AAiT
ADDIS ABABA INSTITUTE OF TECHNOLOGY
አዲስ አበባ ቴክኖሎጂ ኢንስቲትዩት
ADDIS ABABA UNIVERSITY
አዲስ አበባ ዩኒቨርሲቲ

Enterprise Systems and Network Administration

CHAPTER TWO

Network Administration

Getnet M.
getnet.mamo@aau.edu.et

Switch Boot Sequence

After a Cisco switch is powered on, it goes through the following five-step boot sequence:

Step 1: First, the switch loads a power-on self-test (POST) program stored in ROM. POST checks the CPU subsystem. It tests the CPU, DRAM, and the portion of the flash device that makes up the flash file system.

Step 2: Next, the switch loads the boot loader software. The boot loader is a small program stored in ROM that is run immediately after POST successfully completes.

Step 3: The boot loader performs low-level CPU initialization. It initializes the CPU registers, which control where physical memory is mapped, the quantity of memory, and its speed.

Step 4: The boot loader initializes the flash file system on the system board.

Step 5: Finally, the boot loader locates and loads a default IOS operating system software image into memory and gives control of the switch over to the IOS.

Recovering from a System Crash

The boot loader provides access into the switch if the operating system cannot be used because of missing or damaged system files. The boot loader has a command line that provides access to the files stored in flash memory. The boot loader can be accessed through a console connection following these steps:

Step 1. Connect a PC by console cable to the switch console port. Configure terminal emulation software to connect to the switch.

Step 2. Unplug the switch power cord.

Step 3. Reconnect the power cord to the switch and, within 15 seconds, press and hold down the **Mode** button while the System LED is still flashing green.

Step 4. Continue pressing the **Mode** button until the System LED turns briefly amber and then solid green; then release the **Mode** button.

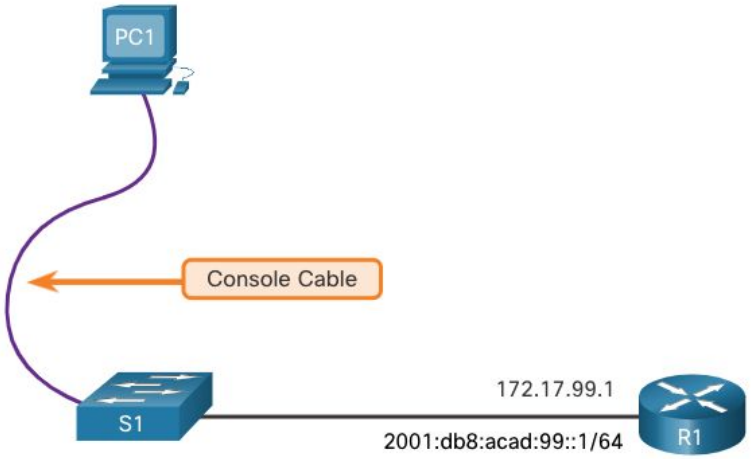
Step 5. The boot loader **switch:** prompt appears in the terminal emulation software on the PC.

The boot loader command line supports commands to format the flash file system, reinstall the operating system software, and recover a lost or forgotten password. For example, the **dir** command can be used to view a list of files within a specified directory.

Switch Management Access

To prepare a switch for remote management access, the switch must be configured with an IP address and a subnet mask.

- To manage the switch from a remote network, the switch must be configured with a default gateway. This is very similar to configuring the IP address information on host devices.



- In the figure, the switch virtual interface (SVI) on S1 should be assigned an IP address. The SVI is a virtual interface, not a physical port on the switch. A console cable is used to connect to a PC so that the switch can be initially configured.

Switch SVI Configuration Example

By default, the switch is configured to have its management controlled through VLAN 1. All ports are assigned to VLAN 1 by default. For security purposes, it is considered a best practice to use a VLAN other than VLAN 1 for the management VLAN,

Step 1: **Configure the Management Interface:** From VLAN interface configuration mode, an IPv4 address and subnet mask is applied to the management SVI of the switch.

Note: The SVI for VLAN 99 will not appear as “up/up” until VLAN 99 is created and there is a device connected to a switch port associated with VLAN 99.

Switch SVI Configuration Example

Task	IOS Commands
Enter global configuration mode.	S1# **configure terminal**
Enter interface configuration mode for the SVI.	S1(config)# **interface vlan 99**
Configure the management interface IPv4 address.	S1(config-if)# **ip address 172.17.99.11 255.255.255.0**
Enable the management interface.	S1(config-if)# **no shutdown**
Return to the privileged EXEC mode.	S1(config-if)# **end**
Save the running config to the startup config.	S1# **copy running-config startup-config**

Switch SVI Configuration Example

Step 2: Configure the Default Gateway

- The switch should be configured with a default gateway if it will be managed remotely from networks that are not directly connected.

Task	IOS Commands
Enter global configuration mode.	S1# **configure terminal**
Configure the default gateway for the switch.	S1(config)# **ip default-gateway 172.17.99.1**
Return to the privileged EXEC mode.	S1(config-if)# **end**
Save the running config to the startup config.	S1# **copy running-config startup-config**

Switch SVI Configuration Example

Step 3: **Verify Configuration**

- The **show ip interface brief** and **show ipv6 interface brief** commands are useful for determining the status of both physical and virtual interfaces. The output shown confirms that interface VLAN 99 has been configured with an IPv4 and IPv6 address.

Note: An IP address applied to the SVI is only for remote management access to the switch; this does not allow the switch to route Layer 3 packets.

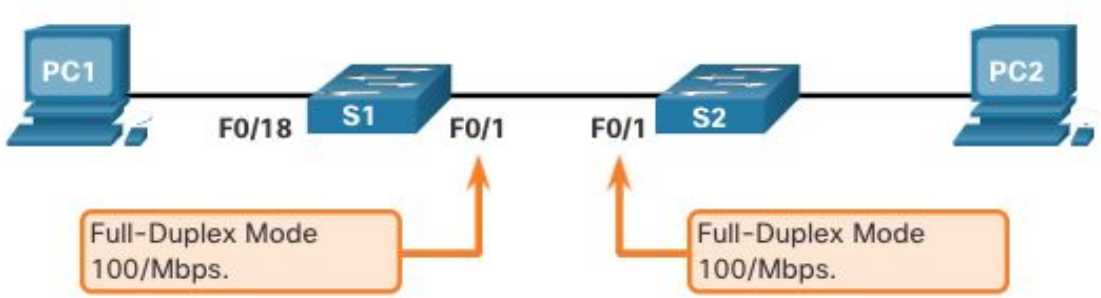
```
S1# show ip interface brief
Interface      IP-Address      OK? Method   Status     Protocol
Vlan99         172.17.99.11    YES manual   down       down
(output omitted)
S1# show ipv6 interface brief
Vlan99                    [down/down]
    FE80::C27B:BCFF:FEC4:A9C1
    2001:DB8:ACAD:99::1
(output omitted)
```

Configure Switch Ports: Configure Switch Ports

- Switch ports can be manually configured with specific duplex and speed settings. The respective interface configuration commands are **duplex** and **speed**.
    - Duplex can be: auto, half, full
    - Speed can be: 10/100/1000/auto
- The default setting for both duplex and speed for switch ports on Cisco Catalyst 2960 and 3560 switches is auto. The 10/100/1000 ports operate in either half- or full-duplex mode when they are set to 10 or 100 Mbps and operate only in full-duplex mode when it is set to 1000 Mbps (1 Gbps).
- Autonegotiation is useful when the speed and duplex settings of the device connecting to the port are unknown or may change. When connecting to known devices such as servers, dedicated workstations, or network devices, a best practice is to manually set the speed and duplex settings.
- When troubleshooting switch port issues, it is important that the duplex and speed settings are checked.

Note: Mismatched settings for the duplex mode and speed of switch ports can cause connectivity issues.

Configure Switch Ports: Configure Switch Ports



Task	IOS Commands
Enter global configuration mode.	S1# **configure terminal**
Enter interface configuration mode.	S1(config)# **interface FastEthernet 0/1**
Configure the interface duplex.	S1(config-if)# **duplex full**
Configure the interface speed.	S1(config-if)# **speed 100**
Return to the privileged EXEC mode.	S1(config-if)# **end**
Save the running config to the startup config.	S1# **copy running-config startup-config**

Configure Switch Ports: Switch Verification Commands

Task	IOS Commands
Display interface status and configuration.	S1# **show interfaces** [*interface-id*]
Display current startup configuration.	S1# **show startup-config**
Display current running configuration.	S1# **show running-config**
Display information about flash file system.	S1# **show flash**
Display system hardware and software status.	S1# **show version**
Display history of command entered.	S1# **show history**
Display IP information about an interface.	S1# **show ip interface** [*interface-id*]
Display the MAC address table.	S1# **show mac-address-table** OR S1# **show mac address-table**

Secure Remote Access: Telnet Operation

Telnet uses TCP port 23. It is an older protocol that uses unsecure plaintext transmission of both the login authentication (username and password) and the data transmitted between the communicating devices.

A threat actor can monitor packets using Wireshark.

```
Switch>enable
Switch#configure terminal
Switch(config)#enable secret password
Switch(config)#service password-encryption
Switch(config)#line vty 0 4
Switch(config-line)#password telnetpw
Switch(config-line)#login
Switch(config-line)#exit
Switch(config)#int vlan 1
Switch(config-if)#ip add 10.0.0.1 255.0.0.0
Switch(config-if)#no shutdown
```

Secure Remote Access: SSH Operation

Secure Shell (SSH) is a secure protocol that uses TCP port 22. It provides a secure (encrypted) management connection to a remote device. SSH should replace Telnet for management connections. SSH provides security for remote connections by providing strong encryption when a device is authenticated (username and password) and also for the transmitted data between the communicating devices.

```
Switch(config)# hostname S1
S1(config)#ip domain-name esna.aau.edu.et
S1(config)#crypt key generate rsa
S1(config)#ip ssh version 2
S1(config)#username admin secret MyPwd
S1(config)#line vty 5 15
S1(config-line)#transport input ssh
S1(config-line)#login local
```

VLAN Configuration

Catalyst switches 2960 and 3650 support over 4000 VLANs.

```
Switch# show vlan brief

VLAN Name                Status    Ports
---- ------------------- --------- -------------------
1    default             active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                   Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                   Fa0/9, Fa0/10, Fa0/11, Fa0/12
                                   Fa0/13, Fa0/14, Fa0/15, Fa0/16
                                   Fa0/17, Fa0/18, Fa0/19, Fa0/20
                                   Fa0/21, Fa0/22, Fa0/23, Fa0/24
                                   Gi0/1, Gi0/2
1002 fddi-default                       act/unsup
1003 token-ring-default                 act/unsup
1004 fddinet-default                    act/unsup
1005 trnet-default                      act/unsup
```

Normal Range VLAN 1 – 1005	Extended Range VLAN 1006 - 4095
Used in Small to Medium sized businesses	Used by Service Providers
1002 – 1005 are reserved for legacy VLANs	Are in Running-Config
1, 1002 – 1005 are auto created and cannot be deleted	Supports fewer VLAN features
Stored in the vlan.dat file in flash	Requires VTP configurations
VTP can synchronize between switches	

VLAN Configuration

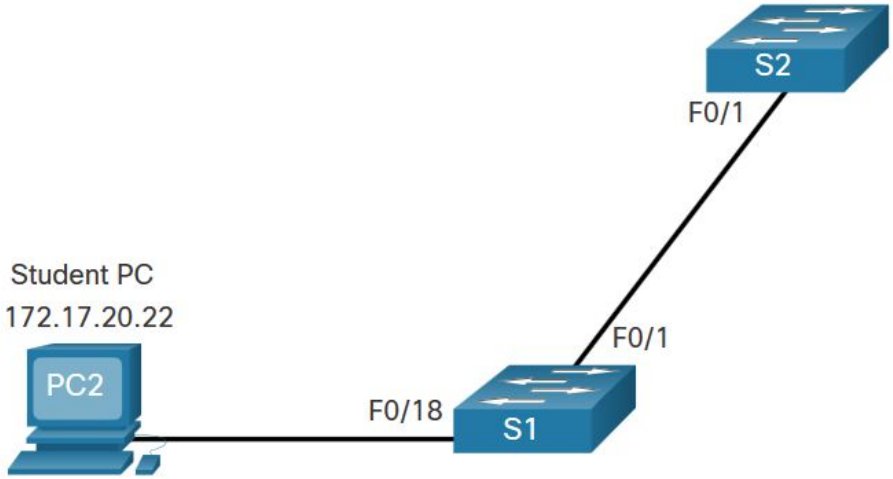
VLAN details are stored in the vlan.dat file. You create VLANs in the global configuration mode.

Task	IOS Command
Enter global configuration mode.	Switch# **configure terminal**
Create a VLAN with a valid ID number.	Switch(config)# **vlan** *vlan-id*
Specify a unique name to identify the VLAN.	Switch(config-vlan)# **name** *vlan-name*
Return to the privileged EXEC mode.	Switch(config-vlan)# **end**
Enter global configuration mode.	Switch# **configure terminal**

VLAN Configuration

- If the Student PC is going to be in VLAN 20, we will create the VLAN first and then name it.
- If you do not name it, the Cisco IOS will give it a default name of vlan and the four digit number of the VLAN. E.g. vlan0020 for VLAN 20.

Prompt	Command
S1#	Configure terminal
S1(config)#	vlan 20
S1(config-vlan)#	name student
S1(config-vlan)#	end



VLAN Configuration

Once the VLAN is created, we can then assign it to the correct interfaces.

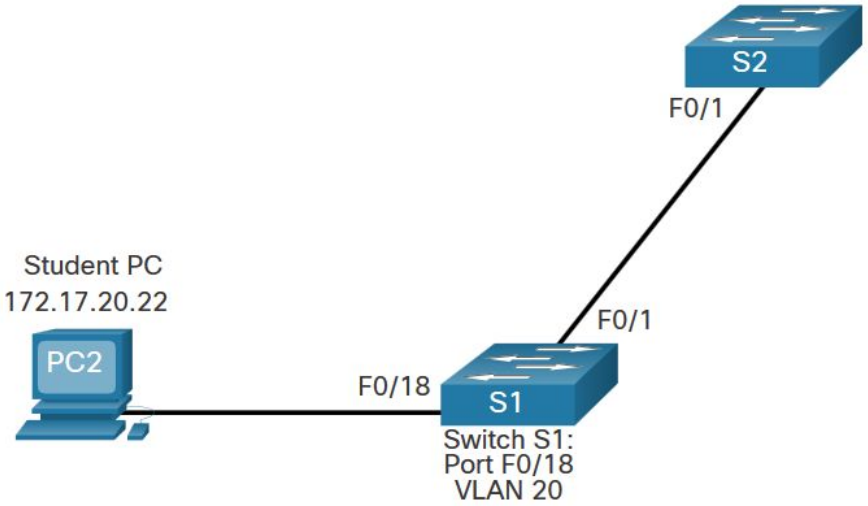
Task	Command
Enter global configuration mode.	Switch# **configure terminal**
Enter interface configuration mode.	Switch(config)# **interface** *interface-id*
Set the port to access mode.	Switch(config-if)# **switchport mode access**
Assign the port to a VLAN.	Switch(config-if)# **switchport access vlan** *vlan-id*
Return to the privileged EXEC mode.	Switch(config-if)# **end**

VLAN Configuration

We can assign the VLAN to the port interface.

- Once the device is assigned the VLAN, then the end device will need the IP address information for that VLAN
- Here, Student PC receives 172.17.20.22

Prompt	Command
S1#	Configure terminal
S1(config)#	Interface fa0/18
S1(config-if)#	Switchport mode access
S1(config-if)#	Switchport access vlan 20
S1(config-if)#	end



VLAN Configuration

Use the **show vlan** command. The complete syntax is:

show vlan [brief | **id** *vlan-id* | **name** *vlan-name* | **summary**]

```
S1# show vlan summary
Number of existing VLANs              : 7
 Number of existing VTP VLANs         : 7
 Number of existing extended VLANS    : 0
```

```
S1# show interface vlan 20
Vlan20 is up, line protocol is up
  Hardware is EtherSVI, address is 001f.6ddb.3ec1 (bia 001f.6ddb.3ec1)
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set

(Output omitted)
```

Task	Command Option
Display VLAN name, status, and its ports one VLAN per line.	**brief**
Display information about the identified VLAN ID number.	**id** *vlan-id*
Display information about the identified VLAN name. The *vlan-name* is an ASCII string from 1 to 32 characters.	**name** *vlan-name*
Display VLAN summary information.	**summary**

VLAN Configuration

There are a number of ways to change VLAN membership:

- re-enter **switchport access vlan** *vlan-id* command
- use the **no switchport access vlan** to place interface back in VLAN 1

Use the **show vlan brief** or the **show interface fa0/18 switchport** commands to verify the correct VLAN association.

```
S1(config)# interface fa0/18
S1(config-if)# no switchport access vlan
S1(config-if)# end
S1#
S1# show vlan brief
VLAN Name                 Status    Ports
---- -------------------- --------- -------------------------------
1    default              active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                    Fa0/5, Fa0/6, Fa0/7, Fa0/8
                                    Fa0/9, Fa0/10, Fa0/11, Fa0/12
                                    Fa0/13, Fa0/14, Fa0/15, Fa0/16
                                    Fa0/17, Fa0/18, Fa0/19, Fa0/20
                                    Fa0/21, Fa0/22, Fa0/23, Fa0/24
                                    Gi0/1, Gi0/2
20   student              active
1002 fddi-default         act/unsup
1003 token-ring-default   act/unsup
1004 fddinet-default      act/unsup
1005 trnet-default        act/unsup
```

```
S1# show interfaces fa0/18 switchport
Name: Fa0/18
Switchport: Enabled
Administrative Mode: static access
Operational Mode: static access
Administrative Trunking Encapsulation: negotiate
Operational Trunking Encapsulation: native
Negotiation of Trunking: Off
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 1 (default)
```

VLAN Configuration

Delete VLANs with the **no vlan** *vlan-id* command.

Caution: Before deleting a VLAN, reassign all member ports to a different VLAN.

- Delete all VLANs with the **delete flash:vlan.dat** or **delete vlan.dat** commands.
- Reload the switch when deleting all VLANs.

Note: To restore to factory default – unplug all data cables, erase the startup-configuration and delete the vlan.dat file, then reload the device.

VLAN Trunks Configuration

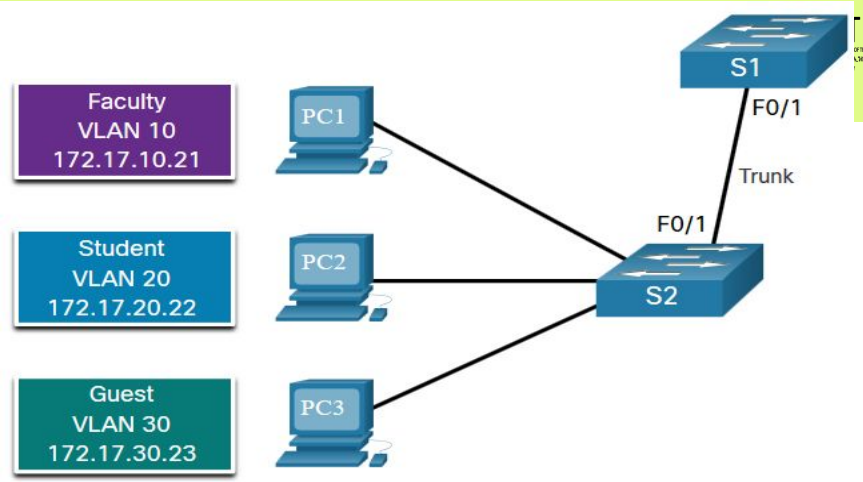
Configure and verify VLAN trunks. Trunks are layer 2 and carry traffic for all VLANs.

Task	IOS Command
Enter global configuration mode.	Switch# **configure terminal**
Enter interface configuration mode.	Switch(config)# **interface** *interface-id*
Set the port to permanent trunking mode.	Switch(config-if)# **switchport mode trunk**
Sets the native VLAN to something other than VLAN 1.	Switch(config-if)# **switchport trunk native vlan** *vlan-id*
Specify the list of VLANs to be allowed on the trunk link.	Switch(config-if)# **switchport trunk allowed vlan** *vlan-list*
Return to the privileged EXEC mode.	Switch(config-if)# **end**

VLAN Trunks Configuration

The subnets associated with each VLAN are:

- VLAN 10 - Faculty/Staff - 172.17.10.0/24
- VLAN 20 - Students - 172.17.20.0/24
- VLAN 30 - Guests - 172.17.30.0/24
- VLAN 99 - Native - 172.17.99.0/24



F0/1 port on S1 is configured as a trunk port.

Note: This assumes a 2960 switch using 802.1q tagging. Layer 3 switches require the encapsulation to be configured before the trunk mode.

Prompt	Command
S1(config)#	Interface fa0/1
S1(config-if)#	Switchport mode trunk
S1(config-if)#	Switchport trunk native vlan 99
S1(config-if)#	Switchport trunk allowed vlan 10,20,30,99
S1(config-if)#	end

Implement Port Security

Layer 2 attacks are some of the easiest for hackers to deploy but these threats can also be mitigated with some common Layer 2 solutions.

- All switch ports (interfaces) should be secured before the switch is deployed for production use. How a port is secured depends on its function.
- A simple method that many administrators use to help secure the network from unauthorized access is to disable all unused ports on a switch. Navigate to each unused port and issue the Cisco IOS **shutdown** command. If a port must be reactivated at a later time, it can be enabled with the **no shutdown** command.
- To configure a range of ports, use the **interface range** command.

```
Switch(config)# interface range type module/first-number - last-number
```

Implement Port Security

The simplest and most effective method to prevent MAC address table overflow attacks is to enable port security.

- Port security limits the number of valid MAC addresses allowed on a port. It allows an administrator to manually configure MAC addresses for a port or to permit the switch to dynamically learn a limited number of MAC addresses. When a port configured with port security receives a frame, the source MAC address of the frame is compared to the list of secure source MAC addresses that were manually configured or dynamically learned on the port.
- By limiting the number of permitted MAC addresses on a port to one, port security can be used to control unauthorized access to the network.

Implement Port Security

- Port security is enabled with the **switchport port-security** interface configuration command.
- Notice in the example, the **switchport port-security** command was rejected. This is because port security can only be configured on manually configured access ports or manually configured trunk ports. By default, Layer 2 switch ports are set to dynamic auto (trunking on). Therefore, in the example, the port is configured with the **switchport mode access** interface configuration command.

```
S1(config)# interface f0/1
S1(config-if)# switchport port-security
Command rejected: FastEthernet0/1 is a dynamic port.
S1(config-if)# switchport mode access
S1(config-if)# switchport port-security
S1(config-if)# end
S1#
```

Implement Port Security

After port security is enabled, other port security specifics can be configured, as shown in the example.

```
Switch(config-if)# switchport port-security maximum value
```

```
S1(config-if)# switchport port-security ?
  aging          Port-security aging commands
  mac-address    Secure mac address
  maximum        Max secure addresses
  violation      Security violation mode
  <cr>
S1(config-if)# switchport port-security
```

To set the maximum number of MAC addresses allowed on a port, use the following command:

- The default port security value is 1.
- The maximum number of secure MAC addresses that can be configured depends the switch and the IOS.
- In this example, the maximum is 8192.

```
S1(config)# interface f0/1
S1(config-if)# switchport port-security maximum ?
  <1-8192>  Maximum addresses
S1(config-if)# switchport port-security maximum
```

Implement Port Security

The switch can be configured to learn about MAC addresses on a secure port in one of three ways:

1. Manually Configured: The administrator manually configures a static MAC address(es) by using the following command for each secure MAC address on the port:

```
Switch(config-if)# switchport port-security mac-address mac-address
```

2. Dynamically Learned: When the **switchport port-security** command is entered, the current source MAC for the device connected to the port is automatically secured but is not added to the running configuration. If the switch is rebooted, the port will have to re-learn the device's MAC address.

3. Dynamically Learned – Sticky: The administrator can enable the switch to dynamically learn the MAC address and "stick" them to the running configuration by using the following command:

```
Switch(config-if)# switchport port-security mac-address sticky
```

Saving the running configuration will commit the dynamically learned MAC address to NVRAM.

Implement Port Security

The example demonstrates a complete port security configuration for FastEthernet 0/1.

- The administrator specifies a maximum of 4 MAC addresses, manually configures one secure MAC address, and then configures the port to dynamically learn additional secure MAC addresses up to the 4 secure MAC address maximum.
- Use the **show port-security interface** and the **show port-security address** command to verify the configuration.

```
S1(config)# interface fa0/1
S1(config-if)# switchport mode access
S1(config-if)# switchport port-security
S1(config-if)# switchport port-security maximum 4
S1(config-if)# switchport port-security mac-address aaaa.bbbb.1234
S1(config-if)# switchport port-security mac-address sticky
S1(config-if)# end
S1# show port-security interface fa0/1
Port Security              : Enabled
Port Status                : Secure-up
Violation Mode             : Shutdown
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 4
Total MAC Addresses        : 1
Configured MAC Addresses   : 1
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 0000.0000.0000:0
Security Violation Count   : 0
S1# show port-security address
              Secure Mac Address Table
------------------------------------------------------------------------
Vlan    Mac Address       Type                          Ports   Remaining Age
                                                                  (mins)
----    -----------       ----                          -----   -------------
   1    aaaa.bbbb.1234    SecureConfigured              Fa0/1         -
------------------------------------------------------------------------
Total Addresses in System (excluding one mac per port)     : 0
Max Addresses limit in System (excluding one mac per port) : 8192

S1#
```

Implement Port Security

Port security aging can be used to set the aging time for static and dynamic secure addresses on a port and two types of aging are supported per port:

- **Absolute** - The secure addresses on the port are deleted after the specified aging time.
- **Inactivity** - The secure addresses on the port are deleted if they are inactive for a specified time.

Use aging to remove secure MAC addresses on a secure port without manually deleting the existing secure MAC addresses.

- Aging of statically configured secure addresses can be enabled or disabled on a per-port basis.

Use the **switchport port-security aging** command to enable or disable static aging for the secure port, or to set the aging time or type.

```
Switch(config-if)# switchport port-security aging {static | time time | type {absolute | inactivity}}
```

Implement Port Security

The example shows an administrator configuring the aging type to 10 minutes of inactivity.

The **show port-security** command confirms the changes. **interface** command to verify the configuration.

```
S1(config)# interface fa0/1
S1(config-if)# switchport port-security aging time 10
S1(config-if)# switchport port-security aging type inactivity
S1(config-if)# end
S1# show port-security interface fa0/1
Port Security              : Enabled
Port Status                : Secure-shutdown
Violation Mode             : Restrict
Aging Time                 : 10 mins
Aging Type                 : Inactivity
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 4
Total MAC Addresses        : 1
Configured MAC Addresses   : 1
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 0050.56be.e4dd:1
Security Violation Count   : 1
```

Implement Port Security

If the MAC address of a device attached to a port differs from the list of secure addresses, then a port violation occurs and the port enters the error-disabled state.

- To set the port security violation mode, use the following command:

```
Switch(config-if)# switchport port-security violation {shutdown | restrict | protect}
```

The following table shows how a switch reacts based on the configured violation mode.

Mode	Description
shutdown (default)	The port transitions to the error-disabled state immediately, turns off the port LED, and sends a syslog message. It increments the violation counter. When a secure port is in the error-disabled state, an administrator must re-enable it by entering the **shutdown** and **no shutdown** commands.
restrict	The port drops packets with unknown source addresses until you remove a sufficient number of secure MAC addresses to drop below the maximum value or increase the maximum value. This mode causes the Security Violation counter to increment and generates a syslog message.
protect	This is the least secure of the security violation modes. The port drops packets with unknown MAC source addresses until you remove a sufficient number of secure MAC addresses to drop below the maximum value or increase the maximum value. No syslog message is sent.

ROUTER CONFIGURATION

Basic Router Configuration

- Cisco routers and Cisco switches have many similarities.
- They support a similar modal operating system, similar command structures, and many of the same commands.
- In addition, both devices have similar initial configuration steps.
- For example, the following configuration tasks should always be performed.
- Name the device to distinguish it from other routers and configure passwords, as shown in the example.

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# hostname R1
R1(config)# enable secret class
R1(config)# line console 0
R1(config-line)# password cisco
R1(config-line)# login
R1(config-line)# exit
R1(config)# line vty 0 4
R1(config-line)# password cisco
R1(config-line)# login
R1(config-line)# exit
R1(config)# service password-encryption
R1(config)#
```

Basic Router Configuration

Configure a banner to provide legal notification of unauthorized access, as shown in the example.

```
R1(config)# banner motd $ Authorized Access Only! $
R1(config)#
```

Save the changes on a router, as shown in the example.

```
R1# copy running-config startup-config
Destination filename [startup-config]?
Building configuration...
[OK]
```

Basic Router Configuration: Interfaces

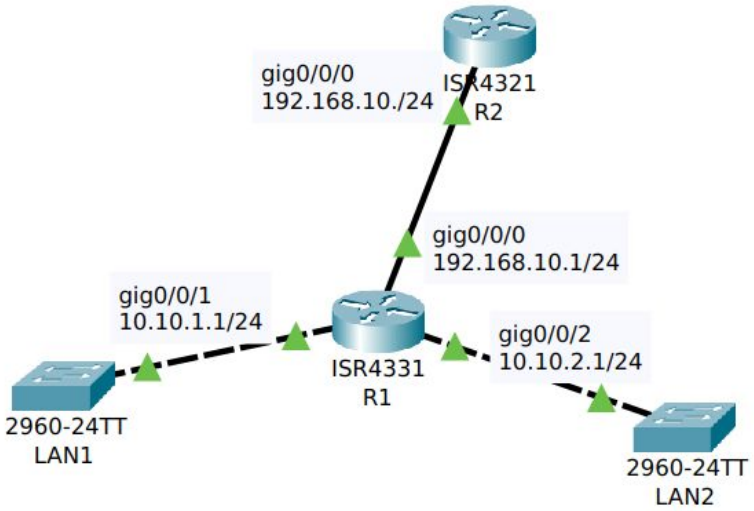
Routers support LANs and WANs and can interconnect different types of networks; therefore, they support many types of interfaces.

To be available, an interface must be:

- **Configured with at least one IP address** - Use the **ip address** *ip-address subnet-mask* interface configuration commands.
- **Activated** - By default, LAN and WAN interfaces are not activated (**shutdown**). To enable an interface, it must be activated using the **no shutdown** command.
    - It is similar to powering on the interface
    - The interface must also be connected to another device (a hub, a switch, or another router) for the physical layer to be active.
- **Description** - Optionally, the interface could also be configured with a short description of up to 240 characters. It is good practice to configure a description on each interface. On production networks, the benefits of interface descriptions are quickly realized as they are helpful in troubleshooting and in identifying a third-party connection and contact information.

Basic Router Configuration: Interfaces

The example shows the configuration for the interfaces on R1:




```
Router#configure terminal
Router(config)#hostname R1
R1(config)#interface gigabitEthernet 0/0/0
R1(config-if)#description Link TO R2
R1(config-if)#ip address 192.168.10.1 255.255.255.0
R1(config-if)#no shutdown
R1(config-if)#exit

R1(config)#interface gigabitEthernet 0/0/1
R1(config-if)#description Link TO LAN1
R1(config-if)#ip address 10.20.1.1 255.255.255.0
R1(config-if)#no shutdown
R1(config-if)#exit

R1(config)#interface gigabitEthernet 0/0/2
R1(config-if)#description Link TO LAN2
R1(config-if)#ip address 10.10.2.1 255.255.255.0
R1(config-if)#no shutdown
```

Basic Router Configuration: Loopback Interfaces

Another common configuration of Cisco IOS routers is enabling a loopback interface.

- The loopback interface is a logical interface that is internal to the router. It is not assigned to a physical port and can never be connected to any other device. It is considered a software interface that is automatically placed in an “up” state, as long as the router is functioning.
- The loopback interface is useful in testing and managing a Cisco IOS device because it ensures that at least one interface will always be available. For example, it can be used for testing purposes, such as testing internal routing processes, by emulating networks behind the router.
- Loopback interfaces are also commonly used in lab environments to create additional interfaces. For example, you can create multiple loopback interfaces on a router to simulate more networks for configuration practice and testing purposes. The IPv4 address for each loopback interface must be unique and unused by any other interface.
- Enabling and assigning a loopback address is simple:

```
Router(config)# interface loopback number
Router(config-if)# ip address ip-address subnet-mask
```

Interface Verification Commands

There are several **show** commands that can be used to verify the operation and configuration of an interface.

The following commands are especially useful to quickly identify the status of an interface:

- **show ip interface brief** - These display a summary for all interfaces including the IPv4 address of the interface and current operational status.
- **show running-config interface** *interface-id* - This displays the commands applied to the specified interface.
- **show ip route** - These display the contents of the IPv4 routing table stored in RAM.
    - In Cisco IOS 15, active interfaces should appear in the routing table with two related entries identified by the code '**C**' (Connected) or '**L**' (Local).

Verify Interface Status

The output of the **show ip interface brief** command can be used to quickly reveal the status of all interfaces on the router. You can verify that the interfaces are active and operational as indicated by the Status of "up" and Protocol of "up", as shown in the example. A different output would indicate a problem with either the configuration or the cabling.

```
R1#show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0/0   192.168.10.1    YES manual up                    up
GigabitEthernet0/0/1   10.20.1.1       YES manual up                    up
GigabitEthernet0/0/2   10.10.2.1       YES manual up                    up
Vlan1                  unassigned      YES unset  administratively down down
R1#show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
C       10.10.2.0/24 is directly connected, GigabitEthernet0/0/2
L       10.10.2.1/32 is directly connected, GigabitEthernet0/0/2
C       10.20.1.0/24 is directly connected, GigabitEthernet0/0/1
L       10.20.1.1/32 is directly connected, GigabitEthernet0/0/1
     192.168.10.0/24 is variably subnetted, 2 subnets, 2 masks
C       192.168.10.0/24 is directly connected, GigabitEthernet0/0/0
L       192.168.10.1/32 is directly connected, GigabitEthernet0/0/0
```

Verify Interface Configuration

The output of the **show running-config interface** command displays the current commands applied to the specified interface, as shown.

The following two commands are used to gather more detailed interface information:

- **show interfaces**- Displays interface information and packet flow count for all interfaces on the device.
- **show ip interface** - Displays the IPv4 related information for all interfaces on a router..

```
R1#show ip interface gigabitEthernet 0/0/0
GigabitEthernet0/0/0 is up, line protocol is up (connected)
  Internet address is 192.168.10.1/24
  Broadcast address is 255.255.255.255
  Address determined by setup command
  MTU is 1500 bytes
  Helper address is not set
  Directed broadcast forwarding is disabled
  Outgoing access list is not set
  Inbound  access list is not set
  Proxy ARP is enabled
  Security level is default
  Split horizon is enabled
  ICMP redirects are always sent
  ICMP unreachables are always sent
  ICMP mask replies are never sent
  IP fast switching is disabled
  IP fast switching on the same interface is disabled
  IP Flow switching is disabled
  IP Fast switching turbo vector
  IP multicast fast switching is disabled
  IP multicast distributed fast switching is disabled
  Router Discovery is disabled
  IP output packet accounting is disabled
  IP access violation accounting is disabled
  TCP/IP header compression is disabled
  RTP/IP header compression is disabled
  Probe proxy name replies are disabled
  Policy routing is disabled
  Network address translation is disabled
  BGP Policy Mapping is disabled
  Input features: MCI Check
  WCCP Redirect outbound is disabled
  WCCP Redirect inbound is disabled
  WCCP Redirect exclude is disabled
```

Inter-VLAN Routing Operation

VLANs are used to segment switched Layer 2 networks for a variety of reasons. Regardless of the reason, hosts in one VLAN cannot communicate with hosts in another VLAN unless there is a router or a Layer 3 switch to provide routing services.

Inter-VLAN routing is the process of forwarding network traffic from one VLAN to another VLAN.

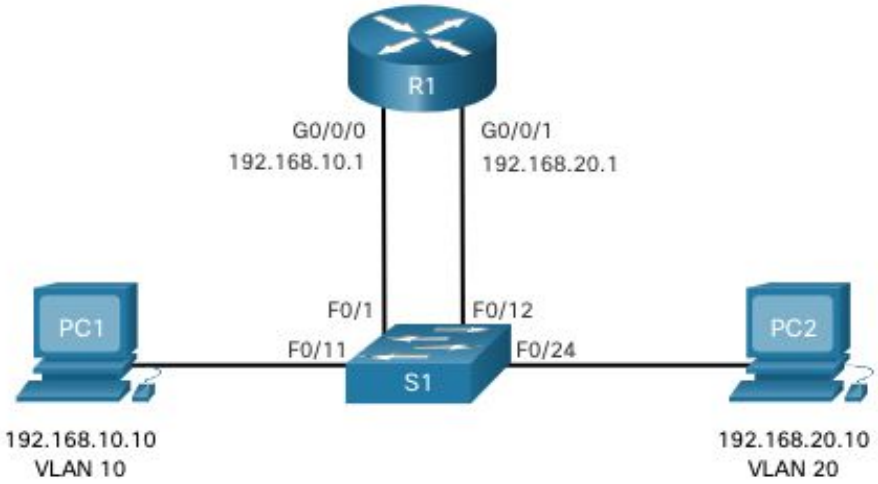
There are three inter-VLAN routing options:

- **Legacy Inter-VLAN routing** - This is a legacy solution. It does not scale well.
- **Router-on-a-Stick** - This is an acceptable solution for a small to medium-sized network.
- **Layer 3 switch using switched virtual interfaces (SVIs)** - This is the most scalable solution for medium to large organizations.

Inter-VLAN Routing Operation: Legacy

- The first inter-VLAN routing solution relied on using a router with multiple Ethernet interfaces. Each router interface was connected to a switch port in different VLANs. The router interfaces served as the default gateways to the local hosts on the VLAN subnet.
- Legacy inter-VLAN routing using physical interfaces works, but it has a significant limitation. It is not reasonably scalable because routers have a limited number of physical interfaces. Requiring one physical router interface per VLAN quickly exhausts the physical interface capacity of a router.

- **Note**: This method of inter-VLAN routing is no longer implemented in switched networks and is included for explanation purposes only.



Inter-VLAN Routing Operation: Router-on-a-Stick

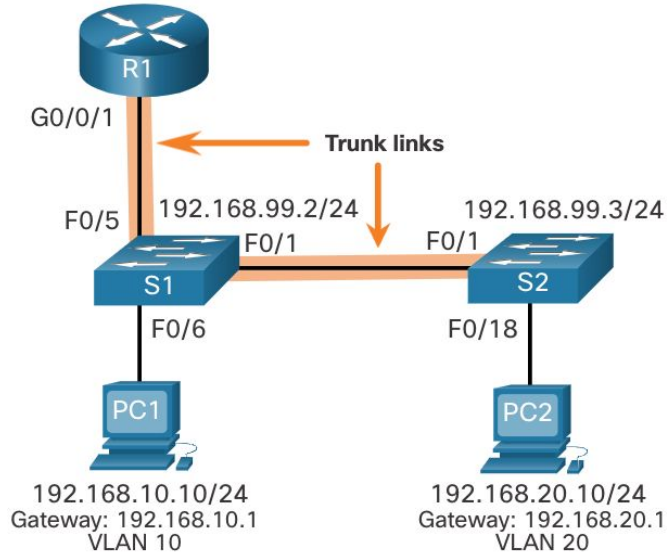
The ‘router-on-a-stick’ inter-VLAN routing method overcomes the limitation of the legacy inter-VLAN routing method. It only requires one physical Ethernet interface to route traffic between multiple VLANs on a network.

- A Cisco IOS router Ethernet interface is configured as an 802.1Q trunk and connected to a trunk port on a Layer 2 switch. Specifically, the router interface is configured using subinterfaces to identify routable VLANs.
- The configured subinterfaces are software-based virtual interfaces.
    - Each is associated with a single physical Ethernet interface.
    - Each subinterface is independently configured with an IP address and VLAN assignment.
    - Subinterfaces are configured for different subnets that correspond to their VLAN assignment.
- When VLAN-tagged traffic enters the router interface, it is forwarded to the VLAN subinterface.
    - After a routing decision is made based on the destination IP network address, the router determines the exit interface for the traffic.
    - If the exit interface is configured as an 802.1q subinterface, the data frames are VLAN-tagged with the new VLAN and sent back out the physical interface

Note: The router-on-a-stick method of inter-VLAN routing does not scale beyond 50 VLANs.

Inter-VLAN Routing Operation: Router-on-a-Stick

- In the figure, the R1 GigabitEthernet 0/0/1 interface is connected to the S1 FastEthernet 0/5 port. The S1 FastEthernet 0/1 port is connected to the S2 FastEthernet 0/1 port. These are trunk links that are required to forward traffic within and between VLANs.
- To route between VLANs, the R1 GigabitEthernet 0/0/1 interface is logically divided into three subinterfaces, as shown in the table. The table also shows the three VLANs that will be configured on the switches.
- Assume that R1, S1, and S2 have initial basic configurations. Currently, PC1 and PC2 cannot **ping** each other because they are on separate networks. Only S1 and S2 can **ping** each other, but they but are unreachable by PC1 or PC2 because they are also on different networks.
- To enable devices to ping each other, the switches must be configured with VLANs and trunking, and the router must be configured for inter-VLAN routing.



Subinterface	VLAN	IP Address
G0/0/1.10	10	192.168.10.1/24
G0/0/1.20	20	192.168.20.1/24
G0/0/1.30	99	192.168.99.1/24

Inter-VLAN Routing Operation: Router-on-a-Stick

```
S2(config)# vlan 10
S2(config-vlan)# name LAN10
S2(config-vlan)# exit
S2(config)# vlan 20
S2(config-vlan)# name LAN20
S2(config-vlan)# exit
S2(config)# vlan 99
S2(config-vlan)# name Management
S2(config-vlan)# exit
S2(config)#
S2(config)# interface vlan 99
S2(config-if)# ip add 192.168.99.3 255.255.255.0
S2(config-if)# no shut
S2(config-if)# exit
S2(config)# ip default-gateway 192.168.99.1
S2(config)# interface fa0/18
S2(config-if)# switchport mode access
S2(config-if)# switchport access vlan 20
S2(config-if)# no shut
S2(config-if)# exit
S2(config)# interface fa0/1
S2(config-if)# switchport mode trunk
S2(config-if)# no shut
S2(config-if)# exit
S2(config-if)# end
*Mar  1 00:23:52.137: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1,
changed state to up
```

Inter-VLAN Routing Operation: Router-on-a-Stick

The router-on-a-stick method requires you to create a subinterface for each VLAN to be routed. A subinterface is created using the **interface** *interface_id subinterface_id* global configuration mode command. The subinterface syntax is the physical interface followed by a period and a subinterface number. Although not required, it is customary to match the subinterface number with the VLAN number.

Each subinterface is then configured with the following two commands:

- **encapsulation dot1q** *vlan_id* **[native]** - This command configures the subinterface to respond to 802.1Q encapsulated traffic from the specified *vlan-id*. The **native** keyword option is only appended to set the native VLAN to something other than VLAN 1.
- **ip address** *ip-address subnet-mask* - This command configures the IPv4 address of the subinterface. This address typically serves as the default gateway for the identified VLAN.

Repeat the process for each VLAN to be routed. Each router subinterface must be assigned an IP address on a unique subnet for routing to occur. When all subinterfaces have been created, enable the physical interface using the **no shutdown** interface configuration command. If the physical interface is disabled, all subinterfaces are disabled.

Inter-VLAN Routing Operation: Router-on-a-Stick

In the configuration, the R1 G0/0/1 subinterfaces are configured for VLANs 10, 20, and 99.

```
R1(config)# interface G0/0/1.10
R1(config-subif)# Description Default Gateway for VLAN 10
R1(config-subif)# encapsulation dot1Q 10
R1(config-subif)# ip add 192.168.10.1 255.255.255.0
R1(config-subif)# exit
R1(config)#
R1(config)# interface G0/0/1.20
R1(config-subif)# Description Default Gateway for VLAN 20
R1(config-subif)# encapsulation dot1Q 20
R1(config-subif)# ip add 192.168.20.1 255.255.255.0
R1(config-subif)# exit
R1(config)#
R1(config)# interface G0/0/1.99
R1(config-subif)# Description Default Gateway for VLAN 99
R1(config-subif)# encapsulation dot1Q 99
R1(config-subif)# ip add 192.168.99.1 255.255.255.0
R1(config-subif)# exit
R1(config)#
R1(config)# interface G0/0/1
R1(config-if)# Description Trunk link to S1
R1(config-if)# no shut
R1(config-if)# end
R1#
*Sep 15 19:08:47.015: %LINK-3-UPDOWN: Interface GigabitEthernet0/0/1, changed state to down
*Sep 15 19:08:50.071: %LINK-3-UPDOWN: Interface GigabitEthernet0/0/1, changed state to up
*Sep 15 19:08:51.071: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0/1,
changed state to up
R1#
```

Inter-VLAN Routing Operation: Router-on-a-Stick

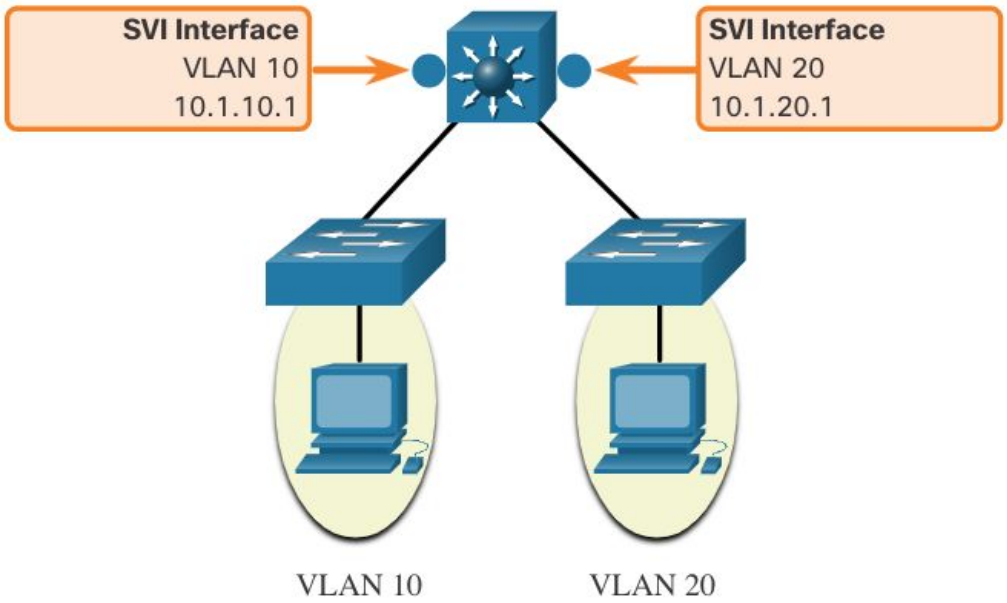
- From a host, verify connectivity to a host in another VLAN using the **ping** command.
- Next, use **ping** to verify connectivity with PC2 and S1, as shown in the figure. The **ping** output successfully confirms inter-VLAN routing is operating.
- In addition to using **ping** between devices, the following **show** commands can be used to verify and troubleshoot the router-on-a-stick configuration.
  - **show ip route**
  - **show ip interface brief**
  - **show interfaces**
  - **show interfaces trunk**

```
C:\Users\PC1> ping 192.168.20.10
Pinging 192.168.20.10 with 32 bytes of data:
Reply from 192.168.20.10: bytes=32 time<1ms TTL=127
Reply from 192.168.20.10: bytes=32 time<1ms TTL=127
Reply from 192.168.20.10: bytes=32 time<1ms TTL=127
Reply from 192.168.20.10: bytes=32 time<1ms TTL=127
Ping statistics for 192.168.20.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss).
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
C:\Users\PC1>
C:\Users\PC1> ping 192.168.99.2
Pinging 192.168.99.2 with 32 bytes of data:
Request timed out.
Request timed out.
Reply from 192.168.99.2: bytes=32 time=2ms TTL=254
Reply from 192.168.99.2: bytes=32 time=1ms TTL=254
Ping statistics for 192.168.99.2:
    Packets: Sent = 4, Received = 2, Lost = 2 (50% loss).
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 2ms, Average = 1ms
C:\Users\PC1>
```

Inter-VLAN Routing Operation: Layer 3 Switch

- The modern method of performing inter-VLAN routing is to use Layer 3 switches and **switched virtual interfaces (SVI)**.
- An SVI is a virtual interface that is configured on a Layer 3 switch, as shown in the figure.

- **Note**: A Layer 3 switch is also called a multilayer switch as it operates at Layer 2 and Layer 3.



Inter-VLAN Routing Operation: Layer 3 Switch

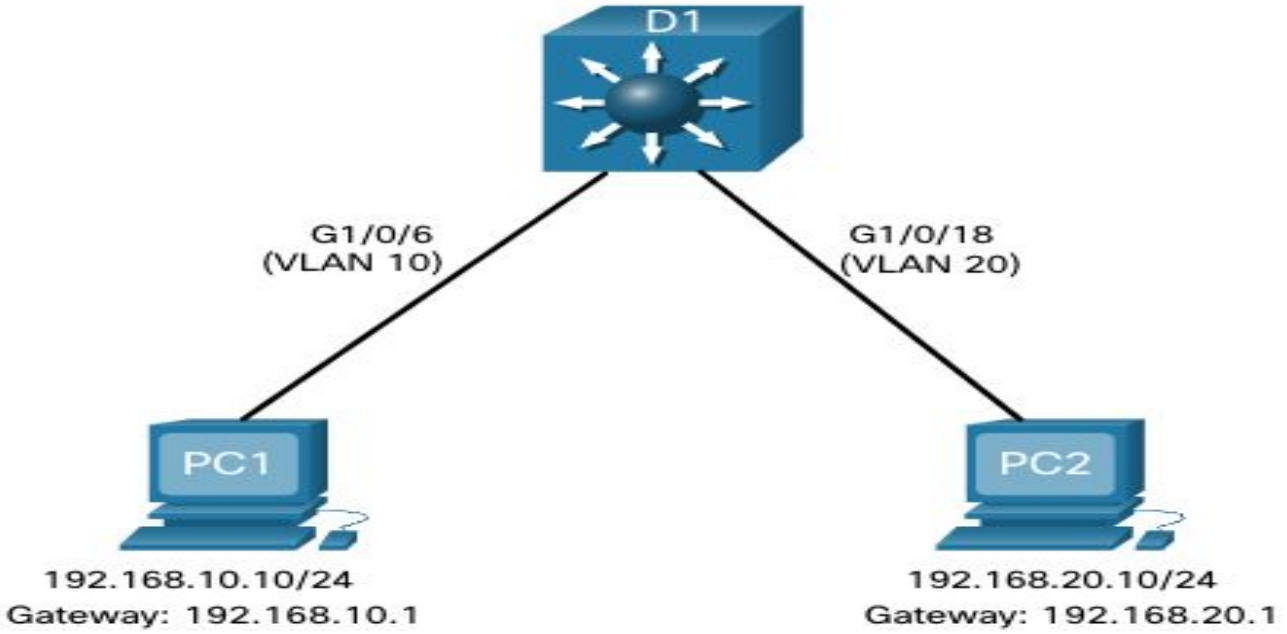
Inter-VLAN SVIs are created the same way that the management VLAN interface is configured. The SVI is created for a VLAN that exists on the switch. Although virtual, the SVI performs the same functions for the VLAN as a router interface would. Specifically, it provides Layer 3 processing for packets that are sent to or from all switch ports associated with that VLAN.

The following are advantages of using Layer 3 switches for inter-VLAN routing:

- They are much faster than router-on-a-stick because everything is hardware switched and routed.
- There is no need for external links from the switch to the router for routing.
- They are not limited to one link because Layer 2 EtherChannels can be used as trunk links between the switches to increase bandwidth.
- Latency is much lower because data does not need to leave the switch in order to be routed to a different network.
- They more commonly deployed in a campus LAN than routers.
- The only disadvantage is that **Layer 3 switches are more expensive**.

Inter-VLAN Routing Operation: Layer 3 Switch

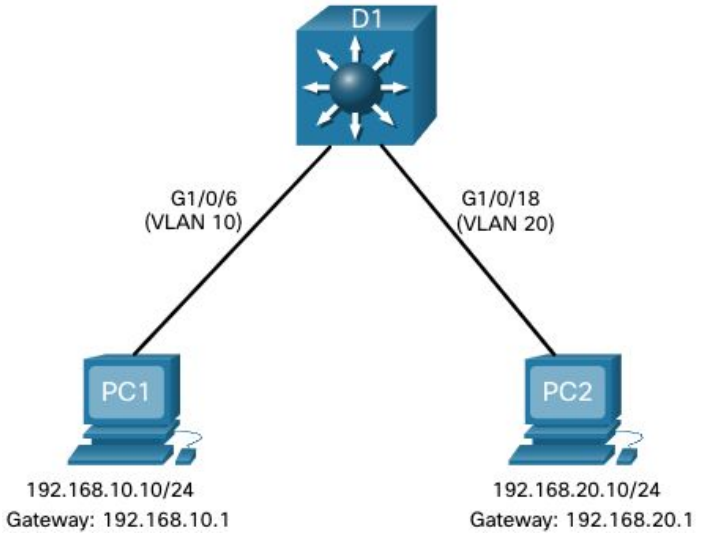
- In the figure, the Layer 3 switch, D1, is connected to two hosts on different VLANs. PC1 is in VLAN 10 and PC2 is in VLAN 20, as shown.
- The Layer 3 switch will provide inter-VLAN routing services to the two hosts.



Inter-VLAN Routing Operation: Layer 3 Switch

Complete the following steps to configure S1 with VLANs and trunking:

- **Step 1**. Create the VLANs. In the example, VLANs 10 and 20 are used.
- **Step 2**. Create the SVI VLAN interfaces. The IP address configured will serve as the default gateway for hosts in the respective VLAN.
- **Step 3**. Configure access ports. Assign the appropriate port to the required VLAN.
- **Step 4**. Enable IP routing. Issue the **ip routing** global configuration command to allow traffic to be exchanged between VLANs 10 and 20. This command must be configured to enable inter-VAN routing on a Layer 3 switch for IPv4.



Inter-VLAN Routing Operation: Layer 3 Switch

Inter-VLAN routing using a Layer 3 switch is simpler to configure than the router-on-a-stick method. After the configuration is complete, the configuration can be verified by testing connectivity between the hosts.

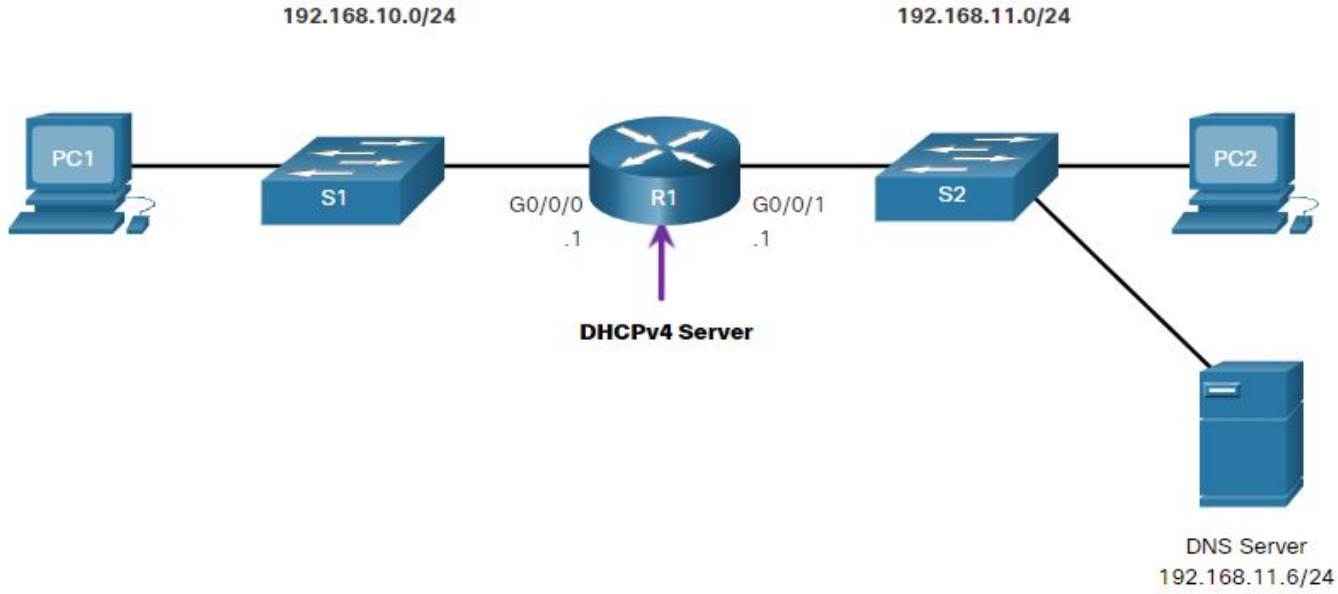
- From a host, verify connectivity to a host in another VLAN using the **ping** command. It is a good idea to first verify the current host IP configuration using the **ipconfig** Windows host command.
- Next, verify connectivity with PC2 using the **ping** Windows host command. The successful **ping** output confirms inter-VLAN routing is operating.

DHCPv4 Concepts: DHCPv4 Server and Client

- Dynamic Host Configuration Protocol v4 (DHCPv4) assigns IPv4 addresses and other network configuration information dynamically. Because desktop clients typically make up the bulk of network nodes, DHCPv4 is an extremely useful and timesaving tool for network administrators.
- The DHCPv4 server dynamically assigns, or leases, an IPv4 address from a pool of addresses for a limited period of time chosen by the server, or until the client no longer needs the address.
- Clients lease the information from the server for an administratively defined period. Administrators configure DHCPv4 servers to set the leases to time out at different intervals. The lease is typically anywhere from 24 hours to a week or more. When the lease expires, the client must ask for another address, although the client is typically reassigned the same address.

Configure DHCPv4 Server

Now you have a basic understanding of how DHCPv4 works and how it can make your job a bit easier. A Cisco router running Cisco IOS software can be configured to act as a DHCPv4 server. The Cisco IOS DHCPv4 server assigns and manages IPv4 addresses from specified address pools within the router to DHCPv4 clients.



Configure DHCPv4 Server

Use the following steps to configure a Cisco IOS DHCPv4 server:

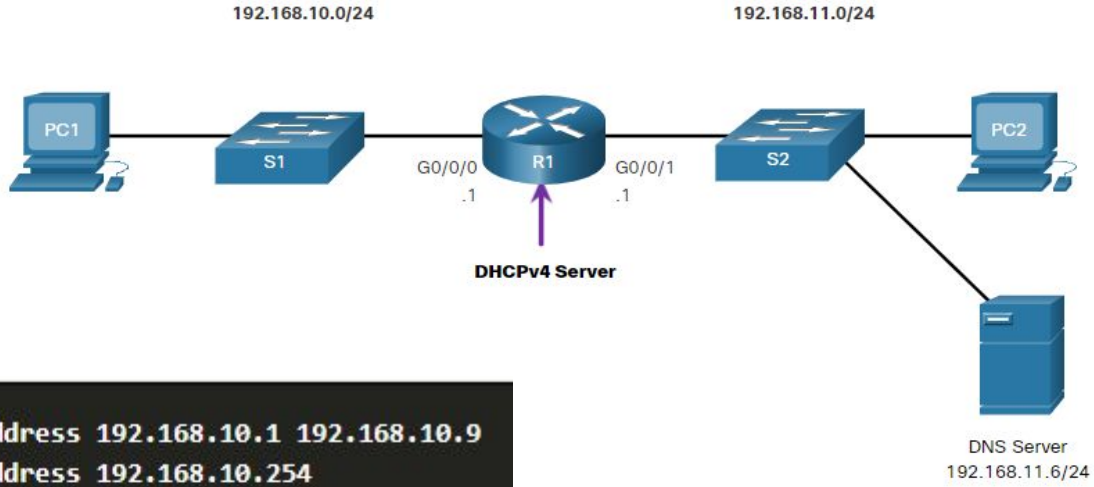
- **Step 1**. Exclude IPv4 addresses. A single address or a range of addresses can be excluded by specifying the *low-address* and *high-address* of the range. Excluded addresses should be those addresses that are assigned to routers, servers, printers, and other devices that have been, or will be, manually configured. You can also enter the command multiple times. The command is **ip dhcp excluded-address** ***low-address [high-address]***
- **Step 2**. Define a DHCPv4 pool name. The **ip dhcp pool** ***pool-name*** command creates a pool with the specified name and puts the router in DHCPv4 configuration mode, which is identified by the prompt **Router(dhcp-config)#.**

Configure DHCPv4 Server

- **Step 3**. Configure the DHCPv4 pool. The address pool and default gateway router must be configured. Use the **network** statement to define the range of available addresses. Use the **default-router** command to define the default gateway router. These commands and other optional commands are shown in the table.

Task	IOS Command
Define the address pool.	**network** *network-number* [*mask* | / *prefix-length*]
Define the default router or gateway.	**default-router** address [*address2….address8*]
Define a DNS server.	**dns-server** *address* [*address2…address8*]
Define the domain name.	**domain-name** *domain*
Define the duration of the DHCP lease.	**lease** {*days* [*hours* [*minutes*]] | **infinite**}
Define the NetBIOS WINS server.	**netbios-name-server** *address* [*address2…address8*]

Configure DHCPv4 Server




```
R1(config)# ip dhcp excluded-address 192.168.10.1 192.168.10.9
R1(config)# ip dhcp excluded-address 192.168.10.254
R1(config)# ip dhcp pool LAN-POOL-1
R1(dhcp-config)# network 192.168.10.0 255.255.255.0
R1(dhcp-config)# default-router 192.168.10.1
R1(dhcp-config)# dns-server 192.168.11.5
R1(dhcp-config)# domain-name example.com
R1(dhcp-config)# end
R1#
```

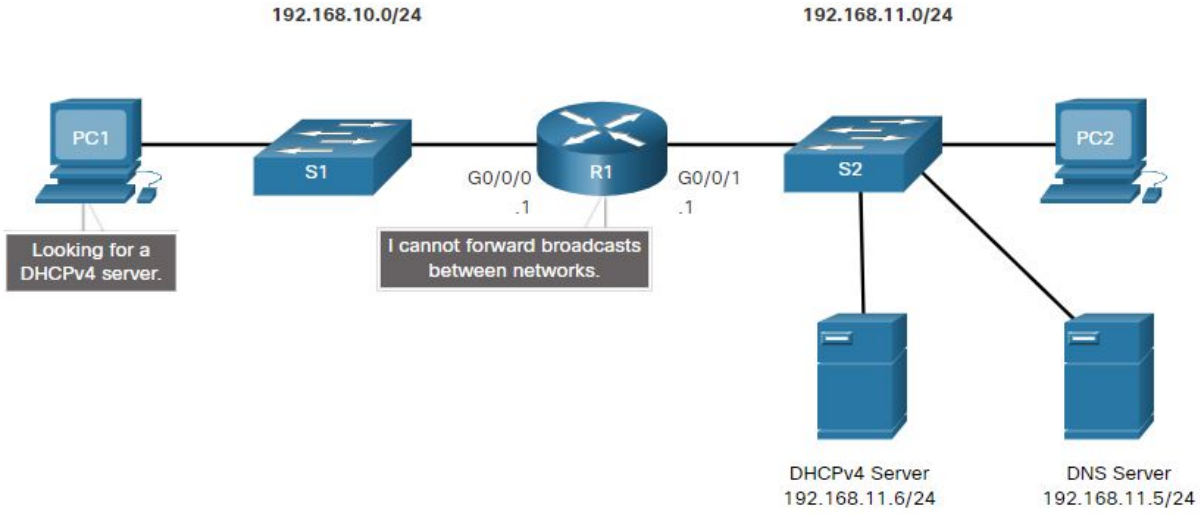
DHCPv4 Verification

Use the commands in the table to verify that the Cisco IOS DHCPv4 server is operational.

Command	Description
show running-config | section dhcp	Displays the DHCPv4 commands configured on the router.
show ip dhcp binding	Displays a list of all IPv4 address to MAC address bindings provided by the DHCPv4 service.
show ip dhcp server statistics	Displays count information regarding the number of DHCPv4 messages that have been sent and received

DHCPv4 Relay

- In a complex hierarchical network, enterprise servers are usually located centrally. These servers may provide DHCP, DNS, TFTP, and FTP services for the network. Network clients are not typically on the same subnet as those servers. In order to locate the servers and receive services, clients often use broadcast messages.
- In the figure, PC1 is attempting to acquire an IPv4 address from a DHCPv4 server using a broadcast message. In this scenario, R1 is not configured as a DHCPv4 server and does not forward the broadcast. Because the DHCPv4 server is located on a different network, PC1 cannot receive an IP address using DHCP. R1 must be configured to relay DHCPv4 messages to the DHCPv4 server.



DHCPv4 Relay

- Configure R1 with the **ip helper-address** *address* interface configuration command. This will cause R1 to relay DHCPv4 broadcasts to the DHCPv4 server. As shown in the example, the interface on R1 receiving the broadcast from PC1 is configured to relay DHCPv4 address to the DHCPv4 server at 192.168.11.6.
- When R1 has been configured as a DHCPv4 relay agent, it accepts broadcast requests for the DHCPv4 service and then forwards those requests as a unicast to the IPv4 address 192.168.11.6. The network administrator can use the **show ip interface** command to verify the configuration.

```
R1(config)# interface g0/0/0
R1(config-if)# ip helper-address 192.168.11.6
R1(config-if)# end
R1#
```

```
R1# show ip interface g0/0/0
GigabitEthernet0/0/0 is up, line protocol is up
  Internet address is 192.168.10.1/24
  Broadcast address is 255.255.255.255
  Address determined by setup command
  MTU is 1500 bytes
  Helper address is 192.168.11.6
(output omitted)
```

Routing

- When a router receives an IP packet on one interface, it determines which interface to use to forward the packet to the destination.
  - This is known as **routing**.
  - The interface that the router uses to forward the packet may be the final destination, or it may be a network connected to another router that is used to reach the destination network.
  - Each network that a router connects to typically requires a separate interface, but this may not always be the case.

- The two primary functions of a router are to **determine the best path** to forward packets based on the information in its routing table, and to **forward packets** toward their destination.

Static Routes

Static routes are commonly implemented on a network. This is true even when there is a dynamic routing protocol configured.

CISCO routers support the following types of static routes:

- Standard static route
- Default static route
- Floating static route
- Summary static route

Static routes are configured using the **ip route** global configuration commands.

Static Routes: Next-Hop Options

When configuring a static route, the next hop can be identified by an IP address, exit interface, or both. How the destination is specified creates one of the three following types of static route:

- **Next-hop route** - Only the next-hop IP address is specified
- **Directly connected static route** - Only the router exit interface is specified
- **Fully specified static route** - The next-hop IP address and exit interface are specified

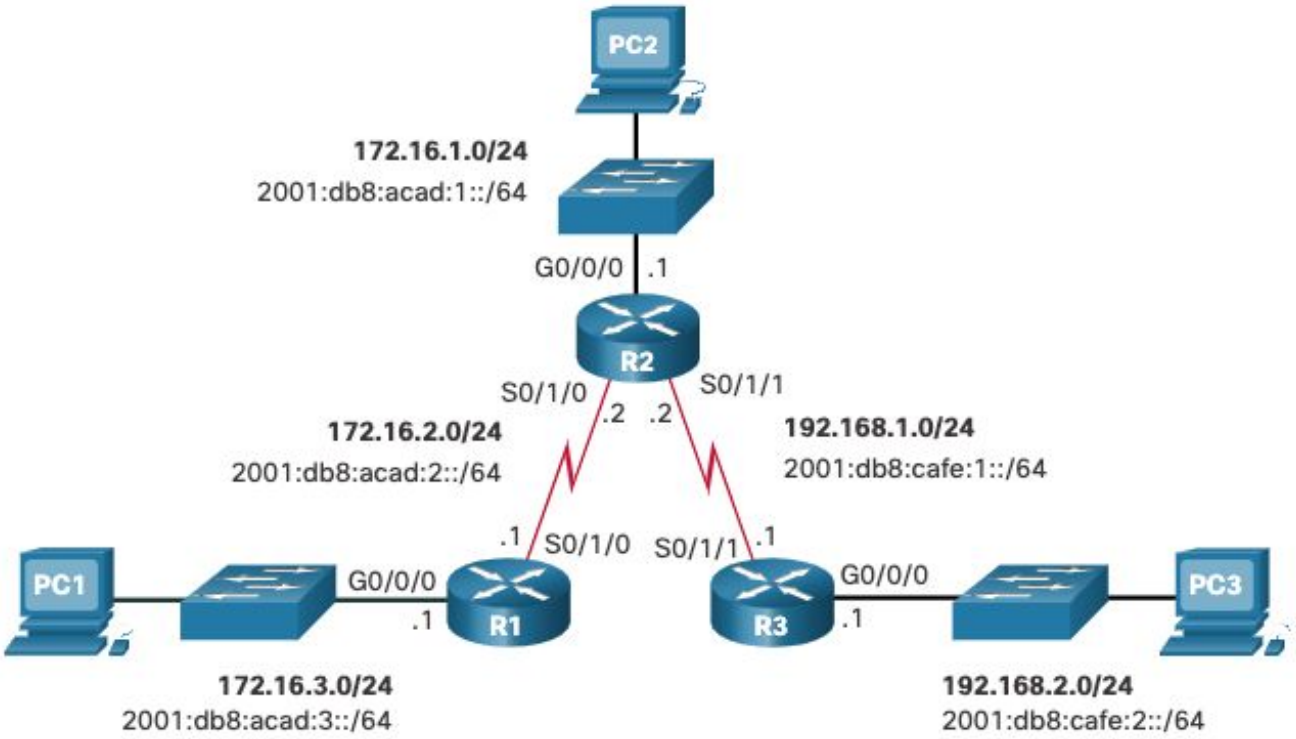
IPv4 static routes are configured using the following global configuration command:

```
Router(config)# ip route network-address subnet-mask { ip-address |
exit-intf [ip-address]} [distance]
```

Note: Either the *ip-address*, *exit-intf,* or the *ip-address* and *exit-intf* parameters must be configured.

Static Routes: Dual-Stack Topology

The figure shows a dual-stack network topology. Currently, no static routes are configured.



Static Routes: IPv4 Starting Routing Tables

- Each router has entries only for directly connected networks and associated local addresses.
- R1 can ping R2, but cannot ping the R3 LAN

```
R1# show ip route | begin Gateway
Gateway of last resort is not set
     172.16.0.0/16 is variably subnetted, 4 subnets, 2 masks
C       172.16.2.0/24 is directly connected, Serial0/1/0
L       172.16.2.1/32 is directly connected, Serial0/1/0
C       172.16.3.0/24 is directly connected, GigabitEthernet0/0/0
L       172.16.3.1/32 is directly connected, GigabitEthernet0/0/0
R1#
R1# ping 172.16.2.2
Type escape sequence to abort. Sending 5, 100-byte ICMP Echos to 172.16.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5)
R1# ping 192.168.2.1
Type escape sequence to abort. Sending 5, 100-byte ICMP Echos to 192.168.2.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
```

Static Routes: Next-Hop

In a next-hop static route, only the next-hop IP address is specified. The exit interface is derived from the next hop. For example, three next-hop IPv4 static routes are configured on R1 using the IP address of the next hop, R2.

```
R1(config)# ip route 172.16.1.0 255.255.255.0 172.16.2.2
R1(config)# ip route 192.168.1.0 255.255.255.0 172.16.2.2
R1(config)# ip route 192.168.2.0 255.255.255.0 172.16.2.2
```

The resulting routing table entries on R1:

```
R1# show ip route | begin Gateway
Gateway of last resort is not set
      172.16.0.0/16 is variably subnetted, 5 subnets, 2 masks
S        172.16.1.0/24 [1/0] via 172.16.2.2
C        172.16.2.0/24 is directly connected, Serial0/1/0
L        172.16.2.1/32 is directly connected, Serial0/1/0
C        172.16.3.0/24 is directly connected, GigabitEthernet0/0/0
L        172.16.3.1/32 is directly connected, GigabitEthernet0/0/0
S     192.168.1.0/24 [1/0] via 172.16.2.2
S     192.168.2.0/24 [1/0] via 172.16.2.2
```

Static Routes: Directly Connected

When configuring a static route, another option is to use the exit interface to specify the next-hop address. Three directly connected IPv4 static routes are configured on R1 using the exit interface.

Note: Using a next-hop address is generally recommended. Directly connected static routes should only be used with point-to-point serial interfaces.

```
R1(config)# ip route 172.16.1.0 255.255.255.0 s0/1/0
R1(config)# ip route 192.168.1.0 255.255.255.0 s0/1/0
R1(config)# ip route 192.168.2.0 255.255.255.0 s0/1/0
```

```
R1# show ip route | begin Gateway
Gateway of last resort is not set
      172.16.0.0/16 is variably subnetted, 5 subnets, 2 masks
S        172.16.1.0/24 is directly connected, Serial0/1/0
C        172.16.2.0/24 is directly connected, Serial0/1/0
L        172.16.2.1/32 is directly connected, Serial0/1/0
C        172.16.3.0/24 is directly connected, GigabitEthernet0/0/0
L        172.16.3.1/32 is directly connected, GigabitEthernet0/0/0
S     192.168.1.0/24 is directly connected, Serial0/1/0
S     192.168.2.0/24 is directly connected, Serial0/1/0
```

Static Routes: Fully Specified

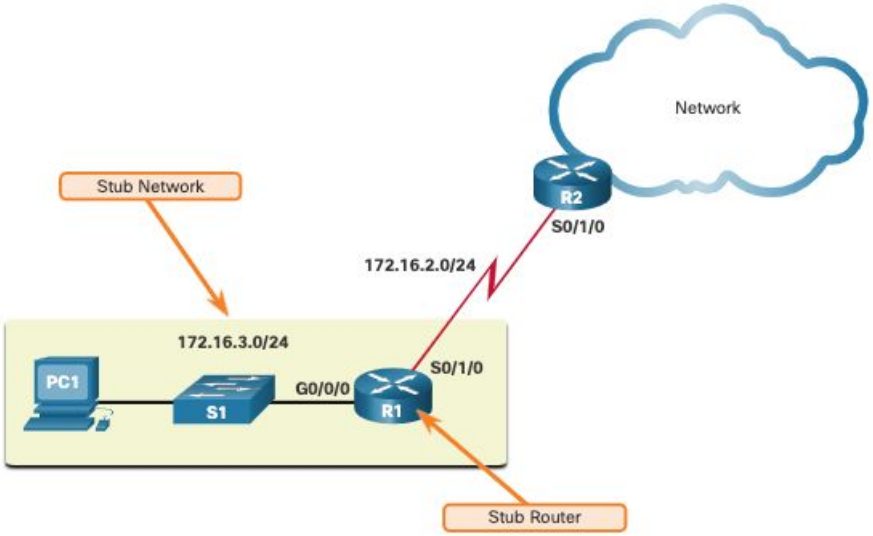
- In a fully specified static route, both the exit interface and the next-hop IP address are specified. This form of static route is used when the exit interface is a multi-access interface and it is necessary to explicitly identify the next hop. The next hop must be directly connected to the specified exit interface. Using an exit interface is optional, however it is necessary to use a next-hop address.
- It is recommended that when the exit interface is an Ethernet network, that the static route includes a next-hop address. You can also use a fully specified static route that includes both the exit interface and the next-hop address.

```
R1(config)# ip route 172.16.1.0 255.255.255.0 GigabitEthernet 0/0/1 172.16.2.2
R1(config)# ip route 192.168.1.0 255.255.255.0 GigabitEthernet 0/0/1 172.16.2.2
R1(config)# ip route 192.168.2.0 255.255.255.0 GigabitEthernet 0/0/1 172.16.2.2
```

```
R1# show ip route | begin Gateway
Gateway of last resort is not set
      172.16.0.0/16 is variably subnetted, 5 subnets, 2 masks
S        172.16.1.0/24 [1/0] via 172.16.2.2, GigabitEthernet0/0/1
C        172.16.2.0/24 is directly connected, GigabitEthernet0/0/1
L        172.16.2.1/32 is directly connected, GigabitEthernet0/0/1
C        172.16.3.0/24 is directly connected, GigabitEthernet0/0/0
L        172.16.3.1/32 is directly connected, GigabitEthernet0/0/0
S     192.168.1.0/24 [1/0] via 172.16.2.2, GigabitEthernet0/0/1
S     192.168.2.0/24 [1/0] via 172.16.2.2, GigabitEthernet0/0/1
```

Default Static Route

- A default route is a static route that matches all packets.
  - A single default route represents any network that is not in the routing table.
- Routers commonly use default routes that are either configured locally or learned from another router. The default route is used as the Gateway of Last Resort.
- Default static routes are commonly used when connecting an edge router to a service provider network, or a stub router (a router with only one upstream neighbor router).
- The figure shows a typical default static route scenario.



Default Static Route

IPv4 Default Static Route: The command syntax for an IPv4 default static route is similar to any other IPv4 static route, except that the network address is **0.0.0.0** and the subnet mask is **0.0.0.0**. The 0.0.0.0 0.0.0.0 in the route will match any network address.

Note: An IPv4 default static route is commonly referred to as a quad-zero route.

The basic command syntax for an IPv4 default static route is as follows:

```
Router(config)# ip route 0.0.0.0 0.0.0.0 {ip-address | exit-intf}
```

The example shows an IPv4 default static route configured on R1. With the configuration shown in the example, any packets not matching more specific route entries are forwarded to R2 at 172.16.2.2.

```
R1(config)# ip route 0.0.0.0 0.0.0.0 172.16.2.2
```

Default Static Route

The **show ip route static** command output from R1 displays the contents of the static routes in the routing table. Note the asterisk (*) next to the route with code 'S'. The asterisk indicates that this static route is a candidate default route, which is why it is selected as the Gateway of Last Resort.

Notice that the static default route configuration uses the /0 mask for IPv4 default routes. Remember that the IPv4 subnet mask in a routing table determines how many bits must match between the destination IP address of the packet and the route in the routing table. A /0 mask indicates that none of the bits are required to match. As long as a more specific match does not exist, the default static route matches all packets.

```
R1# show ip route static
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2
i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, * - candidate default, U - per-user static route
o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
+ - replicated route, % - next hop override

Gateway of last resort is 172.16.2.2 to network 0.0.0.0

S* 0.0.0.0/0 [1/0] via 172.16.2.2
```

Static Host Routes

A host route can be a manually configured static route to direct traffic to a specific destination device, such as the server shown in the figure. The static route uses a destination IP address and a 255.255.255.255 (/32) mask for IPv4 host routes.

The example shows the IPv4 static host route configuration on the Branch router to access the server.

```
Branch(config)# ip route 209.165.200.238 255.255.255.255 198.51.100.2
Branch(config)# exit
Branch#
```

```
Branch# show ip route | begin Gateway
Gateway of last resort is not set
      198.51.100.0/24 is variably subnetted, 2 subnets, 2 masks
C        198.51.100.0/30 is directly connected, Serial0/1/0
L        198.51.100.1/32 is directly connected, Serial0/1/0
      209.165.200.0/32 is subnetted, 1 subnets
S        209.165.200.238 [1/0] via 198.51.100.2
```

ACCESS CONTROL LIST

Access Control List

- An Access Control List (ACL) is a series of IOS commands that are used to filter packets based on information found in the packet header.
  - By default, a router does not have any ACLs configured.
  - When an ACL is applied to an interface, the router performs the additional task of evaluating all network packets as they pass through the interface to determine if the packet can be forwarded.
- An ACL uses a sequential list of permit or deny statements, known as access control entries (ACEs).
  - **Note:** ACEs are also 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 process is called **packet filtering**.

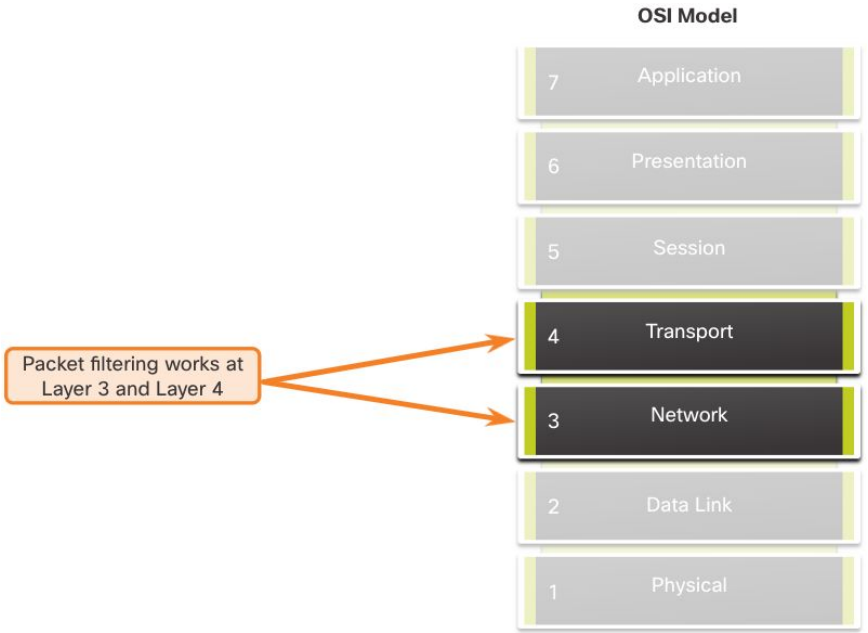
Access Control List

Several tasks performed by routers require the use of ACLs to identify traffic:

- Limit network traffic in order to increase network performance
- Provide traffic flow control
- Provide a basic level of security for network access
- Filter traffic based on traffic type
- Screen hosts to permit or deny access to network services
- Provide priority to certain classes of network traffic

Access Control List: Packet Filtering

- Packet filtering controls access to a network by analyzing the incoming and/or outgoing packets and forwarding them or discarding them based on given criteria.
- Packet filtering can occur at Layer 3 (Network) or Layer 4 (Transport).
- Cisco routers support two types of ACLs:
  - **Standard ACLs** - ACLs only filter at Layer 3 using the source IPv4 address only.
  - **Extended ACLs** - ACLs filter at:
    - Layer 3 using the source and / or destination IPv4 address.
    - Layer 4 using TCP, UDP ports, and optional protocol type information for finer control.



Access Control List: Operation

- 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.
  - **Note**: ACLs do not act on packets that originate from the router itself.
- An inbound ACL filters packets before they are routed to the outbound interface.
  - An inbound ACL is efficient because it saves the overhead of routing lookups if the packet is discarded.
- An outbound ACL filters packets after being routed, regardless of the inbound interface.

Access Control List: Operation

- When an ACL is applied to an interface, it follows a specific operating procedure. Here are the operational steps used when traffic has entered a router interface with an inbound standard IPv4 ACL configured:
  1. The router extracts the source IPv4 address from the packet header.
  2. The router starts at the top of the ACL and compares the source IPv4 address to each ACE in a sequential order.
  3. When a match is made, the router carries out the instruction, either permitting or denying the packet, and the remaining ACEs in the ACL, if any, are not analyzed.
  4. If the source IPv4 address does not match any ACEs in the ACL, the packet is discarded because there is an implicit deny ACE automatically applied to all ACLs.
- The last ACE statement of an ACL is always an implicit deny that blocks all traffic. It is hidden and not displayed in the configuration.
  - **Note**: An ACL must have at least one permit statement otherwise all traffic will be denied due to the implicit deny ACE statement.

Access Control List: Wildcard Mask

- A wildcard mask is similar to a subnet mask in that it uses the ANDing process to identify which bits in an IPv4 address to match. Unlike a subnet mask, in which binary 1 is equal to a match and binary 0 is not a match, in a wildcard mask, the reverse is true.
  - An IPv4 ACE uses a 32-bit wildcard mask to determine which bits of the address to examine for a match.

Wildcard Mask	Last Octet (Binary)	Meaning (0 - match, 1 - ignore)
0.0.0.0	**00000000**	• Match all octets.
0.0.0.63	**00111111**	• Match the first three octets • Match the two left most bits of the last octet • Ignore the last 6 bits
0.0.0.248	**11111100**	• Match the first three octets • Ignore the six left most bits of the last octet • Match the last two bits
0.0.0.255	**11111111**	• Match the first three octet • Ignore the last octet

Access Control List: Wildcard Mask Overview

Wildcard to Match a Host:

- Assume ACL 10 needs an ACE that only permits the host with IPv4 address 192.168.1.1. Recall that "0" equals a match and "1" equals ignore. To match a specific host IPv4 address, a wildcard mask consisting of all zeroes (i.e., 0.0.0.0) is required.
- When the ACE is processed, the wildcard mask will permit only the 192.168.1.1 address. The resulting ACE in ACL 10 would be **access-list 10 permit 192.168.1.1 0.0.0.0.**

	Decimal	Binary
IPv4 address	192.168.1.1	11000000.10101000.00000001.00000001
Wildcard Mask	0.0.0.0	00000000.00000000.00000000.00000000
Permitted IPv4 Address	**192.168.1.1**	11000000.10101000.00000001.00000001

Access Control List: Wildcard Mask

Wildcard Mask to Match an IPv4 Subnet

- ACL 10 needs an ACE that permits all hosts in the 192.168.1.0/24 network. The wildcard mask 0.0.0.255 stipulates that the very first three octets must match exactly but the fourth octet does not.
- When processed, the wildcard mask 0.0.0.255 permits all hosts in the 192.168.1.0/24 network. The resulting ACE in ACL 10 would be **access-list 10 permit 192.168.1.0 0.0.0.255.**

	Decimal	Binary
IPv4 address	192.168.1.1	11000000.10101000.00000001.00000001
Wildcard Mask	0.0.0.255	00000000.00000000.00000000.11111111
Permitted IPv4 Address	**192.168.1.0/24**	11000000.10101000.00000001.00000000

Access Control List: Wildcard Mask

Wildcard Mask to Match an IPv4 Address Range

- ACL 10 needs an ACE that permits all hosts in the 192.168.16.0/24, 192.168.17.0/24, …, 192.168.31.0/24 networks.
- When processed, the wildcard mask 0.0.15.255 permits all hosts in the 192.168.16.0/24 to 192.168.31.0/24 networks. The resulting ACE in ACL 10 would be **access-list 10 permit 192.168.16.0 0.0.15.255.**

	Decimal	Binary
IPv4 address	192.168.16.0	11000000.10101000.00010000.00000000
Wildcard Mask	0.0.15.255	00000000.00000000.00001111.11111111
Permitted IPv4 Address	**192.168.16.0/24 to 192.168.31.0/24**	11000000.10101000.00010000.00000000 11000000.10101000.00011111.00000000

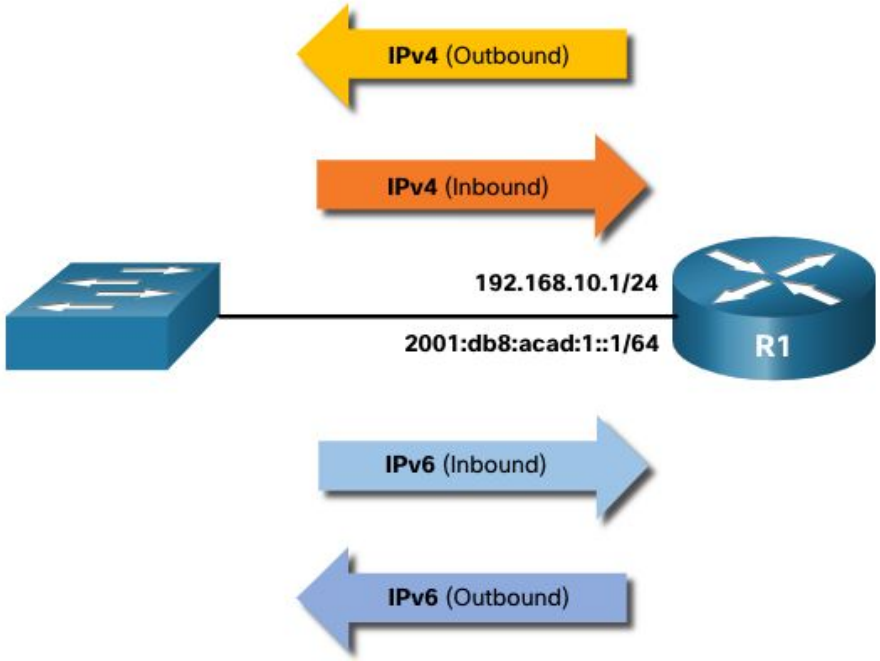
Access Control List: Wildcard Mask

- A shortcut method to calculate wildcard mask is to subtract the subnet mask from 255.255.255.255. Some examples:
  - Assume you wanted an ACE in ACL 10 to permit access to all users in the 192.168.3.0/24 network.
  - To calculate the wildcard mask, subtract the subnet mask (255.255.255.0) from 255.255.255.255.
  - This produces the wildcard mask 0.0.0.255.
  - The ACE would be **access-list 10 permit 192.168.3.0 0.0.0.255.**
- The Cisco IOS provides two keywords to identify the most common uses of wildcard masking. The two keywords are:
  - **host** - This keyword substitutes for the 0.0.0.0 mask. This mask states that all IPv4 address bits must match to filter just one host address.
  - **any** - This keyword substitutes for the 255.255.255.255 mask. This mask says to ignore the entire IPv4 address or to accept any addresses.

Access Control List: Number of ACLs per Interface

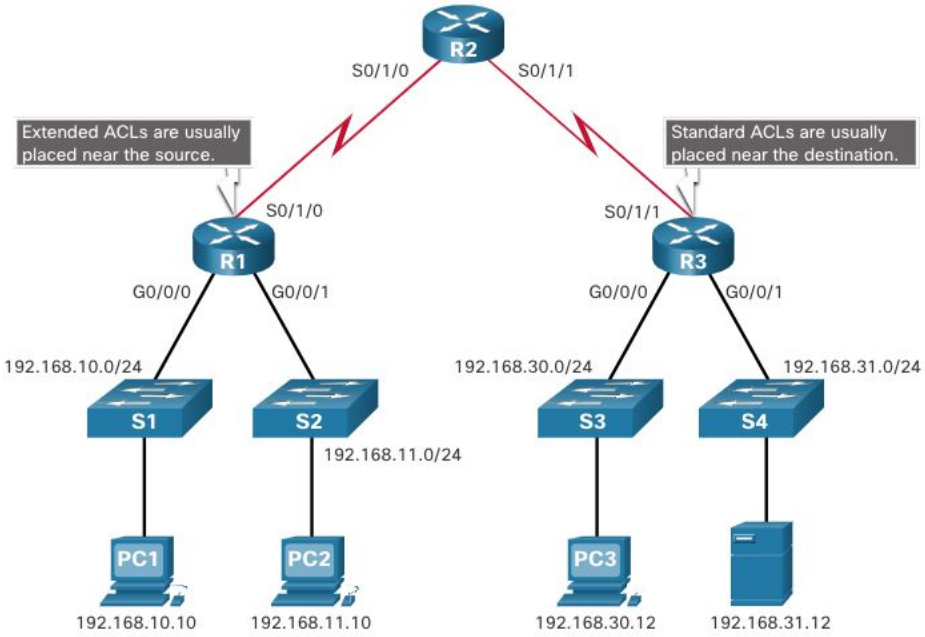
- There is a limit on the number of ACLs that can be applied on a router interface. For example, a dual-stacked (i.e, IPv4 and IPv6) router interface can have up to four ACLs applied, as shown in the figure.
- Specifically, a router interface can have:
  - One outbound IPv4 ACL.
  - One inbound IPv4 ACL.
  - One inbound IPv6 ACL.
  - One outbound IPv6 ACL.

- **Note**: ACLs do not have to be configured in both directions. The number of ACLs and their direction applied to the interface will depend on the security policy of the organization.



Access Control List: Types

- There are two types of IPv4 ACLs:
  - **Standard ACLs** - These permit or deny packets based only on the source IPv4 address.
  - **Extended ACLs** - These permit or deny packets based on the source IPv4 address and destination IPv4 address, protocol type, source and destination TCP or UDP ports and more.
- Every ACL should be placed where it has the greatest impact on efficiency.
- Extended ACLs should be located as close as possible to the source of the traffic to be filtered.
- Standard ACLs should be located as close to the destination as possible.



Access Control List: Types

Numbered ACLs

- ACLs numbered 1-99, or 1300-1999 are standard ACLs, while ACLs numbered 100-199, or 2000-2699 are extended ACLs.

```
R1(config)# access-list ?
 <1-99> IP standard access list
 <100-199> IP extended access list
 <1100-1199> Extended 48-bit MAC address access list
 <1300-1999> IP standard access list (expanded range)
 <200-299> Protocol type-code access list
 <2000-2699> IP extended access list (expanded range)
 <700-799> 48-bit MAC address access list
 rate-limit Simple rate-limit specific access list
 template Enable IP template acls
Router(config)# access-list
```

Access Control List: Types

Named ACLs

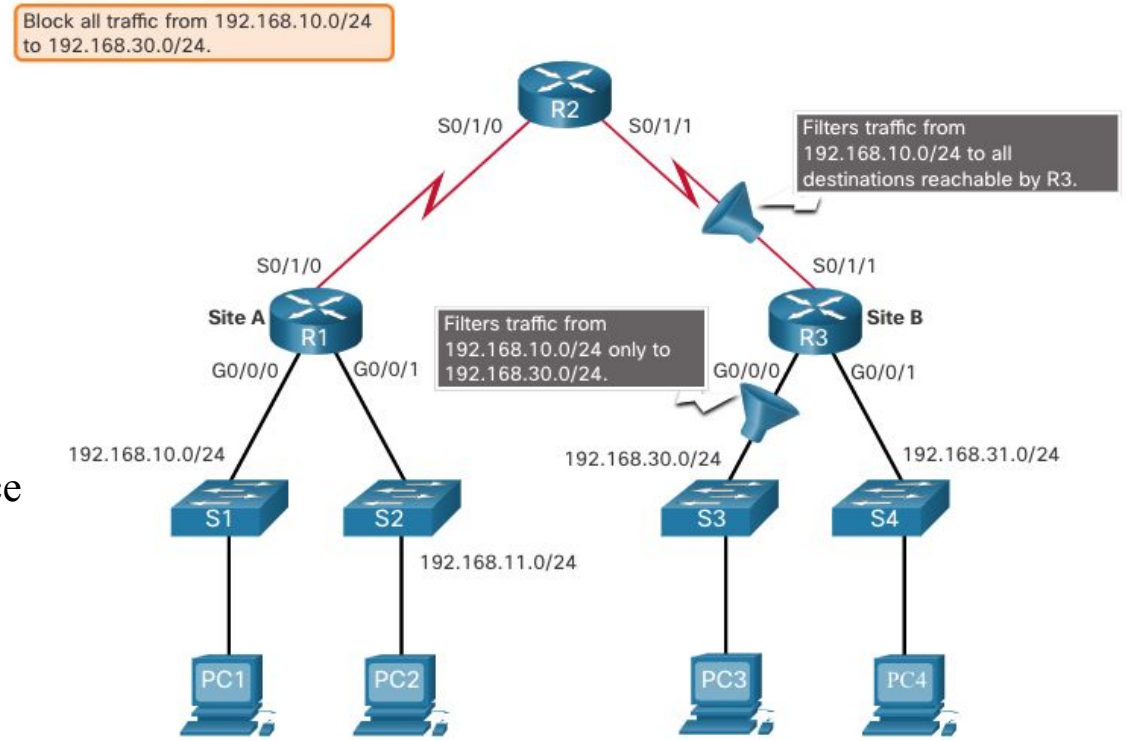
- Named ACLs are the preferred method to use when configuring ACLs. Specifically, standard and extended ACLs can be named to provide information about the purpose of the ACL. For example, naming an extended ACL FTP-FILTER is far better than having a numbered ACL 100.
- The **ip access-list** global configuration command is used to create a named ACL, as shown in the following example.

```
R1(config)# ip access-list extended FTP-FILTER
R1(config-ext-nacl)# permit tcp 192.168.10.0 0.0.0.255 any eq ftp
R1(config-ext-nacl)# permit tcp 192.168.10.0 0.0.0.255 any eq ftp-data
R1(config-ext-nacl)#
```

Access Control List: Types

In the figure, the administrator wants to prevent traffic originating in the 192.168.10.0/24 network from reaching the 192.168.30.0/24 network.

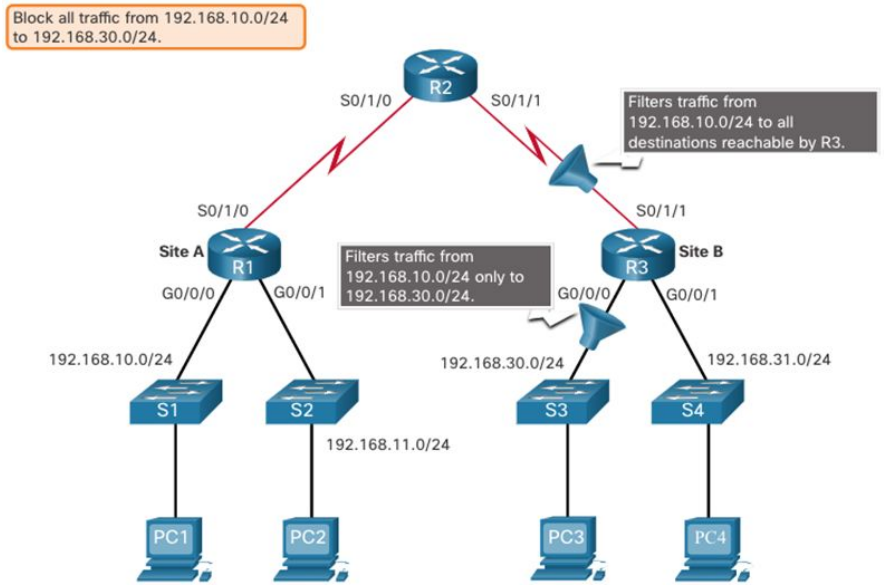
Following the basic placement guidelines, the administrator would place a standard ACL on router R3.



Access Control List: Types

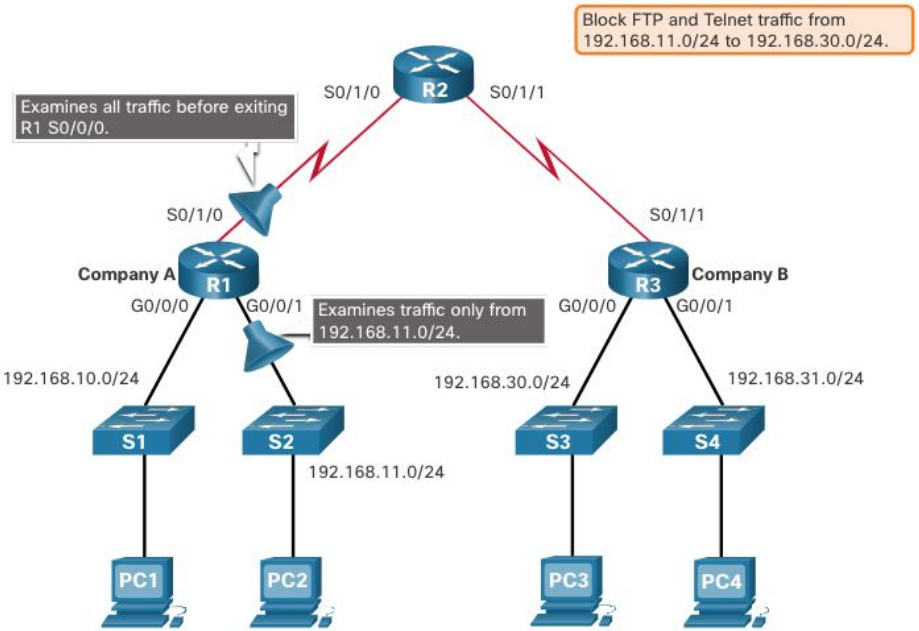
There are two possible interfaces on R3 to apply the standard ACL:

- **R3 S0/1/1 interface (inbound)** - The standard ACL can be applied inbound on the R3 S0/1/1 interface to deny traffic from .10 network. However, it would also filter .10 traffic to the 192.168.31.0/24 (.31 in this example) network. Therefore, the standard ACL should not be applied to this interface.
- **R3 G0/0 interface (outbound)** - The standard ACL can be applied outbound on the R3 G0/0/0 interface. This will not affect other networks that are reachable by R3. Packets from .10 network will still be able to reach the .31 network. This is the best interface to place the standard ACL to meet the traffic requirements.



Access Control List: Types

- Extended ACLs should be located as close to the source as possible.
- However, the organization can only place ACLs on devices that they control. Therefore, the extended ACL placement must be determined in the context of where organizational control extends.
- In the figure, for example, Company A wants to deny Telnet and FTP traffic to Company B's 192.168.30.0/24 network from their 192.168.11.0/24 network, while permitting all other traffic.



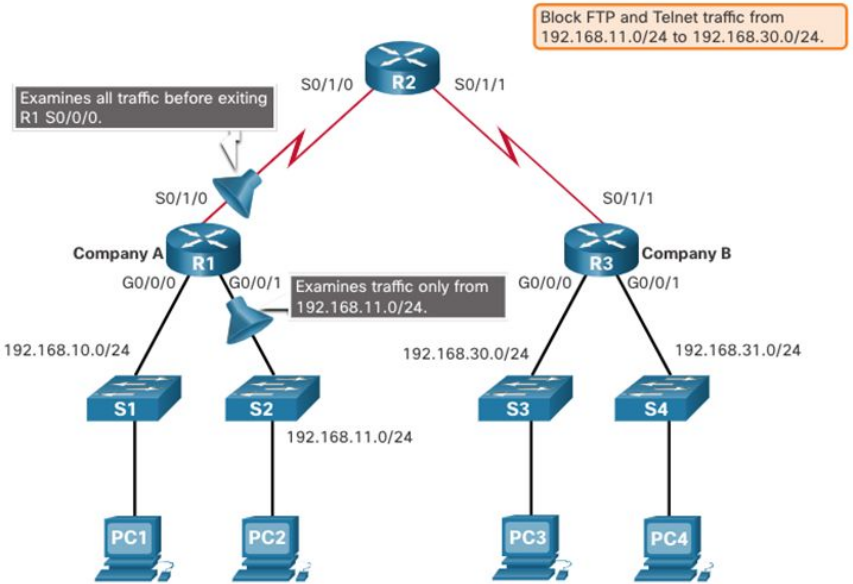
Access Control List: Types

An extended ACL on R3 would accomplish the task, but the administrator does not control R3. In addition, this solution allows unwanted traffic to cross the entire network, only to be blocked at the destination.

The solution is to place an extended ACL on R1 that specifies both source and destination addresses.

There are two possible interfaces on R1 to apply the extended ACL:

- **R1 S0/1/0 interface (outbound)** - The extended ACL can be applied outbound on the S0/1/0 interface. This solution will process all packets leaving R1 including packets from 192.168.10.0/24.
- **R1 G0/0/1 interface (inbound)** - The extended ACL can be applied inbound on the G0/0/1 and only packets from the 192.168.11.0/24 network are subject to ACL processing on R1. Because the filter is to be limited to only those packets leaving the 192.168.11.0/24 network, applying the extended ACL to G0/1 is the best solution.



Configure Standard ACL

To create a numbered standard ACL, use the **access-list** command.

```
Router(config)# access-list access-list-number {deny | permit | remark text} source [source-wildcard]
[log]
```

Parameter	Description
access-list-number	Number range is 1 to 99 or 1300 to 1999
deny	Denies access if the condition is matched
permit	Permits access if the condition is matched
remark *text*	(Optional) text entry for documentation purposes
source	Identifies the source network or host address to filter
source-wildcard	(Optional) 32-bit wildcard mask that is applied to the source
log	(Optional) Generates and sends an informational message when the ACE is matched

Note: Use the **no access-list** *access-list-number* global configuration command to remove a numbered standard ACL.

Configure Standard ACL

To create a named standard ACL, use the **ip access-list standard** command.

- ACL names are alphanumeric, case sensitive, and must be unique.
- Capitalizing ACL names is not required but makes them stand out when viewing the running-config output.

```
Router(config)# ip access-list standard access-list-name
```

```
R1(config)# ip access-list standard NO-ACCESS
R1(config-std-nacl)# ?
Standard Access List configuration commands:
  <1-2147483647>  Sequence Number
  default         Set a command to its defaults
  deny            Specify packets to reject
  exit            Exit from access-list configuration mode
  no              Negate a command or set its defaults
  permit          Specify packets to forward
  remark          Access list entry comment
R1(config-std-nacl)#
```

Configure Standard ACL

After a standard IPv4 ACL is configured, it must be linked to an interface or feature.

- The **ip access-group** command is used to bind a numbered or named standard IPv4 ACL to an interface.
- To remove an ACL from an interface, first enter the **no ip access-group** interface configuration command.

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

Configure Standard ACL

The example ACL permits traffic from host 192.168.10.10 and all hosts on the 192.168.20.0/24 network out interface serial 0/1/0 on router R1.

```
R1(config)# access-list 10 remark ACE permits ONLY host 192.168.10.10 to the internet
R1(config)# access-list 10 permit host 192.168.10.10
R1(config)# do show access-lists
Standard IP access list 10
    10 permit 192.168.10.10
R1(config)#
```

```
R1(config)# access-list 10 remark ACE permits all host in LAN 2
R1(config)# access-list 10 permit 192.168.20.0 0.0.0.255
R1(config)# do show access-lists
Standard IP access list 10
    10 permit 192.168.10.10
    20 permit 192.168.20.0, wildcard bits 0.0.0.255
R1(config)#
```

```
R1(config)# interface Serial 0/1/0
R1(config-if)# ip access-group 10 out
R1(config-if)# end
R1#
```

Configure Standard ACL

The example ACL permits traffic from host 192.168.10.10 and all hosts on the 192.168.20.0/24 network out interface serial 0/1/0 on router R1.

```
R1(config)# no access-list 10
R1(config)# ip access-list standard PERMIT-ACCESS
R1(config-std-nacl)# remark ACE permits host 192.168.10.10
R1(config-std-nacl)# permit host 192.168.10.10
R1(config-std-nacl)#
```

```
R1(config-std-nacl)# remark ACE permits host 192.168.10.10
R1(config-std-nacl)# permit host 192.168.10.10
R1(config-std-nacl)# remark ACE permits all hosts in LAN 2
R1(config-std-nacl)# permit 192.168.20.0 0.0.0.255
R1(config-std-nacl)# exit
R1(config)#
```

```
R1(config)# interface Serial 0/1/0
R1(config-if)# ip access-group PERMIT-ACCESS out
R1(config-if)# end
R1#
```

Modify ACL

- After an ACL is configured, it may need to be modified. ACLs with multiple ACEs can be complex to configure. Sometimes the configured ACE does not yield the expected behaviors.
- There are two methods to use when modifying an ACL:
  - Use a text editor.
  - Use sequence numbers.

- To correct an error in an ACL:
  - Copy the ACL from the running configuration and paste it into the text editor.
  - Make the necessary edits or changes.
  - Remove the previously configured ACL on the router.
  - Copy and paste the edited ACL back to the router.

```
R1# show run | section access-list
access-list 1 deny   19.168.10.10
access-list 1 permit 192.168.10.0 0.0.0.255
R1#
```

```
R1(config)# no access-list 1
R1(config)#
R1(config)# access-list 1 deny 192.168.10.10
R1(config)# access-list 1 permit 192.168.10.0 0.0.0.255
R1(config)#
```

Modify ACL

An ACL ACE can be deleted or added using the ACL sequence numbers.

- Use the **ip access-list standard** command to edit an ACL.
- Statements cannot be overwritten using an existing sequence number. The current statement must be deleted first with the **no 10** command. Then the correct ACE can be added using sequence number.

```
R1# show access-lists
Standard IP access list 1
    10 deny   19.168.10.10
    20 permit 192.168.10.0, wildcard bits 0.0.0.255
R1#
```

```
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# show access-lists
Standard IP access list 1
    10 deny   192.168.10.10
    20 permit 192.168.10.0, wildcard bits 0.0.0.255
R1#
```

Extended ACLs

- Extended ACLs provide a greater degree of control. They can filter on source address, destination address, protocol (i.e., IP, TCP, UDP, ICMP), and port number.
- Extended ACLs can be created as:
  - **Numbered Extended ACL** - Created using the **access-list** *access-list-number* global configuration command.
  - **Named Extended ACL** - Created using the **ip access-list extended** *access-list-name*.

```
R1(config)# access-list 100 permit ?
  <0-255>        An IP protocol number
  ahp            Authentication Header Protocol
  dvmrp          dvmrp
  eigrp          Cisco's EIGRP routing protocol
  esp            Encapsulation Security Payload
  gre            Cisco's GRE tunneling
  icmp           Internet Control Message Protocol
  igmp           Internet Gateway Message Protocol
  ip             Any Internet Protocol
  ipinip         IP in IP tunneling
  nos            KA9Q NOS compatible IP over IP tunneling
  object-group   Service object group
  ospf           OSPF routing protocol
  pcp            Payload Compression Protocol
  pim            Protocol Independent Multicast
  tcp            Transmission Control Protocol
  udp            User Datagram Protocol
R1(config)# access-list 100 permit
```

Extended ACLs

- Extended ACLs can filter on different port number and port name options.
- This example configures an extended ACL 100 to filter HTTP traffic. The first ACE uses the **www** port name. The second ACE uses the port number **80**. Both ACEs achieve exactly the same result.

```
R1(config)# access-list 100 permit tcp any any eq www
!or...
R1(config)# access-list 100 permit tcp any any eq 80
```

- Configuring the port number is required when there is not a specific protocol name listed such as SSH (port number 22) or an HTTPS (port number 443), as shown in the next example.

```
R1(config)# access-list 100 permit tcp any any eq 22
R1(config)# access-list 100 permit tcp any any eq 443
R1(config)#
```

Extended ACLs

- In this example, the ACL permits both HTTP and HTTPS traffic from the 192.168.10.0 network to go to any destination.

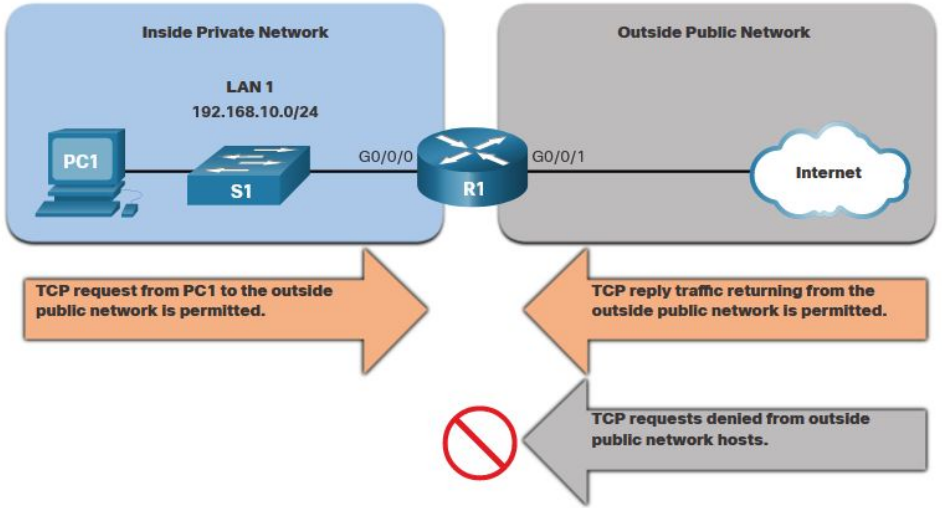
- Extended ACLs can be applied in various locations. However, they are commonly applied close to the source. Here ACL 110 is applied inbound on the R1 G0/0/0 interface.

```
R1(config)# access-list 110 permit tcp 192.168.10.0 0.0.0.255 any eq www
R1(config)# access-list 110 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1(config)# interface g0/0/0
R1(config-if)# ip access-group 110 in
R1(config-if)# exit
R1(config)#
```

Extended ACLs

TCP can also perform basic stateful firewall services using the TCP **established** keyword.

- The **established** keyword enables inside traffic to exit the inside private network and permits the returning reply traffic to enter the inside private network.
- TCP traffic generated by an outside host and attempting to communicate with an inside host is denied.



Extended ACLs

- ACL 120 is configured to only permit returning web traffic to the inside hosts. The ACL is then applied outbound on the R1 G0/0/0 interface.
- The **show access-lists** command shows that inside hosts are accessing the secure web resources from the internet.

Note: A match occurs if the returning TCP segment has the ACK or reset (RST) flag bits set, indicating that the packet belongs to an existing connection.

```
R1(config)# access-list 120 permit tcp any 192.168.10.0 0.0.0.255 established
R1(config)# interface g0/0/0
R1(config-if)# ip access-group 120 out
R1(config-if)# end
R1# show access-lists
Extended IP access list 110
    10 permit tcp 192.168.10.0 0.0.0.255 any eq www
    20 permit tcp 192.168.10.0 0.0.0.255 any eq 443 (657 matches)
Extended IP access list 120
    10 permit tcp any 192.168.10.0 0.0.0.255 established (1166 matches)
R1#
```

Extended ACLs

Naming an ACL makes it easier to understand its function. To create a named extended ACL, use the **ip access-list extended** configuration command.

In the example, a named extended ACL called NO-FTP-ACCESS is created and the prompt changed to named extended ACL configuration mode. ACE statements are entered in the named extended ACL sub configuration mode.

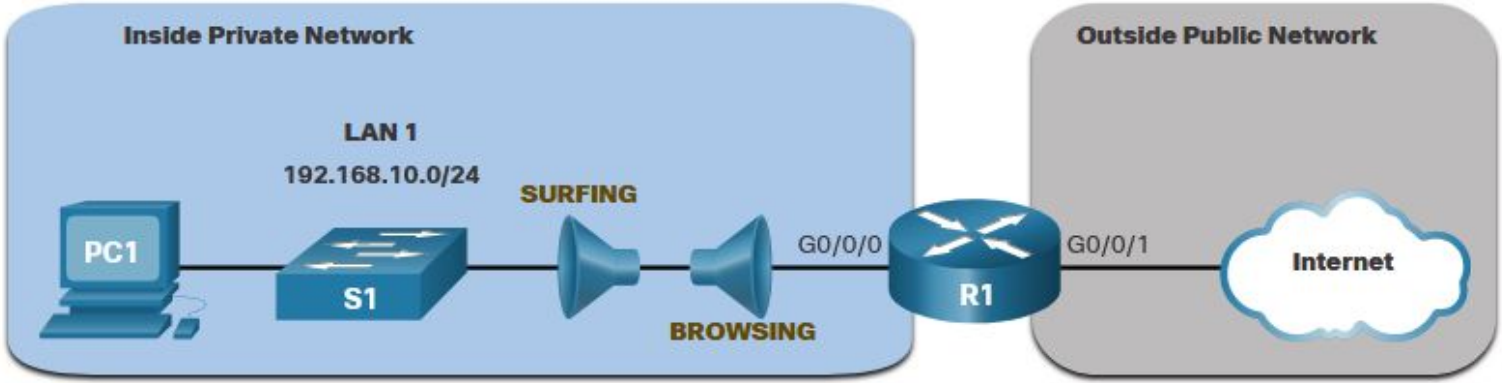
```
Router(config)# ip access-list extended access-list-name
```

```
R1(config)# ip access-list extended NO-FTP-ACCESS
R1(config-ext-nacl)#
```

Extended ACLs

The topology below is used to demonstrate configuring and applying two named extended IPv4 ACLs to an interface:

- **SURFING** - This will permit inside HTTP and HTTPS traffic to exit to the internet.
- **BROWSING** - This will only permit returning web traffic to the inside hosts while all other traffic exiting the R1 G0/0/0 interface is implicitly denied.



Extended ACLs

- The SURFING ACL permits HTTP and HTTPS traffic from inside users to exit the G0/0/1 interface connected to the internet. Web traffic returning from the internet is permitted back into the inside private network by the BROWSING ACL.
- The SURFING ACL is applied inbound and the BROWSING ACL is applied outbound on the R1 G0/0/0 interface.

```
R1(config)# ip access-list extended SURFING
R1(config-ext-nacl)# Remark Permits inside HTTP and HTTPS traffic
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)#
R1(config)# ip access-list extended BROWSING
R1(config-ext-nacl)# Remark Only permit returning HTTP and HTTPS traffic
R1(config-ext-nacl)# permit tcp any 192.168.10.0 0.0.0.255 established
R1(config-ext-nacl)# exit
R1(config)# interface g0/0/0
R1(config-if)# ip access-group SURFING in
R1(config-if)# ip access-group BROWSING out
R1(config-if)# end
R1# show access-lists
Extended IP access list SURFING
    10 permit tcp 192.168.10.0 0.0.0.255 any eq www
    20 permit tcp 192.168.10.0 0.0.0.255 any eq 443 (124 matches)
Extended IP access list BROWSING
    10 permit tcp any 192.168.10.0 0.0.0.255 established (369 matches)
R1#
```

Extended ACLs

The show access-lists command is used to verify the ACL statistics. Notice that the permit secure HTTPS counters (i.e., eq 443) in the SURFING ACL and the return established counters in the BROWSING ACL have increased.

```
R1# show access-lists
Extended IP access list BROWSING
    10 permit tcp any 192.168.10.0 0.0.0.255 established
Extended IP access list SURFING
    10 permit tcp 19.168.10.0 0.0.0.255 any eq www
    20 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1#
```

Extended ACLs

An extended ACL can be edited using a text editor when many changes are required. Or, if the edit applies to one or two ACEs, then sequence numbers can be used.

Example:

- The ACE sequence number 10 in the SURFING ACL has an incorrect source IP networks address.

```
R1# show access-lists
Extended IP access list BROWSING
    10 permit tcp any 192.168.10.0 0.0.0.255 established
Extended IP access list SURFING
    10 permit tcp 19.168.10.0 0.0.0.255 any eq www
    20 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1#
```

Extended ACLs

- To correct this error the original statement is removed with the **no** *sequence_#* command and the corrected statement is added replacing the original statement.
- The **show access-lists** command output verifies the configuration change.

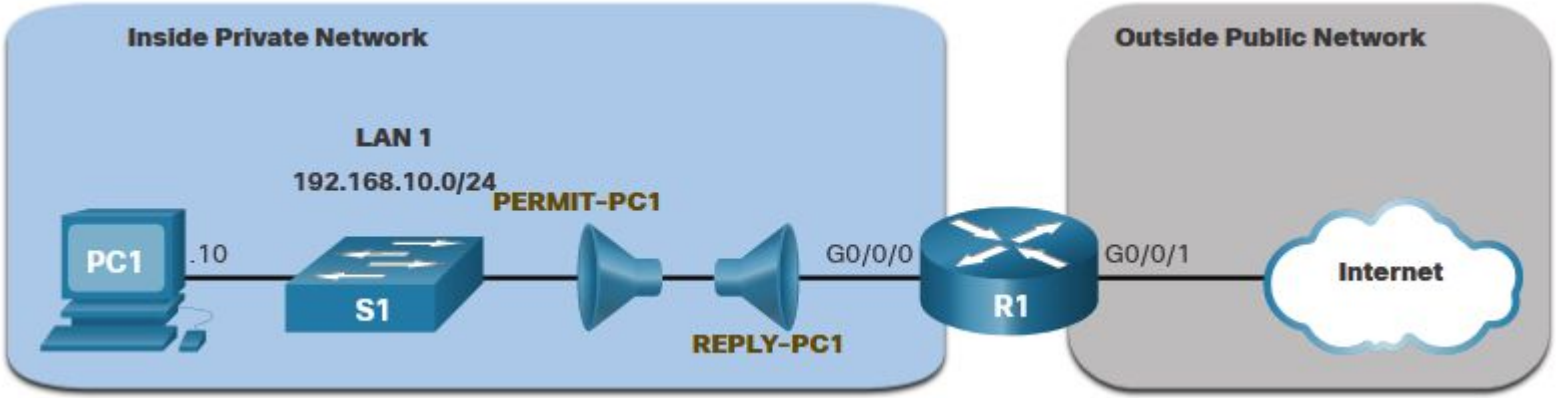
```
R1# configure terminal
R1(config)# ip access-list extended SURFING
R1(config-ext-nacl)# no 10
R1(config-ext-nacl)# 10 permit tcp 192.168.10.0 0.0.0.255 any eq www
R1(config-ext-nacl)# end
```

```
R1# show access-lists
Extended IP access list BROWSING
    10 permit tcp any 192.168.10.0 0.0.0.255 established
Extended IP access list SURFING
    10 permit tcp 192.168.10.0 0.0.0.255 any eq www
    20 permit tcp 192.168.10.0 0.0.0.255 any eq 443
R1#
```

Extended ACLs

Two named extended ACLs will be created:

- **PERMIT-PC1** - This will only permit PC1 TCP access to the internet and deny all other hosts in the private network.
- **REPLY-PC1** - This will only permit specified returning TCP traffic to PC1 implicitly deny all other traffic.



Extended ACLs

- The **PERMIT-PC1** ACL permits PC1 (192.168.10.10) TCP access to the FTP, SSH, Telnet, DNS , HTTP, and HTTPS traffic.
- The **REPLY-PC1** ACL will permit return traffic to PC1.
- The **PERMIT-PC1** ACL is applied inbound and the **REPLY-PC1** ACL applied outbound on the R1 G0/0/0 interface.

```
R1(config)# ip access-list extended PERMIT-PC1
R1(config-ext-nacl)# Remark Permit PC1 TCP access to internet
R1(config-ext-nacl)# permit tcp host 192.168.10.10 any eq 20
R1(config-ext-nacl)# permit tcp host 192.168.10.10 any eq 21
R1(config-ext-nacl)# permit tcp host 192.168.10.10 any eq 22
R1(config-ext-nacl)# permit tcp host 192.168.10.10 any eq 23
R1(config-ext-nacl)# permit tcp host 192.168.10.10 any eq 53
R1(config-ext-nacl)# permit tcp host 192.168.10.10 any eq 80
R1(config-ext-nacl)# permit tcp host 192.168.10.10 any eq 443
R1(config-ext-nacl)# deny ip 192.168.10.0 0.0.0.255 any
R1(config-ext-nacl)# exit
R1(config)#
R1(config)# ip access-list extended REPLY-PC1
R1(config-ext-nacl)# Remark Only permit returning traffic to PC1
R1(config-ext-nacl)# permit tcp any host 192.168.10.10 established
R1(config-ext-nacl)# exit
R1(config)# interface g0/0/0
R1(config-if)# ip access-group PERMIT-PC1 in
R1(config-if)# ip access-group REPLY-PC1 out
R1(config-if)# end
R1#
```

Extended ACLs

The **show ip interface** command is used to verify the ACL on the interface and the direction in which it was applied.

```
R1# show ip interface g0/0/0
GigabitEthernet0/0/0 is up, line protocol is up (connected)
  Internet address is 192.168.10.1/24
  Broadcast address is 255.255.255.255
  Address determined by setup command
  MTU is 1500 bytes
  Helper address is not set
  Directed broadcast forwarding is disabled
  Outgoing access list is REPLY-PC1
  Inbound  access list is PERMIT-PC1
  Proxy ARP is enabled
  Security level is default
  Split horizon is enabled
  ICMP redirects are always sent
  ICMP unreachables are always sent
  ICMP mask replies are never sent
  IP fast switching is disabled
  IP fast switching on the same interface is disabled
  IP Flow switching is disabled
  IP Fast switching turbo vector
  IP multicast fast switching is disabled
  IP multicast distributed fast switching is disabled
  Router Discovery is disabled
R1#
R1# show ip interface g0/0/0 | include access list
Outgoing access list is REPLY-PC1
Inbound access list is PERMIT-PC1
R1#
```

Extended ACLs

The **show access-lists** command can be used to confirm that the ACLs work as expected. The command displays statistic counters that increase whenever an ACE is matched.

Note: Traffic must be generated to verify the operation of the ACL.

```
R1# show access-lists
Extended IP access list PERMIT-PC1
10 permit tcp host 192.168.10.10 any eq 20
20 permit tcp host 192.168.10.10 any eq ftp
30 permit tcp host 192.168.10.10 any eq 22
40 permit tcp host 192.168.10.10 any eq telnet
50 permit tcp host 192.168.10.10 any eq domain
60 permit tcp host 192.168.10.10 any eq www
70 permit tcp host 192.168.10.10 any eq 443
80 deny ip 192.168.10.0 0.0.0.255 any
Extended IP access list REPLY-PC1
10 permit tcp any host 192.168.10.10 established
R1#
```

Extended ACLs

The **show running-config** command can be used to validate what was configured. The command also displays configured remarks.

```
R1# show running-config | begin ip access-list
ip access-list extended PERMIT-PC1
remark Permit PC1 TCP access to internet
permit tcp host 192.168.10.10 any eq 20
permit tcp host 192.168.10.10 any eq ftp
permit tcp host 192.168.10.10 any eq 22
permit tcp host 192.168.10.10 any eq telnet
permit tcp host 192.168.10.10 any eq domain
permit tcp host 192.168.10.10 any eq www
permit tcp host 192.168.10.10 any eq 443
deny ip 192.168.10.0 0.0.0.255 any
ip access-list extended REPLY-PC1
remark Only permit returning traffic to PC1
permit tcp any host 192.168.10.10 established
!
```