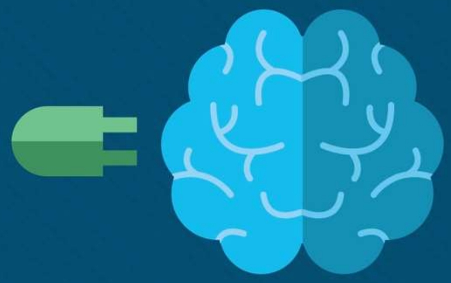
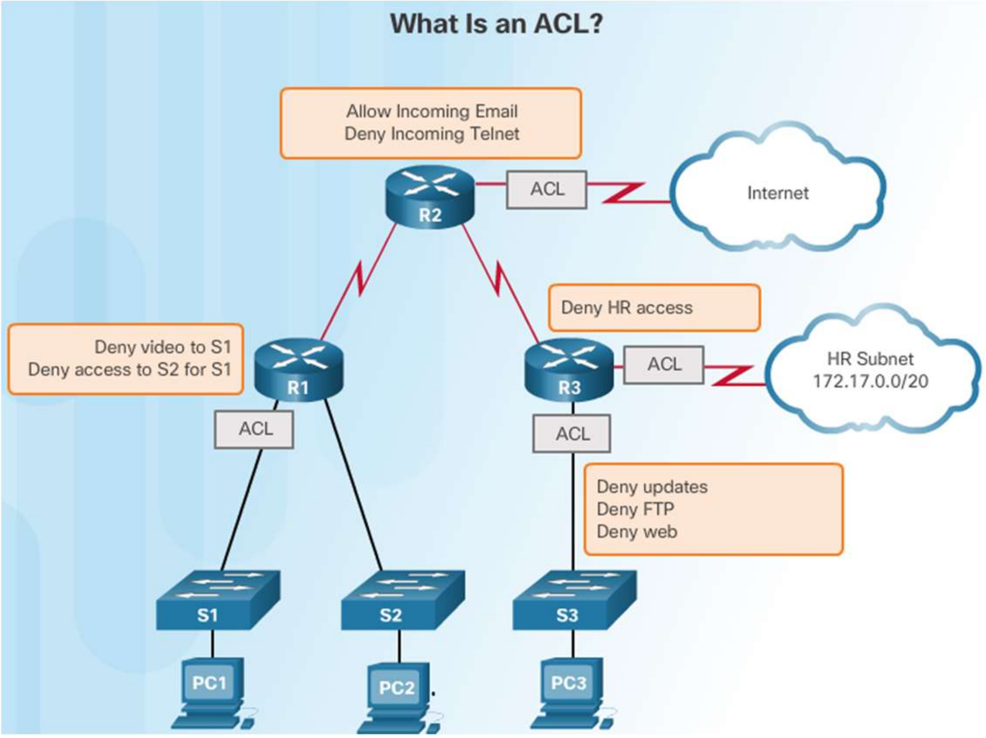
Access Control Lists



ACL Operation

Purpose of ACLs

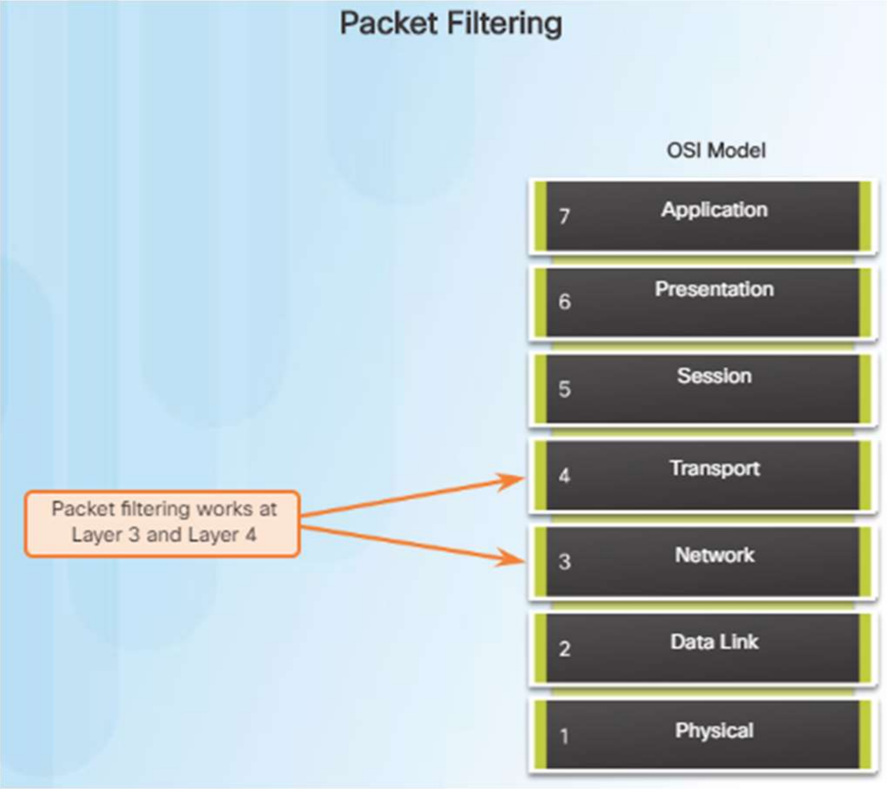
What is an ACL?



- 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. ACLs are not configured by default on a router.
- ACL's 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. ACLs can help verify routing updates are from a known source.
  - ACLs 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 ACLs

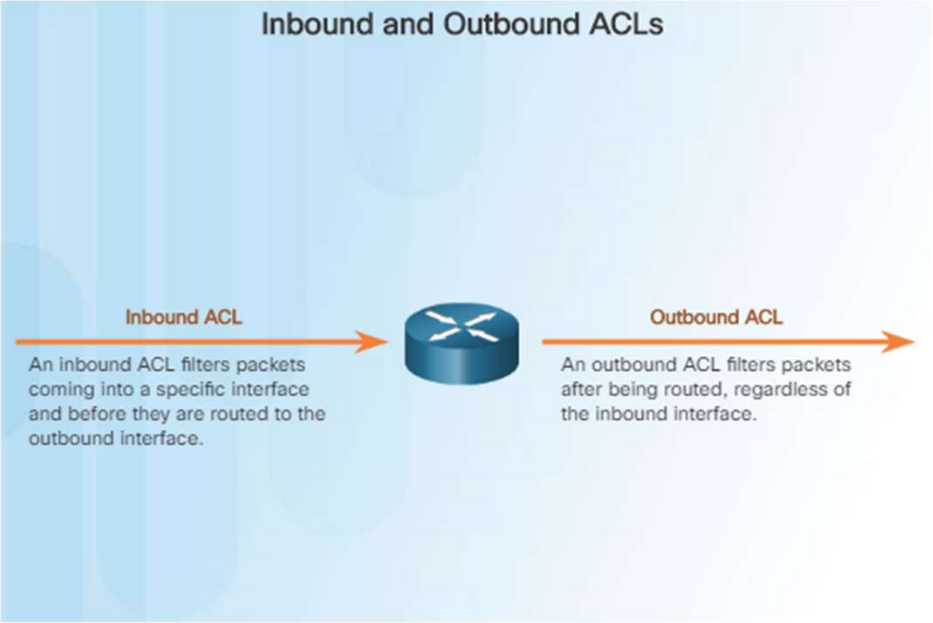
Packet Filtering



- An ACL is a sequential list of permit or deny statements, known as access control entries (ACEs).
  - ACEs 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 ACEs. This is referred to as packet filtering.
- Packet Filtering:
  - 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 ACLs should have at least one permit statement.

Purpose of ACLs

ACL Operation



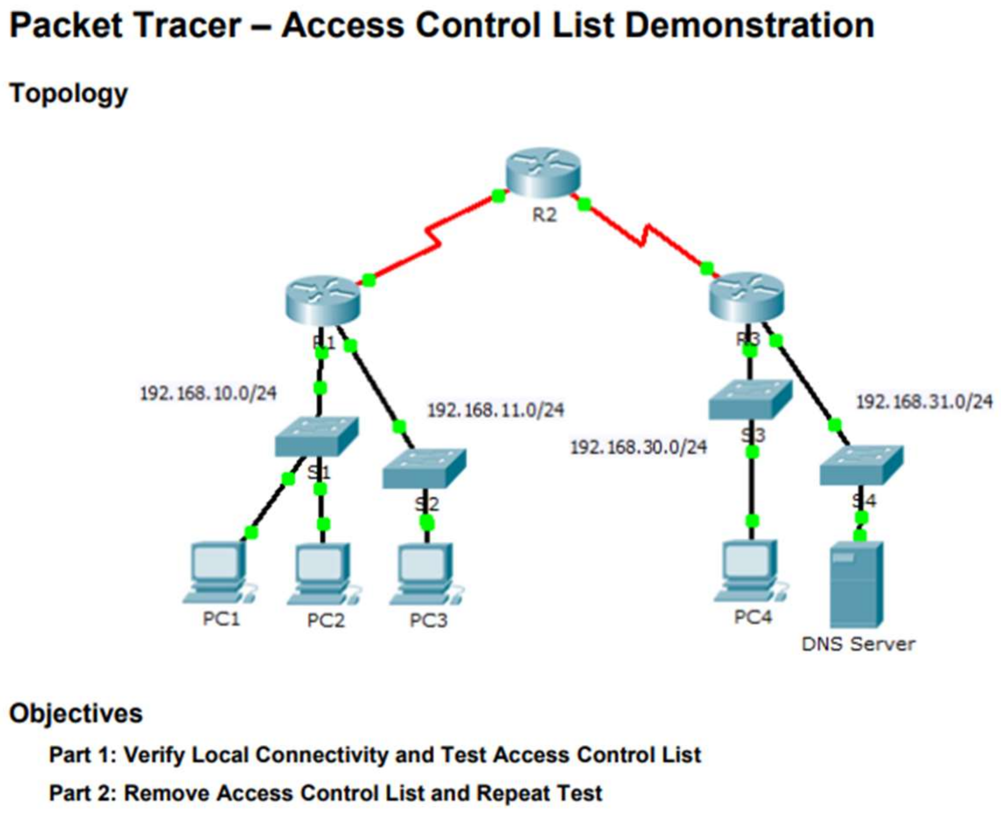
An inbound ACL filters packets coming into a specific interface and before they are routed to the outbound interface.

An outbound ACL filters packets after being routed, regardless of the inbound interface.

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

Purpose of ACLs

Packet Tracer – ACL Demonstration



Packet Tracer – Access Control List Demonstration

Topology

R2
R1
R3
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
DNS Server

Objectives

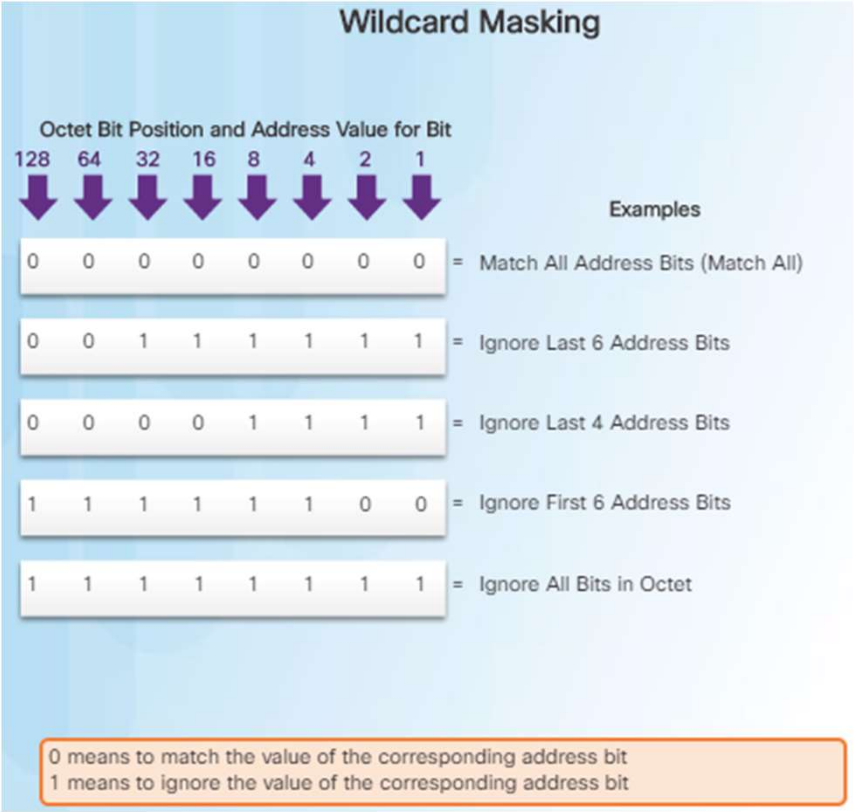
Part 1: Verify Local Connectivity and Test Access Control List

Part 2: Remove Access Control List and Repeat Test

- In this Packet Tracer activity, you will observe how an ACL can be used to prevent a ping from reaching hosts on a network.
- After removing the ACL from the configuration, the pings will be successful.

Wildcard Masks in ACLs

Introducing ACL Wildcard Masking



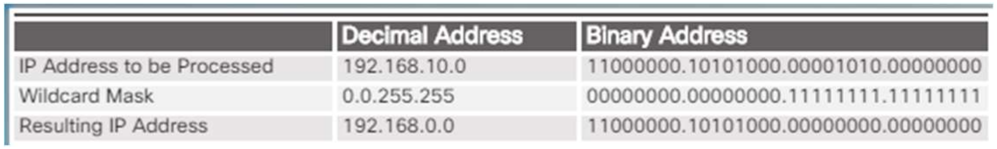
Wildcard Masking

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

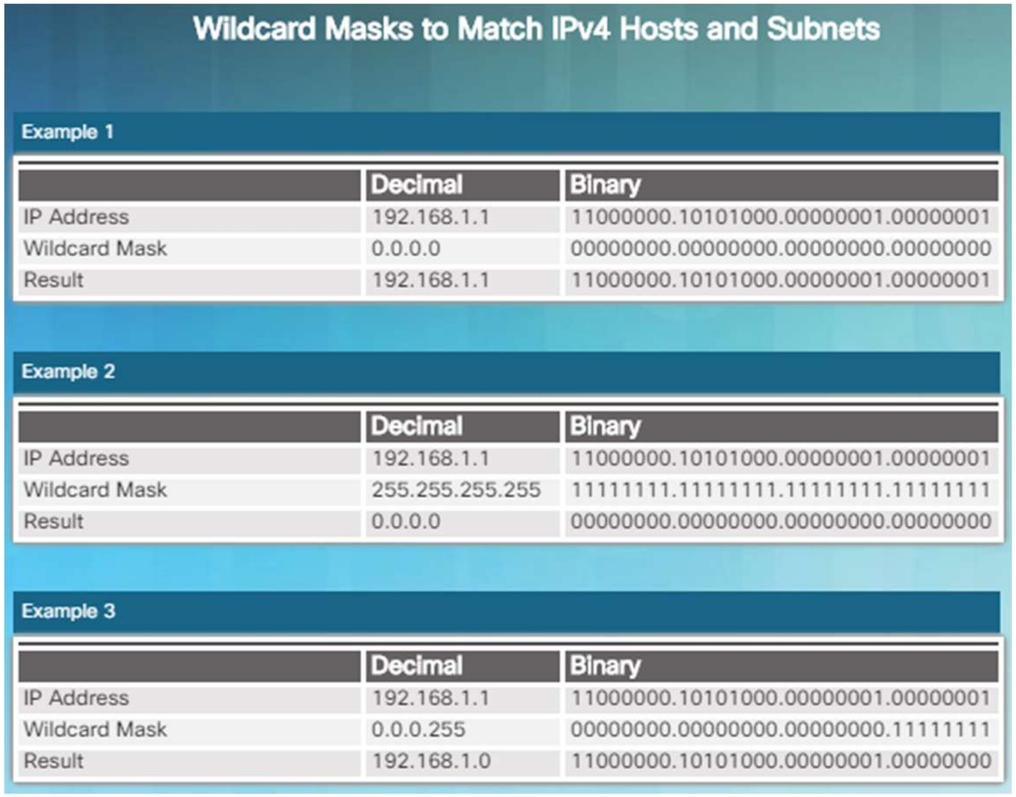
- IPv4 ACEs require the use of wildcard masks.
- A wildcard mask is a string of 32 binary digits (1s and 0s) used by the router to determine which bits of the address to examine for a match.
- Wildcard masks are often referred to as an inverse mask since unlike a subnet mask where a binary 1 is a match,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

Wildcard Mask Examples



Wildcard Masks to Match IPv4 Hosts and Subnets

Example 1

	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 2

	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

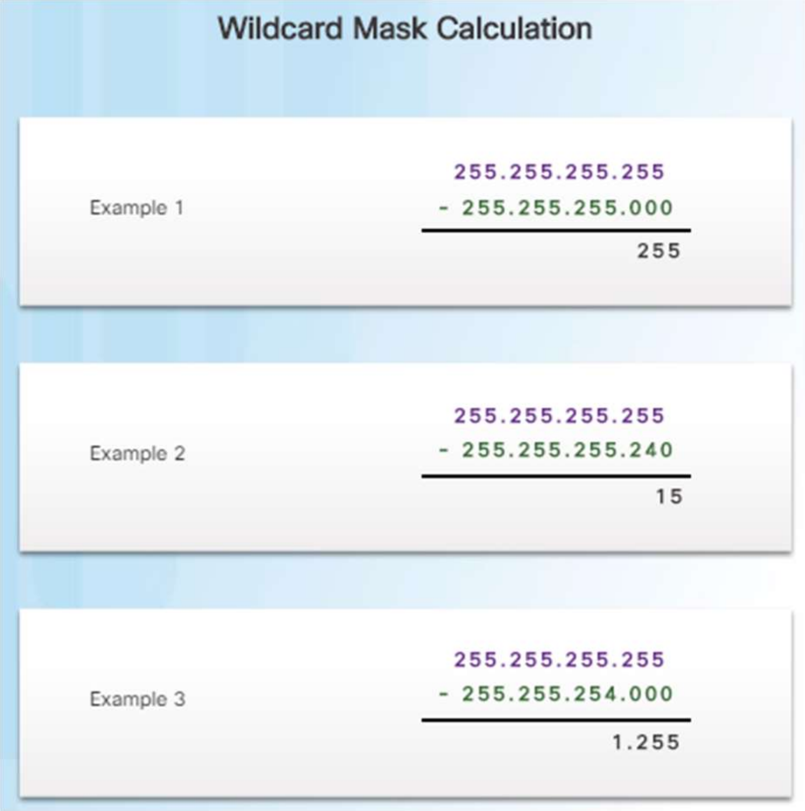
Example 3

	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

- Calculating the wildcard mask to match IPV4 subnets takes practice. In the first to the left:
  - Example 1: The wildcard mask stipulates that every bit in the IPv4 192.168.1.1 address must match exactly.
  - Example 2: The wildcard mask stipulates that anything will match.
  - Example 3: The wildcard mask stipulates that any host within the 192.168.1.0/24 network will match.

Wildcard Masks in ACLs

Calculating the Wildcard Mask



Wildcard Mask Calculation

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
```

- Calculating wildcard mask examples:
  - 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 maks 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.

Match the correct wildcard to the ACE

- 0.0.0.0
- 0.255.255.255
- 0.0.0.255
- 0.0.0.31
- 0.0.255.255

a Denegar todos los host de la red 10.10.10.0/24

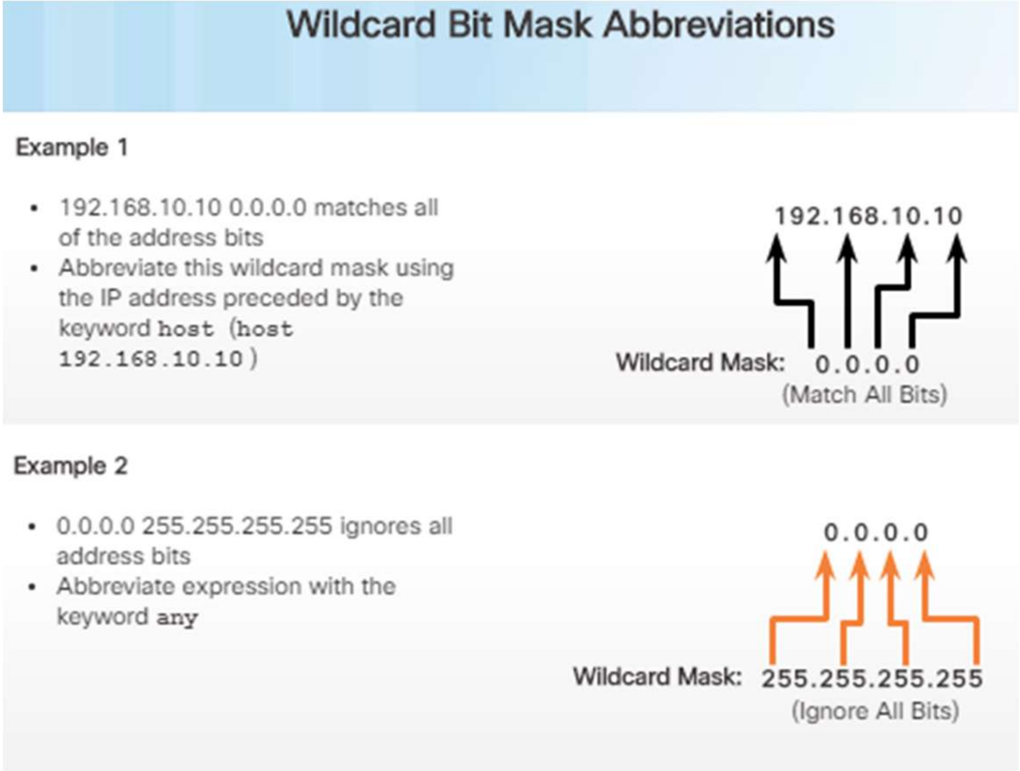
b Denegar todos los hosts de la red 172.18.0.0/16

c Denegar todos los hosts de la subred 192.168.5.0/27

d Permitir todos los hosts de la red 10.0.0.0/8

e Denegar el host 192.168.5.7

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`)

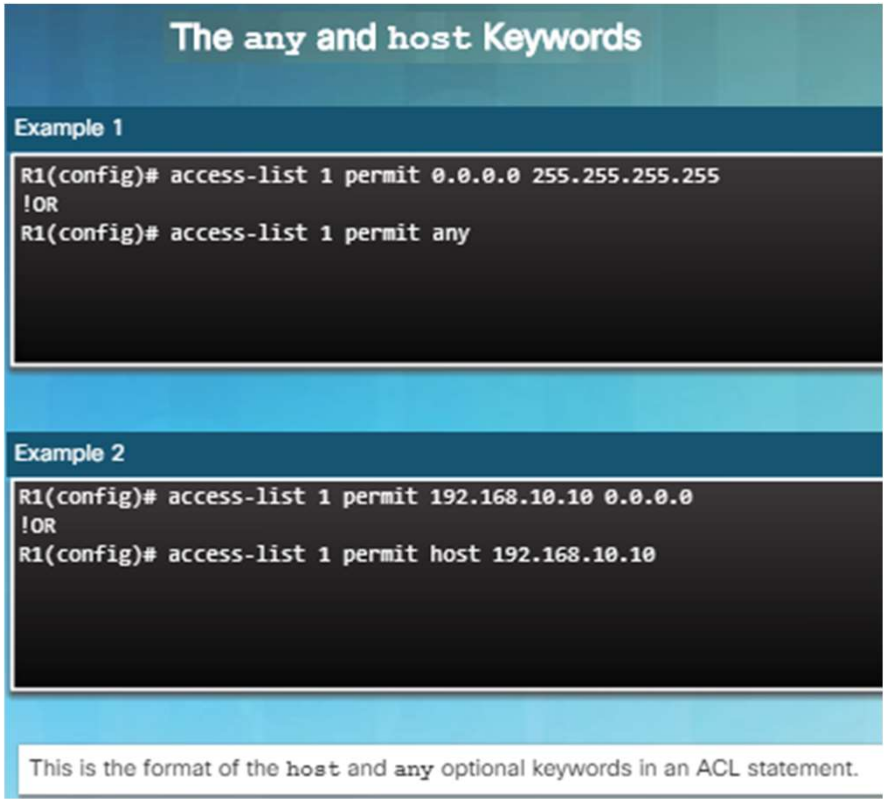
Example 2

- 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 ACLs

Wildcard Mask Keyword Examples



The any and host Keywords

Example 1

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

Example 2

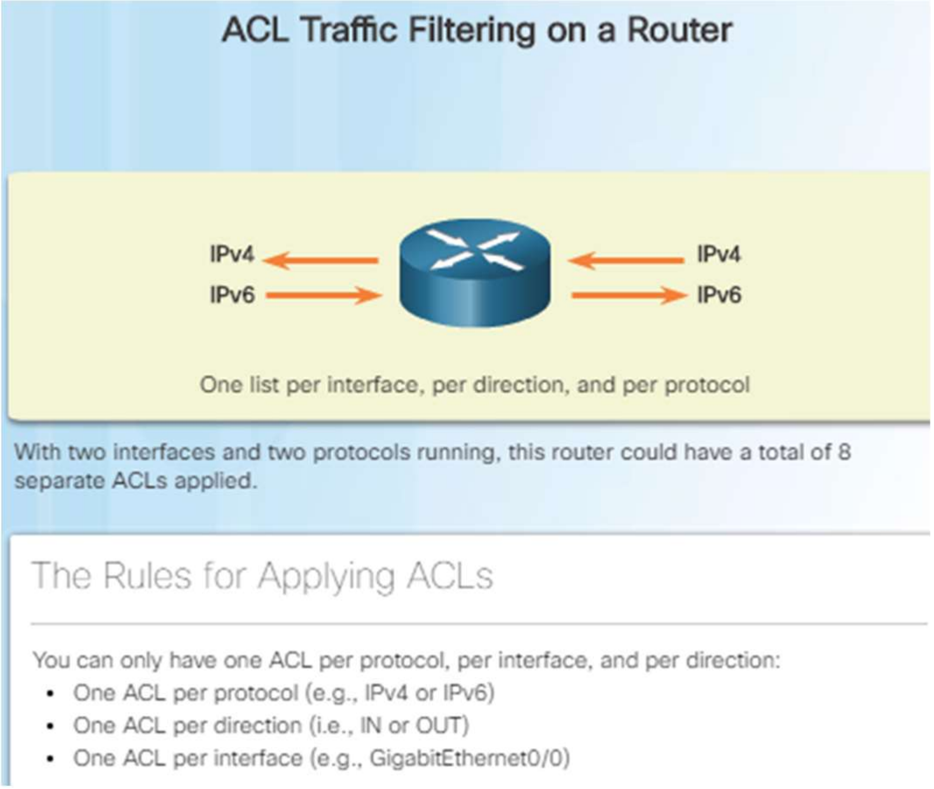
```
R1(config)# access-list 1 permit 192.168.10.10 0.0.0.0
!OR
R1(config)# access-list 1 permit host 192.168.10.10
```

This is the format of the host and any optional keywords in an ACL statement.

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

Guidelines for ACL Creation

General Guidelines for Creating ACLs



ACL Traffic Filtering on a Router

One list per interface, per direction, and per protocol

With two interfaces and two protocols running, this router could have a total of 8 separate ACLs applied.

The Rules for Applying ACLs

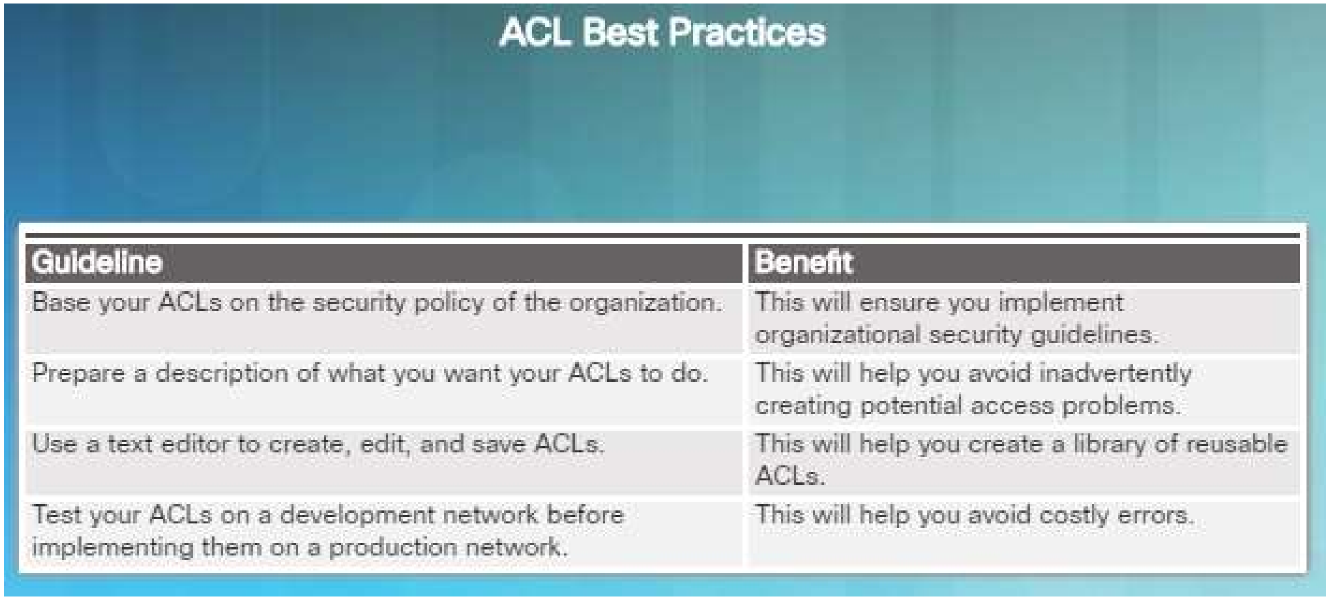
You can only have one ACL per protocol, per interface, and per direction:

- 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)

- 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 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 ACLs for each network protocol configured on the border router interfaces.

Guidelines for ACL Creation

ACL Best Practices



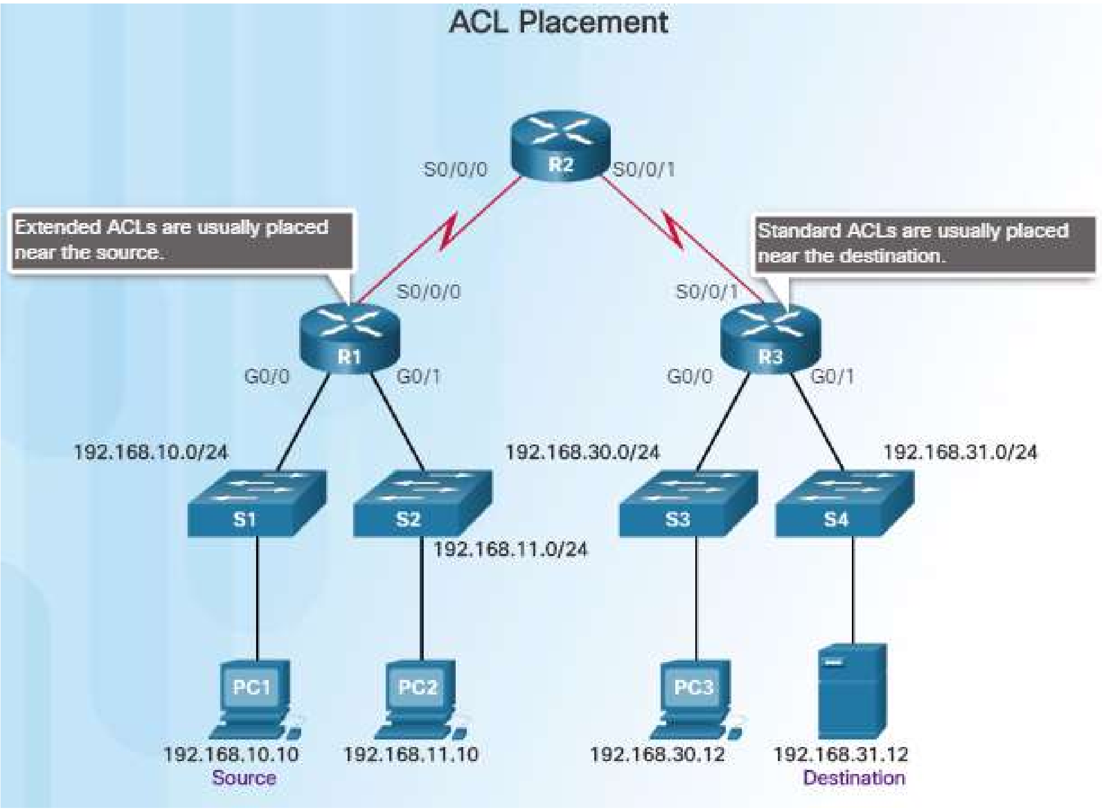
ACL Best Practices

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.

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

Guidelines for ACL Creation

General Guidelines for Creating ACLs



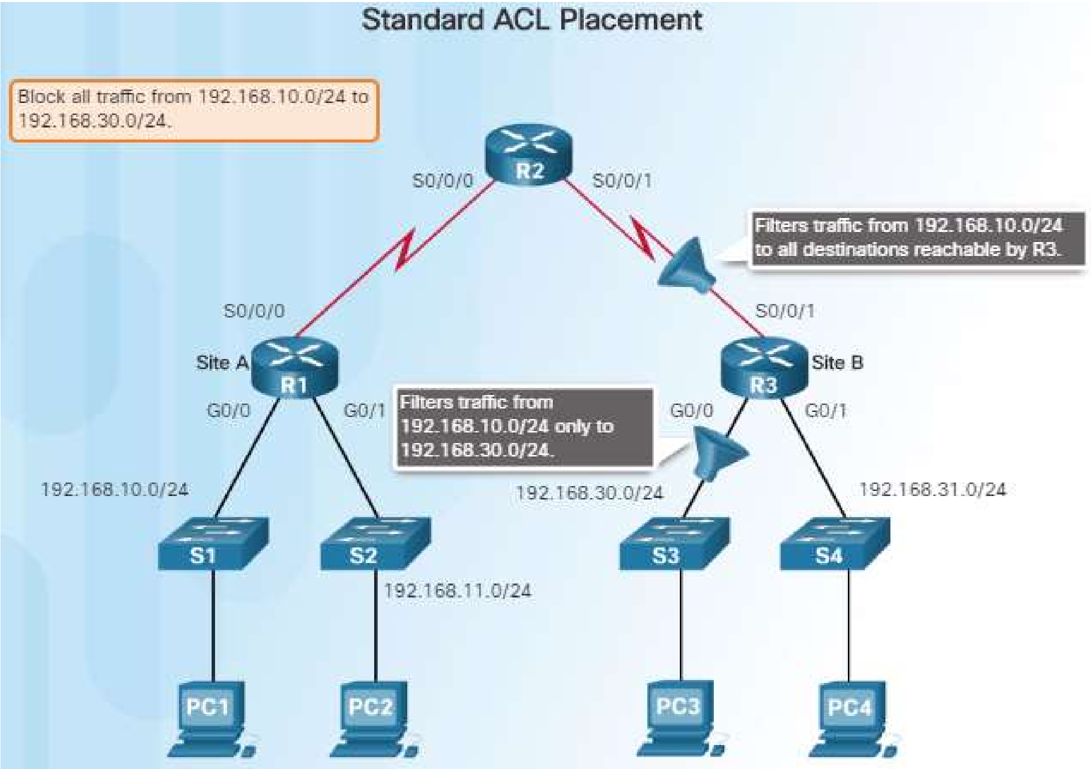
Extended ACLs are usually placed near the source.

Standard ACLs are usually placed near the destination.

- 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 ACLs – Configure extended ACLs 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 ACLs – Since standard ACLs do not specify destination addresses, they should be configured as close to the destination as possible.

Guidelines for ACL Creation

Standard ACL Placement

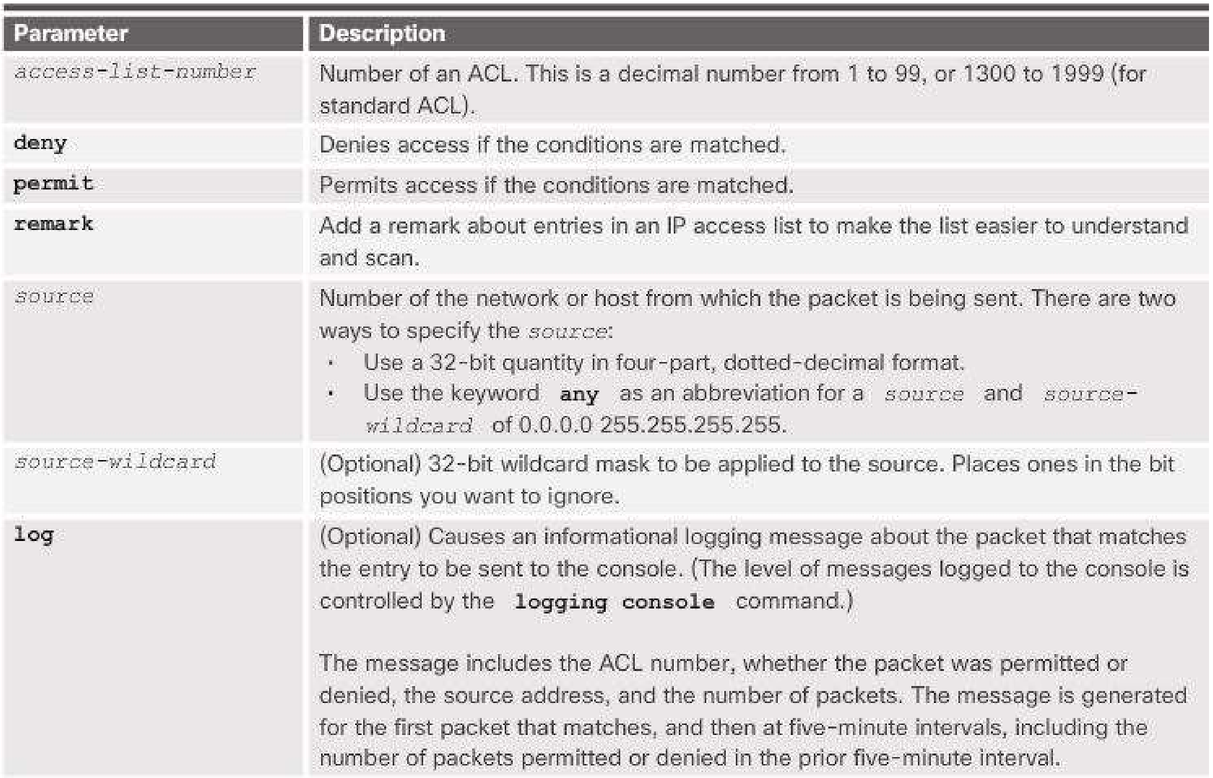

Standard ACL Placement
Block all traffic from 192.168.10.0/24 to 192.168.30.0/24.
R2
S0/0/0
S0/0/1
Filters traffic from 192.168.10.0/24 to all destinations reachable by R3.
S0/0/0
S0/0/1
Site A
R1
Site B
R3
G0/0
G0/1
Filters traffic from 192.168.10.0/24 only to 192.168.30.0/24.
G0/0
G0/1
192.168.10.0/24
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.

Standard IPv4 ACLs

Configure Standard IPv4 ACLs

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.

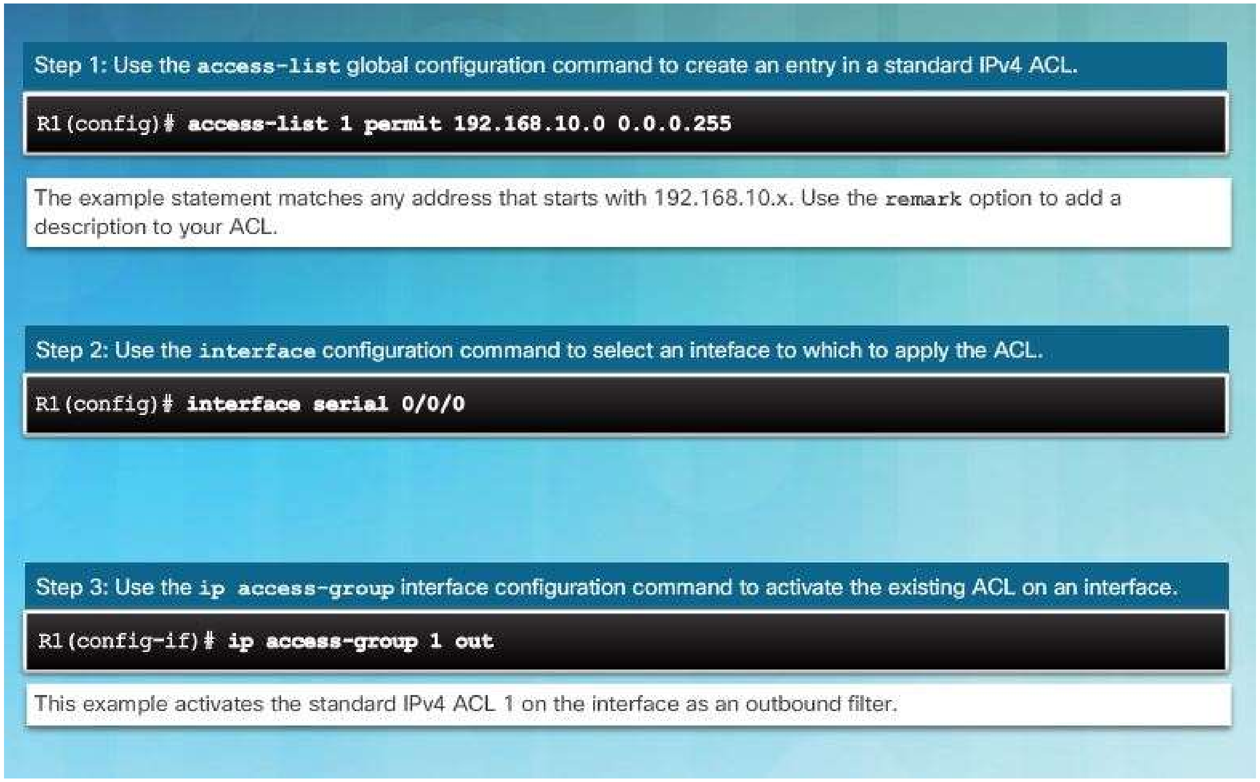
- The **access-list** global configuration command defines a standard ACL with a number in the range of 1 through 99.
- The full syntax of the standard ACL command is as follows:

  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 ACLs

Applying Standard IPv4 ACLs to Interfaces



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.

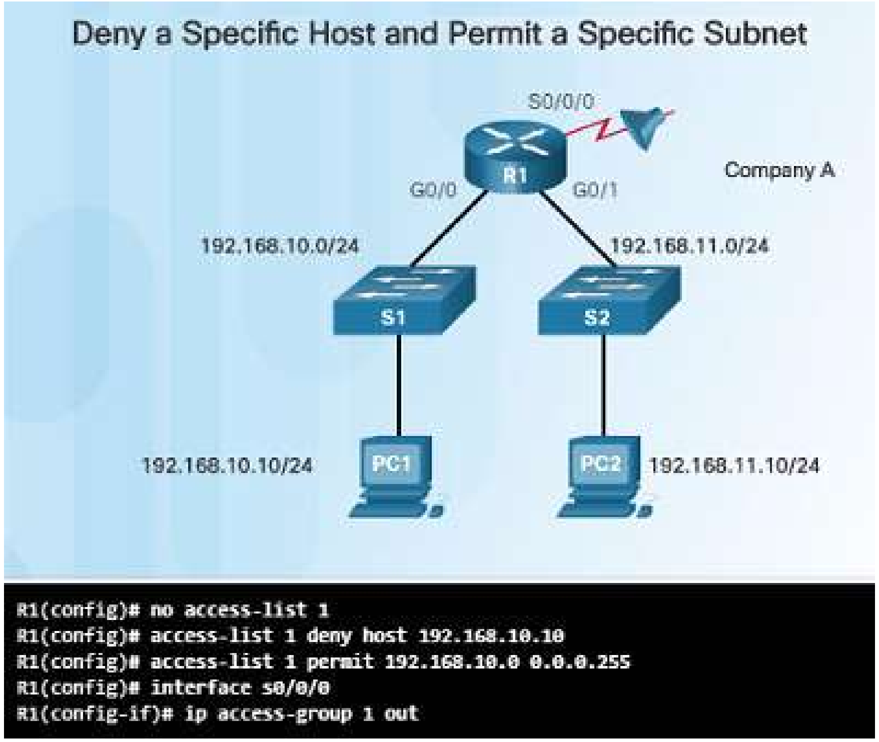
- After a standard IPv4 ACL is configured, it is linked to an interface using the **ip access-group** command in interface configuration mode:

  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 ACLs

Numbered Standard IPv4 ACL Examples

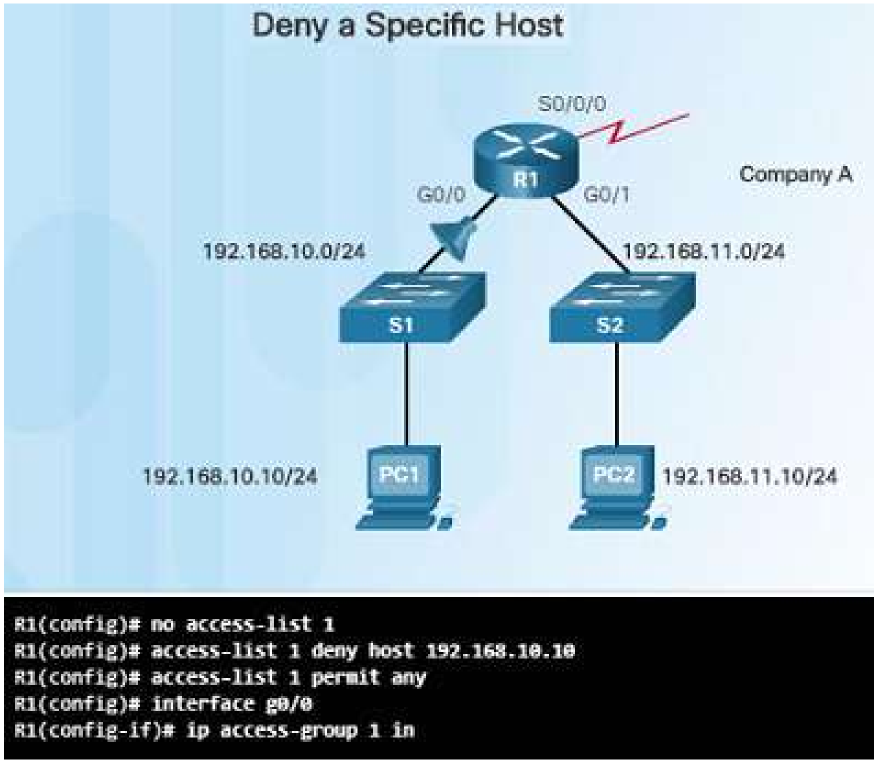


```
R1(config)# no access-list 1
R1(config)# access-list 1 deny host 192.168.10.10
R1(config)# access-list 1 permit 192.168.10.0 0.0.0.255
R1(config)# interface s0/0/0
R1(config-if)# ip access-group 1 out
```

- 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.

Configure Standard IPv4 ACLs

Numbered Standard IPv4 ACL Examples (Cont.)

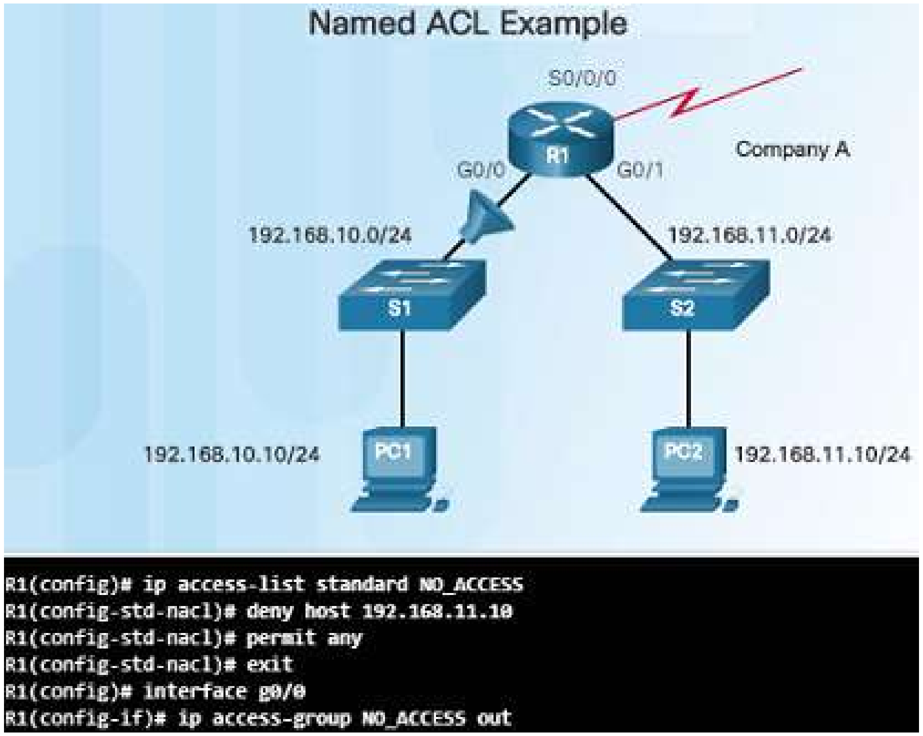


```
R1(config)# no access-list 1
R1(config)# access-list 1 deny host 192.168.10.10
R1(config)# access-list 1 permit any
R1(config)# interface g0/0
R1(config-if)# ip access-group 1 in
```

- 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 ACLs

Named Standard IPv4 ACL Syntax

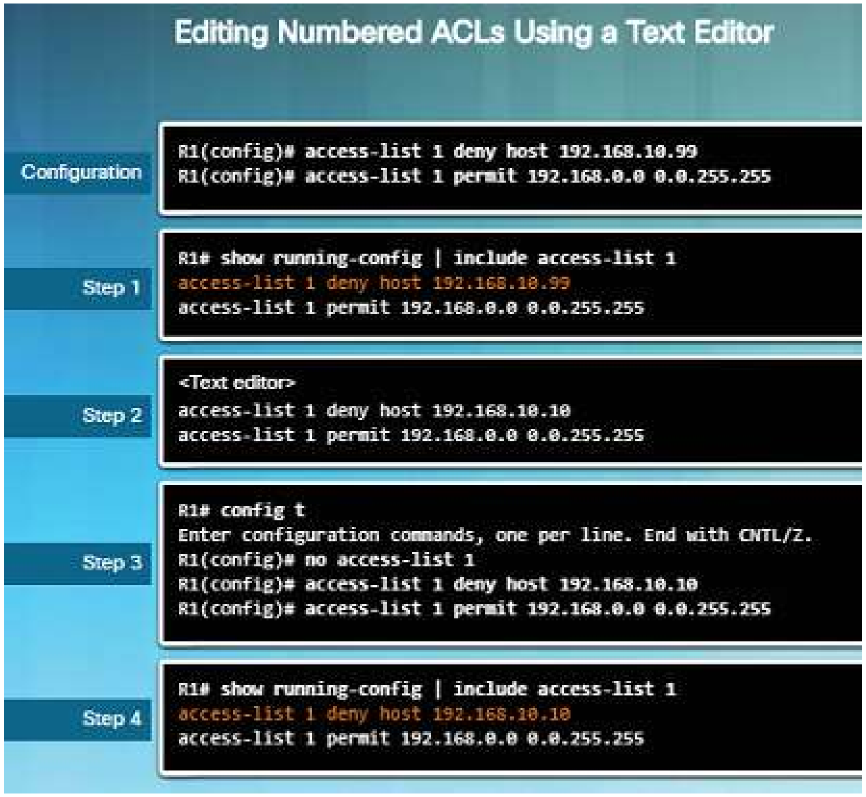


```
R1(config)# ip access-list standard NO_ACCESS
R1(config-std-nacl)# deny host 192.168.11.10
R1(config-std-nacl)# permit any
R1(config-std-nacl)# exit
R1(config)# interface g0/0
R1(config-if)# ip access-group NO_ACCESS out
```

- 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.

Modify IPv4 ACLs

Method 1 – Use a Text Editor



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 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
```

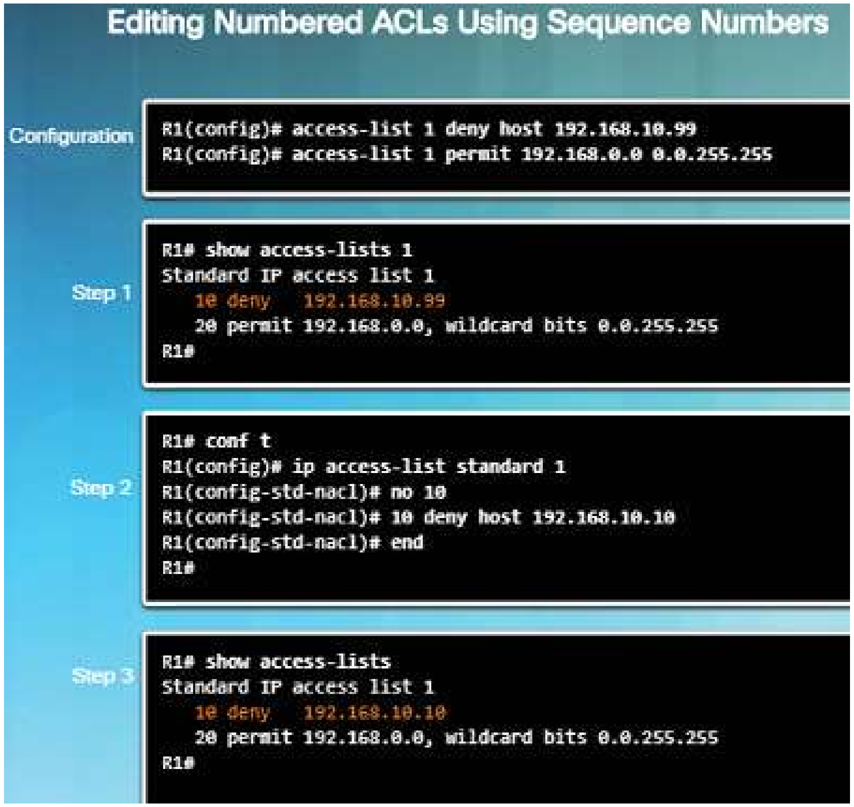
Step 4

```
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
```

- It is sometimes easier to create and edit ACLs in a text editor such as Microsoft Notepad rather making changes directly on the router.
- For an existing ACL, use the **show running-config** command to display the ACL, copy and paste it into the text editor, make the necessary changes, and then paste it back in to the router interface.
- It is important to note that when using the no access-list command, different IOS software releases act differently.
  - If the ACL that has been deleted is still applied to the interface, some IOS versions act as if no ACL is protecting your network while others deny all traffic.

Modify IPv4 ACLs

Method 2 – Use Sequence Numbers



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#
```

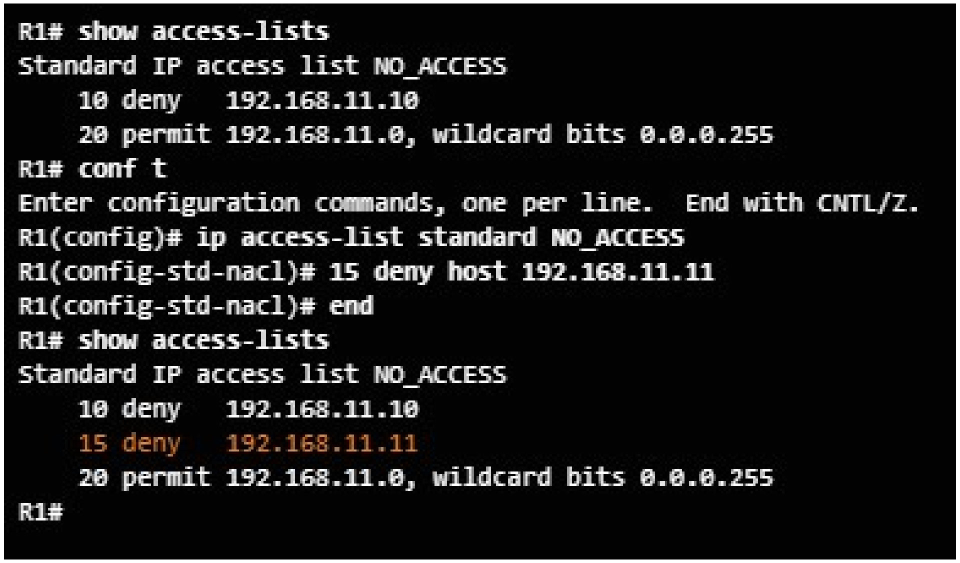
Step 3

```
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#
```

- The figure to the left demonstrates the steps used to make changes to a numbered ACL using sequence numbers.
- Step 1 identifies the problem. The **deny 192.168.10.99** statement is incorrect. The host to deny should be 192.168.10.10
- To make the edit, Step 2 shows how to go into standard access-list 1 and make the change. The misconfigured statement had to be deleted with the no command: **no 10**
- Once it was deleted, the new statement with the correct host was added: **10 deny host 192.168.10.10**

Modify IPv4 ACLs

Editing Standard Named ACLs



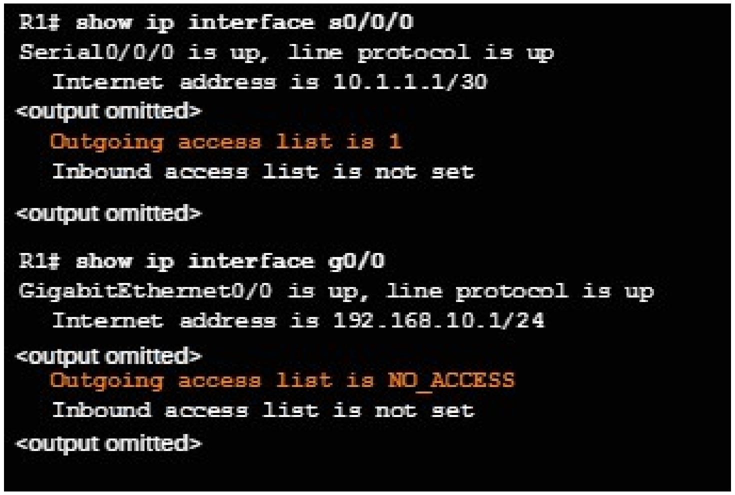
```
R1# show access-lists
Standard IP access list NO_ACCESS
    10 deny   192.168.11.10
    20 permit 192.168.11.0, wildcard bits 0.0.0.255
R1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# ip access-list standard NO_ACCESS
R1(config-std-nacl)# 15 deny host 192.168.11.11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

- The **no** *sequence-number* named ACL command is used to delete individual statements.

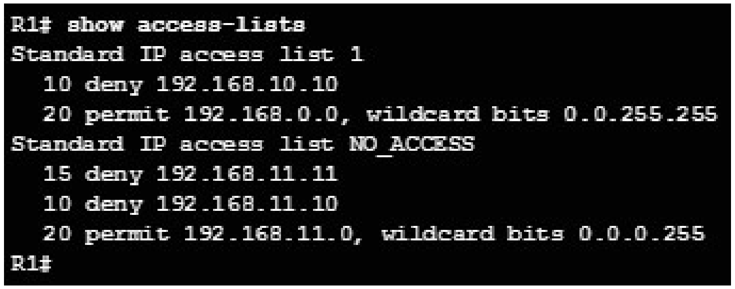
- By referring to statement sequence numbers, individual statements can be easily inserted or deleted.
- The figure to the left shows an example of how to insert a line into a named ACL.
- By numbering it 15, it will place the command in between statement 10 and 20.
- Please notice that when the ACL was originally created, the network administrator spaced each command by 10 which left room for edits and additions.

Modify IPv4 ACLs

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>
  Outgoing access list is NO_ACCESS
  Inbound access list is not set
<output omitted>
```

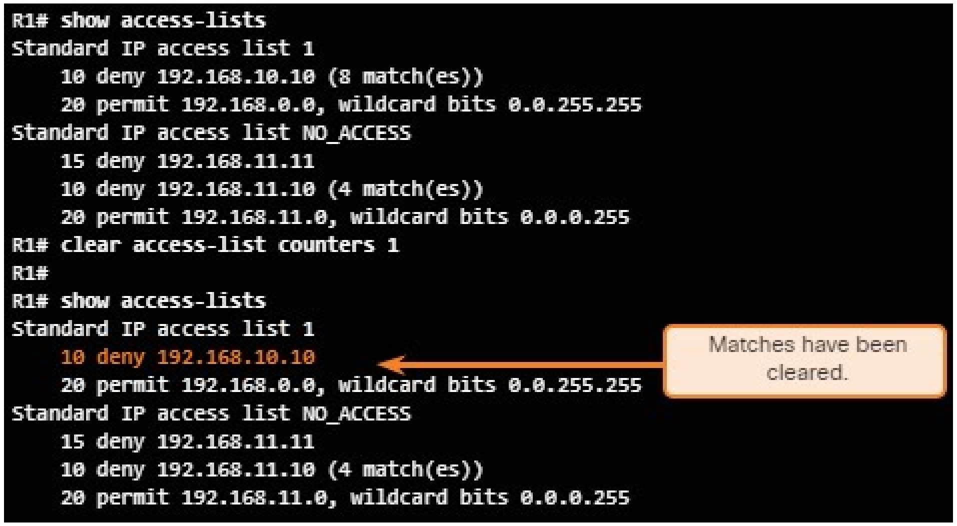


```
R1# show access-lists
Standard IP access list 1
  10 deny 192.168.10.10
  20 permit 192.168.0.0, wildcard bits 0.0.255.255
Standard IP access list NO_ACCESS
  15 deny 192.168.11.11
  10 deny 192.168.11.10
  20 permit 192.168.11.0, wildcard bits 0.0.0.255
R1#
```

- Use the **show ip interface** command to verify that the ACL is applied to the correct interface.
- The output will display the name of the access list and the direction in which it was applied to the interface.
- Use the **show access-lists** command to display the access-lists configured on the router.
- Notice how the sequence is displayed out of order for the NO_ACCESS access list. This will be discussed later in this section.

Modify IPv4 ACLs

ACL Statistics



```
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# clear access-list counters 1
R1#
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 (4 match(es))
    20 permit 192.168.11.0, wildcard bits 0.0.0.255
```

Matches have been cleared.

- The **show access-lists** command can be used to display matched statistics after an ACL has been applied to an interface and some testing has occurred.

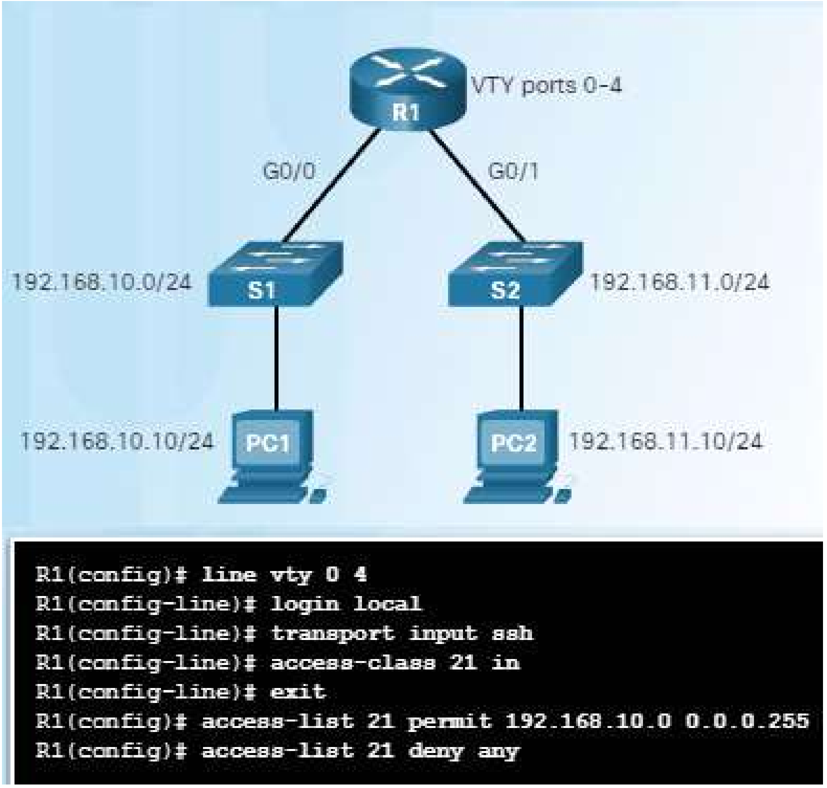
- When traffic is generated that should match an ACL statement, the matches shown in the **show access-lists** command output should increase.

- Recall that every ACL has an implicit **deny any** as the last statement. The statistics for this implicit command will not be displayed. However, if this command is configured manually, the results will be displayed.

- The clear access-list counters command can be used to clear the counters for testing purposes.

Securing VTY ports with a Standard IPv4 ACL

The access-class Command

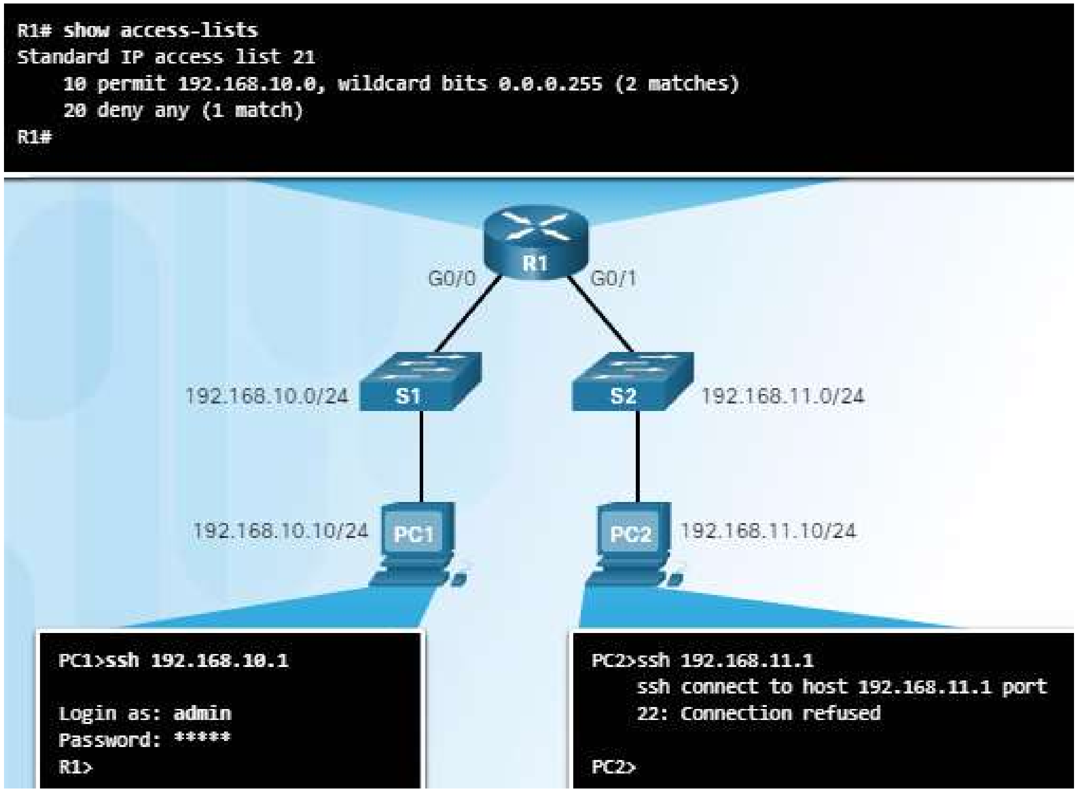


```
R1(config)# line vty 0 4
R1(config-line)# login local
R1(config-line)# transport input ssh
R1(config-line)# access-class 21 in
R1(config-line)# exit
R1(config)# access-list 21 permit 192.168.10.0 0.0.0.255
R1(config)# access-list 21 deny any
```

- 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



```
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#
```

```
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>
```

- 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.