Access Rules

This chapter describes how to control network access through or to the ASA using access rules. You use access rules to control network access in both routed and transparent firewall modes. In transparent mode, you can use both access rules (for Layer 3 traffic) and EtherType rules (for Layer 2 traffic).

Note To access the ASA interface for management access, you do not also need an access rule allowing the host IP address. You only need to configure management access according to the general operations configuration guide.

Controlling Network Access

Access rules determine which traffic is allowed through the ASA. There are several different layers of rules that work together to implement your access control policy:

- Extended access rules (Layer 3+ traffic) assigned to interfaces—You can apply separate rule sets (ACLs) in the inbound and outbound directions. An extended access rule permits or denies traffic based on the source and destination traffic criteria.
- Extended access rules assigned globally—You can create a single global rule set, which serves as your default access control. The global rules are applied after interface rules.
- Management access rules (Layer 3+ traffic)—You can apply a single rule set to cover traffic directed at an interface, which would typically be management traffic. In the CLI, these are “control plane” access groups. For ICMP traffic directed at the device, you can alternatively configure ICMP rules.
- EtherType rules (Layer 2 traffic) assigned to interfaces (bridge group member interfaces only)—You can apply separate rule sets in the inbound and outbound directions. EtherType rules control network access for non-IP traffic. An EtherType rule permits or denies traffic based on the EtherType. You can also apply extended access rules to bridge group member interfaces to control Layer 3+ traffic.

General Information About Rules

The following topics provide general information about access rules and EtherType rules.

Interface Access Rules and Global Access Rules

You can apply an access rule to a specific interface, or you can apply an access rule globally to all interfaces. You can configure global access rules in conjunction with interface access rules, in which case, the specific inbound interface access rules are always processed before the general global access rules. Global access rules apply only to inbound traffic.

Inbound and Outbound Rules

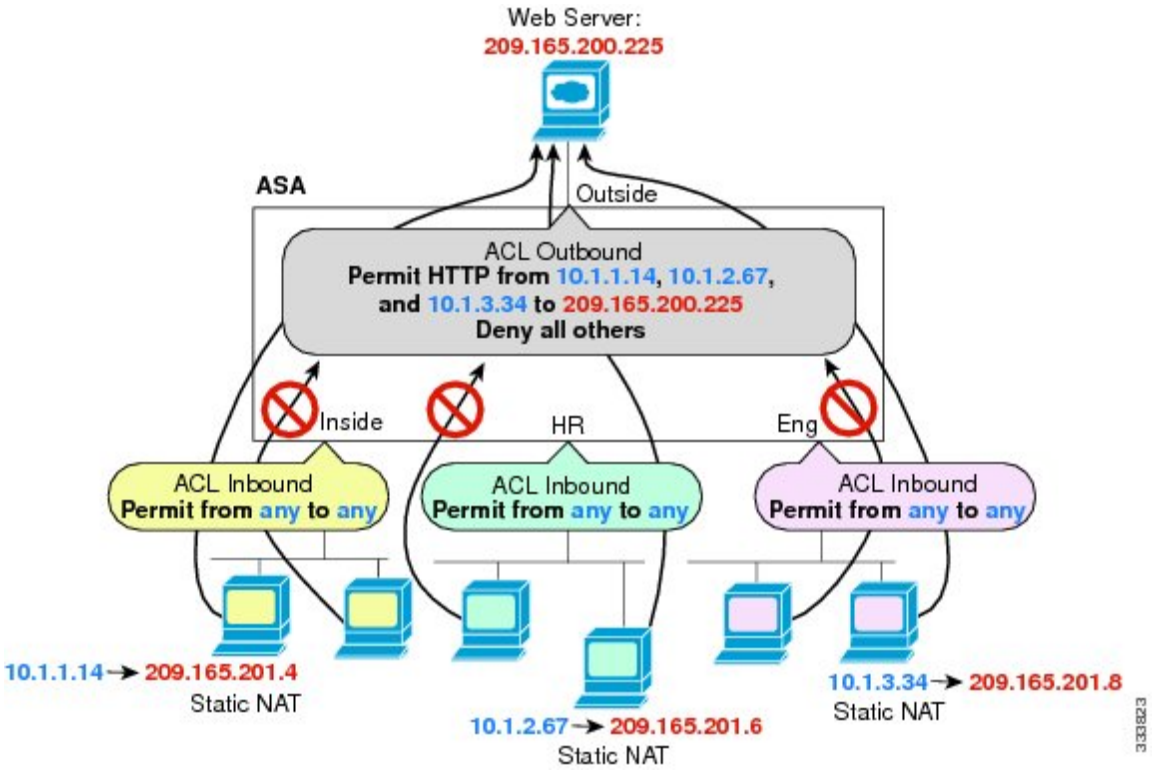
You can configure access rules based on the direction of traffic:

- Inbound—Inbound access rules apply to traffic as it enters an interface. Global and management access rules are always inbound.
- Outbound—Outbound rules apply to traffic as it exits an interface.

Note "Inbound" and "outbound" refer to the application of an ACL on an interface, either to traffic entering the ASA on an interface or traffic exiting the ASA 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.

An outbound ACL is useful, for example, if you want to allow only certain hosts on the inside networks to access a web server on the outside network. Rather than creating multiple inbound ACLs to restrict access, you can create a single outbound ACL that allows only the specified hosts. (See the following figure.) The outbound ACL prevents any other hosts from reaching the outside network.

Figure 1: Outbound ACL



See the following commands for this example:

```
hostname(config)# access-list OUTSIDE extended permit tcp host 10.1.1.14 host 209.165.200.225
 eq www
hostname(config)# access-list OUTSIDE extended permit tcp host 10.1.2.67 host 209.165.200.225
 eq www
hostname(config)# access-list OUTSIDE extended permit tcp host 10.1.3.34 host 209.165.200.225
 eq www
hostname(config)# access-group OUTSIDE out interface outside
```

Rule Order

The order of rules is important. When the ASA decides whether to forward or drop a packet, the ASA tests the packet against each rule in the order in which the rules are listed in the applied ACL. After a match is found, no more rules are checked. For example, if you create an access rule at the beginning that explicitly permits all traffic for an interface, no further rules are ever checked.

Implicit Permits

For routed mode, the following types of traffic are allowed through by default:

- Unicast IPv4 and IPv6 traffic from a higher security interface to a lower security interface.

For transparent mode, the following types of traffic are allowed through by default:

- Unicast IPv4 and IPv6 traffic from a higher security interface to a lower security interface.

- ARPs in both directions. (You can control ARP traffic using ARP inspection, but you cannot control it by access rule.)
- BPDUs in both directions.

For other traffic, you need to use either an extended access rule (IPv4 and IPv6) or an EtherType rule (non-IP).

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 ASA except for particular addresses, then you need to deny the particular addresses and then permit all others.

For management (control plane) ACLs, which control to-the-box traffic, there is no implicit deny at the end of a set of management rules for an interface. Instead, any connection that does not match a management access rule is then evaluated by regular access control rules.

For EtherType ACLs, the implicit deny at the end of the ACL does not affect IP traffic or ARPs; for example, if you allow EtherType 8037, the implicit deny at the end of the ACL does not now block any IP traffic that you previously allowed with an extended ACL (or implicitly allowed from a high security interface to a low security interface). However, if you explicitly deny all traffic with an EtherType rule, then IP and ARP traffic is denied; only physical protocol traffic, such as auto-negotiation, is still allowed.

If you configure a global access rule, then the implicit deny comes *after* the global rule is processed. See the following order of operations:

1. Interface access rule.
2. Global access rule.
3. Implicit deny.

NAT and Access Rules

Access rules always use the real IP addresses when determining an access rule match, even if you configure NAT. For example, if you configure NAT for an inside server, 10.1.1.5, so that it has a publicly routable IP address on the outside, 209.165.201.5, then the access rule to allow the outside traffic to access the inside server needs to reference the server's real IP address (10.1.1.5), and not the mapped address (209.165.201.5).

Same Security Level Interfaces and Access Rules

Each interface has a security level, and security level checking is performed before access rules are considered. Thus, even if you allow a connection in an access rule, it can be blocked due to same-security-level checking at the interface level. You might want to ensure that your configuration allows same-security-level connections so that your access rules are always considered for permit/deny decisions.

- Connections between the same security level ingress and egress interfaces are subject to the same-security-traffic inter-interface check.

  To allow these connections, enter the **same-security-traffic permit inter-interface** command.

  To allow these connections, choose **Configuration** > **Device Setup** > **Interface Settings** > **Interfaces**, then select the **Enable traffic between two or more interfaces which are configured with the same security levels** option.

- Connections with the same ingress and egress interfaces are subject to the same-security-traffic intra-interface check.

  To allow these connections, enter the **same-security-traffic permit intra-interface** command.

  To allow these connections, choose **Configuration** > **Device Setup** > **Interface Settings** > **Interfaces**, then select the **Enable traffic between two or more hosts connected to the same interface** option.

Extended Access Rules

This section describes information about extended access rules.

Extended Access Rules for Returning Traffic

For TCP and UDP connections for both routed and transparent mode, you do not need an access rule to allow returning traffic because the ASA allows all returning traffic for established, bidirectional connections.

For connectionless protocols such as ICMP, however, the ASA establishes unidirectional sessions, so you either need access rules to allow ICMP in both directions (by applying ACLs 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. For example, to control ping, specify **echo-reply** (**0**) (ASA to host) or **echo** (**8**) (host to ASA).

Allowing Broadcast and Multicast Traffic

In routed firewall mode, broadcast and multicast traffic is blocked even if you allow it in an access rule, including unsupported dynamic routing protocols and DHCP. You must configure the dynamic routing protocols or DHCP relay to allow this traffic.

For interfaces that are members of the same bridge group in transparent firewall mode, you can allow any IP traffic through using access rules.

Note Because these special types of traffic are connectionless, you need to apply an access rule to both the inbound and outbound interfaces, so returning traffic is allowed through.

The following table lists common traffic types that you can allow using access rules between interfaces that are members of the same bridge group.

Table 1: Special Traffic for Access Rules between Members of the Same Bridge Group

Traffic Type	Protocol or Port	Notes
DHCP	UDP ports 67 and 68	If you enable the DHCP server, then the ASA does not pass DHCP packets.
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).

Traffic Type	Protocol or Port	Notes
RIP (v1 or v2)	UDP port 520	—

Management Access Rules

You can configure access rules that control management traffic destined to the ASA. Access control rules for to-the-box management traffic (defined by such commands as **http**, **ssh**, or **telnet**) have higher precedence than a management access rule applied with the **control-plane** option. Therefore, such permitted management traffic will be allowed to come in even if explicitly denied by the to-the-box ACL.

Unlike regular access rules, there is no implicit deny at the end of a set of management rules for an interface. Instead, any connection that does not match a management access rule is then evaluated by regular access control rules.

Alternatively, you can use ICMP rules to control ICMP traffic to the device. Use regular extended access rules to control ICMP traffic through the device.

EtherType Rules

This section describes EtherType rules.

Supported EtherTypes and Other Traffic

An EtherType rule controls the following:

- EtherType identified by a 16-bit hexadecimal number, including common types IPX and MPLS unicast or multicast.
- Ethernet V2 frames.
- BPDUs, which are permitted by default. BPDUs are SNAP-encapsulated, and the ASA is designed to specifically handle BPDUs.
- Trunk port (Cisco proprietary) BPDUs. Trunk BPDUs have VLAN information inside the payload, so the ASA modifies the payload with the outgoing VLAN if you allow BPDUs.
- Intermediate System to Intermediate System (IS-IS).
- The IEEE 802.2 Logical Link Control packet. You can control access based on the Destination Service Access Point address.

The following types of traffic are not supported:

- 802.3-formatted frames—These frames are not handled by the rule because they use a length field as opposed to a type field.

EtherType Rules for Returning Traffic

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

Allowing MPLS

If you allow MPLS, ensure that Label Distribution Protocol and Tag Distribution Protocol TCP connections are established through the ASA by configuring both MPLS routers connected to the ASA to use the IP address on the ASA 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 ASA.

mpls ldp router-id *interface* **force**

Or

tag-switching tdp router-id *interface* **force**

Guidelines for Access Control

IPv6 Guidelines

Supports IPv6. The source and destination addresses can include any mix of IPv4 and IPv6 addresses.

Per-User ACL Guidelines

- The per-user ACL uses the value in the **timeout uauth** command, but it can be overridden by the AAA per-user session timeout value.
- If traffic is denied because of a per-user ACL, syslog message 109025 is logged. If traffic is permitted, no syslog message is generated. The **log** option in the per-user ACL has no effect.

Additional Guidelines and Limitations

- Over time, your list of access rules can grow to include many obsolete rules. Eventually, the ACLs for the access groups can become so large that they impact overall system performance. If you find that the system is having issues sending syslog messages, communicating for failover synchronization, establishing and maintaining SSH/HTTPS management access connections, and so forth, you might need to prune your access rules. In general, you should actively maintain your rule lists to remove obsolete rules, rules that are never hit, FQDN objects that can no longer be resolved, and so forth. Also consider implementing object group search.
- You can reduce the memory required to search access rules by enabling object group search, but this is at the expense rule of lookup performance and increased CPU utilization. When enabled, object group search does not expand network or service objects, but instead searches access rules for matches based on those group definitions. You can set this option using the **object-group-search access-control** command.

  For each connection, both the source and destination IP addresses are matched against network objects. If the number of objects matched by the source address times the number matched by the destination address exceeds 10,000, the connection is dropped. This check is to prevent performance degradation. Configure your rules to prevent an excessive number of matches.

Note Object group search works with network and service objects only. It does not work with security group or user objects. Do not enable this feature if the ACLs include security groups. The result can be inactive ACLs or other unexpected behavior.

- You can improve system performance and reliability by using the transactional commit model for access groups. See the basic settings chapter in the general operations configuration guide for more information. Use the **asp rule-engine transactional-commit access-group** command.
- In ASDM, rule descriptions are based on the access list remarks that come before the rule in the ACL; for new rules you create in ASDM, any descriptions are also configured as remarks before the related rule. However, the packet tracer in ASDM matches the remark that is configured after the matching rule in the CLI.
- Normally, you cannot reference an object or object group that does not exist in an ACL or object group, or delete one that is currently referenced. You also cannot reference an ACL that does not exist in an **access-group** command (to apply access rules). However, you can change this default behavior so that you can "forward reference" objects or ACLs before you create them. Until you create the objects or ACLs, any rules or access groups that reference them are ignored. To enable forward referencing, use the **forward-reference enable** command.

Configure Access Control

The following topics explain how to configure access control.

Configure an Access Group

Before you can create an access group, create the ACL.

To bind an ACL to an interface or to apply it globally, use the following command:

access-group *access_list* { {**in** | **out**} **interface** *interface_name* [**per-user-override** | **control-plane**] | **global**}

For an interface-specific access group:

- Specify the extended or EtherType ACL name. You can configure one **access-group** command per ACL type per interface per direction, and one control plane ACL. The control plane ACL must be an extended ACL. Ethertype ACLs are allowed on bridge group member interfaces only.
- The **in** keyword applies the ACL to inbound traffic. The **out** keyword applies the ACL to the outbound traffic.
- Specify the **interface** name.
- The per-user-override keyword (for inbound extended ACLs only) allows dynamic user ACLs that are downloaded for user authorization to override the ACL assigned to the interface. For example, if the interface ACL denies all traffic from 10.0.0.0, but the dynamic ACL permits all traffic from 10.0.0.0, then the dynamic ACL overrides the interface ACL for that user.

By default, VPN remote access traffic is not matched against interface ACLs. However, if you use the **no sysopt connection permit-vpn** command to turn off this bypass, the behavior depends on whether there is a **vpn-filter** applied in the group policy and whether you set the **per-user-override** option:

- No **per-user-override**, no **vpn-filter**—Traffic is matched against the interface ACL.
- No **per-user-override**, **vpn-filter**—Traffic is matched first against the interface ACL, then against the VPN filter.
- **per-user-override**, **vpn-filter**—Traffic is matched against the VPN filter only.

- The **control-plane** keyword specifies if the extended ACL is for to-the-box traffic.

  Unlike regular access rules, there is no implicit deny at the end of a set of management (control plane) rules for an interface. Instead, any connection that does not match a management access rule is then evaluated by regular access control rules.

For a global access group, specify the **global** keyword to apply the extended ACL to the inbound direction of all interfaces.

Example

The following example shows how to use the **access-group** command:

```
hostname(config)# access-list outside_access permit tcp any host 209.165.201.3 eq 80
hostname(config)# access-group outside_access in interface outside
```

The **access-list** command lets any host access the host address using port 80. The **access-group** command specifies that the **access-list** command applies to traffic entering the outside interface.

Configure ICMP Access Rules

By default, you can send ICMP packets to any interface using either IPv4 or IPv6, with these exceptions:

- The ASA does not respond to ICMP echo requests directed to a broadcast address.
- The ASA only responds to ICMP traffic sent to the interface that traffic comes in on; you cannot send ICMP traffic through an interface to a far interface.

To protect the device from attacks, you can use ICMP rules to limit ICMP access to interfaces to particular hosts, networks, or ICMP types. ICMP rules function like access rules, where the rules are ordered, and the first rule that matches a packet defines the action.

If you configure any ICMP rule for an interface, an implicit deny ICMP rule is added to the end of the ICMP rule list, changing the default behavior. Thus, if you want to simply deny a few message types, you must include a permit any rule at the end of the ICMP rule list to allow the remaining message types.

We recommend that you always grant permission for the ICMP unreachable message type (type 3). Denying ICMP unreachable messages disables ICMP path MTU discovery, which can halt IPsec and PPTP traffic. Additionally ICMP packets in IPv6 are used in the IPv6 neighbor discovery process.

Procedure

Step 1 Create rules for ICMP traffic.

icmp {**permit** | **deny**} {**host** *ip_address* | *ip_address mask* | **any**} [*icmp_type*] *interface_name*

If you do not specify an *icmp_type*, the rule applies to all types. You can enter the number or the name. To control ping, specify echo-reply (0) (ASA-to-host) or echo (8) (host-to-ASA).

For the address, you can apply the rule to **any** address, to a single **host**, or to a network (*ip_address mask*).

Step 2 Create rules for ICMPv6 (IPv6) traffic.

ipv6 icmp {**permit** | **deny**} {**host** *ipv6_address* | *ipv6-network*/*prefix-length* | **any**} [*icmp_type*] *interface_name*

If you do not specify an *icmp_type*, the rule applies to all types.

For the address, you can apply the rule to **any** address, to a single **host**, or to a network (*ipv6-network*/*prefix-length*).

Step 3 (Optional.) Set rate limits on ICMP Unreachable messages so that the ASA will appear on trace route output.

icmp unreachable rate-limit *rate* **burst-size** *size*

The rate limit can be 1-100, with 1 being the default. The burst size is meaningless, but must be 1-10.

Example:

Increasing the rate limit, along with enabling the **set connection decrement-ttl** command in a service policy, is required to allow a traceroute through the ASA that shows the ASA as one of the hops. For example, the following policy increases the rate limit and decrements the time-to-live (TTL) value for all traffic through the ASA.

```
icmp unreachable rate-limit 50 burst-size 1
class-map global-class
  match any
policy-map global_policy
  class global-class
  set connection decrement-ttl
```

Examples

The following example shows how to allow all hosts except the one at 10.1.1.15 to use ICMP to the inside interface:

```
hostname(config)# icmp deny host 10.1.1.15 inside
hostname(config)# icmp permit any inside
```

The following example shows how to allow the host at 10.1.1.15 to use only ping to the inside interface:

```
hostname(config)# icmp permit host 10.1.1.15 inside
```

The following example shows how to deny all ping requests and permit all packet-too-big messages (to support path MTU discovery) at the outside interface:

```
hostname(config)# ipv6 icmp deny any echo-reply outside
hostname(config)# ipv6 icmp permit any packet-too-big outside
```

The following example shows how to permit host 2000:0:0:4::2 or hosts on prefix 2001::/64 to ping the outside interface:

```
hostname(config)# ipv6 icmp permit host 2000:0:0:4::2 echo-reply outside
hostname(config)# ipv6 icmp permit 2001::/64 echo-reply outside
hostname(config)# ipv6 icmp permit any packet-too-big outside
```

Monitoring Access Rules

To monitor network access, enter the following commands:

- **clear access-list** *id* **counters**

  Clear the hit counts for the access list.

- **show access-list** [*name*]

  Displays the access lists, including the line number for each ACE and hit counts. Include an ACL name or you will see all access lists.

- **show running-config access-group**

  Displays the current ACL bound to the interfaces.

Evaluating Syslog Messages for Access Rules

Use a syslog event viewer, such as the one in ASDM, to view messages related to access rules.

If you use default logging, you see syslog message 106023 for explicitly denied flows only. Traffic that matches the “implicit deny” entry that ends the rule list is not logged.

If the ASA is attacked, the number of syslog messages for denied packets can be very large. We recommend that you instead enable logging using syslog message 106100, which provides statistics for each rule (including permit rules) and enables you to limit the number of syslog messages produced. Alternatively, you can disable all logging for a given rule.

When you enable logging for message 106100, if a packet matches an ACE, the ASA creates a flow entry to track the number of packets received within a specific interval. The ASA generates a syslog message at the first hit and at the end of each interval, identifying the total number of hits during the interval and the time stamp for the last hit. At the end of each interval, the ASA resets the hit count to 0. If no packets match the ACE during an interval, the ASA deletes the flow entry. When you configure logging for a rule, you can control the interval and even the severity level of the log message, per rule.

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 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 *syslog messages guide* for detailed information about these messages.

Tip When you enable logging for message 106100, if a packet matches an ACE, the ASA creates a flow entry to track the number of packets received within a specific interval. The ASA has a maximum of 32 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 ASA places a limit on the number of concurrent *deny* flows; the limit is placed on deny flows only (not on permit flows) because they can indicate an attack. When the limit is reached, the ASA does not create a new deny flow for logging until the existing flows expire, and issues message 106101. You can control the frequency of this message using the **access-list alert-interval** *secs* command, and the maximum number of deny flows cached using the **access-list deny-flow-max** *number* command.

Configuration Examples for Permitting or Denying Network Access

Following are some typical configuration examples for permitting or denying network access.

Extended ACL Examples

The following example adds a network object for inside server 1, performs static NAT for the server, and enables access from the outside for inside server 1.

```
hostname(config)# object network inside-server1
hostname(config)# host 10.1.1.1
hostname(config)# nat (inside,outside) static 209.165.201.12

hostname(config)# access-list outside_access extended permit tcp any object inside-server1
 eq www
hostname(config)# access-group outside_access in interface outside
```

The following example allows all hosts to communicate between the inside and hr networks but only specific hosts to access the outside network:

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

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

The following example uses object groups to permit specific traffic on the inside interface:

```
!
hostname (config)# object-group service myaclog
```

```
hostname (config-service)# service-object tcp source range 2000 3000
hostname (config-service)# service-object tcp source range 3000 3010 destinatio$
hostname (config-service)# service-object ipsec
hostname (config-service)# service-object udp destination range 1002 1006
hostname (config-service)# service-object icmp echo

hostname(config)# access-list outsideacl extended permit object-group myaclog interface
inside any
```

EtherType Examples

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

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

The following example allows some EtherTypes through the ASA, but it denies all others:

```
hostname(config)# access-list ETHER ethertype permit 0x1234
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 example 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 nonIP in interface inside
hostname(config)# access-group nonIP in interface outside
```

History for Access Rules

Feature Name	Platform Releases	Description
Interface access rules	7.0(1)	Controlling network access through the ASA using ACLs. We introduced the following command: **access-group**.
Global access rules	8.3(1)	Global access rules were introduced. We modified the following command: **access-group.**
Support for Identity Firewall	8.4(2)	You can now use identity firewall users and groups for the source and destination. You can use an identity firewall ACL with access rules, AAA rules, and for VPN authentication. We modified the following commands: **access-list extended**.

Feature Name	Platform Releases	Description
EtherType ACL support for IS-IS traffic	8.4(5), 9.1(2)	In transparent firewall mode, the ASA can now pass IS-IS traffic using an EtherType ACL. We modified the following command: **access-list ethertype** {**permit** | **deny**} **isis**.
Support for TrustSec	9.0(1)	You can now use TrustSec security groups for the source and destination. You can use an identity firewall ACL with access rules. We modified the following commands: **access-list extended**.
Unified ACL for IPv4 and IPv6	9.0(1)	ACLs now support IPv4 and IPv6 addresses. You can even specify a mix of IPv4 and IPv6 addresses for the source and destination. The **any** keyword was changed to represent IPv4 and IPv6 traffic. The **any4** and **any6** keywords were added to represent IPv4-only and IPv6-only traffic, respectively. The IPv6-specific ACLs are deprecated. Existing IPv6 ACLs are migrated to extended ACLs. See the release notes for more information about migration. We modified the following commands: **access-list extended**, **access-list webtype**. We removed the following commands: **ipv6 access-list**, **ipv6 access-list webtype**, **ipv6-vpn-filter**
Extended ACL and object enhancement to filter ICMP traffic by ICMP code	9.0(1)	ICMP traffic can now be permitted/denied based on ICMP code. We introduced or modified the following commands: **access-list extended** , **service-object, service**.
Transactional Commit Model on Access Group Rule Engine	9.1(5)	When enabled, a rule update is applied after the rule compilation is completed; without affecting the rule matching performance. We introduced the following commands: **asp rule-engine transactional-commit**, **show running-config asp rule-engine transactional-commit**, **clear configure asp rule-engine transactional-commit**.
Configuration session for editing ACLs and objects. Forward referencing of objects and ACLs in access rules.	9.3(2)	You can now edit ACLs and objects in an isolated configuration session. You can also forward reference objects and ACLs, that is, configure rules and access groups for objects or ACLs that do not yet exist. We introduced the **clear config-session**, **clear session**, **configure session**, **forward-reference**, and **show config-session** commands.