Access Control Lists

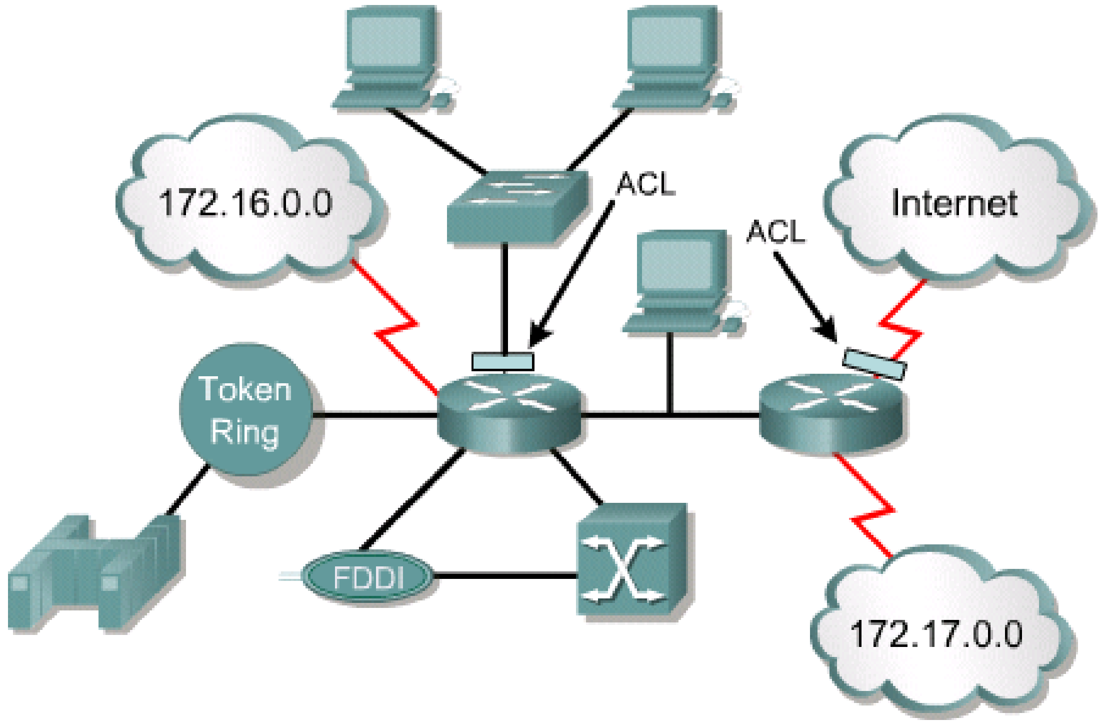
Faculty of Technology

University of Sri Jayewardenepura

2020

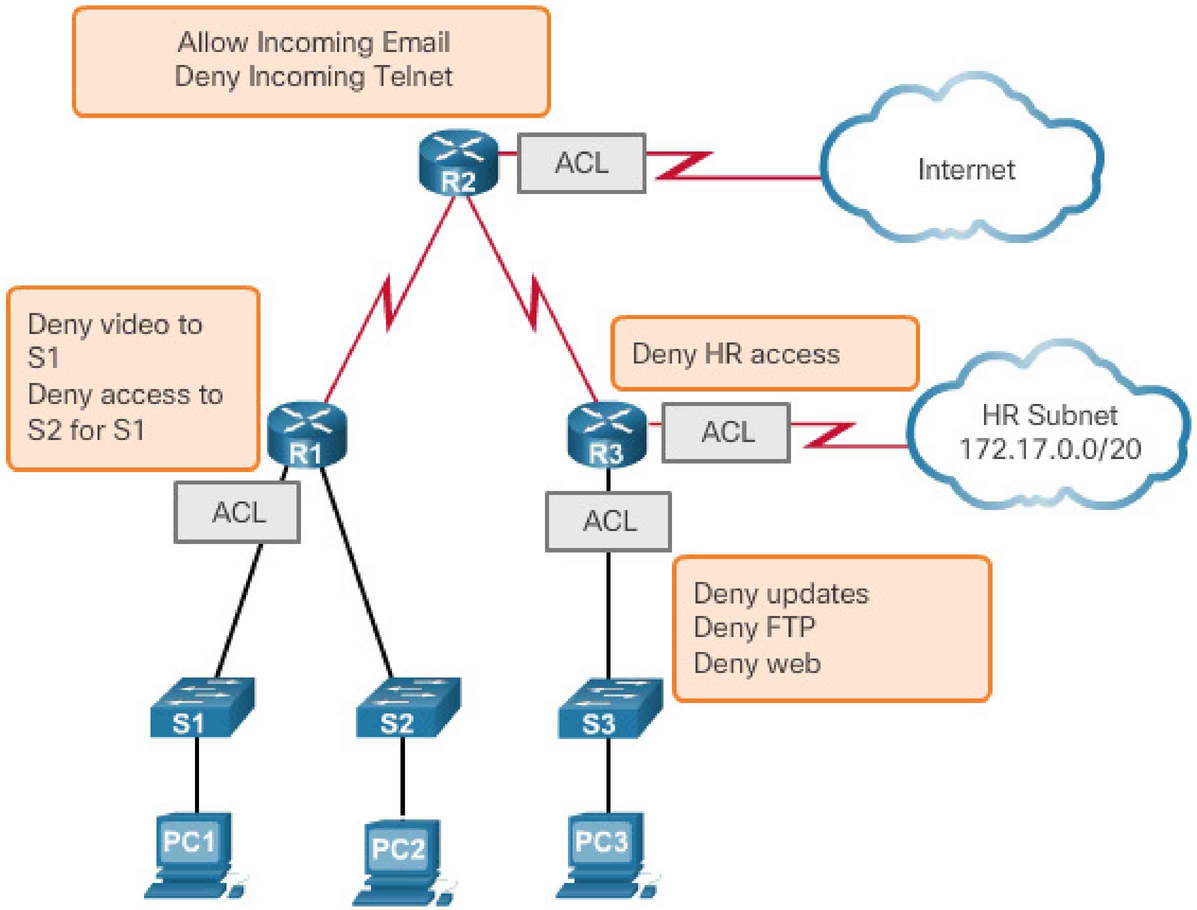
What is an ACL?

- An ACL is a sequential collection of permit or deny statements that apply to addresses or upper-layer protocols.
- Routers provide basic traffic filtering capabilities, such as blocking internet traffic, with access control lists (ACLs.
- ACLs are lists of instructions you apply to a router's interface. These lists tell the router what kinds of packets to accept and what kinds of packets to deny.



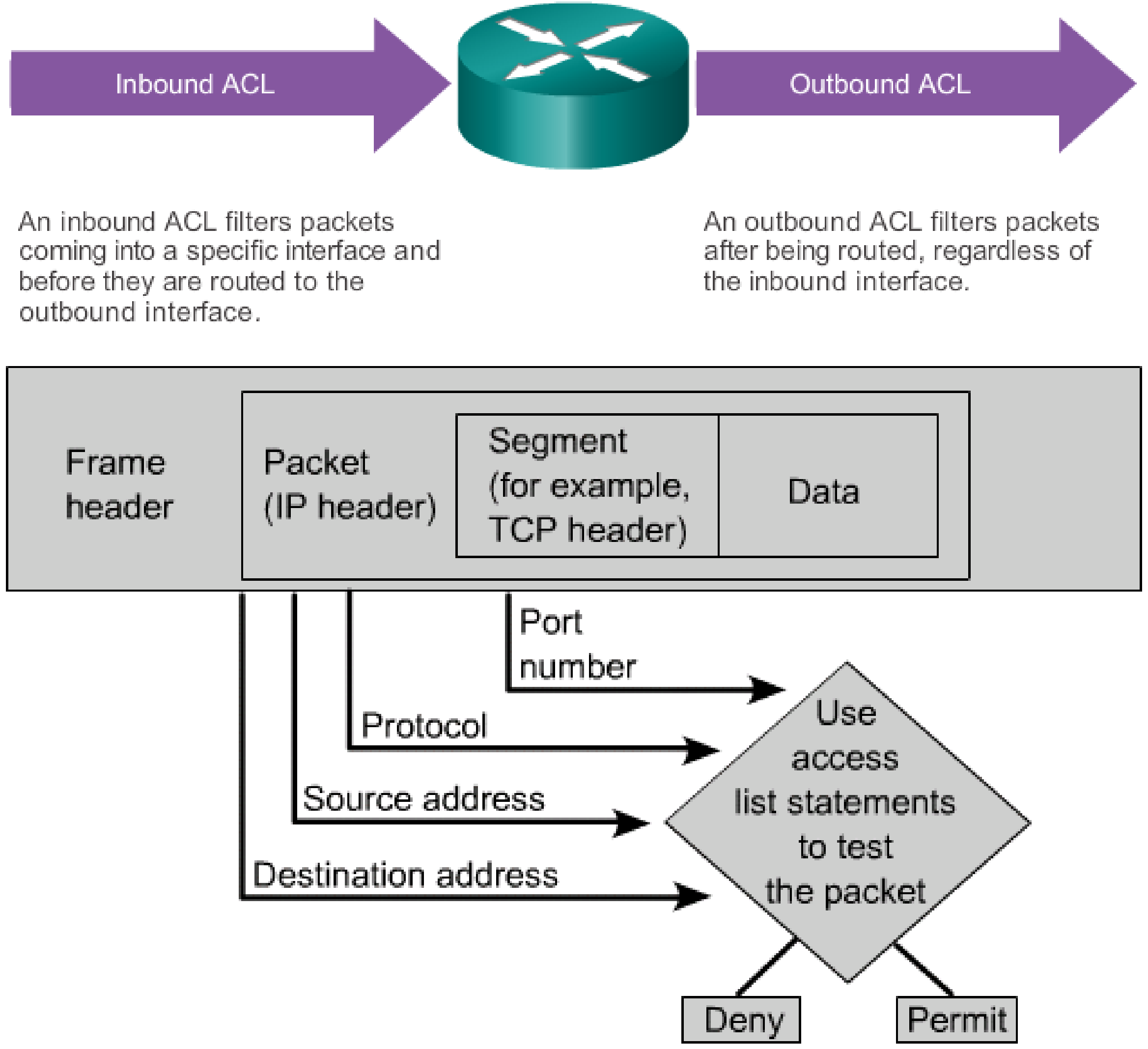
What is an ACL?

- ACL's can perform the following tasks:
  - Limit network traffic to increase network performance. For example, video traffic could be blocked.
  - Provide traffic flow control.
  - Help verify routing updates are from a known source.
  - 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/HTTP.



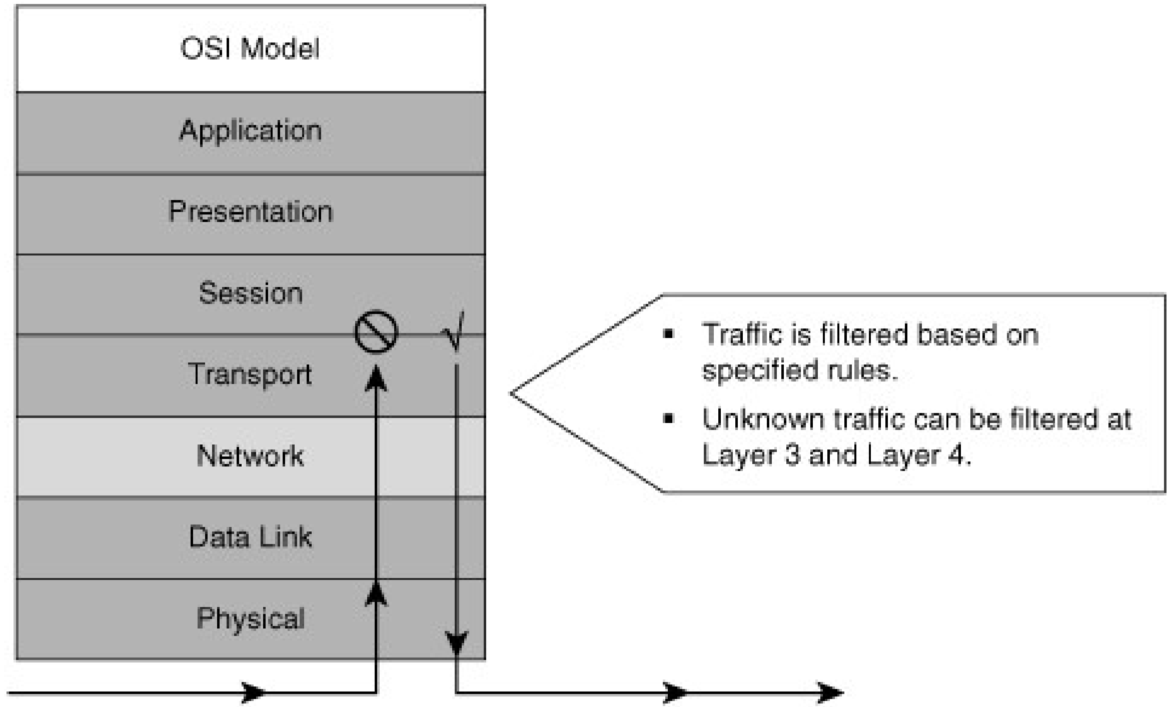
ACL Operation

- ACLs can be created for all routed network protocols, such as Internet Protocol (IP) and Internetwork Packet Exchange (IPX).
- ACLs must be defined on a per-protocol, per direction, or per port basis.
- ACLs control traffic in one direction at a time on an interface.
- A separate ACL would need to be created for each direction, one for inbound and one for outbound traffic.
- Finally every interface can have multiple protocols and directions defined.


Inbound ACL
Outbound ACL
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.
Frame header
Packet (IP header)
Segment (for example, TCP header)
Data
Port number
Protocol
Source address
Destination address
Use access list statements to test the packet
Deny
Permit

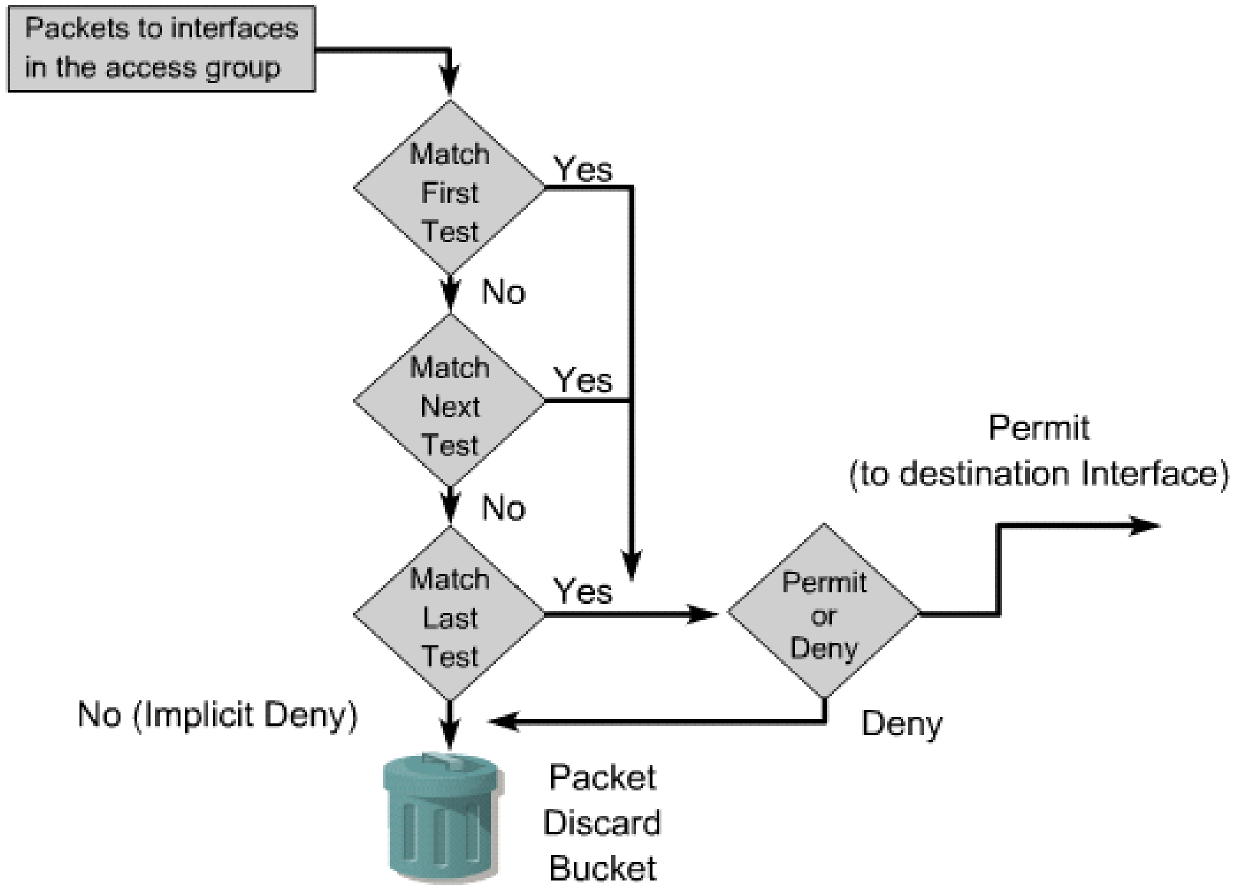
Packet Filtering

- 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.


OSI Model
Application
Presentation
Session
Transport
Network
Data Link
Physical
▪ Traffic is filtered based on specified rules.
▪ Unknown traffic can be filtered at Layer 3 and Layer 4.

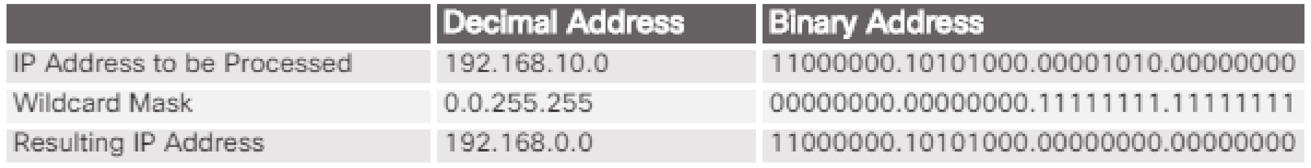
Packet Filtering

- The Cisco IOS software tests the packet against each condition statement in order from the top of the list to the bottom.
- Once a match is found in the list, the accept or reject action is performed and no other ACL statements are checked.
- If a condition statement that permits all traffic is located at the top of the list, no statements added below that will ever be checked.

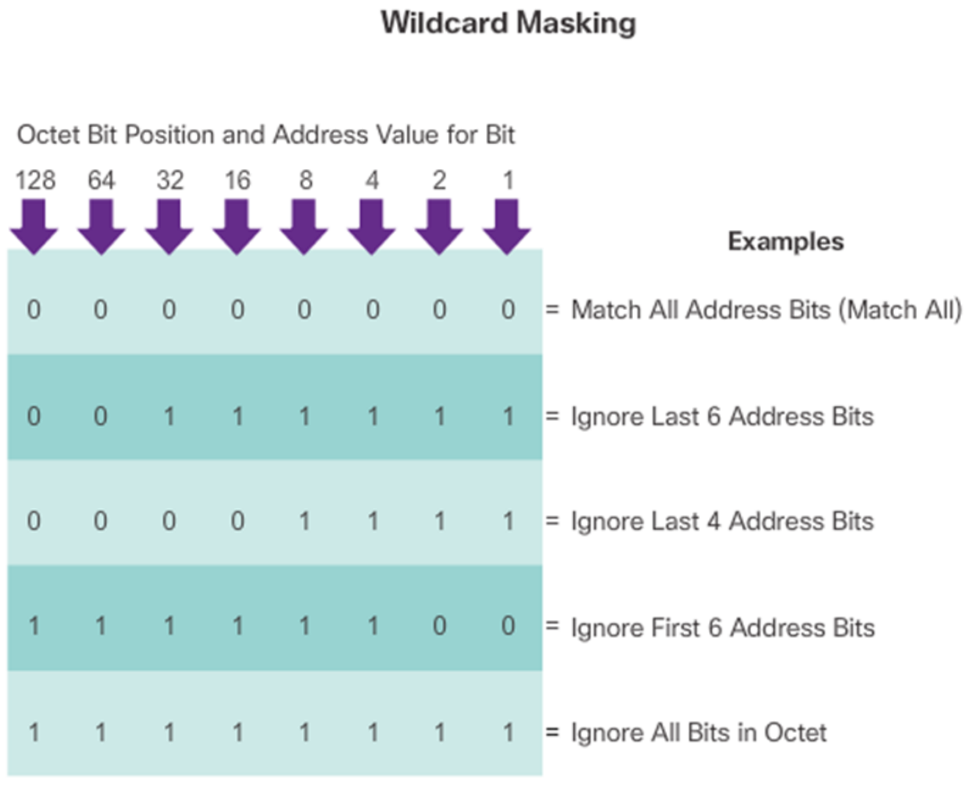


Wildcard Masks in ACLs

- IPv4 ACEs require the use of wildcard masks.
- A wildcard mask is a string of 32 binary digits (1s, 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

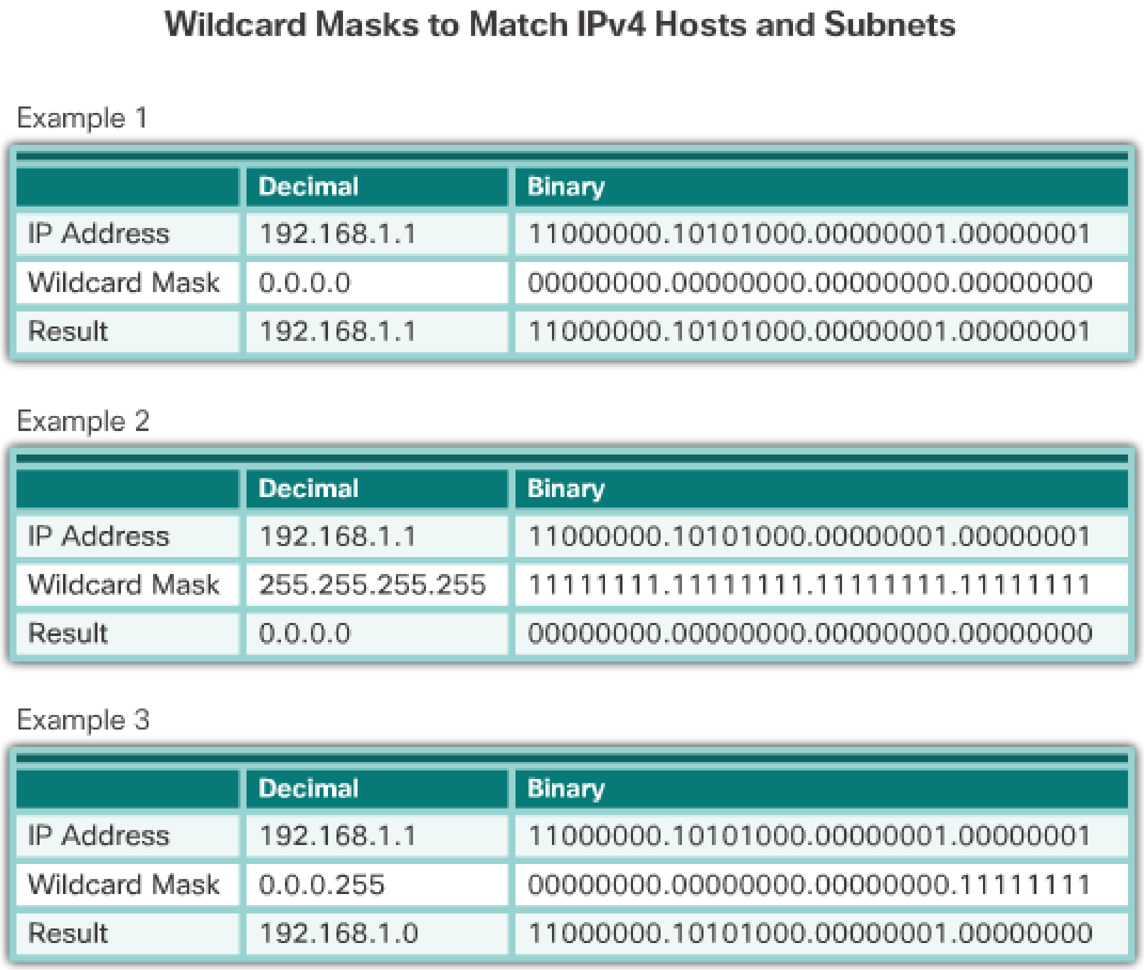


128	64	32	16	8	4	2	1	Examples
0	0	0	0	0	0	0	0	= Match All Address Bits (Match All)
0	0	1	1	1	1	1	1	= Ignore Last 6 Address Bits
0	0	0	0	1	1	1	1	= Ignore Last 4 Address Bits
1	1	1	1	1	1	0	0	= Ignore First 6 Address Bits
1	1	1	1	1	1	1	1	= Ignore All Bits in Octet

0 means to match the value of the corresponding address bit
1 means to ignore the value of the corresponding address bit

Wildcard Mask Examples

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

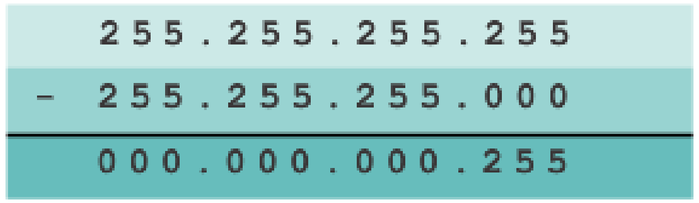
Example 3

	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

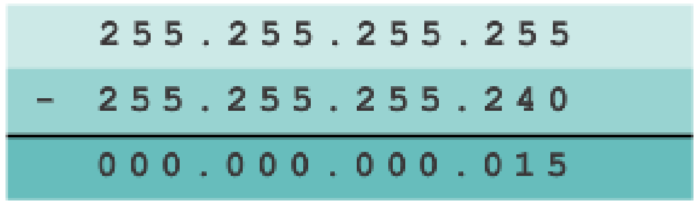
- Example 1:
  - Assume you want to permit access to all users in the 192.168.3.0 network with the subnet mask of 255.255.255.0.
  - After subtracting the subnet mask from 255.255.255.255, the result is 0.0.0.255.
- Example 2:
  - Assume you want to permit network access for the 14 users in the subnet 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.

Example 1



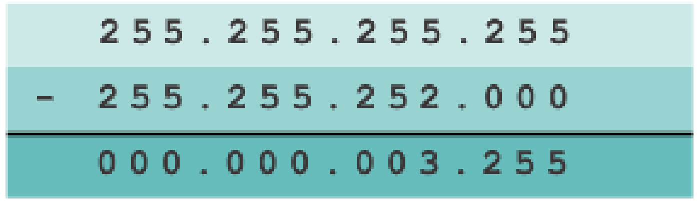
	255.255.255.255
–	255.255.255.000
	000.000.000.255

Example 2



	255.255.255.255
–	255.255.255.240
	000.000.000.015

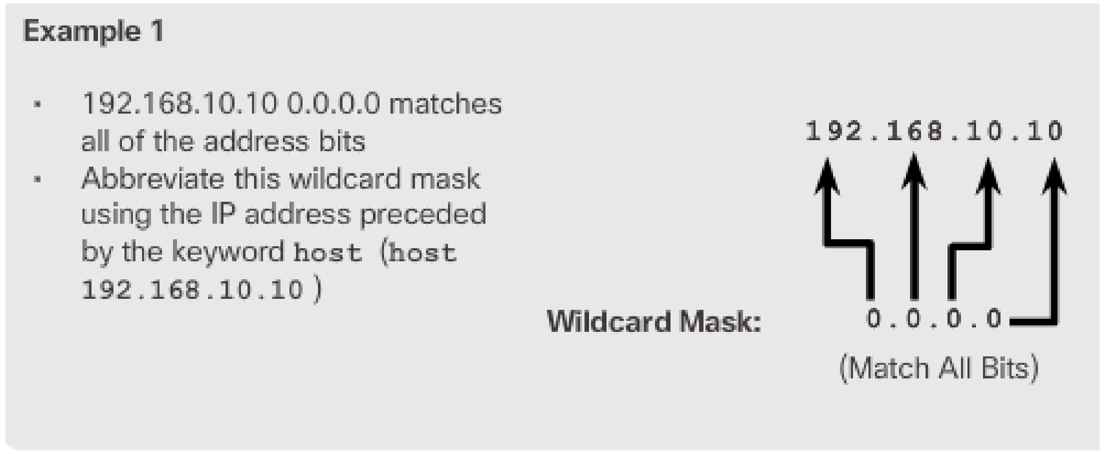
Example 3



	255.255.255.255
–	255.255.252.000
	000.000.003.255

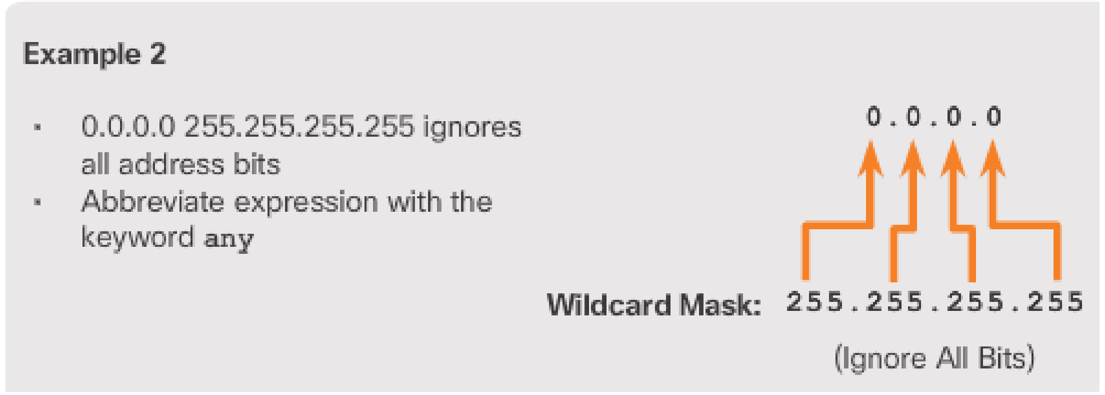
Wildcard Mask Keywords

- 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.



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`)

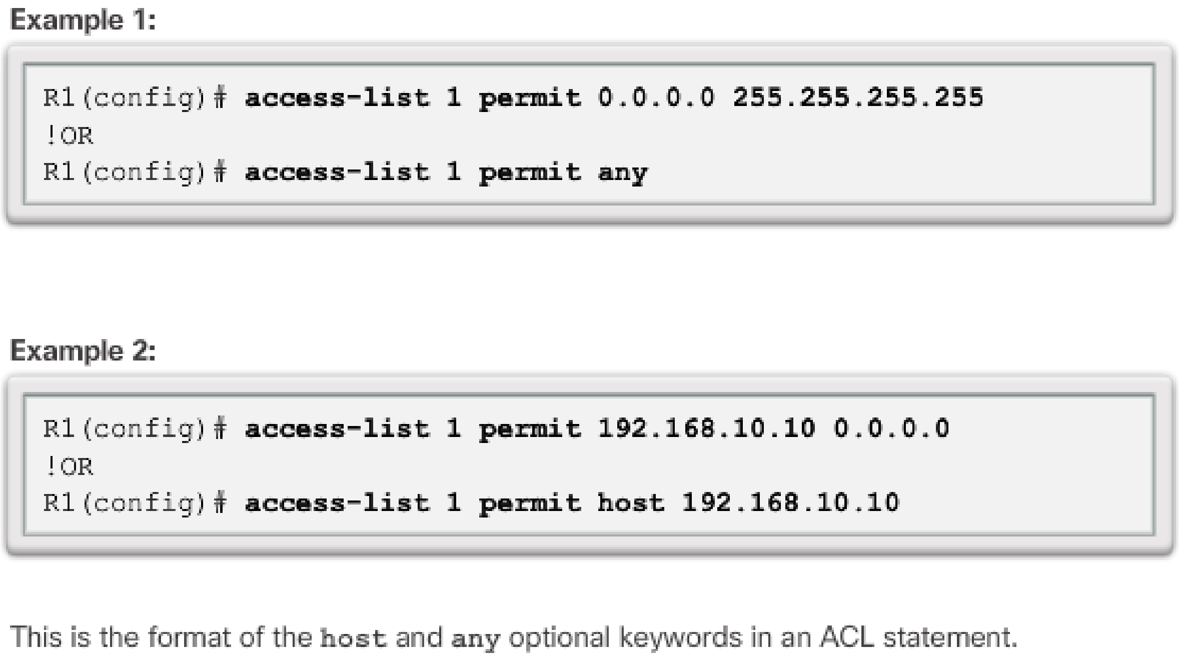


Example 2

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

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



Example 1:

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

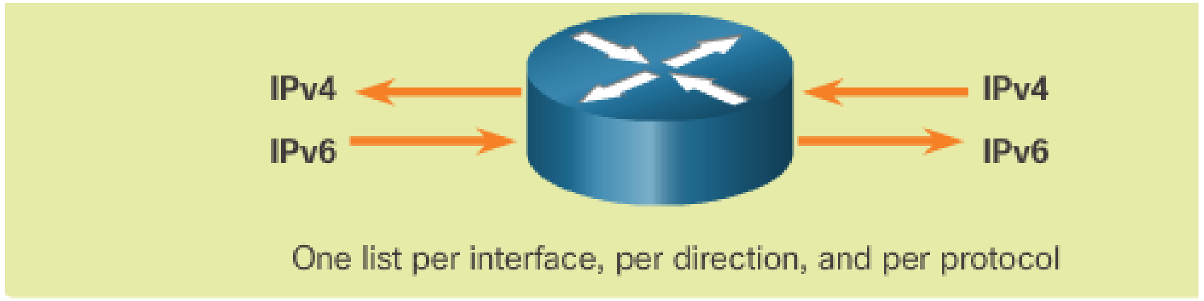
Example 2:

```
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
```

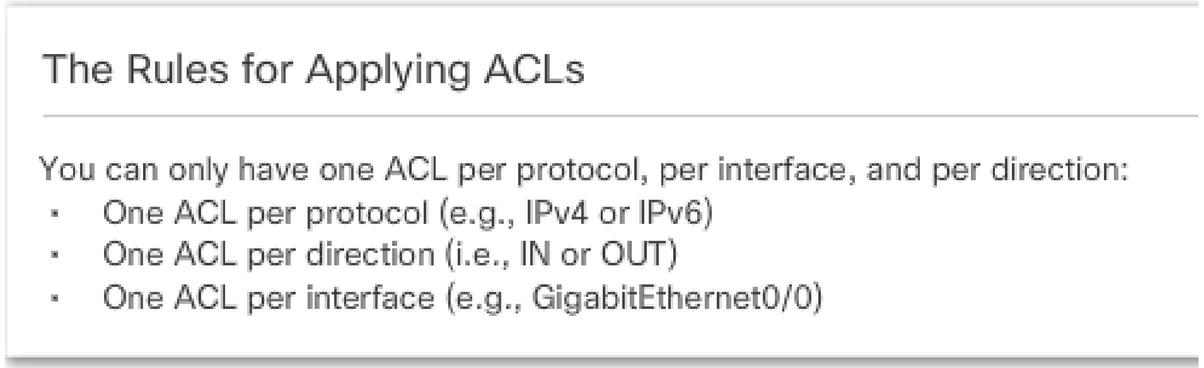
This is the format of the `host` and `any` optional keywords in an ACL statement.

General Guidelines for Creating ACLs

- 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.



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



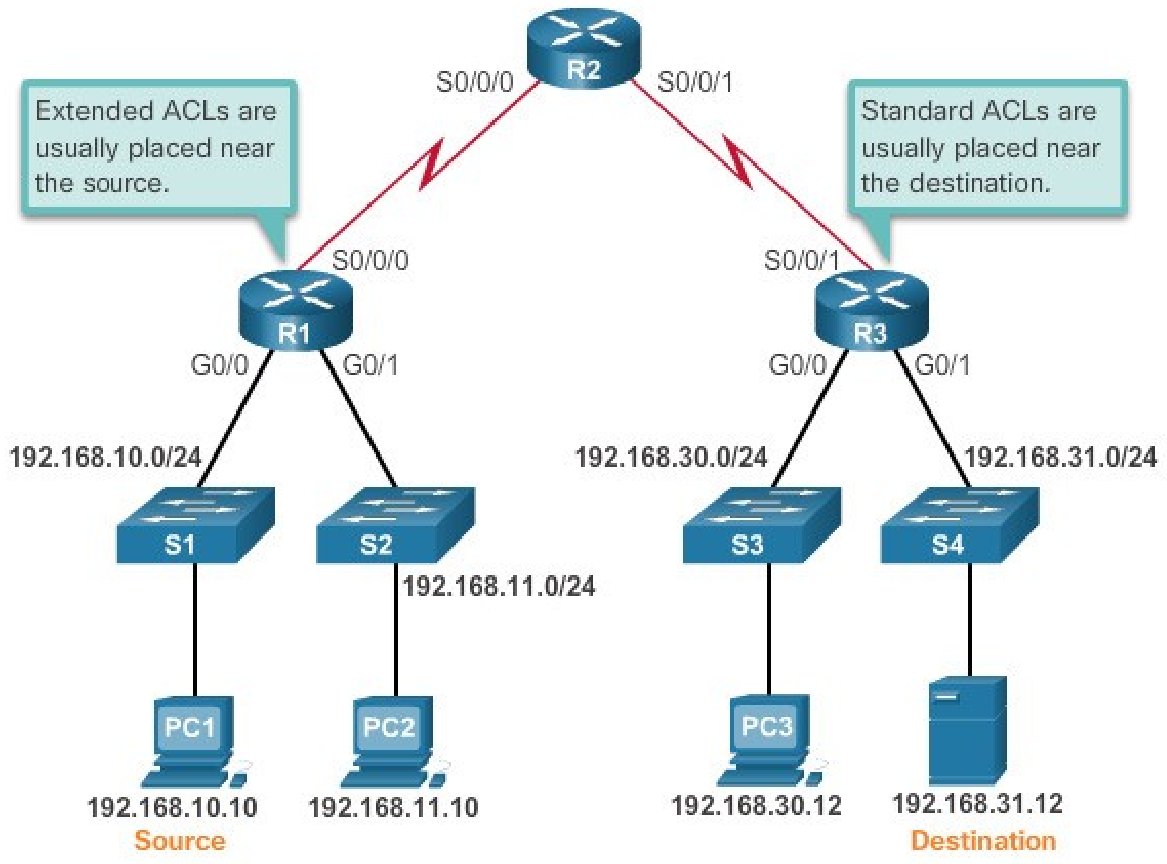
The Rules for Applying ACLs

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)

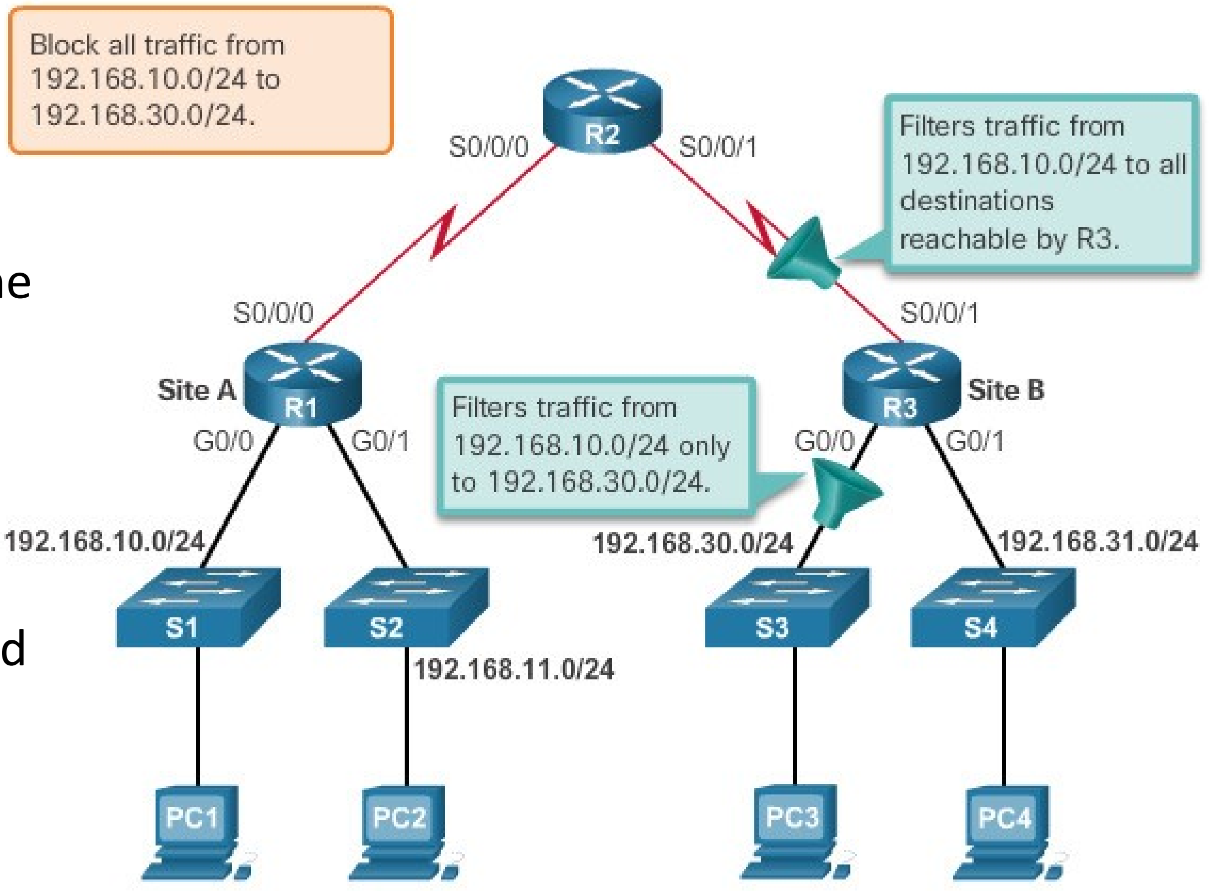
General Guidelines for Creating ACLs

- 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.



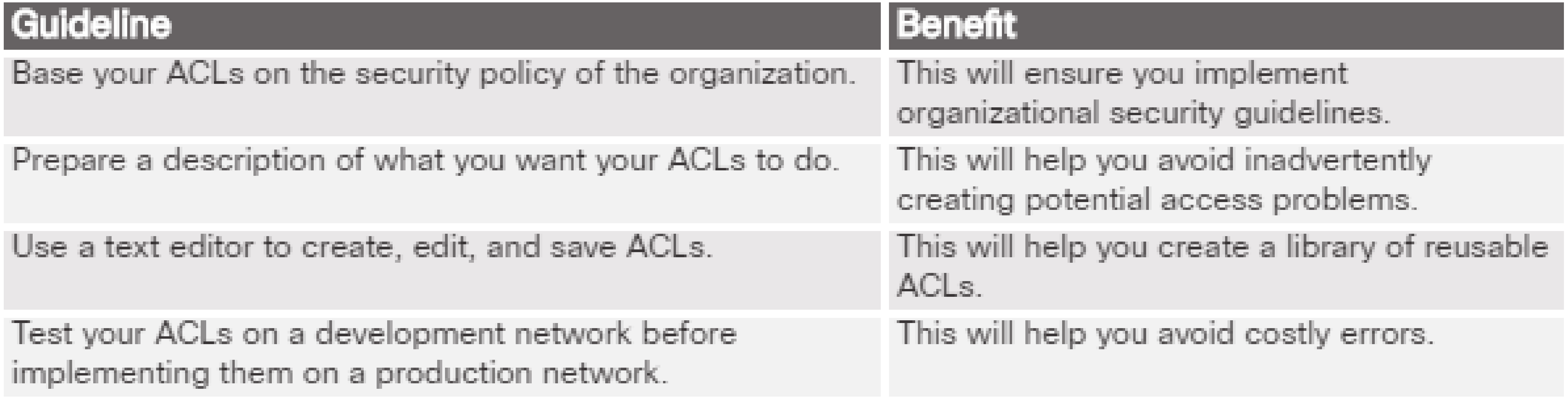
Standard ACL Placement

- 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.


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
G0/0
G0/1
Filters traffic from 192.168.10.0/24 only to 192.168.30.0/24.
R3
Site B
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

ACL Best Practices

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

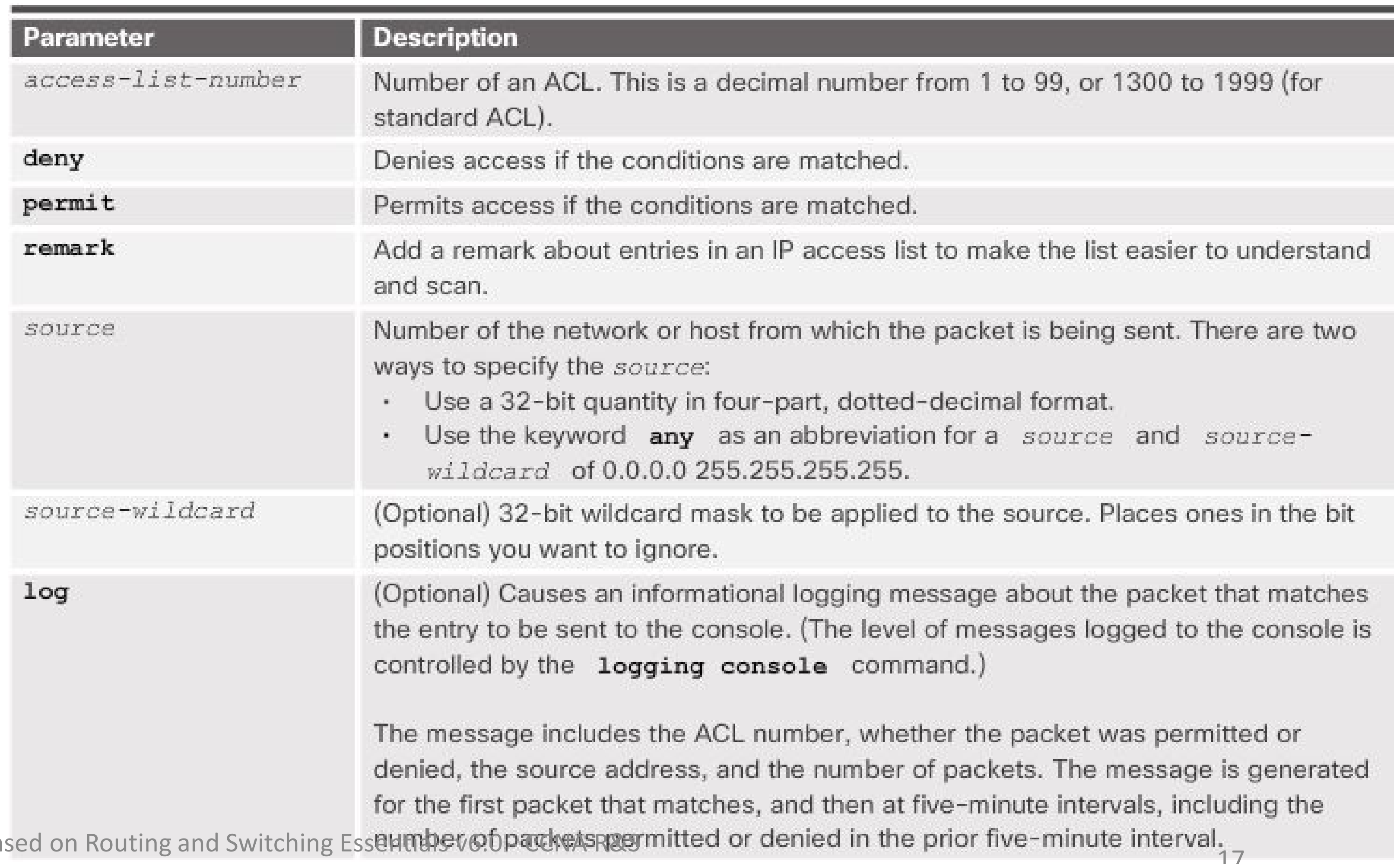


Guideline	Benefit
Base your ACLs on the security policy of the organization.	This will ensure you implement organizational security guidelines.
Prepare a description of what you want your ACLs to do.	This will help you avoid inadvertently creating potential access problems.
Use a text editor to create, 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.

Configure Standard IPv4 ACLs

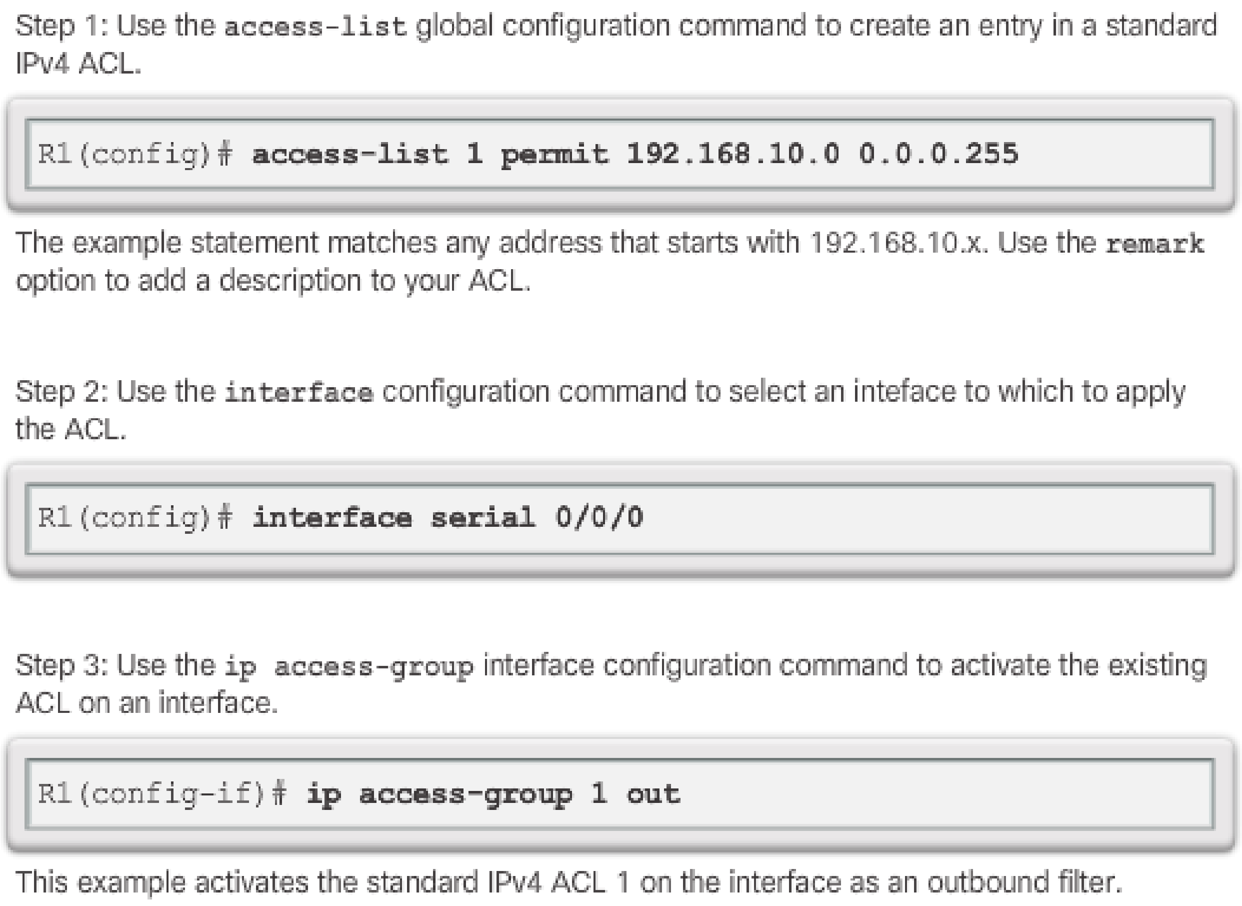
Numbered Standard IPv4 ACL Syntax

- 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.



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.

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:

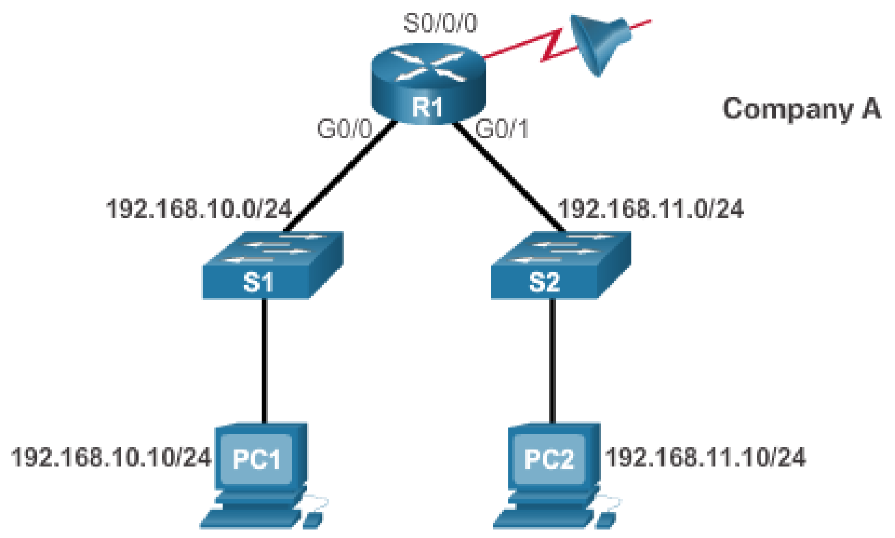
  Router(config-if)# **ip access-group** { *access-list-number* | *access-list-name* } { **in** | **out** }

- 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.

Numbered Standard IPv4 ACL Examples

- The figure to the right 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.

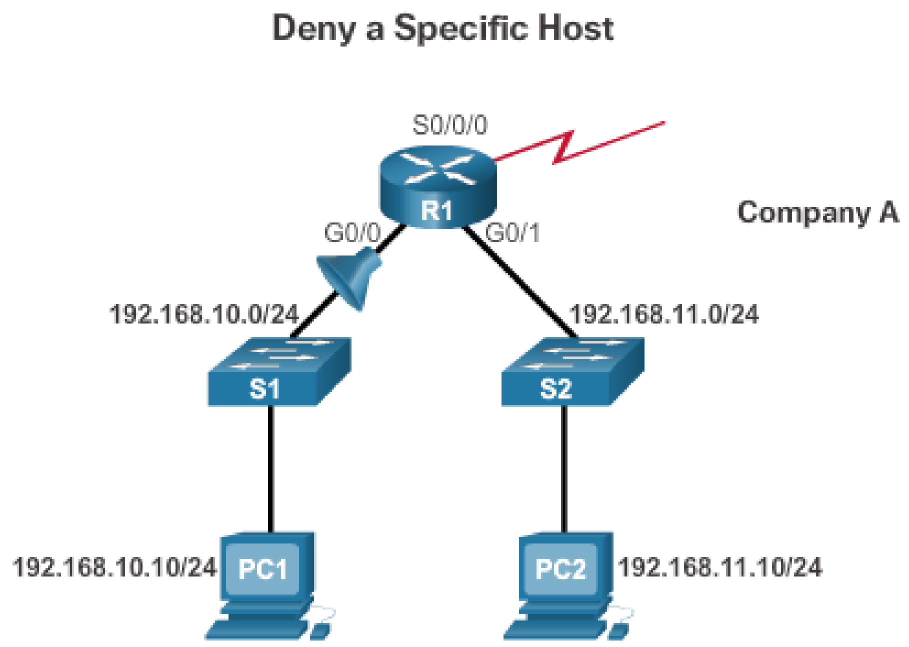
Deny a Specific Host and Permit a Specific Subnet



```
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
```

Numbered Standard IPv4 ACL Examples

- 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.

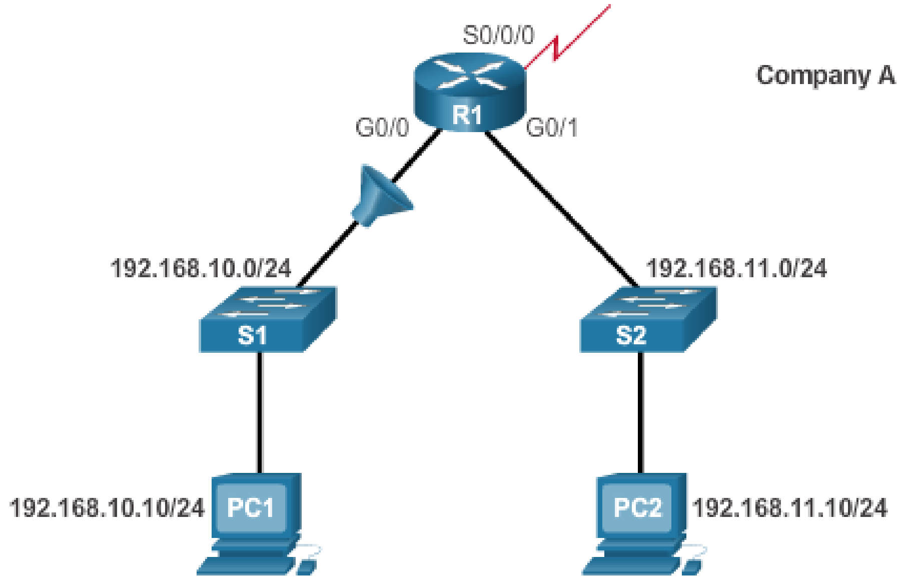


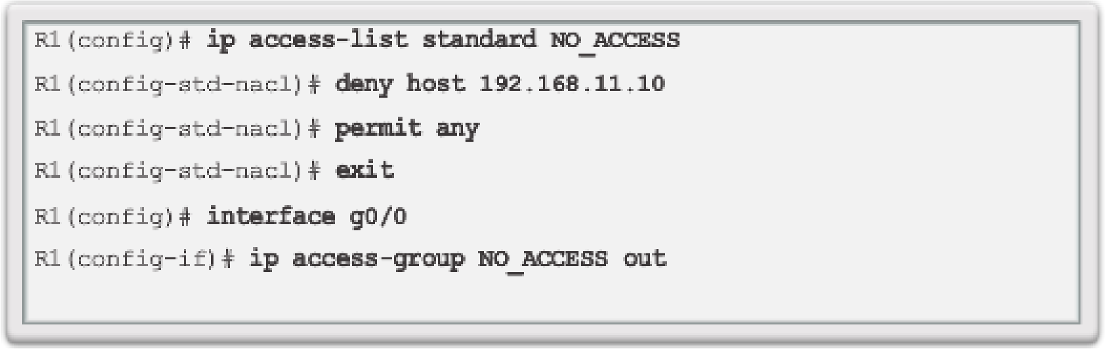
```
R1(config)# no access-list 1
R1(config)# access-list 1 deny host 192.168.10.10
R1(config)# access-list 1 permit any
R1(config)# interface g0/0
R1(config-if)# ip access-group 1 in
```

Named Standard IPv4 ACL Syntax

- Identifying an ACL with a name rather than with a number makes it easier to understand its function.
- The example to the right 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.

Named ACL Example

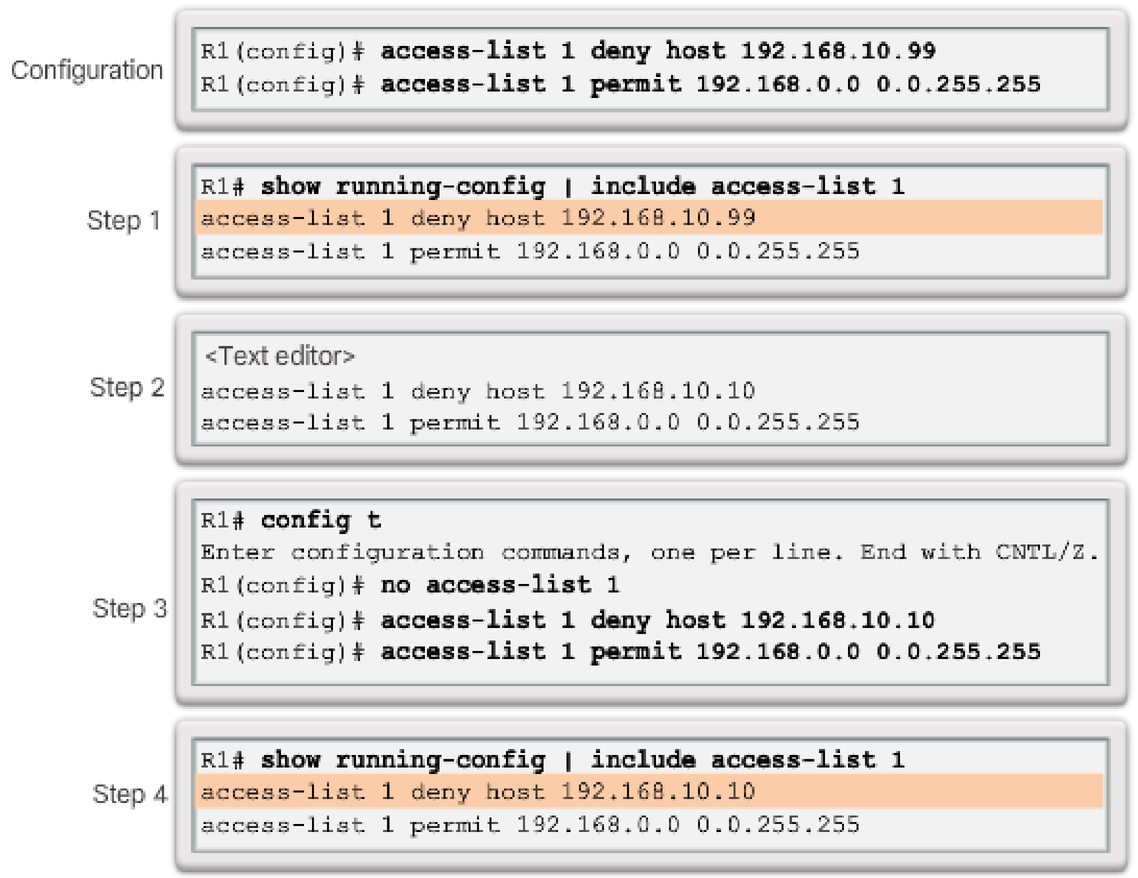




```
R1(config)# ip access-list standard NO_ACCESS
R1(config-std-nacl)# deny host 192.168.11.10
R1(config-std-nacl)# permit any
R1(config-std-nacl)# exit
R1(config)# interface g0/0
R1(config-if)# ip access-group NO_ACCESS out
```

Modify IPv4 ACLs

Editing Numbered ACLs Using a Text Editor



Configuration

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

Step 1

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

Step 2

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

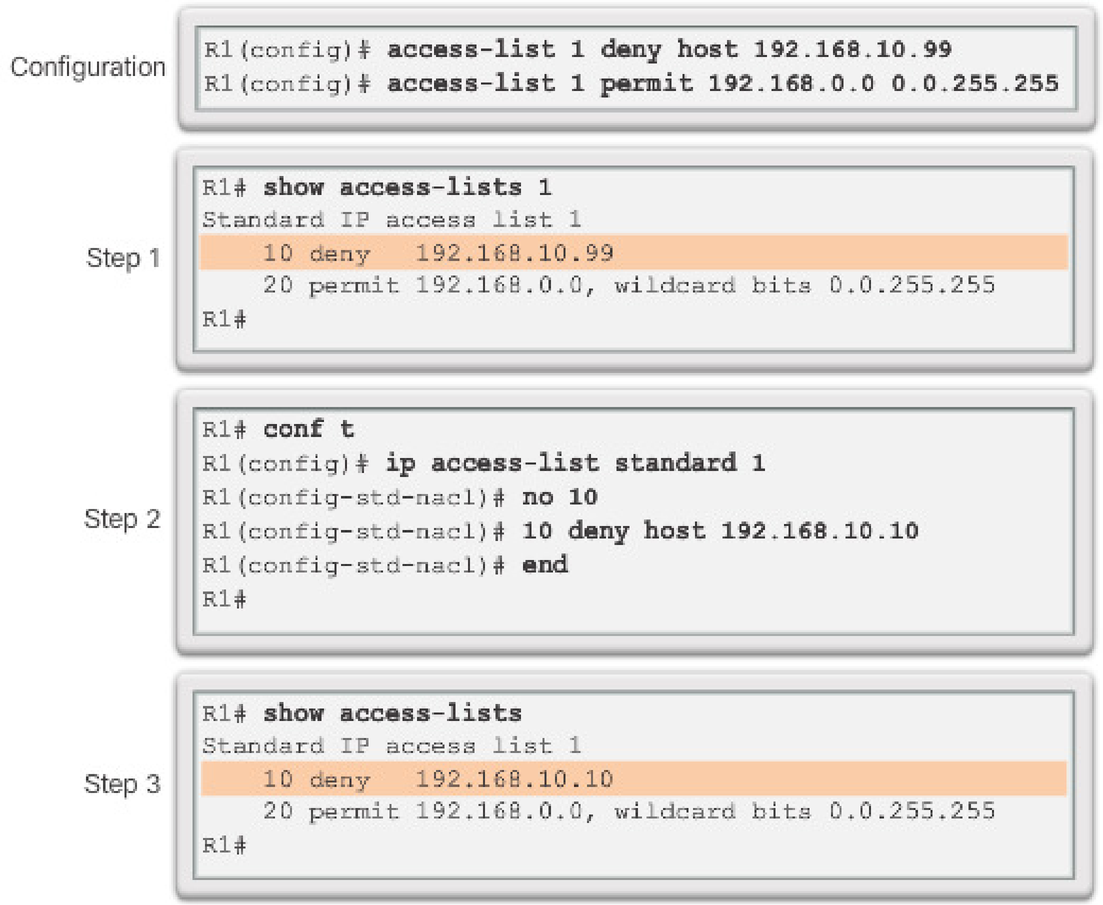
Step 3

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

Step 4

```
R1# show running-config | include access-list 1
access-list 1 deny host 192.168.10.10
access-list 1 permit 192.168.0.0 0.0.255.255
```

Editing 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

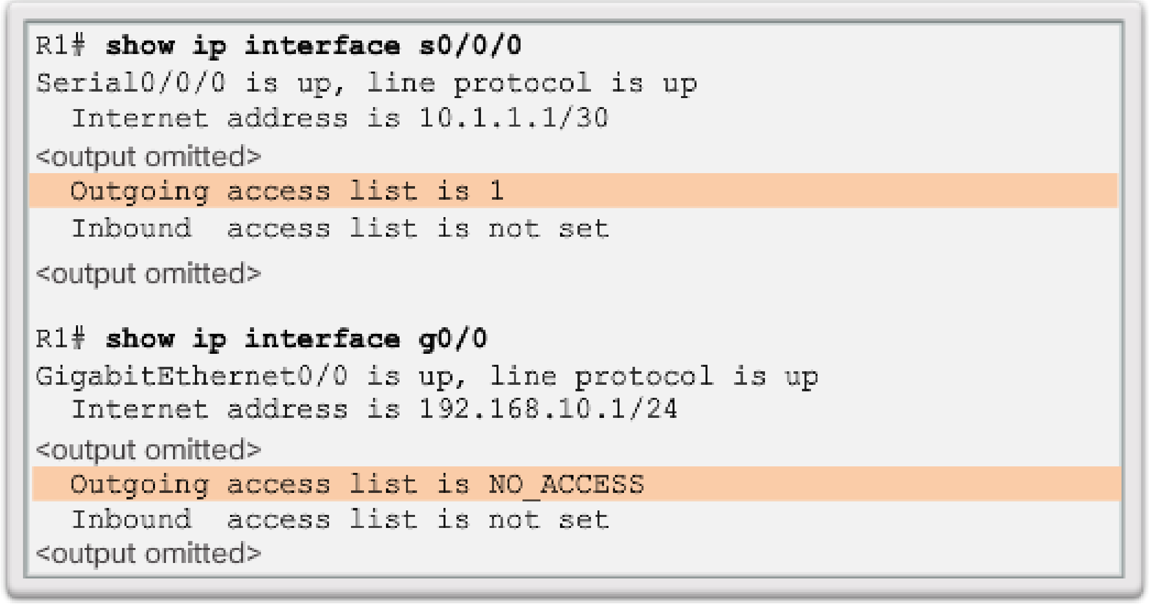
```
R1# conf t
R1(config)# ip access-list standard 1
R1(config-std-nacl)# no 10
R1(config-std-nacl)# 10 deny host 192.168.10.10
R1(config-std-nacl)# end
R1#
```

Step 3

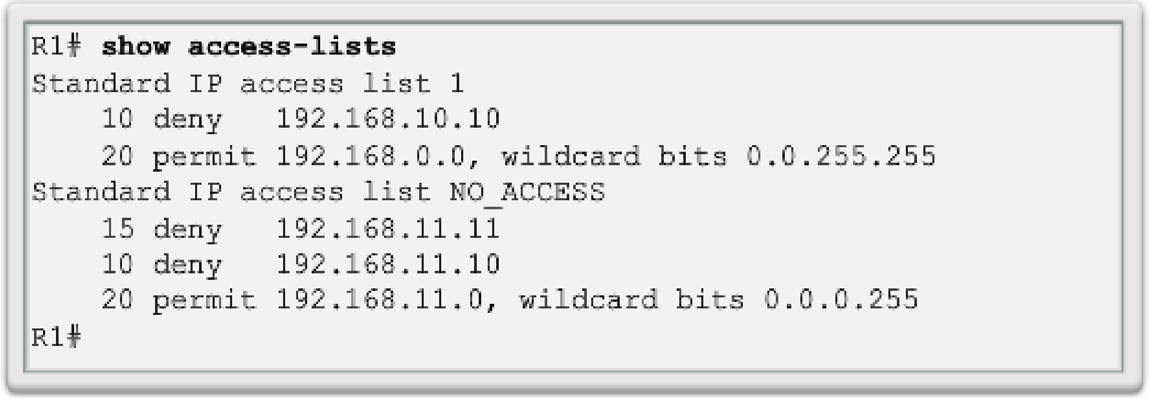
```
R1# show access-lists
Standard IP access list 1
    10 deny   192.168.10.10
    20 permit 192.168.0.0, wildcard bits 0.0.255.255
R1#
```

Verifying ACLs

- 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.



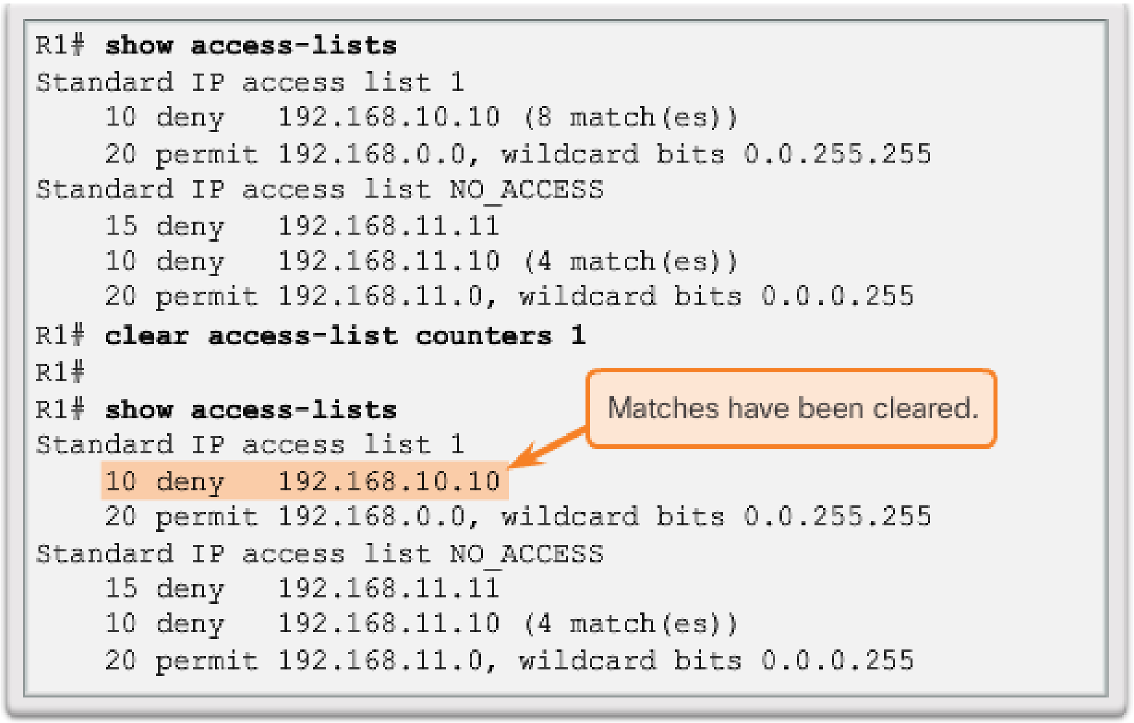
```
R1# show ip interface s0/0/0
Serial0/0/0 is up, line protocol is up
  Internet address is 10.1.1.1/30
<output omitted>
  Outgoing access list is 1
  Inbound  access list is not set
<output omitted>

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#
```

ACL Statistics

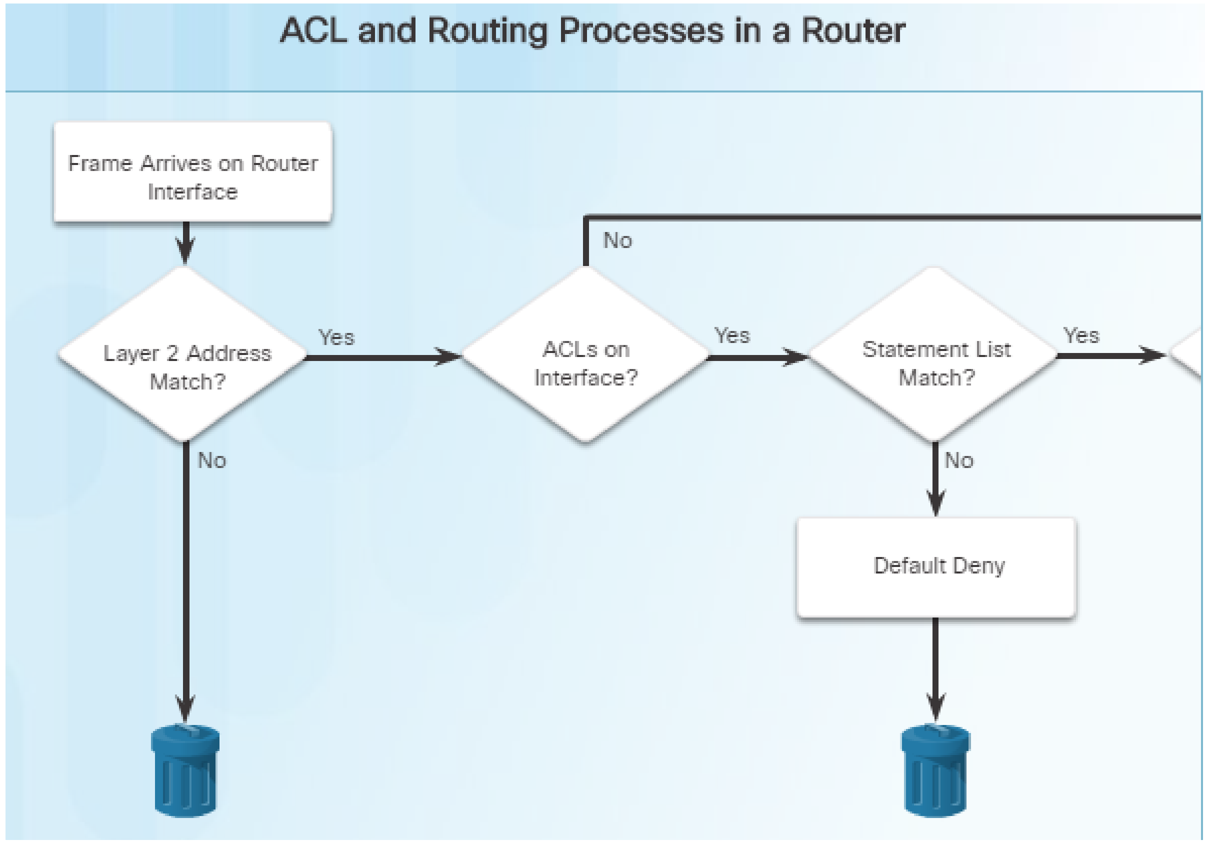
- The **show access-lists** command can be used to display matched statistics after an ACL has been applied to an interface and some testing has occurred.
- When traffic is generated that should match an ACL statement, the matches shown in the **show access-lists** command output should increase.
- Recall 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.



```
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
```

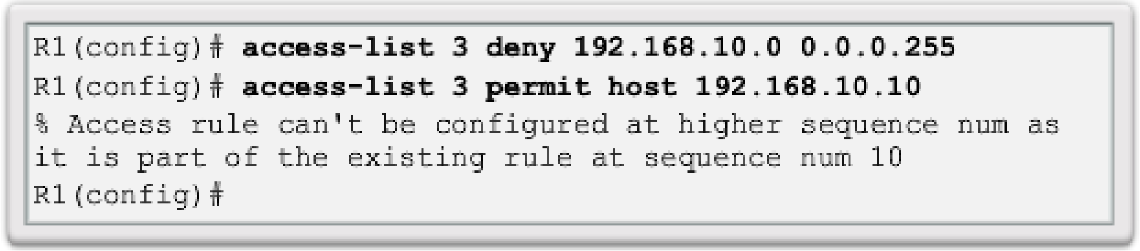
Routing Processes and ACLs

- The figure shows the logic of routing and ACL processes.
- When a packet arrives at a router interface, the router process is the same, whether ACLs are configured or not.
- After the frame information is stripped off, the router checks for an ACL on the inbound interface. If an ACL exists, the packet is tested against the statements.
- If the packet matches a statement, the packet is either permitted or denied.
- If the packet is permitted, and after the router processes the packet, the outgoing interface will also be checked for an ACL.



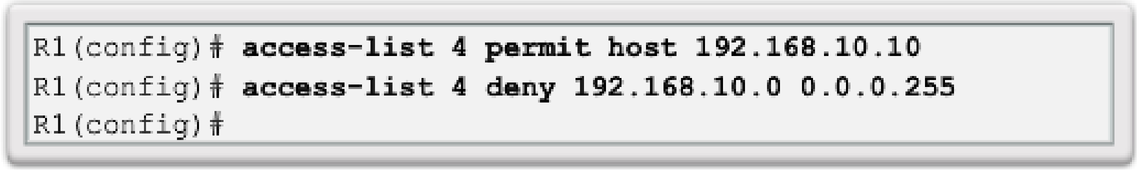
The Order of ACEs in an ACL

- The order in which ACEs are configured are important since ACEs are processed sequentially.
- The figure to the right 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.



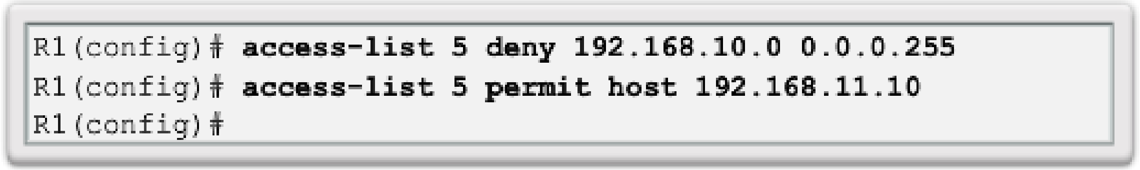
```
R1(config)# access-list 3 deny 192.168.10.0 0.0.0.255
R1(config)# access-list 3 permit host 192.168.10.10
% Access rule can't be configured at higher sequence num as
it is part of the existing rule at sequence num 10
R1(config)#
```

ACL 3: Host statement conflicts with previous range statement.



```
R1(config)# access-list 4 permit host 192.168.10.10
R1(config)# access-list 4 deny 192.168.10.0 0.0.0.255
R1(config)#
```

ACL 4: Host statement can always be configured before range statements.

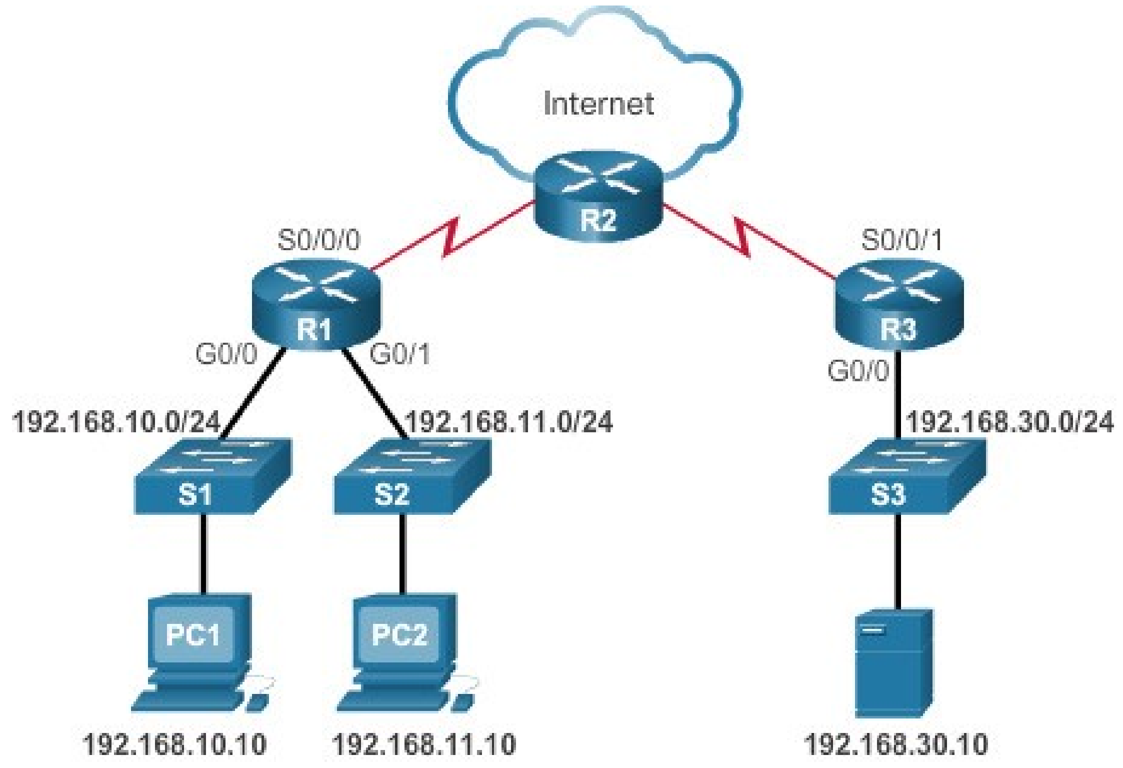


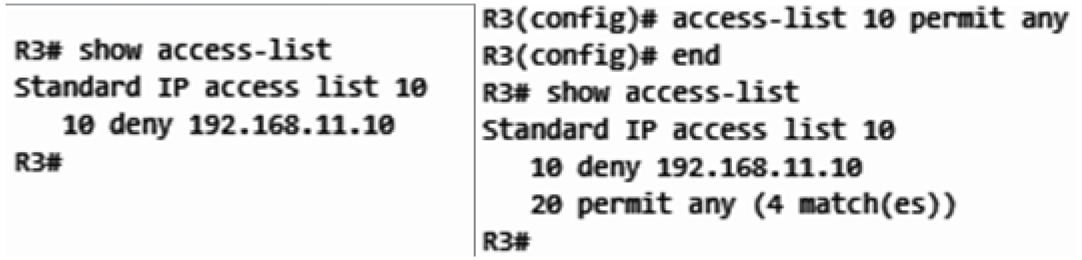
```
R1(config)# access-list 5 deny 192.168.10.0 0.0.0.255
R1(config)# access-list 5 permit host 192.168.11.10
R1(config)#
```

ACL 5: Host statement can be configured after range statement if there is no conflict.

Common IPv4 Standard ACL Errors

- The most common errors involving ACLs:
  - Entering ACEs 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 right, 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.



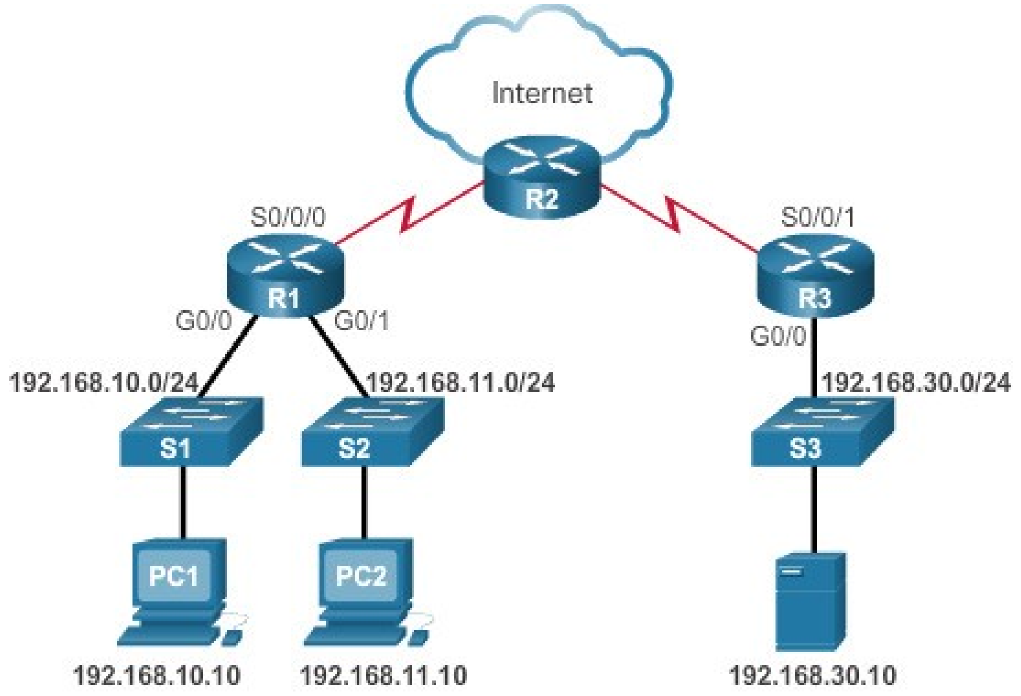


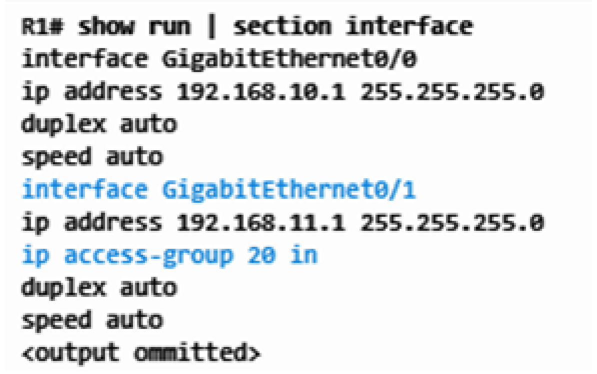
```
R3# show access-list
Standard IP access list 10
   10 deny 192.168.11.10
R3#
```

```
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#
```

Common IPv4 Standard ACL Errors

- 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.

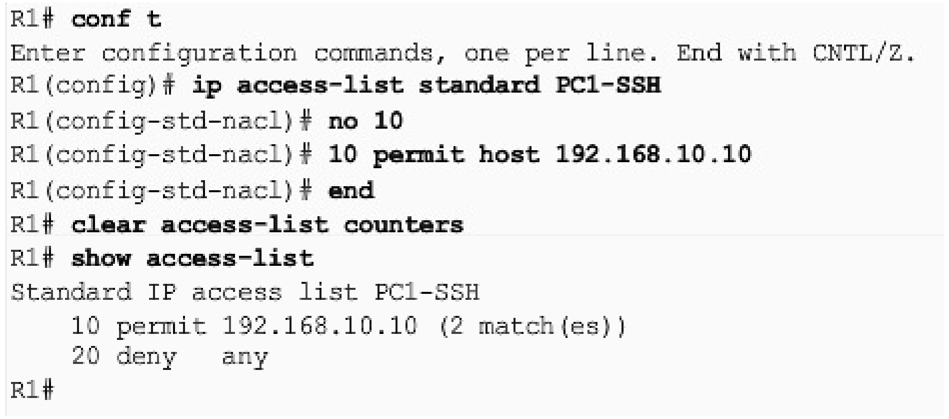




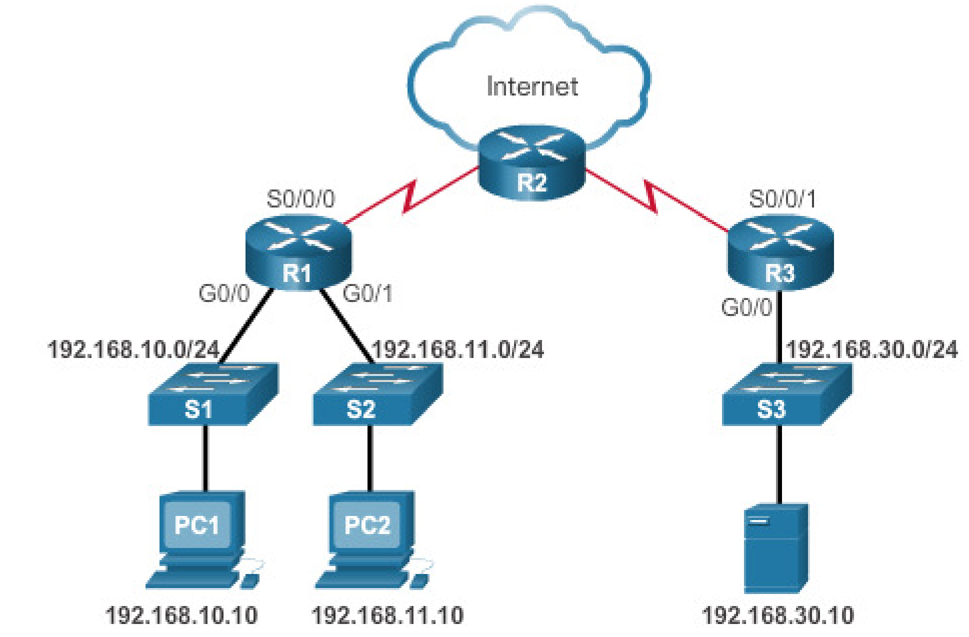
```
R1# show run | section interface
interface GigabitEthernet0/0
ip address 192.168.10.1 255.255.255.0
duplex auto
speed auto
interface GigabitEthernet0/1
ip address 192.168.11.1 255.255.255.0
ip access-group 20 in
duplex auto
speed auto
<output ommitted>
```

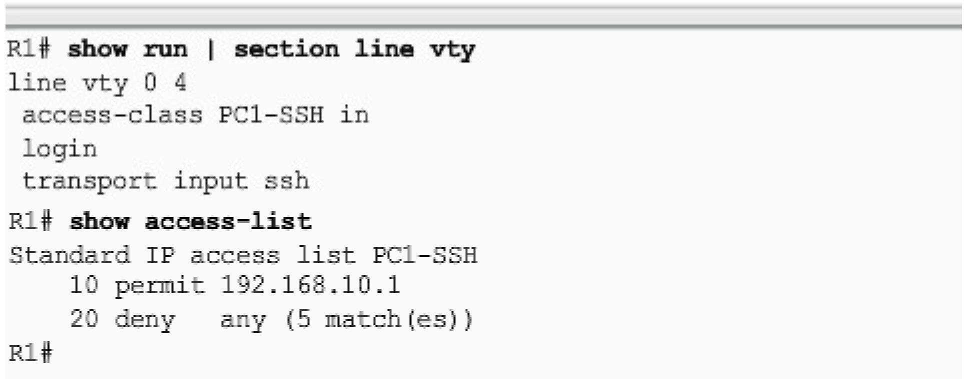
Common IPv4 Standard ACL Errors

- Only PC1 should be allowed to SSH to R1.
- There is a problem with the config in the figure to the right 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#
```





```
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#
```

Summary

- ACL Operation
  - The purpose and operation of ACLs in small to medium-sized business networks.
- Standard IPv4 ACLs
  - Configure standard IPv4 ACLs to filter traffic in a small to medium-sized business network.