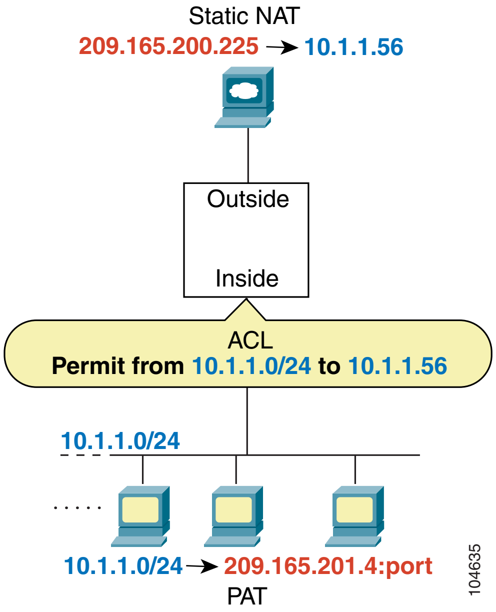


See the following commands for this example:

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

If you perform NAT on both interfaces, then 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 ACLs: NAT used for Source and Destination Addresses*



See the following commands for this example:

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

For an example of IP addresses used in outbound ACLs, see Figure 10-5 on page 10-12.

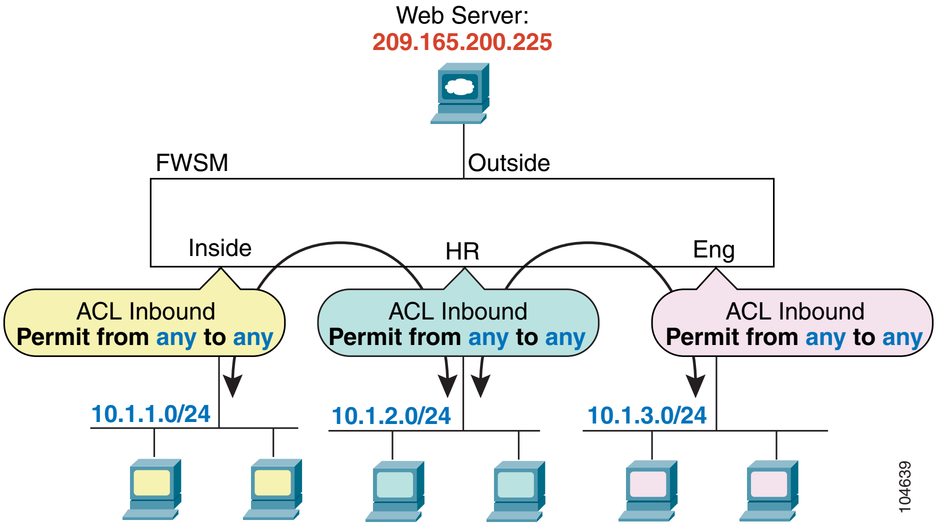
## Inbound and Outbound Access Control Lists

Traffic flowing across an interface in the FWSM can be controlled in two ways. Traffic that enters the FWSM can be controlled by attaching an inbound ACL to the source interface. Traffic that exits the FWSM can be controlled by attaching an outbound ACL to the destination interface. To allow any traffic to enter the FWSM, you must attach an inbound ACL to an interface; otherwise, the FWSM automatically drops all traffic that enters that interface. By default, traffic can exit the FWSM on any interface unless you restrict it using an outbound ACL, which adds restrictions to those already configured in the inbound ACL.

**Note** "Inbound" and "outbound" refer to the application of an ACL on an interface, either to traffic entering the FWSM on an interface or traffic exiting the FWSM on an interface. These terms do not refer to the movement of traffic from a lower security interface to a higher security interface, commonly known as inbound, or from a higher to lower interface, commonly known as outbound.

You might want to use an outbound ACL to simplify your ACL configuration. For example, if you want to allow three inside networks on three different interfaces to access each other, you can create a simple inbound ACL that allows all traffic on each inside interface (see Figure 10-4).

*Figure 10-4 Inbound ACLs*



See the following commands for this example:

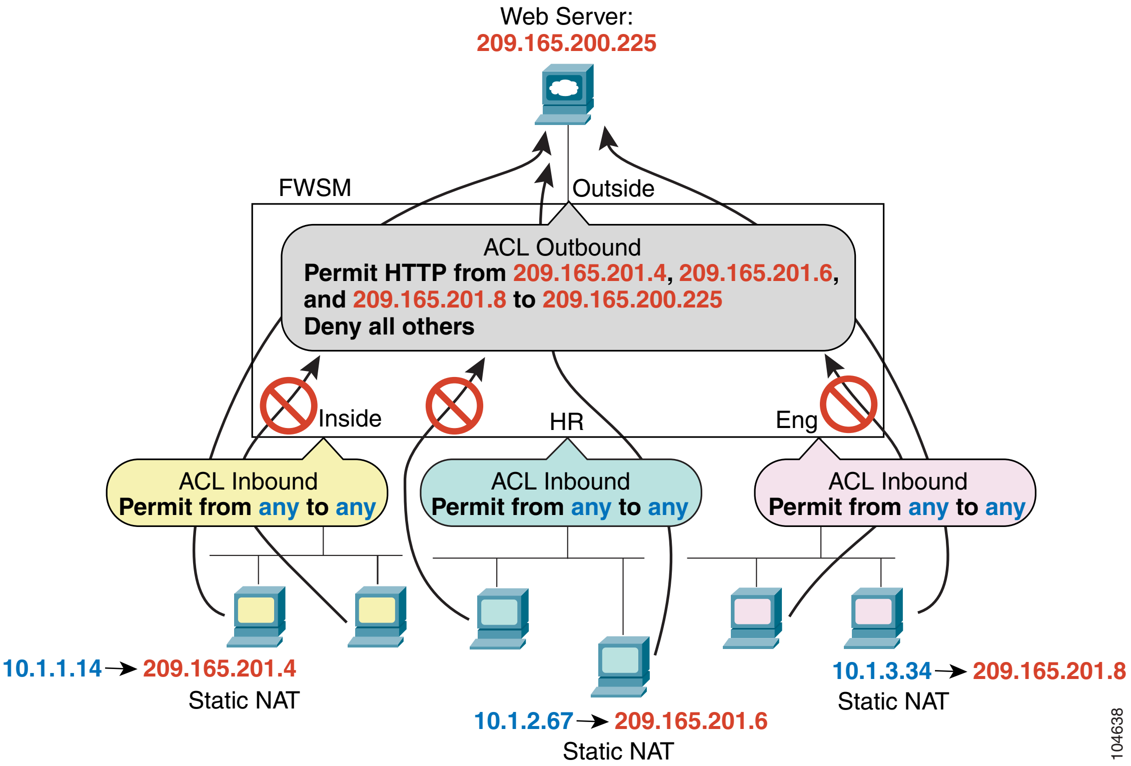
```
FWSM/contexta(config)# access-list INSIDE extended permit ip any any
FWSM/contexta(config)# access-group INSIDE in interface inside

FWSM/contexta(config)# access-list HR extended permit ip any any
FWSM/contexta(config)# access-group HR in interface hr

FWSM/contexta(config)# access-list ENG extended permit ip any any
FWSM/contexta(config)# access-group ENG in interface eng
```

Then, if you want to allow only certain hosts on the inside networks to access a web server on the outside network, you can create a more restrictive ACL that allows only the specified hosts and apply it to the outbound direction of the outside interface (see Figure 10-4). See the "IP Addresses Used for Access Control Lists When You Use NAT" section on page 10-7 for information about NAT and IP addresses. The outbound ACL prevents any other hosts from reaching the outside network.

*Figure 10-5 Outbound ACL*



See the following commands for this example:

```
FWSM/contexta(config)# access-list INSIDE extended permit ip any any
FWSM/contexta(config)# access-group INSIDE in interface inside

FWSM/contexta(config)# access-list HR extended permit ip any any
FWSM/contexta(config)# access-group HR in interface hr

FWSM/contexta(config)# access-list ENG extended permit ip any any
FWSM/contexta(config)# access-group ENG in interface eng

FWSM/contexta(config)# access-list OUTSIDE extended permit tcp host 209.165.201.4
host 209.165.200.225 eq www
FWSM/contexta(config)# access-list OUTSIDE extended permit tcp host 209.165.201.6
host 209.165.200.225 eq www
FWSM/contexta(config)# access-list OUTSIDE extended permit tcp host 209.165.201.8
host 209.165.200.225 eq www
FWSM/contexta(config)# access-group OUTSIDE out interface outside
```

# Adding an Extended Access Control List

An extended ACL is made up of one or more ACEs, in which you can specify the 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 Control Lists with Object Grouping" section on page 10-17.

For TCP and UDP connections, you do not need to also apply an ACL on the destination interface to allow returning traffic, because the FWSM allows all returning traffic for established connections. See the "Stateful Inspection Feature" section on page 1-5 for more information. For connectionless protocols such as ICMP, however, you either need ACLs to allow ICMP in both directions (by applying ACLs to the source and destination interfaces), or you need to enable the ICMP inspection engine. (See the "ICMP Inspection Engine" section on page 13-10.) The ICMP inspection engine treats ICMP sessions as stateful connections. For transparent mode, you can allow protocols with an extended ACL that are otherwise blocked by a routed mode FWSM, including BGP, DHCP, and multicast streams. Because these protocols do not have sessions on the FWSM to allow returning traffic, these protocols also require ACLs on both interfaces.

You can apply only one ACL of each type (extended and EtherType) to each direction of an interface. You can apply the same ACLs on multiple interfaces.

**Note** If you change the ACL configuration, and you do not want to wait for existing connections to time out before the new ACL information is used, you can clear the translation table using the **clear xlate** command. However, clearing the translation table disconnects all current connections.

To add an extended ACL and apply it to an interface, follow these steps:

**Step 1** Add one or more ACEs of the following types using the same ACL name.

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

**Tip** Enter the *acl_name* in upper case letters so the name is easy to see in the configuration. You might want to name the ACL for the interface (for example, INSIDE), or for the purpose (for example, NO_NAT or VPN).

**Note** You specify a network mask in the **access-list** command (for example, 255.255.255.0 for a class C mask). This method is different from the Cisco IOS software **access-list** command, which uses wildcard bits (for example, 0.0.0.255).

- Add an ACE for a specific protocol by entering the following command:

  ```
  FWSM/contexta(config)# access-list acl_name [extended] {deny | permit} protocol
  source_address mask dest_address mask
  ```

  This type of ACE lets you specify any protocol for the source and destination addresses, but not ports. Typically, you identify **ip** for the protocol, but other protocols are accepted.

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

For a list of protocol names, see the "Protocols and Applications" section on page D-5.

For information about logging options that you can add to the end of the ACE, see the "Logging Extended Access Control List Activity" section on page 10-26.

See the following examples:

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

```
FWSM/contexta(config)# access-list ACL_IN extended permit ip any any
```

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

```
FWSM/contexta(config)# access-list ACL_IN extended deny tcp 192.168.1.0 255.255.255.0
209.165.201.0 255.255.255.224
FWSM/contexta(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.

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

- Add an ACE for TCP or UDP ports by entering the following command:

```
FWSM/contexta(config)# access-list acl_name [extended] {deny | permit} {tcp | udp}
source_address mask [operator port] dest_address mask [operator port]
```

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

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
```

For a list of permitted keywords and well-known port assignments, see the "TCP and UDP Ports" section on page D-6. 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.

For information about logging options that you can add to the end of the ACE, see the "Logging Extended Access Control List Activity" section on page 10-26.

See the following example:

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

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

- Add an ACE for ICMP by entering the following command:

  ```
  FWSM/contexta(config)# access-list acl_name [extended] {deny | permit} icmp
  source_address mask dest_address mask [icmp_type]
  ```

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

  Because ICMP is a connectionless protocol, you either need ACLs to allow ICMP in both directions (by applying ACLs to the source and destination interfaces), or you need to enable the ICMP inspection engine (see the "ICMP Inspection Engine" section on page 13-10). 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 "ICMP Types" section on page D-9 for a list of ICMP types.

  For information about logging options that you can add to the end of the ACE, see the "Logging Extended Access Control List Activity" section on page 10-26.

**Step 2** To apply an extended ACL to the inbound or outbound direction of an interface, enter the following command:

```
FWSM/contexta(config)# access-group acl_name {in | out} interface interface_name
```

You can apply one ACL of each type (extended and EtherType) to both directions of the interface. See the "Inbound and Outbound Access Control Lists" section on page 10-10 for more information about ACL directions.

For connectionless protocols, you need to apply the ACL to the source and destination interfaces if you want traffic to pass in both directions. For example, you can allow BGP in an ACL in transparent mode, and you need to apply the ACL to both interfaces.

---

The following example illustrates the commands required to enable access to an inside web server with the IP address 209.165.201.12 (this IP address is the address visible on the outside interface after NAT):

```
FWSM/contexta(config)# access-list ACL_OUT extended permit tcp any host 209.165.201.12 eq
www
FWSM/contexta(config)# access-group ACL_OUT in interface outside
```

You also need to configure NAT for the web server. See the "Using Static NAT" section on page 9-26 for more information.

The following ACLs allow all hosts to communicate between the inside and hr networks, but only specific hosts to access the outside network:

```
FWSM/contexta(config)# access-list ANY extended permit ip any any
FWSM/contexta(config)# access-list OUT extended permit ip host 209.168.200.3 any
FWSM/contexta(config)# access-list OUT extended permit ip host 209.168.200.4 any

FWSM/contexta(config)# access-group ANY in interface inside
FWSM/contexta(config)# access-group ANY in interface hr
FWSM/contexta(config)# access-group OUT out interface outside
```

# Adding an EtherType Access Control List

**Transparent firewall mode only**

An EtherType ACE controls any EtherType identified by a 16-bit hexadecimal number. You can identify some types by a keyword for convenience.

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

For example, you can permit or deny bridge protocol data units (BPDUs). By default, all BPDUs are denied. 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 ACL to avoid bridging loops.

If you allow MPLS, ensure that Label Distribution Protocol (LDP) and Tag Distribution Protocol (TDP) TCP connections are established through the FWSM by configuring both MPLS routers connected to the FWSM to use the IP address on the FWSM interface 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:

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

Or

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

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

To add an EtherType ACL and apply it to an interface, follow these steps:

**Step 1** Add one or more ACEs using the same ACL name by entering the following command:

```
FWSM/contexta(config)# access-list acl_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 ACL name, the ACE is added to the end of the ACL.

**Tip** Enter the *acl_name* in upper case letters so the name is easy to see in the configuration. You might want to name the ACL for the interface (for example, INSIDE), or for the purpose (for example, MPLS or IPX).

**Step 2** To apply an EtherType ACL to the inbound or outbound direction of an interface, enter the following command:

```
FWSM/contexta(config)# access-group acl_name {in | out} interface interface_name
```

You can apply one ACL of each type (extended and EtherType) to both directions of the interface. See the "Inbound and Outbound Access Control Lists" section on page 10-10 for more information about ACL directions.

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

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

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

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

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

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

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

# Adding a Standard Access Control List

**Single context mode only**

Standard ACLs identify the destination IP addresses of OSPF routes, and can be used in a route map for OSPF redistribution. Standard ACLs cannot be applied to interfaces to control traffic.

The following command adds a standard ACE. To add another ACE at the end of the ACL, enter another **access-list** command specifying the same ACL name. Apply the ACL using the "Adding a Route Map" section on page 8-6.

To add an ACE, enter the following command:

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

The following sample ACL identifies routes to 192.168.1.0/24:

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

# Simplifying Access Control Lists with Object Grouping

This section describes how to use object grouping to simplify ACL creation and maintenance, and includes the following topics:

- How Object Grouping Works, page 10-18
- Adding Object Groups, page 10-18
- Nesting Object Groups, page 10-22
- Displaying Object Groups, page 10-24
- Removing Object Groups, page 10-24
- Using Object Groups with an Access Control List, page 10-23

# 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 ACLs. 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 ACL, enter the **show access-list** *acl_name* command.

# Adding Object Groups

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

- Adding a Protocol Object Group, page 10-19
- Adding a Network Object Group, page 10-19
- Adding a Service Object Group, page 10-20
- Adding an ICMP Type Object Group, page 10-21

**Note** If you add new members to an existing object group that is already in use by an ACE in a large ACL, recommitting the ACL can take a long time, depending on the size of the ACL and the object group. In some cases, making this change can cause the FWSM to devote over an hour to committing the ACL, during which time you cannot access the terminal. We recommend that you first remove the ACE that refers to the object group, make your change, and then add the ACE back to the ACL. See the "Manually Committing Access Control Lists and Rules" section on page 10-24 to insert an ACE in an ACL.

## Adding a Protocol Object Group

To add or change a protocol object group, follow these 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, follow these steps:

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

```
FWSM/contexta(config)# object-group protocol grp_id
```

The *grp_id* is a text string up to 64 characters in length.

The prompt changes to the protocol subcommand mode.

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

```
FWSM/contexta(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:

```
FWSM/contexta(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-5.

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

```
FWSM/contexta(config)# object-group protocol tcp_udp_icmp
FWSM/contexta(config-protocol)# protocol-object tcp
FWSM/contexta(config-protocol)# protocol-object udp
FWSM/contexta(config-protocol)# protocol-object icmp
```

## Adding a Network Object Group

To add or change a network object group, follow these 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 network group, follow these steps:

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

```
FWSM/contexta(config)# object-group network grp_id
```

The *grp_id* is a text string up to 64 characters in length.

The prompt changes to the network subcommand mode.

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

```
FWSM/contexta(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:

```
FWSM/contexta(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:

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

## Adding a Service Object Group

To add or change a service object group, follow these 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, follow these steps:

---

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

```
FWSM/contexta(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**. Enter **tcp-udp** if your service uses both TCP and UDP with the same port number, for example, DNS (port 53).

The prompt changes to the service subcommand mode.

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

```
FWSM/contexta(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:

```
FWSM/contexta(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-5.

---

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

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

```
FWSM/contexta(config-service)# object-group service services2 udp
FWSM/contexta(config-service)# description RADIUS Group
FWSM/contexta(config-service)# port-object eq radius
FWSM/contexta(config-service)# port-object eq radius-acct

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

## Adding an ICMP Type Object Group

To add or change an ICMP type object group, follow these 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, follow these steps.:

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

```
FWSM/contexta(config)# object-group icmp-type grp_id
```

The *grp_id* is a text string up to 64 characters in length.

The prompt changes to the ICMP type subcommand mode.

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

```
FWSM/contexta(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:

```
FWSM/contexta(config-icmp-type)# icmp-object icmp_type
```

See the "ICMP Types" section on page D-9 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:

```
FWSM/contexta(config)# object-group icmp-type ping
FWSM/contexta(config-service)# description Ping Group
FWSM/contexta(config-icmp-type)# icmp-object echo
FWSM/contexta(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 10-18. Then follow these steps:

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

```
FWSM/contexta(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:

```
FWSM/contexta(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:

```
FWSM/contexta(config)# object-group network eng
FWSM/contexta(config-network)# network-object host 10.1.1.5
FWSM/contexta(config-network)# network-object host 10.1.1.9
FWSM/contexta(config-network)# network-object host 10.1.1.89

FWSM/contexta(config-network)# object-group network hr
FWSM/contexta(config-network)# network-object host 10.1.2.8
FWSM/contexta(config-network)# network-object host 10.1.2.12

FWSM/contexta(config-network)# object-group network finance
FWSM/contexta(config-network)# network-object host 10.1.4.89
FWSM/contexta(config-network)# network-object host 10.1.4.100
```

You then nest all three groups together as follows:

```
FWSM/contexta(config)# object-group network admin
FWSM/contexta(config-network)# group-object eng
FWSM/contexta(config-network)# group-object hr
FWSM/contexta(config-network)# group-object finance
```

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

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

## Using Object Groups with an Access Control List

To use object groups in an ACL, replace the normal protocol (*protocol*), network (*source_address mask*, etc.), service (*operator port*), or ICMP type (*icmp_type*) parameter with **object-group** *grp_id*.

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

```
FWSM(config)# access-list acl_name [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]
[log [[level] [interval secs] | disable | default]]
```

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 ACL that does not use object groups restricts several hosts on the inside network from accessing several web servers. All other traffic is allowed.

```
FWSM/contexta(config)# access-list ACL_IN extended deny tcp host 10.1.1.4 host
209.165.201.29 eq www
FWSM/contexta(config)# access-list ACL_IN extended deny tcp host 10.1.1.78 host
209.165.201.29 eq www
FWSM/contexta(config)# access-list ACL_IN extended deny tcp host 10.1.1.89 host
209.165.201.29 eq www
FWSM/contexta(config)# access-list ACL_IN extended deny tcp host 10.1.1.4 host
209.165.201.16 eq www
FWSM/contexta(config)# access-list ACL_IN extended deny tcp host 10.1.1.78 host
209.165.201.16 eq www
FWSM/contexta(config)# access-list ACL_IN extended deny tcp host 10.1.1.89 host
209.165.201.16 eq www
FWSM/contexta(config)# access-list ACL_IN extended deny tcp host 10.1.1.4 host
209.165.201.78 eq www
FWSM/contexta(config)# access-list ACL_IN extended deny tcp host 10.1.1.78 host
209.165.201.78 eq www
FWSM/contexta(config)# access-list ACL_IN extended deny tcp host 10.1.1.89 host
209.165.201.78 eq www
FWSM/contexta(config)# access-list ACL_IN extended permit ip any any
FWSM/contexta(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:

```
FWSM/contexta(config)# object-group network denied
FWSM/contexta(config-network)# network-object host 10.1.1.4
FWSM/contexta(config-network)# network-object host 10.1.1.78
FWSM/contexta(config-network)# network-object host 10.1.1.89

FWSM/contexta(config-network)# object-group network web
FWSM/contexta(config-network)# network-object host 209.165.201.29
FWSM/contexta(config-network)# network-object host 209.165.201.16
FWSM/contexta(config-network)# network-object host 209.165.201.78

FWSM/contexta(config-network)# access-list ACL_IN extended deny tcp object-group denied
object-group web eq www
FWSM/contexta(config)# access-list ACL_IN extended permit ip any any
FWSM/contexta(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:

```
FWSM/contexta(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 example shows sample output from the **show object-group** command.

```
FWSM/contexta# 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 ACL.

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

  ```
  FWSM/contexta(config)# no object-group grp_id
  ```

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

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

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

# Manually Committing Access Control Lists and Rules

By default, the FWSM automatically commits ACLs as you enter them; the FWSM waits a short period of time after you last entered an **access-list** command before committing the ACL. See the "Access Control List Commit" section on page 10-6 for more information about committing ACLs.

You might want to manually commit ACLs if you have one of the following situations:

- You are running scripts and want to make sure the ACL was committed in its entirety. With auto-commit, you might commit partial ACLs if you run into memory limitations or other errors in the middle of the ACL entry.
- You want to modify an ACL, such as inserting lines, but do not want to disrupt traffic. For example, with auto-commit, you cannot insert a line into an ACL. You have to create a new ACL (with the inserted line), and then change the ACL name that is assigned to the interface, causing a brief

disruption. With manual commit, you can remove the ACL (from the configuration; not from running), enter a modified ACL with the same name, and then commit the ACL. Because the ACL name is the same, you do not need to change the interface assignment, and there is no disruption of traffic.

- You want to add several ACEs to a large ACL at the command line, and do not want the ACL to commit before you finish making your additions. For example, If you enter a line at the end of a 40,000 line ACL, and you do not enter each additional line within a second of the last line, then the ACL will commit each time you enter a line. A large ACL can take several minutes to commit, and you do not want to wait for the ACL to commit before entering the next line.

If you enable manual commit, then you must remember to manually commit any changes you make to ACLs or other rules, whether the change is an addition or a subtraction. Also, you must manually commit an ACL before you assign it to an interface (**access-group** command); the FWSM cannot assign an ACL to an interface if the ACL does not exist yet.

- To enable manual commit, or to return to auto-commit mode, enter the following command:

  ```
  FWSM/contexta(config)# access-list mode {manual-commit | auto-commit}
  ```

  Auto-commit is the default.

- To commit ACL changes in manual commit mode, enter the following command:

  ```
  FWSM/contexta(config)# access-list commit
  ```

- To view which ACLs are committed and which are uncommitted, enter the following command:

  ```
  FWSM/contexta(config)# show access-list
  ```

# Adding Remarks to Access Control Lists

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

To add a remark after the last **access-list** command you entered, enter the following command:

```
FWSM/contexta(config)# access-list acl_id remark text
```

If you enter the remark before any **access-list** statements, then the remark is the first line in the ACL.

If you delete an ACL using the **no access-list** *acl_id* 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 ACL in this location. Entering a dash (-) at the beginning of the remark helps set it apart from ACEs.

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

# Logging Extended Access Control List Activity

This section describes how to configure ACL logging, and includes the following topics:

- Access Control List Logging Overview, page 10-26
- Configuring Logging for an Access Control Entry, page 10-27
- Managing Deny Flows, page 10-28

## Access Control 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:

```
%FWSM-4-106023: 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 ACL generate logging messages; the implicit deny at the end of the ACL does not generate a message. If you want all denied traffic to generate messages, add the implicit ACE manually to the end of the ACL, as follows:

```
FWSM/contexta(config)# access-list TEST deny ip any any log
```

The **log** options at the end of the extended **access-list** command allow 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:

```
%FWSM-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.

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. See the "Managing Deny Flows" section on page 10-28 to limit the number of logging flows.

Permitted packets that belong to established connections do not need to be checked against ACLs; 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 System Messages Guide* for detailed information about this system message.

# Configuring Logging for an Access Control Entry

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

```
FWSM/contexta(config)# access-list acl_name [extended] {deny | permit}...[log [[level]
[interval secs] | disable | default]]
```

See the "Adding an Extended Access Control List" section on page 10-13 for complete **access-list** syntax.

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 ACL logging.

**default**—Enables logging to message 106023. This setting is the same as having no **log** option.

For example, you configure the following ACL:

```
FWSM/contexta(config)# access-list outside-acl permit ip host 1.1.1.1 any log 7 interval
600
FWSM/contexta(config)# access-list outside-acl permit ip host 2.2.2.2 any
FWSM/contexta(config)# access-list outside-acl deny ip any any log 2
FWSM/contexta(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:

```
%FWSM-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 ACL, 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:

```
%FWSM-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:

```
%FWSM-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:

```
%FWSM-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 32K 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 denial of service (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:

```
%FWSM-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:

  ```
  FWSM/contexta(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:

  ```
  FWSM/contexta(config)# access-list alert-interval secs
  ```

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