Chapter 2
Access Control Lists

CCNA Routing and Switching 6.0

Routing and Switching Essentials - Chapter 7

Connecting Networks – Chapter 4

Chapter 2 - Sections & Objectives

- ACL Operation
  - Explain the purpose and operation of ACL in small to medium-sized business networks.
    - Explain how ACL filter traffic.
    - Explain how ACL use wildcard masks.
    - Explain how to create ACL.
    - Explain how to place ACL.
- Standard IPv4 ACL
  - Configure standard IPv4 ACL to filter traffic in a small to medium-sized business network.
    - Configure standard IPv4 ACL to filter traffic to meet networking requirements.
    - Use sequence numbers to edit existing standard IPv4 ACL.
    - Configure a standard ACL to secure VTY access.

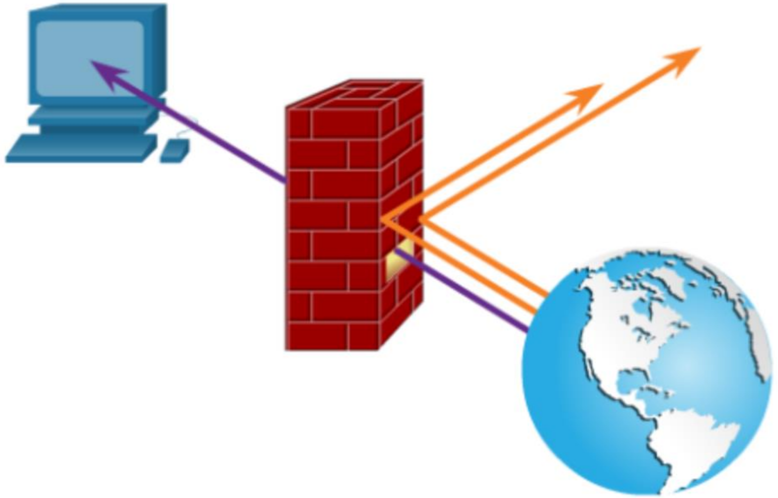
Chapter 7 - Sections & Objectives (Cont.)

- Extended IPv4 ACL
  - Configure extended IPv4 ACL.
    - Explain the structure of an extended access control entry (ACE).
    - Configure extended IPv4 ACL to filter traffic according to networking requirements.
- IPv6 ACL
  - Configure IPv6 ACL.
    - Compare IPv4 and IPv6 ACL creation.
    - Configure IPv6 ACL to filter traffic according to networking requirements.
- Troubleshoot ACL
    - Explain how a router processes packets when an ACL is applied.
    - Troubleshoot common ACL errors using CLI commands.

ACL Operation

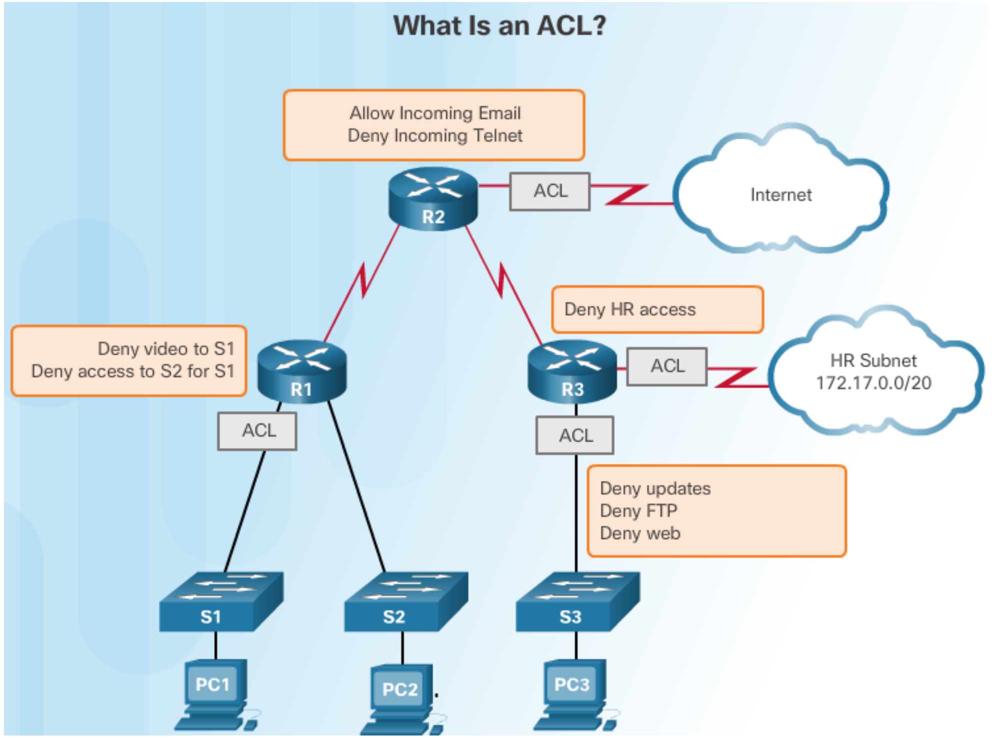
What is an ACL?

- An ACL is a series of IOS commands that control whether a router forwards or drops packets based on information found in the packet header. ACL are not configured by default on a router.



Purpose of ACL

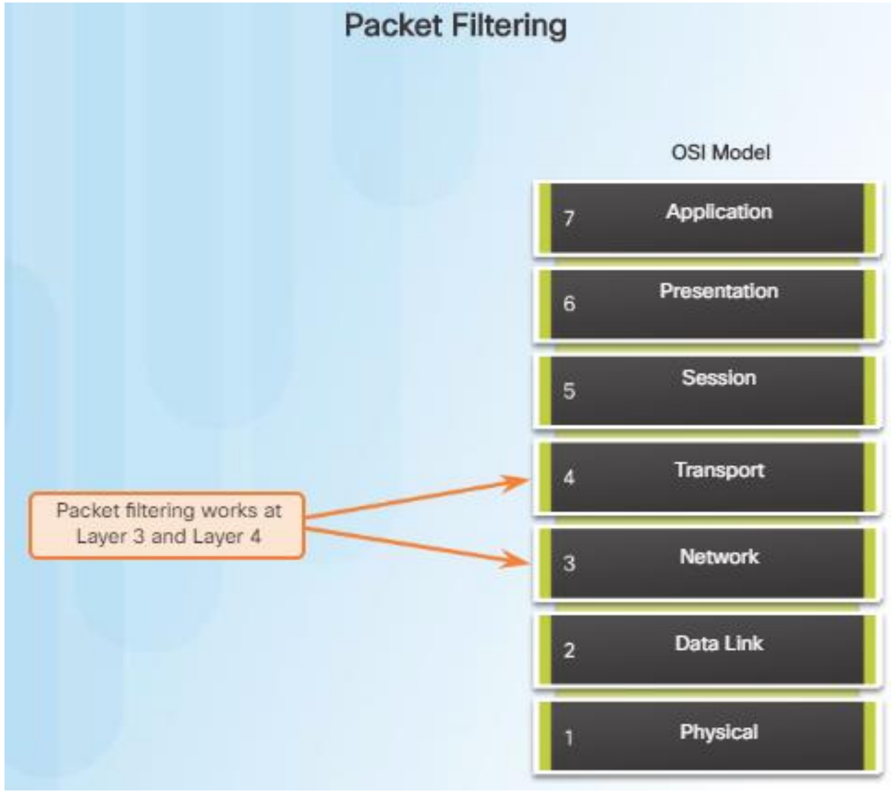
What is an ACL?



- ACL can perform the following tasks:
  - Limit network traffic to increase network performance. For example, video traffic could be blocked if it's not permitted.
  - Provide traffic flow control. ACL can help verify routing updates are from a known source.
  - ACL provide security for network access and can block a host or a network.
  - Filter traffic based on traffic type such as Telnet traffic.
  - Screen hosts to permit or deny access to network services such as FTP or HTTP.

Purpose of ACL

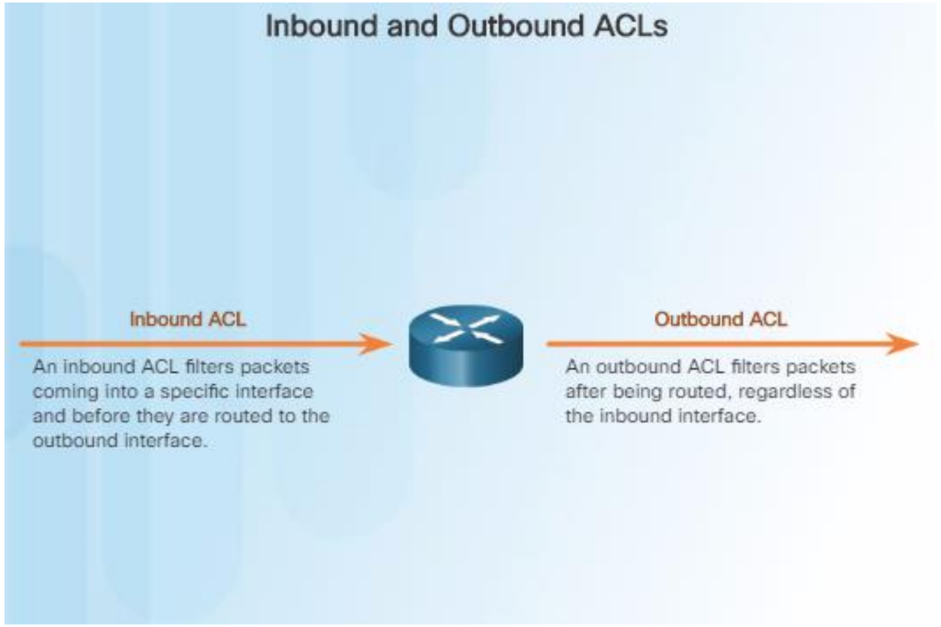
Packet Filtering



- An ACL is a sequential list of permit or deny statements, known as access control entries (ACE). ACE are commonly called ACL statements.
- When network traffic passes through an interface configured with an ACL, the router compares the information within the packet against each ACE, in sequential order, to determine if the packet matches one of the ACE. This is referred to as packet filtering, that:
  - Can analyze incoming and/or outgoing packets.
  - Can occur at Layer 3 or Layer 4.
- The last statement of an ACL is always an implicit deny. This is automatically inserted at the end of each ACL and blocks all traffic. Because of this, all ACL should have at least one permit statement.

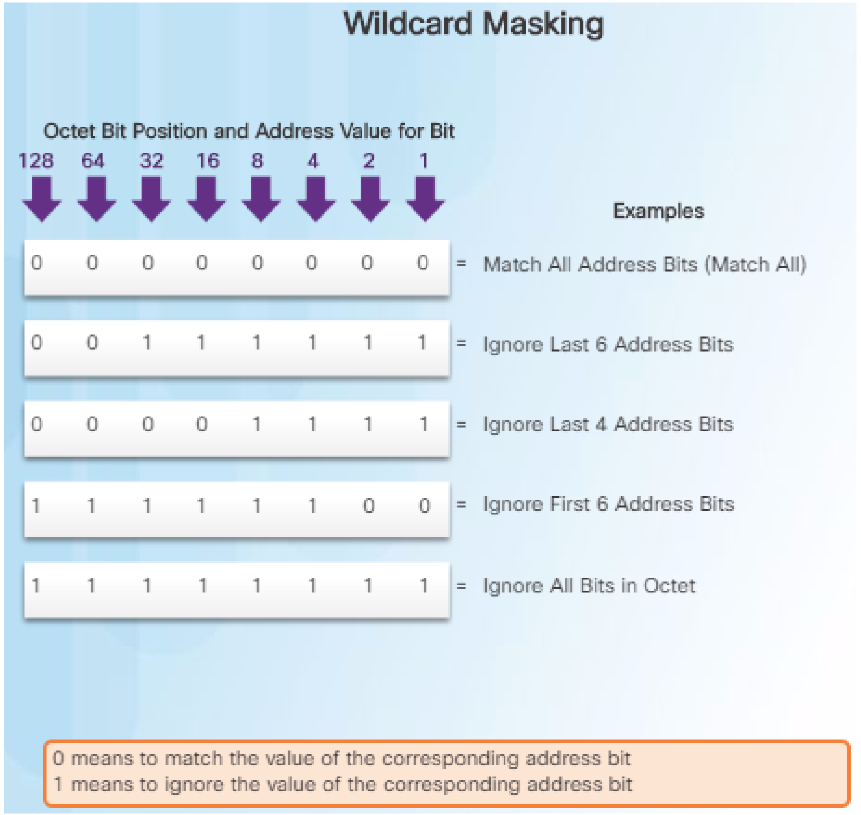
Purpose of ACL

ACL Operation

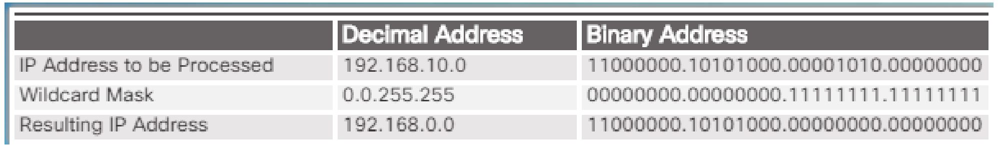

Inbound and Outbound ACLs
Inbound ACL
An inbound ACL filters packets coming into a specific interface and before they are routed to the outbound interface.
Outbound ACL
An outbound ACL filters packets after being routed, regardless of the inbound interface.

- ACL do not act on packets that originate from the router itself.
  - ACL define the set of rules that give added control for packets that enter inbound interfACE, packets that relay through the router, and packets that exit outbound interfACE of the router.
- ACL can be configured to apply to inbound traffic and outbound traffic:
  - Inbound ACL – Incoming packets are processed before they are routed to the outbound interface.
  - Outbound ACL – Incoming packets are routed to the outbound interface, and then they are processed through the outbound ACL.

Introducing ACL Wildcard Masking


Wildcard Masking
Octet Bit Position and Address Value for Bit
128 64 32 16 8 4 2 1
Examples
0 0 0 0 0 0 0 0 = Match All Address Bits (Match All)
0 0 1 1 1 1 1 1 = Ignore Last 6 Address Bits
0 0 0 0 1 1 1 1 = Ignore Last 4 Address Bits
1 1 1 1 1 1 0 0 = Ignore First 6 Address Bits
1 1 1 1 1 1 1 1 = Ignore All Bits in Octet
0 means to match the value of the corresponding address bit
1 means to ignore the value of the corresponding address bit

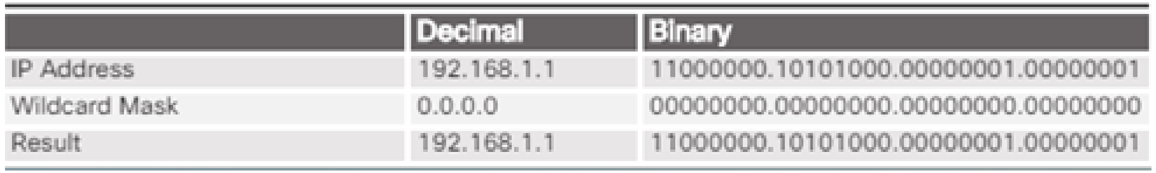
- IPv4 ACE require the use of wildcard masks.
- A wildcard mask is a string of 32 binary digits (1s and 0s) used by the router to determine which bits of the address to examine for a match.
- Wildcard masks are often referred to as an inverse mask since unlike a subnet mask where a binary 1 is a match, a binary 0 is a match with wildcard masks. For example:



	Decimal Address	Binary Address
IP Address to be Processed	192.168.10.0	11000000.10101000.00001010.00000000
Wildcard Mask	0.0.255.255	00000000.00000000.11111111.11111111
Resulting IP Address	192.168.0.0	11000000.10101000.00000000.00000000

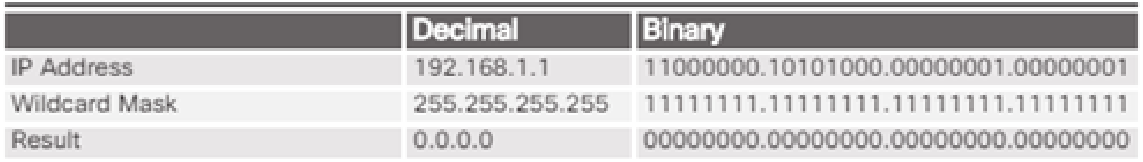
Wildcard Mask Examples

- Calculating the wildcard mask to match IPV4 subnets takes practice.



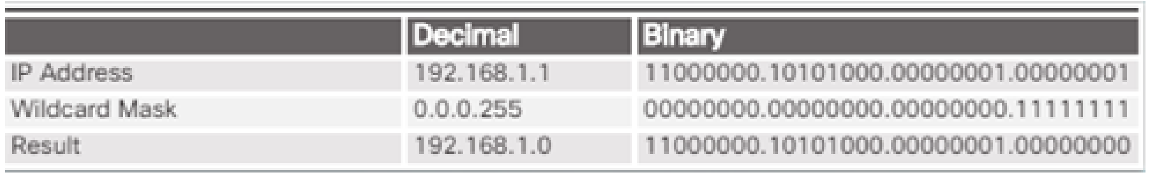
	Decimal	Binary
IP Address	192.168.1.1	11000000.10101000.00000001.00000001
Wildcard Mask	0.0.0.0	00000000.00000000.00000000.00000000
Result	192.168.1.1	11000000.10101000.00000001.00000001

- Example 1: The wildcard mask stipulates that every bit in the IPv4 192.168.1.1 address must match exactly.



	Decimal	Binary
IP Address	192.168.1.1	11000000.10101000.00000001.00000001
Wildcard Mask	255.255.255.255	11111111.11111111.11111111.11111111
Result	0.0.0.0	00000000.00000000.00000000.00000000

- Example 2: The wildcard mask stipulates that anything will match.



	Decimal	Binary
IP Address	192.168.1.1	11000000.10101000.00000001.00000001
Wildcard Mask	0.0.0.255	00000000.00000000.00000000.11111111
Result	192.168.1.0	11000000.10101000.00000001.00000000

- Example 3: The wildcard mask stipulates that any host within the 192.168.1.0/24 network will match.

Calculating the Wildcard Mask

Example 1

```
  255.255.255.255
- 255.255.255.000
-----------------
              255
```

Example 2

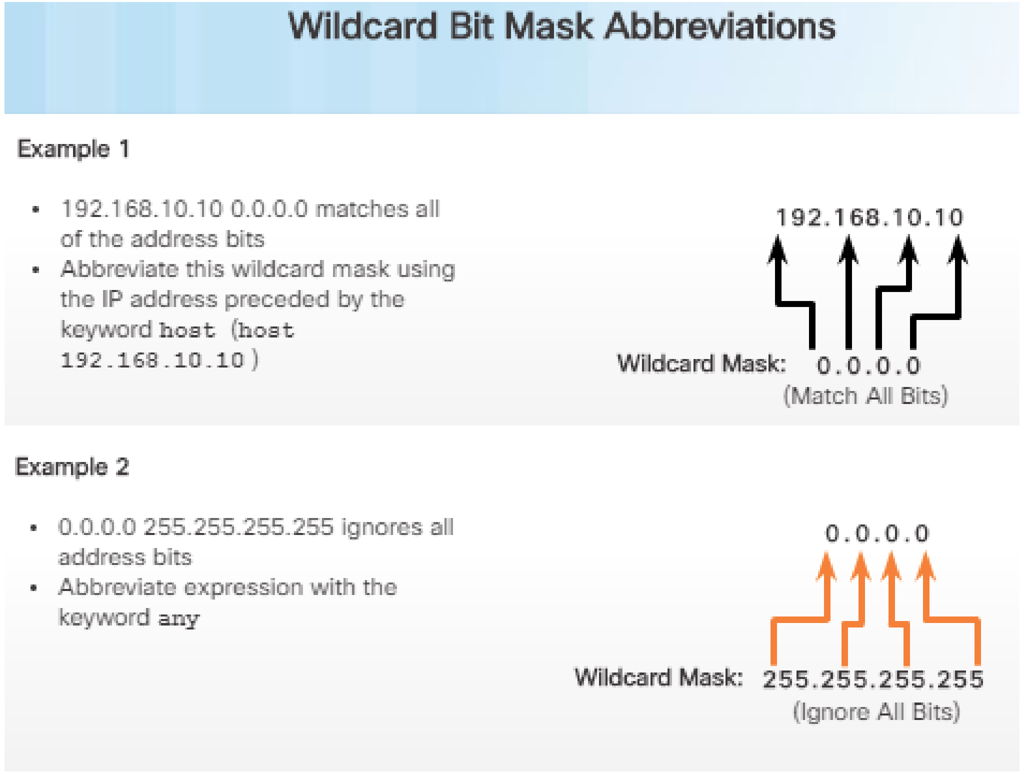
```
  255.255.255.255
- 255.255.255.240
-----------------
               15
```

Example 3

```
  255.255.255.255
- 255.255.254.000
-----------------
            1.255
```

- Example 1: Assume you want to permit access to all users in the 192.168.3.0 network with the subnet mask of 255.255.255.0. Subtract the subnet from 255.255.255.255 and the result is: 0.0.0.255.
- Example 2: Assume you want to permit network access for the 14 users in the subnet 192.168.3.32/28 with the subnet mask of 255.255.255.240. After subtracting the subnet mask from 255.255.255.255, the result is 0.0.0.15.
- Example 3: Assume you want to match only networks 192.168.10.0 and 192.168.11.0 with the subnet mask of 255.255.254.0. After subtracting the subnet mask from 255.255.255.255, the result is 0.0.1.255.

Wildcard Mask Keywords



Wildcard Bit Mask Abbreviations

Example 1

- 192.168.10.10 0.0.0.0 matches all of the address bits
- Abbreviate this wildcard mask using the IP address preceded by the keyword `host` (`host 192.168.10.10`)

Example 2

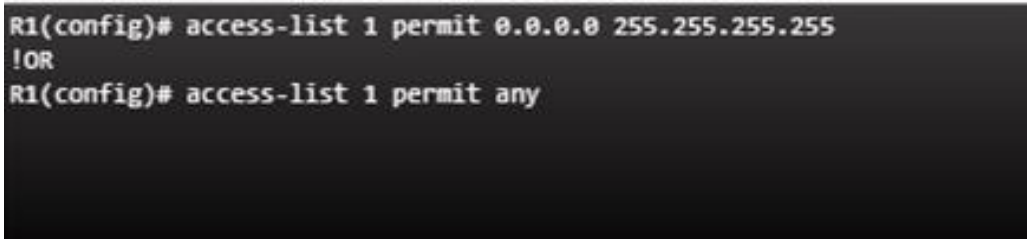
- 0.0.0.0 255.255.255.255 ignores all address bits
- Abbreviate expression with the keyword `any`

- To make wildcard masks easier to read, the keywords **host** and **any** can help identify the most common uses of wildcard masking.
  - **host** substitutes for the 0.0.0.0 mask
  - **any** substitutes for the 255.255.255.255 mask
- If you would like to match the 192.169.10.10 address, you could use **192.168.10.10 0.0.0.0** or, you can use: **host 192.168.10.10**
- In Example 2, instead of entering **0.0.0.0 255.255.255.255**, you can use the keyword **any** by itself.

Wildcard Masks in ACL

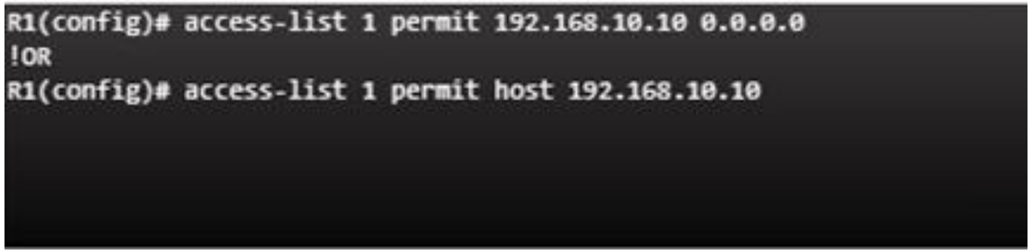
Wildcard Mask Keyword Examples

- Example 1 in the figure demonstrates how to use the **any** keyword to substitute the IPv4 address 0.0.0.0 with a wildcard mask of 255.255.255.255.



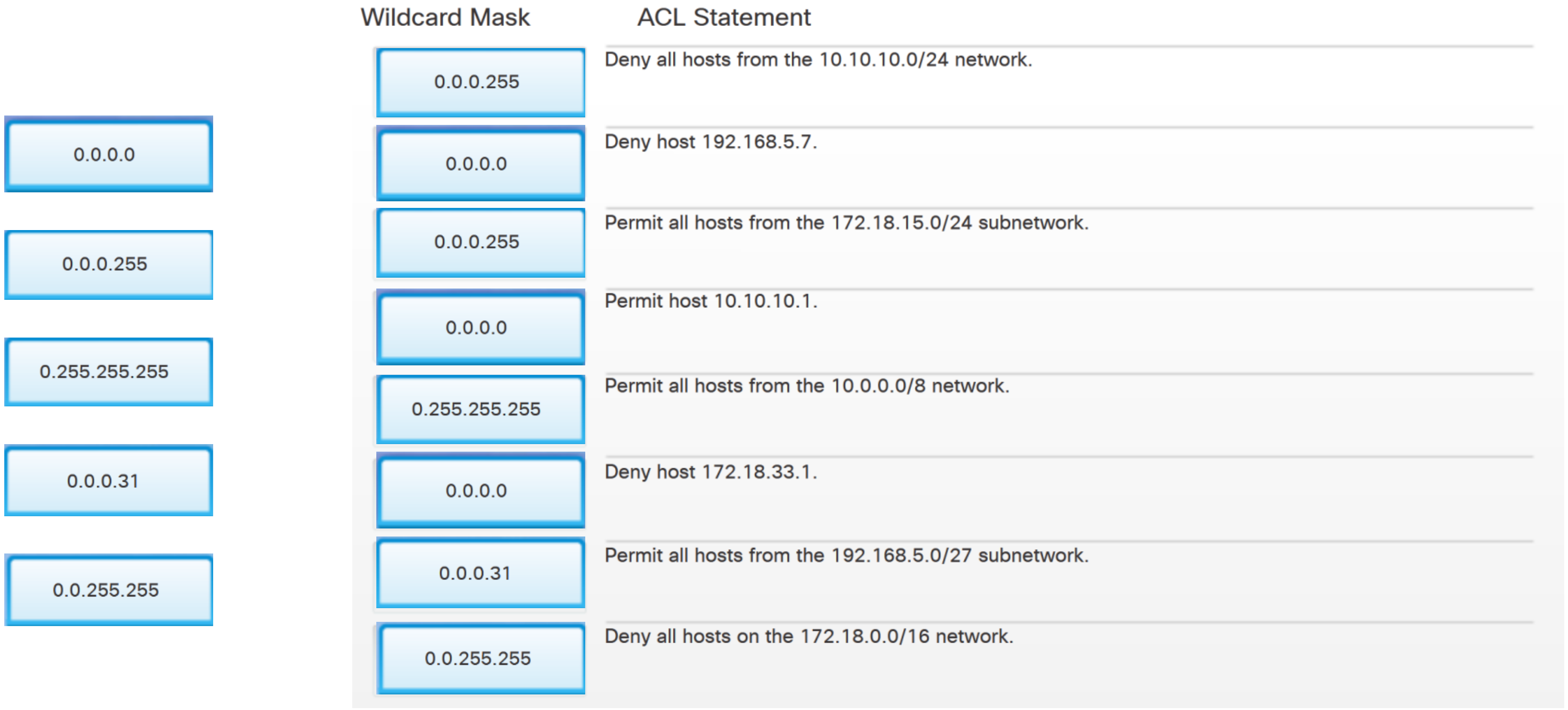
```
R1(config)# access-list 1 permit 0.0.0.0 255.255.255.255
!OR
R1(config)# access-list 1 permit any
```

- Example 2 demonstrates how to use the **host** keyword to substitute for the wildcard mask when identifying a single host.



```
R1(config)# access-list 1 permit 192.168.10.10 0.0.0.0
!OR
R1(config)# access-list 1 permit host 192.168.10.10
```

Activity – Determine the correct wildcard mask



0.0.0.0

0.0.0.255

0.255.255.255

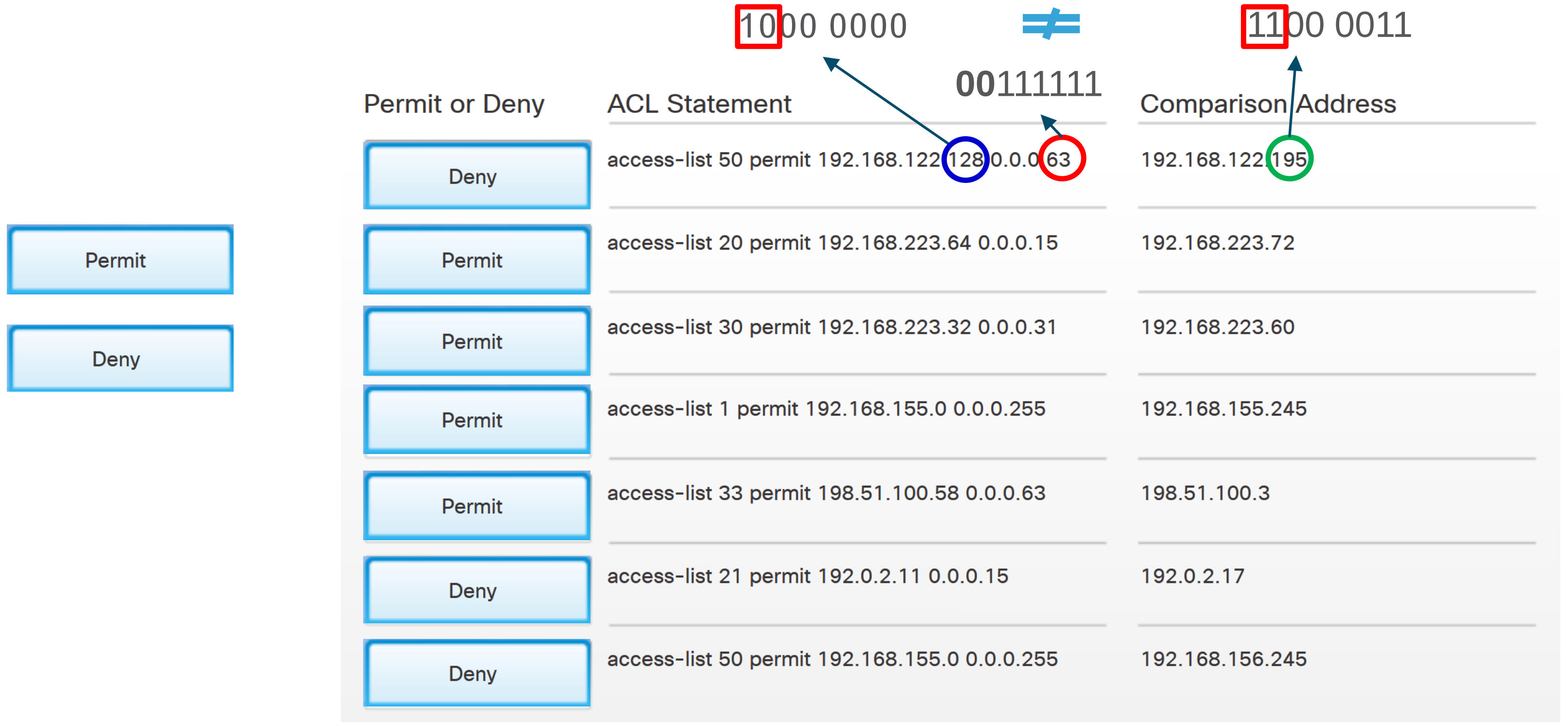
0.0.0.31

0.0.255.255

Wildcard Mask	ACL Statement
0.0.0.255	Deny all hosts from the 10.10.10.0/24 network.
0.0.0.0	Deny host 192.168.5.7.
0.0.0.255	Permit all hosts from the 172.18.15.0/24 subnetwork.
0.0.0.0	Permit host 10.10.10.1.
0.255.255.255	Permit all hosts from the 10.0.0.0/8 network.
0.0.0.0	Deny host 172.18.33.1.
0.0.0.31	Permit all hosts from the 192.168.5.0/27 subnetwork.
0.0.255.255	Deny all hosts on the 172.18.0.0/16 network.

Activity – Determine the Permit or Deny

128-64-32-16-8-4-2-1



Permit or Deny	ACL Statement	Comparison Address
Deny	access-list 50 permit 192.168.122.128 0.0.0.63	192.168.122.195
Permit	access-list 20 permit 192.168.223.64 0.0.0.15	192.168.223.72
Permit	access-list 30 permit 192.168.223.32 0.0.0.31	192.168.223.60
Permit	access-list 1 permit 192.168.155.0 0.0.0.255	192.168.155.245
Permit	access-list 33 permit 198.51.100.58 0.0.0.63	198.51.100.3
Deny	access-list 21 permit 192.0.2.11 0.0.0.15	192.0.2.17
Deny	access-list 50 permit 192.168.155.0 0.0.0.255	192.168.156.245

Activity – Determine the Permit or Deny

128-64-32-16-8-4-2-1

Permit

Deny

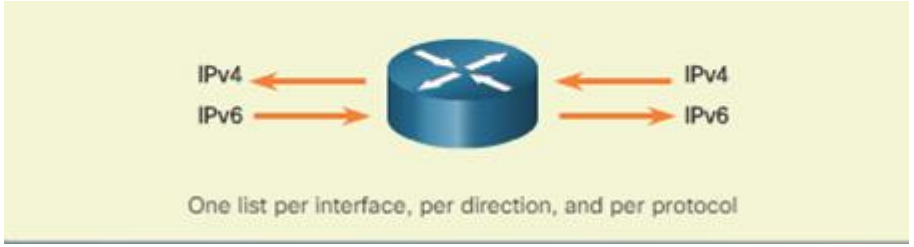
Permit or Deny	ACL Statement	Comparison Address
Deny	access-list 66 permit 172.16.0.0 0.0.255.255	172.17.0.5
Permit	access-list 65 permit 172.16.1.1 0.0.0.0	172.16.1.1
Deny	access-list 55 permit 192.168.15.0 0.0.0.3	192.168.15.5
Permit	access-list 16 permit 201.201.100.0 0.0.0.255	201.201.100.33
Permit	access-list 18 permit 10.10.10.0 0.0.0.63	10.10.10.50
Permit	access-list 60 permit 10.10.0.0 0.0.255.255	10.10.33.33
Deny	access-list 25 permit 172.18.5.0 0.0.0.255	172.18.6.20

General Guidelines for Creating ACL

The Rules for Applying ACLs

You can only have one ACL per protocol, per interface, and per direction:

- One ACL per protocol (e.g., IPv4 or IPv6)
- One ACL per direction (i.e., IN or OUT)
- One ACL per interface (e.g., GigabitEthernet0/0)



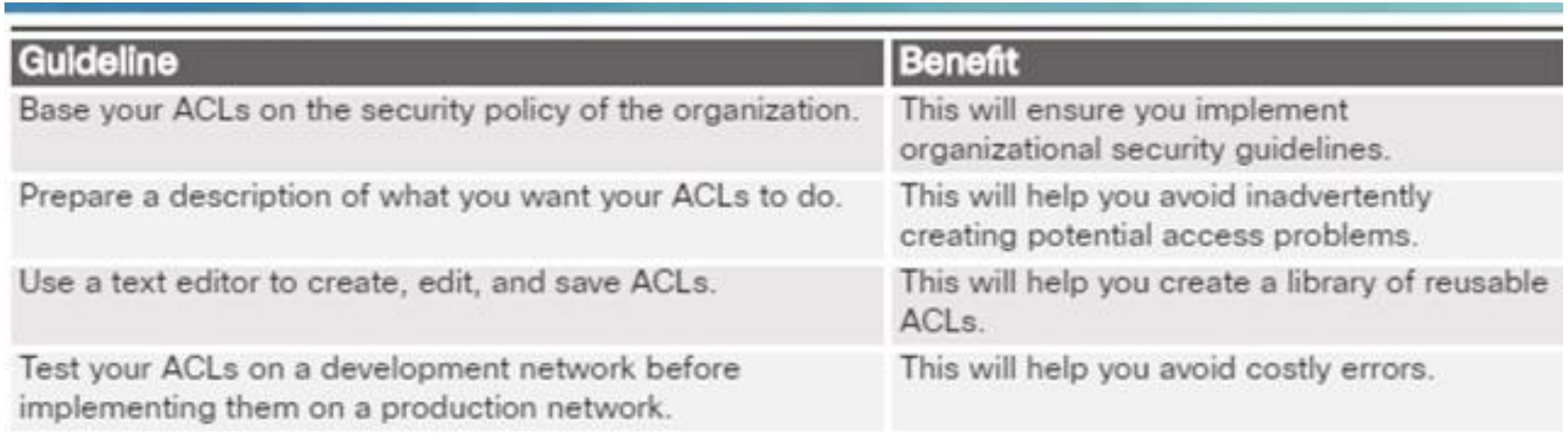
One list per interface, per direction, and per protocol

With 2 interfaces and 2 protocols running, this router could have a total of 8 separated ACL applied.

- Use ACL in firewall routers positioned between your internal network and an external network such as the Internet.
- Use ACL on a router positioned between two parts of your network to control traffic entering or exiting a specific part of your internal network.
- Configure ACL on border routers such as those situated at the edge of your network. This will provide a basic buffer from the outside network that is less controlled.
- Configure ACL for each network protocol configured on the border router interface.

ACL Best Practices

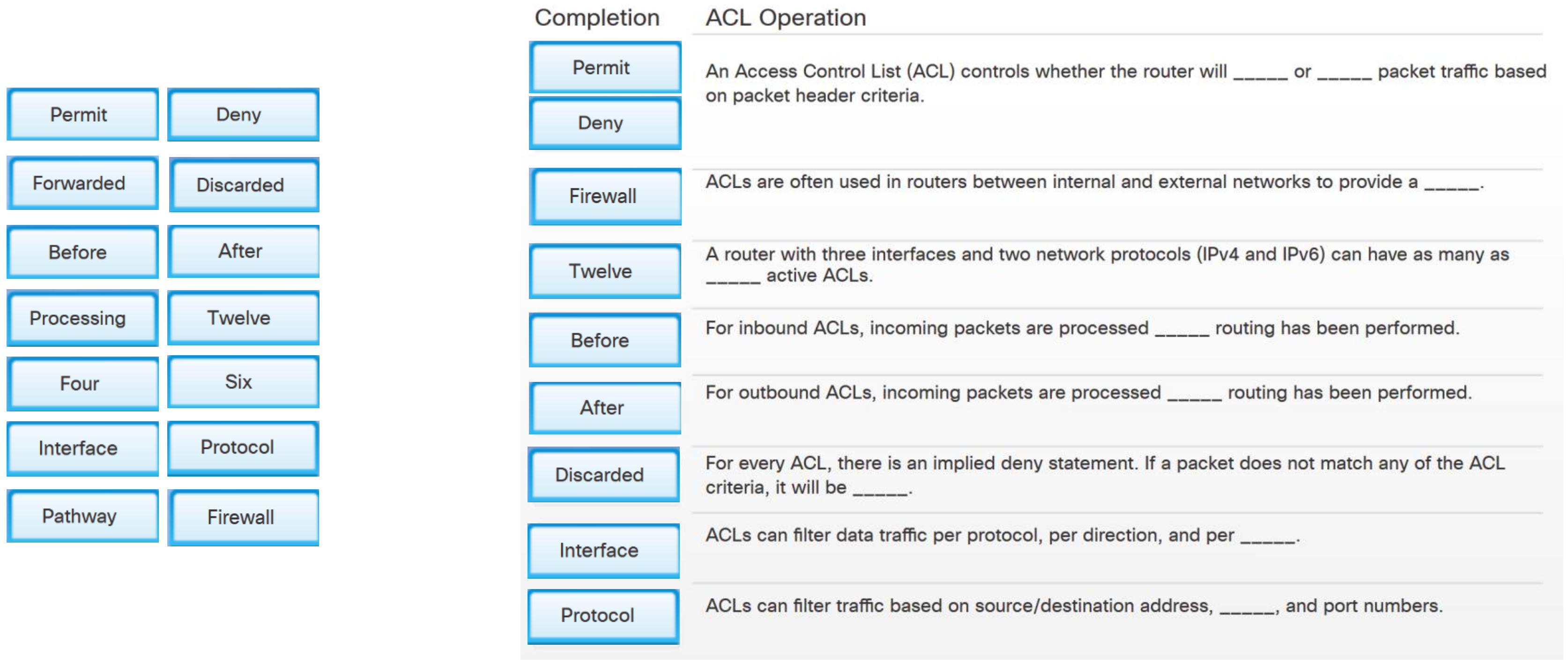
- Using ACL requires significant attention to detail. Mistakes can be very costly in terms of downtime, troubleshooting efforts, and poor network performance.



Guideline	Benefit
Base your ACLs on the security policy of the organization.	This will ensure you implement organizational security guidelines.
Prepare a description of what you want your ACLs to do.	This will help you avoid inadvertently creating potential access problems.
Use a text editor to create, edit, and save ACLs.	This will help you create a library of reusable ACLs.
Test your ACLs on a development network before implementing them on a production network.	This will help you avoid costly errors.

Guidelines for ACL Creation

Activity – ACL Operation



Permit	Deny
Forwarded	Discarded
Before	After
Processing	Twelve
Four	Six
Interface	Protocol
Pathway	Firewall

Completion	ACL Operation
Permit Deny	An Access Control List (ACL) controls whether the router will _____ or _____ packet traffic based on packet header criteria.
Firewall	ACLs are often used in routers between internal and external networks to provide a _____.
Twelve	A router with three interfaces and two network protocols (IPv4 and IPv6) can have as many as _____ active ACLs.
Before	For inbound ACLs, incoming packets are processed _____ routing has been performed.
After	For outbound ACLs, incoming packets are processed _____ routing has been performed.
Discarded	For every ACL, there is an implied deny statement. If a packet does not match any of the ACL criteria, it will be _____.
Interface	ACLs can filter data traffic per protocol, per direction, and per _____.
Protocol	ACLs can filter traffic based on source/destination address, _____, and port numbers.

Types of IPv4 ACL

Types of IPv4 ACL

Standard and Extended IPv4 ACL

Standard ACLs filter IP packets based on the source address only.

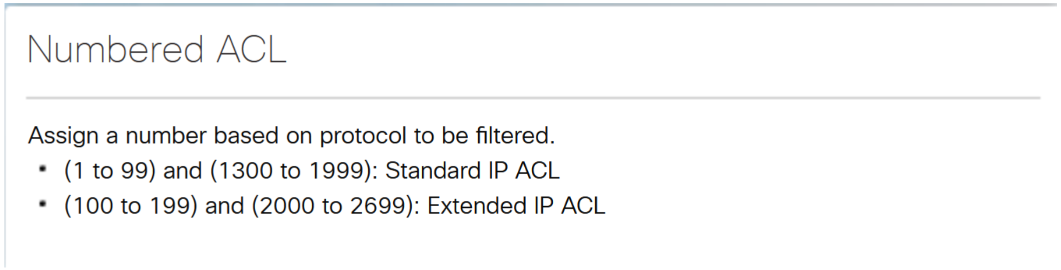
```
access-list 10 permit 192.168.30.0 0.0.0.255
```

Extended ACLs filter IP packets based on several attributes, including the following:

- Source and destination IP addresses
- Source and destination TCP and UDP ports
- Protocol type/Protocol number (example: IP, ICMP, UDP, TCP, etc.)

```
access-list 103 permit tcp 192.168.30.0 0.0.0.255 any eq 80
```

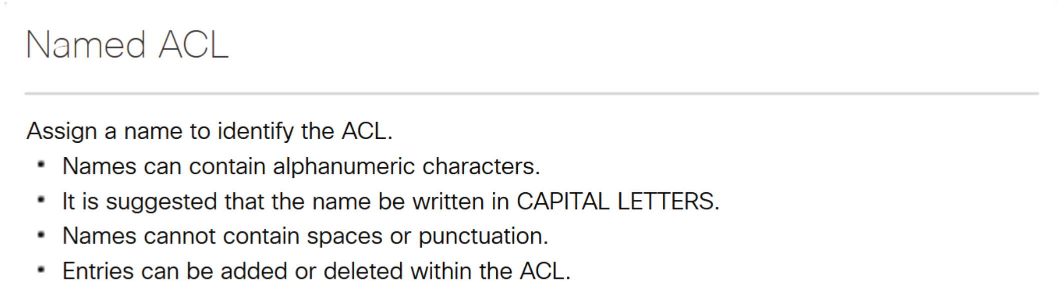
Numbered and Named IPv4 ACL



Numbered ACL

Assign a number based on protocol to be filtered.

- (1 to 99) and (1300 to 1999): Standard IP ACL
- (100 to 199) and (2000 to 2699): Extended IP ACL



Named ACL

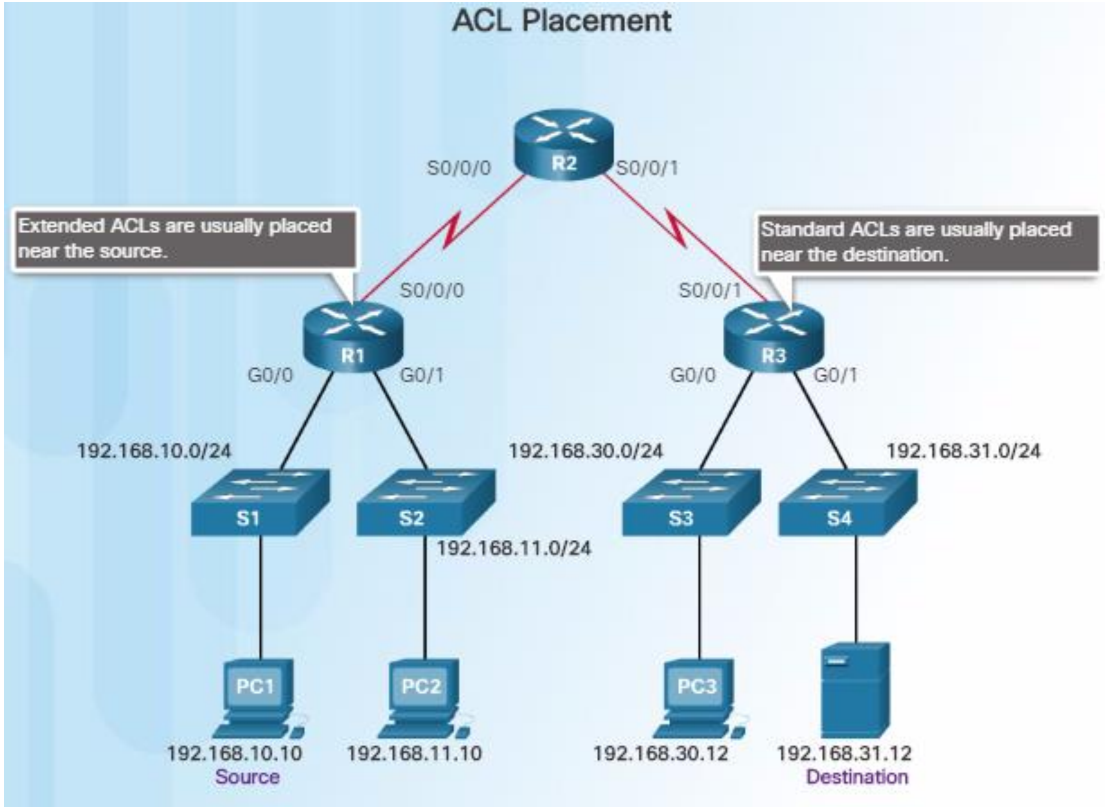
Assign a name to identify the ACL.

- Names can contain alphanumeric characters.
- It is suggested that the name be written in CAPITAL LETTERS.
- Names cannot contain spaces or punctuation.
- Entries can be added or deleted within the ACL.

- Standard and extended ACL can be created using either a number or a name to identify the ACL and its list of statements.

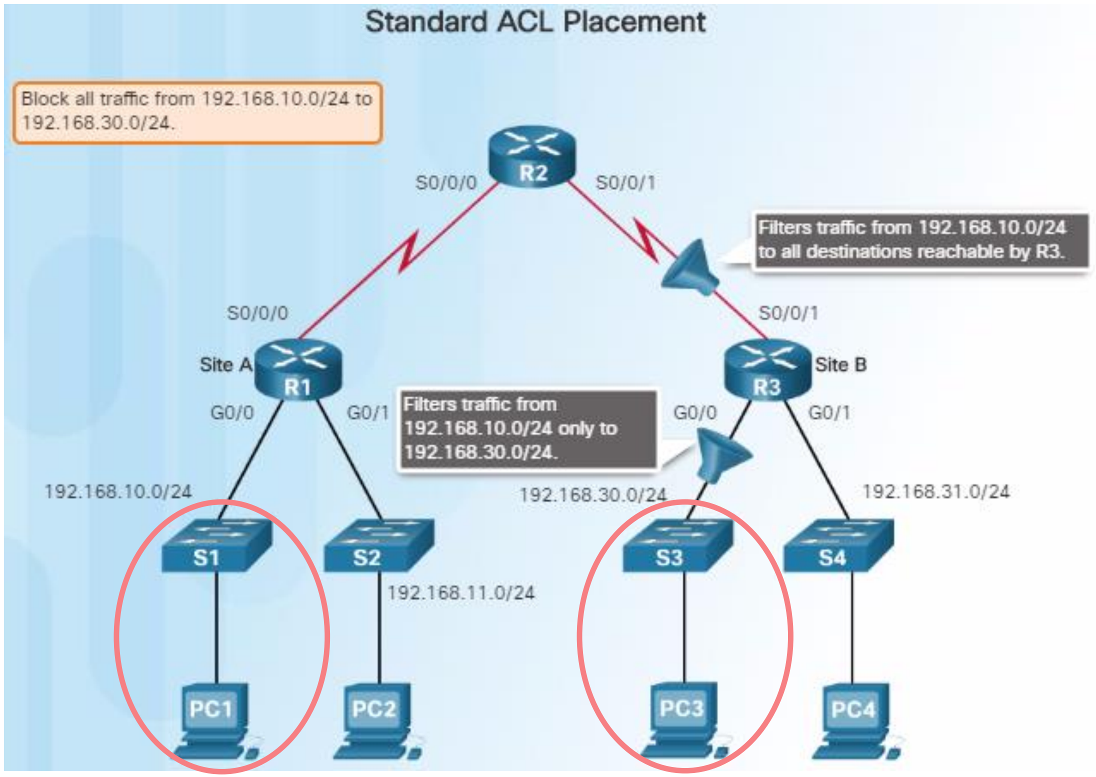
- Using numbered ACL is an effective method for determining the ACL type on smaller networks with more homogeneously defined traffic. However, a number does not provide information about the purpose of the ACL. For this reason, a name can be used to identify a Cisco ACL.

Where to place ACL



- The proper placement of an ACL can make the network operate more efficiently. For example, and ACL can be placed to reduce unnecessary traffic.
- Every ACL should be placed where it has the greatest impact on efficiency.
  - Extended ACL – Configure extended ACL as close as possible to the source of the traffic to be filtered. This will prevent undesirable traffic as close to the source without it crossing the network infrastructure.
  - Standard ACL – Since standard ACL do not specify destination addresses, they should be configured as close to the destination as possible.

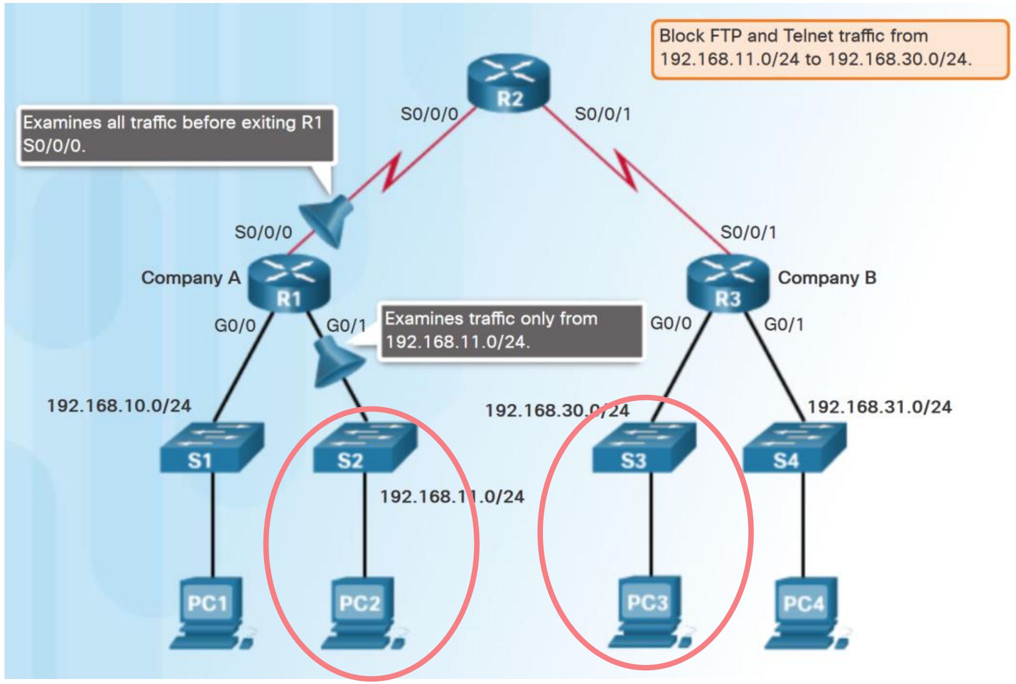
Standard ACL Placement Example


Standard ACL Placement
Block all traffic from 192.168.10.0/24 to 192.168.30.0/24.
R2
S0/0/0
S0/0/1
Filters traffic from 192.168.10.0/24 to all destinations reachable by R3.
S0/0/0
S0/0/1
Site A
R1
Site B
R3
G0/0
G0/1
Filters traffic from 192.168.10.0/24 only to 192.168.30.0/24.
G0/0
G0/1
192.168.10.0/24
192.168.30.0/24
192.168.31.0/24
S1
S2
S3
S4
192.168.11.0/24
PC1
PC2
PC3
PC4

- This example demonstrates the proper placement of the standard ACL that is configured to block traffic from the 192.168.10.0/24 network to the 192.168.30.0/24 network.
- There are two possible places to configure the access-list on R3.
- If the access-list is applied to the S0/0/1 interface, it will block traffic to the 192.168.30.0/24 network, **but also**, going to the 192.168.31.0/24 network.
- The best place to apply the access list is on R3's G0/0 interface. The access-list list should be applied to traffic exiting the G0/0 interface. Packets from 192.168.10.0/24 can still reach 192.168.31.0/24.

Guidelines for ACL Creation

Extended ACL Placement Example



Block FTP and Telnet traffic from 192.168.11.0/24 to 192.168.30.0/24.
Examines all traffic before exiting R1 S0/0/0.
Examines traffic only from 192.168.11.0/24.
R2
S0/0/0
S0/0/1
S0/0/0
S0/0/1
Company A
R1
Company B
R3
G0/0
G0/1
G0/0
G0/1
192.168.10.0/24
192.168.11.0/24
192.168.30.0/24
192.168.31.0/24
S1
S2
S3
S4
PC1
PC2
PC3
PC4

- This example demonstrates the proper placement of the extended ACL that is configured to <u>block Telnet and FTP traffic from the 192.168.11.0/24 network to the 192.168.30.0/24 network</u>.
- There are several ways to accomplish these goals. An extended ACL on R3 that blocks Telnet and FTP from the .11 network would accomplish the task, but the administrator does not control R3. In addition, this solution also allows unwanted traffic to cross the entire network, only to be blocked at the destination. This affects overall network efficiency.
- A better solution is to place an extended ACL on R1 that specifies both source and destination addresses (.11 network and .30 network, respectively), and enforces the rule, "Telnet and FTP traffic from the .11 network is not allowed to go to the .30 network."
- The basic rule for placing an extended ACL is to place it as close to the source as possible.

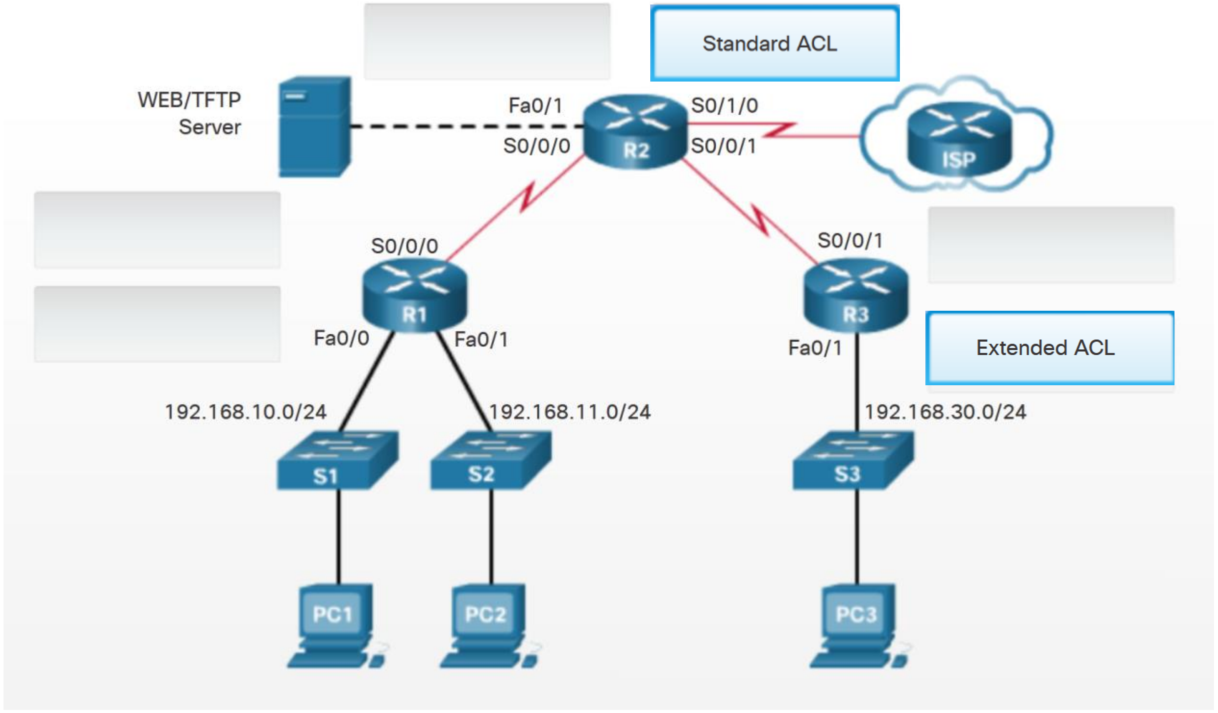
Activity – Placing Standard and Extended ACL

Network Policy #1: Use a standard ACL to stop the 192.168.10.0/24 network from accessing the Internet through ISP.

Network Policy #2: Use an extended ACL to stop the 192.168.30.0/24 network from accessing the Web/TFTP Server.

Standard ACL

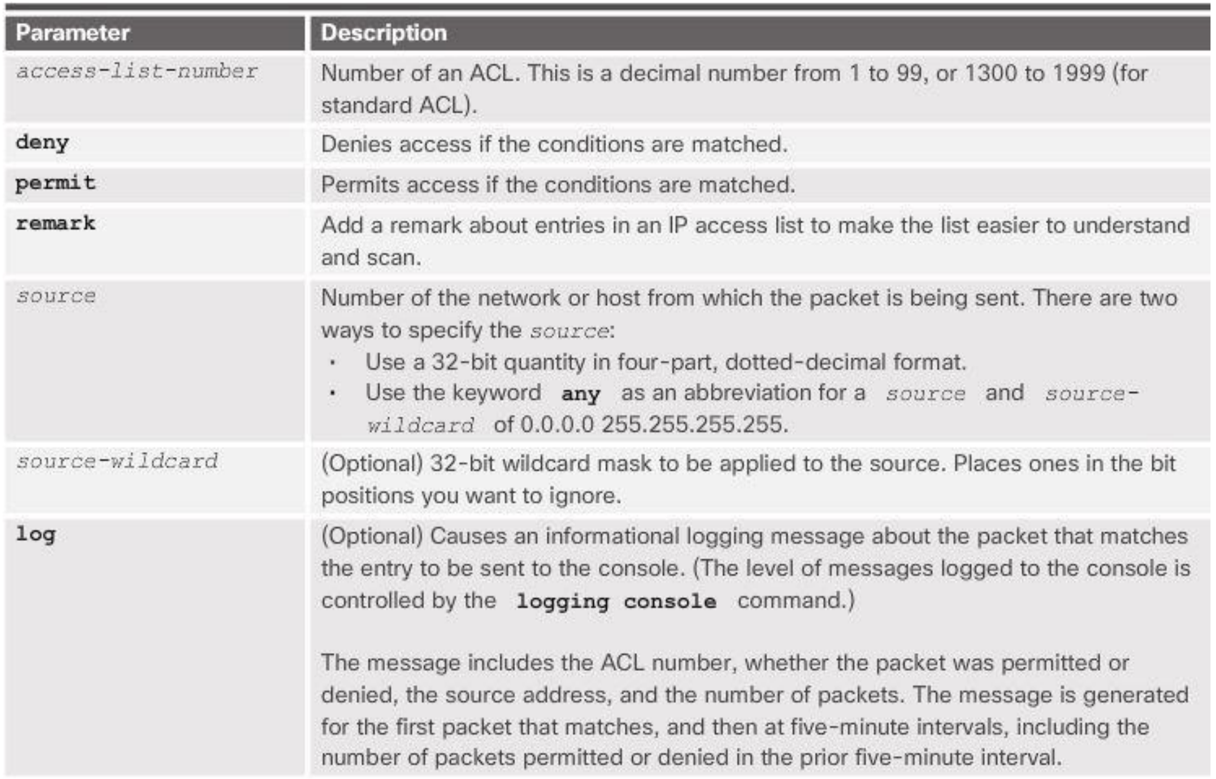
Extended ACL



Standard IPv4 ACL

Configure Standard IPv4 ACL

Numbered Standard IPv4 ACL Syntax



Parameter	Description
access-list-number	Number of an ACL. This is a decimal number from 1 to 99, or 1300 to 1999 (for standard ACL).
deny	Denies access if the conditions are matched.
permit	Permits access if the conditions are matched.
remark	Add a remark about entries in an IP access list to make the list easier to understand and scan.
source	Number of the network or host from which the packet is being sent. There are two ways to specify the *source*: • Use a 32-bit quantity in four-part, dotted-decimal format. • Use the keyword **any** as an abbreviation for a *source* and *source-wildcard* of 0.0.0.0 255.255.255.255.
source-wildcard	(Optional) 32-bit wildcard mask to be applied to the source. Places ones in the bit positions you want to ignore.
log	(Optional) Causes an informational logging message about the packet that matches the entry to be sent to the console. (The level of messages logged to the console is controlled by the **logging console** command.) The message includes the ACL number, whether the packet was permitted or denied, the source address, and the number of packets. The message is generated for the first packet that matches, and then at five-minute intervals, including the number of packets permitted or denied in the prior five-minute interval.

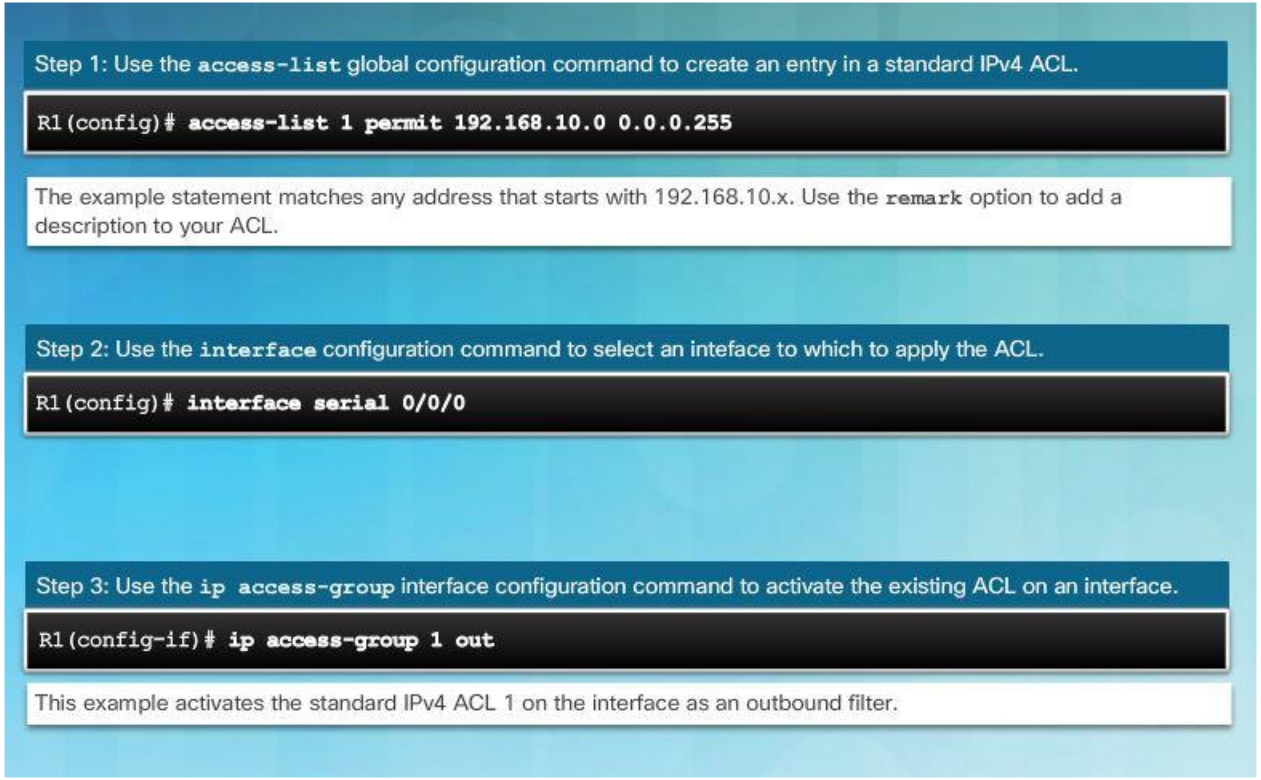
- The **access-list** global configuration command defines a standard ACL with a number in the range of 1 through 99.
- The full syntax of the standard ACL command is as follows:

  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.
- Use the **show access-list** command to verify the removal of the ACL.

Configure Standard IPv4 ACL

Applying Standard IPv4 ACL to InterfACE



Step 1: Use the `access-list` global configuration command to create an entry in a standard IPv4 ACL.

```
R1(config)# access-list 1 permit 192.168.10.0 0.0.0.255
```

The example statement matches any address that starts with 192.168.10.x. Use the `remark` option to add a description to your ACL.

Step 2: Use the `interface` configuration command to select an inteface to which to apply the ACL.

```
R1(config)# interface serial 0/0/0
```

Step 3: Use the `ip access-group` interface configuration command to activate the existing ACL on an interface.

```
R1(config-if)# ip access-group 1 out
```

This example activates the standard IPv4 ACL 1 on the interface as an outbound filter.

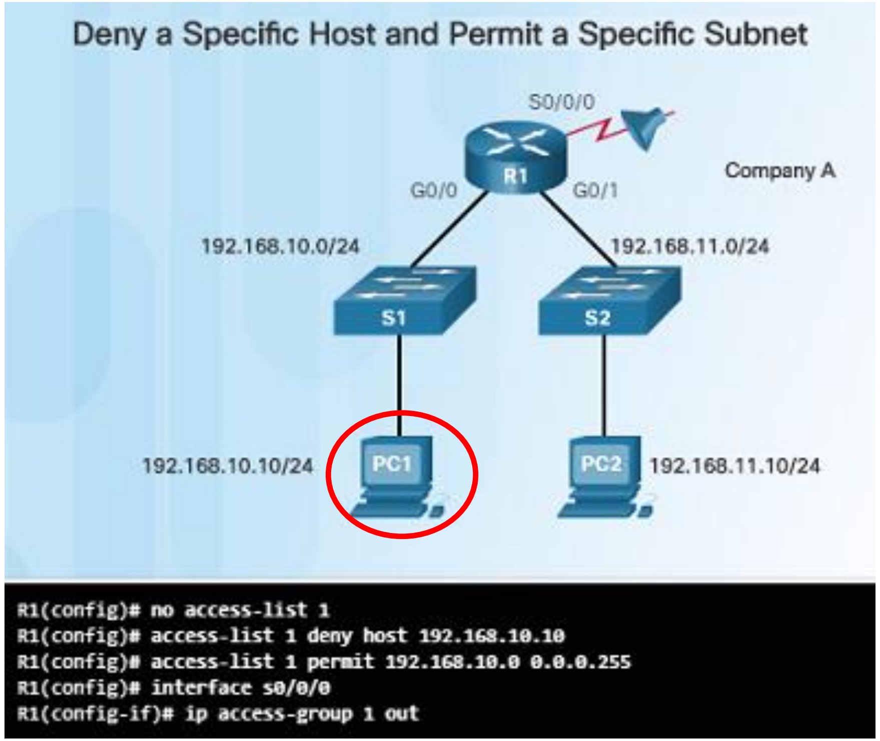
- After a standard IPv4 ACL is configured, it is linked to an interface using the **`ip access-group`** command in interface configuration mode:

  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.

Configure Standard IPv4 ACL

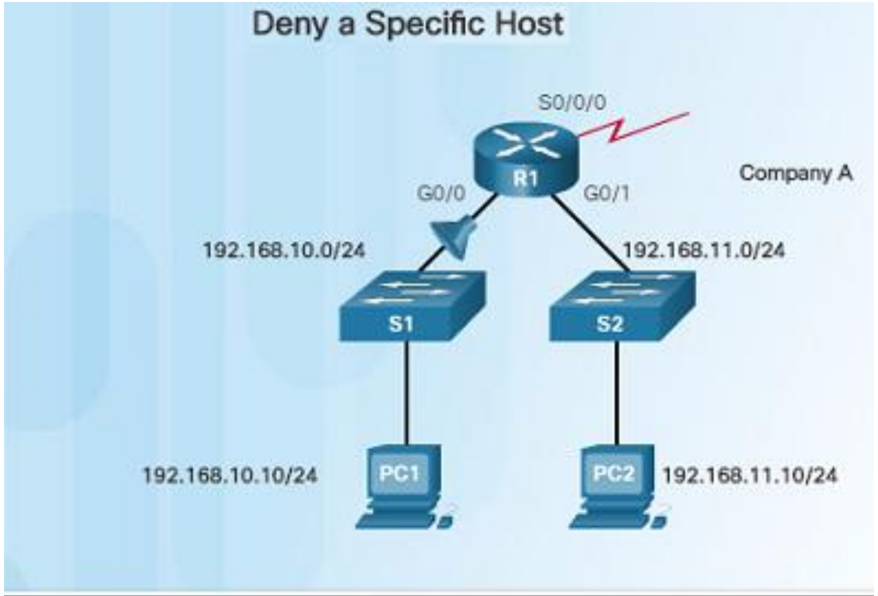
Numbered Standard IPv4 ACL Examples



```
R1(config)# no access-list 1
R1(config)# access-list 1 deny host 192.168.10.10
R1(config)# access-list 1 permit 192.168.10.0 0.0.0.255
R1(config)# interface s0/0/0
R1(config-if)# ip access-group 1 out
```

- The figure to the left shows an example of an ACL that permits traffic from a specific subnet but denies traffic from a specific host on that subnet.
  - The **`no access-list 1`** command deletes the previous version of ACL 1.
  - The next ACL statement denies the host 192.168.10.10.
  - What is another way to write this command without using **`host`**?
  - All other hosts on the 192.168.10.0/24 network are then permitted.
  - There is an implicit deny statement that matches every other network.
  - Next, the ACL is reapplied to the interface in an outbound direction.

Numbered Standard IPv4 ACL Examples (Cont.)

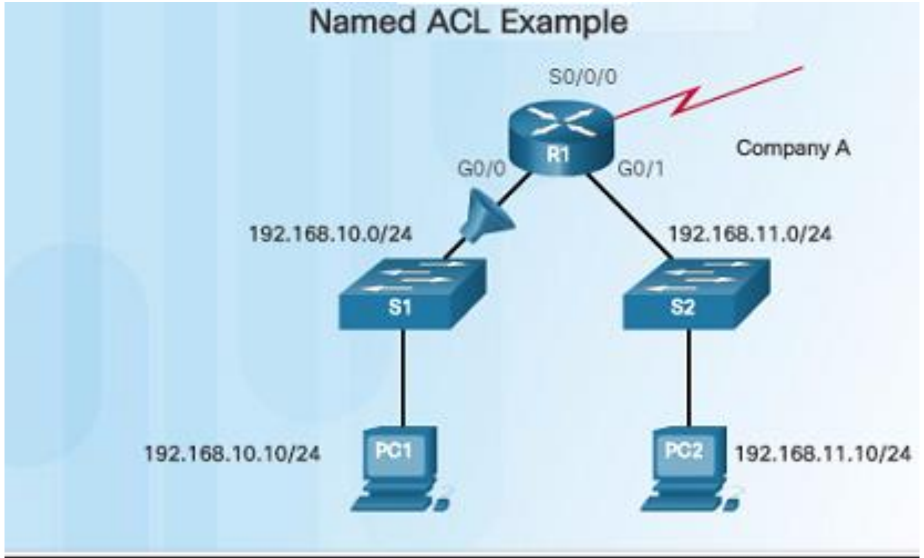


```
R1(config)# no access-list 1
R1(config)# access-list 1 deny host 192.168.10.10
R1(config)# access-list 1 permit any
R1(config)# interface g0/0
R1(config-if)# ip access-group 1 in
```

- This next example demonstrates an ACL that denies a specific host but will permit all other traffic.
  - The first ACL statement deletes the previous version of ACL 1.
  - The next command, with the deny keyword, will deny traffic from the PC1 host that is located at 192.168.10.10.
  - The **`access-list 1 permit any`** statement will permit all other hosts.
  - This ACL is applied to interface G0/0 in the inbound direction since it only affects the 192.168.10.0/24 LAN.

Configure Standard IPv4 ACL

Named Standard IPv4 ACL Syntax

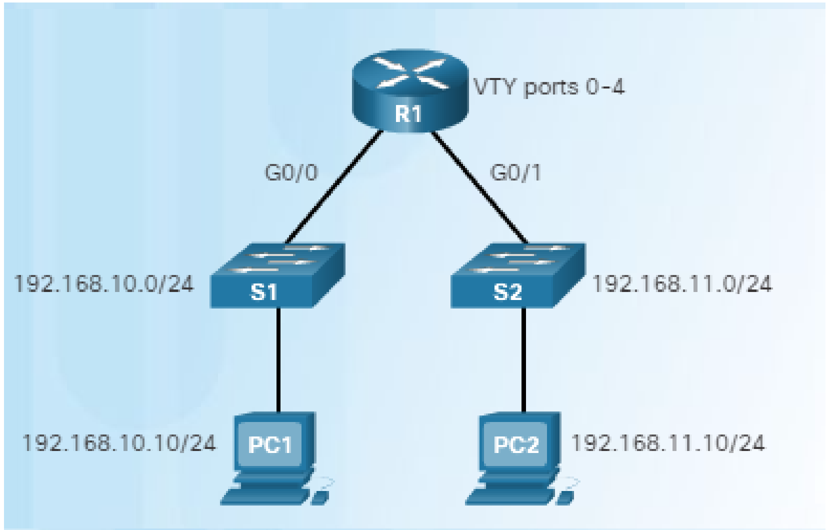


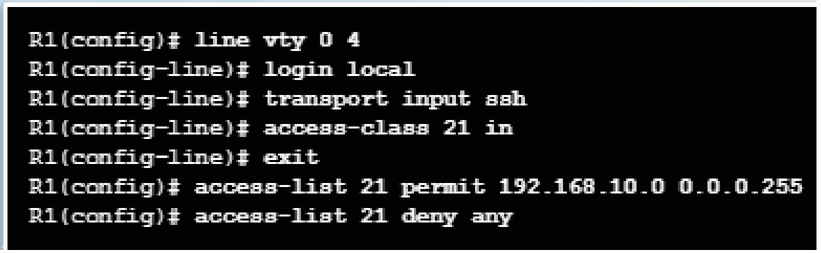
```
R1(config)# ip access-list standard NO_ACCESS
R1(config-std-nacl)# deny host 192.168.11.10
R1(config-std-nacl)# permit any
R1(config-std-nacl)# exit
R1(config)# interface g0/0
R1(config-if)# ip access-group NO_ACCESS out
```

- Identifying an ACL with a name rather than with a number makes it easier to understand its function.
- The example to the left shows how to configured a named standard access list. Notice how the commands are slightly different:
  - Use the **`ip access-list`** command to create a named ACL. Names are alphanumeric, case sensitive, and must be unique.
  - Use permit or deny statements as needed. You can also use the **`remark`** command to add comments.
  - Apply the ACL to an interface using the **`ip access-group`** *name* command.

Securing VTY ports with a Standard IPv4 ACL

The access-class Command



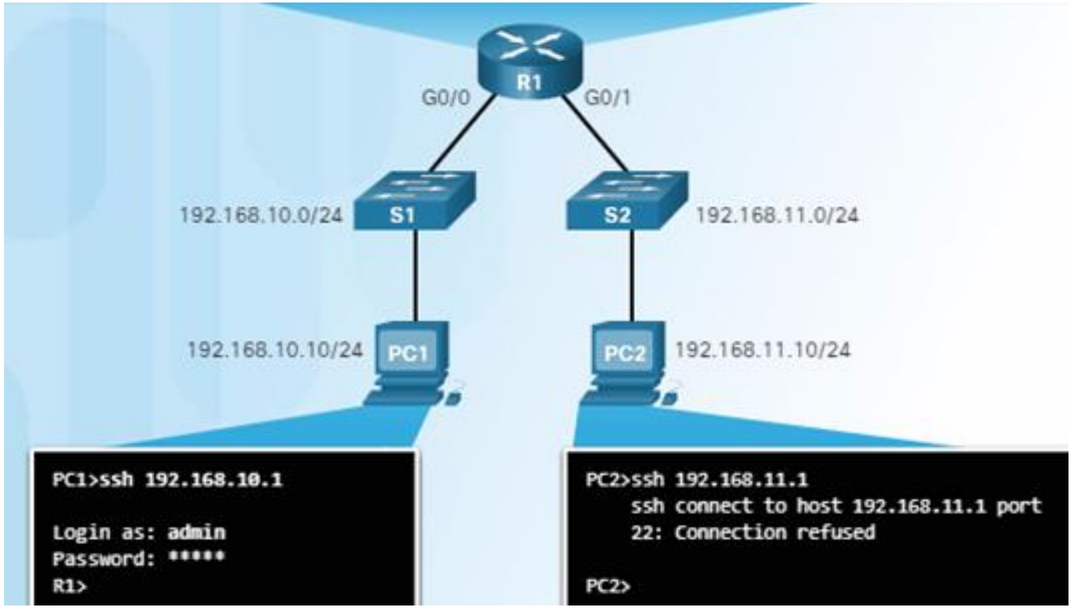


```
R1(config)# line vty 0 4
R1(config-line)# login local
R1(config-line)# transport input ssh
R1(config-line)# access-class 21 in
R1(config-line)# exit
R1(config)# access-list 21 permit 192.168.10.0 0.0.0.255
R1(config)# access-list 21 deny any
```

- Administrative VTY access to Cisco devices should be restricted to help improve security.
- Restricting VTY access is a technique that allows you define which IP addresses are allowed remote access to the router EXEC process.
- The access-class command configured in line configuration mode will restrict incoming and outgoing connections between a particular VTY (into a Cisco device) and the addresses in an access list.
- Router(config-line)# **`access-class`** *access-list-number* {`in` [vrf-also] | `out` }

Securing VTY ports with a Standard IPv4 ACL

Verifying the VTY Port is Secured

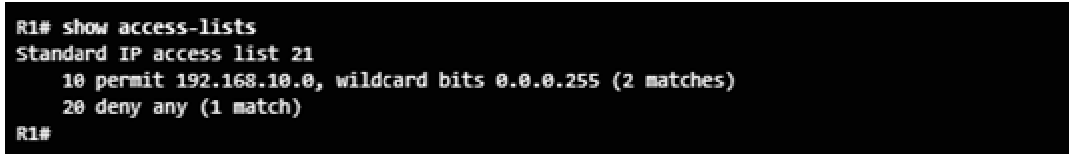


```
PC1>ssh 192.168.10.1

Login as: admin
Password: *****
R1>
```

```
PC2>ssh 192.168.11.1
    ssh connect to host 192.168.11.1 port
    22: Connection refused

PC2>
```



```
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#
```

- Verification of the ACL configuration used to restrict VTY access is important.
- The figure to the left shows two devices trying to ssh into two different devices.
- The show access-lists command output shows the results after the SSH attempts by PC1 and PC2.
- Notice the match results in the permit and the deny statements.

Extended IPv4 ACL

Extended IPv4 ACL

Structure of an Extended IPv4 ACL

Extended ACLs Can Filter On

- Source address
- Destination address
- Protocol
- Port number

- Extended IPv4 ACL provide more precise filtering.
  - Extended ACL are numbered 100 to 199 and 2000 to 2699, providing a total of 799 possible extended numbered ACL.
  - Extended ACL can also be named.
  - Extended ACL are used more often than standard ACL because they provide a greater degree of control.

Structure of an Extended IPv4 ACL

- Extended ACL can filter on protocol and port number.

- An application can be specified by configuring either:
  - The port number

```
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
```

  - The name of a well-known port.

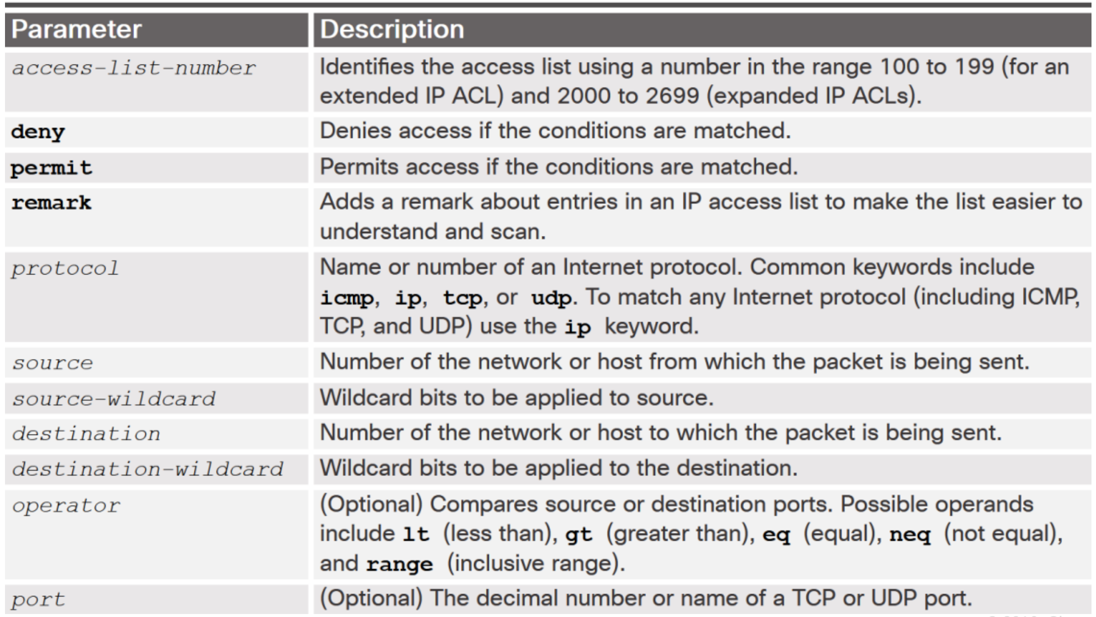
```
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq telnet
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq ftp
access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq ftp-data
```

- Note:
  - Use the question mark (?) to see available well-known port names.
  - E.g., **access-list 101 permit tcp any any eq ?**

Extended IPv4 ACL

Configure Extended IPv4 ACL

- The full syntax of the extended ACL command is as follows:
  - **access-list** *ACL-#* {**deny** | **permit** | **remark**} *protocol* {*source source-wildcard*] [*operator* [*port-number* | *port-name*]] {*destination destination-wildcard*] [*operator* [*port-number* | *port-name*]]



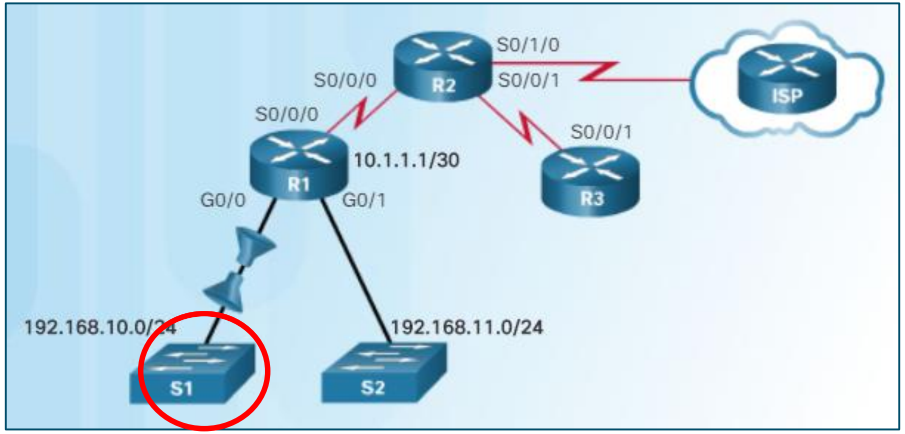
Parameter	Description
access-list-number	Identifies the access list using a number in the range 100 to 199 (for an extended IP ACL) and 2000 to 2699 (expanded IP ACLs).
deny	Denies access if the conditions are matched.
permit	Permits access if the conditions are matched.
remark	Adds a remark about entries in an IP access list to make the list easier to understand and scan.
protocol	Name or number of an Internet protocol. Common keywords include **icmp**, **ip**, **tcp**, or **udp**. To match any Internet protocol (including ICMP, TCP, and UDP) use the **ip** keyword.
source	Number of the network or host from which the packet is being sent.
source-wildcard	Wildcard bits to be applied to source.
destination	Number of the network or host to which the packet is being sent.
destination-wildcard	Wildcard bits to be applied to the destination.
operator	(Optional) Compares source or destination ports. Possible operands include **lt** (less than), **gt** (greater than), **eq** (equal), **neq** (not equal), and **range** (inclusive range).
port	(Optional) The decimal number or name of a TCP or UDP port.

Extended IPv4 ACL

Configure Extended IPv4 ACL

- Applying extended ACL is similar to standard ACL except that they should be applied as close to the source.

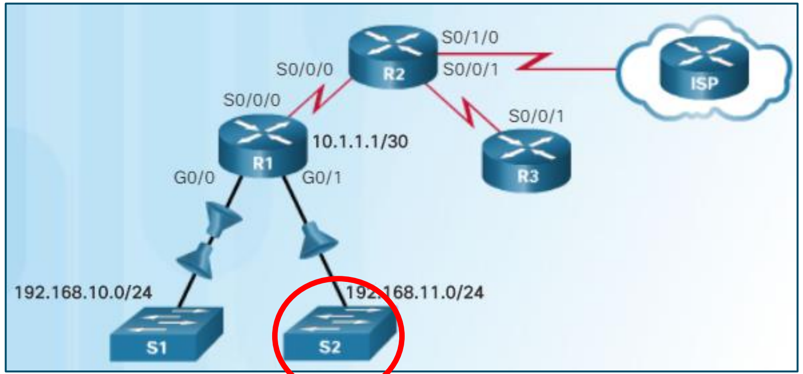
- For example:
  - ACL 103 only allows requests to port 80 and 443.
  - ACL 104 allows established HTTP and HTTPS replies.
  - The **established** parameter allows only responses to traffic that originates from the 192.168.10.0/24 network to return to that network.



```
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
```

Configure Extended IPv4 ACL

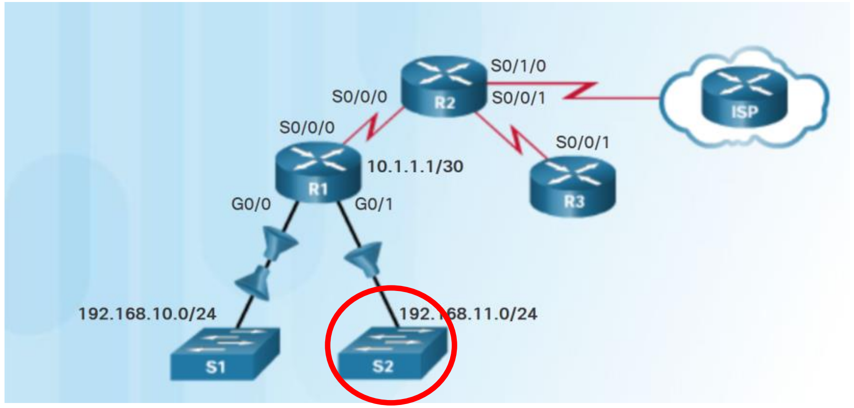
- In this example, FTP traffic from subnet 192.168.11.0 going to subnet 192.168.10.0 is denied, but all other traffic is permitted.
  - FTP utilizes two port numbers (TCP port 20 and 21) therefore two ACE are required.
  - The example uses the well-known port names **ftp** and **ftp-data**.
  - Without at least one permit statement in an ACL, all traffic on the interface where that ACL was applied would be dropped.
  - The ACL is applied incoming on the R1 G0/1 interface.



```
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
```

Configure Extended IPv4 ACL

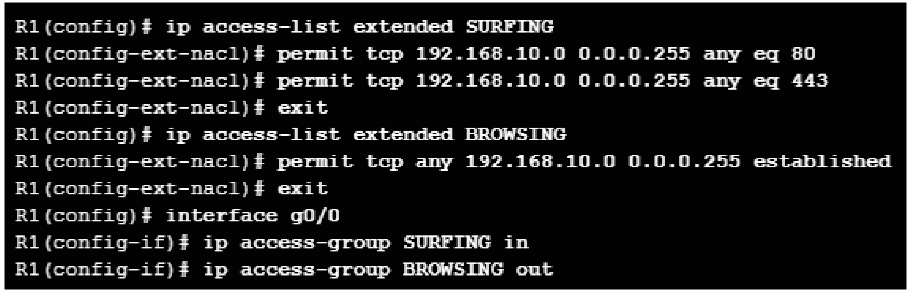
- The example denies Telnet traffic from any source to the 192.168.11.0/24 LAN but allows all other IP traffic.
  - Because traffic destined for the 192.168.11.0/24 LAN is outbound on interface G0/1, the ACL would be applied to G0/1 using the **out** keyword.
  - Note the use of the **any** keywords in the permit statement. This permit statement is added to ensure that no other traffic is blocked.



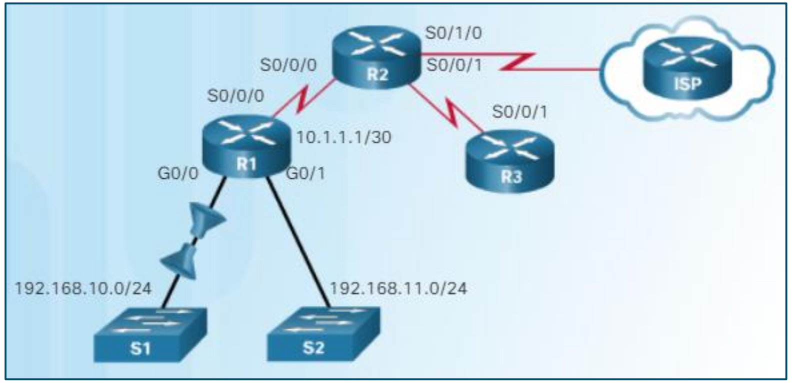
```
R1(config)# access-list 102 deny tcp any 192.168.11.0 0.0.0.255 eq 23
R1(config)# access-list 102 permit ip any any
R1(config)# interface g0/1
R1(config-if)# ip access-group 102 out
```

Configure Extended IPv4 ACL

- Named extended ACL are created in the same way that named standard ACL are created.
- In this example, two named ACL are created.
  - SURFING permits users on the 192.168.10.0/24 network to exit going to ports 80 and 443.
  - BROWSING enables return HTTP and HTTPs traffic.

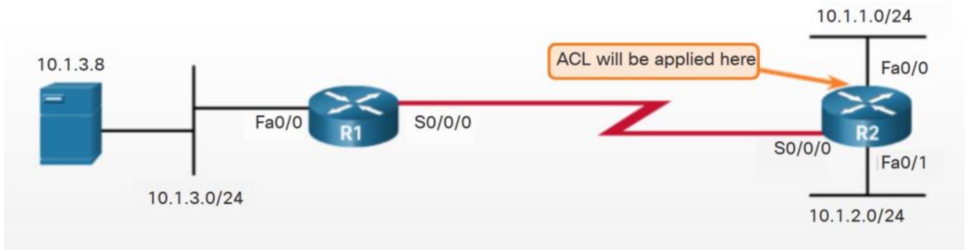


```
R1(config)# ip access-list extended SURFING
R1(config-ext-nacl)# permit tcp 192.168.10.0 0.0.0.255 any eq 80
R1(config-ext-nacl)# permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1(config-ext-nacl)# exit
R1(config)# ip access-list extended BROWSING
R1(config-ext-nacl)# permit tcp any 192.168.10.0 0.0.0.255 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
```



Configure Extended IPv4 ACL

Create a numbered ACL statement that will only allow users on the 10.1.1.0/24 network to have HTTP access to the web server on the 10.1.3.0/24 network. The ACL is applied to R2 Fa0/0 inbound.



access-list	101	permit	tcp	10.1.1.0	0.0.0.255	host	10.1.3.8	eq 80

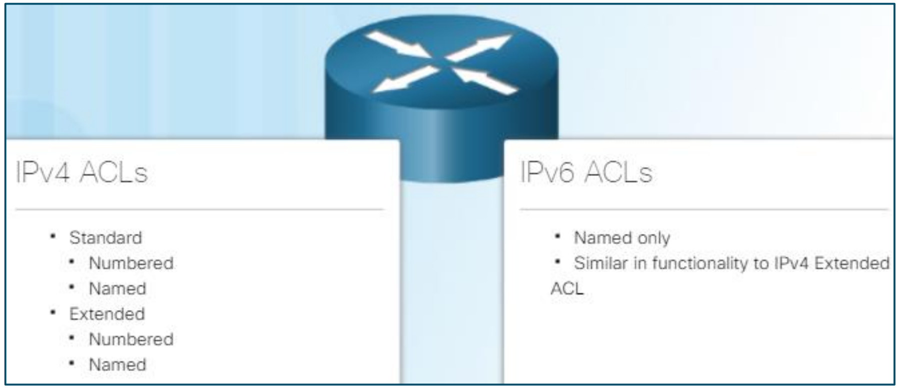
99	0.0.0.0	10.1.3.0	10.1.2.0	permit	0.0.0.255	udp	any	eq 21
tcp	host	101	eq 80	10.1.1.0	10.1.3.8	access-list	deny	ip

IPv6 ACL

IPv6 ACL Creation

- IPv6 ACL are similar to IPv4 ACL in both operation and configuration.

In IPv4 there are two types of ACL, standard and extended and both types of ACL can be either numbered or named ACL.



IPv4 ACLs

- Standard
  - Numbered
  - Named
- Extended
  - Numbered
  - Named

IPv6 ACLs

- Named only
- Similar in functionality to IPv4 Extended ACL

With IPv6, there is only one type of ACL, which is equivalent to an IPv4 extended named ACL and there are no numbered ACL in IPv6.

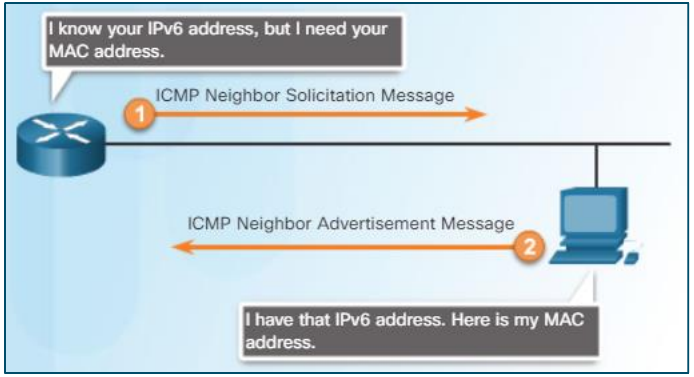
- **Note:**
  - An IPv4 ACL and an IPv6 ACL cannot share the same name.

IPv6 ACL

IPv6 ACL Creation

- There are three significant differences between IPv4 and IPv6 ACL:
  - The command used to apply an IPv6 ACL to an interface is **ipv6 traffic-filter** command.
  - IPv6 ACL do not use wildcard masks but instead specifies the prefix-length to indicate how much of an IPv6 source or destination address should be matched.
  - An IPv6 ACL adds two implicit permit statements at the end of each IPv6 access list.
    - **permit icmp any any nd-na**
    - **permit icmp any any nd-ns**
    - **deny ipv6 any any statement**

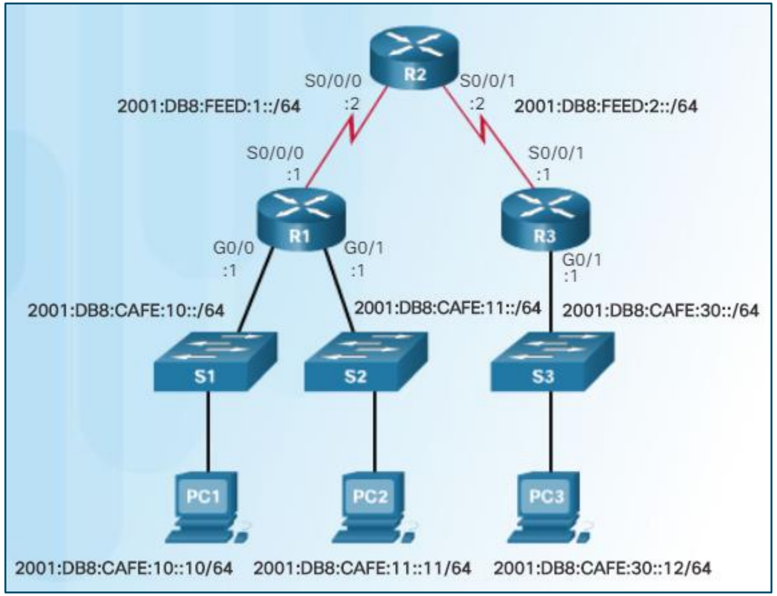
- These two additional statements allow IPv6 ICMP Neighbor Discovery (ND) and Neighbor Solicitation (NS) messages to accomplish the same thing as IPv4 ARP.


I know your IPv6 address, but I need your MAC address.
1
ICMP Neighbor Solicitation Message
ICMP Neighbor Advertisement Message
2
I have that IPv6 address. Here is my MAC address.

Configuring IPv6 ACL

- The following is the sample topology that will be used to demonstrate IPv6 ACL.
  - All interfACE are configured and active.

```
R1# show ipv6 interface brief
GigabitEthernet0/0      [up/up]
    FE80::FE99:47FF:FE75:C3E0
    2001:DB8:CAFE:10::1
GigabitEthernet0/1      [up/up]
    FE80::FE99:47FF:FE75:C3E1
    2001:DB8:CAFE:11::1
Serial0/0/0             [up/up]
    FE80::FE99:47FF:FE75:C3E0
    2001:DB8:FEED:1::1
 <output omitted>
R1#
```



```
R2# show ipv6 interface brief
Serial0/0/0             [up/up]
    FE80::FE99:47FF:FE71:78A0
    2001:DB8:FEED:1::2
Serial0/0/1             [up/up]
    FE80::FE99:47FF:FE71:78A0
    2001:DB8:FEED:2::2
 <output omitted>
R2#
```

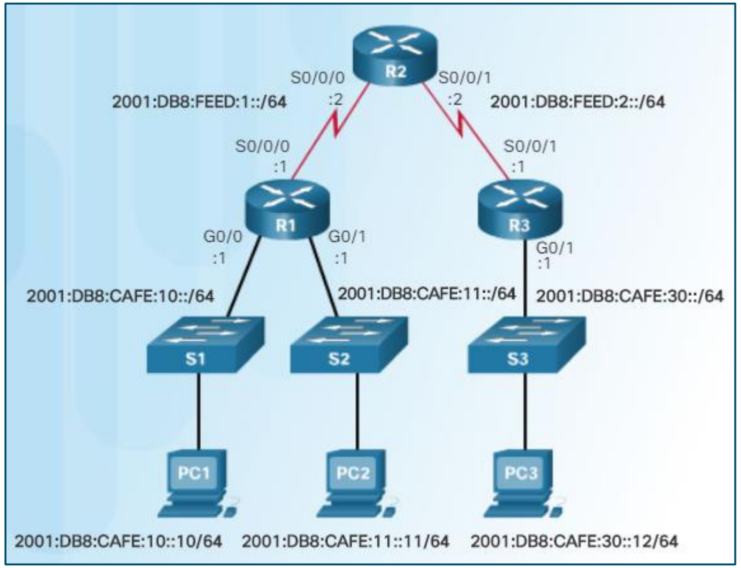
```
R3# show ipv6 interface brief
GigabitEthernet0/0      [up/up]
    FE80::FE99:47FF:FE71:7A20
    2001:DB8:CAFE:30::1
Serial0/0/1             [up/up]
    FE80::FE99:47FF:FE71:7A20
    2001:DB8:FEED:2::1
R3#
```

Configuring IPv6 ACL

- In IPv6 there are only named ACL and the configuration is similar to IPv4 extended named ACL.

```
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]]
```

```
R1(config)# ipv6 access-list NO-R3-LAN-ACCESS
R1(config-ipv6-acl)# deny ipv6 2001:db8:cafe:30::/64 any
R1(config-ipv6-acl)# permit ipv6 any any
R1(config-ipv6-acl)# end
R1#
```



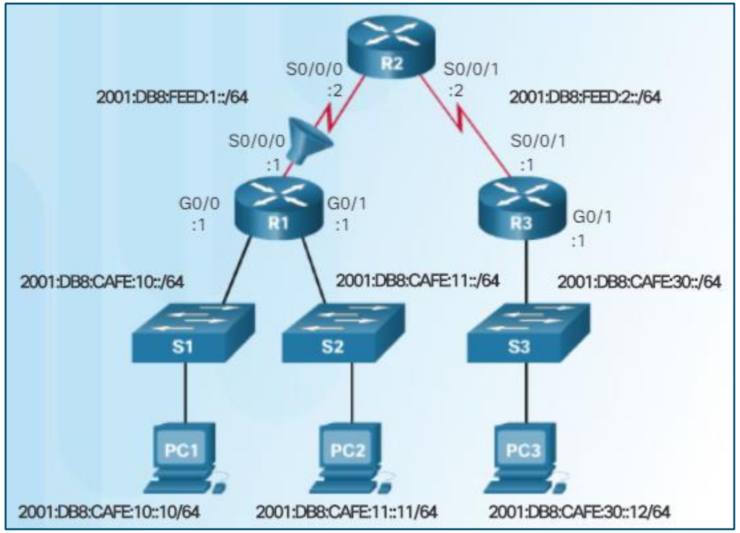
- In this example:
  - The 1st statement names the IPv6 ACL **NO-R3-LAN-ACCESS**.
  - The 2nd statement denies all IPv6 packets from the 2001:DB8:CAFE:30::/64 destined for any IPv6 network.
  - The 3rd statement allows all other IPv6 packets.

Configuring IPv6 ACL

- After an IPv6 ACL is configured, it is linked to an interface using the following interface command:
  - **`ipv6 traffic-filter`** *`access-list-name`* `{`**`in`**` | `**`out`**`}`

The command applies the NO-R3-LAN-ACCESS IPv6 ACL inbound to the S0/0/0 interface of R1.

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



- To remove an IPv6 ACL, enter the **no ipv6 traffic-filter** command on the interface, and then enter the global **no ipv6 access-list** command to remove the access list.

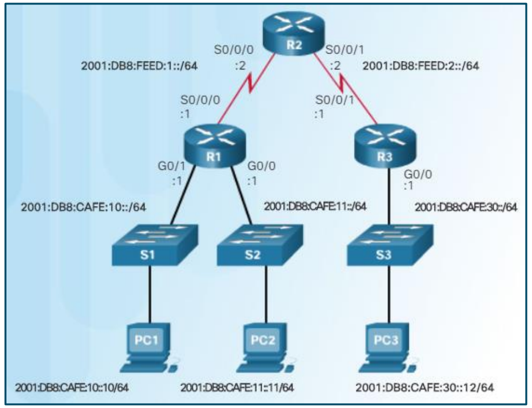
- Note that IPv4 and IPv6 both use the **access-class** command to apply an access list to VTY ports.

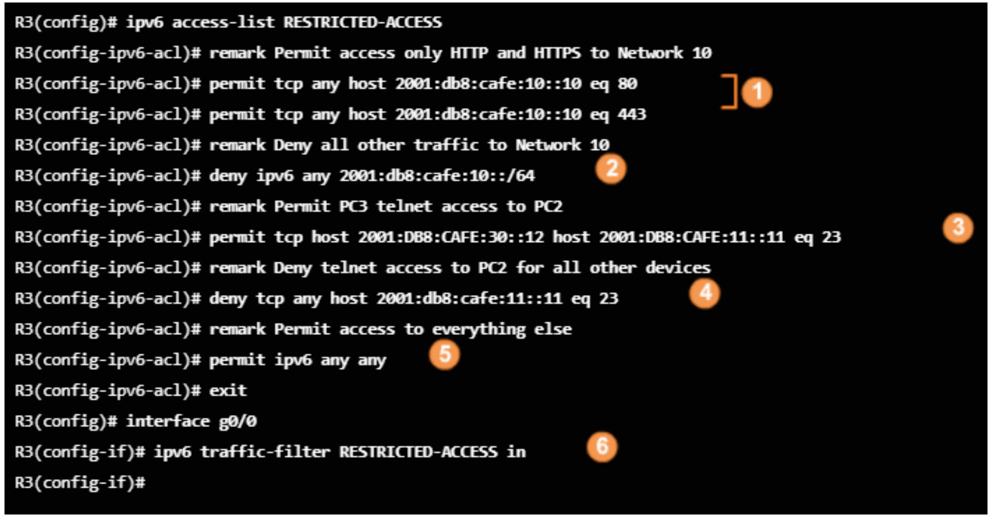
IPv6 ACL

Configuring IPv6 ACL

- In this example, an IPv6 ACL permits R3 LAN users limited access to the LANs on R1.

1. These ACE allow access from any device to the web server (2001:DB8:CAFE:10::10).
2. All other devices are denied access to the 2001:DB8:CAFE:10::/64 network.
3. PC3 (2001:DB8:CAFE:30::12) is permitted Telnet access to PC2 (2001:DB8:CAFE:11::11).
4. All others are denied Telnet access to PC2.
5. All other IPv6 traffic is permitted to all other destinations.
6. The IPv6 access list is applied inbound on G0/0 so only the 2001:DB8:CAFE:30::/64 network is affected.



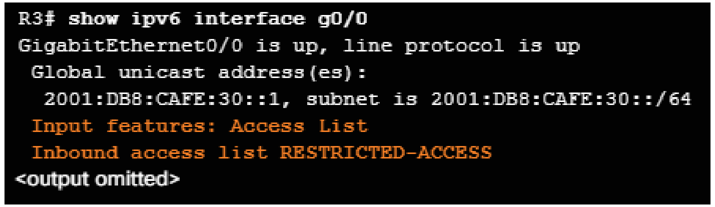


```
R3(config)# ipv6 access-list RESTRICTED-ACCESS
R3(config-ipv6-acl)# remark Permit access only HTTP and HTTPS to Network 10
R3(config-ipv6-acl)# permit tcp any host 2001:db8:cafe:10::10 eq 80
R3(config-ipv6-acl)# permit tcp any host 2001:db8:cafe:10::10 eq 443
R3(config-ipv6-acl)# remark Deny all other traffic to Network 10
R3(config-ipv6-acl)# deny ipv6 any 2001:db8:cafe:10::/64
R3(config-ipv6-acl)# remark Permit PC3 telnet access to PC2
R3(config-ipv6-acl)# permit tcp host 2001:DB8:CAFE:30::12 host 2001:DB8:CAFE:11::11 eq 23
R3(config-ipv6-acl)# remark Deny telnet access to PC2 for all other devices
R3(config-ipv6-acl)# deny tcp any host 2001:db8:cafe:11::11 eq 23
R3(config-ipv6-acl)# remark Permit access to everything else
R3(config-ipv6-acl)# permit ipv6 any any
R3(config-ipv6-acl)# exit
R3(config)# interface g0/0
R3(config-if)# ipv6 traffic-filter RESTRICTED-ACCESS in
R3(config-if)#
```

IPv6 ACL

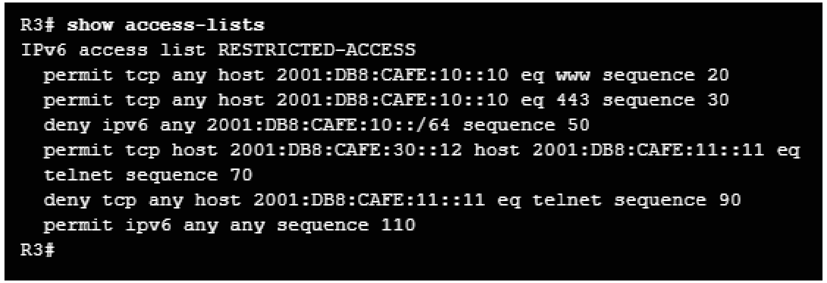
Configuring IPv6 ACL

- The commands used to verify an IPv6 access list are similar to those used for IPv4 ACL.
- Use the **show ipv6 interface** command to see which ACL and direction is configured on an interface.



```
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
<output omitted>
```

- Use the **show access-lists** command displays all configured IPv4 and IPv6 access lists
  - Notice that IPv6 ACL sequence numbers are displayed at the end of the ACE.



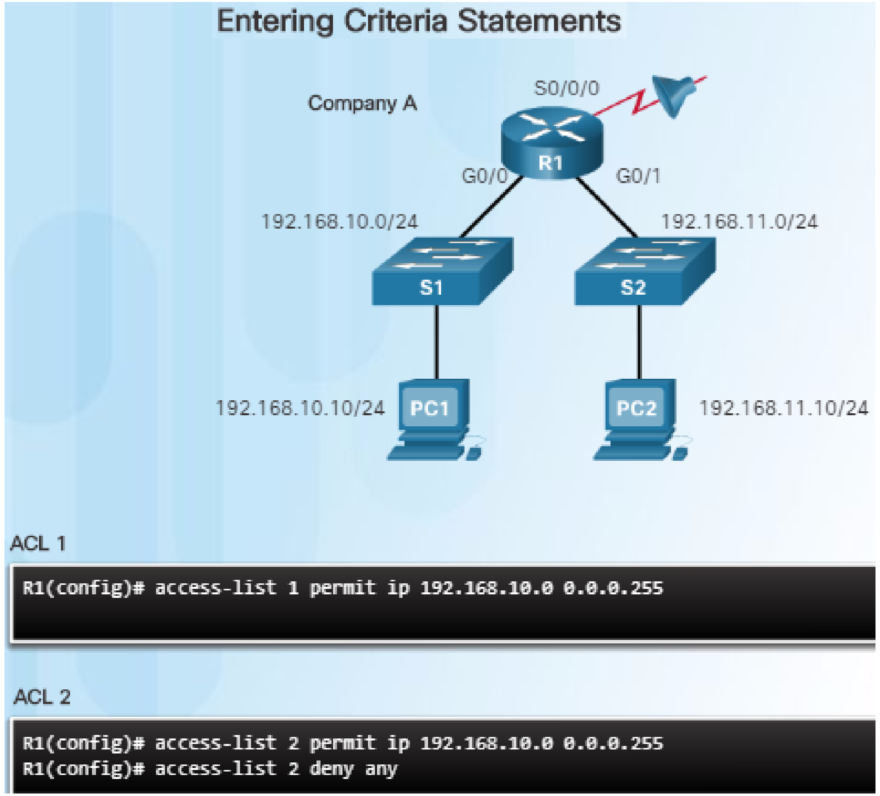
```
R3# show access-lists
IPv6 access list RESTRICTED-ACCESS
  permit tcp any host 2001:DB8:CAFE:10::10 eq www sequence 20
  permit tcp any host 2001:DB8:CAFE:10::10 eq 443 sequence 30
  deny ipv6 any 2001:DB8:CAFE:10::/64 sequence 50
  permit tcp host 2001:DB8:CAFE:30::12 host 2001:DB8:CAFE:11::11 eq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

- The **show running-config** command displays all of the ACE and remark statements.

Troubleshoot IPv4 ACL

Troubleshoot ACL

The Implicit Deny Any



ACL 1

```
R1(config)# access-list 1 permit ip 192.168.10.0 0.0.0.255
```

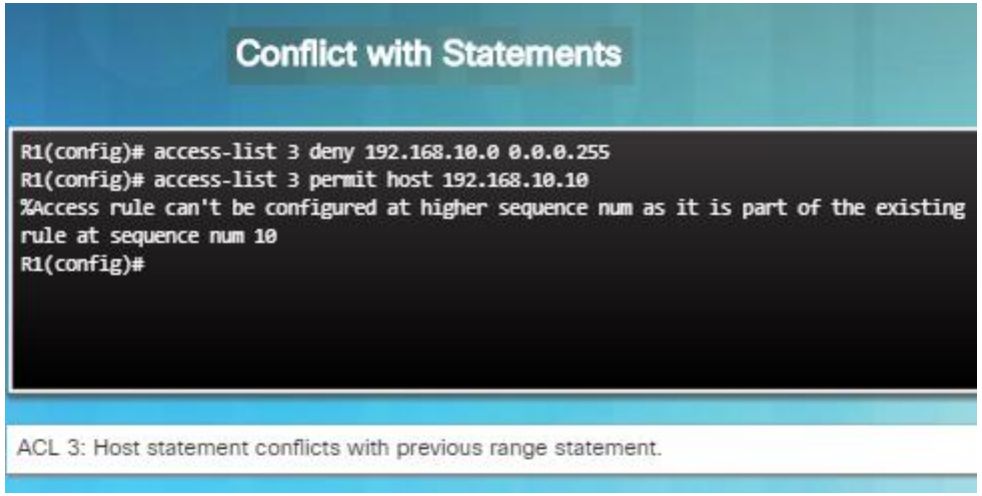
ACL 2

```
R1(config)# access-list 2 permit ip 192.168.10.0 0.0.0.255
R1(config)# access-list 2 deny any
```

- A single-entry ACL with only one deny entry has the effect of denying all traffic.
- At least one permit ACE must be configured in an ACL or all traffic will be blocked.
- Study the two ACL in the figure to the left.
  - Will the results be the same or different?

Troubleshoot ACL

The Order of ACE in an ACL



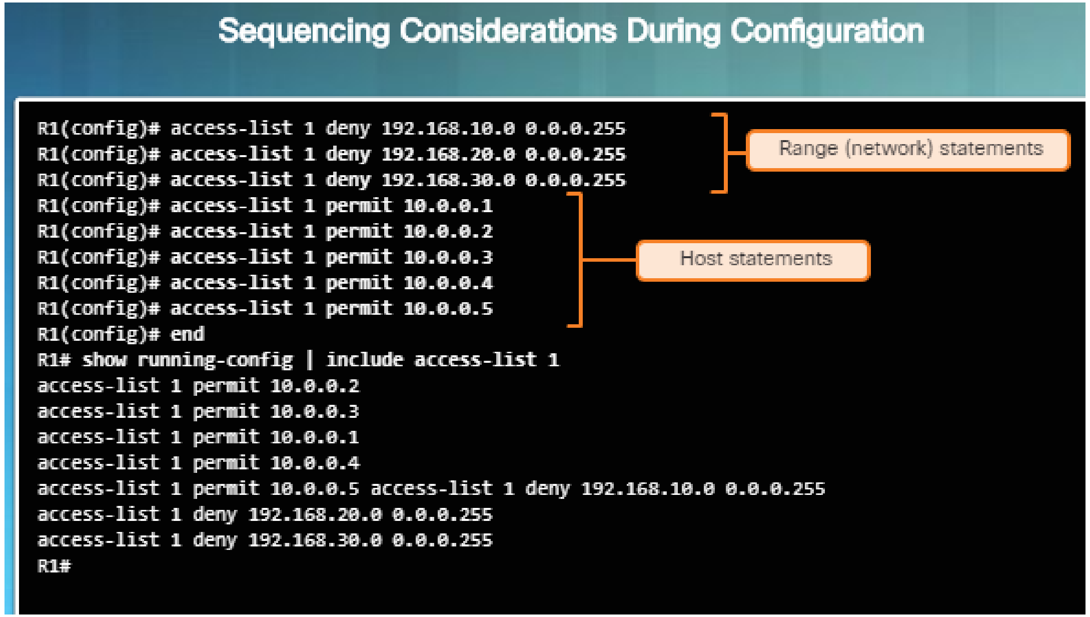
```
R1(config)# access-list 3 deny 192.168.10.0 0.0.0.255
R1(config)# access-list 3 permit host 192.168.10.10
%Access rule can't be configured at higher sequence num as it is part of the existing
rule at sequence num 10
R1(config)#
```

ACL 3: Host statement conflicts with previous range statement.

- The order in which ACE are configured are important since ACE are processed sequentially.
- The figure to the left demonstrates a conflict between two statements since they are in the wrong order.
  - The first deny statement blocks everything in the 192.168.10.0/24 network.
  - However, the second permit statement is attempting to allow host 192.168.10.10 through.
  - This statement is rejected since it is a subset of the previous statement.
  - Reversing the order of these two statements will solve the problem.

Troubleshoot ACL

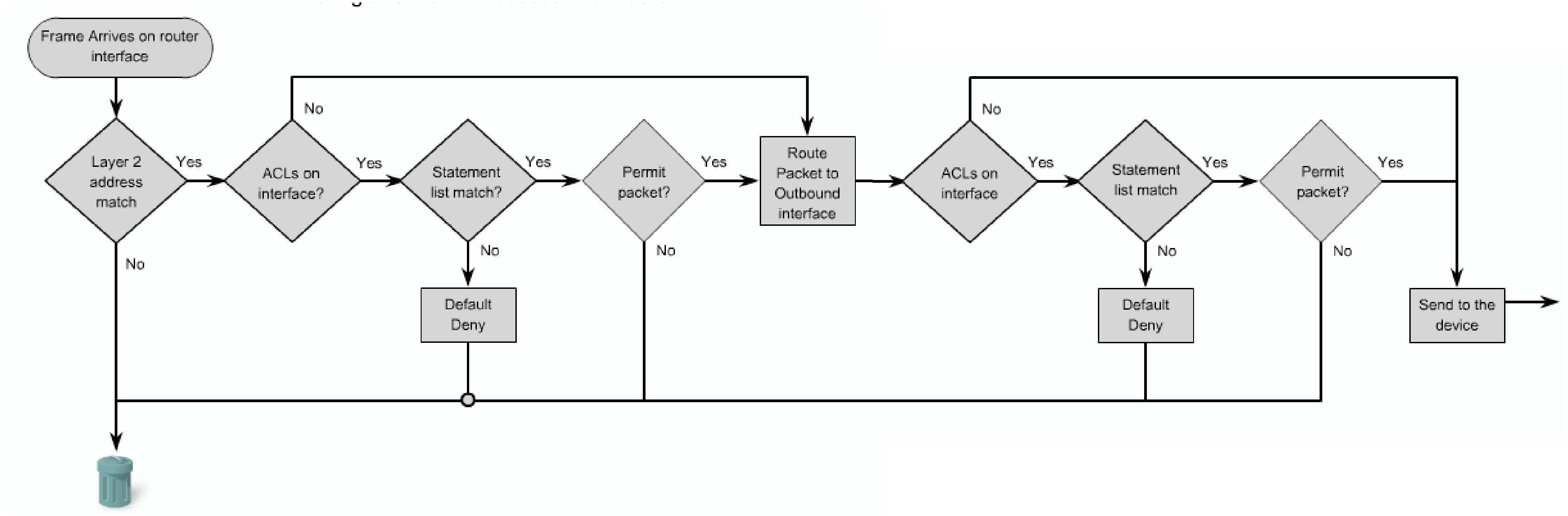
Cisco IOS Reorders Standard ACL


Sequencing Considerations During Configuration

```
R1(config)# access-list 1 deny 192.168.10.0 0.0.0.255
R1(config)# access-list 1 deny 192.168.20.0 0.0.0.255
R1(config)# access-list 1 deny 192.168.30.0 0.0.0.255
R1(config)# access-list 1 permit 10.0.0.1
R1(config)# access-list 1 permit 10.0.0.2
R1(config)# access-list 1 permit 10.0.0.3
R1(config)# access-list 1 permit 10.0.0.4
R1(config)# access-list 1 permit 10.0.0.5
R1(config)# end
R1# show running-config | include access-list 1
access-list 1 permit 10.0.0.2
access-list 1 permit 10.0.0.3
access-list 1 permit 10.0.0.1
access-list 1 permit 10.0.0.4
access-list 1 permit 10.0.0.5 access-list 1 deny 192.168.10.0 0.0.0.255
access-list 1 deny 192.168.20.0 0.0.0.255
access-list 1 deny 192.168.30.0 0.0.0.255
R1#
```

- Note the order in which the access-list statements were entered during configuration.
- Notice how the order was changed when you enter the **show running-config** command.
- The host statements are listed first, however, not in the order they were entered.
- The IOS puts host statements in an order using a special hashing function. The resulting order optimizes the search for a host ACL entry.
- The range statements are displayed in the order they were entered. The hashing function is applied to host statements.

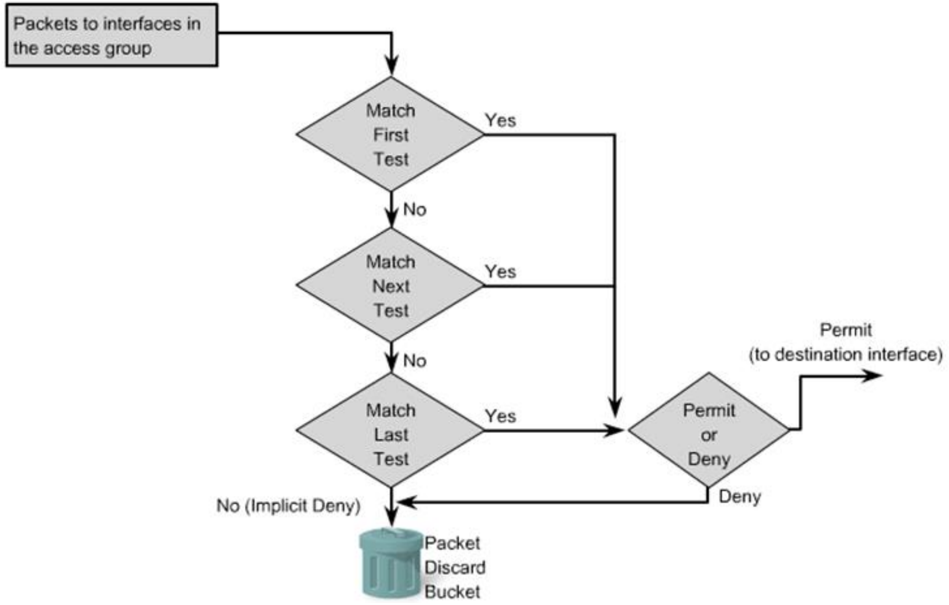
Routing Processes and ACL



Troubleshoot ACL

Processing Packets with ACL

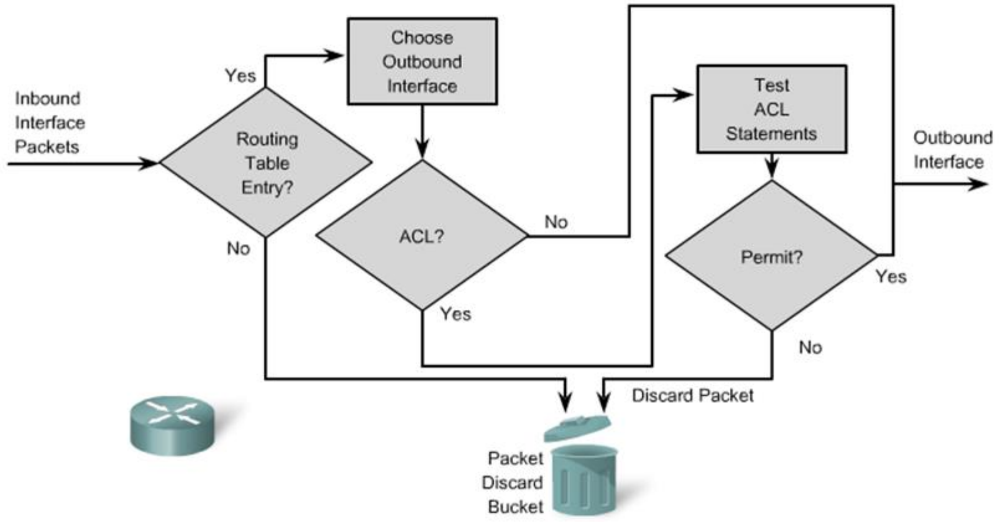
- It is beneficial to consider how an inbound and outbound ACL is processed.
- Inbound ACL operate as follows:
  - If the information in a packet header and an ACL statement match, the rest of the statements in the list are skipped, and the packet is permitted or denied as specified by the matched statement.
  - If a packet header does not match an ACL statement, the packet is tested against the next statement in the list and this matching process continues until the end of the list is reached.
  - At the end of every ACL is a statement is an implicit deny any statement and because of this statement, an ACL should have at least one permit statement in it; otherwise, the ACL blocks all traffic.



Troubleshoot ACL

Processing Packets with ACL

- Outbound ACL operate as follows:
  - The router checks the routing table to see if the packet is routable.
  - The router checks to see whether the outbound interface is grouped to an ACL.
  - If it is, the ACL is tested by the combination of ACE that are associated with that interface.
  - Based on the ACL tests, the packet is permitted or denied.



Processing Packets with ACL

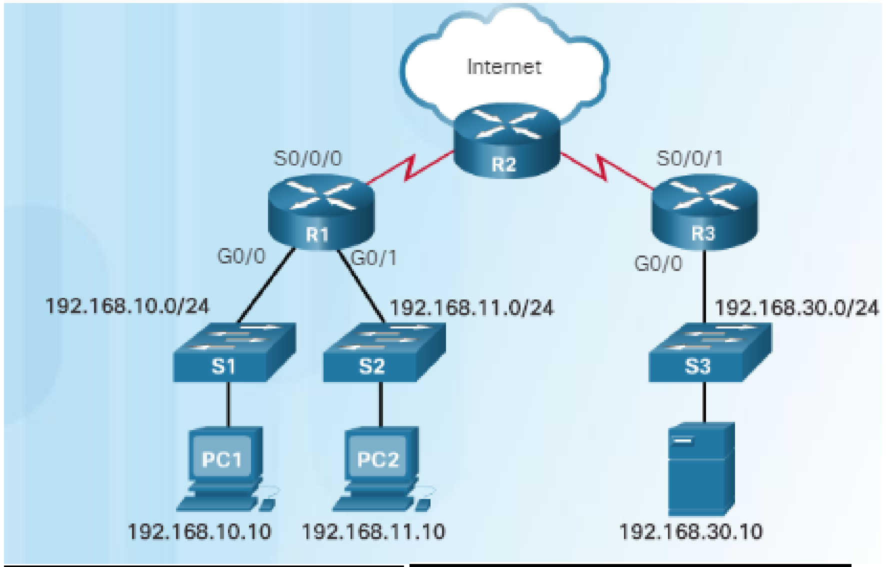
- Standard ACL only examine the source IPv4 address.
  - The destination of the packet and the ports involved are not considered.
- The Cisco IOS software tests addresses against the ACL ACE.
  - The first match determines whether the software accepts or rejects the address.
  - Because the software stops testing conditions after the first match, the order of the conditions is critical.
  - If no conditions match, the address is rejected.

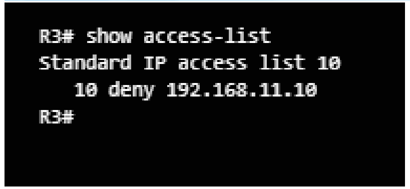
Processing Packets with ACL

- Extended ACL filter on protocol, source address, destination address, and port numbers.
- The ACL first filters on the source address, then on the port and protocol of the source.
- It then filters on the destination address, then on the port and protocol of the destination, and makes a final permit or deny decision.

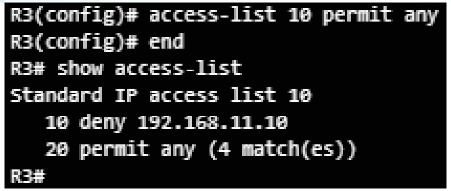
Common IPv4 ACL Errors

Troubleshooting Standard IPv4 ACL – Example 1





```
R3# show access-list
Standard IP access list 10
   10 deny 192.168.11.10
R3#
```

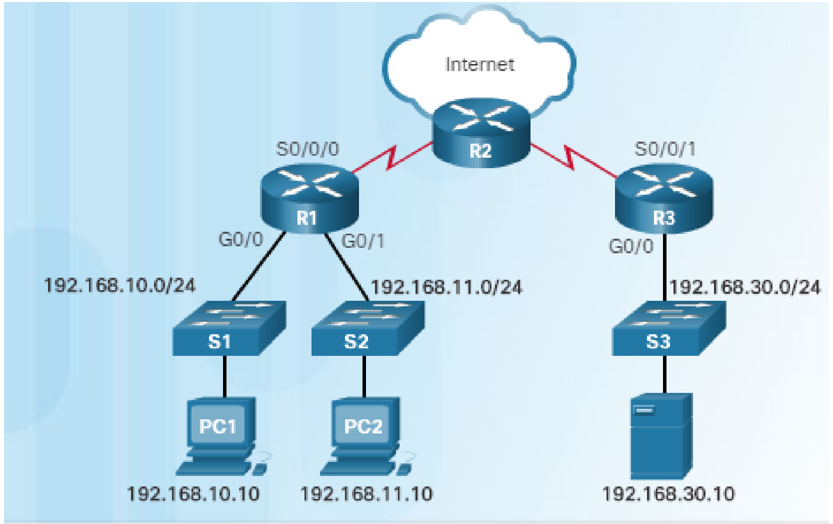


```
R3(config)# access-list 10 permit any
R3(config)# end
R3# show access-list
Standard IP access list 10
   10 deny 192.168.11.10
   20 permit any (4 match(es))
R3#
```

- The most common errors involving ACL:
  - Entering ACE in the wrong order
  - Not specifying adequate ACL rules
  - Applying the ACL using the wrong direction, wrong interface, or wrong source address
- In the figure to the left, PC2 should not be able to access the File Server. However, PC1 can not access it either.
- The output of the show access-list command shows the one deny statement in the ACL.
- The set of commands on the right shows the solution. The permit statement allows other devices to access since the implicit deny was blocking other traffic.

Common IPv4 ACL Errors

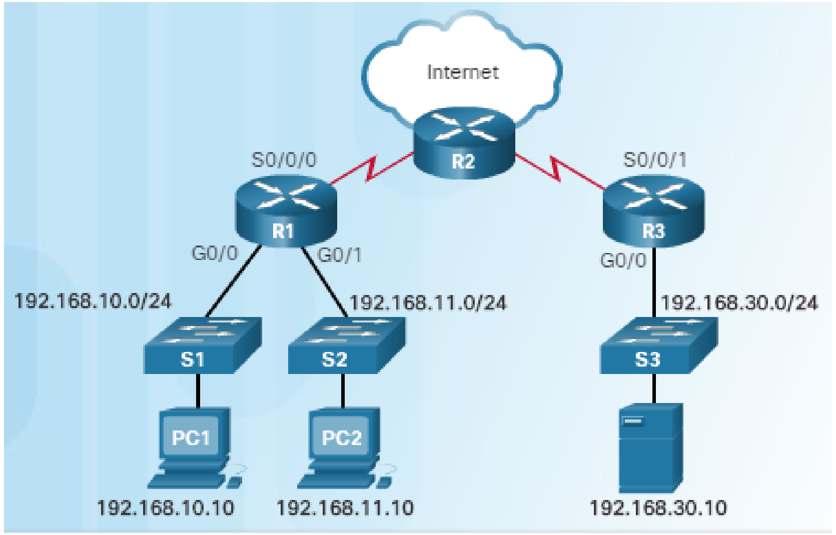
Troubleshooting Standard IPv4 ACL – Example 2

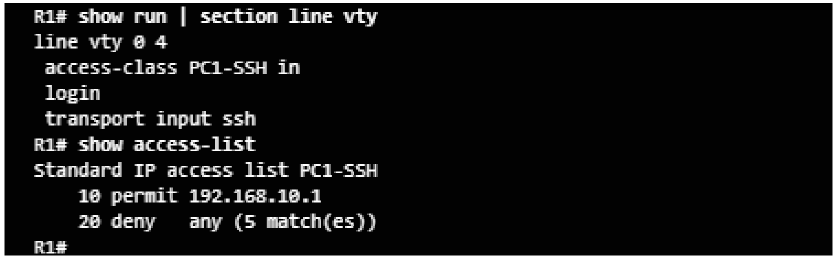


```
R1# show run | section interface
interface GigabitEthernet0/0
ip address 192.168.10.1 255.255.255.0
duplex auto
speed auto
interface GigabitEthernet0/1
ip address 192.168.11.1 255.255.255.0
ip access-group 20 in
duplex auto
speed auto
<output ommitted>
```

- The 192.168.11.0/24 network should not be able to access the 192.168.10.0/24 network.
- PC2 cannot access PC1 as planned, however, it also cannot access the Internet through R2.
- Problem: access-list 20 was applied to G0/1 on an inbound direction
- Where should ACL 20 be applied and in which direction?
- In order for PC2 to access the Internet, ACL 20 needs to be removed from the G0/1 interface and applied outbound on the G0/0 interface.

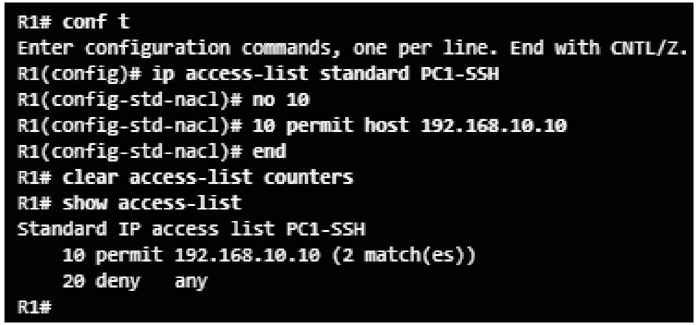
Troubleshooting Standard IPv4 ACL – Example 3





```
R1# show run | section line vty
line vty 0 4
 access-class PC1-SSH in
 login
 transport input ssh
R1# show access-list
Standard IP access list PC1-SSH
    10 permit 192.168.10.1
    20 deny   any (5 match(es))
R1#
```

- Only PC1 should be allowed to SSH to R1.
- There is a problem with the config in the figure to the left since PC1 is unable to SSH to R1.
- The ACL is permitting the 192.168.10.1 address which is the G0/0 interface. However, the address that should be permitted is the PC1 host address of 192.168.10.10.
- The solution is provided below:

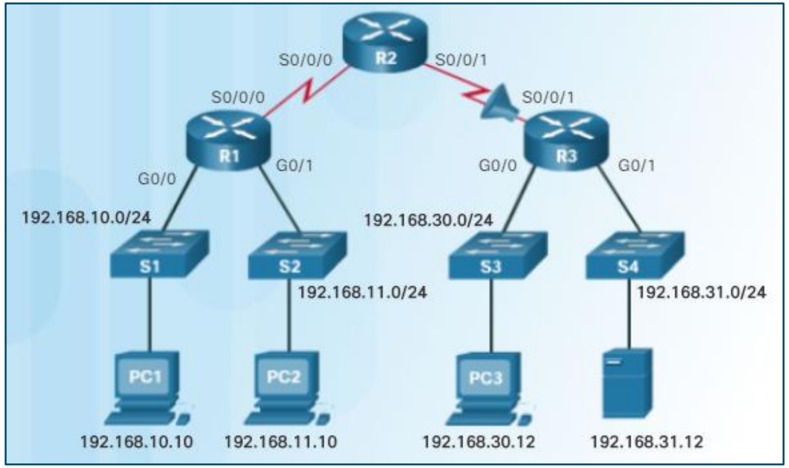


```
R1# conf t
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)# ip access-list standard PC1-SSH
R1(config-std-nacl)# no 10
R1(config-std-nacl)# 10 permit host 192.168.10.10
R1(config-std-nacl)# end
R1# clear access-list counters
R1# show access-list
Standard IP access list PC1-SSH
    10 permit 192.168.10.10 (2 match(es))
    20 deny   any
R1#
```

Troubleshooting Extended IPv4 ACL – Example 1

- In this example, host 192.168.10.10 has no Telnet connectivity with 192.168.30.12.
  - The **show access-lists** command displays matches for the first deny statement indicating that this ACE has been matched by traffic.

- **Solution:**
  - Host 192.168.10.10 has no connectivity with 192.168.30.12 because statement 10 denies host 192.168.10.10, therefore statement 20 can never be matched.
  - Statements 10 and 20 should be reversed.

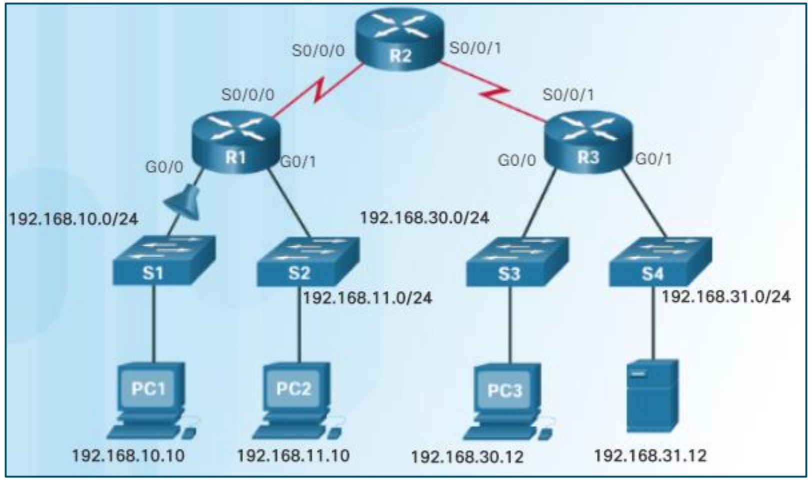


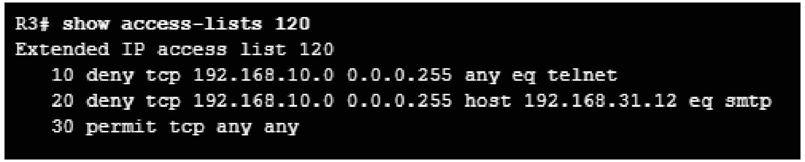
```
R3# show access-lists
Extended IP access list 110
   10 deny tcp 192.168.10.0 0.0.0.255 any (12 match(es))
   20 permit tcp 192.168.10.0 0.0.0.255 any eq telnet
   30 permit ip any any
```

Troubleshooting Extended IPv4 ACL – Example 2

- In this example, the 192.168.10.0/24 network cannot use TFTP to connect to the 192.168.30.0/24 network.

- **Solution:**
  - Statement 30 in access list 120 allows all TCP traffic.
  - However, TFTP uses UDP instead of TCP and therefore it is implicitly denied.
  - Statement 30 should be **permit ip any any**.

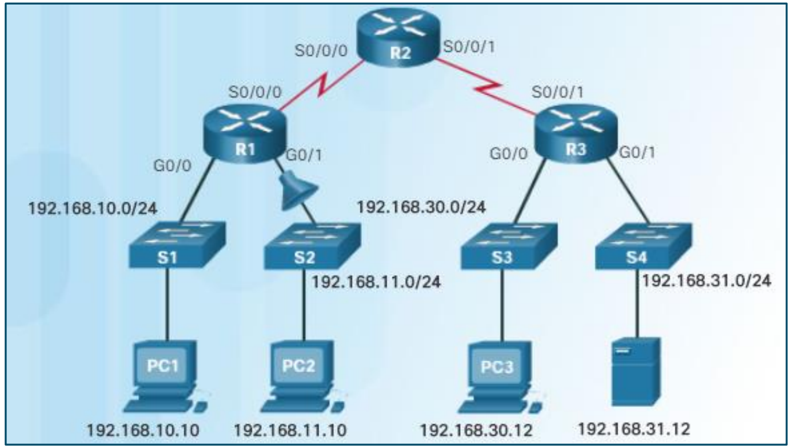


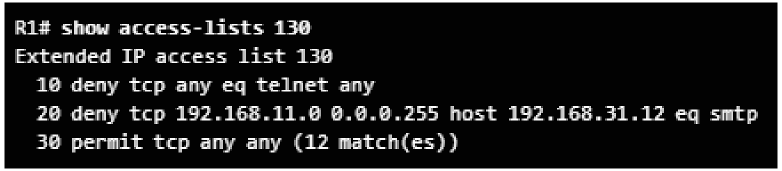


```
R3# show access-lists 120
Extended IP access list 120
    10 deny tcp 192.168.10.0 0.0.0.255 any eq telnet
    20 deny tcp 192.168.10.0 0.0.0.255 host 192.168.31.12 eq smtp
    30 permit tcp any any
```

Troubleshooting Extended IPv4 ACL – Example 3

- In this example, the 192.168.11.0/24 network can use Telnet to connect to 192.168.30.0/24, but according to company policy, this connection should not be allowed.
- The results of the **show access-lists 130** command indicate that the permit statement has been matched.



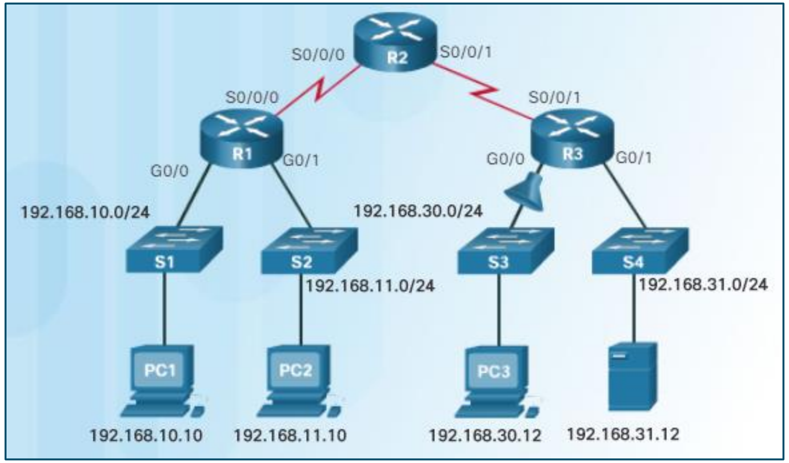


```
R1# show access-lists 130
Extended IP access list 130
  10 deny tcp any eq telnet any
  20 deny tcp 192.168.11.0 0.0.0.255 host 192.168.31.12 eq smtp
  30 permit tcp any any (12 match(es))
```

- **Solution:**
  - The Telnet port number in statement 10 of ACL 130 is listed in the wrong order as it currently denies any source packet with a port number equal to Telnet.
  - Configure **10 deny tcp 192.168.11.0 0.0.0.255 192.168.30.0 0.0.0.255 eq telnet**.

Troubleshooting Extended IPv4 ACL – Example 4

- In this example, host 192.168.30.12 is able to Telnet to connect to 192.168.31.12, but company policy states that this connection should not be allowed.
- Output from the **show access-lists 140** command indicate that the permit statement has been matched.
- **Solution:**
  - Host 192.168.30.12 can use Telnet to connect to 192.168.31.12 because there are no rules that deny host 192.168.30.12 or its network as the source.
  - Statement 10 of access list 140 denies the router interface on which traffic enters the router.
  - The host IPv4 address in statement 10 should be 192.168.30.12.

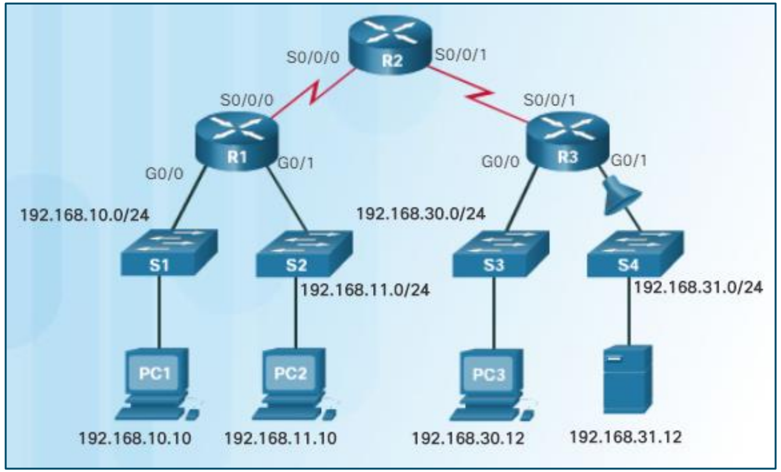


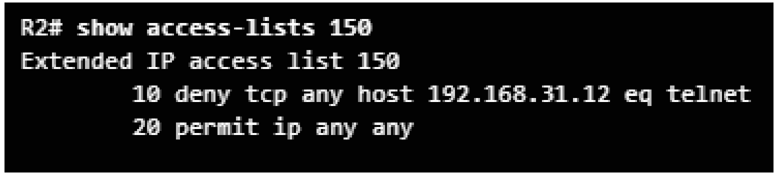
```
R3# show access-lists 140
Extended IP access list 140
   10 deny tcp host 192.168.30.1 any eq telnet
   20 permit ip any any (5 match(es))
```

Common IPv4 ACL Errors

Troubleshooting Extended IPv4 ACL – Example 5

- In this example, host 192.168.30.12 can use Telnet to connect to 192.168.31.12, but according to the security policy, this connection should not be allowed.
- Output from the **show access-lists 150** command indicate that no matches have occurred for the deny statement as expected.
- **Solution:**
  - Host 192.168.30.12 can use Telnet to connect to 192.168.31.12 because of the direction in which access list 150 is applied to the G0/1 interface.
  - Statement 10 denies any source address to connect to host 192.168.31.12 using Telnet.
  - However, this filter should be applied outbound on G0/1 to filter correctly.




```
R2# show access-lists 150
Extended IP access list 150
        10 deny tcp any host 192.168.31.12 eq telnet
        20 permit ip any any
```

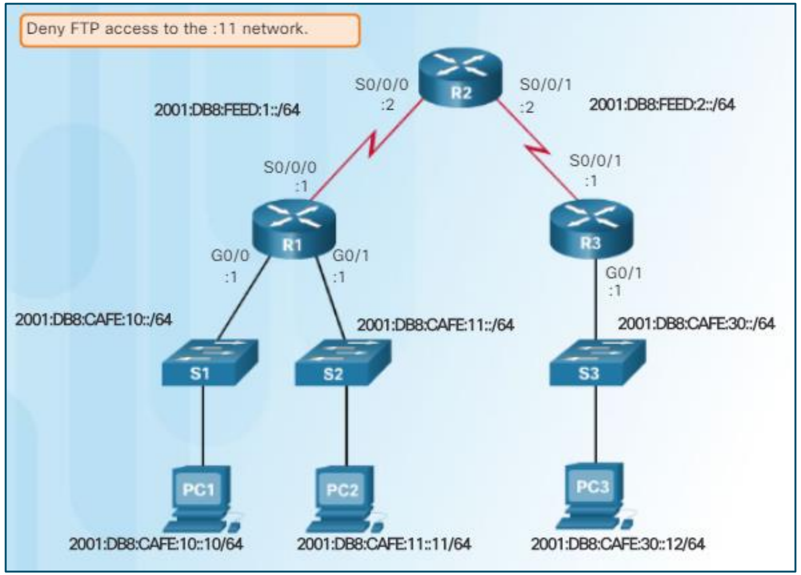
Troubleshoot IPv6 ACL

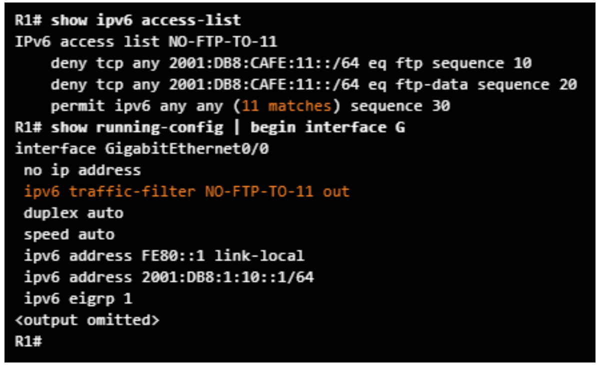
Common IPv6 ACL Errors

Troubleshooting Extended IPv6 ACL – Example 1

- In this example, R1 is configured with an IPv6 ACL to deny FTP access from the :10 network to the :11 network.
  - However, after configuring the ACL, PC1 is still able to connect to the FTP server running on PC2.
  - The output of the **show ipv6 access-list** command displays matches for the permit statement but not the deny statements.

- **Solution:**
  - The ACL was applied using the correct name, but not the correct direction.
  - To correct the issue, remove the **ipv6 traffic-filter NO-FTP-TO-11 out** and replace it with **ipv6 traffic-filter NO-FTP-TO-11 in**.

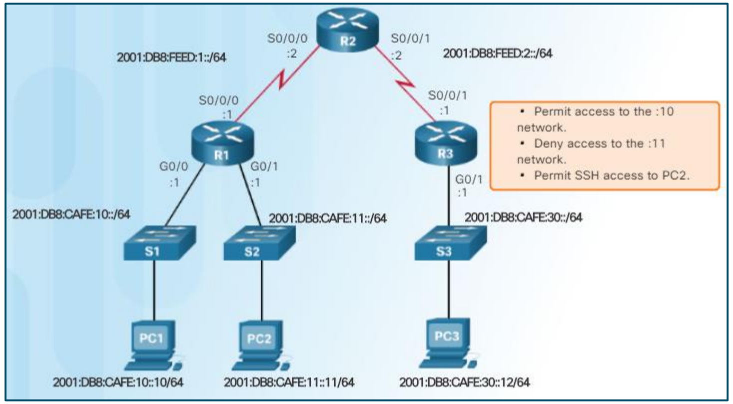


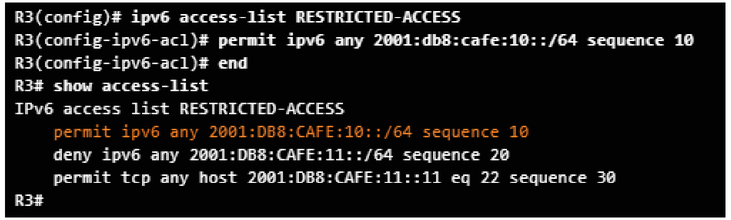

```
R1# show ipv6 access-list
IPv6 access list NO-FTP-TO-11
    deny tcp any 2001:DB8:CAFE:11::/64 eq ftp sequence 10
    deny tcp any 2001:DB8:CAFE:11::/64 eq ftp-data sequence 20
    permit ipv6 any any (11 matches) sequence 30
R1# show running-config | begin interface G
interface GigabitEthernet0/0
 no ip address
 ipv6 traffic-filter NO-FTP-TO-11 out
 duplex auto
 speed auto
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:1:10::1/64
 ipv6 eigrp 1
<output omitted>
R1#
```

Common IPv6 ACL Errors

Troubleshooting Extended IPv6 ACL – Example 2

- In this example, R3 is configured with an IPv6 ACL named RESTRICTED-ACCESS that should permit access to the :10 network, deny access to the :11 network, and permit SSH access to the PC at 2001:DB8:CAFE:11::11
- After configuring the ACL, PC3 cannot reach the 10 or 11 network, and cannot SSH to 2001:DB8:CAFE:11::11.
- **Solution:**
  - The first permit statement should allow access to the :10 network but only access to the 2001:DB8:CAFE:10:: host is allowed.
  - To correct this issue, remove the host argument and change the prefix to /64. You can do this without removing the ACL by replacing the ACE using the sequence number 10.
  - The second error in the ACL is the order of the next two statements therefore remove the statements first, and then enter them in the correct order.

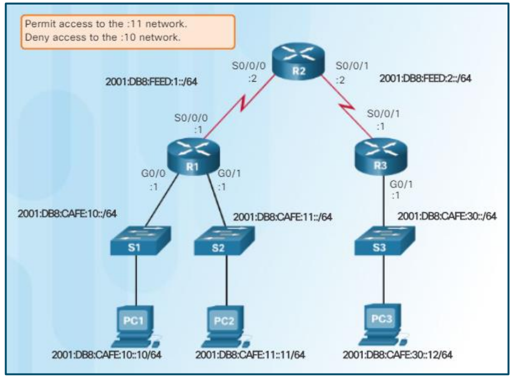



```
R3(config)# ipv6 access-list RESTRICTED-ACCESS
R3(config-ipv6-acl)# permit ipv6 any 2001:db8:cafe:10::/64 sequence 10
R3(config-ipv6-acl)# end
R3# show access-list
IPv6 access list RESTRICTED-ACCESS
    permit ipv6 any 2001:DB8:CAFE:10::/64 sequence 10
    deny ipv6 any 2001:DB8:CAFE:11::/64 sequence 20
    permit tcp any host 2001:DB8:CAFE:11::11 eq 22 sequence 30
R3#
```

Common IPv4 ACL Errors

Troubleshooting Extended IPv6 ACL – Example 3

- In this example, R1 is configured with an IPv6 ACL named DENY-ACCESS that should permit access to the :11 network from the :30 network, but deny access to the :10 network.
  - The DENY-ACCESS ACL is supposed to permit access to the :11 network from the :30 network while denying access to the :10 network.
  - However, after applying the ACL to the interface the :10 network is still reachable from the :30 network.
- **Solution:**
  - The problem is with the location of the ACL and should be applied closest to the source of the traffic.
  - Remove the ACL on R1 and apply the ACL on R3.



```
R1# show access-list
IPv6 access list DENY-ACCESS
    permit ipv6 any 2001:DB8:CAFE:11::/64 sequence 10
    deny ipv6 any 2001:DB8:CAFE:10::/64 sequence 20
R1# show running-config | section interface GigabitEthernet0/1
interface GigabitEthernet0/1
 no ip address
 duplex auto
 speed auto
 ipv6 address FE80::1 link-local
 ipv6 address 2001:DB8:CAFE:11::1/64
 ipv6 eigrp 1
 ipv6 traffic-filter DENY-ACCESS out
R1#
```

```
R1(config)# no ipv6 access-list DENY-ACCESS
R1(config)# interface g0/1
R1(config-if)# no ipv6 traffic-filter DENY-ACCESS out
R1(config-if)#
!----------------------------------------------------
R3(config)# ipv6 access-list DENY-ACCESS
R3(config-ipv6-acl)# permit ipv6 any 2001:DB8:CAFE:11::/64
R3(config-ipv6-acl)# deny ipv6 any 2001:DB8:CAFE:10::/64
R3(config-ipv6-acl)# exit
R3(config)# interface g0/0
R3(config-if)# ipv6 traffic-filter DENY-ACCESS in
R3(config-if)#
```

Chapter Summary

Conclusion

Chapter 2: Access Control Lists

- By default a router does not filter traffic. Traffic that enters the router is routed solely based on information within the routing table.
- An ACL is a sequential list of permit or deny statements. The last statement of an ACL is always an implicit deny any statement which blocks all traffic. To prevent the implied deny any statement at the end of the ACL from blocking all traffic, the **permit ip any any** statement can be added.
- When network traffic passes through an interface configured with an ACL, the router compares the information within the packet against each entry, in sequential order, to determine if the packet matches one of the statements. If a match is found, the packet is processed accordingly.
- ACL can be applied to inbound traffic or to outbound traffic.
- Standard ACL can be used to permit or deny traffic only from a source IPv4 addresses. The basic rule for placing a standard ACL is to place it close to the destination.
- Extended ACL filter packets based on several attributes: protocol type, source or destination IPv4 address, and source or destination ports. The basic rule for placing an extended ACL is to place it as close to the source as possible.

Conclusion

Chapter 2: Access Control Lists (Cont.)

- The **access-list** global configuration command defines a standard ACL with a number in the range of 1 through 99 or an extended ACL with numbers in the range of 100 to 199. The **ip access-list standard** *name* is used to create a standard named ACL, whereas the command **ip access-list extended** *name* is for an extended access list.
- After an ACL is configured, it is linked to an interface using the **ip access-group** command in interface configuration mode. A device an only have one ACL per protocol, per direction, per interface.
- 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.
- The **show running-config** and **show access-lists** commands are used to verify ACL configuration. The **show ip interface** command is used to verify the ACL on the interface and the direction in which it was applied.
- The **access-class** command configured in line configuration mode is used to link an ACL to a particular VTY line.

Conclusion

Chapter 2: Access Control Lists(Cont.)

- From global configuration mode, use the **ipv6 access-list** *name* command to create an IPv6 ACL. Unlike IPv4 ACL, IPv6 ACL do not use wildcard masks. Instead, the prefix-length is used to indicate how much of an IPv6 source or destination address should be matched.
- After an IPv6 ACL is configured, it is linked to an interface using the **ipv6 traffic-filter** command.
- Unlike IPv4, IPv6 ACL do not have support for a standard or extended option.

Chapter 2

New Terms and Commands

- access control lists (ACL)
- firewalls
- access control entries (ACE)
- packet filtering
- Standard ACL
- Extended ACL
- implicit deny
- Inbound ACL
- Outbound ACL
- wildcard masks
- named ACL
- inverse mask