Chapter 9: Access Control Lists

Routing & Switching
What is an ACL?

- Functions of an access control list:
  - Provide a basic level of security for network access
  - Control which areas a host can access on a network
Packet Filtering

- Packet filtering, sometimes called static packet filtering, controls access to a network by analyzing the incoming and outgoing packets and passing or dropping them based on given criteria, such as the source IP address, destination IP addresses, and the protocol carried within the packet.

- A router acts as a packet filter when it forwards or denies packets according to filtering rules.

- An ACL is a sequential list of permit or deny statements, known as access control entries (ACEs).
The last statement of an ACL is always an implicit deny. This statement is automatically inserted at the end of each ACL even though it is not physically present. The implicit deny blocks all traffic. Because of this implicit deny, an ACL that does not have at least one permit statement will block all traffic.
ACL Logic Operations

- If the packet is accepted, it is then checked against routing table entries to determine the destination interface. If a routing table entry exists for the destination, the packet is then switched to the outgoing interface, otherwise the packet is dropped.

- Next, the router checks whether the outgoing interface has an ACL. If an ACL exists, the packet is tested against the statements in the list.

- If there is no ACL or the packet is permitted, the packet is encapsulated in the new Layer 2 protocol and forwarded out the interface to the next device.
ACL Logic Operations

- When a packet arrives at a router interface, the router process is the same, whether ACLs are used or not. As a frame enters an interface, the router checks to see whether the destination Layer 2 address matches its the interface Layer 2 address or if the frame is a broadcast frame.

- If the frame address is accepted, the frame information is stripped off and the router checks for an ACL on the inbound interface. If an ACL exists, the packet is tested against the statements in the list.
Types of Cisco IPv4 ACLs

- **Standard ACLs**
  - Filter traffic based on source IP addresses only
    
    ```
    access-list 10 permit 192.168.30.0 0.0.0.255
    ```

- **Extended ACLs**
  - Filter traffic based on
    - Source and destination IP addresses
    - Source and destination TCP and UDP ports
    - Protocol type/Protocol number (IP, ICMP, UDP, TCP)
    
    ```
    access-list 103 permit tcp 192.168.30.0 0.0.0.255 any eq 80
    ```
Numbering and Naming ACLs

- Both Standard and Extended ACLs:
  - Include an implicit deny as a final ACE
  - Can be created by using either a descriptive name or number

**Numbered ACL:**

You assign a number based on which protocol you want filtered:
- (1 to 99) and (1300 and 1999): Standard IP ACL
- (100 to 199) and (2000 to 2699): Extended IP ACL

**Named ACL:**

You assign a name by providing the name of the ACL:
- Names can contain alphanumeric characters.
- It is suggested that the name be written in CAPITAL LETTERS.
- Names cannot contain spaces or punctuation.
- You can add or delete entries within the ACL.
Wildcard Mask

- Wildcard masks and subnet masks differ in the way they match binary 1s and 0s. Wildcard masks use the following rules to match binary 1s and 0s:
  - **Wildcard mask bit 0** - Match the corresponding bit value in the address.
  - **Wildcard mask bit 1** - Ignore the corresponding bit value in the address.

- Wildcard masks are often referred to as an *inverse mask*. The reason is that, unlike a subnet mask in which binary 1 is equal to a match and binary 0 is not a match, in a wildcard mask the reverse is true.
Wildcard Mask

- Which IPv4 address range covers all IP addresses that match the ACL filter specified by 152.46.2.0 with wildcard mask 0.0.1.255?

  152.46.2.0 to 152.46.3.255
Wildcard Mask Keywords

- **Any**
  - Allow all IP addresses
  - Wildcard mask of 255.255.255.255 (check no bits)

```
R1(config)#access-list 1 permit 0.0.0.0 255.255.255.255
R1(config)#access-list 1 permit any
```
**Wildcard Mask Keywords**

- **Host**
  - Allow only a single IP address
  - Wildcard mask of 0.0.0.0 (check all bits)

```
R1(config)#access-list 1 permit 192.168.10.10 0.0.0.0
R1(config)#access-list 1 permit host 192.168.10.10
```
General Guidelines and Best Practices

- Use ACLs in firewall routers positioned between your internal network and an external network such as the Internet.
- Use ACLs on a router positioned between two parts of your network to control traffic entering or exiting a specific part of your internal network.
- Configure ACLs on border routers, that is routers situated at the edges of your networks.
- Configure ACLs for each network protocol configured on the border router interfaces.
- Filter unwanted traffic before it travels onto a low-bandwidth link.
- Use a text editor to create, edit and save ACLs.
- Test your ACLs on a development network before implementing them on a production network.
General Guidelines and Best Practices

- **One ACL per protocol** - To control traffic flow on an interface, an ACL must be defined for each protocol enabled on the interface.

- **One ACL per direction** - ACLs control traffic in one direction at a time on an interface. Two separate ACLs must be created to control inbound and outbound traffic.

- **One ACL per interface** - ACLs control traffic for an interface, for example, GigabitEthernet 0/0.
Configuring a Standard ACL

Example ACL

- `access-list 2 deny host 192.168.10.10`
- `access-list 2 permit 192.168.10.0 0.0.0.255`
- `access-list 2 deny 192.168.0.0 0.0.255.255`
- `access-list 2 permit 192.0.0.0 0.255.255.255`
Configuring a Standard ACL

- The full syntax of the standard ACL command is as follows:
  Router(config)# access-list access-list-number
deny permit remark source [ source-wildcard ] [ log ]

- To remove the ACL, the global configuration no access-list command is used.

  The remark keyword is used for documentation and makes access lists a great deal easier to understand.
Configuring a Standard ACL

- A network administrator needs to configure a standard ACL so that only the workstation of the administrator with the IP address 92.68.15.23 can access the virtual terminal of the main router. What are two ways to achieve the task?

  ```
  access-list 10 permit host 92.68.15.23
  access-list 10 permit 92.68.15.23 0.0.0.0
  ```
Internal Logic

- Cisco IOS applies an internal logic when accepting and processing standard access list statements. As discussed previously, access list statements are processed sequentially. Therefore, the order in which statements are entered is important.
- An implicit deny any rejects any packet that does not match any ACE.
- A packet can either be rejected or forwarded as directed by the ACE that is matched.
- Each statement is checked only until a match is detected or until the end of the ACE list.

```
R1(config)#access-list 3 deny 192.168.10.0 0.0.0.255
R1(config)#access-list 3 permit host 192.168.10.10
% Access rule can't be configured at higher sequence num as it is part of the existing rule at sequence num 10
R1(config)#
```
Where to Place ACLs

- Every ACL should be placed where it has the greatest impact on efficiency. The basic rules are:
  - **Standard** ACLs – Locate as close as possible to the destination of the traffic to be filtered.
  - **Extended** ACLs - Locate as close as possible to the source of the traffic to be filtered.

- Placement of the ACL and therefore the type of ACL used may also depend on: the extent of the network administrator’s control, bandwidth of the networks involved, and ease of configuration.
Standard ACL Placement

Block all traffic from 192.168.10.0/24 to 192.168.30.0/24.

Filters traffic from 192.168.10.0/24 to all destinations reachable by R3.

Filters traffic from 192.168.10.0/24 only to 192.168.30.0/24.

192.168.10.0/24
S1
PC1
S2
PC2

192.168.11.0/24

R1

G0/0
192.168.10.0/24
R2

S0/0/0
R3

G0/0
192.168.30.0/24
R4

S0/0/1
S3

192.168.31.0/24

G0/0
S4

192.168.30.0/24
PC3
PC4
Extended ACL Placement

Examines traffic from all networks before exiting R1 S0/0/0.

Examines traffic only from the 192.168.11.0/24.

Block FTP and Telnet traffic from 192.168.11.0/24 to 192.168.30.0/24.
ACL Placement

- If a router has three interfaces and is routing both IPv4 and IPv6 traffic, how many ACLs could be created and applied to it?

  12
Applying Standard ACLs to Interfaces

- After a standard ACL is configured, it is linked to an interface using the `ip access-group` command in interface configuration mode:

  ```
  Router(config-if)# ip access-group { access-list-number | access-list-name } { in | out }
  ```

- To remove an ACL from an interface, first enter the `no ip access-group` command on the interface, and then enter the global `no access-list` command to remove the entire ACL.
Applying Standard ACLs to Interfaces

Deny a Specific Host

```
R1(config)#no access-list 1
R1(config)#access-list 1 deny host 192.168.10.10
R1(config)#access-list 1 permit any
R1(config)#interface g0/0
R1(config-if)#ip access-group 1 in
```
Creating Named Standard ACLs

Router(config)#ip access-list [standard | extended ] name

Alphanumeric name string must be unique and cannot begin with a number.

Router(config-std-nacl)#[permit | deny | remark] {source [source- wildcard]} [log]

Router(config-if)#ip access-group name [in | out]

Activates the named IP ACL on an interface.
Commenting ACLs

Example 1: Commenting a numbered ACL

```
R1(config)#access-list 1 remark Do not allow Guest workstation through
R1(config)#access-list 1 deny host 192.168.10.10
R1(config)#access-list 1 remark Allow devices from all other 192.168.x.x subnets
R1(config)#access-list 1 permit 192.168.0.0 0.0.255.255
R1(config)#interface s0/0/0
R1(config-if)#ip access-group 1 out
R1(config-if)#
```

Example 2: Commenting a named ACL

```
R1(config)#ip access-list standard NO_ACCESS
R1(config-standard-nacl)#remark Do not allow access from Lab workstation
R1(config-standard-nacl)#deny host 192.168.11.10
R1(config-standard-nacl)#remark Allow access from all other networks
R1(config-standard-nacl)#permit any
R1(config-standard-nacl)#interface G0/0
R1(config-if)#ip access-group NO_ACCESS out
R1(config-if)#
```
Editing Standard Numbered ACLs

Editing Numbered ACLs Using a Text Editor

Configuration

R1(config)#access-list 1 deny host 192.168.10.99
R1(config)#access-list 1 permit 192.168.0.0 0.0.255.255

Step 1

R1#show running-config | include access-list 1
access-list 1 deny host 192.168.10.99
access-list 1 permit 192.168.0.0 0.0.255.255

Step 2

<Text editor>
access-list 1 deny host 192.168.10.10
access-list 1 permit 192.168.0.0 0.0.255.255

Step 3

R1(config)#config t
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#no access-list 1
R1(config)#access-list 1 deny host 192.168.10.10
R1(config)#access-list 1 permit 192.168.0.0 0.0.255.255

Step 4

R1#show running-config | include access-list 1
access-list 1 deny host 192.168.10.10
access-list 1 permit 192.168.0.0 0.0.255.255
Editing Standard Numbered ACLs

Editing Numbered ACLs Using Sequence Numbers

**Configuration**

```
R1(config)#access-list 1 deny host 192.168.10.99
R1(config)#access-list 1 permit 192.168.0.0 0.0.255.255
```

**Step 1**

```
R1#show access-lists 1
Standard IP access list 1
  10 deny 192.168.10.99
  20 permit 192.168.0.0, wildcard bits 0.0.255.255
R1#
```

**Step 2**

```
R1#conf t
R1(config)#ip access-list standard 1
R1(config-std-nacl)#no 10
R1(config-std-nacl)#10 deny host 192.168.10.10
R1(config-std-nacl)#end
R1#
```

**Step 3**

```
R1#show access-lists
Standard IP access list 1
  10 deny 192.168.10.10
  20 permit 192.168.0.0, wildcard bits 0.0.255.255
R1#
```
Editing Standard Named ACLs

Adding a Line to a Named ACL

```
R1#show access-lists
Standard IP access list NO_ACCESS
  10 deny 192.168.11.10
  20 permit 192.168.11.0, wildcard bits 0.0.0.255
R1#conf t
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#ip access-list standard NO_ACCESS
R1(config-std-nacl)#15 deny host 192.168.11.11
R1(config-std-nacl)#end
R1#show access-lists
Standard IP access list NO_ACCESS
  10 deny 192.168.11.10
  15 deny 192.168.11.11
  20 permit 192.168.11.0, wildcard bits 0.0.0.255
R1#
```

Note: The `no sequence-number` named-ACL command is used to delete individual statements.
A router has an existing ACL that permits all traffic from the 172.16.0.0 network. The administrator attempts to add a new ACE to the ACL that denies packets from host 172.16.0.1 and receives the error message that is shown in the exhibit. What action can the administrator take to block packets from host 172.16.0.1 while still permitting all other traffic from the 172.16.0.0 network?

Manually add the new deny ACE with a sequence number of 5
Verifying ACLs

R1# show ip interface s0/0/0
Serial0/0/0 is up, line protocol is up
    Internet address is 10.1.1.1/30
<output omitted>
Outgoing access list is 1
    Inbound access list is not set
<output omitted>

R1# show ip interface g0/0
GigabitEthernet0/0 is up, line protocol is up
    Internet address is 192.168.10.1/24
<output omitted>

R1# show access-lists
Standard IP access list 1
    10 deny 192.168.10.10
    20 permit 192.168.0.0, wildcard bits 0.0.255.255
Standard IP access list NO_ACCESS
    15 deny 192.168.11.11
    10 deny 192.168.11.10
    20 permit 192.168.11.0, wildcard bits 0.0.0.255
R1#
ACL Statistics

R1#show access-lists
Standard IP access list 1
  10 deny 192.168.10.10 (4 match(es))
  20 permit 192.168.0.0, wildcard bits 0.0.255.255
Standard IP access list NO_ACCESS
  15 deny 192.168.11.11
  10 deny 192.168.11.10 (4 match(es))
  20 permit 192.168.11.0, wildcard bits 0.0.0.255
R1#

Output after pinging PC3 from PC1.

R1#show access-lists
Standard IP access list 1
  10 deny 192.168.10.10 (8 match(es))
  20 permit 192.168.0.0, wildcard bits 0.0.255.255
Standard IP access list NO_ACCESS
  15 deny 192.168.11.11
  10 deny 192.168.11.10 (4 match(es))
  20 permit 192.168.11.0, wildcard bits 0.0.0.255
R1#
Configuring a Standard ACL to Secure a VTY Port

- Filtering Telnet or SSH traffic is typically considered an extended IP ACL function because it filters a higher level protocol. However, because the `access-class` command is used to filter incoming or outgoing Telnet/SSH sessions by source address, a standard ACL can be used.

  Router(config-line)# access-class access-list-number { in [ vrf-also ] | out }
An administrator has configured an access list on R1 to allow SSH administrative access from host 12.16.1.100. Which command correctly applies the ACL?

```plaintext
R1(config)#int s0/0/0
R1(config-line)# access-class 1 in
```
Verifying a Standard ACL used to Secure a VTY Port

R1#show access-lists
Standard IP access list 21
    10 permit 192.168.10.0, wildcard bits 0.0.0.255 (2 matches)
    20 deny any (1 match)
R1#
Extended ACLs

Using Port Numbers

access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq 23
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq 21
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq 20

Using Keywords

access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq telnet
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq ftp
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq ftp-data
Configuring Extended ACLs

- The procedural steps for configuring extended ACLs are the same as for standard ACLs. The extended ACL is first configured, and then it is activated on an interface. However, the command syntax and parameters are more complex to support the additional features provided by extended ACLs.

```
access-list access-list-number {deny | permit | remark}
protocol source [source-wildcard] [operator operand]
[port port-number or name] destination [destination-wildcard]
[operator operand] [port port-number or name][established]
```
Applying Extended ACLs to Interfaces

```
R1(config)#access-list 103 permit tcp 192.168.10.0 0.0.0.255 any eq 80
R1(config)#access-list 103 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1(config)#access-list 104 permit tcp any 192.168.10.0 0.0.0.255 established
R1(config)#interface g0/0
R1(config-if)#ip access-group 103 in
R1(config-if)#ip access-group 104 out
```
Filtering Traffic with Extended ACLs

Extended ACL to Deny FTP

```
R1(config)#access-list 101 deny tcp 192.168.11.0 0.0.0.255 192.168.10.0 0.0.0.255 eq ftp
R1(config)#access-list 101 deny tcp 192.168.11.0 0.0.0.255 192.168.10.0 0.0.0.255 eq ftp-data
R1(config)#access-list 101 permit ip any any
R1(config)#interface g0/1
R1(config-if)#ip access-group 101 in
```
Creating Named Extended ACLs

R1(config)#ip access-list extended SURFING
R1(config-ext-nacl)#permit tcp 192.168.10.0 0.0.0.255 any eq 80
R1(config-ext-nacl)#permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1(config-ext-nacl)#exit
R1(config)#ip access-list extended BROWSING
R1(config-ext-nacl)#permit tcp any 192.168.10.0 0.0.0.255 established
R1(config-ext-nacl)#exit
R1(config)#interface g0/0
R1(config-if)#ip access-group SURFING in
R1(config-if)#ip access-group BROWSING out
Verifying Extended ACLs

R1#show access-lists
Extended IP access list BROWSING
   10 permit tcp any 192.168.10.0 0.0.0.255 established
Extended IP access list SURFING
   10 permit tcp 192.168.10.0 0.0.0.255 any eq www
   20 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1#
R1#show ip interface g0/0
GigabitEthernet0/0 is up, line protocol is up
   Internet address is 192.168.10.1/24
<output omitted for brevity>
   Outgoing access list is BROWSING
   Inbound access list is SURFING
<output omitted for brevity>
Editing Extended ACLs

- Editing an extended ACL can be accomplished using the same process as editing a standard. An extended ACL can be modified using:
  - Method 1 - Text editor
  - Method 2 – Sequence numbers
Inbound ACL Logic

- Packets are tested against an inbound ACL, if one exists, before being routed.
- If an inbound packet matches an ACL statement with a permit, it is sent to be routed.
- If an inbound packet matches an ACL statement with a deny, it is dropped and not routed.
- If an inbound packet does not meet any ACL statements, then it is “implicitly denied” and dropped without being routed.
Outbound ACL Logic

- Packets are first checked for a route before being sent to an outbound interface. If there is no route, the packets are dropped.

- If an outbound interface has no ACL, then the packets are sent directly to that interface.

- If there is an ACL on the outbound interface, it is tested before being sent to that interface.

- If an outbound packet matches an ACL statement with a permit, it is sent to the interface.

- If an outbound packet matches an ACL statement with a deny, it is dropped.

- If an outbound packet does not meet any ACL statements, then it is “implicitly denied” and dropped.
The network administrator that has the IP address of 10.0.70.23/25 needs to have access to the corporate FTP server (10.0.54.5/28). The FTP server is also a web server that is accessible to all internal employees on networks within the 10.x.x.x address. No other traffic should be allowed to this server. Which extended ACL would be used to filter this traffic, and how would this ACL be applied?

```
access-list 105 permit tcp host 10.0.70.23 host 10.0.54.5 eq 20
access-list 105 permit tcp host 10.0.70.23 host 10.0.54.5 eq 21
access-list 105 permit tcp 10.0.0.0 0.255.255.255 host 10.0.54.5 eq www
access-list 105 deny ip any host 10.0.54.5
access-list 105 permit ip any any
```

R1(config)# interface gi0/0
R1(config-if)# ip access-group 105 out
Configuring ACLs

- Consider the following access list that allows IP phone configuration file transfers from a particular host to a TFTP server:

  R1(config)# access-list 105 permit udp host 10.0.70.23 host 10.0.54.5 range 1024 5000
  R1(config)# access-list 105 deny ip any any
  R1(config)# interface gi0/0
  R1(config-if)# ip access-group 105 out

Which method would allow the network administrator to modify the ACL and include FTP transfers from any source IP address?

  R1(config)# interface gi0/0
  R1(config-if)# no ip access-group 105 out
  R1(config)# no access-list 105
  R1(config)# access-list 105 permit udp host 10.0.70.23 host 10.0.54.5 range 1024 5000
  R1(config)# access-list 105 permit tcp any host 10.0.54.5 eq 20
  R1(config)# access-list 105 permit tcp any host 10.0.54.5 eq 21
  R1(config)# access-list 105 deny ip any any
  R1(config)# interface gi0/0
  R1(config-if)# ip access-group 105 out
The network engineering team on the 10.0.55.0/28 network have a private FTP server. To protect their server, the engineers would like to deny other networks from access to FTP or any other service on this server. Why is the ACL not working?

The ACL is applied to the wrong interface
Comparing IPv4 and IPv6 ACLs

- Although IPv4 and IPv6 ACLs are very similar, there are four significant differences between them:
  - The named extended is the only type of ACL available for IPv6.
  - Applying an IPv6 ACL
    - IPv6 uses the `ipv6 traffic-filter` command to perform the same function for IPv6 interfaces.
  - No Wildcard Masks
    - The prefix-length is used to indicate how much of an IPv6 source or destination address should be matched.
  - Additional Default Statements
    - an implicit permit of neighbor discovery packets
      
      ```
      permit icmp any any nd-na
      permit icmp any any nd-ns
      ```
There are three basic steps to configure an IPv6 ACL:

- From global configuration mode, use the **ipv6 access-list name** command to create an IPv6 ACL.
- From the named ACL configuration mode, use the **permit** or **deny** statements to specify one or more conditions to determine if a packet is forwarded or dropped.
- Return to privileged EXEC mode with the **end** command.

```bash
R1(config)# ipv6 access-list access-list-name
R1(config-ipv6-acl)# deny | permit protocol {source-ipv6-prefix/prefix-length | any | host source-ipv6-address} [operator [port-number]] {destination-ipv6-prefix/ prefix-length | any | host destination-ipv6-address} [operator [port-number]]
```
Applying an IPv6 ACL to an Interface

---

```
R1(config)#interface s0/0/0
R1(config-if)#ipv6 traffic-filter NO-R3-LAN-ACCESS in
```
IPv6 ACL Examples

- Which IPv6 ACL command entry will permit traffic from any host to an SMTP server on network 2001:DB8:10:10::/64?

```
permit tcp any host 2001:DB8:10:10::/64 eq 25
```
The IPv6 access list LIMITED_ACCESS is applied on the S0/0/0 interface of R1 in the inbound direction. Which IPv6 packets from the ISP will be dropped by the ACL on R1?

A. HTTPS packets to PC1
B. ICMPv6 packets that are destined to PC1
C. Packets that are destined to PC1 on port 80
D. Neighbor advertisements that are received from the ISP router
IPv6 ACL Examples

- The IPv6 access list LIMITED_ACCESS is applied on the S0/0/0 interface of R1 in the inbound direction. Which IPv6 packets from the ISP will be dropped by the ACL on R1?

  A. HTTPS packets to PC1
  B. ICMPv6 packets that are destined to PC1
  C. Packets that are destined to PC1 on port 80
  D. Neighbor advertisements that are received from the ISP router
Verifying IPv6 ACLs

```plaintext
R3#show ipv6 interface g0/0
GigabitEthernet0/0 is up, line protocol is up
Global unicast address(es):
  2001:DB8:CAFE:30::1, subnet is 2001:DB8:CAFE:30::/64
Input features: Access List
Inbound access list REstricted-ACCESS

<some output omitted for brevity>
```

```plaintext
R3#show access-lists
IPv6 access list REstricted-ACCESS
  permit tcp any host 2001:DB8:CAFE:10::10 eq www sequence 20
  permit tcp any host 2001:DB8:CAFE:10::10 eq 443 sequence 30
  deny ipv6 any 2001:DB8:CAFE:10::/64 sequence 50
  permit tcp host 2001:DB8:CAFE:30::12 host 2001:DB8:CAFE:11::11 eq
telnet sequence 70
  deny tcp any host 2001:DB8:CAFE:11::11 eq telnet sequence 90
  permit ipv6 any any sequence 110
R3#
```