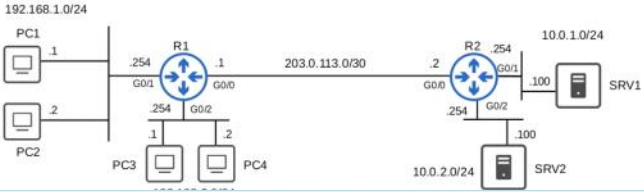
CONCEPTS PART 7

Thursday, February 16, 2023 6:43 PM

What are ACLs?

- ACLs (Access Control Lists) have multiple uses.
- In Day 34 and Day 35, we will focus on ACLs from a security perspective.
- ACLs function as a packet filter, instructing the router to permit or discard specific traffic.
- ACLs can filter traffic based on source/destination IP addresses, source/destination Layer 4 ports, etc.



- REQUIREMENT:
  Hosts in 192.168.1.0/24 can access the 10.0.1.0/24 network
  Hosts in 192.168.2.0/24 cannot access the 10.0.1.0/24 network.
- ACLs are configured globally on the router. (global config mode)
- They are an ordered sequence of ACEs. (Access Control Entries)

ACL 1:
1: if source IP = 192.168.1.0/24, then permit
2: if source IP = 192.168.2.0/24, then deny
3: if source IP = any, then permit

- Configuring an ACL in global config mode will not make the ACL take effect.
- The ACL must be applied to an interface.
- ACLs are applied either inbound or outbound.

ACL 1:
1: if source IP = 192.168.1.0/24, then permit
2: if source IP = 192.168.2.0/24, then deny
3: if source IP = any, then permit

192.168.1.0/24

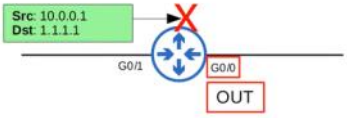
How ACLs work

- Configuring an ACL in global config mode will not make the ACL take effect.
- The ACL must be applied to an interface.
- ACLs are applied either inbound or outbound.
- ACLs are made up of one or more ACEs.
- When the router checks a packet against the ACL, it processes the ACEs in order, from top to bottom.
- If the packet matches one of the ACEs in the ACL, the router takes the action and stops processing the ACL. All entries below the matching entry will be ignored.

ACL 2:
1: if source IP = 192.168.1.0/24, then permit
2: if source IP = 192.168.0.0/16, then deny

Implicit deny

- What will happen if a packet doesn't match any of the entries in an ACL?



ACL 2:
1: if source IP = 192.168.1.0/24, then permit
2: if source IP = 192.168.0.0/16, then deny

- There is an 'implicit deny' at the end of all ACLs.
- The implicit deny tells the router to deny all traffic that doesn't match any of the configured entries in the ACL.

ACL Types

- Standard ACLs: Match based on **Source IP address** only
  - → Standard Numbered ACLs
  - → Standard Named ACLs

- Extended ACLs: Match based on **Source/Destination IP, Source/Destination port,** etc.
  - → Extended Numbered ACLs
  - → Extended Named ACLs

Standard Numbered ACLs

- Standard ACLs match traffic based only on the source IP address of the packet.
- Numbered ACLs are identified with a number (ie. ACL 1, ACL 2, etc)
- Different types of ACLs have a different range of numbers that can be used.
  - → Standard ACLs can use 1-99 and 1300-1999.

Protocol	Range
Standard IP	1-99 and 1300-1999
Extended IP	100-199 and 2000-2699
Ethernet type code	200-299
Ethernet address	700-799
Transparent bridging (protocol type)	200-299
Transparent bridging (vendor code)	700-799
Extended transparent bridging	1100-1199
DECnet and extended DECnet	300-399

Xerox Network Systems (XNS)	400-499
Extended XNS	500-599
AppleTalk	600-699
Source-route bridging (protocol type)	200-299
Source-route bridging (vendor code)	700-799
Internetwork Packet Exchange (IPX)	800-899
Extended IPX	900-999
IPX Service Advertising Protocol (SAP)	1000-1099

- The basic command to configure a standard numbered ACL is:

```
R1(config)# access-list number {deny | permit} ip wildcard-mask
```

```
R1(config)# access-list 1 deny 1.1.1.1 0.0.0.0
R1(config)# access-list 1 deny 1.1.1.1
R1(config)# access-list 1 deny host 1.1.1.1
R1(config)# access-list 1 permit any
R1(config)# access-list 1 permit 0.0.0.0 255.255.255.255
R1(config)# access-list 1 remark ## BLOCK BOB FROM ACCOUNTING ##
```

Remark is like description for the ACL

R1(config)# do show access-lists

R1(config)# do show ip access-lists

R1(config)# do show running-config | include/section access-list

Apply to an interface:

```
R1(config-if)# ip access-group number {in | out}
```

- Number is the acl number

Standard Named ACLs

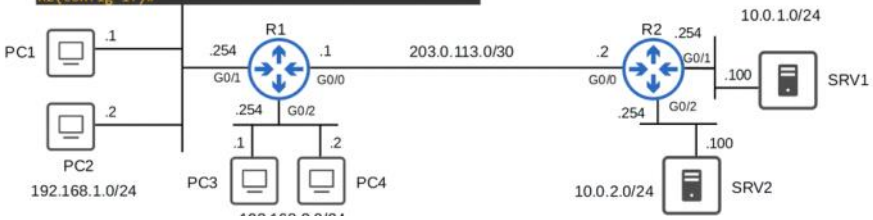
- Standard ACLs match traffic based only on the source IP address of the packet.
- Named ACLs are identified with a name (ie. 'BLOCK_BOB')
- Standard named ACLs are configured by entering 'standard named ACL config mode', and then configuring each entry within that config mode.
  `R1(config)# ip access-list standard acl-name`
  `R1(config-std-nacl)# [entry-number] {deny | permit} ip wildcard-mask`

```
R1(config)#ip access-list standard BLOCK_BOB
R1(config-std-nacl)#5 deny 1.1.1.1
R1(config-std-nacl)#10 permit any
R1(config-std-nacl)#remark ## CONFIGURED NOV 21 2020 ##
R1(config-std-nacl)#interface g0/0
R1(config-if)#ip access-group BLOCK_BOB in
```

Standard Named ACLs

```
R2(config)#ip access-list standard TO_10.0.2.0/24
R2(config-std-nacl)#deny 192.168.1.0 0.0.0.255
R2(config-std-nacl)#permit any
R2(config-std-nacl)#interface g0/2
R2(config-if)#ip access-group TO_10.0.2.0/24 out
R2(config-if)#
R2(config-if)#ip access-list standard TO_10.0.1.0/24
R2(config-std-nacl)#deny 192.168.2.1
R2(config-std-nacl)#permit 192.168.2.0 0.0.0.255
R2(config-std-nacl)#permit 192.168.1.1
R2(config-std-nacl)#deny 192.168.1.0 0.0.0.255
R2(config-std-nacl)#permit any
R2(config-std-nacl)#interface g0/1
R2(config-if)#ip access-group TO_10.0.1.0/24 out
R2(config-if)#
```

Requirements:
- PCs in 192.168.1.0/24 can't access 10.0.2.0/24.
- PC3 can't access 10.0.1.0/24.
- Other PCs in 192.168.2.0/24 can access 10.0.1.0/24.
- PC1 can access 10.0.1.0/24.
- Other PCs in 192.168.1.0/24 can't access 10.0.1.0/24.



- The router may re-order the /32 entries.
- This improves the efficiency of processing the ACL.
- It **does not** change the effect of the ACL.
- This applies to both standard named and standard numbered ACLs.
- Packet Tracer does not do this.

Can also use numbered acl with named acl command as well

R1(config)# ip access-list standard ?(number,word)

```
R1(config-std-nacl)#do show access-lists
Standard IP access list 1
    10 deny   192.168.1.1
    20 deny   192.168.1.2
    30 deny   192.168.3.0, wildcard bits 0.0.0.255
    40 permit any
R1(config-std-nacl)#
R1(config-std-nacl)#no 30
R1(config-std-nacl)#
R1(config-std-nacl)#do show access-lists
Standard IP access list 1
    10 deny   192.168.1.1
    20 deny   192.168.1.2
    40 permit any
R1(config-std-nacl)#
R1(config)#no access-list 1 deny 192.168.3.0 0.0.0.255
R1(config)#do show access-lists
R1(config)#do show running-config | section access-list
R1(config)#
```

When configuring/editing numbered ACLs from global config mode, you can't delete individual entries, you can only delete the entire ACL!

- You can insert new entries in between other entries by specifying the sequence number.

```
R1(config-std-nacl)#do show access-lists
Standard IP access list 1
    10 deny   192.168.1.1
    20 deny   192.168.1.2
    40 permit any
R1(config-std-nacl)#
R1(config-std-nacl)#30 deny 192.168.2.0 0.0.0.255
R1(config-std-nacl)#
R1(config-std-nacl)#do show access-lists
Standard IP access list 1
    10 deny   192.168.1.1
    20 deny   192.168.1.2
    30 deny   192.168.2.0, wildcard bits 0.0.0.255
    40 permit any
R1(config-std-nacl)#
R1(config-std-nacl)#do show running-config | section access-list
access-list 1 deny   192.168.1.1
access-list 1 deny   192.168.1.2
access-list 1 deny   192.168.2.0 0.0.0.255
access-list 1 permit any
```

Resequencing ACLs

- There is a *resequencing* function that helps edit ACLs.
- The command is **ip access-list resequence** *acl-id starting-seq-num increment*

```
R1(config)#do show access-lists
Standard IP access list 1
    1 deny   192.168.1.1
    3 deny   192.168.3.1
    2 deny   192.168.2.1
    4 deny   192.168.4.1
    5 permit any
R1(config)#
R1(config)#ip access-list resequence 1 10 10
R1(config)#
R1(config)#do show access-lists
Standard IP access list 1
    10 deny   192.168.1.1
    20 deny   192.168.3.1
    30 deny   192.168.2.1
    40 deny   192.168.4.1
    50 permit any
```

Extended ACLs

- Extended ACLs function mostly the same as standard ACLs.
- They can be numbered or named, just like standard ACLs.
  → Numbered ACLs use the following ranges: 100 – 199, 2000 – 2699
- They are processed from top to bottom, just like standard ACLs.
- However, the can match traffic based on more parameters, so they are more precise (and more complex) than standard ACLs.
- We will focus on matching based on these main parameters: **Layer 4 protocol/port**, **source address**, and **destination address**.

```
R1(config)# access-list number [permit | deny] protocol src-ip dest-ip

R1(config)# ip access-list extended {name | number}
R1(config-ext-nacl)# [seq-num] [permit | deny] protocol src-ip dest-ip
```

```
R1(config)#ip access-list extended EXAMPLE
R1(config-ext-nacl)#deny ?
  <0-255>        An IP protocol number
  ahp            Authentication Header Protocol
  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
  sctp           Stream Control Transmission Protocol
  tcp            Transmission Control Protocol
  udp            User Datagram Protocol
```

1: ICMP
6: TCP
17: UDP
88: EIGRP
89: OSPF

```
R1(config-ext-nacl)#deny tcp ?
  A.B.C.D        Source address
  any            Any source host
  host           A single source host
  object-group   Source network object group

R1(config-ext-nacl)#deny tcp any ?
  A.B.C.D        Destination address
  any            Any destination host
  eq             Match only packets on a given port number
  gt             Match only packets with a greater port number
  host           A single destination host
  lt             Match only packets with a lower port number
  neq            Match only packets not on a given port number
  object-group   Destination network object group
  range          Match only packets in the range of port numbers

R1(config-ext-nacl)#deny tcp any 10.0.0.0 ?
  A.B.C.D  Destination wildcard bits

R1(config-ext-nacl)#deny tcp any 10.0.0.0 0.0.0.255
R1(config-ext-nacl)#
```

1. Allow all traffic

```
R1(config-ext-nacl)#permit ip any any
```

2. Prevent 10.0.0.0/16 from sending UDP traffic to 192.168.1.1/32

```
R1(config-ext-nacl)#deny udp 10.0.0.0 0.0.255.255 host 192.168.1.1
```

3. Prevent 172.16.1.1/32 from pinging hosts in 192.168.0.0/24

```
R1(config-ext-nacl)#deny icmp host 172.16.1.1 192.168.0.0 0.0.0.255
```

Matching the TCP/UDP port numbers

- When matching TCP/UDP, you can optionally specify the source and/or destination port numbers to match.

```
R1(config-ext-nacl)#deny tcp src-ip  eq    src-port-num   dest-ip  eq    dst-port-num
                                     gt                             gt
                                     lt                             lt
                                     neq                            neq
                                     range                          range
```

- **eq 80** = equal to port 80
- **gt 80** = greater than 80 (81 and greater)
- **lt 80** = less than 80 (79 and less)
- **neq 80** = NOT 80
- **range 80 100** = from port 80 to port 100

After the destination IP address and/or destination port numbers, there are many more options you can use to match (not necessary for the CCNA).
Some examples:

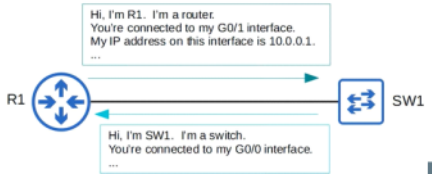
- **ack**: match the TCP ACK flag
- **fin**: match the TCP FIN flag
- **syn**: match the TCP SYN flag
- **ttl**: match packets with a specific TTL value
- **dscp**: match packets with a specific DSCP value

DSCP(differentiated service code point) is in the ipv4 header

```
R1(config)#ip access-list extended BLOCK_10.0.2.0/24
R1(config-ext-nacl)#deny ip 192.168.2.0 0.0.0.255 10.0.2.0 0.0.0.255
R1(config-ext-nacl)#permit ip any any
R1(config-ext-nacl)#interface g0/2
R1(config-if)#ip access-group BLOCK_10.0.2.0/24 in
```

Layer 2 Discovery Protocols

- Layer 2 discovery protocols such as CDP and LLDP share information with and discover information about neighboring (connected) devices.
- The shared information includes host name, IP address, device type, etc.
- CDP is a Cisco proprietary protocol.
- LLDP is an industry standard protocol (IEEE 802.1AB).
- Because they share information about the devices in the network, they can be considered a security risk and are often not used. It is up to the network engineer/admin to decide if they want to use them in the network or not.



Cisco Discovery Protocol

- CDP is a Cisco proprietary protocol.
- It is enabled on Cisco devices (routers, switches, firewalls, IP phones, etc) by default.
- CDP messages are periodically sent to multicast MAC address `0100.0CCC.CCCC`.
- When a device receives a CDP message, it processes and discards the message. It does NOT forward it to other devices.
- By default, CDP messages are sent once every **60 seconds**.
- By default, the CDP holdtime is **180 seconds**. If a message isn't received from a neighbor for 180 seconds, the neighbor is removed from the CDP neighbor table.
- CDPv2 messages are sent by default.

R1# show cdp (if enabled shows cdp info)
R1# show cdp traffic (shows packets received /sent from/to interfaces)
R1#show cdp interface(see setting of cdpin each interface)
R1# show cdp neighbors(device ID with theie interface that is neighbors to this)
R1# show cdp neighbors detail
R1# show cdp entry R2(neightbor hostname)

- CDP is globally enabled by default.
- CDP is also enabled on each interface by default.
- To enable/disable CDP globally: `R1(config)# [no] cdp run`
- To enable/disable CDP on specific interfaces: `R1(config-if)# [no] cdp enable`
- Configure the CDP timer: `R1(config)# cdp timer seconds`
- Configure the CDP holdtime: `R1(config)# cdp holdtime seconds`
- Enable/disable CDPv2: `R1(config)# [no] cdp advertise-v2`

Link Layer Discovery Protocol

- LLDP is an industry standard protocol (IEEE 802.1AB).
- It is usually disabled on Cisco devices by default, so it must be manually enabled.
- A device can run CDP and LLDP at the same time.
- LLDP messages are periodically sent to multicast MAC address `0180.C200.000E`.
- When a device receives an LLDP message, it processes and discards the message. It does NOT forward it to other devices.
- By default, LLDP messages are sent once every **30 seconds**.
- By default, the LLDP holdtime is **120 seconds**.
- LLDP has an additional timer called the 'reinitialization delay'. If LLDP is enabled (globally or on an interface), this timer will delay the actual initialization of LLDP. **2 seconds** by default.
- LLDP is usually globally disabled by default.
- LLDP is also disabled on each interface by default.
- To enable LLDP globally: `R1(config)# lldp run`
- To enable LLDP on specific interfaces (tx): `R1(config-if)# lldp transmit`
- To enable LLDP on specific a interface (rx): `R1(config-if)# lldp receive`
- Configure the LLDP timer: `R1(config)# lldp timer seconds`
- Configure the LLDP holdtime: `R1(config)# lldp holdtime seconds`
- Configure the LLDP reinit timer: `R1(config)# lldp reinit seconds`

R1# show lldp traffic
R1# show lldp interface
R1# show lldp
R1# show lldp neighbors (detail)
R1# show lldp entry SW1(hostname)

R1# show interfaces status

The importance of time

- All devices have an internal clock (routers, switches, your PC, etc)
- In Cisco IOS, you can view the time with the **show clock** command.

```
R1#show clock
*00:16:00.857 UTC Sat Dec 26 2020
```

Default is UTC

- If you use the **show clock detail** command, you can see the time source.

```
R1#show clock detail
*00:19:49.411 UTC Sat Dec 26 2020
Time source is hardware calendar
```

* = time is not considered authoritative

The hardware calendar is the default time source.

- The internal hardware clock of a device will drift over time, so it is not the ideal time source.

- From a CCNA perspective, the most important reason to have accurate time on a device is to have accurate logs for troubleshooting.

R2# show logging

Manual Time Configuration

- You can manually configure the time on the device with the **clock set** command.

```
R2#clock set ?
  hh:mm:ss  Current Time

R2#clock set 14:30:00 ?
  <1-31>  Day of the month
  MONTH   Month of the year

R2#clock set 14:30:00 27 ?
  MONTH  Month of the year

R2#clock set 14:30:00 27 Dec ?
  <1993-2035>  Year

R2#clock set 14:30:00 27 Dec 2020 ?
  <cr>

R2#clock set 14:30:00 27 Dec 2020
R2#show clock detail
14:30:05.887 UTC Sun Dec 27 2020
Time source is user configuration
```

Jeremy's IT Lab

<cr> -> carriage return (indicates nothing there here afterr)

- Although the hardware calendar (built-in clock) is the default time-source, the hardware clock and software clock are separate and can be configured separately

Hardware Clock (Calendar) Configuration

- You can manually configure the hardware clock with the **calendar set** command.

```
R2#calendar set 14:35:00 ?
  <1-31>  Day of the month
  MONTH   Month of the year

R2#calendar set 14:35:00 27 ?
  MONTH  Month of the year

R2#calendar set 14:35:00 27 Dec ?
  <1993-2035>  Year

R2#calendar set 14:35:00 27 Dec 2020 ?
  <cr>

R2#calendar set 14:35:00 27 Dec 2020
R2#show calendar
14:35:07 UTC Sun Dec 27 2020
```

- Typically you will want to synchronize the 'clock' and 'calendar'.
- Use the command **clock update-calendar** to sync the calendar to the clock's time.
- Use the command **clock read-calendar** to sync the clock to the calendar's time.

```
R2#show clock
14:38:14.301 UTC Sun Dec 27 2020
R2#show calendar
00:00:03 UTC Sun Dec 27 2020
R2#clock update-calendar
R2#show clock
14:38:22.181 UTC Sun Dec 27 2020
R2#show calendar
14:38:23 UTC Sun Dec 27 2020
```

```
R2#show clock
00:00:15.788 UTC Mon Sep 6 1993
R2#show calendar
14:55:07 UTC Sun Dec 27 2020
R2#clock read-calendar
R2#show clock
14:55:12.522 UTC Sun Dec 27 2020
R2#show calendar
14:55:15 UTC Sun Dec 27 2020
```

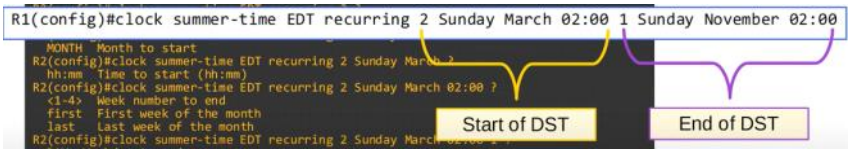
Configuring the Time Zone

- You can configure the time zone with the **clock timezone** command.

```
R2(config)#do show clock
15:13:33.985 UTC Sun Dec 27 2020
R2(config)#clock timezone ?
  WORD  name of time zone

R2(config)#clock timezone JST ?
  <-23 - 23>  Hours offset from UTC

R2(config)#clock timezone JST 9 ?
  <0-59>  Minutes offset from UTC
  <cr>

R2(config)#clock timezone JST 9
R2(config)#do show clock
00:13:45.414 JST Mon Dec 28 2020
R2(config)#do clock set 15:15:00 Dec 27 2020
R2(config)#do show clock
15:15:02.129 JST Sun Dec 27 2020
```

DAYLIGHT SAVING TIME(SUMMER TIME)

```
R2(config)#clock summer-time ?
  WORD  name of time zone in summer
R2(config)#clock summer-time EDT ?
  date       Configure absolute summer time
  recurring  Configure recurring summer time
R2(config)#clock summer-time EDT recurring ?
  <1-4>  Week number to start
  first  First week of the month
  last   Last week of the month
  <cr>
R2(config)#clock summer-time EDT recurring 2 ?
  DAY  Weekday to start
R2(config)#clock summer-time EDT recurring 2 Sunday ?
  MONTH  Month to start
R2(config)#clock summer-time EDT recurring 2 Sunday March ?
  hh:mm  Time to start (hh:mm)
R2(config)#clock summer-time EDT recurring 2 Sunday March 02:00 ?
  <1-4>  Week number to end
  first  First week of the month
  last   Last week of the month
R2(config)#clock summer-time EDT recurring 2 Sunday March 02:00 1 ?
  DAY  Weekday to end
R2(config)#clock summer-time EDT recurring 2 Sunday March 02:00 1 Sunday ?
  MONTH  Month to end
R2(config)#$r-time EDT recurring 2 Sunday March 02:00 1 Sunday November ?
  hh:mm  Time to end (hh:mm)
R2(config)#$ recurring 2 Sunday March 02:00 1 Sunday November 02:00 ?
  <1-1440>  Offset to add in minutes
  <cr>
R2(config)#$ recurring 2 Sunday March 02:00 1 Sunday November 02:00
```



Network Time Protocol

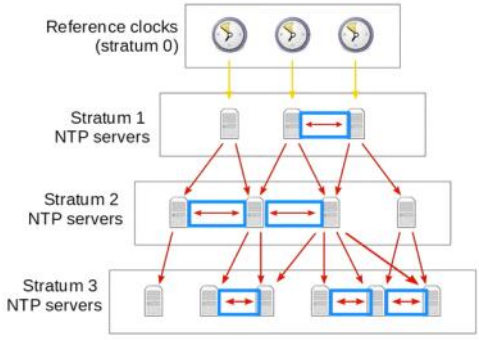
- Manually configuring the time on devices is not scalable.
- The manually configured clocks will drift, resulting in inaccurate time.
- NTP (Network Time Protocol) allows automatic syncing of time over a network.
- NTP clients request the time from NTP servers.
- A device can be an NTP server and an NTP client at the same time.
- NTP allows accuracy of time within ~1 millisecond if the NTP server is in the same LAN, or within ~50 milliseconds if connecting to the NTP server over a WAN/the Internet.
- Some NTP servers are 'better' than others. The 'distance' of an NTP server from the original **reference clock** is called **stratum**.
- NTP uses UDP port 123 to communicate.

Reference Clocks

- A reference clock is usually a very accurate time device like an atomic clock or a GPS clock.
- Reference clocks are **stratum 0** within the NTP hierarchy.
- NTP servers directly connected to reference clocks are **stratum 1**.

NTP Hierarchy

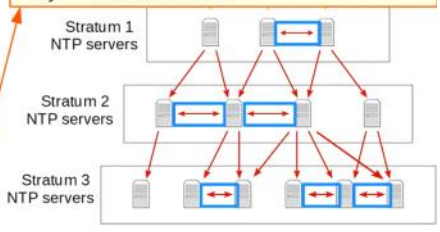
- Reference clocks are **stratum 0**.
- **Stratum 1** NTP servers get their time from reference clocks.
- **Stratum 2** NTP servers get their time from stratum 1 NTP servers.
- **Stratum 3** NTP servers get their time from stratum 2 NTP servers.
- **Stratum 15** is the maximum. Anything above that is considered unreliable.
- Devices can also 'peer' with devices at the same stratum to provide more accurate time.



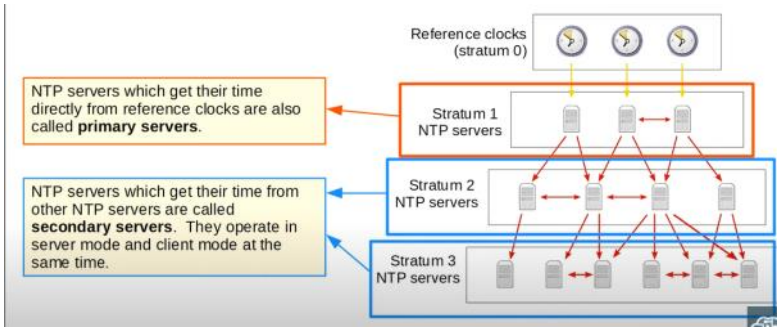
NTP Hierarchy

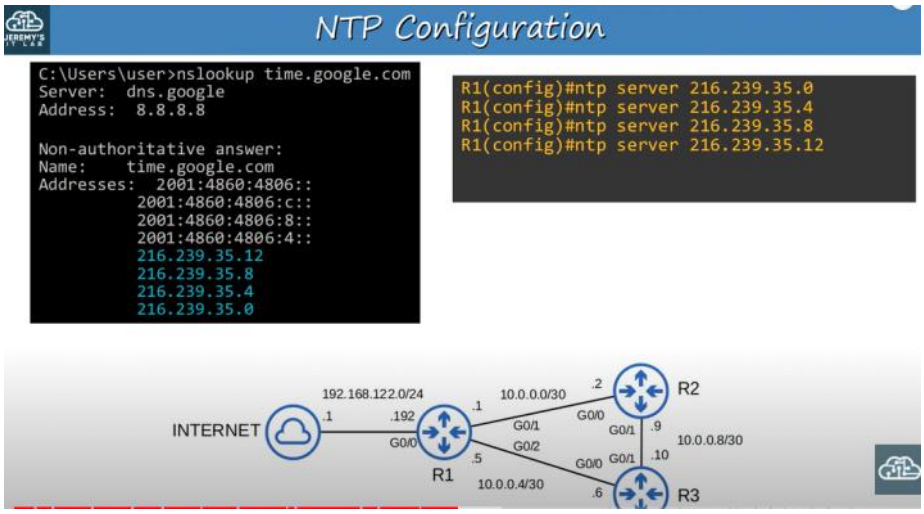
- Reference clocks are **stratum 0**.
- **Stratum 1** NTP servers get their time from reference clocks.
- **Stratum 2** NTP servers get their time from stratum 1 NTP servers.
- **Stratum 3** NTP servers get their time from stratum 2 NTP servers.
- **Stratum 15** is the maximum. Anything above that is considered unreliable.
- Devices can also 'peer' with devices at the same stratum to provide more accurate time.

This is called 'symmetric active' mode.
Cisco devices can operate in three NTP modes:
- Server mode
- Client mode
- Symmetric active mode



- An NTP client can sync to multiple NTP servers.





Ntp server ipv4 prefer (to prefer this ip over others)

R1# show ntp associations

R1# show ntp status (to see whether clock is synchronized , startum level etc..)

```
R1(config)#do show clock detail
06:56:32.315 UTC Mon Dec 28 2020
Time source is NTP
R1(config)#do show calendar
05:23:06 UTC Mon Dec 28 2020
R1(config)#clock timezone JST 9
R1(config)#ntp update-calendar
R1(config)#do show clock detail
15:57:33.078 JST Mon Dec 28 2020
Time source is NTP
R1(config)#do show calendar
15:57:36 JST Mon Dec 28 2020
```

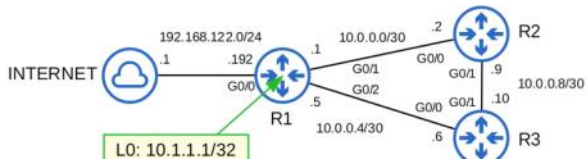
Configures the router to update the hardware clock (calendar) with the time learned via NTP.

The hardware clock tracks the date and time on the device even if it restarts, power is lost, etc. When the system is restarted, the hardware clock is used to initialize the software clock.

NTP Configuration

```
R1(config)#interface loopback0
R1(config-if)#ip address 10.1.1.1 255.255.255.255
R1(config-if)#exit
R1(config)#ntp source loopback0
```

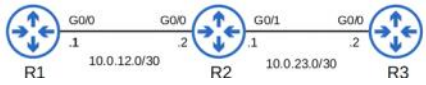
```
R2(config)#ntp server 10.1.1.1
R2(config)#do show ntp associations

  address         ref clock       st   when   poll reach  delay  offset   disp
*~10.1.1.1        216.239.35.12    2      0     64     1  7.038 -13.128 3937.5
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~ configured
R2(config)#do show ntp status
Clock is synchronized, stratum 3, reference is 10.1.1.1
...
```



Servers with lower stratum levels are preferred.

Configuring NTP server mode

```
R1(config)#ntp master ?
  <1-15>  Stratum number
  <cr>

R1(config)#ntp master
R1(config)#do show ntp associations

  address         ref clock       st   when   poll reach  delay  offset   disp
*~127.127.1.1     .LOCL.           7      2     16   377  0.000   0.000  0.292
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~ configured
R1(config)#do show ntp status
Clock is synchronized, stratum 8, reference is 127.127.1.1
...
```



Configuring NTP symmetric active mode

```
R2(config)#ntp peer 10.0.23.2

R2(config)#do show ntp associations

  address         ref clock       st   when   poll reach  delay  offset   disp
*~10.0.12.1       127.127.1.1      8     60     64    17 24.040 206.682  0.987
 ~10.0.23.2       10.0.12.1        9     33     64     0  0.000   0.000 15937.
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~ configured
```

```
R3(config)#ntp peer 10.0.23.1

R3(config)#do show ntp associations

  address         ref clock       st   when   poll reach  delay  offset   disp
*~10.0.12.1       127.127.1.1      8     11     64    37 12.605  -7.406 63.575
 ~10.0.23.1       10.0.12.1        9      1     64     0  0.000   0.000 15937.
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~ configured
```

Configuring NTP Authentication

- NTP authentication can be configured, although it is optional.
- It allows NTP clients to ensure they only sync to the intended servers.
- To configure NTP authentication:

```
ntp authenticate
ntp authentication-key key-number md5 key
ntp trusted-key key-number
ntp server ip-address key key-number
```

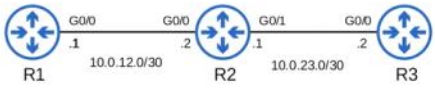
Enable NTP authentication

Create the NTP authentication key(s)

Specify the trusted key(s)

Specify which key to use for the server

This command isn't needed on the server (R1).




```
R1(config)#ntp authenticate
R1(config)#ntp authentication-key 1 md5 jeremysitlab
R1(config)#ntp trusted-key 1
```

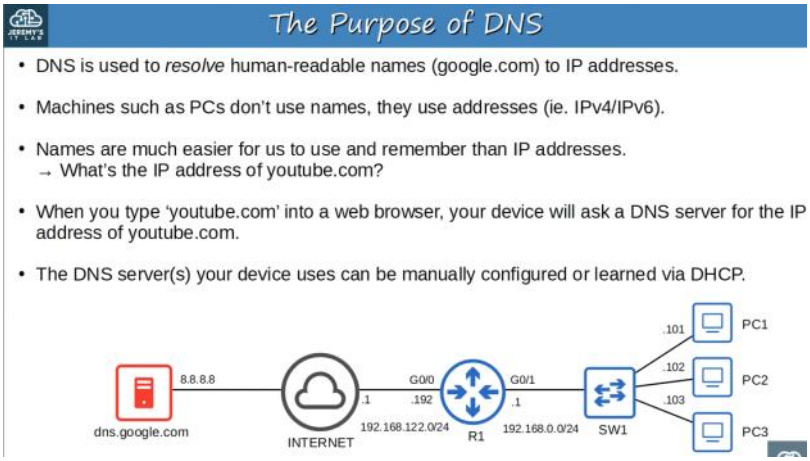
```
R2(config)#ntp authenticate
R2(config)#ntp authentication-key 1 md5 jeremysitlab
R2(config)#ntp trusted-key 1
R2(config)#ntp server 10.0.12.1 key 1
R2(config)#ntp peer 10.0.23.2 key 1
```

```
R3(config)#ntp authenticate
R3(config)#ntp authentication-key 1 md5 jeremysitlab
R3(config)#ntp trusted-key 1
R3(config)#ntp server 10.0.12.1 key 1
R2(config)#ntp peer 10.0.23.1 key 1
```

R1# clock read-calendar (software clock matches the hardwae clock)

CONCEPTS PART 8

Saturday, February 18, 2023 2:58 PM



ipconfig /all (to display various info)

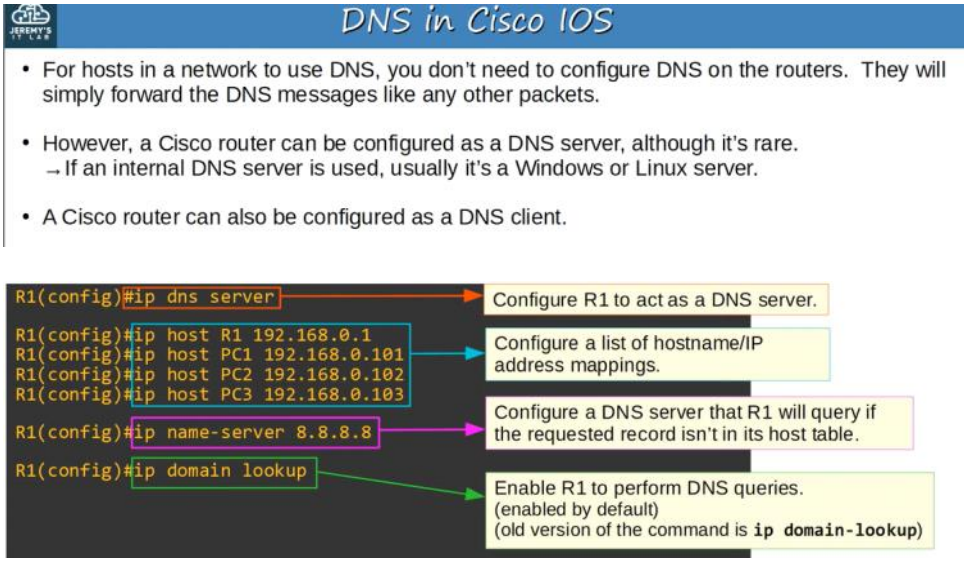
> Standard DNS queries/responses typically use **UDP**.
> **TCP** is used for DNS messages greater than 512 bytes.
> In either case, port 53 is used.

Ipconfig /displaydns (displays dns info in the form of cache)

> Devices will save the DNS server's responses to a local DNS cache.
> This means they don't have to query the server every single time they want to access a particular destination.

Ipconfig /flushdns (to clear the cache)

In widnows : Windows/System32/drivers/etc/hosts(has hosts and ip)
In linux: /etc/hosts



R1# show hosts (name server details and hosts learned or

configured with flasgs(perm(permannent),temp(learned))

```
R1(config)#ip domain name jeremysitlab.com
```

(optional)
- Configure the default *domain* name.

- This will be automatically appended to any hostnames without a specified *domain*.
- ie. **`ping pc1`** will become **`ping pc1.jeremysitlab.com`**
- (old version of the command: **`ip domain-name`**)

The Purpose of DHCP

- DHCP is used to allow hosts to automatically/dynamically learn various aspects of their network configuration, such as IP address, subnet mask, default gateway, DNS server, etc, without manual/static configuration.
- It is an essential part of modern networks.
  →When you connect a phone/laptop to WiFi, do you ask the network admin which IP address, subnet mask, default gateway, etc, the phone/laptop should use?
- Typically used for 'client devices' such as workstations (PCs), phones, etc.
- Devices such as routers, servers, etc, are usually manually configured.
- In small networks (such as home networks) the router typically acts as the DHCP server for hosts in the LAN.
- In larger networks, the DHCP server is usually a Windows/Linux server.

If preferred is displayed in the ipv4 address in ipconfig /all then this pc was previously assigned this ip address by the dhcp server, so it asked to receive the same addres again this time.

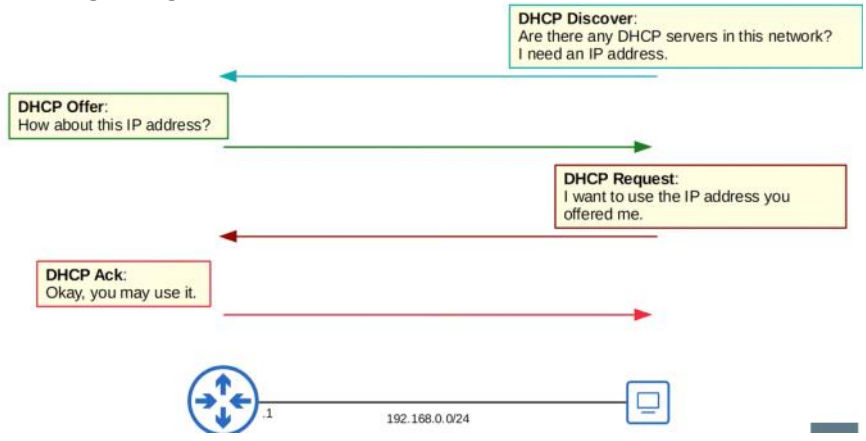
DHCP server 'lease' IP address to clients.
These leases are usually not permanent, and the client must give up the address at the end of the lease.

/user> ipconfig /release (releases the ip address)

DHCP **servers** use UDP **67**.
DHCP **clients** use UDP **68**.

/user> ipconfig /renew (contacts dhcp server to get ip)

4 messages to get:

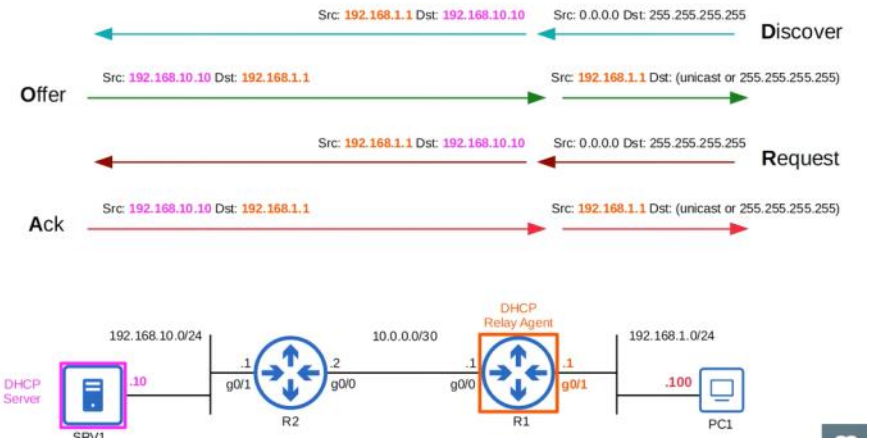


The dhcp offer message can be either broadcast or unicast

Discover	Client → Server	Broadcast
Offer	Server → Client	Broadcast or Unicast
Request	Client → Server	Broadcast
Ack	Server → Client	Broadcast or Unicast
Release	Client → Server	Unicast

DHCP Relay

- Some network engineers might choose to configure each router to act as the DHCP server for its connected LANs.
- However, large enterprises often choose to use a centralized DHCP server.
- If the server is centralized, it won't receive the DHCP clients' broadcast DHCP messages. (broadcast messages don't leave the local subnet)
- To fix this, you can configure a router to act as a **DHCP relay agent**.
- The router will forward the clients' broadcast DHCP messages to the remote DHCP server as unicast messages.



DHCP Server Configuration in IOS

```
R1(config)#ip dhcp excluded-address 192.168.1.1 192.168.1.10
R1(config)#ip dhcp pool LAB_POOL
R1(dhcp-config)#network 192.168.1.0 ?
  /nn or A.B.C.D  Network mask or prefix length
  <cr>
R1(dhcp-config)#network 192.168.1.0 /24
R1(dhcp-config)#dns-server 8.8.8.8
R1(dhcp-config)#domain-name jeremysitlab.com
R1(dhcp-config)#default-router 192.168.1.1
R1(dhcp-config)#lease 0 5 30
```

Specify a range of addresses that **won't** be given to DHCP clients.

Create a DHCP pool.

Specify the subnet of addresses to be assigned to clients (except the excluded addresses)

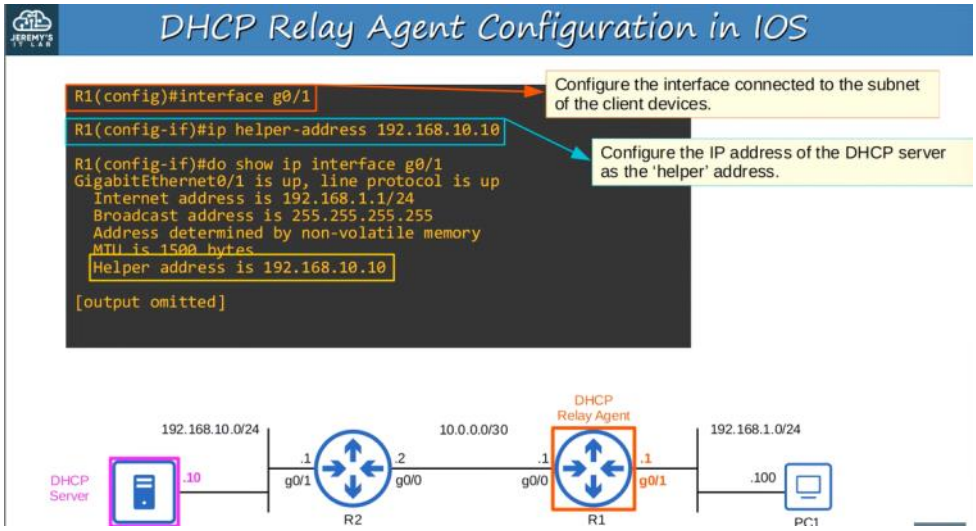
Specify the DNS server that DHCP clients should use.

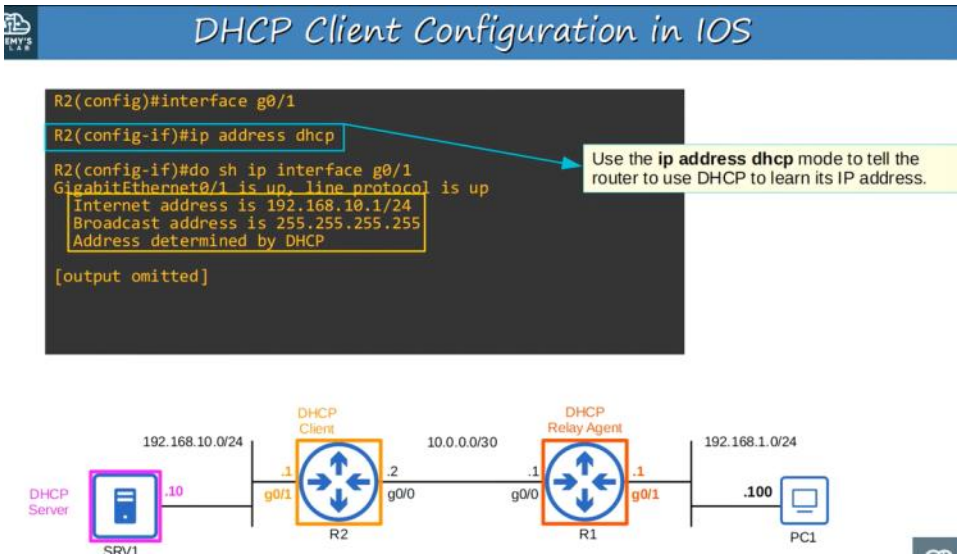
Specify the domain name of the network. (ie. PC1 = pc1.jeremysitlab.com)

Specify the default gateway.

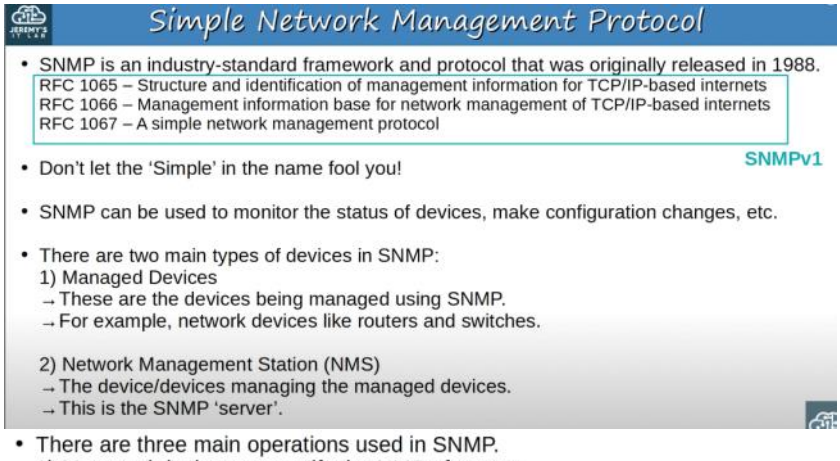
Specify the lease time.
lease *days hours minutes* OR
lease infinite

R1# show ip dhcp binding
(shows the client dhcp ,client mac address,and their lease time etc..)

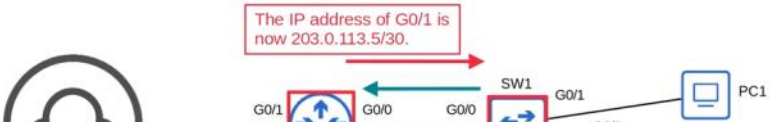




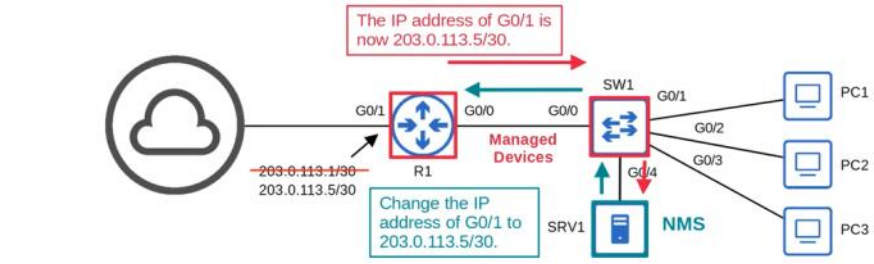
Simple network management protocol:

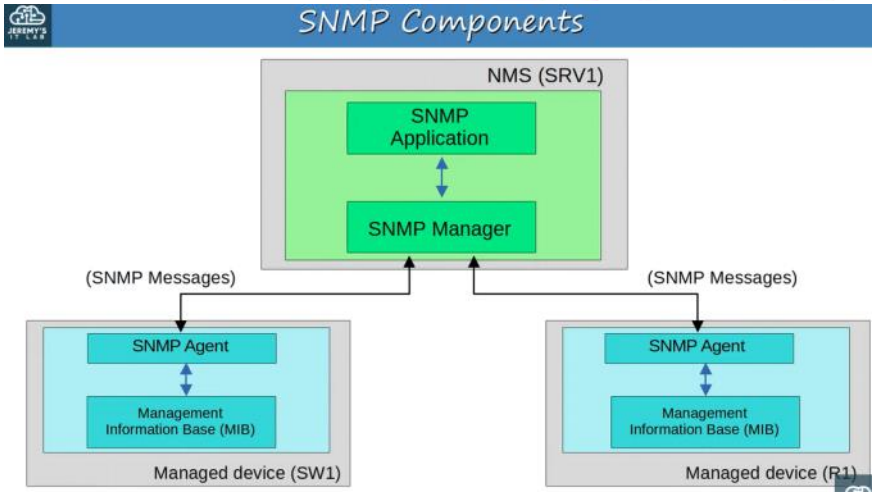


- There are three main operations used in SNMP.
  1) Managed devices can notify the NMS of events.
  2) The NMS can ask the managed devices for information about their current status.
  3) The NMS can tell the managed devices to change aspects of their configuration.



- There are three main operations used in SNMP.
  1) Managed devices can notify the NMS of events.
  2) The NMS can ask the managed devices for information about their current status.
  3) The NMS can tell the managed devices to change aspects of their configuration.





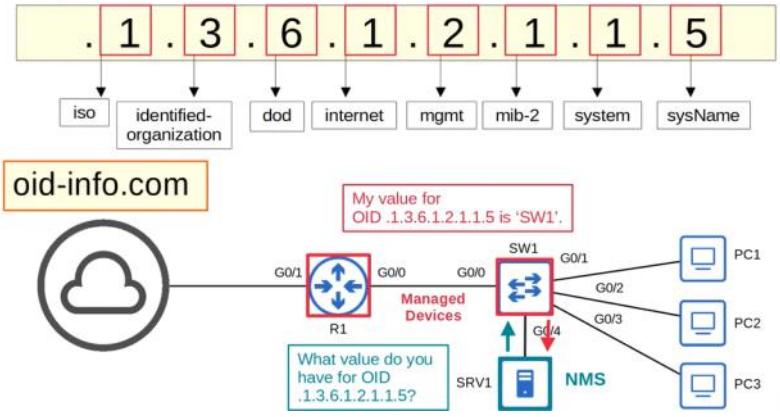
- The **SNMP Manager** is the software on the NMS that interacts with the managed devices.
  → It receives notifications, sends requests for information, sends configuration changes, etc.

- The **SNMP Application** provides an interface for the network admin to interact with.
  → Displays alerts, statistics, charts, etc.

- The **SNMP Agent** is the SNMP software running on the managed devices that interacts with the SNMP Manager on the NMS.
  → It sends notifications to/receives messages from the NMS.

- The **Management Information Base (MIB)** is the structure that contains the variables that are managed by SNMP.
  → Each variable is identified with an Object ID (OID)
  → Example variables: Interface status, traffic throughput, CPU usage, temperature, etc.

- SNMP Object IDs are organized in a hierarchical structure.



SNMP Versions

- Many versions of SNMP have been proposed/developed, however only three major versions have achieved wide-spread use:

- **SNMPv1**
  → The original version of SNMP.

- **SNMPv2c**
  → Allows the NMS to retrieve large amounts of information in a single request, so it is more efficient.
  → 'c' refers to the 'community strings' used as passwords in SNMPv1, removed from SNMPv2, and then added back for SNMPv2c.

- **SNMPv3**
  → A much more secure version of SNMP that supports strong **encryption** and **authentication**. Whenever possible, this version should be used!

SNMP Messages

Message Class	Description	Messages
Read	Messages sent by the **NMS** to read information from the **managed devices**. (ie. What's your current CPU usage %?)	*Get* *GetNext* *GetBulk*
Write	Messages sent by the **NMS** to change information on the **managed devices.** (ie. change an IP address)	*Set*
Notification	Messages sent by the **managed devices** to alert the **NMS** of a particular event. (ie. interface going down)	*Trap* *Inform*
Response	Messages sent in response to a previous message/request.	*Response*

- **Get**
  → A request sent from the manager to the agent to retrieve the value of a variable (OID), or multiple variables. The agent will send a *Response* message with the current value of each variable.

- **GetNext**
  → A request sent from the manager to the agent to discover the available variables in the MIB.

- **GetBulk**
  → A more efficient version of the **GetNext** message (introduced in SNMPv2).

- **Set**
  → A request sent from the manager to the agent to change the value of one or more variables. The agent will send a *Response* message with the new values.

SNMP uses UDP not TCP

- **Trap**
  → A notification sent from the agent to the manager. The manager does not send a Response message to acknowledge that it received the Trap, so these messages are 'unreliable'.

- **Inform**
  → A notification message that is acknowledged with a Response message.
  → Originally used for communications between managers, but later updates allow agents to send Inform messages to managers, too.

SNMP Agent = UDP 161
SNMP Manager = UDP 162

SNMPv2c Configuration

```
R1(config)#snmp-server contact jeremy@jeremysitlab.com
R1(config)#snmp-server location Jeremy's House

R1(config)#snmp-server community Jeremy1 ro

R1(config)#snmp-server community Jeremy2 rw

R1(config)#snmp-server host 192.168.1.1 version 2c Jeremy1

R1(config)#snmp-server enable traps snmp linkdown linkup
R1(config)#snmp-server enable traps config
```

Optional information

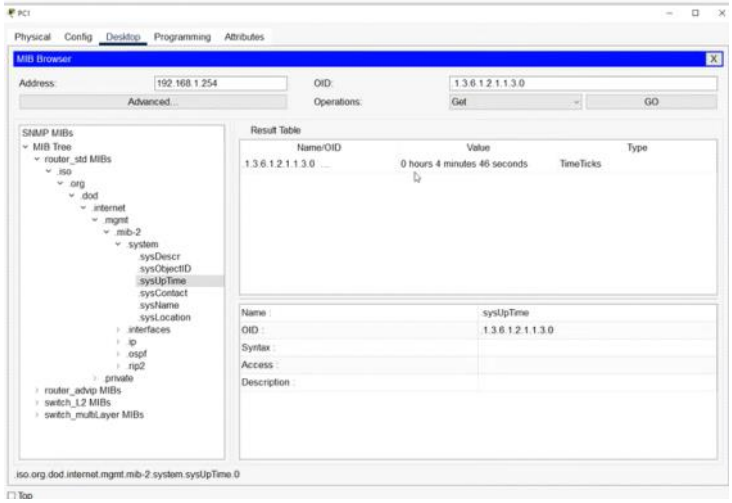
Configure the SNMP *community strings* (passwords)
ro = read only = no Set messages
rw = read/write = can use Set messages

Specify the NMS, version, and community

Configure the Trap types to send to the NMS

SNMP Summary

- SNMP helps manage devices over a network.
- **Managed Devices** are the devices being managed using SNMP, such as network devices (routers, switches, firewalls)
- **Network Management Stations (NMS)** are the SNMP 'servers' that manage the devices.
  - → NMS receives notifications from managed devices
  - → NMS changes settings on managed devices
  - → NMS checks status of managed devices
- Variables such as interface status, temperature, traffic load, host name, etc. are stored in the Management Information Base (MIB) and identified using Object IDs (OIDs)
- Main SNMP versions: SNMPv1, SNMPv2c, SNMPv3
- SNMP messages: Get, GetNext, GetBulk, Set, Trap, Inform, Response

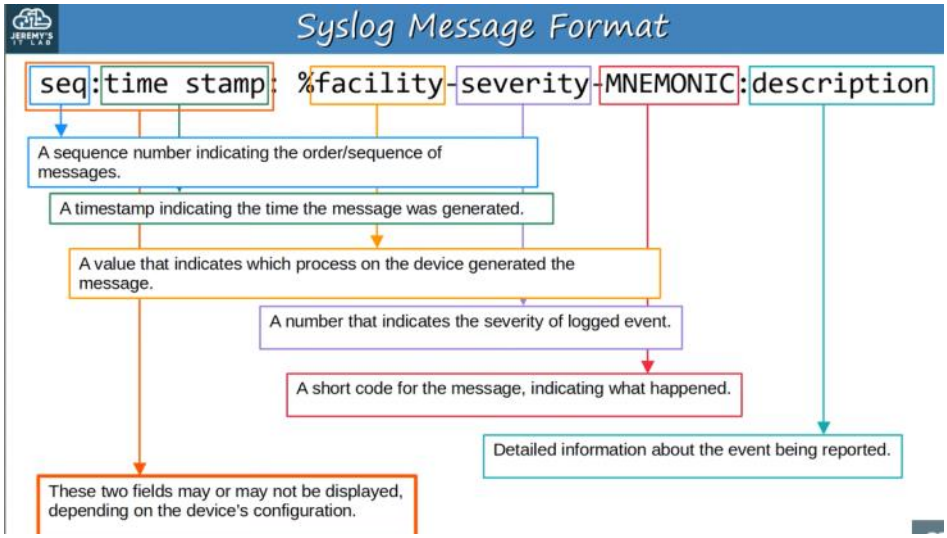


Syslog

- Syslog is an industry standard protocol for message logging.
- On network devices, Syslog can be used to log events such as changes in interface status (up⇨ down), changes in OSPF neighbor status (up⇨ down), system restarts, etc.
- The messages can be displayed in the CLI, saved in the device's RAM, or sent to an external Syslog server.

```
R1(config)#int g0/0
R1(config-if)#no shutdown
R1(config-if)#
*Feb 11 03:02:55.304: %LINK-3-UPDOWN: Interface GigabitEthernet0/0, changed state to up
*Feb 11 03:02:56.305: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0, changed state to up
```

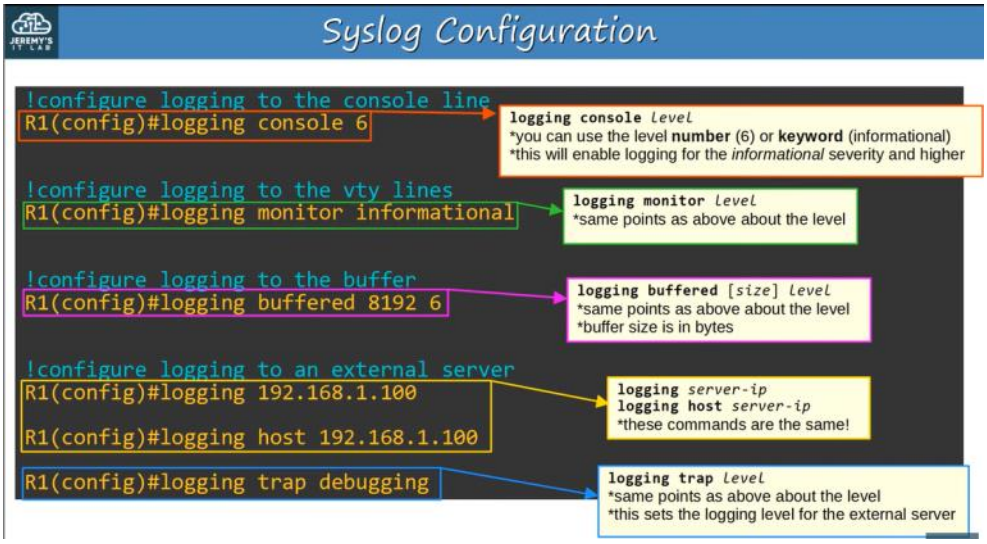
- Logs are essential when troubleshooting issues, examining the cause of incidents, etc.
- Syslog and SNMP are both used for monitoring and troubleshooting of devices. They are complementary, but their functionalities are different.



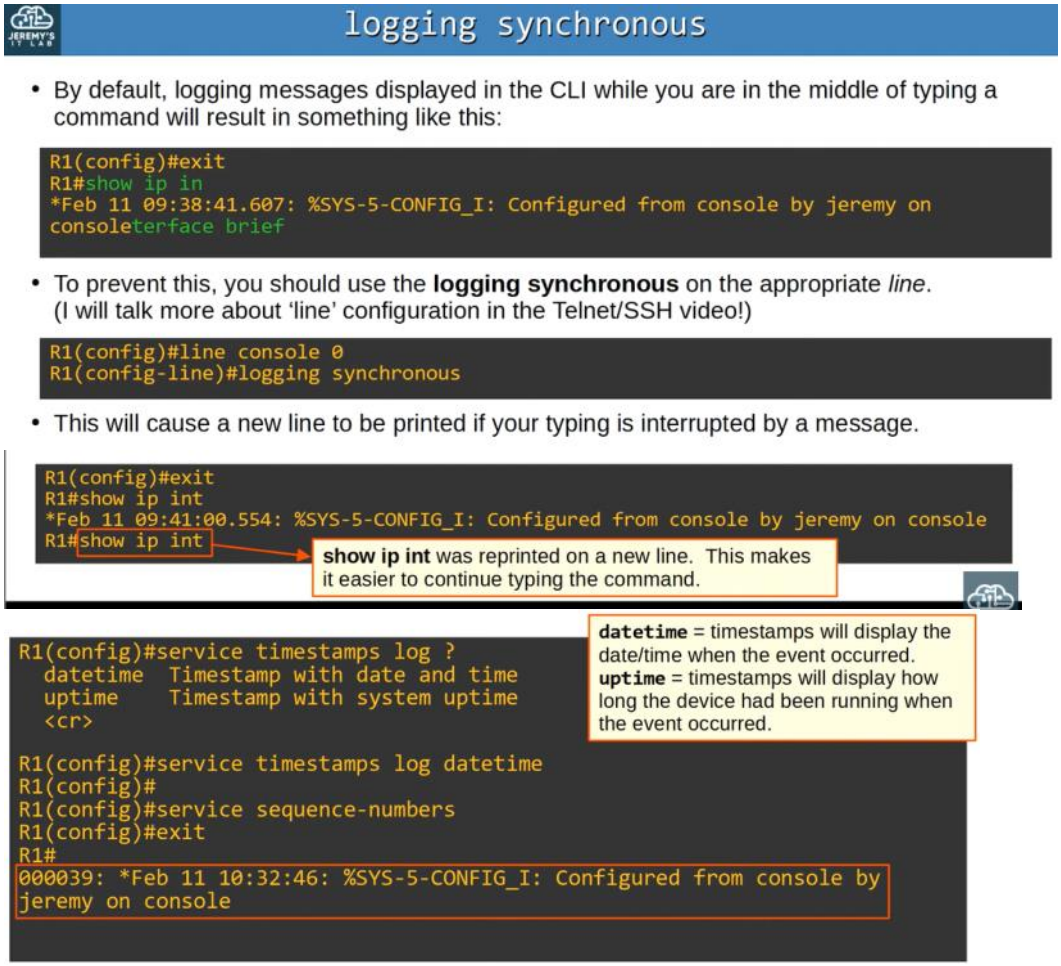
Keyword	Description
Emergency	System is unusable
Alert	Action must be taken immediately
Critical	Critical conditions
Error	Error conditions
Warning	Warning conditions
Notice	Normal but significant condition (**Notification**)
Informational	Informational messages
Debugging	Debug-level messages

```
*Feb 11 03:02:55.304: %LINK-3-UPDOWN: Interface GigabitEthernet0/0, changed state to up
*Feb 11 05:04:39.606: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.1.2 on GigabitEthernet0/0 from
LOADING to FULL, Loading Done
000043: *Feb 11 05:06:43.331: %SYS-5-CONFIG_I: Configured from console by jeremy on console
```

- **Console line:** Syslog messages will be displayed in the CLI when connected to the device via the console port. By default, all messages (level 0 – level 7) are displayed.
- **VTY lines:** Syslog messages will be displayed in the CLI when connected to the device via *Telnet/SSH* (coming in a later video). Disabled by default.
- **Buffer:** Syslog messages will be saved to RAM. By default, all messages (level 0 – level 7) are displayed.
  → You can view the messages with **`show logging`**.
- **External server:** You can configure the device to send Syslog messages to an external server.
  *Syslog servers will listen for messages on **UDP port 514**. Remember that port number!



When connecting via telnet or ssh we need to use everytime we connect this command
R1# terminal monitor



Console Port Security – login

- By default, no password is needed to access the CLI of a Cisco IOS device via the console port
- You can configure a password on the *console line*.
  → A user will have to enter a password to access the CLI via the console port.

```
R1(config)#line console 0
R1(config-line)#password ccna
R1(config-line)#login
R1(config-line)#end
R1#exit

R1 con0 is now available

Press RETURN to get started.

User Access Verification

Password:
R1>
```

There is only a single console *line*, so the number is always 0.

Configure the console line's password.

Tell the device to require a user to enter the configured password to access the CLI via the console port.

The password isn't displayed as you type it.

- Alternatively, you can configure the console line to require users to login using one of the configured usernames on the device.

```
R1(config)#username jeremy secret ccnp
R1(config)#line console 0
R1(config-line)#login local
R1(config-line)#end
R1#exit


R1 con0 is now available

Press RETURN to get started.

User Access Verification

Username: jeremy
Password:
R1>
```

Tell the device to require a user to login using one of the configured usernames on the device.

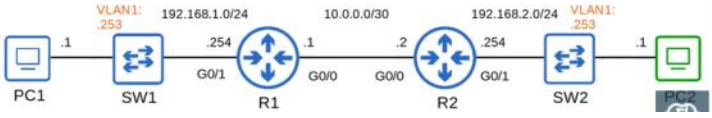
Layer 2 Switch – Management IP

- Layer 2 switches don't perform packet routing and don't build a routing table. They aren't IP routing aware.
- However, you can assign an IP address to an SVI to allow remote connections to the CLI of the switch (using Telnet or SSH).

```
SW1(config)#interface vlan1
SW1(config-if)#ip address 192.168.1.253 255.255.255.0
SW1(config-if)#no shutdown
SW1(config-if)#exit
SW1(config)#ip default-gateway 192.168.1.254
```

Configure the IP address on the SVI in the same way as on a multilayer switch.
Enable the interface if necessary.

Configure the switch's default gateway.
In this case, PC2 isn't in the same LAN as SW1.
If SW1 doesn't have a default gateway, it can't communicate with PC2.



Telnet

Telnet (Teletype Network) is a protocol used to remotely access the CLI of a remote host.

Telnet was developed in 1969.

Telnet has been largely replaced by SSH, which is more secure.

Telnet sends data in plain text. No encryption!

The Telnet *server* (the device being connected to) listens for Telnet traffic on **TCP port 23**.

Telnet Configuration

```
SW1(config)#enable secret ccna
SW1(config)#username jeremy secret ccna
SW1(config)#access-list 1 permit host 192.168.2.1
SW1(config)#line vty 0 15
SW1(config-line)#login local
SW1(config-line)#exec-timeout 5 0
SW1(config-line)#transport input telnet
SW1(config-line)#access-class 1 in
```

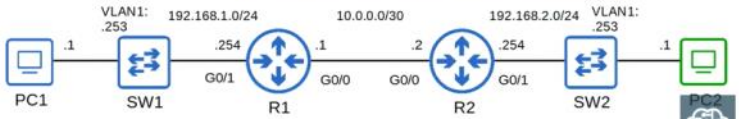
If an enable password/secret isn't configured, you won't be able to access privileged exec mode when connecting via Telnet.

Configure an ACL to limit which devices can connect to the *VTY lines*.

Telnet/SSH access is configured on the VTY lines. There are 16 lines available, so up to 16 users can be connected at once. (VTY stands for Virtual TeleType)

transport input telnet allows only Telnet connections.
transport input ssh allows only SSH connections.
transport input telnet ssh allows both.
transport input all allows all connections.
transport input none allows no connections.

Apply the ACL to the VTY lines.
***access-class** applies an ACL to the VTY lines,
ip access-group applies an ACL to an interface.

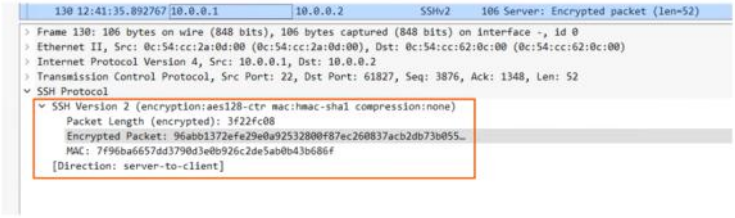


SSH (Secure Shell)

- SSH (Secure Shell) was developed in 1995 to replace less secure protocols like Telnet.

> In computing, a **shell** is a computer program which exposes an operating system's services to a human user or other program. In general, operating system shells use either a command-line interface (CLI) or graphical user interface (GUI), depending on a computer's role and particular operation. It is named a shell because it is the outermost layer around the operating system.[1] [2]

- SSHv2, a major revision of SSHv1, was released in 2006.
- If a device supports both version 1 and version 2, it is said to run 'version 1.99'.
- Provides security features such as data encryption and authentication.



The SSH *server* (the device being connected to) listens for SSH traffic on **TCP port 22**.

SSH Configuration: Check SSH Support

```
SW1#show version
Cisco IOS Software, vios_l2 Software (vios_l2-ADVENTERPRISEK9-M), Version 15.2(4.0.55)E, TEST
ENGINEERING ESTG_WEEKLY BUILD, synced to  END_OF_FLO_ISP
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2015 by Cisco Systems, Inc.
Compiled Tue 28-Jul-15 18:52 by sasyamal

SW1#show ip ssh
SSH Disabled - version 1.99
%Please create RSA keys to enable SSH (and of atleast 768 bits for SSH v2).
Authentication methods:publickey,keyboard-interactive,password
Authentication Publickey Algorithms:x509v3-ssh-rsa,ssh-rsa
Hostkey Algorithms:x509v3-ssh-rsa,ssh-rsa
Encryption Algorithms:aes128-ctr,aes192-ctr,aes256-ctr,aes128-cbc,3des-cbc,aes192-cbc,aes256-cbc
MAC Algorithms:hmac-sha1,hmac-sha1-96
Authentication timeout: 120 secs; Authentication retries: 3
Minimum expected Diffie Hellman key size : 1024 bits
IOS Keys in SECSH format(ssh-rsa, base64 encoded): NONE
```

- IOS images that support SSH will have 'K9' in their name.
- Cisco exports NPE (No Payload Encryption) IOS images to countries that have restrictions on encryption technologies.
- NPE IOS images do not support cryptographic features such as SSH.

SSH Configuration: RSA Keys

- To enable and use SSH, you must generate an RSA public and private key pair.
- The keys are used for data encryption/decryption, authentication, etc.

```
SW1(config)#ip domain name jeremysitlab.com
SW1(config)#crypto key generate rsa
The name for the keys will be: SW1.jeremysitlab.com
Choose the size of the key modulus in the range of 360 to 4096 for your
  General Purpose Keys. Choosing a key modulus greater than 512 may take
  a few minutes.

How many bits in the modulus [512]: 2048
% Generating 2048 bit RSA keys, keys will be non-exportable...
[OK] (elapsed time was 1 seconds)

SW1(config)#
*Feb 21 04:22:35.778: %SSH-5-ENABLED: SSH 1.99 has been enabled

SW1(config)#do show ip ssh
SSH Enabled - version 1.99
Authentication methods:publickey,keyboard-interactive,password
Encryption Algorithms:aes128-ctr,aes192-ctr,aes256-ctr,aes128-cbc,3des-cbc,aes192-cbc,aes256-cbc
MAC Algorithms:hmac-sha1,hmac-sha1-96
Authentication timeout: 120 secs; Authentication retries: 3
Minimum expected Diffie Hellman key size : 1024 bits
IOS Keys in SECSH format(ssh-rsa, base64 encoded): SW1.jeremysitlab.com
```

The **FQDN** of the device is used to name the RSA keys.
FQDN = Fully Qualified Domain Name (host name + domain name)

Generate the RSA keys.
crypto key generate rsa modulus *length*
is an alternate method.
*length must be 768 bits or greater for SSHv2

SSH Configuration: VTY Lines

```
SW1(config)#enable secret ccna
SW1(config)#username jeremy secret ccna
SW1(config)#access-list 1 permit host 192.168.2.1
SW1(config)#ip ssh version 2
SW1(config)#line vty 0 15
SW1(config-line)#login local
SW1(config-line)#exec-timeout 5 0
SW1(config-line)#transport input ssh
SW1(config-line)#access-class 1 in
```

(optional, but recommended) Restrict SSH to version 2 only.

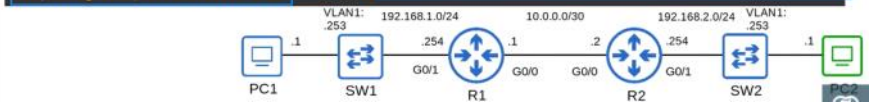
Configure all VTY lines, just like Telnet.

Enable local user authentication.
*you cannot use **login** for SSH, only **login local**.

(optional, but recommended) Configure the exec timeout.

Best practice is to limit VTY line connections to SSH only.

(optional, but recommended) Apply the ACL to restrict VTY line connections.



1) Configure host name
2) Configure DNS domain name
3) Generate RSA key pair
4) Configure enable PW, username/PW
5) Enable SSHv2 (only)
6) Configure VTY lines

Connect: **ssh -l** *username ip-address* OR **ssh** *username@ip-address*

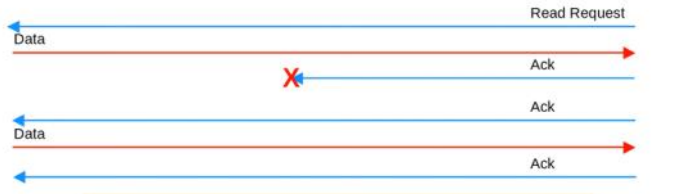
FTP & TFTP

- FTP (File Transfer Protocol) and TFTP (Trivial File Transfer Protocol) are industry standard protocols used to transfer files over a network.
- They both use a client-server model.
  → Clients can use FTP or TFTP to copy files from a server.
  → Clients can use FTP or TFTP to copy files to a server.
- As a network engineer, the most common use for FTP/TFTP is in the process of upgrading the operating system of a network device.
- You can use FTP/TFTP to download the newer version of IOS from a server, and then reboot the device with the new IOS image.

- TFTP was first standardized in 1981.
- Named 'Trivial' because it is simple and has only basic features compared to FTP.
  → Only allows a client to copy a file to or from a server.
- Was released after FTP, but is not a replacement for FTP. It is another tool to use when lightweight simplicity is more important than functionality.
- No authentication (username/PW), so servers will respond to all TFTP requests.
- No encryption, so all data is sent in plain text.
- Best used in a controlled environment to transfer small files quickly.
- TFTP servers listen on **UDP port 69**.
- UDP is connectionless and doesn't provide reliability with retransmissions.
- However, TFTP has similar built-in features within the protocol itself.

TFTP Reliability

- Every TFTP data message is acknowledged.
  → If the client is transferring a file to the server, the server will send Ack messages.
  → If the server is transferring a file to the client, the client will send Ack messages.
- Timers are used, and if an expected message isn't received in time, the waiting device will re-send its previous message.



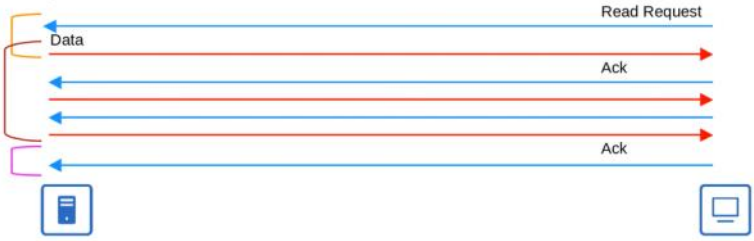
TFTP uses 'lock-step' communication. The client and server alternately send a message and then wait for a reply. (+retransmissions are sent as needed)

TFTP 'Connections'

- TFTP file transfers have three phases:
  1: **Connection**: TFTP client sends a request to the server, and the server responds back, initializing the connection.

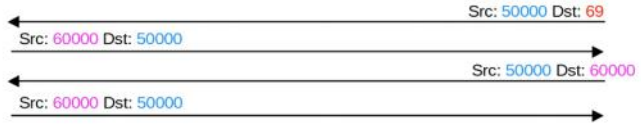
  2: **Data Transfer**: The client and server exchange TFTP messages. One sends data and the other sends acknowledgments.

  3: **Connection Termination**: After the last data message has been sent, a final acknowledgment is sent to terminate the connection.



TFTP TID

- When the client sends the first message to the server, the destination port is UDP 69 and the source is a random ephemeral port.
- This random port is called a 'Transfer Identifier' (TID) and identifies the data transfer.
- The server then also selects a random TID to use as the source port when it replies, **not 69**.
- When the client sends the next message, the destination port will be the server's TID, **not 69**.



*This is beyond the scope of the CCNA, but is an interesting part of TFTP's operation.

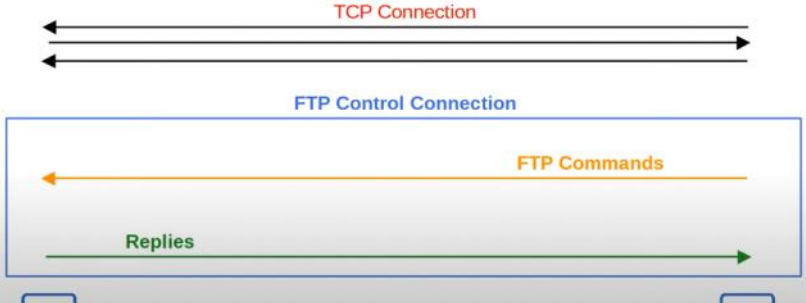
File Transfer Protocol

- FTP was first standardized in 1971.
- FTP uses **TCP ports 20 and 21**.
- Usernames and passwords are used for authentication, however there is no encryption.
- For greater security, FTPS (FTP over SSL/TLS) can be used. ← Upgrade to FTP
- SSH File Transfer Protocol (SFTP) can also be used for greater security. ← New protocol
- FTP is more complex than TFTP and allows not only file transfers, but clients can also navigate file directories, add and remove directories, list files, etc.
- The client sends *FTP commands* to the server to perform these functions.

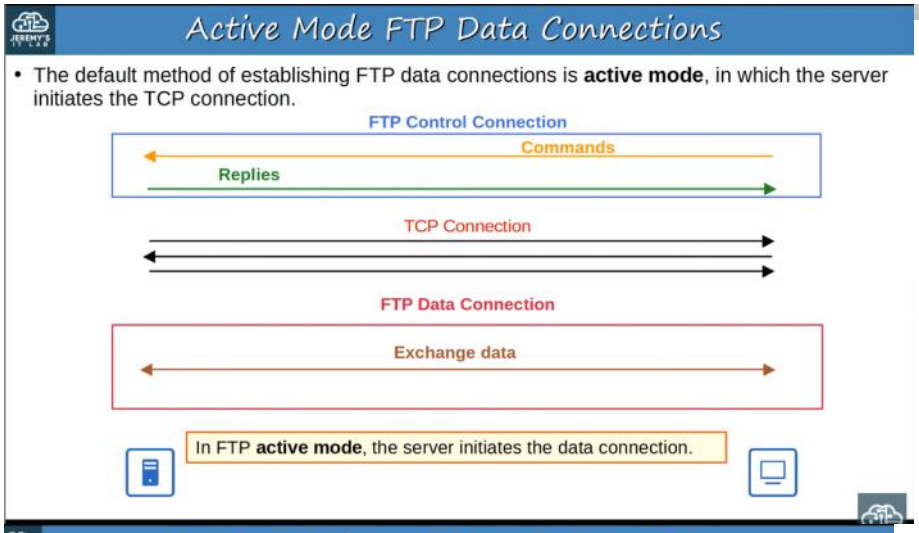
https://en.wikipedia.org/wiki/List_of_FTP_commands

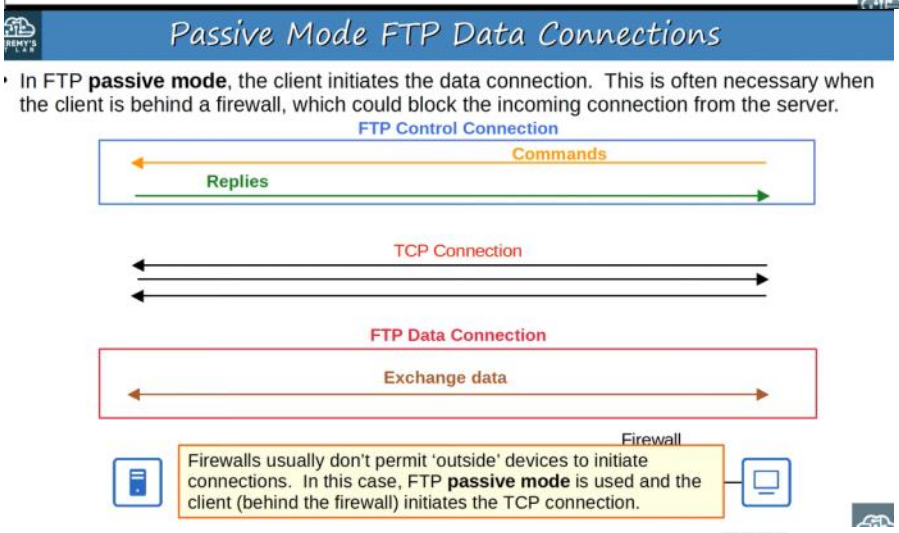
FTP Control Connections

- FTP uses two types of connections:
  → An **FTP control** connection (**TCP 21**) is established and used to send FTP commands and replies.
  → When files or data are to be transferred, separate **FTP data** (**TCP 20**) connections are established and terminated as needed.



Active Mode FTP Data Connections





FTP	TFTP
-Uses TCP (20 for data, 21 for control) for connection-based communication	-Uses UDP (69) for connectionless communication (although a basic form of 'connection' is used within the protocol itself)
-Clients can use FTP commands to perform various actions, not just copy files	-Clients can only copy files to or from the server
-Username/PW authentication	-No authentication
-More complex	-Simpler

R1# show file systems (shows different file systems available)

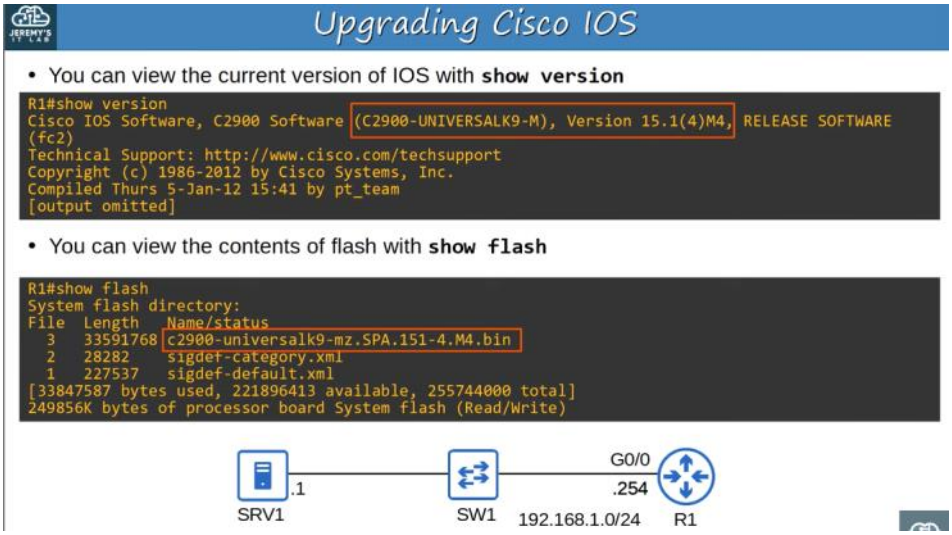
```
Router#show file systems
File Systems:

      Size(b)       Free(b)      Type  Flags  Prefixes
*   2142715904    1994403840      disk    rw   flash0: flash:#
          -             -         disk    rw   flash1:
     966656        962560         disk    rw   flash2:#
          -             -         disk    rw   flash3:
          -             -       opaque    rw   archive:
          -             -       opaque    rw   system:
     262144        256791        nvram    rw   nvram:
          -             -       opaque    rw   tmpsys:
          -             -      network    rw   snmp:
          -             -       opaque    rw   null:
          -             -      network    rw   tftp:
          -             -       opaque    ro   xmodem:
          -             -       opaque    ro   ymodem:
          -             -       opaque    wo   syslog:
          -             -      network    rw   rcp:
          -             -      network    rw   pram:
          -             -      network    rw   ftp:
[output omitted]
```

disk: Storage devices such as flash memory.

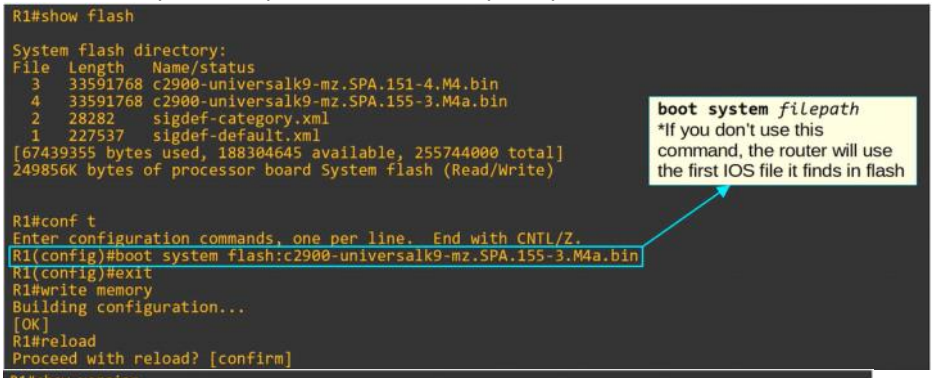
opaque: Used for internal functions

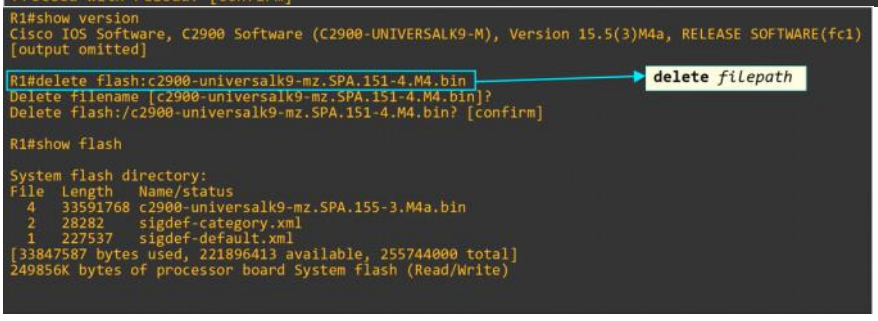
nvram: Internal NVRAM. The startup-config file is stored here.

network: Represents external file systems, for example external FTP/TFTP servers.

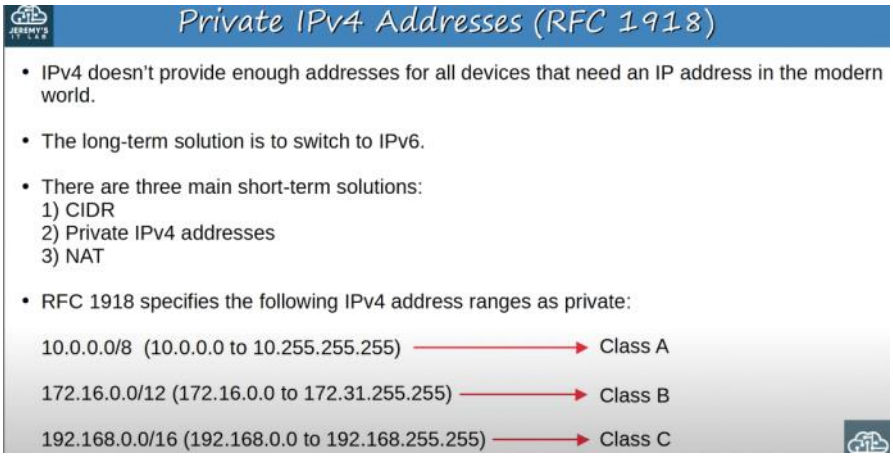


R1# copy source destination (copy tftp: flash:)
(enter the tftp server ip in the command prompt and the source filename , dest filename)



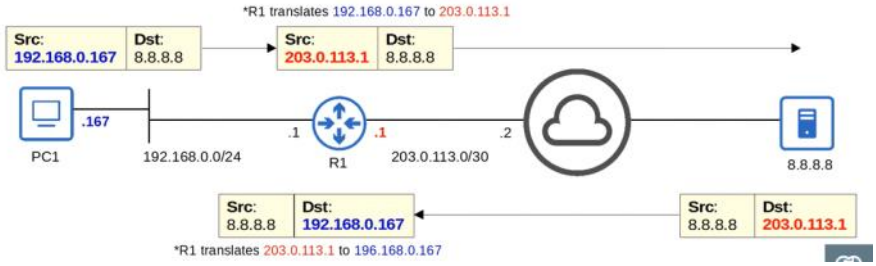


R1(config)# ip ftp username cisco
R1(config)# ip ftp password cisco
R1# copy ftp: flash:
(give the host ip and filenames in the prompts)



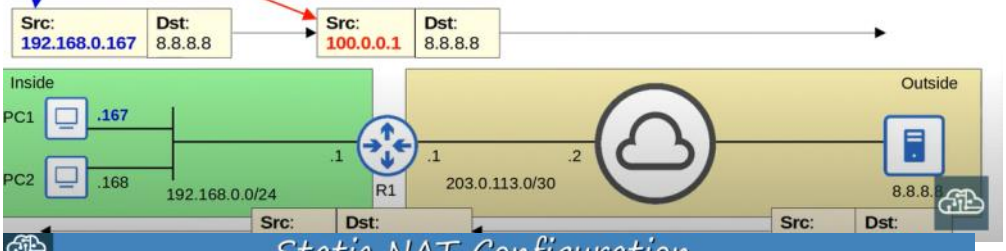
Network Address Translation (NAT)

- Network Address Translation (NAT) is used to modify the source and/or destination IP addresses of packets.
- There are various reasons to use NAT , but the most common reason is to allow hosts with private IP addresses to communicate with other hosts over the Internet.
- For the CCNA you have to understand **source NAT** and how to configure it on Cisco routers.



Static NAT

- **Static NAT** involves statically configuring one-to-one mappings of private IP addresses to public IP addresses.
- An *inside local* IP address is mapped to an *inside global* IP address.
  - **Inside Local** = The IP address of the *inside* host, from the perspective of the local network
    *the IP address actually configured on the inside host, usually a private address
  - **Inside Global** = The IP address of the *inside* host, from the perspective of *outside* hosts
    *the IP address of the inside host after NAT, usually a public address



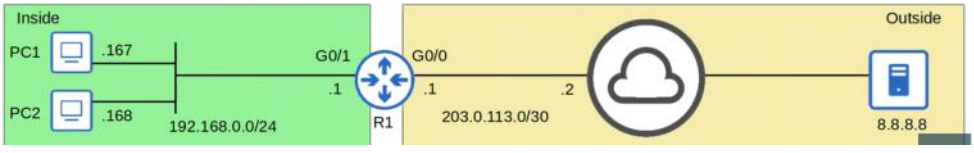
Static NAT Configuration

```
R1(config)#int g0/1
R1(config-if)#ip nat inside
```
Define the 'inside' interface(s) connected to the internal network.

```
R1(config-if)#int g0/0
R1(config-if)#ip nat outside
R1(config-if)#exit
```
Define the 'outside' interface(s) connected to the external network.

```
R1(config)#ip nat inside source static 192.168.0.167 100.0.0.1
R1(config)#ip nat inside source static 192.168.0.168 100.0.0.2
R1(config)#exit
```
Configure the one-to-one IP address mappings.
ip nat inside source static *inside-local-ip inside-global-ip*

```
R1#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
udp 100.0.0.1:56310    192.168.0.167:56310 8.8.8.8:53        8.8.8.8:53
--- 100.0.0.1          192.168.0.167      ---                ---
udp 100.0.0.2:62321    192.168.0.168:62321 8.8.8.8:53        8.8.8.8:53
--- 100.0.0.2          192.168.0.168      ---                ---
```



- **Inside Local** = The IP address of the *inside* host, from the perspective of the local network
  *the IP address actually configured on the inside host, usually a private address
- **Inside Global** = The IP address of the *inside* host, from the perspective of *outside* hosts
  *the IP address of the inside host after NAT, usually a public address
- **Outside Local** = The IP address of the *outside* host, from the perspective of the local network
- **Outside Global** = The IP address of the *outside* host, from the perspective of the outside network

```
R1#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
udp 100.0.0.1:56310    192.168.0.167:56310 8.8.8.8:53         8.8.8.8:53
--- 100.0.0.1          192.168.0.167       ---                ---
udp 100.0.0.2:62321    192.168.0.168:62321 8.8.8.8:53         8.8.8.8:53
--- 100.0.0.2          192.168.0.168       ---                ---

R1#clear ip nat translation *

R1#show ip nat translations
Pro Inside global      Inside local       Outside local      Outside global
--- 100.0.0.1          192.168.0.167       ---                ---
--- 100.0.0.2          192.168.0.168       ---                ---
```

R1# show ip nat statistics

Dynamic NAT

- In **dynamic NAT**, the router dynamically maps *inside local* addresses to *inside global* addresses as needed.
- An ACL is used to identify which traffic should be translated.
  → If the source IP is **permitted** by the ACL, the source IP will be translated.
  → If the source IP is **denied** by the ACL, the source IP will NOT be translated. *the traffic will NOT be dropped!
- A NAT pool is used to define the available *inside global* addresses.

On R1:
ACL 1: permit 192.168.0.0/24
deny any
POOL1: 100.0.0.1 to 100.0.0.10
If a packet with a source IP permitted by **ACL 1** arrives, translate the source IP to an address from **POOL1**.

- Although they are dynamically assigned, the mappings are still one-to-one (one *inside local* IP address per *inside global* IP address).
- If there aren't enough *inside global* IP addresses available (=all are currently being used), it is called 'NAT pool exhaustion'.
  → If a packet from another inside host arrives and needs NAT but there are no available addresses, the router will drop the packet.
  → The host will be unable to access outside networks until one of the *inside global* IP addresses becomes available.

→ Dynamic NAT entries will time out automatically if not used, or you can clear them manually.

Dynamic NAT Configuration

```
R1(config)#int g0/1
R1(config-if)#ip nat inside
```
Define the 'inside' interface(s) connected to the internal network.

```
R1(config-if)#int g0/0
R1(config-if)#ip nat outside
R1(config-if)#exit
```
Define the 'outside' interface(s) connected to the external network.

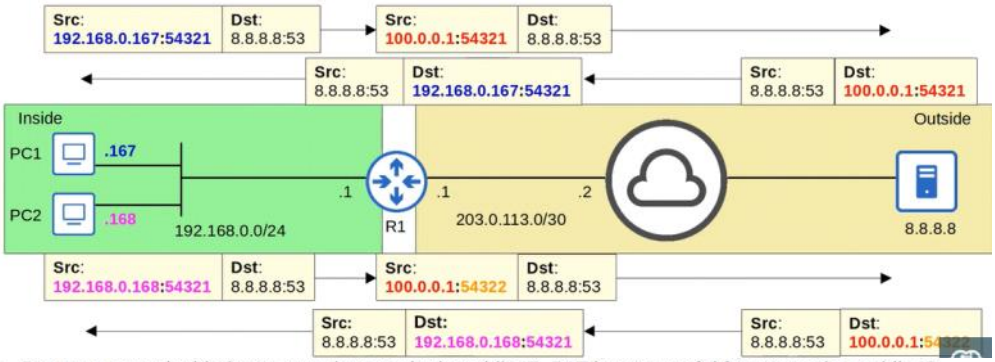
```
R1(config)#access-list 1 permit 192.168.0.0 0.0.0.255
```
Define the traffic that should be translated.
*Traffic permitted by this ACL will be translated.

```
R1(config)#ip nat pool POOL1 100.0.0.0 100.0.0.255 prefix-length 24
```
Define the pool of inside global IP addresses.
*instead of **prefix-length 24**, you can use **netmask 255.255.255.0**

```
R1(config)#ip nat inside source list 1 pool POOL1
```
Configure dynamic NAT by mapping the ACL to the pool.

PAT (NAT Overload)

- **PAT** (aka **NAT overload**) translates both the IP address and the port number (if necessary).
- By using a unique port number for each communication flow, a single public IP address can be used by many different internal hosts. (port number are 16 bits = over 65,000 available port numbers).
- The router will keep track of which *inside local* address is using which *inside global* address and port.



- Because many inside hosts can share a single public IP, PAT is very useful for preserving public IP addresses, and it is used in networks all over the world.

PAT Configuration (pool)

```
R1(config)#int g0/1
R1(config-if)#ip nat inside
```
Define the 'inside' interface(s) connected to the internal network.

```
R1(config-if)#int g0/0
R1(config-if)#ip nat outside
R1(config-if)#exit
```
Define the 'outside' interface(s) connected to the external network.

```
R1(config)#access-list 1 permit 192.168.0.0 0.0.0.255
```
Define the traffic that should be translated.
*Traffic permitted by this ACL will be translated.

```
R1(config)#ip nat pool POOL1 100.0.0.0 100.0.0.3 prefix-length 24
```
Define the pool of inside global IP addresses.

```
R1(config)#ip nat inside source list 1 pool POOL1 overload
```
Configure PAT by mapping the ACL to the pool and using the **overload** keyword at the end.

PAT Configuration (interface)

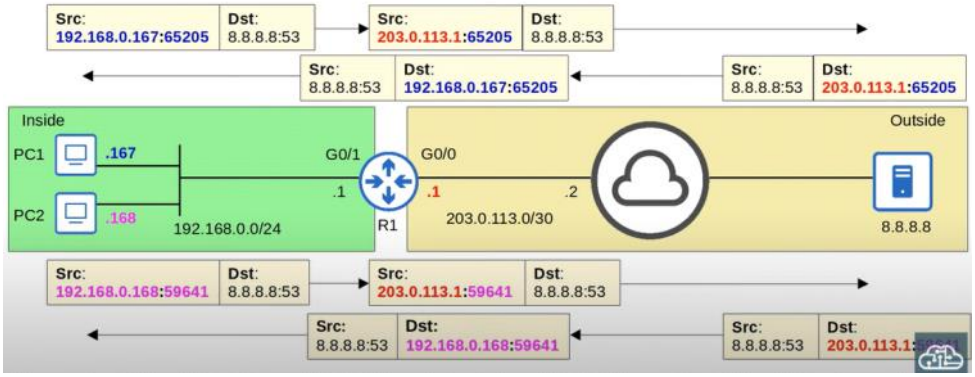
```
R1(config)#int g0/1
R1(config-if)#ip nat inside
```
Define the 'inside' interface(s) connected to the internal network.

```
R1(config-if)#int g0/0
R1(config-if)#ip nat outside
R1(config-if)#exit
```
Define the 'outside' interface(s) connected to the external network.

```
R1(config)#access-list 1 permit 192.168.0.0 0.0.0.255
```
Define the traffic that should be translated.
*Traffic permitted by this ACL will be translated.

```
R1(config)#ip nat inside source list 1 interface g0/0 overload
```
Configure PAT by mapping the ACL to the interface and enabling **overload**.

PAT Configuration (interface)



IP Phones

- Traditional phones operate over the *public switched telephone network* (PSTN).
- Sometimes this is called POTS (Plain Old Telephone Service).
- IP phones use VoIP (Voice over IP) technologies to enable phone calls over an IP network, such as the Internet.
- IP phones are connected to a switch just like any other end host.

IP Phones

- IP phones have an internal 3-port switch.
  - → 1 port is the 'uplink' to the external switch.
  - → 1 port is the 'downlink' to the PC.
  - → 1 port connects internally to the phone itself.
- This allows the PC and the IP phone to share a single switch port. Traffic from the PC passes through the IP phone to the switch.
- It is recommended to separate 'voice' traffic (from the IP phone) and 'data' traffic (from the PC) by placing them in separate VLANs.

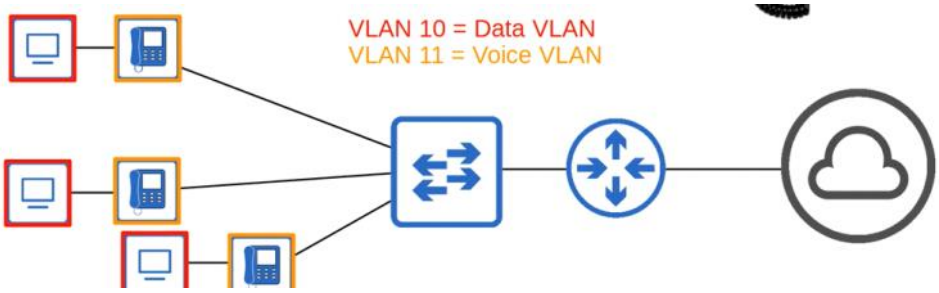
→ This can be accomplished using a *voice VLAN*
→ Traffic from the PC will be untagged, but traffic from the phone will be tagged with a VLAN ID

IP Phones / Voice VLAN

```
SW1(config)#interface gigabitethernet0/0
SW1(config-if)#switchport mode access
SW1(config-if)#switchport access vlan 10
SW1(config-if)#switchport voice vlan 11

SW1#show interfaces g0/0 switchport
Name: Gi0/0
Switchport: Enabled
Administrative Mode: static access
Operational Mode: static access
Administrative Trunking Encapsulation: negoti
Operational Trunking Encapsulation: native
Negotiation of Trunking: Off
Access Mode VLAN: 10 (VLAN0010)
Trunking Native Mode VLAN: 1 (default)
Administrative Native VLAN tagging: enabled
Voice VLAN: 11 (VLAN0011)
![output omitted]
```

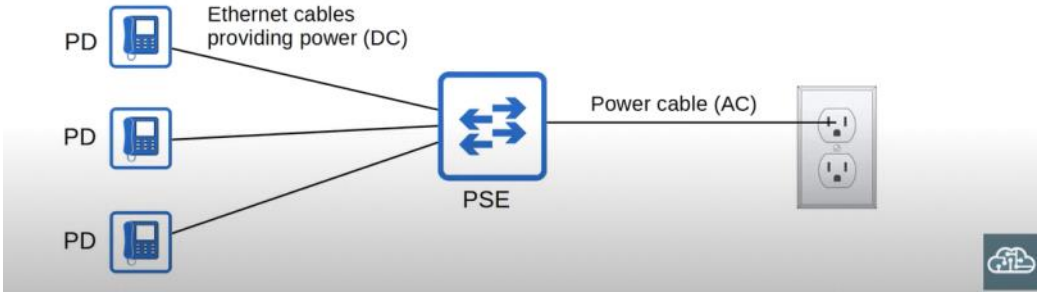
PC1 will send traffic untagged, as normal. SW1 will use CDP to tell PH1 to tag PH1's traffic in VLAN 11.

Although the interface sends/receives traffic from two VLANs, it is not considered a trunk port. It is considered an access port.



Power over Ethernet (PoE)

- PoE allows Power Sourcing Equipment (PSE) to provide power to Powered Devices (PD) over an Ethernet cable.
- Typically the PSE is a switch and the PDs are IP phones, IP cameras, wireless access points, etc.
- The PSE receives AC power from the outlet, converts it to DC power, and supplies that DC power to the PDs.



- Too much electrical current can damage electrical devices.
- PoE has a process to determine if a connected device needs power, and how much power it needs.

  → When a device is connected to a PoE-enabled port, the PSE (switch) sends low power signals, monitors the response, and determines how much power the PD needs.

  → If the device needs power, the PSE supplies the power to allow the PD to boot.

  → The PSE continues to monitor the PD and supply the required amount of power (but not too much!)

- *Power policing* can be configured to prevent a PD from taking too much power.

  → **`power inline police`** configures power policing with the default settings: disable the port and send a Syslog message if a PD draws too much power.
  - → equivalent to **`power inline police action err-disable`**
  - → the interface will be put in an 'error-disabled' state and can be re-enabled with **`shutdown`** followed by **`no shutdown`**.

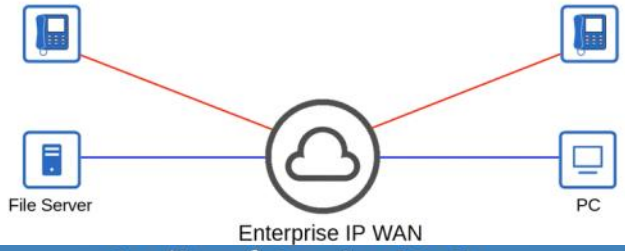
  → **`power inline police action log`** does not shut down the interface if the PD draws too much power. It will restart the interface and send a Syslog message.

```
SW1# conf t
Enter configuration commands, one per line. End with CNTL/Z.
SW1(config)# int g0/0
SW1(config-if)# power inline police action log
SW1(config-if)# end
SW1# show power inline police g0/0
Available:800(w) Used:32(w) Remaining:768(w)
Interface Admin  Oper       Admin      Oper       Cutoff Oper
          State  State      Police     Police     Power  Power
--------- ------ ---------- ---------- ---------- ------ -----
Gi0/0     auto   on         log        ok          17.2  16.7
```

Name	Standard #	Watts	Powered Wire Pairs
Cisco Inline Power (ILP)	Made by Cisco, not standard	7	2
PoE (Type 1)	802.3af	15	2
PoE+ (Type 2)	802.3at	30	2
UPoE (Type 3)	802.3bt	60	4
UPoE+ (Type 4)	802.3bt	100	4

Quality of Service (QoS)

- Modern networks are typically *converged networks* in which IP phones, video traffic, regular data traffic, etc all share the same IP network.
- This enable cost savings as well as more advanced features for voice and video traffic, for example integrations with collaboration software (Cisco WebEx, Microsoft Teams, etc).
- However, the different kinds of traffic now have to compete for bandwidth.
- QoS is a set of tools used by network devices to apply different treatment to different packets.



Quality of Service (QoS)

- QoS is used to manage the following characteristics of network traffic:
  1) **Bandwidth**
  → The overall capacity of the link, measured in bits per second (Kbps, Mbps, Gbps, etc)
  → QoS tools allow you to reserve a certain amount of a link's bandwidth for specific kinds of traffic.
  For example: 20% voice traffic, 30% for specific kinds of data traffic, leaving 50% for all other traffic.

  2) **Delay**
  → The amount of time it takes traffic to go from source to destination = **one-way delay**
  → The amount of time it takes traffic to go from source to destination and return = **two-way delay**

  3) **Jitter**
  → The variation in one-way delay between packets sent by the same application
  → IP phones have a 'jitter buffer' to provide a fixed delay to audio packets.

  4) **Loss**
  → The % of packets sent that do not reach their destination
  → Can be caused by faulty cables.
  → Can also be caused when a device's packet *queues* get full and the device starts discarding packets.

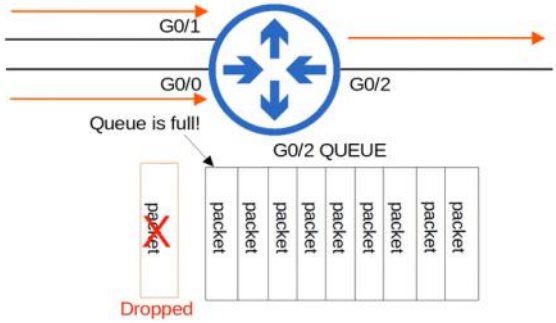
- The following standards are recommended for acceptable interactive audio (ie. phone call) quality:

  One-way delay: 150 ms or less
  Jitter: 30 ms or less
  Loss: 1% or less

- If these standards are not met, there could be a noticeable reduction in the quality of the phone call.

QoS - Queuing

- If a network device receives messages faster than it can forward them out of the appropriate interface, the messages are placed in a queue.
- By default, queued messages will be forwarded in a First In First Out (FIFO) manner.
  → Messages will be sent in the order they are received.
- If the queue is full new packets will be dropped.
- This is called *tail drop*.



- **Tail drop** is harmful because it can lead to **TCP global synchronization**.
- Review of the **TCP sliding window**:
  → Hosts using TCP use the 'sliding window' increase/decrease the rate at which they send traffic as needed.
  → When a packet is dropped it will be re-transmitted.
  → When a drop occurs, the sender will reduce the rate it sends traffic.
  → It will then gradually increase the rate again.
- When the queue fills up and **tail drop** occurs, all TCP hosts sending traffic will slow down the rate at which they send traffic.
- They will all then increase the rate at which they send traffic, which rapidly leads to more congestion, dropped packets, and the process repeats again.

- A solution to prevent tail drop and TCP global synchronization is **Random Early Detection** (RED).
- When the amount of traffic in the queue reaches a certain threshold, the device will start randomly dropping packets from select TCP flows.
- Those TCP flows that dropped packets will reduce the rate at which traffic is sent, but you will avoid global TCP synchronization, in which ALL TCP flows reduce and then increase the rate of transmission at the same time in waves.
- In standard RED, all kinds of traffic are treated the same.
- An improved version, **Weighted Random Early Detection** (WRED), allows you to control which packets are dropped depending on the traffic class.
- We will cover traffic classes and details about how QoS actually works in the next video.

You issue the **`power inline police`** command on a PoE-enabled switch port. What will happen if the connected device draws too much power from the switch?

a) A Syslog message will be generated.

b) The interface will be restarted and a Syslog message will be generated.

c) The interface will be err-disabled and a Syslog message will be generated.

Sometimes a switch port may behave like a trunk port even if it is not configured as a trunk port. For example, an access port might accept frames from VLANs different from the VLAN to which it is assigned. This is called VLAN leaking, which is caused by a mismatched native VLAN or misconfigured trunk

To troubleshoot issues when a trunk is not forming or when VLAN leaking is occurring, proceed as follows:

Use the show interfaces trunk command to check whether the local and peer native VLANs match. If the native VLAN does not match on both sides, VLAN leaking occurs.
Use the show interfaces trunk command to check whether a trunk has been established between switches. Statically configure trunk links whenever possible. Cisco Catalyst switch ports use DTP by default and attempt to negotiate a trunk link.

CONCEPTS PART 9

Sunday, February 19, 2023 2:28 PM

QUALITY OF SERVICE

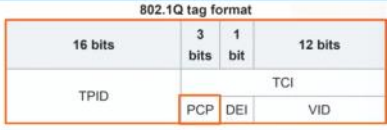
Classification

- The purpose of QoS is to give certain kinds of network traffic priority over others during congestion.
- **Classification** organizes network traffic (packets) into traffic classes (categories).
- Classification is fundamental to QoS. To give priority to certain types of traffic, you have to identify which types of traffic to give priority to.
- There are many methods of classifying traffic. Some examples:
  → An ACL. Traffic which is permitted by the ACL will be given certain treatment, other traffic will not.
  → **NBAR** (Network Based Application Recognition) performs a *deep packet inspection*, looking beyond the Layer 3 and Layer 4 information up to Layer 7 to identify the specific kind of traffic.
  → In the Layer 2 and Layer 3 headers there are specific fields used for this purpose.
- The **PCP** (Priority Code Point) field of the 802.1Q tag (in the Ethernet header) can be used to identify high/low priority traffic.
  → Only when there is a dot1q tag!
- The **DSCP** (Differentiated Services Code Point) field of the IP header can also be used to identify high/low priority traffic.

PCP/CoS

Ethernet Header

802.1Q tag format



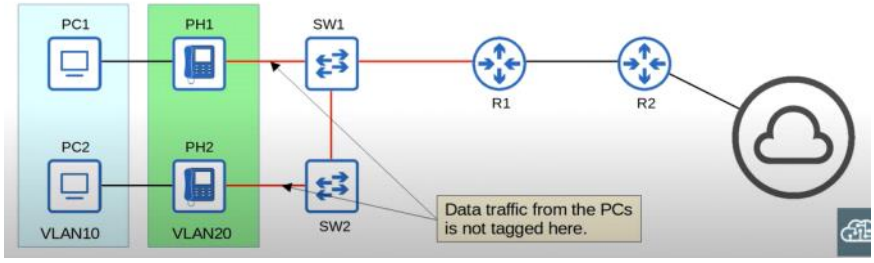
- PCP is also known as CoS (Class of Service). Its use is defined by IEEE 802.1p.
- 3 bits = 8 possible values (2^3 = 8).

'Best effort' delivery means there is no guarantee that data is delivered or that it meets any QoS standard. This is regular traffic, not high-priority.

IP phones **mark** call signaling traffic (used to establish calls) as PCP3.
They **mark** the actual voice traffic as PCP5.

PCP value	Traffic types
0	Best effort (default)
1	Background
2	Excellent effort
3	Critical applications
4	Video
5	Voice
6	Internetwork control
7	Network control

- Because PCP is found in the dot1q header, it can only be used over the following connections:
  → trunk links
  → access links with a voice VLAN
- In the diagram below, traffic between R1 and R2, or between R2 and external destinations will not have a dot1q tag. So, traffic over those links PCP cannot be marked with a PCP value.



ECN in ipv4 header (explicit congestion notification)

Now DSCP and ECN fields are used for ToS Byte for priority markikng etc..

Before IPP(ip precedence) is used in that place

IP Precedence

IPP (IP Precedence)	(Defined for various purposes, mostly unused)	3 bits = 8 values (0-7)

- Standard IPP markings are similar to PCP:
  - → 6 and 7 are reserved for 'network control 'traffic (ie. OSPF messages between routers)
  - → 5 = voice
  - → 4 = video
  - → 3 = voice signaling
  - → 0 = best effort
- With 6 and 7 reserved, 6 possible values remain.
- Although 6 values is sufficient for many networks, the QoS requirements of some networks demand more flexibility.

DSCP

DSCP (Differentiated Services Code Point)	ECN	6 bits = 64 values (0-63)

- RFC 2474 (1998) defines the DSCP field, and other 'DiffServ' RFCs elaborate on its use.
- With IPP updated to DSCP, new standard markings had to be decided upon.
  - → By having generally agreed upon standard markings for different kinds of traffic, QoS design & implementation is simplified, QoS works better between ISPs and enterprises, among other benefits.

- You should be aware of the following standard markings:
  - → Default Forwarding (DF) – best effort traffic
  - → Expedited Forwarding (EF) – low loss/latency/jitter traffic (usually voice)
  - → Assured Forwarding (AF) – A set of 12 standard values
  - → Class Selector (CS) – A set of 8 standard values, provides backward compatibility with IPP

```
R1(config)#class-map TEST
R1(config-cmap)#match dscp ?
  <0-63>   Differentiated services codepoint value
  af11     Match packets with AF11 dscp (001010)
  af12     Match packets with AF12 dscp (001100)
  af13     Match packets with AF13 dscp (001110)
  af21     Match packets with AF21 dscp (010010)
  af22     Match packets with AF22 dscp (010100)
  af23     Match packets with AF23 dscp (010110)
  af31     Match packets with AF31 dscp (011010)
  af32     Match packets with AF32 dscp (011100)
  af33     Match packets with AF33 dscp (011110)
  af41     Match packets with AF41 dscp (100010)
  af42     Match packets with AF42 dscp (100100)
  af43     Match packets with AF43 dscp (100110)
  cs1      Match packets with CS1(precedence 1) dscp (001000)
  cs2      Match packets with CS2(precedence 2) dscp (010000)
  cs3      Match packets with CS3(precedence 3) dscp (011000)
  cs4      Match packets with CS4(precedence 4) dscp (100000)
  cs5      Match packets with CS5(precedence 5) dscp (101000)
  cs6      Match packets with CS6(precedence 6) dscp (110000)
  cs7      Match packets with CS7(precedence 7) dscp (111000)
  default  Match packets with default dscp (000000)
  ef       Match packets with EF dscp (101110)
```

DF (Default Forwarding):

32	16	8	4	2	1		
0	0	0	0	0	0		

- **DF** is used for best-effort traffic.
- The DSCP marking for DF is 0.

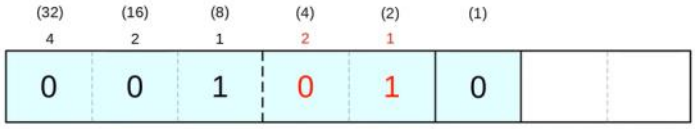
EF (Expedited Forwarding):

32	16	8	4	2	1		
1	0	1	1	1	0		

- **EF** is used for traffic that requires low loss/latency/jitter.
- The DSCP marking for EF is 46.

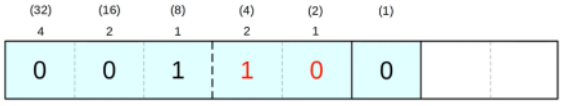
AF

- **AF** (Assured Forwarding) defines four traffic classes. All packets in a class have the same priority.
- Within each class, there are three levels of *drop precedence*.
  → Higher drop precedence = more likely to drop the packet during congestion



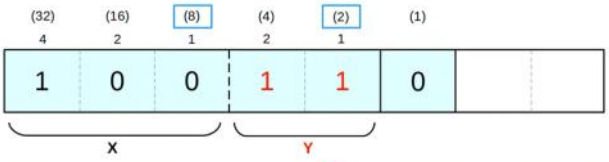
= AF11

(DSCP 10)



= AF12

(DSCP 12)



= AF43

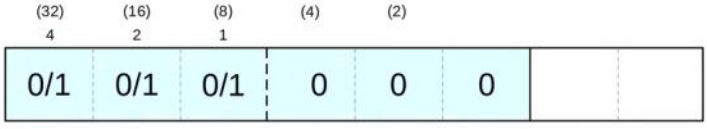
(DSCP 38)

Formula to convert from AF value to decimal DSCP value: 8X + 2Y

	Lowest drop precedence		Highest drop precedence
Highest priority	AF41 (34)	AF42 (36)	AF43 (38)
	AF31 (26)	AF32 (28)	AF33 (30)
	AF21 (18)	AF22 (20)	AF23 (22)
Lowest priority	AF11 (10)	AF12 (12)	AF13 (14)

CS

- **CS** (Class Selector) defines eight DSCP values for backward compatibility with IPP.
- The three bits that were added for DSCP are set to 0, and the original IPP bits are used to make 8 values.



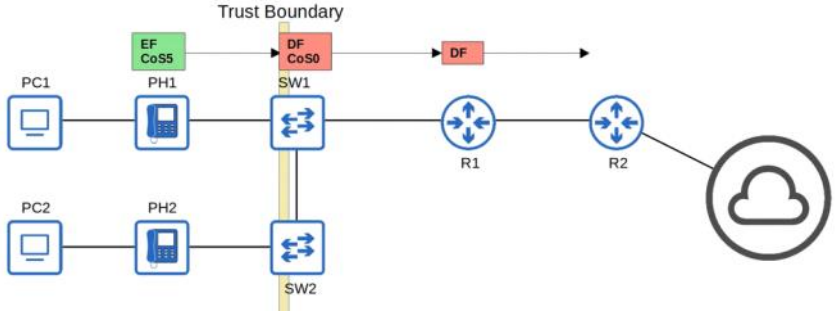
IPP:	0	1	2	3	4	5	6	7
CS:	CS0	CS1	CS2	CS3	CS4	CS5	CS6	CS7
DSCP: (decimal)	0	8	16	24	32	40	48	56

RFC 4954

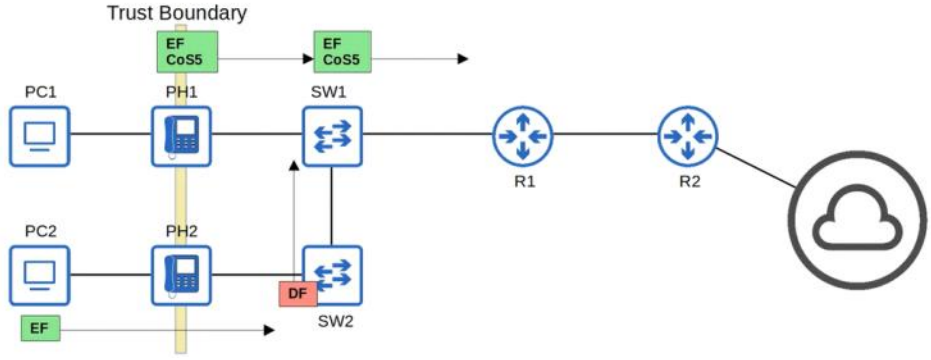
- RFC 4954 was developed with the help of Cisco to bring all of these values together and standardize their use.
- The RFC offers many specific recommendations, but here are a few key ones:
  - → Voice traffic: **EF**
  - → Interactive video: **AF4x**
  - → Streaming video: **AF3x**
  - → High priority data: **AF2x**
  - → Best effort: **DF**

Trust Boundaries

- The *trust boundary* of a network defines where devices trust/don't trust the QoS markings of received messages.
- If the markings are trusted, the device will forward the message without changing the markings.
- If the markings aren't trusted, the device will change the markings according to the configured policy.

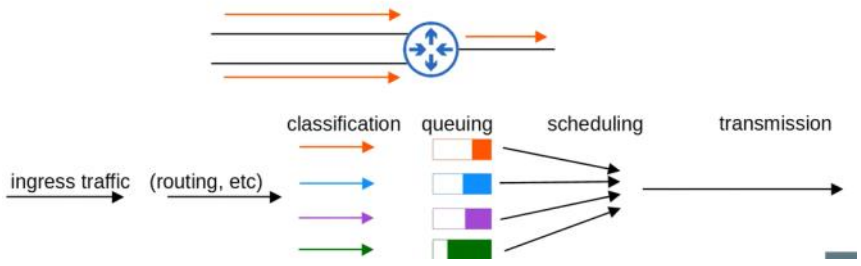


- If an IP phone is connected to the switch port, it is recommended to move the trust boundary to the IP phones.
- This is done via configuration on the switch port connected to the IP phone.
- If a user marks their PC's traffic with a high priority, the marking will be changed (not trusted)



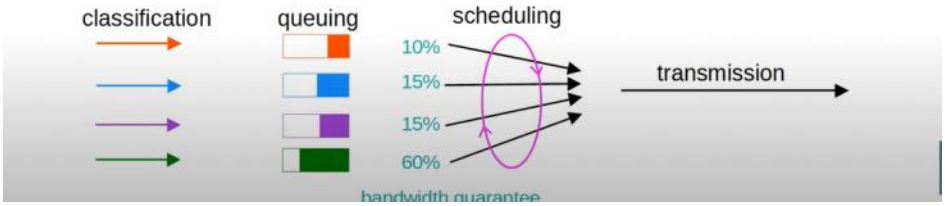
Queuing/Congestion Management

- An essential part of QoS is the use of multiple queues.
  → This is where classification plays a role. The device can match traffic based on various factors (for example the DSCP marking in the IP header) and then place it in the appropriate queue.
- However, the device is only able to forward one frame out of an interface at once, so a *scheduler* is used to decide which queue traffic is forwarded from next.
  → *Prioritization* allows the scheduler to give certain queues more priority than others.



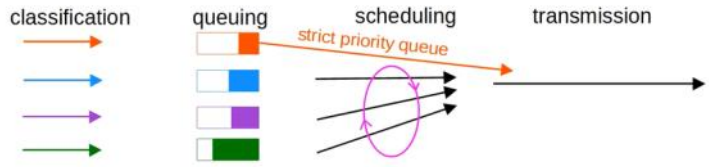
Queuing/Congestion Management

- A common scheduling method is *weighted round-robin*.
  → **round-robin** = packets are taken from each queue in order, cyclically
  → **weighted** = more data is taken from high priority queues each time the scheduler reaches that queue
- **CBWFQ** (Class-Based Weighted Fair Queuing) is a popular method of scheduling, using a weighted r robin scheduler while guaranteeing each queue a certain percentage of the interface's bandwidth duri congestion.
- Round-robin scheduling is not ideal for voice/video traffic. Even if the voice/video traffic receives a guaranteed minimum amount of bandwidth, round-robin can add delay and jitter because even the hig priority queues have to wait their turn in the scheduler.



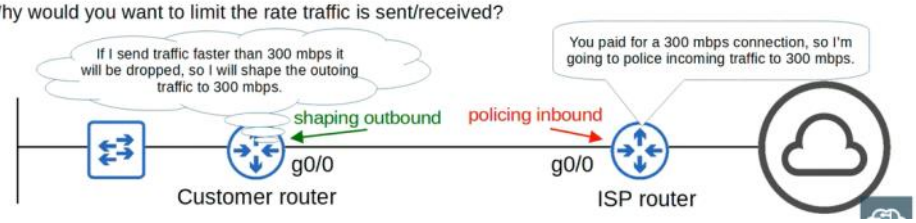
Queuing/Congestion Management

- **LLQ** (Low Latency Queuing) designates one (or more) queues as *strict priority queues*.
  → This means that if there is traffic in the queue, the scheduler will always take the next packet from that queue until it is empty.
- This is very effective for reducing the delay and jitter of voice/video traffic.
- However, it has the downside of potentially starving other queues if there is always traffic in the designated strict priority queue.
  → *Policing* (next slide) can control the amount of traffic allowed in the strict priority queue so that it can't take all of the link's bandwidth.

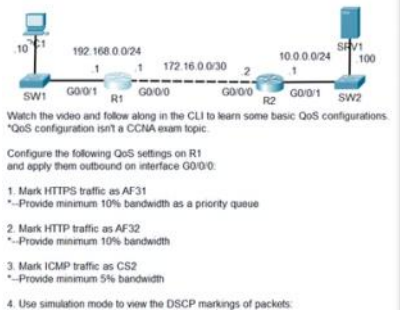


Shaping and Policing

- Traffic **shaping** and **policing** are both used to control the rate of traffic.
- **Shaping** buffers traffic in a queue if the traffic rate goes over the configured rate.
- **Policing** drops traffic if the traffic rate goes over the configured rate.
  *policing also has the option of re-marking the traffic instead of dropping it.
  → 'Burst' traffic over the configured rate is allowed for a short period of time.
  → This accommodates data applications which typically are 'bursty' in nature. Instead of a constant stream of data, they send data in bursts.
  → The amount of burst traffic allowed is configurable.
- In both cases, classification can be used to allow for different rates for different kinds of traffic.
- Why would you want to limit the rate traffic is sent/received?



CLASS-MAPS(for giving priority to traffic(marking)):




```
R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z
R1(config)#class
R1(config)#class-map HTTPS_MAP
R1(config-cmap)#match prot
R1(config-cmap)#match protocol https
R1(config-cmap)#exit
R1(config)#class
R1(config)#class-map HTTP_MAP
R1(config-cmap)#match proto
R1(config-cmap)#match protocol http
R1(config-cmap)#exit
R1(config)#class
R1(config)#class-map ICMP_MAP
R1(config-cmap)#match prot
R1(config-cmap)#match protocol icmp
R1(config-cmap)#exit
R1(config)#
```

By default if we specify only the name of the classmap it will default to match-all
we can give match-any as well and it stype as well

POLICY MAP:
R1(config)# policy-map G0/0/0_OUT
R1(config-pmap)#class HTTPS_MAP(class-map we created before)
R1(config-pmap-c)# set ip dscp AF31
R1(config-pmap-c)# priority percent 10

R1(config-pmap)#class HTTP_MAP(class-map we created before)
R1(config-pmap-c)# set ip dscp AF32
R1(config-pmap-c)# bandwidth percent 10

R1(config)# do show run | section policy-map

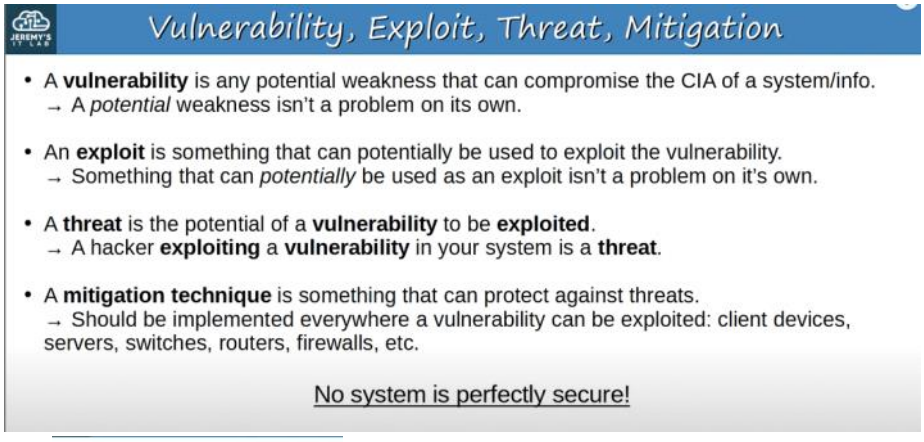
SERVICE POLICY(apply policy to interface):
R1(config)# int g0/0/0
R1(config-if)# service-policy output G0/0/0_OUT(policyname)

The principles of the **CIA Triad** form the foundation of security:

- **C**onfidentiality
  → Only authorized users should be able to access data.
  → Some information/data is public and can be accessed by anyone, some is secret and should only be accessed by specific people.

- **I**ntegrity
  → Data should not be tampered with (modified) by unauthorized users.
  → Data should be correct and authentic.

- **A**vailability
  → The network/systems should be operational and accessible to authorized users.

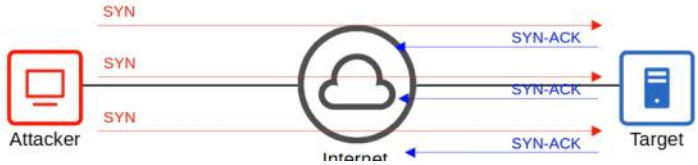


- DoS (denial-of-service) attack
- Spoofing attacks
- Reflection/amplification attack
- Man-in-the-middle attacks
- Reconnaissance attacks
- Malware
- Social engineering attacks
- Password-related attacks

Denial-of-service attacks

- DoS attacks threaten the availability of a system.

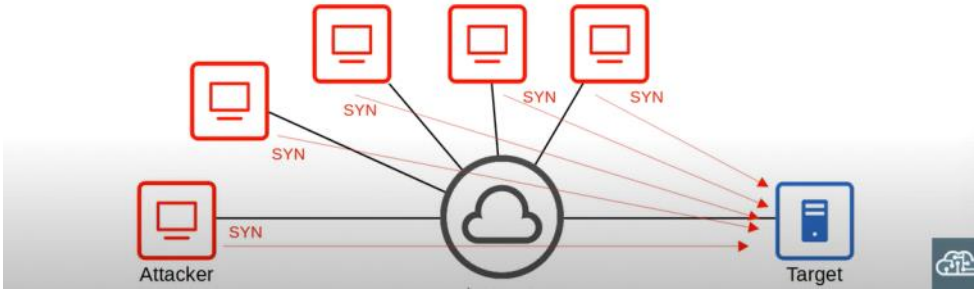
- One common DoS attack is the TCP SYN flood.
  → TCP three-way handshake: SYN | SYN-ACK | ACK
  → The attacker sends countless TCP SYN messages to the target.
  → The target sends a SYN-ACK message in response to each SYN it receives.
  → The attacker never replies with the final ACK of the TCP three-way handshake.
  → The incomplete connections fill up the target's TCP connection table.
  → The attacker continues sending SYN messages.
  → The target is no longer able to make legitimate TCP connections.



Denial-of-service attacks

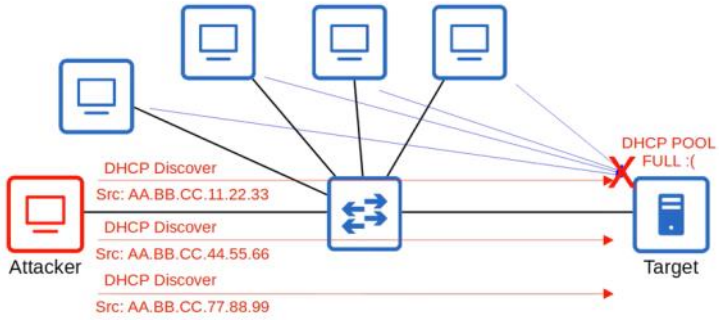
- In a DDoS (Distributed Denial-Of-Service) attack, the attacker infects many target computers with malware and uses them all to initiate a denial-of-service attack, for example a TCP SYN flood attack.

- This group of infected computers is called a **botnet**.



Spoofing attacks

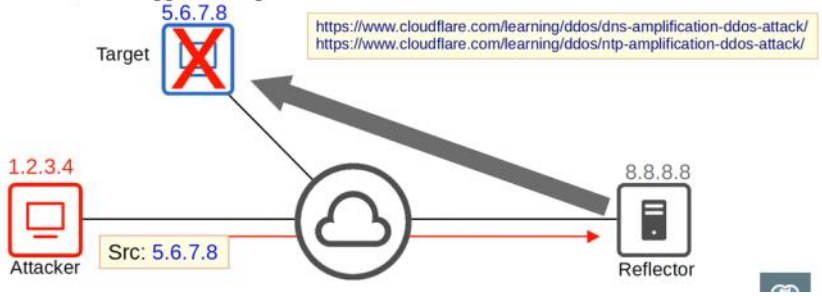
- To **spoof** an address is to use a fake source address (IP or MAC address).
- Numerous attacks involve spoofing, it's not a single kind of attack.
- An example is a **DHCP exhaustion** attack.
- An attacker uses spoofed MAC addresses to flood DHCP Discover messages.
- The target server's DHCP pool becomes full, resulting in a denial-of-service to other devices.



Reflection/Amplification attacks

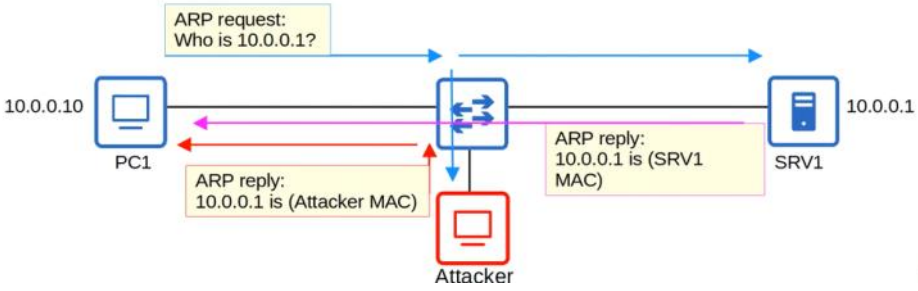
- In a **reflection** attack, the attacker sends traffic to a reflector, and spoofs the source address of its packets using the target's IP address.
- The reflector (ie. a DNS server) sends the reply to the target's IP address.
- If the amount of traffic sent to the target is large enough, this can result in a denial-of-service.

- A reflection attack becomes an **amplification** attack when the amount of traffic sent by the attacker is small, but it triggers a large amount of traffic to be sent from the reflector to the target.



Man-in-the-middle attacks

- In a **man-in-the-middle** attack, the attacker places himself between the source and destination to eavesdrop on communications, or to modify traffic before it reaches the destination.
- A common example is **ARP spoofing**, also known as **ARP poisoning**.
- A host sends an ARP request, asking for the MAC address of another device.
- The target of the request sends an ARP reply, informing the requester of its MAC address.
- The attacker waits and sends another ARP reply after the legitimate replier.
- If the attacker's ARP reply arrives last, it will overwrite the legitimate ARP entry in PC1's ARP table.



- In PC1's ARP table, the entry for 10.0.0.1 will have the attacker's MAC address.
- When PC1 tries to send traffic to SRV1, it will be forwarded to the attacker instead.
- The attacker can inspect the messages, and then forward them on to SRV1.
- The attacker can also modify the messages before forwarding them to SRV1.

Reconnaissance attacks

- Reconnaissance attacks aren't attacks themselves, but they are used to gather information about a target which can be used for a future attack.
- This is often publicly available information.
- ie. **nslookup** to learn the IP address of a site:

```
C:\Users\user>nslookup jeremysitlab.com
Server:  UnKnown
Address:  192.168.0.1

Non-authoritative answer:
Name:    jeremysitlab.com
Address:  162.241.216.233
```

- Or a WHOIS query to learn email addresses, phone numbers, physical addresses, etc.

https://lookup.icann.org/lookup

Malware

- Malware (malicious software) refers to a variety of harmful programs that can infect a computer.
- **Viruses** infect other software (a 'host program'). The virus spreads as the software is shared by users. Typically they corrupt or modify files on the target computer.
- **Worms** do not require a host program. They are standalone malware and they are able to spread on their own, without user interaction. The spread of worms can congest the network, but the 'payload' of a worm can cause additional harm to target devices.
- **Trojan Horses** are harmful software that is disguised as legitimate software. They are spread through user interaction such as opening email attachments, or downloading a file from the Internet.

The above malware types can exploit various vulnerabilities to threaten any of the CIA of the target device.
*there are many types of malware!

Social Engineering attacks

- Social engineering attacks target the most vulnerable part of any system – people!
- They involve psychological manipulation to make the target reveal confidential information or perform some action.
- **Phishing** typically involves fraudulent emails that appear to come from a legitimate business (Amazon, bank, credit card company, etc) and contain links to a fraudulent website that seems legitimate.
  Users are told to login to the fraudulent website, providing their login credentials to the attacker.
  → **spear phishing** is a more targeted form of phishing, ie. aimed at employees of a certain company.
  → **whaling** is phishing targeted at high-profile individuals, ie. a company president.
- **Vishing** (voice phishing) is phishing performed over the phone.
  → 'Hi, this is Jeremy from the IT department. Due to company policy we need to reset your password, could you tell me the password you're currently using and I'll reset it for you?'
- **Smishing** (SMS phishing) is phishing using SMS text messages.
- **Watering hole** attacks compromise sites that the target victim frequently visits. If a malicious link is placed on a website the target trusts, they might not hesitate to click it.
- **Tailgating** attacks involve entering restricted, secured areas by simply walking in behind an authorized person as they enter.

Password-related attacks

- Most systems use a username/password combination to authenticate users.
- The username is often simple/easy to guess (for example the user's email address), and the strength and secrecy of the password is relied on to provide the necessary security.
- Attackers can learn a user's passwords via multiple methods:
  → Guessing
  → **Dictionary attack**: A program runs through a 'dictionary' or list of common words/passwords to find the target's password.
  → **Brute force attack**: A program tries every possible combination of letters, numbers, and special characters to find the target's password.
- Strong passwords should contain:
  → at LEAST 8 characters (preferably more).
  → a mixture of UPPERCASE and lowercase letters.
  → a mixture of letters and numbers.
  → one or more special characters (# @ ! ? etc.)
  + should be changed regularly

Multi-factor authentication

- **Multi-factor authentication** involves providing more than just a username/password to prove your identity.
- It usually involves providing two of the following (=two-factor authentication):

- **Something you know**
  → a username/password combination, a PIN, etc.

- **Something you have**
  → pressing a notification that appears on your phone, a badge that is scanned, etc.

- **Something you are**
  → biometrics such as a face scan, palm scan, fingerprint scan, retina scan, etc.

- Requiring multiple factors of authentication greatly increases the security. Even if an attacker learns the target's password (**something you know**), they won't be able to login to the target's account.

Digital certificates

Digital certificates are another form of authentication used to prove the identity of the holder of the certificate.

They are used for websites to verify that the website being accessed is legitimate.

Entities that want a certificate to prove their identity send a CSR (Certificate Signing Request) to a CA (Certificate Authority), which will generate and sign the certificate.

Controlling and monitoring users with AAA

- **AAA** (triple-A) stands for **A**uthentication, **A**uthorization, and **A**ccounting.
- It is a framework for controlling and monitor users of a computer system (ie. a network).

- **Authentication** is the process of verifying a user's identity.
  → logging in = authentication

- **Authorization** is the process of granting the user the appropriate access and permissions.
  → granting the user access to some files/services, restricting access to other files/services = authorization

- **Accounting** is the process of recording the user's activities on the system.
  → logging when a user makes a change to a file = accounting

- Enterprises typically use a AAA server to provide AAA services.
  → ISE (Identity Services Engine) is Cisco's AAA server.

- AAA servers usually support the following two AAA protocols:
  → RADIUS: an open standard protocol. Uses UDP ports 1812 and 1813.
  → TACACS+: A Cisco propriety protocol. Uses TCP port 49.

Security Program Elements

- **User awareness** programs are designed to make employees aware of potential security threats and risks.
  → For example, a company might send out false phishing emails to make employees click a link and sign in with their login credentials.
  → Although the emails are harmless, employees who fall for the false emails will be informed that it is part of a user awareness program and they should be more careful about phishing emails.

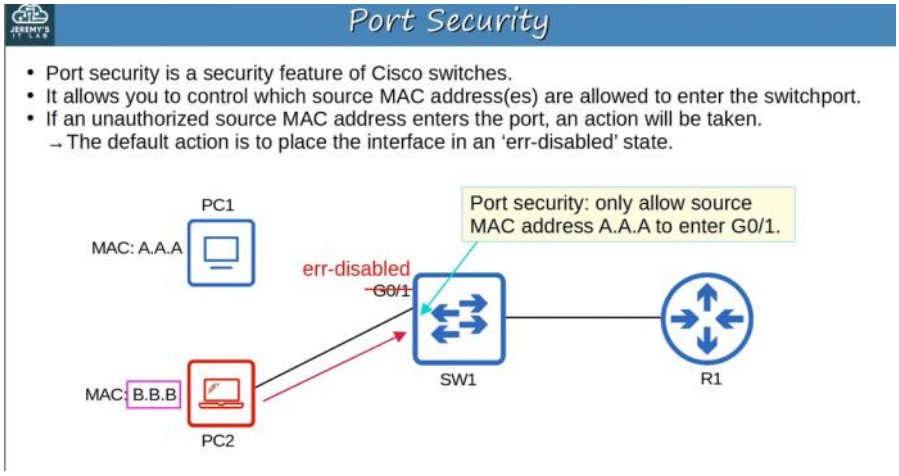
- **User training** programs are more formal than user awareness programs.
  → For example, dedicated training sessions which educate users on the corporate security policies, how to create strong passwords, and how to avoid potential threats.

- **Physical access control** protects equipment and data from potential attackers by only allowing authorized users into protected areas such as network closets or data center floors.
  → Multifactor locks can protect access to restricted areas.
  → ie. a door that requires users to swipe a badge and scan their fingerprint to enter.
  → Permissions of the badge can easily be changed, for example permissions can be removed when an employee leaves the company.

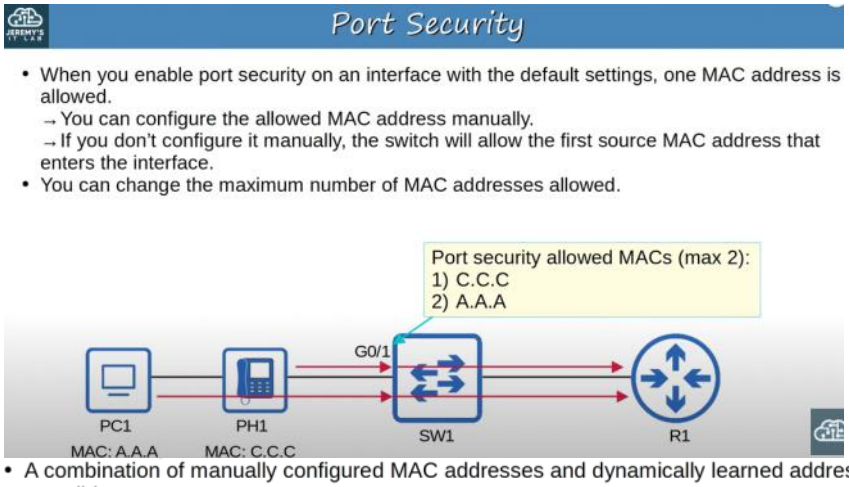
R1# show ip dhcp pool(to see available addressses in the pool)
R1# show ip dhcp binding (to see the binding of ip addresses to clients)
We can use yersinia tool in kali linux to laucnh attacks on protocols (dhcp,tcp etc..)

CONCEPTS 10

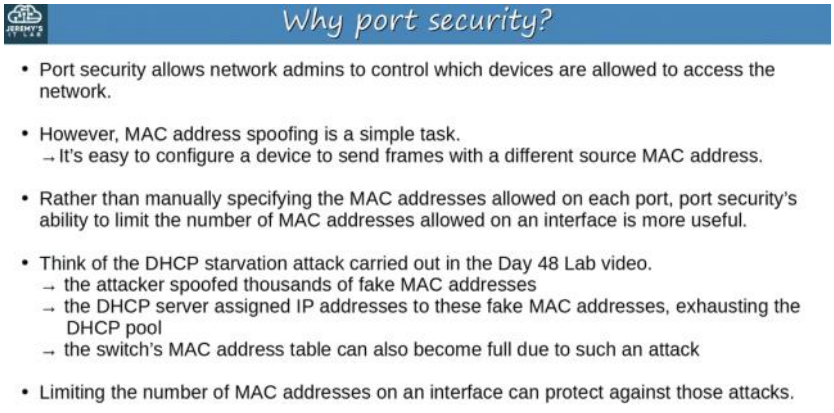
Monday, February 20, 2023 10:20 AM

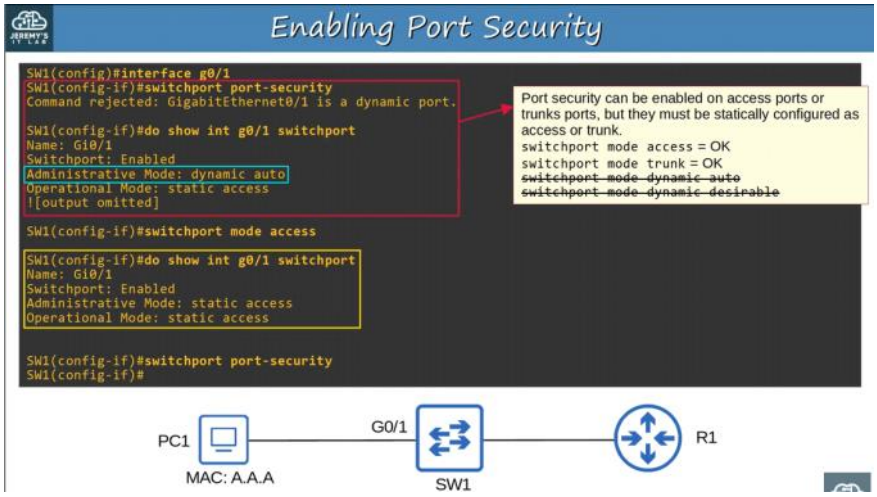


After it goes to err-disabled state then even if pc1 sends packet it cant go to that since it is err-disables



We can manually configure the learning of mac addresses to be allowed or dynamically learned

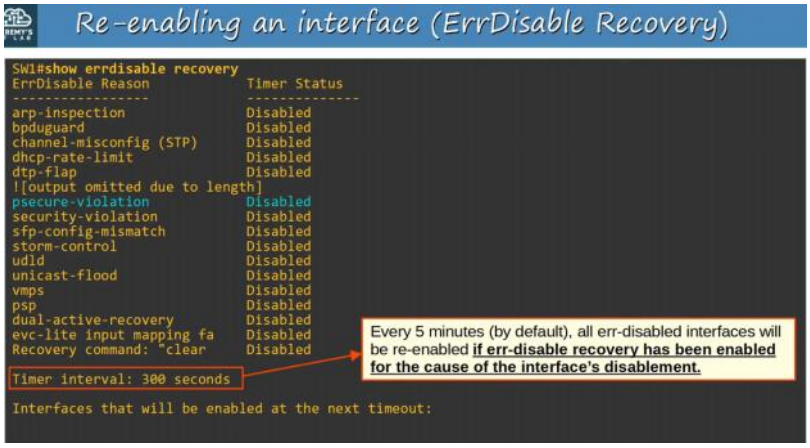


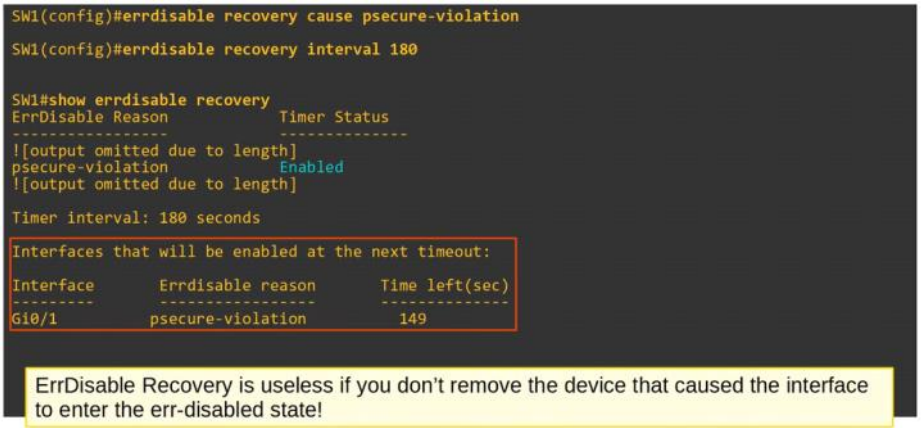


SW1# show port-security interface g0/1
(status of port security max,total mac addresses,violation counts etc..)

If one mac address is learned and after that if another pc is plugged in the same inteface then it becomes err-disabled state . So we need to unplug the device and manually enablethe interface

SW1(config)# int g0/1
SW1(config-if)# shutdown
SW1(config-if)#no shutdown





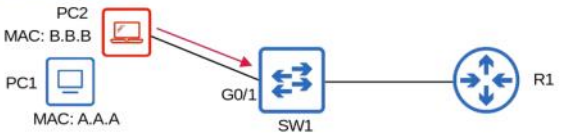
Violation Modes

There are three different violation modes that determine what the switch will do if an unauthorized frame enters an interface configured with port security.

- **Shutdown**
  - → Effectively shuts down the port by placing it in an err-disabled state.
  - → Generates a Syslog and/or SNMP message when the interface is disabled.
  - → The violation counter is set to 1 when the interface is disabled.
- **Restrict**
  - → The switch discards traffic from unauthorized MAC addresses.
  - → The interface is NOT disabled.
  - → Generates a Syslog and/or SNMP message each time an unauthorized MAC is detected.
  - → The violation counter is incremented by 1 for each unauthorized frame.
- **Protect**
  - → The switch discards traffic from unauthorized MAC addresses.
  - → The interface is NOT disabled.
  - → It does NOT generate Syslog/SNMP messages for unauthorized traffic.
  - → It does NOT increment the violation counter.

Violation mode: Restrict

```
SW1(config-if)#switchport port-security
SW1(config-if)#switchport port-security mac-address 000a.000a.000a
SW1(config-if)#switchport port-security violation restrict

*May 23 22:54:09.951: %PORT_SECURITY-2-PSECURE_VIOLATION: Security violation occurred, caused by MAC
address 000b.000b.000b on port GigabitEthernet0/1.

SW1#show port-security interface g0/1
Port Security              : Enabled
Port Status                : Secure-up
Violation Mode             : Restrict
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 1
Total MAC Addresses        : 1
Configured MAC Addresses   : 1
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 000b.000b.000b:1
Security Violation Count   : 12
```



Violation mode: Protect

```
SW1(config-if)#switchport port-security
SW1(config-if)#switchport port-security mac-address 000a.000a.000a
SW1(config-if)#switchport port-security violation protect


SW1#show port-security interface g0/1
Port Security              : Enabled
Port Status                : Secure-up
Violation Mode             : Protect
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 1
Total MAC Addresses        : 1
Configured MAC Addresses   : 1
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 000b.000b.000b:1
Security Violation Count   : 0
```

Secure MAC address aging

```
SW1#show port-security interface g0/1
Port Security              : Enabled
Port Status                : Secure-up
Violation Mode             : Shutdown
Aging Time                 : 0 mins
Aging Type                 : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses      : 1
Total MAC Addresses        : 1
Configured MAC Addresses   : 0
Sticky MAC Addresses       : 0
Last Source Address:Vlan   : 000a.000a.000a:1
Security Violation Count   : 0
```

- By default secure MAC addresses will not 'age out' (`Aging Time : 0 mins`)
  - → Can be configured with **`switchport port-security aging time`** *minutes*
- The default aging type is **Absolute**
  - → **Absolute**: After the secure MAC address is learned, the aging timer starts and the MAC is removed after the timer expires, even if the switch continues receiving frames from that source MAC address.
  - → **Inactivity**: After the secure MAC address is learned, the aging timer starts but is reset every time a frame from that source MAC address is received on the interface.
  - → Aging type is configured with **`switchport port-security aging type {absolute | inactivity}`**
- Secure Static MAC aging (addresses configured with **`switchport port-security mac-address`** *x.x.x*) is disabled by default.

SW1# show port-security
(shows which ports have port security enabled and other details of them)

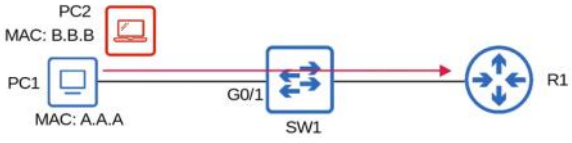
→ Can be enabled with **`switchport port-security aging static`**

Sticky Secure MAC Addresses

- 'Sticky' secure MAC address learning can be enabled with the following command:
  `SW1(config-if)# switchport port-security mac-address sticky`
- When enabled, dynamically-learned secure MAC addresses will be added to the running config like this:
  `switchport port-security mac-address sticky` *`mac-address`*
- The 'sticky' secure MAC addresses will never age out.
  → You need to save the running-config to the startup-config to make them truly permanent (or else they will not be kept if the switch restarts)
- When you issue the **`switchport port-security mac-address sticky`** command, all current dynamically-learned secure MAC addresses will be converted to sticky secure MAC addresses.
- If you issue the **`no switchport port-security mac-address sticky`** command, all current sticky secure MAC addresses will be converted to regular dynamically-learned secure MAC addresses.

Sticky Secure MAC Addresses

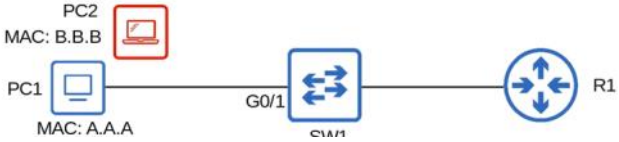
```
SW1(config-if)#switchport port-security
SW1(config-if)#switchport port-security mac-address sticky
SW1(config-if)#do show running-config interface g0/1
!
interface GigabitEthernet0/1
 switchport mode access
 switchport port-security mac-address sticky
 switchport port-security mac-address sticky 000a.000a.000a
 switchport port-security
 negotiation auto
```



MAC Address Table

- Secure MAC addresses will be added to the MAC address table like any other MAC address
  → Sticky and Static secure MAC addresses will have a type of STATIC
  → Dynamically-learned secure MAC addresses will have a type of DYNAMIC
  → You can view all secure MAC addresess with **show mac address-table secure**

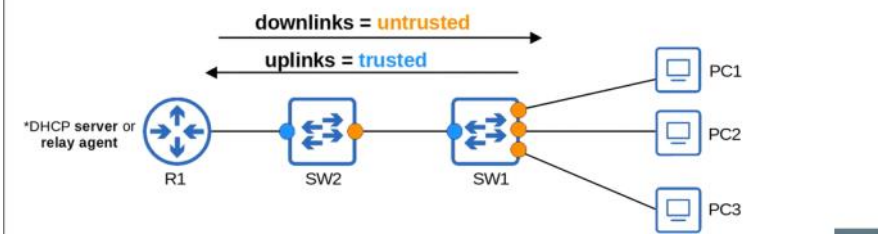
```
SW1#show mac address-table secure
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
   1    000a.000a.000a    STATIC      Gi0/1
Total Mac Addresses for this criterion: 1
```



SW2(config-if)# switchport port-security maximum 4(max 4 mac)

DHCP Snooping

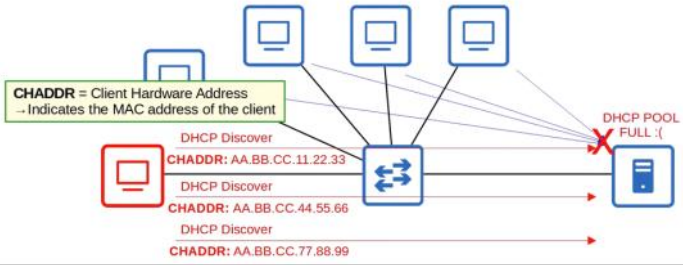
- DHCP snooping is a security feature of switches that is used to filter DHCP messages received on *untrusted* ports.
- DHCP snooping only filters DHCP messages. Non-DHCP messages aren't affected.
- All ports are *untrusted* by default.
  → Usually, **uplink** ports are configured as *trusted* ports, and **downlink** ports remain *untrusted*.



Here if pc1 is t4rustewd source and it initiates the dhcp discover message then the switch willl inspect it and fprwards to thre router but if pc2 is untrusted source then the switch will discard the packet while entering the switch itself

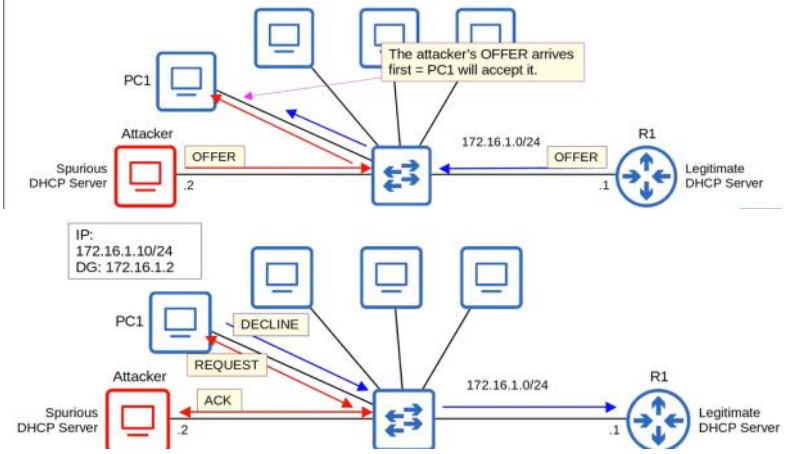
DHCP Starvation

- An example of a DHCP-based attack is a **DHCP starvation** attack.
- An attacker uses spoofed MAC addresses to flood DHCP Discover messages.
- The target server's DHCP pool becomes full, resulting in a denial-of-service to other devices.



DHCP Poisoning (Man-in-the-Middle)

- Similar to ARP Poisoning, DHCP Poisoning can be used to perform a Man-in-the-Middle attack.
- A spurious DHCP server replies to clients' DHCP Discover messages and assigns them IP addresses, but makes the client use the spurious server's IP as the default gateway.
  *Clients usually accept the first OFFER message they receive.
- This will cause the client to send traffic to the attacker instead of the legitimate default gateway.
- The attacker can then examine/modify the traffic before forwarding it to the legitimate default gateway.



DHCP Messages

- When DHCP Snooping filters messages, it differentiates between **DHCP Server** messages and **DHCP Client** messages

- Messages sent by **DHCP Servers**:
  → OFFER
  → ACK
  → NAK = Opposite of ACK, used to decline a client's REQUEST

- Messages sent by **DHCP Clients:**
  → DISCOVER
  → REQUEST
  → RELEASE = Used to tell the server that the client no longer needs its IP address
  → DECLINE = Used to decline the IP address offered by a DHCP server

DHCP Snooping Operations

- If a DHCP message is received on a **trusted port**, forward it as normal without inspection.
- If a DHCP message is received on an **untrusted port**, inspect it and act as follows:
  → If it is a **DHCP Server** message, discard it.

  → If it is a **DHCP Client** message, perform the following checks:

  DISCOVER/REQUEST messages: Check if the frame's source MAC address and the DHCP message's CHADDR fields match. Match = forward, mismatch = discard

  RELEASE/DECLINE messages: Check if the packet's source IP address and the receiving interface match the entry in the *DHCP Snooping Binding Table*. Match = forward, mismatch = discard

- When a client successfully leases an IP address from a server, create a new entry in the *DHCP Snooping Binding Table*.

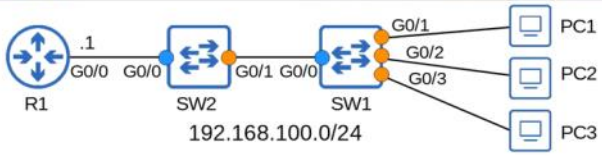
DHCP Snooping

```
SW2(config)#ip dhcp snooping
SW2(config)#ip dhcp snooping vlan 1
SW2(config)#no ip dhcp snooping information option
SW2(config)#interface g0/0
SW2(config-if)#ip dhcp snooping trust
```

I will explain this later!

```
SW1(config)#ip dhcp snooping
SW1(config)#ip dhcp snooping vlan 1
SW1(config)#no ip dhcp snooping information option
SW1(config)#interface g0/0
SW1(config-if)#ip dhcp snooping trust

SW1#show ip dhcp snooping binding
MacAddress          IpAddress        Lease(sec)  Type           VLAN  Interface
------------------  ---------------  ----------  -------------  ----  --------------------
0C:29:2F:18:79:00   192.168.100.10   86294       dhcp-snooping  1     GigabitEthernet0/3
0C:29:2F:90:91:00   192.168.100.11   86302       dhcp-snooping  1     GigabitEthernet0/1
0C:29:2F:67:E9:00   192.168.100.12   86314       dhcp-snooping  1     GigabitEthernet0/2
Total number of bindings: 3
```

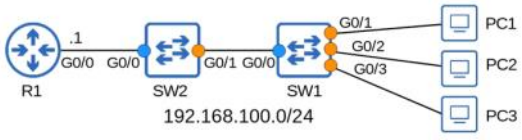
RELEASE/DECLINE messages will be checked to make sure their IP address/interface ID match the entry in the DHCP snooping table.



DHCP Snooping Rate-Limiting

- DHCP snooping can limit the rate at which DHCP messages are allowed to enter an interface.
- If the rate of DHCP messages crosses the configured limit, the interace is err-disabled.
- Like with Port Security, the interface can be manually re-enabled, or automatically re-enabled with errdisable recovery.

```
SW1(config)#interface range g0/1 - 3
SW1(config-if-range)#ip dhcp snooping limit rate 1

*Jun  5 13:15:14.180: %DHCP_SNOOPING-4-DHCP_SNOOPING_ERRDISABLE_WARNING: DHCP Snooping received 1 DHCP packets on
interface Gi0/1
*Jun  5 13:15:14.181: %DHCP_SNOOPING-4-DHCP_SNOOPING_RATE_LIMIT_EXCEEDED: The interface Gi0/1 is receiving more
than the threshold set.
*Jun  5 13:15:14.182: %PM-4-ERR_DISABLE: dhcp-rate-limit error detected on Gi0/1, putting Gi0/1 in err-disable
state
*Jun  5 13:15:15.185: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed state to down
*Jun  5 13:15:16.190: %LINK-3-UPDOWN: Interface GigabitEthernet0/1, changed state to down
```

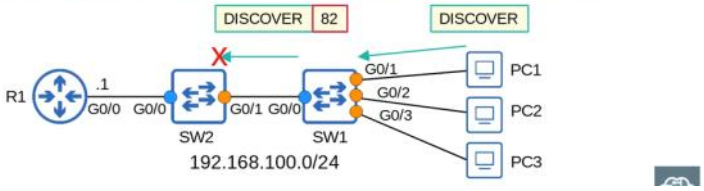


```
SW1(config)#errdisable recovery cause dhcp-rate-limit

SW1#show errdisable recovery
ErrDisable Reason            Timer Status
-----------------            --------------
arp-inspection               Disabled
bpduguard                    Disabled
channel-misconfig (STP)      Disabled
dhcp-rate-limit              Enabled
dtp-flap                     Disabled
gbic-invalid                 Disabled
inline-power                 Disabled
![output omitted due to length]


Timer interval: 300 seconds

Interfaces that will be enabled at the next timeout:

Interface       Errdisable reason       Time left(sec)
---------       -----------------       --------------
Gi0/1            dhcp-rate-limit          293
```

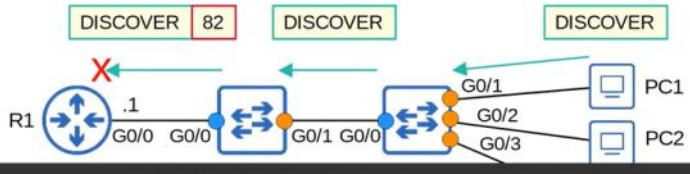
DHCP Option 82 (Information Option)

- Option 82, also known as the 'DHCP relay agent information option' is one of many DHCP options.
- It provides additional information about which DHCP relay agent received the client's message, on which interface, in which VLAN, etc.
- DHCP relay agents can add Option 82 to messages they forward to the remote DHCP server.
- With DHCP snooping enabled, by default Cisco switches will add Option 82 to DHCP messages they receive from clients, even if the switch isn't acting as a DHCP relay agent.
- By default, Cisco switches will drop DHCP messages with Option 82 that are received on an untrusted port.

```
SW2#
*Jun  6 01:36:15.298: %DHCP_SNOOPING-5-DHCP_SNOOPING_NONZERO_GIADDR: DHCP_SNOOPING drop message with non-
zero giaddr or option82 value on untrusted port, message type: DHCPDISCOVER, MAC sa: 0c29.2f67.e900
```




```
SW1(config)#no ip dhcp snooping information option
```




```
R1#
*Jun  6 01:46:46.763: DHCPD: inconsistent relay information.
*Jun  6 01:46:46.763: DHCPD: relay information option exists, but giaddr is zero.
```

In a Media Access Control (MAC) spoofing attack, an attacker uses the MAC address of another known host on the network in order to bypass port security measures. MAC spoofing can also be used to impersonate another host on the network. Implementing port security with sticky secure MAC addresses can help mitigate MAC spoofing attacks.

In a MAC flooding attack, an attacker generates thousands of forged frames every minute with the intention of overwhelming the switch's MAC address table. Once this table is flooded, the switch can no longer make intelligent forwarding decisions and all traffic is flooded. This allows the attacker to view all data sent through the switch because all traffic will be sent out each port. Implementing port security can help mitigate MAC flooding attacks by limiting the number of MAC addresses that can be learned on each interface to a maximum of 128. A MAC flooding attack is also known as a Content Addressable Memory (CAM) table overflow attack.

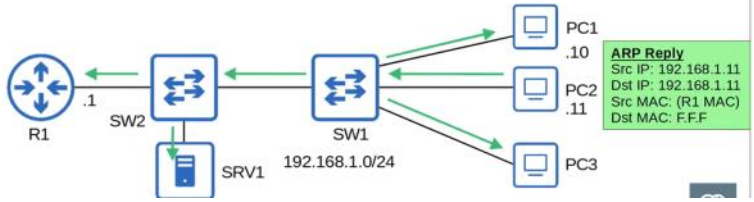
In an Address Resolution Protocol (ARP) poisoning attack, which is also known as an ARP spoofing attack, the attacker sends a gratuitous ARP (GARP) message to a host. The GARP message associates the attacker's MAC address with the Internet Protocol (IP) address of a valid host on the network. Subsequently, traffic sent to the valid host address will go through the attacker's computer rather than directly to the intended recipient. Implementing Dynamic ARP Inspection (DAI) can help mitigate ARP poisoning attacks.

In a virtual local area network (VLAN) hopping attack, an attacker attempts to inject packets into other VLANs by accessing the VLAN trunk and double-tagging 802.1Q frames. A successful VLAN hopping attack enables an attacker to send traffic to other VLANs without the use of a router. You can prevent VLAN hopping by disabling Dynamic Trunking Protocol (DTP) on trunk ports, by changing the native VLAN, and by configuring user-facing ports as access ports.

In a Dynamic Host Configuration Protocol (DHCP) spoofing attack, an attacker installs a rogue DHCP server on the network in an attempt to intercept DHCP requests. The rogue DHCP server can then respond to the DHCP requests with its own IP address as the default gateway address; hence all traffic is routed through the rogue DHCP server. You should enable DHCP snooping to help prevent DHCP spoofing attacks.

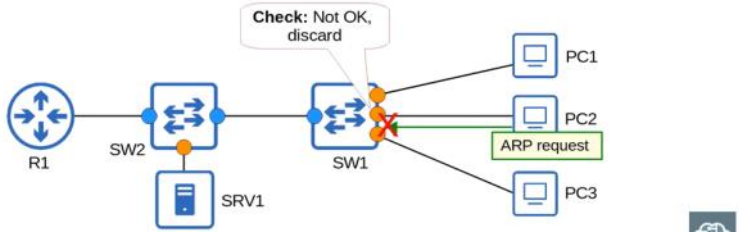
Gratuitous ARP

- A *Gratuitous ARP* message is an ARP reply that is sent without receiving an ARP request.
- It is sent to the broadcast MAC address.
- It allows other devices to learn the MAC address of the sending device without having to send ARP requests.
- Some devices automatically send GARP messages when an interface is enabled, IP address is changed, MAC address is changed, etc.



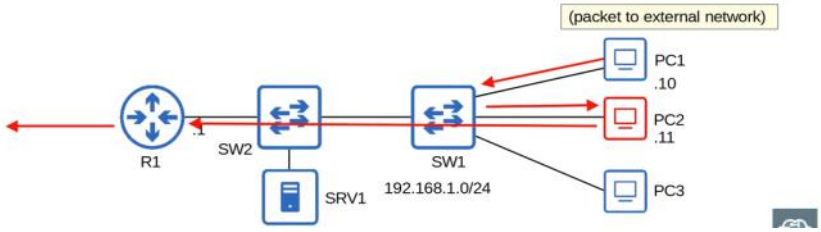
Dynamic ARP Inspection

- DAI is a security feature of switches that is used to filter ARP messages received on *untrusted* ports.
- DAI only filters ARP messages. Non-ARP messages aren't affected.
- All ports are *untrusted* by default.
  → Typically, all ports connected to other network devices (switches, routers) should be configured as **trusted**, while interfaces connected to end hosts should remain **untrusted**.



ARP Poisoning (Man-in-the-Middle)

- Similar to DHCP poisoning, ARP poisoning involves an attacker manipulating targets' ARP tables so traffic is sent to the attacker.
- To do this, the attacker can send gratuitous ARP messages using another device's IP address.
- Other devices in the network will receive the GARP and update their ARP tables, causing them to send traffic to the attacker instead of the legitimate destination.



Dynamic ARP Inspection Operations

- DAI inspects the sender MAC and sender IP fields of ARP messages received on **untrusted** ports and checks that there is a matching entry in the DHCP snooping binding table.
  → If there is a matching entry, the ARP message is forwarded normally.
  → If there isn't a matching entry, the ARP message is discarded.

```
SW1#show ip dhcp snooping binding
MacAddress          IpAddress        Lease(sec)  Type           VLAN  Interface
------------------  ---------------  ----------  -------------  ----  --------------------
0C:29:2F:18:79:00   192.168.100.10   86294       dhcp-snooping  1     GigabitEthernet0/3
0C:29:2F:90:91:00   192.168.100.11   86302       dhcp-snooping  1     GigabitEthernet0/1
0C:29:2F:67:E9:00   192.168.100.12   86314       dhcp-snooping  1     GigabitEthernet0/2
Total number of bindings: 3
```

- DAI doesn't inspect messages received on **trusted** ports. They are forwarded as normal.
- **ARP ACLs** can be manually configured to map IP addresses/MAC addresses for DAI to check.
  → Useful for hosts that don't use DHCP.
- DAI can be configured to perform more in-depth checks also, but these are optional.
- Like DHCP snooping, DAI also supports rate-limiting to prevent attackers from overwhelming the switch with ARP messages.

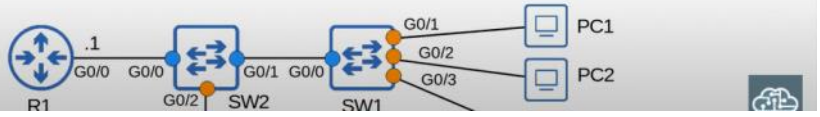
→ DHCP snooping and DAI both require work from the switch's CPU.
→ Even if the attacker's messages are blocked, they can overload the switch CPU with ARP messages.

DAI Configuration

```
SW2(config)#ip arp inspection vlan 1
SW2(config)#interface range g0/0 - 1
SW2(config-if-range)#ip arp inspection trust
```

```
SW1(config)#ip arp inspection vlan 1
SW1(config)#interface g0/0
SW1(config-if)#ip arp inspection trust
```

DHCP snooping requires two commands to enable it:
ip dhcp snooping
ip dchp snooping vlan *vlan-number*
DAI only requires one:
ip arp inspection vlan *vlan-number*




```
SW1#show ip arp inspection interfaces

 Interface        Trust State     Rate (pps)     Burst Interval
 ---------------  -----------     ----------     --------------
 Gi0/0            Trusted               None                N/A
 Gi0/1            Untrusted               15                  1
 Gi0/2            Untrusted               15                  1
 Gi0/3            Untrusted               15                  1
 Gi1/0            Untrusted               15                  1
 Gi1/1            Untrusted               15                  1
 Gi1/2            Untrusted               15                  1
 Gi1/3            Untrusted               15                  1
 Gi2/0            Untrusted               15                  1
 Gi2/1            Untrusted               15                  1
 Gi2/2            Untrusted               15                  1
 Gi2/3            Untrusted               15                  1
 Gi3/0            Untrusted               15                  1
 Gi3/1            Untrusted               15                  1
 Gi3/2            Untrusted               15                  1
 Gi3/3            Untrusted               15                  1
```

DAI rate limiting is enabled on untrusted ports by default with a rate of 15 packets per second.
It is disabled on trusted ports by default.
*DHCP snooping rate limiting is disabled on all interfaces by default.

DHCP snooping rate limiting is configured like this:
x packets per second.

The DAI **burst interval** allows you to configure rate limiting like this:
x packets per y seconds

DAI Rate Limiting

```
SW1(config)#interface range g0/1 - 2
SW1(config-if-range)#ip arp inspection limit rate 25 burst interval 2
SW1(config-if-range)#interface range g0/3
SW1(config-if)#ip arp inspection limit rate 10
SW1(config-if)#do show ip arp inspection interfaces

 Interface        Trust State     Rate (pps)     Burst Interval
 ---------------  -----------     ----------     --------------
 Gi0/0            Trusted               None                N/A
 Gi0/1            Untrusted               25                  2
 Gi0/2            Untrusted               25                  2
 Gi0/3            Untrusted               10                  1
 ![output omitted]
```

The burst interval is optional. If you don't specify it, the default is 1 second.

If ARP messages are received faster than the specified rate, the interface will be err-disabled.
It can be re-enabled in two ways:
1: **shutdown** and **no shutdown**
2: **errdisable recovery cause arp-inspection**

DAI Optional Checks

```
SW1(config)#ip arp inspection validate ?
  dst-mac  Validate destination MAC address
  ip       Validate IP addresses
  src-mac  Validate source MAC address
```

dst-mac: Enables validation of the destination MAC address in the Ethernet header against the target MAC address in the ARP body for ARP responses. The device classifies packets with different MAC addresses as invalid and drops them

ip: Enables validation of the ARP body for invalid and unexpected IP addresses. Addresses include 0.0.0.0, 255.255.255.255, and all IP multicast addresses. The device checks the sender IP addresses in all ARP requests and responses and checks the target IP addresses only in ARP responses.

src-mac: Enables validation of the source MAC address in the Ethernet header against the sender MAC address in the ARP body for ARP requests and responses. The devices classifies packets with different MAC addresses as invalid and drops them.

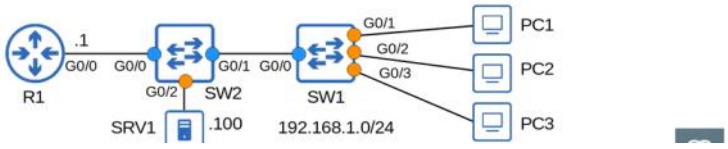
(source: https://www.cisco.com/c/m/en_us/techdoc/dc/reference/cli/n5k/commands/ip-arp-inspection-validate.html)

These checks are done in addition to the standard DAI check (sender MAC/IP).
If configured, an ARP message must pass **all** of the checks to be considered valid.

DAI Optional Checks

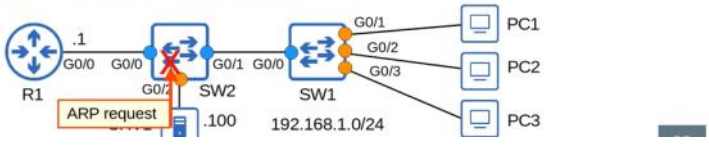
```
SW1(config)#ip arp inspection validate dst-mac
SW1(config)#ip arp inspection validate ip
SW1(config)#ip arp inspection validate src-mac

SW1(config)#do show running-config | include validate
ip arp inspection validate src-mac

SW1(config)#ip arp inspection validate ip src-mac dst-mac

SW1(config)#do show running-config | include validate
ip arp inspection validate src-mac dst-mac ip
```

You must enter all of the validation checks you want in a single command.
*You can specify one, two, or all three.
*The order isn't significant.

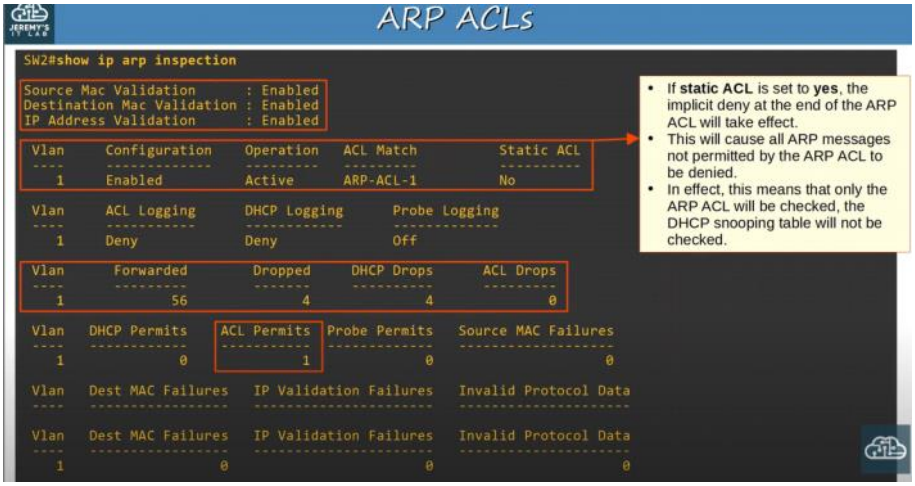


ARP ACLs

```
SW2#show ip dhcp snooping binding
MacAddress          IpAddress        Lease(sec)  Type           VLAN  Interface
------------------  ---------------  ----------  -------------  ----  --------------------
0C:29:2F:18:79:00   192.168.1.12     79226       dhcp-snooping   1     GigabitEthernet0/1
0C:29:2F:90:91:00   192.168.1.10     79188       dhcp-snooping   1     GigabitEthernet0/1
0C:29:2F:67:E9:00   192.168.1.11     79210       dhcp-snooping   1     GigabitEthernet0/1
Total number of bindings: 3

!SRV1 has a static IP address of 192.168.1.100, so it does not have an entry in SW2's DHCP
!snooping binding table.

*Jun 19 05:56:15.538: %SW_DAI-4-DHCP_SNOOPING_DENY: 1 Invalid ARPs (Req) on Gi0/2, vlan 1.
([0c29.2f1e.7700/192.168.1.100/0000.0000.0000/192.168.1.1/05:56:14 UTC Sat Jun 19 2021])

SW2(config)#arp access-list ARP-ACL-1
SW2(config-arp-nacl)#permit ip host 192.168.1.100 mac host 0c29.2f1e.7700

SW2(config)#ip arp inspection filter ARP-ACL-1 vlan 1
```



SW2# show ip arp inspection(summary of DAI inspection)

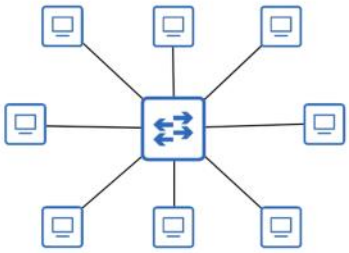

ARP ACLs
SW2#show ip arp inspection
Source Mac Validation : Enabled
Destination Mac Validation : Enabled
IP Address Validation : Enabled
Vlan Configuration Operation ACL Match Static ACL
1 Enabled Active ARP-ACL-1 No
Vlan ACL Logging DHCP Logging Probe Logging
1 Deny Deny Off
Vlan Forwarded Dropped DHCP Drops ACL Drops
1 56 4 4 0
Vlan DHCP Permits ACL Permits Probe Permits Source MAC Failures
1 0 1 0 0
Vlan Dest MAC Failures IP Validation Failures Invalid Protocol Data
Vlan Dest MAC Failures IP Validation Failures Invalid Protocol Data
1 0 0 0
If static ACL is set to yes, the implicit deny at the end of the ARP ACL will take effect.
This will cause all ARP messages not permitted by the ARP ACL to be denied.
In effect, this means that only the ARP ACL will be checked, the DHCP snooping table will not be checked.

PART 10

Monday, February 20, 2023 5:40 PM

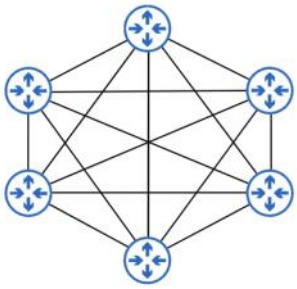
Common Terminologies – Star

- **Star**: When several devices all connect to one central device we can draw them in a 'star' shape like below, so this is often called a 'star topology'.



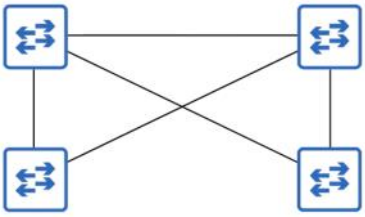
Common Terminologies – Full Mesh

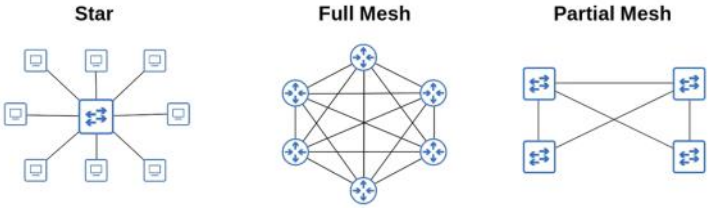
- **Full Mesh**: When each device is connected to each other device.



Common Terminologies – Partial Mesh

- **Partial Mesh**: When some devices are connected to each other, but not all.

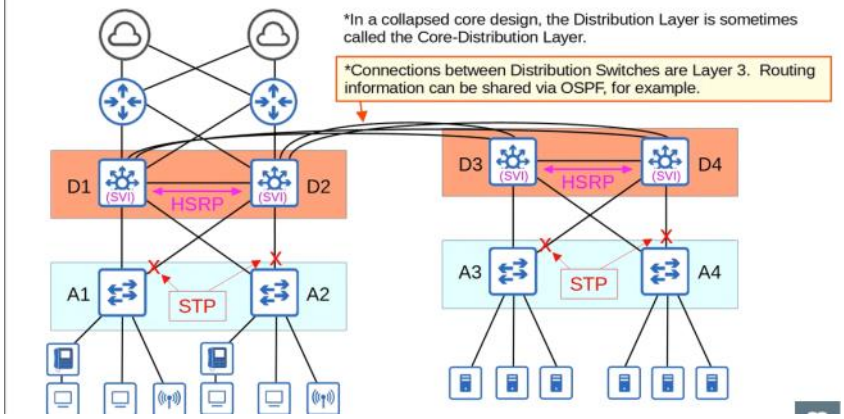




Two-Tier Campus LAN Design

- The two-tier LAN design consists of two hierarchical layers:
  - → **Access Layer**
  - → **Distribution Layer**
- Also called a 'Collapsed Core' design because it omits a layer that is found in the Three Tier design: the **Core Layer**
- **Access Layer:**
  - → the layer that end hosts connect to (PCs, printers, cameras, etc.)
  - → typically Access Layer Switches have lots of ports for end hosts to connect to
  - → QoS marking is typically done here
  - → Security services like port security, DAI, etc are typically performed here
  - → switchports might be PoE-enabled for wireless APs, IP phones, etc.
- **Distribution Layer:**
  - → aggregates connections from the Access Layer Switches
  - → typically is the border between Layer 2 and Layer 3
  - → connects to services such as Internet, WAN, etc.

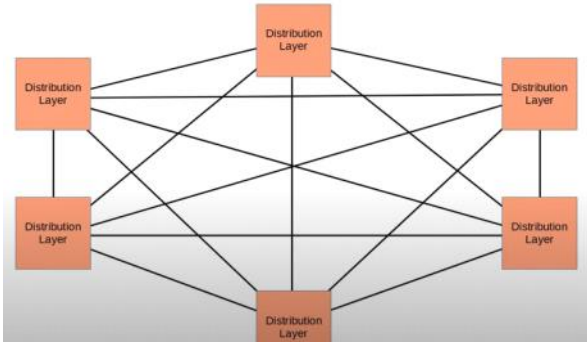
Two-Tier Campus LAN Design



The access layer diagram looks like a **star** topology
And the distribution and access layer connections look like **partial mesh**
Between distribution layer switches it forms **full mesh**

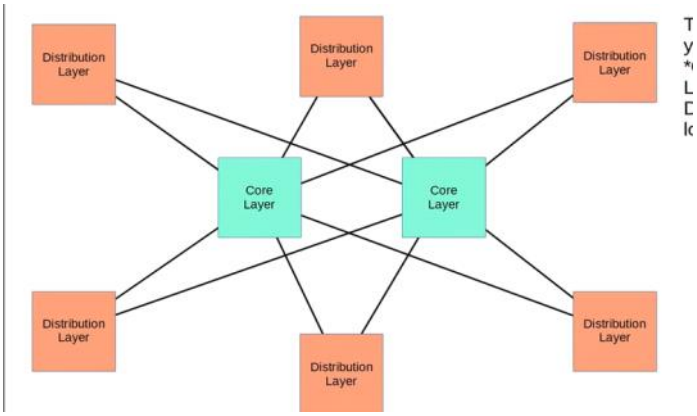
Two-Tier Campus LAN Design

In large LAN networks with many Distribution Layer switches (for example in separate buildings), the number of connections required between Distribution Layer switches grows rapidly.



To help scale large LAN networks, you can add a Core Layer.

*Cisco recommends adding a Core Layer if there are more than three Distribution Layers in a single location.

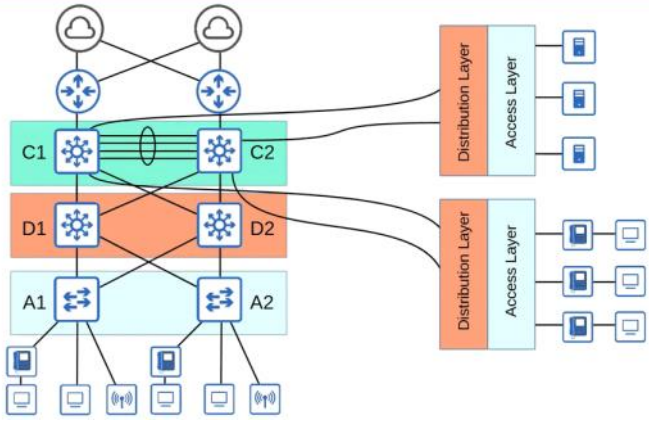


Three-Tier Campus LAN Design

- The three-tier LAN design consists of three hierarchical layers:
  - → **Access Layer**
  - → **Distribution Layer**
  - → **Core Layer**

- **Core Layer:**
  - → Connects Distribution Layers together in large LAN networks
  - → The focus is speed ('fast transport')
  - → CPU-intensive operations such as security, QoS marking/classification, etc. should be avoided at this Layer
  - → Connections are all Layer 3. No spanning-tree!
  - → Should maintain connectivity throughout the LAN even if devices fail

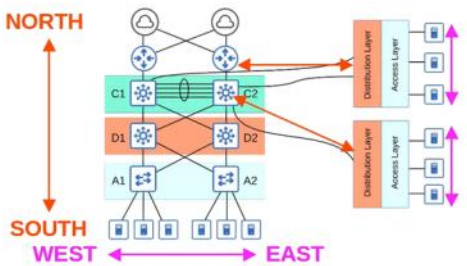
Three-Tier Campus LAN Design



Distribution layer is sometimes called as aggregation layer

Spine-Leaf Architecture

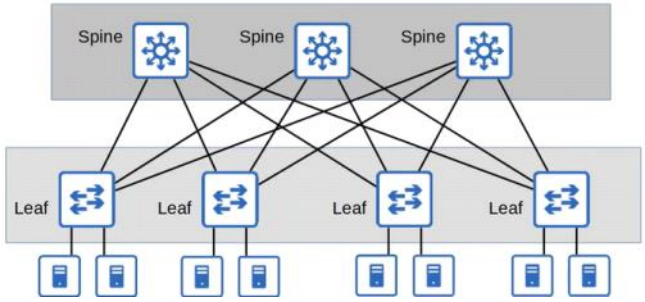
- Data centers are dedicated spaces/buildings used to store computer systems such as servers and network devices.
- Traditional data center designs used a three-tier architecture (Access-Distribution-Core) like we just covered.
- This worked well when most traffic in the data center was North-South.

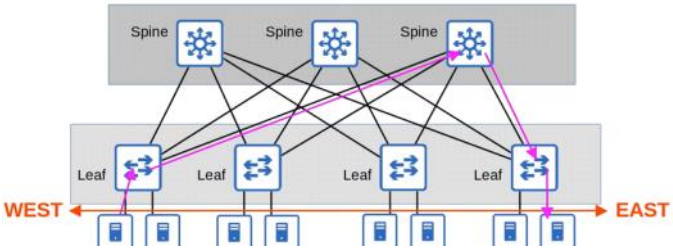


- With the precedence of virtual servers, applications are often deployed in a distributed manner (across multiple physical servers), which increases the amount of East-West traffic in the data center.
- The traditional three-tier architecture led to bottlenecks in bandwidth as well as variability in the server-to-server latency depending on the path the traffic takes.
- To solve this, Spine-Leaf architecture (also called Clos architecture) has become prominent in data centers.

Spine-Leaf Architecture

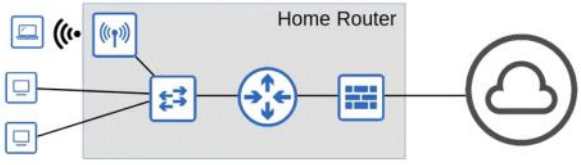
- There are some rules about Spine-Leaf architecture:
  - → Every Leaf switch is connected to every Spine switch.
  - → Every Spine switch is connected to every Leaf switch.
  - → Leaf switches do not connect to other Leaf switches.
  - → Spine switches do not connect to other Spine switches.
  - → End hosts (servers etc.) only connect to Leaf switches.
- The path taken by traffic is randomly chosen to balance the traffic load among the Spine switches.
- Each server is separated by the same number of 'hops' (except those connected to the same Leaf), providing consistent latency for East-West traffic.





SOHO Networks

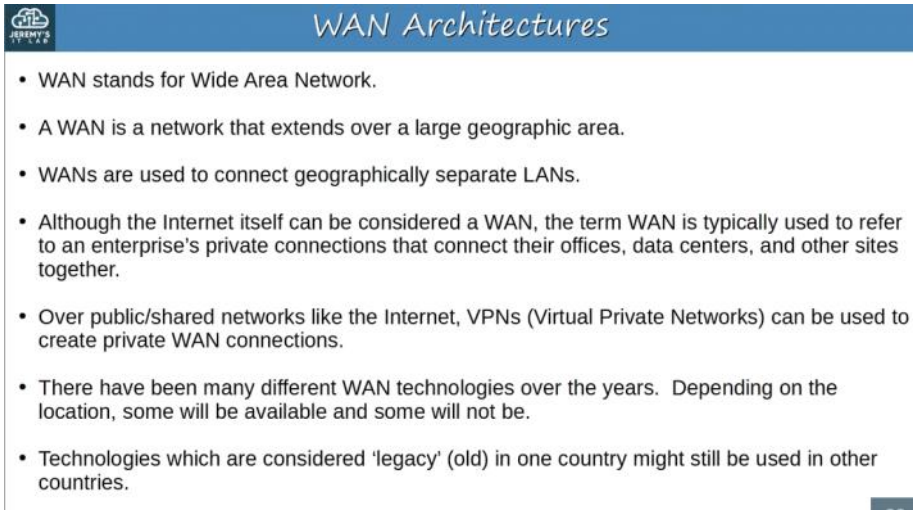
- Small Office/Home Office (SOHO) refers to the office of a small company, or a small home office with few devices.
  - →Doesn't have to be an actual home 'office', if your home has a network connected to the Internet it is considered a SOHO network.
- SOHO networks don't have complex needs, so all networking functions are typically provided by a single device, often called a 'home router' or 'wireless router'.
- This one device can serve as a:
  - → Router
  - → Switch
  - → Firewall
  - → Wireless Access Point
  - → Modem

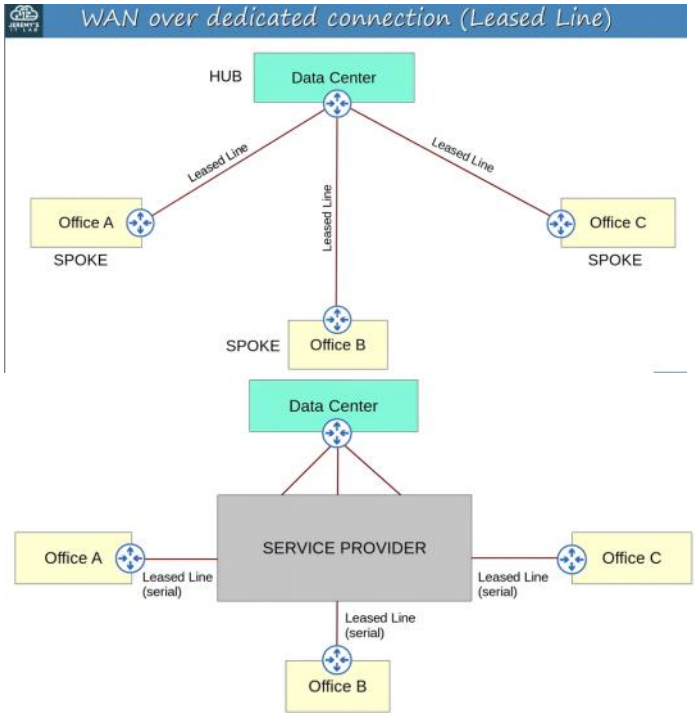


Cisco ACI(application centric infrastructure)=>Spine leaf topology

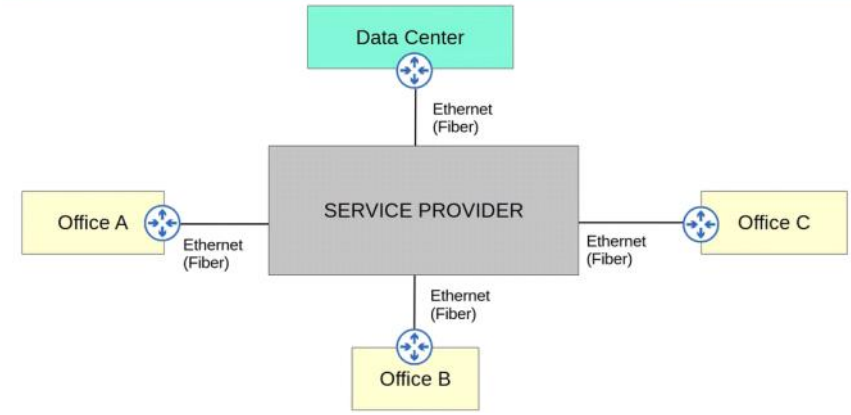
In 2-tier arch's we can make a distribution layer switch the FHRP active for vlan 10 in access layer switch1 and another distribution layer as standby FHRP for the same vlan and same config for all vlan(same config like DLSW1 AS root for vlan 10 and DSW2 as secondary root for vlan 10 and oppposite for vlan 20 in STP also should be configured to synchronize HSRP,STP)

HSRP(hot standby redundancy protocol)
VRRP(virtual router redundancy protocol)

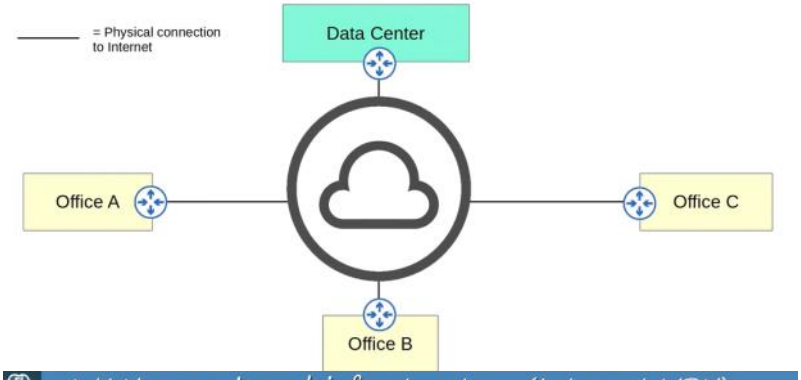




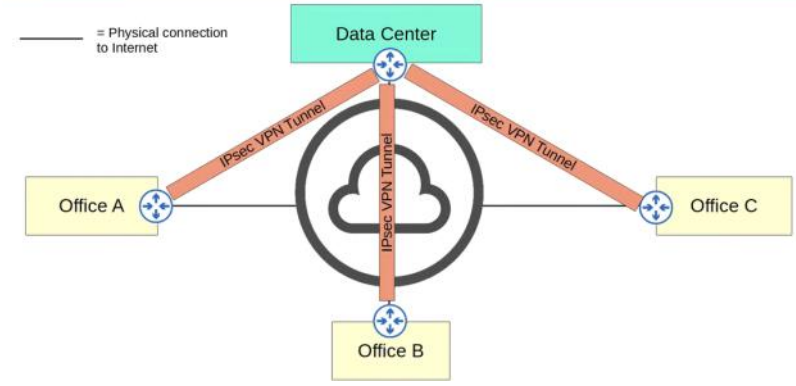
WAN connection via Ethernet (Fiber)



WAN over shared infrastructure (Internet VPN)



WAN over shared infrastructure (Internet VPN)



Leased Lines

- A **leased line** is a dedicated physical link, typically connecting two sites.
- Leased lines use serial connections (PPP or HDLC encapsulation).
- There are various standards that provide different speeds, and different standards are available in different countries.

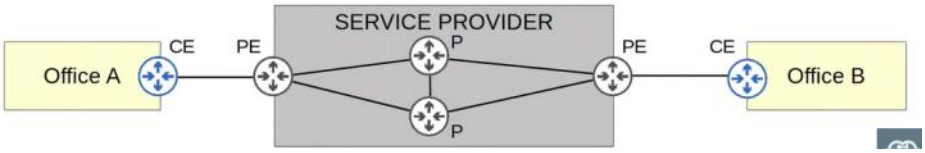
System	North American	Japanese	European (CEPT)
Level zero (channel data rate)	64 kbit/s (DS0)	64 kbit/s	64 kbit/s
First level	1.544 Mbit/s (DS1) (24 user channels) (T1)	1.544 Mbit/s (24 user channels)	2.048 Mbit/s (32 user channels) (E1)
(Intermediate level, T-carrier hierarchy only)	3.152 Mbit/s (DS1C) (48 Ch.)	–	–
Second level	6.312 Mbit/s (DS2) (96 Ch.) (T2)	6.312 Mbit/s (96 Ch.), or 7.786 Mbit/s (120 Ch.)	8.448 Mbit/s (128 Ch.) (E2)
Third level	44.736 Mbit/s (DS3) (672 Ch.) (T3)	32.064 Mbit/s (480 Ch.)	34.368 Mbit/s (512 Ch.) (E3)
Fourth level	274.176 Mbit/s (DS4) (4032 Ch.)	97.728 Mbit/s (1440 Ch.)	139.264 Mbit/s (2048 Ch.) (E4)
Fifth level	400.352 Mbit/s (DS5) (5760 Ch.)	565.148 Mbit/s (8192 Ch.)	565.148 Mbit/s (8192 Ch.) (E5)

Wikipedia: 'Comparison of T-carrier and E-carrier systems'

- Due to the higher cost, higher installation lead time, and slower speeds of leased lines, Ethernet WAN technologies are becoming more popular.

MPLS

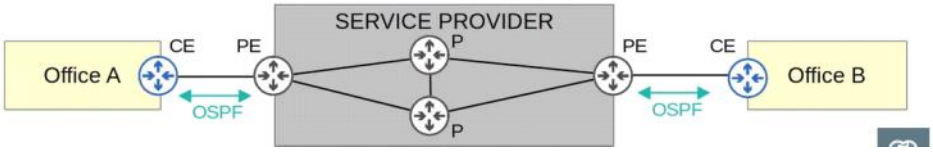
- MPLS stands for 'Multi Protocol Label Switching'.
- Similar to the Internet, service providers' MPLS networks are shared infrastructure because many customer enterprises connect to and share the same infrastructure to make WAN connections.
- However, the *label switching* in the name of MPLS allows VPNs to be created over the MPLS infrastructure through the use of **labels**.
- Some important terms: CE router = Customer Edge router
  PE router = Provider Edge router
  P router = Provider core router



- When the PE routers receive frames from the CE routers, they add a label to the frame.
- These labels are used to make forwarding decisions within the service provider network, not the destination IP.

MPLS

- The CE routers do not use MPLS, it is only used by the PE/P routers.
- When using a *Layer 3 MPLS VPN*, the CE and PE routers peer using OSPF, for example, to share routing information.
- For example, in the diagram below Office A's CE will peer with one PE, and Office B's CE will peer with the other PE.
- Office A's CE will learn about Office B's routes via this OSPF peering, and Office B's CE will learn about Office A's routes too.

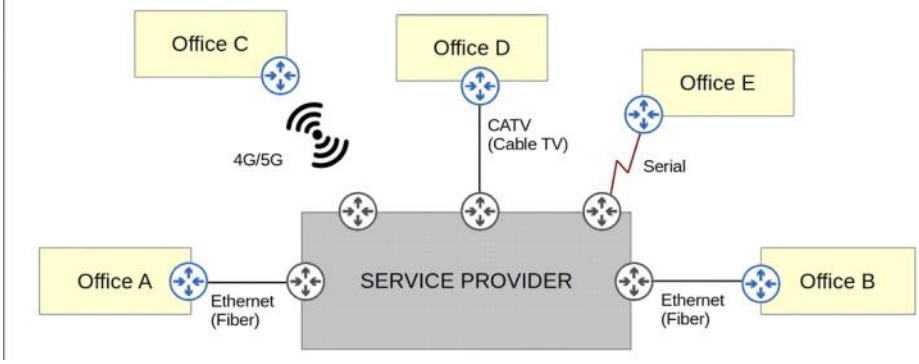


MPLS

- When using a *Layer 2 MPLS VPN*, the CE and PE routers do not form peerings.
- The service provider network is entirely *transparent* to the CE routers.
- In effect, it is like the two CE routers are directly connected.
  → Their WAN interfaces will be in the same subnet.
- If a routing protocol is used, the two CE routers will peer directly with each other.

MPLS

- Many different technologies can be used to connect to a service provider's MPLS network for WAN service.

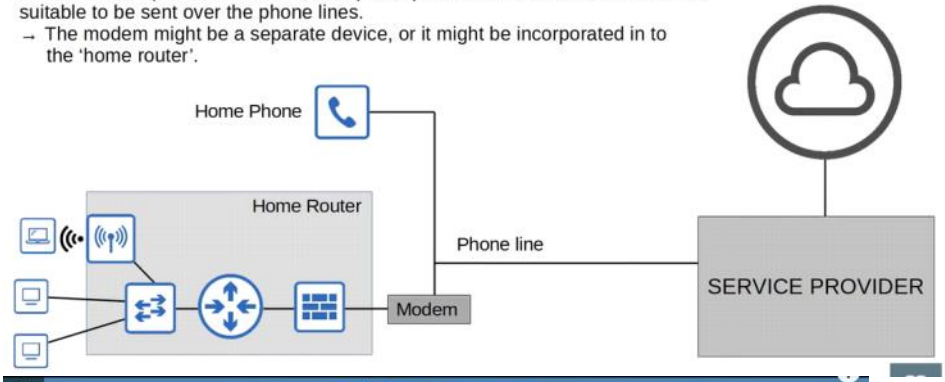


Internet Connections

- There are countless ways for an enterprise to connect to the Internet.
- For example, private WAN technologies such as leased lines and MPLS VPNs can be used to connect to a service provider's Internet infrastructure.
- In addition, technologies such as CATV and DSL commonly used by consumers (home Internet access) can also be used by an Enterprise.
- These days, for both enterprise and consumer Internet access, fiber optic Ethernet connections are growing in popularity due to the high speeds they provide over long distances.
- Let's briefly look at two Internet access technologies mentioned above: cable (CATV) and DSL.

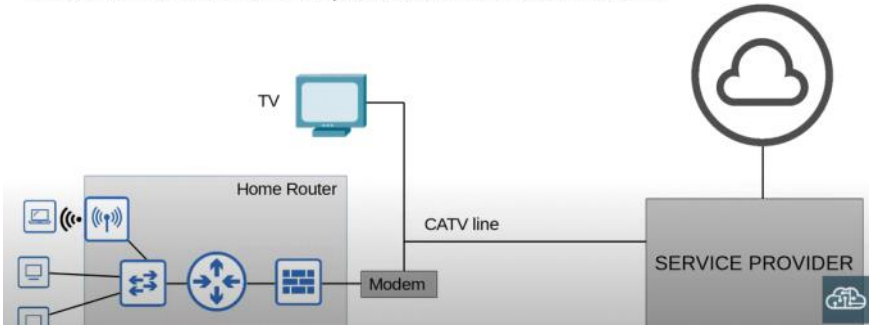
Digital Subscriber Line (DSL)

- DSL provides Internet connectivity to customers over phone lines, and can share the same phone line that is already installed in most homes.
- A DSL modem (modulator-demodulator) is required to convert data into a format suitable to be sent over the phone lines.
  → The modem might be a separate device, or it might be incorporated in to the 'home router'.

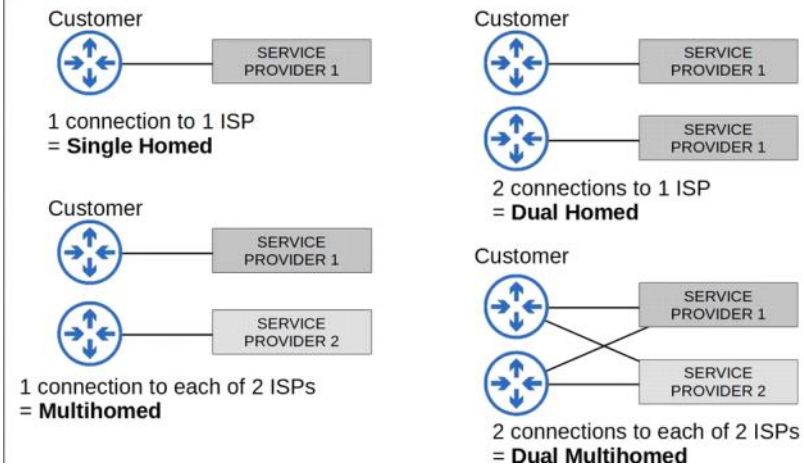


Cable Internet

- Cable Internet provides Internet access via the same CATV (Cable Television) lines used for TV service.
- Like DSL, a cable modem is required to convert data into a format suitable to be sent over the CATV cables
  → Like a DSL modem, this can be a separate device or built into the home router.



Redundant Internet Connections



Internet VPNs

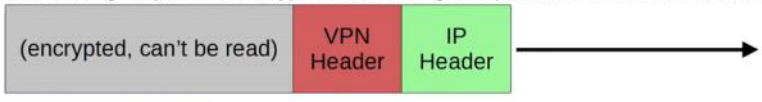
- Private WAN services such as leased lines and MPLS provide security because each customer's traffic is separated by using dedicated physical connections (leased line) or by MPLS tags.
- When using the Internet as a WAN to connect sites together, there is no built-in security by default.
- To provide secure communications over the Internet, VPNs (Virtual Private Networks) are used.
- We will cover two kinds of Internet VPNs:

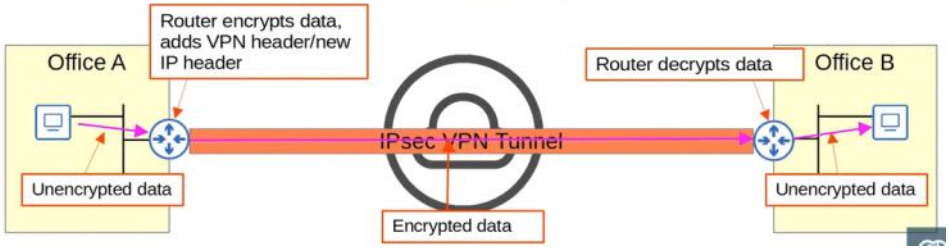
  1) Site-to-Site VPNs using IPsec

  2) Remote-access VPNs using TLS

Site-to-Site VPNs (IPsec)

- A 'site-to-site' VPN is a VPN between two devices and is used to connect two sites together over the Internet.
- A VPN 'tunnel' is created between the two devices by encapsulating the original IP packet with a VPN header and a new IP header.
  → When using IPsec, the original packet is encrypted before being encapsulated with the new header.





Site-to-Site VPNs (IPsec)

- Let's summarize that process:

  1) The sending device combines the original packet and session key (encryption key) and runs them through an encryption formula.

  2) The sending device encapsulates the encrypted packet with a VPN header and a new IP header.

  3) The sending device sends the new packet to the device on the other side of the tunnel.

  4) The receiving device decrypts the data to get the original packet, and then forwards the original packet to its destination.

- In a 'site-to-site' VPN, a tunnel is formed only between two tunnel endpoints (for example, the two routers connected to the Internet).

- All other devices in each site don't need to create a VPN for themselves. They can send unencrypted data to their site's router, which will encrypt it and forward it in the tunnel as described above.

- There are some limitations to standard IPsec:

  1) IPsec doesn't support broadcast and multicast traffic, only unicast. This means that routing protocols such as OSPF can't be used over the tunnels, because they rely on multicast traffic.
  → This can be solved with 'GRE over IPsec'

  2) Configuring a full mesh of tunnels between many sites is a labor-intensive task.
  → This can be solved with Cisco's DMVPN.

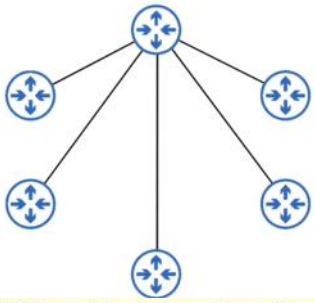
GRE over IPsec

- GRE (Generic Routing Encapsulation) creates tunnels like IPsec, however it does not encrypt the original packet, so it is not secure.
- However, it has the advantage of being able to encapsulate a wide variety of Layer 3 protocols as well as broadcast and multicast messages.
- To get the flexibility of GRE with the security of IPsec, 'GRE over IPsec' can be used.
- The original packet will be encapsulated by a GRE header and a new IP header, and then the GRE packet will be encrypted and encapsulated within an IPsec VPN header and new IP header.

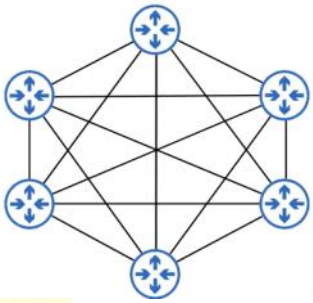
DMVPN

- DMVPN (Dynamic Multipoint VPN) is a Cisco-developed solution that allows routers to dynamically create a full mesh of IPsec tunnels without having to manually configure every single tunnel.

1: Configure IPsec tunnels to a hub site.



2: The hub router gives each router information about how to form an IPsec tunnel with the other routers.

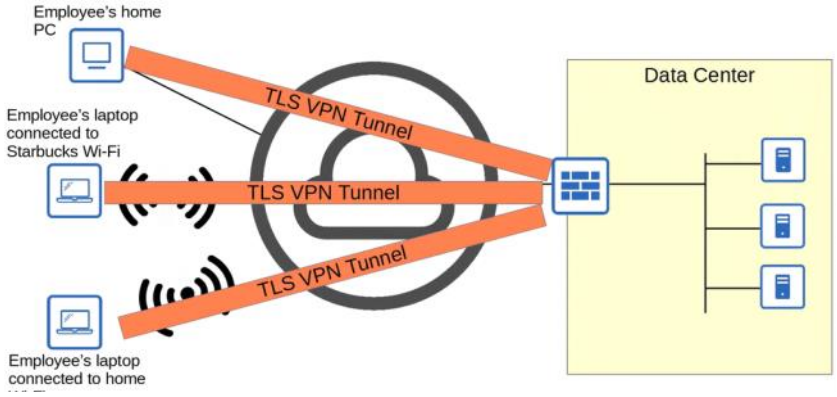


DMVPN provides the configuration simplicity of hub-and-spoke (each spoke router only needs one tunnel configured) and the efficiency of direct spoke-to-spoke communication (spoke routers can communicate directly without traffic passing through the hub)

Remote-Access VPNs

- Whereas site-to-site VPNs are used to make a point-to-point connection between two sites over the Internet, remote-access VPNs are used to allow end devices (PCs, mobile phones) to access the company's internal resources securely over the Internet.
- Remote-access VPNs typically use TLS (Transport Layer Security).
  → TLS is also what provides security for HTTPS (HTTP Secure)
  → TLS was formerly known as SSL (Secure Sockets Layer) and developed by Netscape, but it was renamed to TLS when it was standardized by the IETF.
- VPN client software (for example Cisco AnyConnect) is installed on end devices (for example company-provided laptops that employees use to work from home).
- These end devices then form secure tunnels to one of the company's routers/firewalls acting as a TLS server.
- This allows the end users to securely access resources on the company's internal network without being directly connected to the company network.

Remote-Access VPNs

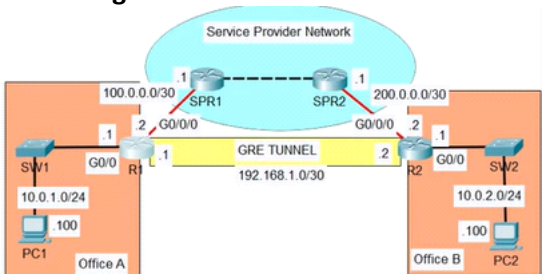


- **Site-to-Site** VPNs typically use IPsec.
- **Remote-Access** VPNs typically use TLS.

- **Site-to-Site** VPNs provide service to many devices within the sites they are connecting.
- **Remote-Access** VPNs provide service to the one end device the VPN client software is installed on.

- **Site-to-Site** VPNs are typically used to permanently connect two sites over the Internet.
- **Remote-Access** VPNs are typically used to provide on-demand access for end devices that want to securely access company resources while connected to a network which is not secure

GRE configuration:



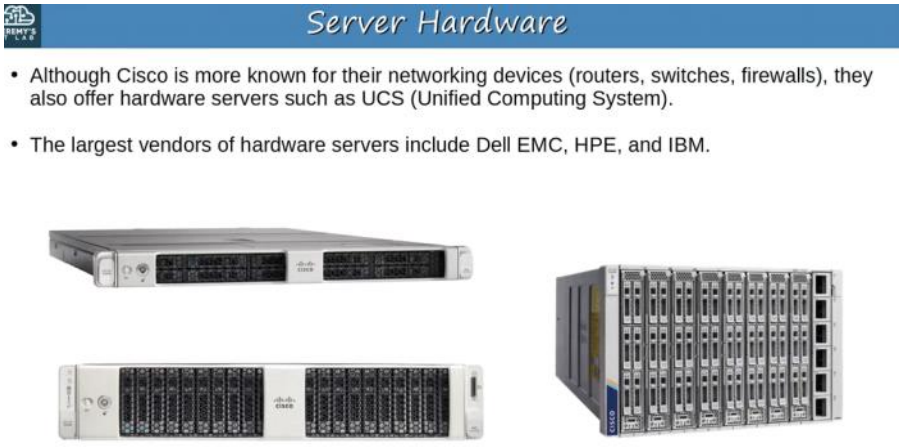

```
R1(config)# int tunnel 0
R1(config-if)# tunnel source g0/0/0
R1(config-if)# tunnel destination 200.0.0.2
R1(config-if)# ip address 192.168.1.1 255.255.255.252

R2(config)# int tunnel 0
R2(config-if)# tunnel source g0/0/0
R2(config-if)# tunnel destination 100.0.0.2
R2(config-if)# ip add 192.168.1.2 255.255.255.252
R2(config)# ip route 0.0.0.0 0.0.0.0 200.0.0.1
```

R1(config)# ip route 0.0.0.0 0.0.0.0 100.0.0.1
(this config will encapsulate the source/dest ip of the tunnel with the original soruce ip's on top of it and send them to the service provider)

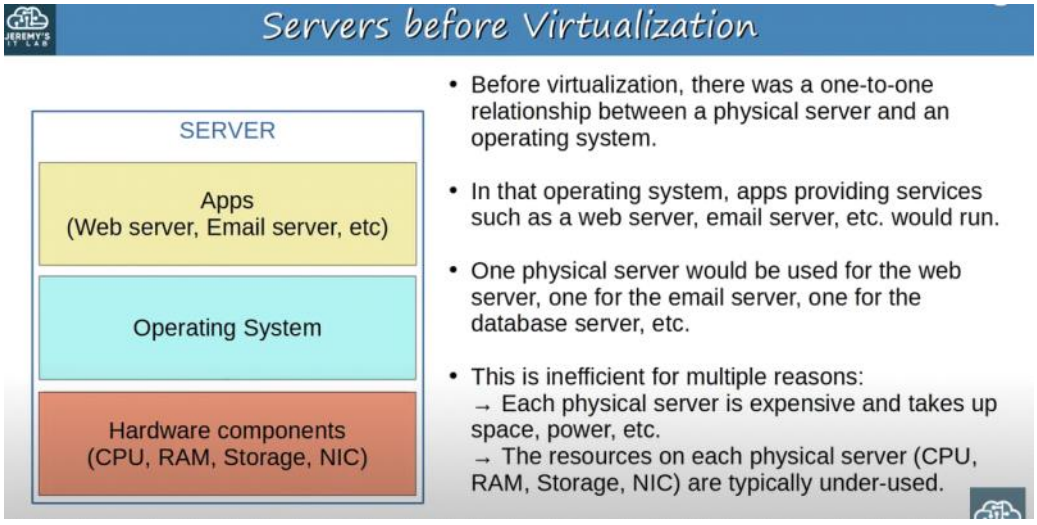
To enable ospf we need to config that as well

R1(config)# router ospf 1
R1(config-if)# network 192.168.1.1 0.0.0.0 area 0
R1(config-if)#network 10.0.1.1 0.0.0.0 area 0
R1(config-if)# passive int g0/0

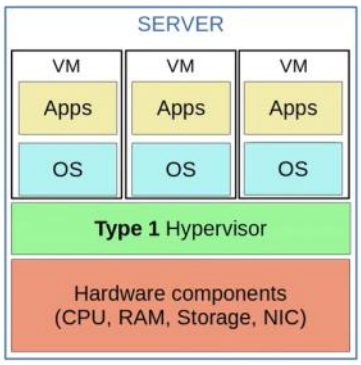
R2(config)# router ospf 1
R2(config-if)# network 192.168.1.2 0.0.0.0 area 0
R2(config-if)#network 10.0.2.1 0.0.0.0 area 0
R2(config-if)# passive int g0/0



In data centers we can see these in a rack full

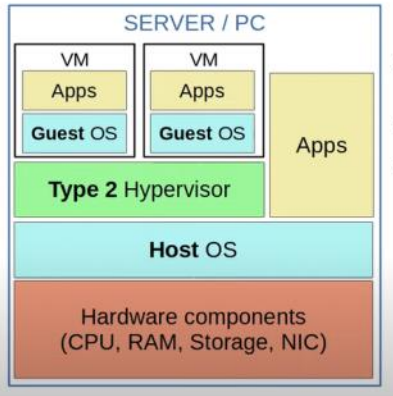


Virtualization (Type 1 Hypervisor)



- Virtualization allows us to break the one-to-one relationship of hardware to OS, allowing multiple OS's to run on a single physical server.
- Each instance is called a VM (Virtual Machine).
- A **hypervisor** is used to manage and allocate the hardware resources (CPU, RAM, etc) to each VM.
- Another name for a hypervisor is VMM (Virtual Machine Monitor).
- The type of hypervisor which runs directly on top of the hardware is called a **Type 1** hypervisor.
  → Examples include VMware ESXi, Microsoft Hyper-V, etc.
- Type 1 hypervisors are also called *bare-metal hypervisors* because they run directly on the hardware (metal).
  → Another term is *native hypervisor*
- This is the type of hypervisor used in data center environments.

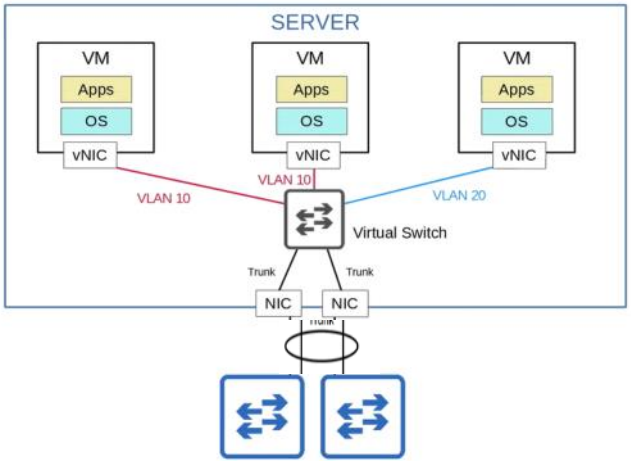
Virtualization (Type 2 Hypervisor)



- **Type 2** hypervisors run as a program on an operating system lik a regular computer program.
  → Examples include VMware Workstation, Oracle VirtualBox, et
- The OS running directly on the hardware is called the **Host OS**, and the OS running in a VM is called a **Guest OS**.
- Another name for a Type 2 hypervisor is *hosted hypervisor*.
- Although Type 2 hypervisors are rarely used in data center environments, they are common on personal-use devices (for example, if a Mac/Linux user needs to run an app that is only supported on Windows, or vice versa).

- **Partitioning:** -Run multiple operating systems on one physical machine.
  -Divide system resources between virtual machines.
- **Isolation:** -Provide fault and security isolation at the hardware level.
  -Preserve performance with advanced resource controls.
- **Encapsulation**: -Save the entire state of a virtual machine to files.
  -Move and copy virtual machines as easily as moving and copying files.
- **Hardware Independence**: -Provision or migrate any virtual machine to any physical server.

Connecting VMs to the Network



- VMs are connected to each other and the external network via a virtual switch running on the hypervisor.
- Just like a regular physical switch, the vSwitch's interfaces can operate as access or trunk ports and use VLANs to separate the VMs at Layer 2.
- Interfaces on the vSwitch connect to the physical NIC (or NICs) of the server to communicate with the external network.

Cloud Services

- Traditional IT infrastructure deployments were some combination of the following:

On-Premises
→ All servers, network devices, and other infrastructure are located on company property.
→ All equipment is purchased and owned by the company using it.
→ The company is responsible for the necessary space, power, and cooling.

Colocation
→ Data centers that rent out space for customers to put their infrastructure (servers, network devices).
→ The data center provides the space, electricity, and cooling.
→ The servers, network devices, etc are still the responsibility of the end customer, although they are not located on the customer's premises.

- Cloud services provide an alternative that is hugely popular, and is continuing to grow.
- Most people associate 'cloud' with public cloud providers such as AWS.
  → Although this is the most common use of cloud services, it's not the only one.

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models.

The five essential characteristics of cloud computing are:

- **On-demand self-service**
- **Broad network access**
- **Resource pooling**
- **Rapid elasticity**
- **Measured service**

The Three Service Models of Cloud

- In cloud computing, everything is provided on a 'service' model.
- For example, rather than the end user buying a physical server, mounting it on a rack, installing the hypervisor, creating the VMs, etc, the service provider offers all of this as a service.
- There are a variety of services referred to as '_____ as a Service' or '__aaS'.

The three service models of cloud computing are:

- **Software as a Service (SaaS)**
- **Platform as a Service (PaaS)**
- **Infrastructure as a Service (IaaS)**

The Four Deployment Models of Cloud

- Most people assume that 'cloud' means public cloud providers such as AWS, Azure, and GCP.
- Although 'Public cloud' is the most common deployment model, it's not the only one.

The four deployment models of cloud computing are:

- **Private cloud**
- **Community cloud**
- **Public cloud**
- **Hybrid cloud**

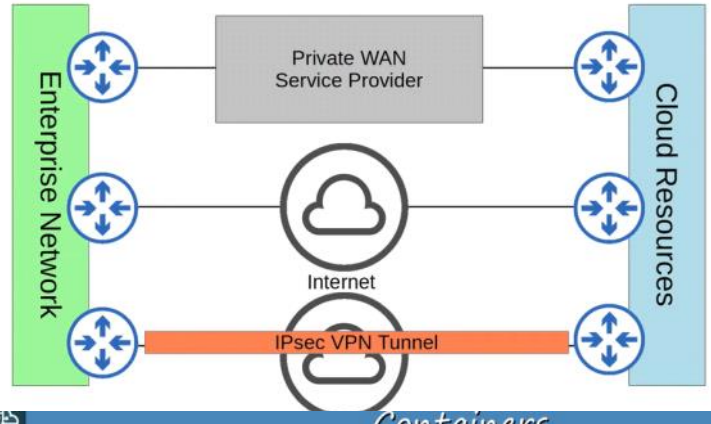
The Four Deployment Models of Cloud

- Most people assume that 'cloud' means public cloud providers such as AWS, Azure, and GCP.
- Although 'Public cloud' is the most common deployment model, it's not the only one.

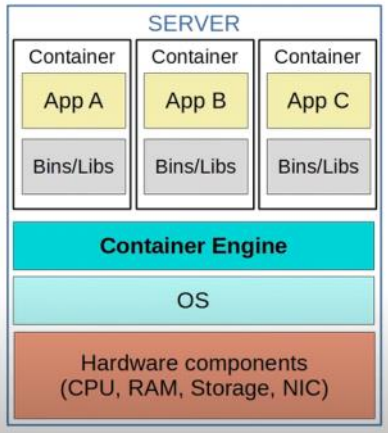
The four deployment models of cloud computing are:

- **Private cloud**
- **Community cloud**
- **Public cloud**
- **Hybrid cloud**

Connecting to Cloud Resources



Containers



- **Containers** are software packages that contain an App and all dependencies (*Bins/Libs* in the diagram) for the contained App to run.
  → Multiple Apps can be run in a single container, but this is not how containers are usually used.
- Containers run on a **Container Engine** (ie. Docker Engine)
  → The container engine is run on a host OS (usually Linux).
- Containers are lightweight (small in size) and include only the dependencies required to run the specific App.
- A **Container Orchestrator** is a software platform for automating the deployment, management, scaling etc. of containers.
  → **Kubernetes** (originally designed by Google) is the most popular container orchestrator.
  → **Docker Swarm** is Docker's container orchestration tool.
- In small numbers manual operation is possible, but large-scale systems (ie. with *Microservices*) can require thousands of containers.

Microservice Architecture is an approach to software architecture that divides a larger solution into smaller parts (microservices).
→ Those microservices all run in containers that can be orchestrated by Kubernetes (or another platform).

- **VMs** are more isolated because each VM runs its own OS.
- **Containers** are less isolated because they all run on the same OS; if the OS crashes, all containers running on it are effected.

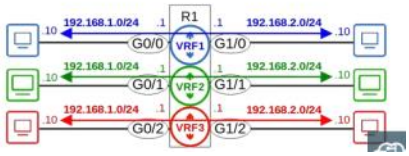
Docker swarm and kubernetes are container orchestrators

PART11

Tuesday, February 21, 2023 10:44 AM

VRF

- **Virtual Routing & Forwarding** is used to divide a single router into multiple virtual routers.
  - →Similar to how VLANs are used to divide a single switch (LAN) into multiple virtual switches (VLANs).
- It does this by allowing a router to build multiple separate routing tables.
  - →Interfaces (Layer 3 only) & routes are configured to be in a specific **VRF** (aka *VRF Instance*).
  - →Router interfaces, SVIs & routed ports on multilayer switches can be configured in a VRF.
- Traffic in one VRF cannot be forwarded out of an interface in another VRF.
  - →As an exception, *VRF Leaking* can be configured to allow traffic to pass between VRF's.
- VRF is commonly used to facilitate MPLS.
  - →The kind of VRF we are talking about is **VRF-lite** (VRF without MPLS).
- VRF is commonly used by service providers to allow one device to carry traffic from multiple customers.
  - →Each customer's traffic is isolated from the others.
  - →Customer IP addresses can overlap without issues.



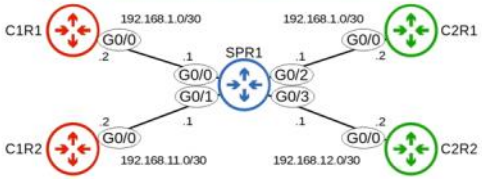
VRF Configuration

```
SPR1(config)# interface g0/0
SPR1(config-if)# ip address 192.168.1.1 255.255.255.252
SPR1(config-if)# no shutdown

SPR1(config-if)# interface g0/1
SPR1(config-if)# ip address 192.168.11.1 255.255.255.252
SPR1(config-if)# no shutdown

SPR1(config-if)# interface g0/2
SPR1(config-if)# ip address 192.168.1.1 255.255.255.252
% 192.168.1.0 overlaps with GigabitEthernet0/0

SPR1(config-if)# ip address 192.168.1.2 255.255.255.252
% 192.168.1.0 overlaps with GigabitEthernet0/0
```

G0/2 cannot use IP address 192.168.1.1 because it is in the same subnet as G0/0 (in this case it's the exact same IP address).

Even if the IP address is different, G0/2 cannot be configured in the same subnet as G0/0.

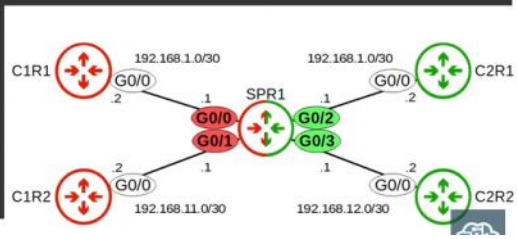
Without the use of VRF, two interfaces on the same router cannot be in the same subnet.



VRF Configuration

```
SPR1(config)# ip vrf CUSTOMER1
SPR1(config-vrf)# ip vrf CUSTOMER2
SPR1(config-vrf)# do show ip vrf
  Name                             Default RD          Interfaces
  CUSTOMER1                        <not set>
  CUSTOMER2                        <not set>

SPR1(config-vrf)# interface g0/0
SPR1(config-if)# ip vrf forwarding CUSTOMER1
% Interface GigabitEthernet0/0 IPv4 disabled and address(es) removed due to enabling VRF CUSTOMER1
SPR1(config-if)# ip address 192.168.1.1 255.255.255.252

SPR1(config-if)# interface g0/1
SPR1(config-if)# ip vrf forwarding CUSTOMER1
% Interface GigabitEthernet0/1 IPv4 disabled and address(es) removed due to enabling VRF CUSTOMER1
SPR1(config-if)# ip address 192.168.11.1 255.255.255.252

SPR1(config-if)# interface g0/2
SPR1(config-if)# ip vrf forwarding CUSTOMER2
SPR1(config-if)# ip address 192.168.1.1 255.255.255.252
SPR1(config-if)# no shutdown
SPR1(config-if)# interface g0/3
SPR1(config-if)# ip vrf forwarding CUSTOMER2
SPR1(config-if)# ip address 192.168.12.1 255.255.255.252
SPR1(config-if)# no shutdown
SPR1(config-if)# do show ip vrf
  Name                 Default RD          Interfaces
  CUSTOMER1            <not set>           Gi0/0
                                           Gi0/1
  CUSTOMER2            <not set>           Gi0/2
                                           Gi0/3
```

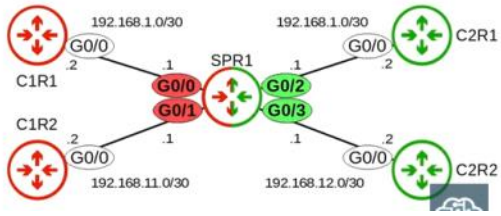
If an interface has an IP address configured, the IP address will be removed when you assign the interface to a VRF.



VRF Configuration

```
SPR1# 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
       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
       a - application route
       + - replicated route, % - next hop override, p - overrides from PfR

Gateway of last resort is not set

SPR1# show ip route vrf CUSTOMER1

Routing Table: CUSTOMER1
!output omitted

      192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.1.0/30 is directly connected, GigabitEthernet0/0
L        192.168.1.1/32 is directly connected, GigabitEthernet0/0
      192.168.11.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.11.0/30 is directly connected, GigabitEthernet0/1
L        192.168.11.1/32 is directly connected, GigabitEthernet0/1

SPR1# show ip route vrf CUSTOMER2

Routing Table: CUSTOMER2
!output omitted

      192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.1.0/30 is directly connected, GigabitEthernet0/2
L        192.168.1.1/32 is directly connected, GigabitEthernet0/2
      192.168.12.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.12.0/30 is directly connected, GigabitEthernet0/3
L        192.168.12.1/32 is directly connected, GigabitEthernet0/3
```

show ip route displays the *global routing table.*
*All of SPR1's interfaces are configured in VRFs, so nothing displays here.
*You can have a mix of interfaces using and not using VRFs.



When pinging from the SPR1 to customer 1 ip then we should use
SPR1# ping vrf CUSTOMER1 192.168.1.2

Wireless Networks

- Although we will briefly look at other types of wireless networks, in this section of the course we will be focusing on wireless LANs using Wi-Fi.
- The standards we use for wireless LANs are defined in IEEE 802.11.
- The term **Wi-Fi** is a trademark of the **Wi-Fi Alliance**, not directly connected to the IEEE.
- The Wi-Fi Alliance tests and certifies equipment for 802.11 standards compliance interoperability with other devices.

- However, Wi-Fi has become the common term that people use to refer to 802.11 wireless LANs, and I will use both terms throughout these videos.

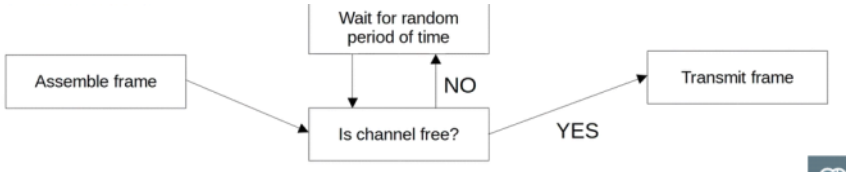
Wireless Networks

- Wireless networks have some issues that we need to deal with.

1) All devices within range receive all frames, like devices connected to an Ethernet hub.
   → Privacy of data within the LAN is a greater concern.
   → CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) is used to facilitate half-duplex communications.

- **CSMA/CD** is used in wired networks to detect and recover from collisions.
- **CSMA/CA** is used in wireless networks to avoid collisions.

- When using **CSMA/CA**, a device will wait for other devices to stop transmitting before it transmits data itself.



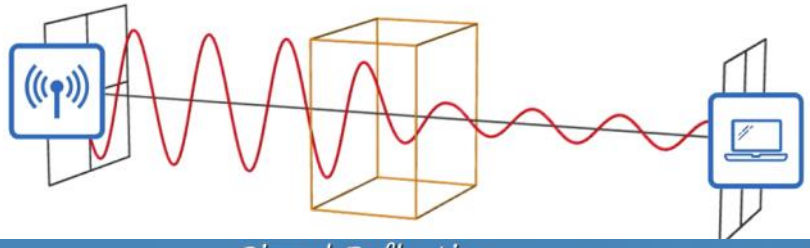
Wifi is like ethernet hub in the center and devices connected to it thourhg wire but here wifi is in the middle and devices are connected wirelessly

2) Wireless communications are regulated by various international and national bodies.

3) Wireless signal coverage area must be considered.
→ Signal range.
→ Signal **absorption**, **reflection**, **refraction**, **diffraction**, and **scattering**.

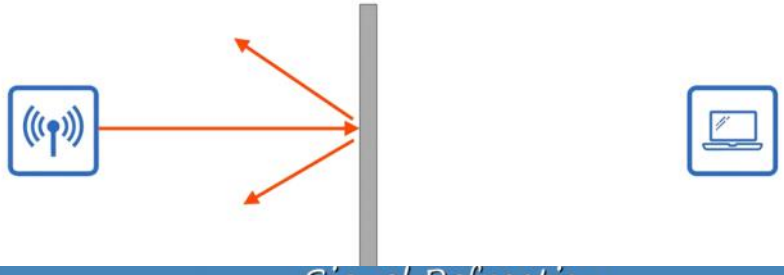
Signal Absorption

- **Absorption** happens when a wireless signal passes through a material and is converted into heat, weakening the original signal.



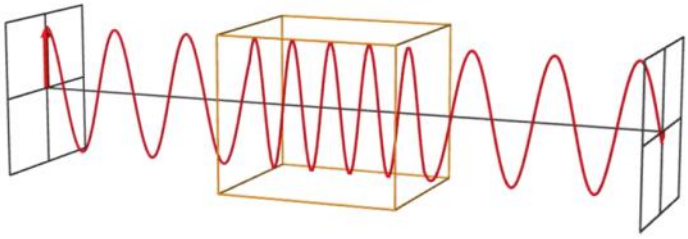
Signal Reflection

- **Reflection** happens when a signal bounces off of a material, for example metal.
  → This is why Wi-Fi reception is usually poor in elevators. The signal bounces off the metal and very little penetrates into the elevator.



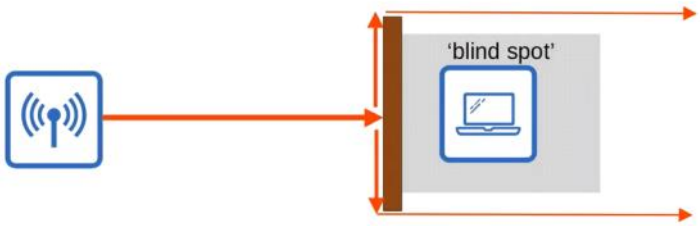
Signal Refraction

- **Refraction** happens when a wave is bent when entering a medium where the signal travels at a different speed.
  → For example, glass and water can refract waves.



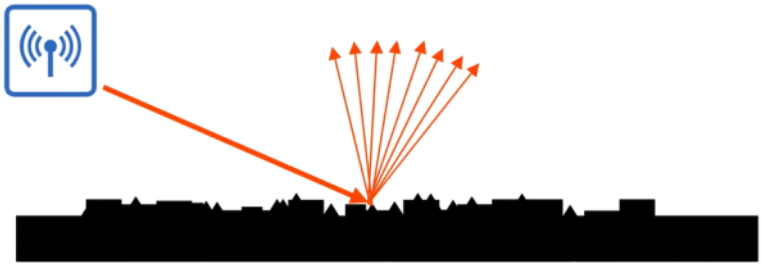
Signal Diffraction

- **Diffraction** happens when a wave encounters an obstacle and travels around it.
  → This can result in 'blind spots' behind the obstacle.



Signal Scattering

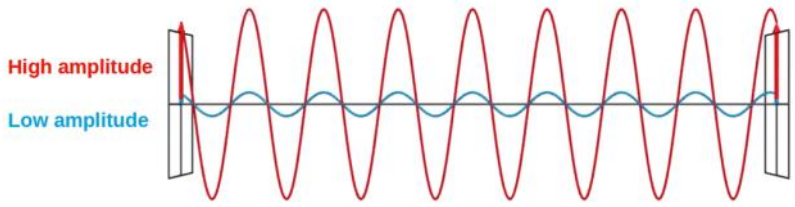
- **Scattering** happens when a material causes a signal to scatter in all directions.
  → Dust, smog, uneven surfaces, etc. can cause scattering.



4) Other devices using the same channels can cause interference.
   → For example, a wireless LAN in your neighbor's house/apartment.

Radio Frequency

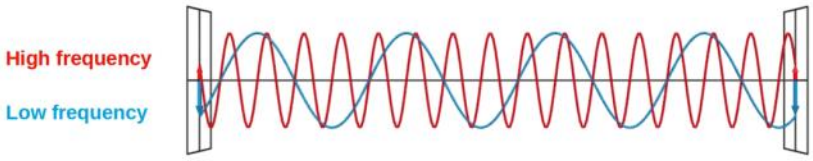
- To send wireless signals, the sender applies an alternating current to an antenna.
  → This creates electromagnetic fields which propagate out as waves.
- Electromagnetic waves can be measured in multiple ways for example **amplitude** and **frequency**.
- **Amplitude** is the maximum strength of the electric and magnetic fields.



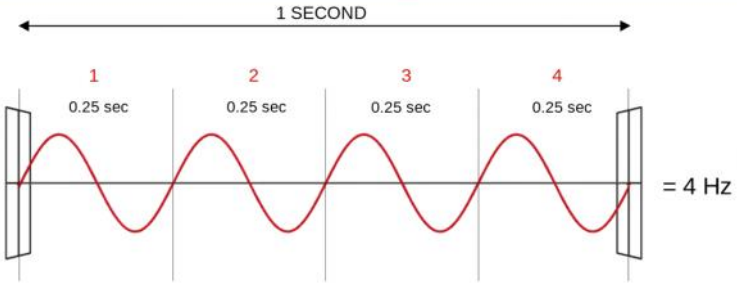
Radio Frequency

- **Frequency** measures the number of up/down cycles per a given unit of time.

- The most common measurement of frequency is **hertz**.
  - → Hz (Hertz) = cycles per second
  - → kHz (Kilohertz) = 1,000 cycles per second
  - → MHz (Megahertz) = 1,000,000 cycles per second
  - → GHz (Gigahertz) = 1,000,000,000 cycles per second
  - → THz (Terahertz) = 1,000,000,000,000 cycles per second



Radio Frequency



- Another important term is **period**, the amount of time of one cycle.
  - → If the **frequency** is 4 Hz, the **period** is 0.25 seconds.

Radio Frequency

- The visible frequency range is about 400 THz to 790 THz.
- The radio frequency range is from 30 Hz to 300 GHz and is used for many purposes.

Band name	Abbreviation	ITU band number	Frequency and Wavelength	Example Uses
Extremely low frequency	ELF	1	3–30 Hz 100,000–10,000 km	Communication with submarines
Super low frequency	SLF	2	30–300 Hz 10,000–1,000 km	Communication with submarines
Ultra low frequency	ULF	3	300–3,000 Hz 1,000–100 km	Submarine communication, communication within mines
Very low frequency	VLF	4	3–30 kHz 100–10 km	Navigation, time signals, submarine communication, wireless heart rate monitors, geophysics
Low frequency	LF	5	30–300 kHz 10–1 km	Navigation, time signals, AM longwave broadcasting (Europe and parts of Asia), RFID, amateur radio
Medium frequency	MF	6	300–3,000 kHz 1,000–100 m	AM (medium-wave) broadcasts, amateur radio, avalanche beacons
High frequency	HF	7	3–30 MHz 100–10 m	Shortwave broadcasts, citizens band radio, amateur radio and over-the-horizon aviation communications, RFID, over-the-horizon radar, automatic link establishment (ALE) / near-vertical incidence skywave (NVIS) radio communications, marine and mobile radio telephony
Very high frequency	VHF	8	30–300 MHz 10–1 m	FM, television broadcasts, line-of-sight ground-to-aircraft and aircraft-to-aircraft communications, land mobile and maritime mobile communications, amateur radio, weather radio
Ultra high frequency	UHF	9	300–3,000 MHz 1–0.1 m	Television broadcasts, microwave oven, microwave devices/communications, radio astronomy, mobile phones, wireless LAN, Bluetooth, ZigBee, GPS and two-way radios such as land mobile, FRS and GMRS radios, amateur radio, satellite radio, Remote control Systems, ADSB.
Super high frequency	SHF	10	3–30 GHz 100–10 mm	Radio astronomy, microwave devices/communications, wireless LAN, DSRC, most modern radars, communications satellites, cable and satellite television broadcasting, DBS, amateur radio, satellite radio.
Extremely high frequency	EHF	11	30–300 GHz 10–1 mm	Radio astronomy, high-frequency microwave radio relay, microwave remote sensing, amateur radio, directed-energy weapon, millimeter wave scanner, Wireless Lan 802.11ad.
Terahertz or Tremendously high frequency	THz or THF	12	300–3,000 GHz 1–0.1 mm	Experimental medical imaging to replace X-rays, ultrafast molecular dynamics, condensed-matter physics, terahertz time-domain spectroscopy, terahertz computing/communications, remote sensing

Radio Frequency Bands

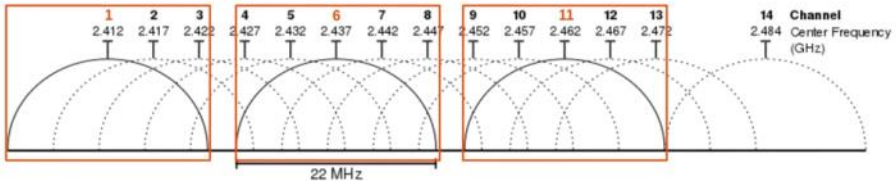
- Wi-Fi uses two main *bands* (frequency ranges)
- **2.4 GHz** band
  → The actual range is **2.400 GHz** to **2.4835 GHz**
- **5 GHz** band
  → The actual range is from **5.150 GHz** to **5.825 GHz**
  → Divided into four smaller bands: **5.150 GHz** to **5.250 GHz**
  5.250 GHz to **5.350 GHz**
  5.470 GHz to **5.725 GHz**
  5.725 GHz to **5.825 GHz**
- The 2.4 GHz band typically provides further reach in open space and better penetration of obstacles such as walls.
  → However, more devices tend to use the 2.4 GHz band so interference can be a bigger problem compared to the 5 GHz band.
- ***Wi-Fi 6** (802.11ax)has expanded the spectrum range to include a band in the **6 GHz** range.

Channels

- Each band is divided up into multiple 'channels'.
  → Devices are configured to transmit and receive traffic on one (or more) of these channels.
- The 2.4 GHz band is divided into several channels, each with a 22 MHz range.

Channels

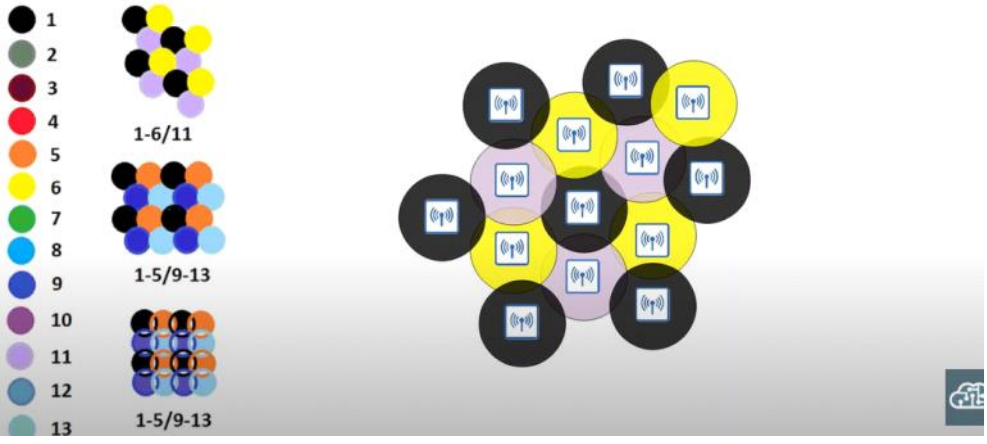
- In a small wireless LAN with only a single AP, you can use any channel.
- However, in larger WLANs with multiple APs, it's important that adjacent APs don't use overlapping channels. This helps avoid interference.
- In the 2.4 GHz band, it is recommended to use channels **1, 6, and 11**.



- Outside of North America you could use other combinations, but for the CCNA exam remember **1, 6, and 11**.
- The 5 GHz band consists of non-overlapping channels, so it is much easier to avoid interference between adjacent APs.

Channels

- Using channels 1, 6, and 11, you can place APs in a 'honeycomb' pattern to provide complete coverage of an area without interference between channels.



802.11 Standards

Standard	Frequencies	Max Data Rate (theoretical)	Alternate Name
802.11	2.4 GHz	2 Mbps	
802.11b	2.4 GHz	11 Mbps	
802.11a	5 GHz	54 Mbps	
802.11g	2.4 GHz	54 Mbps	
802.11n	2.4 / 5 GHz	600 Mbps	'Wi-Fi 4'
802.11ac	5 GHz	6.93 Gbps	'Wi-Fi 5'
802.11ax	2.4 / 5 / 6 GHz	4*802.11ac	Wi-Fi 6'

Service Sets

- 802.11 defines different kinds of **service sets** which are groups of wireless network devices.
- There are three main types:
  - → Independent
  - → Infrastructure
  - → Mesh
- All devices in a service set share the same **SSID** (**service set identifier**).
- The SSID is a human-readable name which identifies the service set.
- The SSID does **not** have to be unique.

Service Sets: IBSS

An **IBSS** (**Independent Basic Service Set**) is a wireless network in which two or more wireless devices connect directly without using an **AP** (**Access Point**).

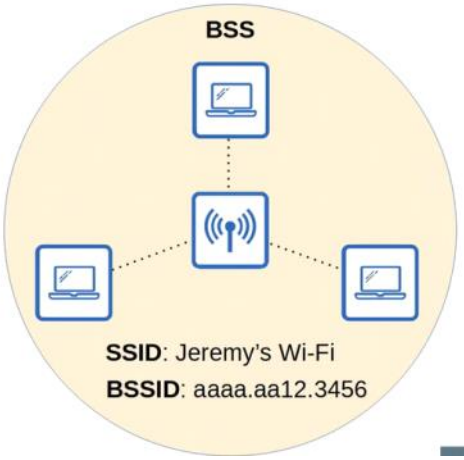
Also called an **ad hoc** network.

Can be used for file transfer (ie. AirDrop).

Not scalable beyond a few devices.

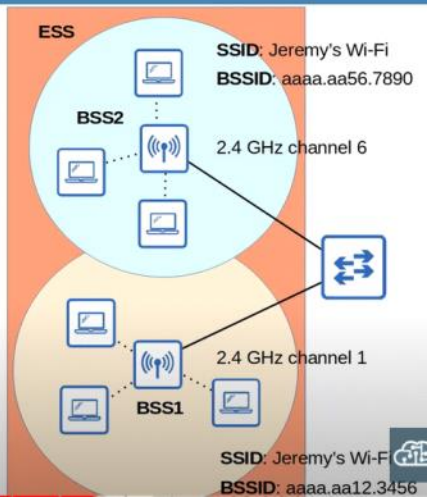
Service Sets: BSS

- A **BSS** (**Basic Service Set**) is a kind of Infrastructure Service Set in which clients connect to each other via an **AP** (**Access Point**), but not directly to each other.
- A **BSSID** (**Basic Service Set ID**) is used to uniquely identify the AP.
  → Other APs can use the same SSID, but not the same BSSID
  → The BSSID is the MAC address of the AP's radio
- Wireless devices request to *associate* with the BSS.
- Wireless devices that have associated with the BSS are called 'clients' or 'stations'.
- *The area around an AP where its signal is usable is called a **BSA** (**Basic Service Area**).



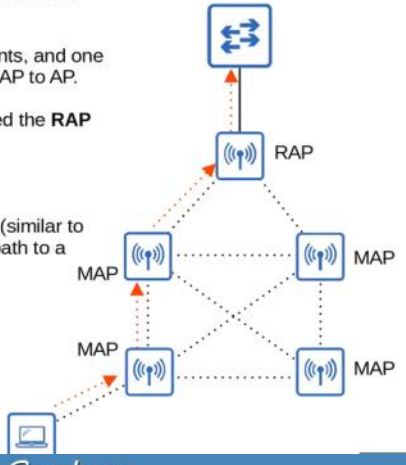
Service Sets: ESS

- To create larger wireless LANs beyond the range of a single AP, we use an **ESS** (**Extended Service Set**).
- APs with their own BSSs are connected by a wired network.
  → Each BSS uses the same SSID.
  → Each BSS has a unique BSSID.
  → Each BSS uses a different channel to avoid interference.
- Clients can pass between APs without having to reconnect, providing a seamless Wi-Fi experience when moving between APs.
  → This is called **roaming**.
- The BSAs should overlap about 10-15%.



Service Sets: MBSS

- An **MBSS** (**Mesh Basic Service Set**) can be used in situations where it's difficult to run an Ethernet connection to every AP.
- Mesh APs use two radios: one to provide a BSS to wireless clients, and one to form a 'backhaul network' which is used to bridge traffic from AP to AP.
- At least one AP is connected to the wired network, and it is called the **RAP** (**Root Access Point**).
- The other APs are called **MAP**s (**Mesh Access Points**).
- A protocol is used to determine the best path through the mesh (similar to how dynamic routing protocols are used to determine the best path to a destination).

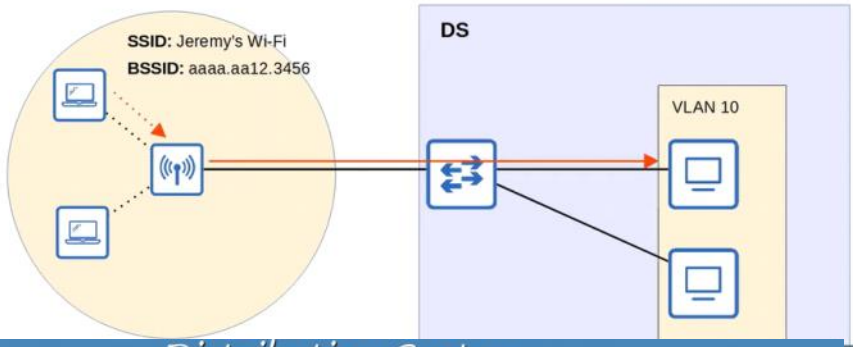


Distribution System

- Most wireless networks aren't standalone networks.
  →Rather, they are a way for wireless clients to connect to the wired network infrastructure.
- In 802.11, the upstream wired network is called the **DS** (**Distribution System**).
- Each wireless BSS or ESS is mapped to a VLAN in the wired network.

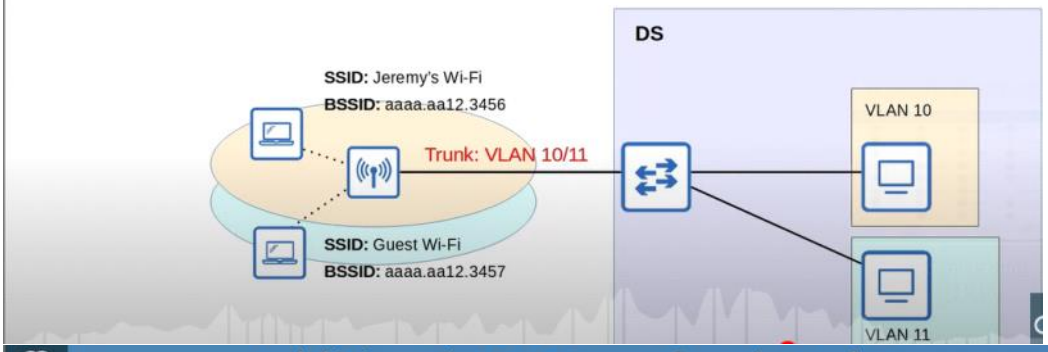
Distribution System

- Most wireless networks aren't standalone networks.
  →Rather, they are a way for wireless clients to connect to the wired network infrastructure.
- In 802.11, the upstream wired network is called the **DS** (**Distribution System**).
- Each wireless BSS or ESS is mapped to a VLAN in the wired network.



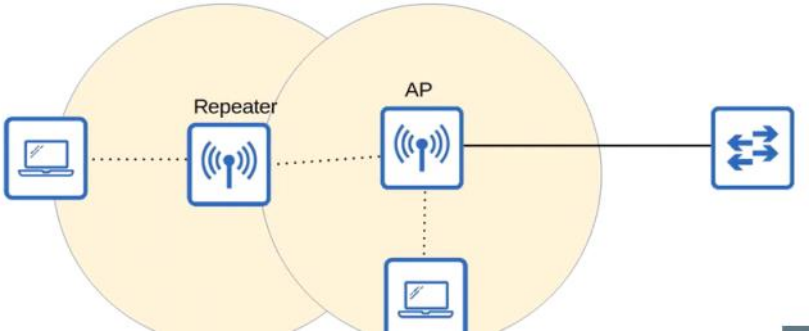
Distribution System

- It's possible for an AP to provide multiple wireless LANs, each with a unique SSID.
- Each WLAN is mapped to a separate VLAN and connected to the wired network via a trunk.
- Each WLAN uses a unique BSSID, usually by incrementing the last digit of the BSSID by one.



Additional AP Operational Modes

- APs can operate in additional modes beyond the ones we've introduced so far.
- An AP in **repeater** mode can be used to extend the range of a BSS.
- The repeater will simply retransmit any signal it receives from the AP.
  →A repeater with a single radio must operate on the same channel as the AP, but this can drastically reduce the overall throughput on the channel.

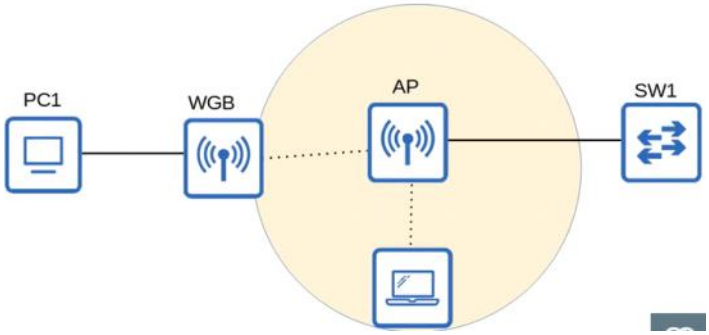


Additional AP Operational Modes

- A **workgroup bridge** (WGB) operates as a wireless client of another AP, and can be used to connect wired devices to the wireless network.
- In the example below, PC1 does not have wireless capabilities, and also does not have access to a wired connection to SW1.
- PC1 has a wired connection to the WGB, which has a wireless connection to the AP.

There are two kinds of WGBs:
Universal WGB (uWGB) is an 802.11 standard that allows one device to be bridged to the wireless network.

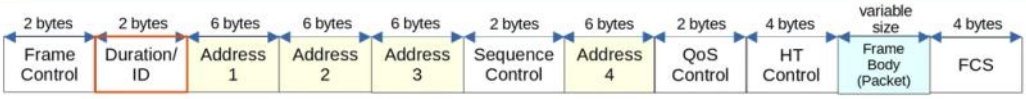
WGB is a Cisco-proprietary version of the 802.11 standard that allows multiple wired clients to be bridged to the wireless network.



Additional AP Operational Modes

- An **outdoor bridge** can be used to connect networks over long distances without a physical cable connecting them.
- The APs will use specialized antennas that focus most of the signal power in one direction, which allows the wireless connection to be made over longer distances than normally possible.
- The connection can be point-to-point as in the diagram below, or point-to-multipoint in which multiple sites connect to one central site.

802.11 Frame Format



- 802.11 frames have a different format than 802.3 Ethernet frames.
- For the CCNA, you don't have to learn it in as much detail as the Ethernet and IP headers.
- Depending on the 802.11 version and the message type, some of the fields might not be present in the frame.
  → For example, not all messages use all 4 address fields.
- **Frame Control**: Provides information such as the message type and subtype.
- **Duration/ID**: Depending on the message type, this field can indicate:
  → the time (in microseconds) the channel will be dedicated for transmission of the frame.
  → and identifier for the association (connection).
- **Addresses**: Up to four addresses can be present in an 802.11 frame. Which addresses are present, and their order, depends on the message type.
  → Destination Address (DA): Final recipient of the frame
  → Source Address (SA): Original sender of the frame
  → Receiver Address (RA): Immediate recipient of the frame
  → Transmitter Address (TA): Immediate sender of the frame
- **Sequence Control**: Used to reassemble fragments and eliminate duplicate frames.
- **QoS Control**: Used in QoS to prioritize certain traffic.

- **Addresses**: Up to four addresses can be present in an 802.11 frame. Which addresses are present, and their order, depends on the message type.
  → Destination Address (DA): Final recipient of the frame
  → Source Address (SA): Original sender of the frame
  → Receiver Address (RA): Immediate recipient of the frame
  → Transmitter Address (TA): Immediate sender of the frame

- **Sequence Control**: Used to reassemble fragments and eliminate duplicate frames.

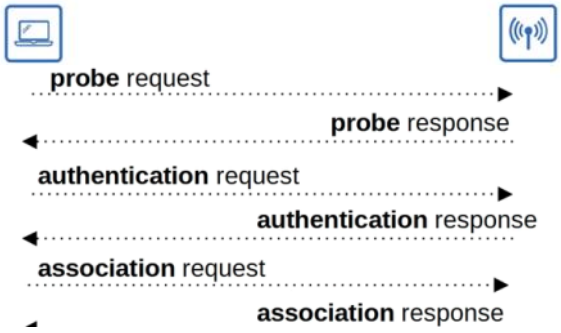
- **QoS Control**: Used in QoS to prioritize certain traffic.

- **HT (High Throughput) Control**: Added in 802.11n to enable High Throughput operations.
  → 802.11n is also known as 'High Throughput' (HT) Wi-Fi
  → 802.11ac is also known as 'Very High Throughput' (VHT) Wi-Fi

- **FCS (Frame Check Sequence)**: Same as in an Ethernet frame, used to check for errors.

802.11 Association Process

- Access Points bridge traffic between wireless stations and other devices.
- For a station to send traffic through the AP, it must be associated with the AP.
- There are three 802.11 connection states:
  → Not authenticated, not associated.
  → Authenticated, not associated.
  → Authenticated and associated.
- The station must be authenticated and associated with the AP to send traffic through it.



There are two ways a station can scan for a BSS:

→ **Active scanning**: The station sends probe requests and listens for a probe response from an AP.

→ **Passive scanning**: The station listens for **beacon** messages from an AP.
Beacon messages are sent periodically by APs to advertise the BSS.

802.11 Message Types

- There are three 802.11 message types:

- **Management**: used to manage the BSS.
  → Beacon
  → Probe request, probe response
  → Authentication
  → Association request, association response

- **Control**: Used to control access to the medium (radio frequency). Assists with delivery of management and data frames.
  → RTS (Request to Send)
  → CTS (Clear to Send)
  → ACK

- **Data**: Used to send actual data packets.

Autonomous APs

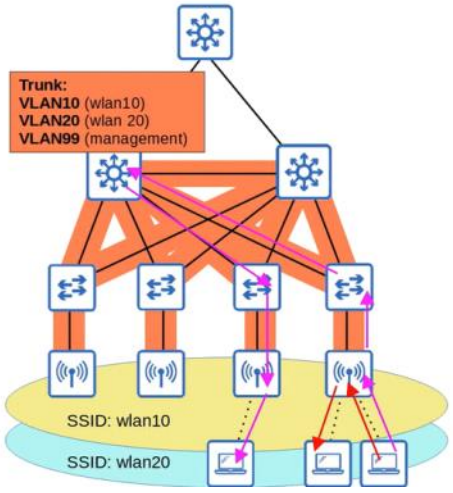
- There are three main wireless AP deployment methods:
  → Autonomous
  → Lightweight
  → Cloud-based

- **Autonomous AP**s are self-contained systems that don't rely on a WLC.

- Autonomous APs are configured individually.
  → Can be configured by console cable (CLI), telnet/SSH (CLI), or HTTP/HTTPS web connection (GUI).
  → An IP address for remote management should be configured.
  → The RF parameters must be manually configured (transmit power, channel, etc.)
  → Security policies are handled individually by each AP.
  → QoS rules etc. are configured individually on each AP.

- There is no central monitoring or management of APs.

Autonomous APs

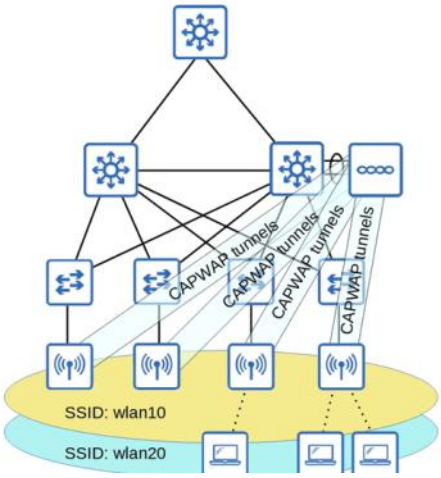


- Autonomous APs connect to the wired network with a trunk link.
- Data traffic from wireless clients has a very direct path to the wired network or to other wireless clients associated with the same AP.
- Each VLAN has to stretch across the entire network. This is considered bad practice.
  → Large broadcast domains
  → Spanning tree will disable links
  → Adding/deleting VLANs is very labor-intensive
- Autonomous APs can be used in small networks, but they are not viable in medium to large networks.
  → Large networks can have thousands of APs.
- Autonomous APs can also function in the modes covered in the previous video: Repeater, Outdoor Bridge, Workgroup Bridge.

Lightweight APs

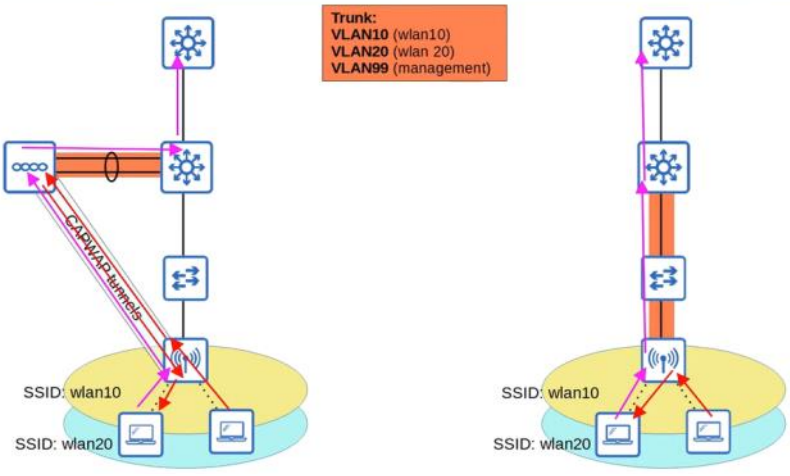
- The functions of an AP can be split between the AP and a **Wireless LAN Controller** (**WLC**).
- **Lightweight APs** handle 'real-time' operations like transmitting/receiving RF traffic, encryption/decryption of traffic, sending out beacons/probes, etc.
- Other functions are carried out by a WLC, for example RF management, security/QoS management, client authentication, client association/roaming management, etc.
- This is called **split-MAC architecture**.
- The WLC is also use to centrally configure the lightweight APs.
- The WLC can be located in the same subnet/VLAN as the lightweight APs it manages, or in a different subnet/VLAN.
- The WLC and the lightweight APs authenticate each other using digital certificates installed on each device (X.509 standard certificates).

Lightweight APs



- The WLC and lightweight APs use a protocol called CAPWAP (Control And Provisioning Of Wireless Access Points) to communicate.
  →Based on an older protocol called LWAPP (Lightweight Access Point Protocol).
- Two tunnels are created between each AP and the WLC:
  →Control tunnel (UDP port 5246). This tunnel is used to configure the APs, and control/manage the operations. All traffic in this tunnel is encrypted by default.
  →Data tunnel (UDP port 5247). All traffic from wireless clients is sent through this tunnel to the WLC. **It does not go directly to the wired network**.
  Traffic in this tunnel is not encrypted by default, but you can configure it to be encrypted with DTLS (Datagram Transport Layer Security).
- Because all traffic from wireless clients is tunneled to the WLC with CAPWAP, APs connect to switch access ports, not trunk ports.

Lightweight APs / Autonomous APs



- There are some key benefits to split-MAC archictecture, here are a few:

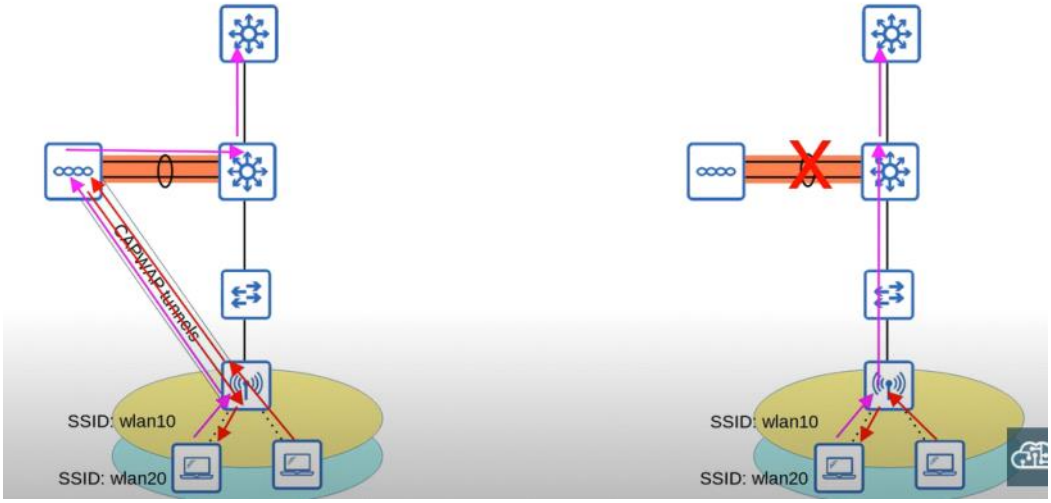
  →**Scalability**: With a WLC (or multiple in very large networks) it's much simpler to build and support a network with thousands of APs.
  →**Dynamic channel assignment**: The WLC can automatically select which channel each AP should use.
  →**Transmit power optimization**: The WLC can automatically set the appropriate transmit power for each AP.
  →**Self-healing wireless coverage**: When an AP stops functioning, the WLC can increase the transmit power of nearby APs to avoid coverage holes.
  →**Seamless roaming**: Clients can roam between APs with no noticeable delay.
  →**Client load balancing**: If a client is in range of two APs, the WLC can associate the client with the least-used AP, to balance the load among APs.
  →**Security/QoS management**: Central management of security and QoS policies ensures consistency across the network.

- Lightweight APs can be configured to operate in various modes:

  →**Local**: This is the default mode where the AP offers a BSS (more multiple BSSs) for clients to associate with.

  →**FlexConnect**: Like a lightweight AP in Local mode, it offers one or more BSSs for clients to associate with. However, FlexConnect allows the AP to locally switch traffic between the wired and wireless networks if the tunnels to the WLC go down.

FlexConnect



Lightweight APs

- Lightweight APs can be configured to operate in various modes:

→**Local**: This is the default mode where the AP offers a BSS (more multiple BSSs) for clients to associate with.

→**FlexConnect**: Like a lightweight AP in Local mode, it offers one or more BSSs for clients to associate with. However, FlexConnect allows the AP to locally switch traffic between the wired and wireless networks if the tunnels to the WLC go down.

→**Sniffer**: The AP does not offer a BSS for clients. It is dedicated to capturing 802.11 frames and sending them to a device running software such as Wireshark.

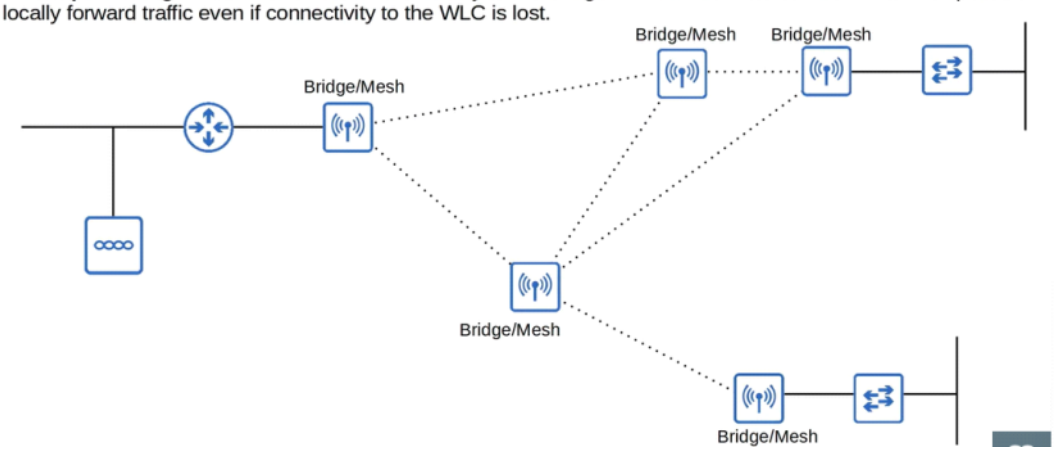
→**Monitor**: The AP does not offer a BSS for clients. It is dedicated to receiving 802.11 frames to detect rogue devices. If a client is found to be a rogue device, an AP can send de-authentication messages to disassociate the rogue device from the AP.

→**Rogue Detector**: The AP does not even use its radio. It listens to traffic on the wired network only, but it receives a list of suspected rogue clients and AP MAC addresses from the WLC. By listening to ARP messages on the wired network and correlating it with the information it receives from the WLC, it can detect rogue devices.

→**SE-Connect (Spectrum Expert Connect)**: The AP does not offer a BSS for clients. It is dedicated to RF spectrum analysis on all channels. It can send information to software such as Cisco Spectrum Expert on a PC to collect and analyze the data.

→**Bridge/Mesh**: Like the autonomous AP's *Outdoor Bridge*, the lightweight AP can be a dedicated bridge between sites, for example over long distances. A mesh can be made between the access points.

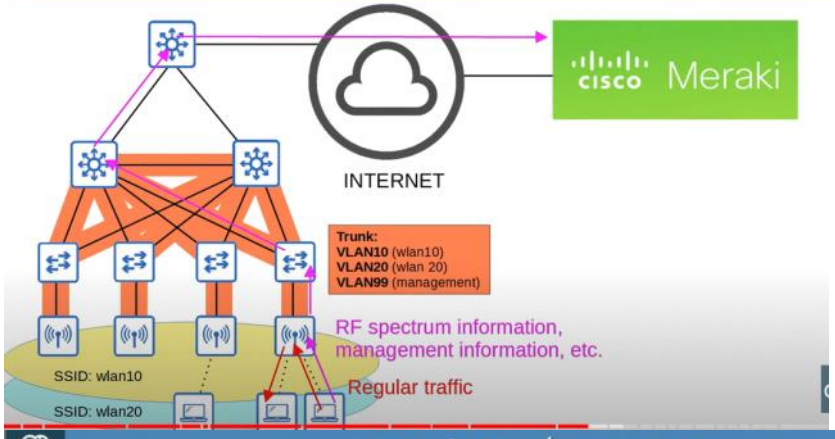
- →**Flex plus Bridge**: Adds FlexConnect functionality to the Bridge/Mesh mode. Allows wireless access points to locally forward traffic even if connectivity to the WLC is lost.



Cloud-based APs

- **Cloud-Based AP** architecture is in between autonomous AP and split-MAC architecture.
  → Autonomous APs that are centrally managed in the cloud.
- Cisco Meraki is a popular cloud-based Wi-Fi solution.
- The Meraki dashboard can be used to configure APs, monitor the network, generate performance reports, etc.
  → Meraki also tells each AP which channel to use, what transmit power, etc.
- However, data traffic is not sent to the cloud. It is sent directly to the wired network like when using autonomous APs.
  → Only management/control traffic is sent to the cloud.

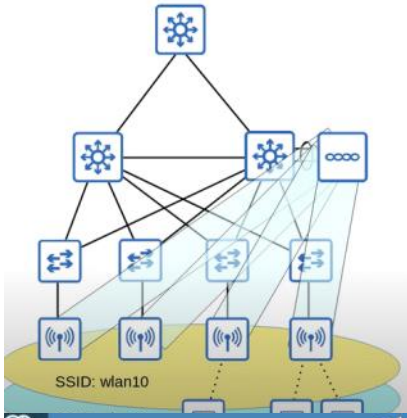
Autonomous APs



WLC Deployments

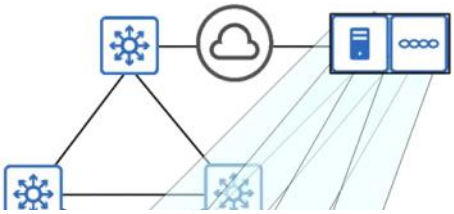
- In a split-MAC architecture, there are four main WLC deployment models:
  - → **Unified**: The WLC is a hardware appliance in a central location of the network.
  - → **Cloud-based**: The WLC is a VM running on a server, usually in a private cloud in a data center. This is not the same as the cloud-based AP architecture dicussed previously.
  - → **Embedded**: The WLC is integrated within a switch.
  - → **Mobility Express**: The WLC is integrated within an AP.

Unified WLC



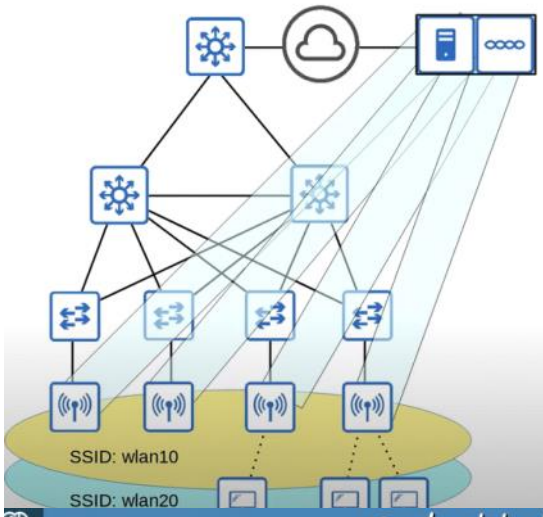
- The WLC is a hardware appliance deployed in a central location of the network.
- A unified WLC can support up to about 6000 APs.
- If more than 6000 APs are needed, additional WLCs can be added to the network.

Cloud-based WLC



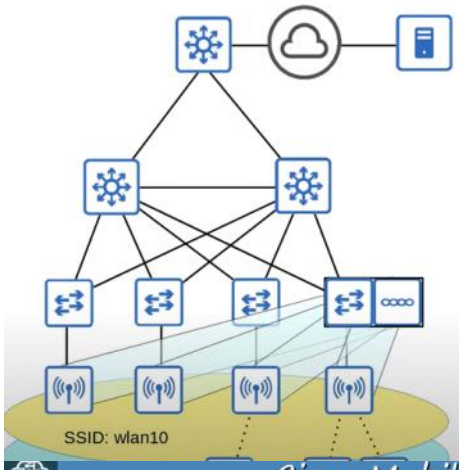
- The WLC is a VM running on a server, typically in a private cloud in a data center.
- Cloud-based WLCs can typically support up to about 3000 APs.
- If more than 3000 APs are needed, more WLC VMs can be deployed.

Cloud-based WLC



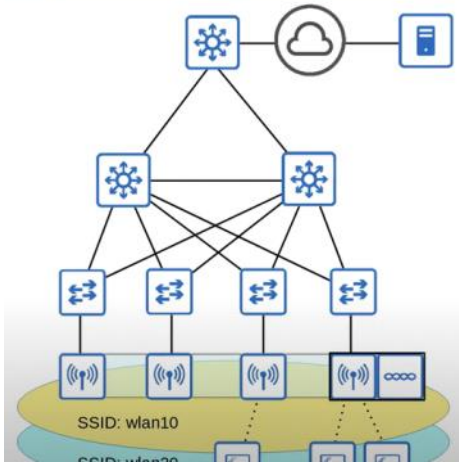
- The WLC is a VM running on a server, typically in a private cloud in a data center.
- Cloud-based WLCs can typically support up to about 3000 APs.
- If more than 3000 APs are needed, more WLC VMs can be deployed.

Embedded WLC



- The WLC is embedded within a switch.
- An embedded WLC can support up to about 200 APs.
- If more than 200 APs are needed, more switches with embedded WLCs can be added.

Cisco Mobility Express WLC



- The WLC is embedded within an AP.
- A Mobility Express WLC can support up to about 100 APs.
- If more than 100 APs are needed, more APs with embedded Mobility Express WLCs can be added.

PART 12

Tuesday, February 21, 2023 6:41 PM

Wireless Network Security

- Although security is important in all networks, it is even more essential in wireless networks.
- Because wireless signals are not contained within a wire, any device within range of the signal can receive the traffic.
- In wired networks, traffic is often only encrypted when sent over an untrusted network such as the Internet.
- In wireless networks, it is very important to encrypt traffic sent between the wireless clients and the AP.
- We will cover three main concepts:
  - → Authentication
  - → Encryption
  - → Integrity

Authentication

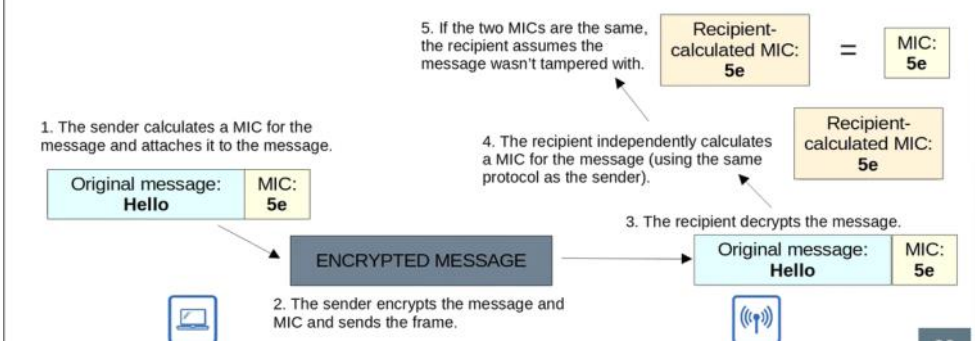
- All clients must be authenticated before they can associate with an AP.
- In a corporate setting, only trusted users/devices should be given access to the network.
  → In corporate settings, a separate SSID which doesn't have access to the corporate network can be provided for guest users.
- Ideally, clients should also authenticate the AP to avoid associating with a malicious AP.
- There are multiple ways to authenticate:
  - → Password
  - → Username/password
  - → Certificates

Encryption

- Traffic sent between clients and APs should be encrypted so that it can't be read by anyone except the AP and the client.
- There are many possible protocols that can be used to encrypt traffic.
- All devices on the WLAN will use the same protocol, however each client will use a unique encryption/decryption key so that other devices can't read its traffic.
- A 'group key' is used by the AP to encrypt traffic that it wants to send to all of its clients.
  → All of the clients associated with the AP keep that key so they can decrypt the traffic.

Integrity

- As explained in the 'Security Fundamentals' video of the course, Integrity ensures that a message is not modified by a third-party. The message that is sent by the source host should be the same as the message that is received by the destination host.
- A MIC (Message Integrity Check) is added to messages to help protect their integrity.



Authentication Methods

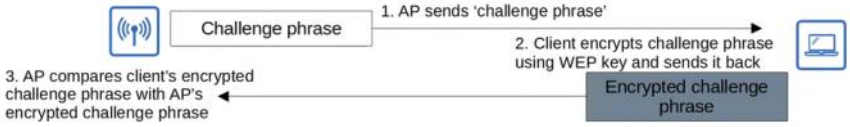
- **Open Authentication**
- **WEP (Wired Equivalent Privacy)**
- **EAP (Extensible Authentication Protocol)**
- **LEAP (Lightweight EAP)**
- **EAP-FAST (EAP Flexible Authentication via Secure Tunneling)**
- **PEAP (Protected EAP)**
- **EAP-TLS (EAP Transport Layer Security)**

Authentication Methods

The original 802.11 standard included two options for authentication:

- **Open Authentication**
  - → The client sends an authentication request, and the AP accepts it. No questions asked!
  - → This is clearly not a secure authentication method.
  - → After the client is authenticated and associated with the AP, it's possible to require the user to authenticate via other methods before access to the network is granted (ie. Starbuck's WiFi).

- **WEP (Wired Equivalent Privacy)**
  - → WEP is used to provide both authentication and encryption of wireless traffic.
  - → For encryption, WEP uses the RC4 algorithm.
  - → WEP is a 'shared-key' protocol, requiring the sender and receiver to have the same key.
  - → WEP keys can be 40 bits or 104 bits in length.
  - → The above keys are combined with a 24-bit 'IV' (Initialization Vector) to bring the total length to 64 bits or 128 bits.
  - → WEP encryption is **not secure** and can easily be cracked.
  - → WEP can be used for authentication like this:



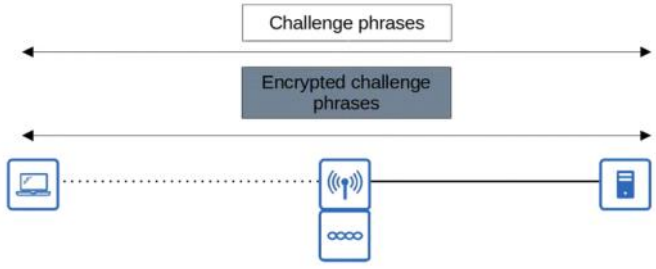
Authentication Methods

- **EAP (Extensible Authentication Protocol)**
  - → EAP is an authentication framework.
  - → It defines a standard set of authentication functions that are used by the various *EAP Methods*.
  - → We will look at four EAP methods: LEAP, EAP-FAST, PEAP, and EAP-TLS.
  - → EAP is integrated with 802.1X, which provides *port-based network access control.*

802.1x is used to limit network access for clients connected to a LAN or WLAN until they authenticate.
There are three main entities in 802.1X:
Supplicant: The device that wants to connect to the network.
Authenticator: The device that provides access to the network.
Authentication Server (AS): The device that receives client credentials and permits/denies access.

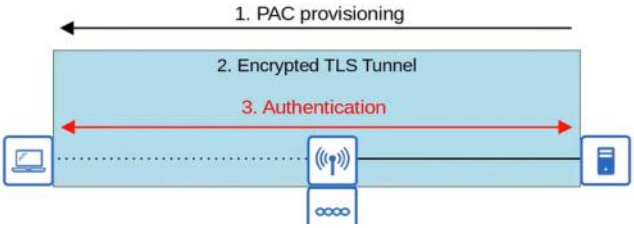
Authentication Methods

- **LEAP (Lightweight EAP)**
  - → LEAP was developed by Cisco as an improvement over WEP.
  - → Clients must provide a username and password to authenticate.
  - → In addition, *mutual authentication* is provided by both the client and server sending a challenge phrase to each other.
  - → *Dynamic* WEP keys are used, meaning that the WEP keys are changed frequently.
  - → Like WEP, LEAP is considered vulnerable and should not be used anymore.



Authentication Methods

- **EAP-FAST (EAP Flexible Authentication via Secure Tunneling)**
  → EAP-FAST was also developed by Cisco.
  → Consists of three phases:
  1) A PAC (Protected Access Credential) is generated and passed from the server to the client.
  2) A secure TLS tunnel is established between the client and authentication server.
  3) Inside of the secure (encrypted) TLS tunnel, the client and server communicate further to authenticate/authorize the client.

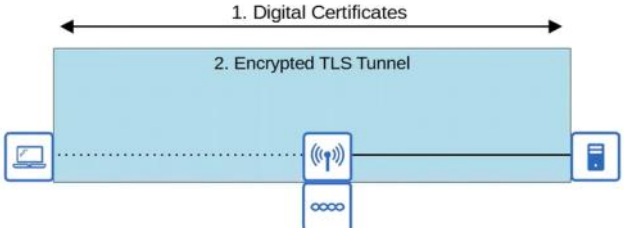


Authentication Methods

- **PEAP (Protected EAP)**
  → Like EAP-FAST, PEAP involves establishing a secure TLS tunnel between the client and server.
  → Instead of a PAC, the server has a digital certificate.
  → The client uses this digital certificate to authenticate the server.
  → The certificate is also used to establish a TLS tunnel.
  → Because only the server provides a certificate for authentication, the client must still be authenticated within the secure tunnel, for example by using MS-CHAP (Microsoft Challenge-Handshake Authentication Protocol)

Authentication Methods

- **EAP-TLS (EAP Transport Layer Security)**
  → Whereas PEAP only requires the AS to have a certificate, EAP-TLS requires a certificate on the AS and on every single client.
  → EAP-TLS is the most secure wireless authentication method, but it is more difficult to implement than PEAP because every client device needs a certificate.
  → Because the client and server authenticate each other with digital certificates, there is no need to authenticate the client within the TLS tunnel.
  → The TLS tunnel is still used to exchange encryption key information (encryption methods will be discussed next!).



Encryption and Integrity Methods

- **TKIP (Temporal Key Integrity Protocol)**
- **CCMP (Counter/CBC-MAC Protocol)**
- **GCMP (Galois/Counter Mode Protocol)**

Encryption and Integrity Methods

- **TKIP (Temporal Key Integrity Protocol)**
  → WEP was found to be vulnerable, but wireless hardware at the time was built to use WEP.
  → A temporary solution was needed until a new standard was created and new hardware was built.
  → TKIP adds various security features:
  -A **MIC** is added to protect the integrity of messages.
  -A **Key mixing algorithm** is used to create a unique WEP key for every frame.
  -The **initialization vector** is doubled in length from 24 bits to 48 bits, making brute-force attacks much more difficult.
  -The MIC includes the **sender MAC address** to identify the frame's sender.
  -A **timestamp** is added to the MIC to prevent replay attacks. Replay attacks involve re-resending a frame that has already been transmitted.
  -A **TKIP sequence number** is used to keep track of frames sent from each source MAC address. This also protects against replay attacks.

'TKIP is used in WPA version 1,

- **CCMP (Counter/CBC-MAC Protocol)**
  → CCMP was developed after TKIP and is more secure.
  → It is used in WPA2.
  → To use CCMP, it must be supported by the device's hardware. Old hardware built only to use WEP/TKIP cannot use CCMP.
  → CCMP consists of two different algorithms to provide encryption and MIC.
  1) **AES (Advanced Encryption Standard) counter mode** encryption
  -AES is the most secure encryption protocol currently available. It is widely used all over the world.
  -There are multiple modes of operation for AES. CCMP uses 'counter mode'.

  2) **CBC-MAC (Cipher Block Chaining Message Authentication Code)** is used as a MIC to ensure the integrity of messages.

- **GCMP (Galois/Counter Mode Protocol)**
  → GCMP is more secure and efficient than CCMP.
  → Its increased efficiency allows higher data throughput than CCMP.
  → It is used in WPA3.
  → GCMP consists of two algorithms:
  1) **AES counter mode** encryption

  2) **GMAC (Galois Message Authentication Code)** is used as a MIC to ensure the integrity of messages.

- **TKIP (Temporal Key Integrity Protocol)**
  -Based on WEP, but more secure.
  -Should not be used in modern networks.
  -WPA

- **CCMP (Counter/CBC-MAC Protocol)**
  -AES counter mode for encryption
  -CBC-MAC for MIC
  -WPA2

- **GCMP (Galois/Counter Mode Protocol)**
  -AES counter mode for encryption
  -GMAC for MIC
  -WPA3

WiFi Protected Access

- The Wi-Fi alliance has developed three WPA certifications for wireless devices:
  → WPA
  → WPA2
  → WPA3

- To be WPA-certified, equipment must be tested in authorized testing labs.

- All of the above support two authentication modes:

  → **Personal mode**: A pre-shared key (PSK) is used for authentication. When you connect to a home Wi-Fi network, enter the password and are authenticated, that is **personal** mode. This is common in small networks.
  *The PSK itself is not sent over the air. A four-way handshake is used for authentication, and the PSK is used to generate encryption keys.

  → **Enterprise mode**: 802.1X is used with an authentication server (RADIUS server).
  *No specific EAP method is specified, so all are supported (PEAP, EAP-TLS, etc).

WiFi Protected Access

- The **WPA** certification was developed after WEP was proven to be vulnerable and includes the following protocols:
  → TKIP (based on WEP) provides encryption/MIC.
  → 802.1X authentication (Enterprise mode) or PSK (Personal mode)

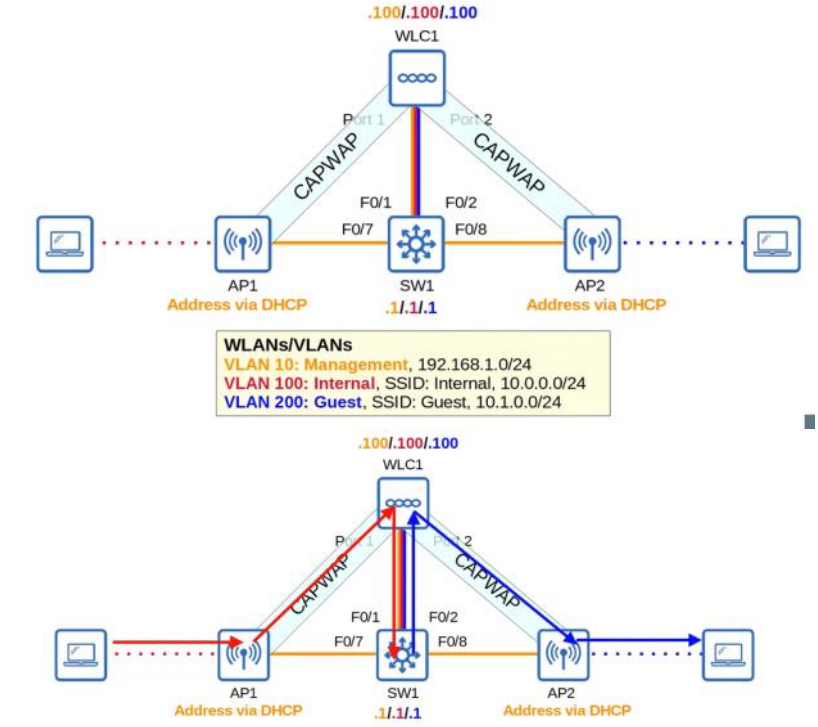
- **WPA2** was released in 2004 and includes the following protocols:
  → CCMP provides encryption/MIC.
  → 802.1X authentication (Enterprise mode) or PSK (Personal mode)

- **WPA3** was released in 2018 and includes the following protocols:
  → GCMP provides encryption/MIC.
  → 802.1X authentication (Enterprise mode) or PSK (Personal mode)
  → WPA3 also provides several additional security features, for example:
  -**PMF (Protected Management Frames)**, protecting 802.11 management frames from eavesdropping/forging.
  -**SAE (Simultaneous Authentication of Equals)** protects the four-way handshake when using personal mode authentication.

-**Forward secrecy** prevents data from being decrypted after it has been transmitted over the air. So, an attacker can't capture wireless frames and then try to decrypt them later.

PMF IS OPTIONAL IN WPA2 BUT MANDATORY IN WPA3

Network Topology




```
SW1(config)#vlan 10
SW1(config-vlan)#name Management
SW1(config-vlan)#vlan 100
SW1(config-vlan)#name Internal
SW1(config-vlan)#vlan 200
SW1(config-vlan)#name Guest
```

I included F0/6 because I will connect my PC to F0/6 to gain access to WLC1's GUI.

```
SW1(config)#int range f0/6 - 8
SW1(config-if-range)#switchport mode access
SW1(config-if-range)#switchport access vlan 10
SW1(config-if-range)#spanning-tree portfast
```

Remember that WLCs only support static LAG, no PAgP or LACP.

```
SW1(config-if-range)#interface range f0/1 - 2
SW1(config-if-range)#channel-group 1 mode on
```

```
SW1(config-if-range)#interface port-channel 1
SW1(config-if)#switchport mode trunk
SW1(config-if)#switchport trunk allowed vlan 10,100,200
```

```
SW1(config)#interface vlan 10
SW1(config-if)#ip address 192.168.1.1 255.255.255.0
SW1(config-if)#interface vlan 100
SW1(config-if)#ip address 10.0.0.1 255.255.255.0
SW1(config-if)#interface vlan 200
SW1(config-if)#ip address 10.1.0.1 255.255.255.0
```

```
SW1(config)#interface vlan 10
SW1(config-if)#ip address 192.168.1.1 255.255.255.0
SW1(config-if)#interface vlan 100
SW1(config-if)#ip address 10.0.0.1 255.255.255.0
SW1(config-if)#interface vlan 200
SW1(config-if)#ip address 10.1.0.1 255.255.255.0

SW1(config)#ip dhcp pool VLAN10
SW1(dhcp-config)#network 192.168.1.0 255.255.255.0
SW1(dhcp-config)#default-router 192.168.1.1
SW1(dhcp-config)#option 43 ip 192.168.1.100

SW1(config)#ip dhcp pool VLAN100
SW1(dhcp-config)#network 10.0.0.0 255.255.255.0
SW1(dhcp-config)#default-router 10.0.0.1

SW1(config)#ip dhcp pool VLAN200
SW1(dhcp-config)#network 10.1.0.0 255.255.255.0
SW1(dhcp-config)#default-router 10.1.0.1

SW1(config)#ntp master
```

Option 43 can be used to tell the APs the IP address of their WLC.
*this is not necessary in this case because the APs and WLC are in the same subnet. The WLC will hear the APs broadcast CAPWAP *discovery* messages.

WLC Initial Setup

```
Welcome to the Cisco Wizard Configuration Tool
Use the '-' character to backup

Would you like to terminate autoinstall? [yes]:

System Name [Cisco_10:65:64] (31 characters max): WLC1
Enter Administrative User Name (24 characters max): admin
Enter Administrative Password (3 to 24 characters): *********
Re-enter Administrative Password                : *********

Enable Link Aggregation (LAG) [yes][NO]: yes

Management Interface IP Address: 192.168.1.100
Management Interface Netmask: 255.255.255.0
Management Interface Default Router: 192.168.1.1
Management Interface VLAN Identifier (0 = untagged): 10
Management Interface DHCP Server IP Address: 192.168.1.1
```

WLC Initial Setup

```
Virtual Gateway IP Address: 172.16.1.1

Multicast IP Address: 239.239.239.239

Mobility/RF Group Name: jITlab

Network Name (SSID): Internal

Configure DHCP Bridging Mode [yes][NO]: no

Allow Static IP Addresses [YES][no]: yes

Configure a RADIUS Server now? [YES][no]: no
Warning! The default WLAN security policy requires a RADIUS server.
Please see documentation for more details.

Enter Country Code list (enter 'help' for a list of countries) [US]: FR
```

WLC Initial Setup

AIR-CAP3502I-E-K9

- -E is the *regulatory domain* of the device.
- -E indicates Europe.
- If the regulatory domain of the country specified in the WLC configuration doesn't match the regulatory domain of the AP, the AP won't be able to join the WLC.
- https://www.cisco.com/c/dam/assets/prod/wireless/wireless-compliance-tool/index.html to check the regulatory domain of each country.

WLC Initial Setup

```
Enable 802.11b Network [YES][no]:
Enable 802.11a Network [YES][no]:
Enable 802.11g Network [YES][no]:
Enable Auto-RF [YES][no]:

Configure a NTP server now? [YES][no]: yes
Enter the NTP server's IP address: 192.168.1.1
Enter a polling interval between 3600 and 604800 secs: 3600

Configuration correct? If yes, system will save it and reset. [yes][NO]:
yes

Configuration saved!
Resetting system with new configuration...
```

Now if we connect to the f0/6 port of the switch from the pc we can enter https://192.168.1.100 to go to wlc login page

WLC Ports/Interfaces

- WLC **ports** are the physical ports that cables connect to.
- WLC **interfaces** are the logical interfaces within the WLC (ie. SVIs on a switch).
- WLCs have a few different kinds of **ports**:

→**Service port**: A dedicated management port. Used for out-of-band management. Must connect to a switch access port because it only supports one VLAN. This port can be used to connect to the device while it is booting, perform system recovery, etc.

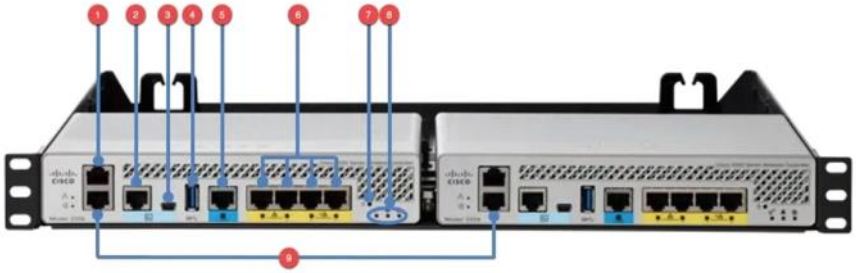
→**Distribution system port**: These are the standard network ports that connect to the 'distribution system' (wired network) and are used for data traffic. These ports usually connect to switch trunk ports, and if multiple distribution ports are used they can form a LAG.

→**Console port**: This is a standard console port, either RJ45 or USB.

→**Redundancy port**: This port is used to connect to another WLC to form a high availability (HA) pair.

WLC Ports/Interfaces

WLC Ports/Interfaces



1) Service port
2) Console port (RJ45)
3) Console port (USB)
4) USB (for software updates)
5) Distribution system port (multi-gigabit)
6) Distribution system ports (1-gig)
7) Reset button
8) Status LEDs
9) Redundancy port

WLC Ports/Interfaces

- WLCs have a few different kinds of **interfaces**:

→**Management interface**: Used for management traffic such as Telnet, SSH, HTTP, HTTPS, RADIUS authentication, NTP, Syslog, etc. CAPWAP tunnels are also formed to/from the WLC's management interface.

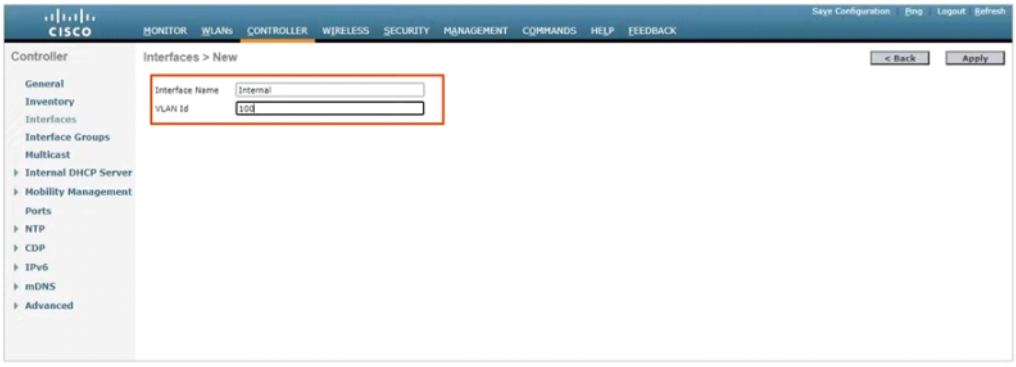
→**Redundancy management interface**: When two WLCs are connected by their redundancy ports, one WLC is 'active' and the other is 'standby'. This interface can be used to connect to and manage the 'standby' WLC.

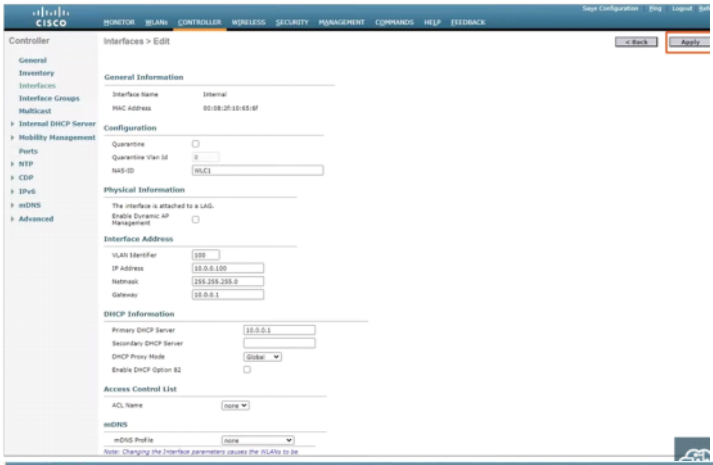
→**Virtual interface**: This interface is used when communicating with wireless clients to relay DHCP requests, perform client web authentication, etc.

→**Service port interface**: If the service port is used, this interface is bound to it and used for out-of-band management.

→**Dynamic interface**: These are the interfaces used to map a WLAN to a VLAN. For example, traffic from the 'Internal' WLAN will be sent to the wired network from the WLC's 'Internal' dynamic interface.

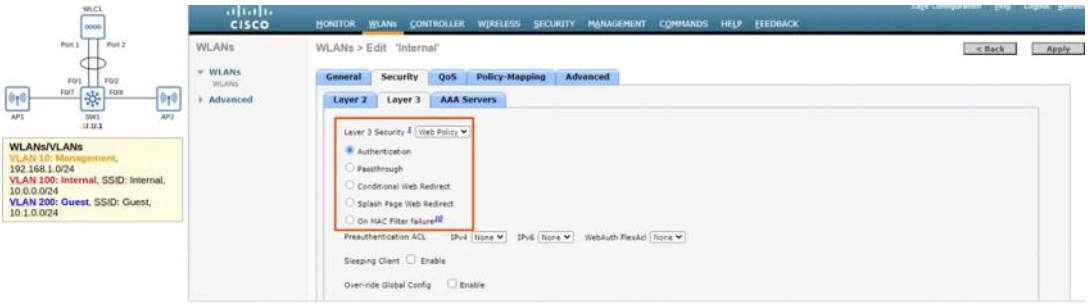
WLC:





In the security setting sof WLAN config we have to use WPA+WPA2 for using preshared key for authentication and authentication key maagemrnt enable PSK

PSK format as ASCII and give the pre shared key in range 8-63 chars in length



- **Web Authentication**: After the wireless clients gets an IP address and tries to access a web page, they will have to enter a username and password to authenticate.
- **Web Passthrough**: Similar to the above, but no username or password are required. A warning or statement is displayed and the client simply has to agree to gain access to the Internet.
- The **Conditional** and **Splash Page** web redirect options are similar, but additionally require 802.1X layer 2 authentication

QOS for the WLAN will be in Platinum(voice),Gold(Video),Silver(best effort),bronze background)

We can enable management via wireless and change configuration settign using connecting to that AP by device
ACL can be applied also(with DSCP etc..)

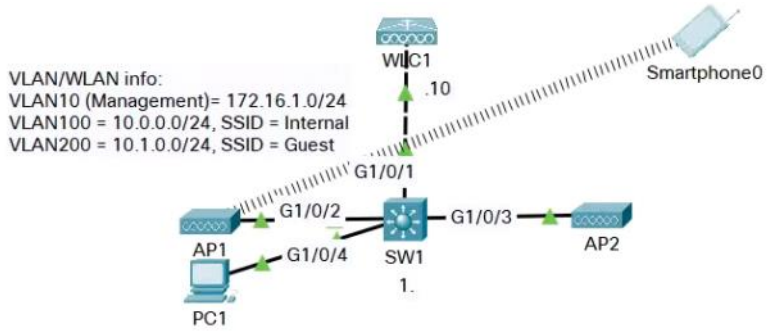
CPU ACLs are used to limit access to the CPU of the WLC. This limits which devices will be able to connect to the WLC via Telnet/SSH, HTTP/HTTPS, retrieve SNMP information from the WLC, etc.

you to configure various wireless network settings. In this scenario, you want to create a normal WLAN named **MyCompanyLAN**. To create a new normal WLAN, you should complete four steps on the **WLANs** > **New** page of the WLC GUI:

1. Select the type of WLAN you are creating from the **Type** drop-down list box; by default, this value is configured to **WLAN**.
2. Enter a 32-character or less profile name in the **Profile Name** field.
3. Enter a 32-character or less Service Set Identifier (SSID) in the **SSID** field.
4. Choose a WLAN ID from the **ID** drop-down list box.

There are three types of WLANs you can create by using the WLC GUI:

1. A normal WLAN, which is the WLAN to which wireless clients inside your company's walls will connect
2. A Guest LAN, which is the WLAN to which guest wireless clients inside your company's walls will connect
3. A Remote LAN, which is the WLAN configuration for wired ports on the WLC



1. Use the web browser on PC1 to access the GUI of WLC1.
Username: admin
Password: Cisco123
*you must use HTTPS

2. Spend some time familiarizing yourself with the WLC GUI.
What information can be viewed from each tab?
What is the current state of the network?

3. Configure dynamic interfaces for the Internal & Guest WLANs.

4. Create the Internal & Guest WLANs using WPA2+PSK.

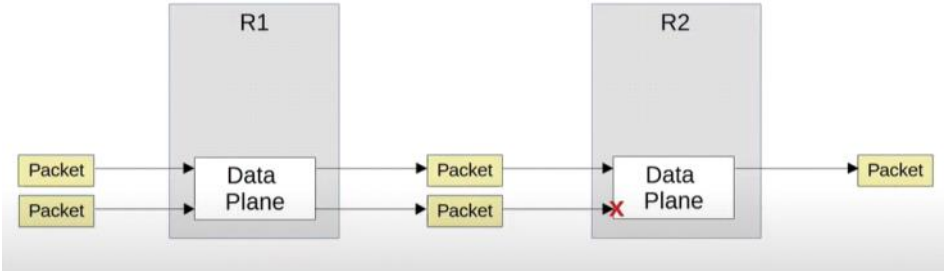
5. Add a wireless client to the network and associate it with an AP.

The various functions of network devices can be logically divided up (categorized) into *planes*:

- Data plane
- Control plane
- Management plane

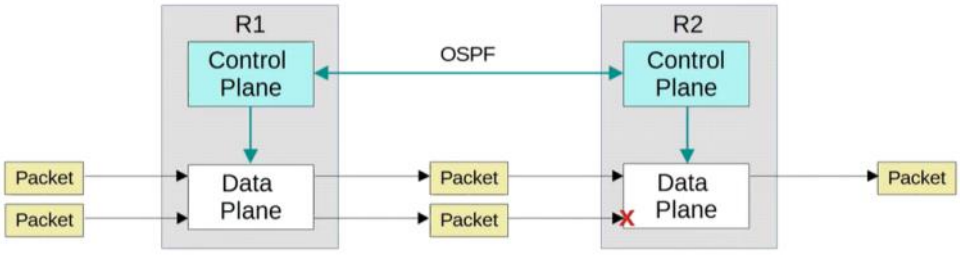
Data Plane

- All tasks involved in forwarding user data/traffic from one interface to another are part of the **data plane**.
- A router receives a message, looks for the most specific matching route in its routing table, and forwards it out of the appropriate interface to the next hop.
  → It also de-encapsulates the original Layer 2 header, and re-encapsulates with a new header destined for the next hop's MAC address.
- A switch receives a message, looks at the destination MAC address, and forwards it out of the appropriate interface (or floods it).
  → This includes functions like adding or removing 802.1q VLAN tags.
- NAT (changing the src/dst addresses before forwarding) is part of the data plane.
- Deciding to forward or discard messages due to ACLs, port security, etc. is part of the data plane.
- The data plane is also called the 'forwarding plane'.



Control Plane

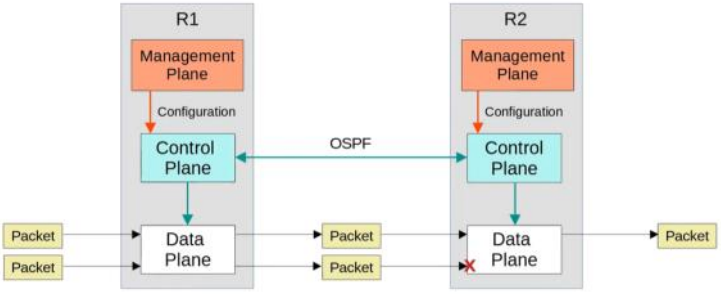
- How does a device's data plane make its forwarding decisions?
  → routing table, MAC address table, ARP table, STP, etc.
- Functions that build these tables (and other functions that influence the data plane) are part of the **control plane**.
- The control plane *controls* what the data plane does, for example by building the router's routing table.
- The control plane performs *overhead* work.
  → OSPF itself doesn't forward user data packets, but it informs the data plane about how packets should be forwarded.
  → STP itself isn't directly involved in the process of forwarding frames, but it informs the data plane about which interfaces should and shouldn't be used to forward frames.
  → ARP messages aren't user data, but they are used to build an ARP table which is used in the process of forwarding data.



In traditional networking, the data plane and control plane are both distributed. Each device has its own data plane and its own control plane. The planes are 'distributed' throughout the network.

Management Plane

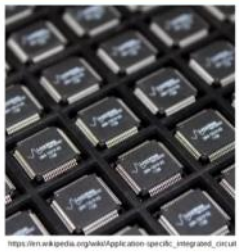
- Like the control plane, the **management plane** performs overhead work.
  → However, the management plane doesn't directly affect the forwarding of messages in the data plane.
- The management plane consists of protocols that are used to manage devices.
  → SSH/Telnet, used to connect to the CLI of a device to configure/manage it.
  → Syslog, used to keep logs of events that occur on the device.
  → SNMP, used to monitor the operations of the device.
  → NTP, used to maintain accurate time on the device.



The Data plane is the reason we buy routers and switches (and network infrastructure in general), to forward messages. However, the Control plane and Management plane are both necessary to enable the data plane to do its job.

Logical Planes

- The operations of the Management plane and Control plane are usually managed by the CPU.
- However, this is not desirable for data plane operations because CPU processing is slow (relatively speaking).
- Instead, a specialized hardware ASIC (Application-Specific Integrated Circuit) is used. ASICs are chips built for specific purposes.
- Using a switch as an example:
  → When a frame is received, the ASIC (not the CPU) is responsible for the switching logic.
  → The MAC address table is stored in a kind of memory called TCAM (Ternary Content-Addressable Memory).
  *Another common name for the MAC address table is *CAM table*.
  → The ASIC feeds the destination MAC address of the frame into the TCAM, which returns the matching MAC address table entry.
  → The frame is then forwarded out of the appropriate interface.
- Modern routers also use a similar hardware data plane: an ASIC designed for forwarding logic, and tables stored in TCAM.

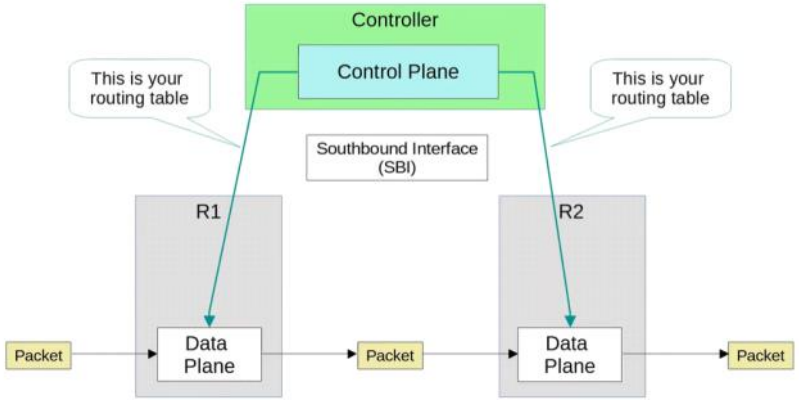


A simple summary:
- When a device receives control/ management traffic (destined for itself), it will be processed in the CPU.
- When a device receives data traffic which should pass through the device, it is processed by the ASIC for maximum speed.

Software-Defined Networking

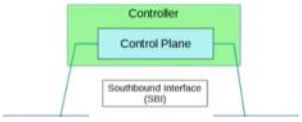
- **Software-Defined Networking (SDN)** is an approach to networking that centralizes the control plane into an application called a *controller.*
  → You are already familiar with this concept from learning about Wireless LAN Controllers.
- SDN is also called **Software-Defined Architecture (SDA)** or **Controller-Based Networking**.
- Traditional control planes use a distributed architecture.
  → For example, each router in the network runs OSPF and the routers share routing information and then calculate their preferred routes to each destination.
- An SDN controller centralizes control plane functions like calculating routes.
  → That is just an example, and how much of the control plane is centralized varies greatly.
- The controller can interact programmatically with the network devices using APIs (Application Programming Interface).

Software-Defined Networking



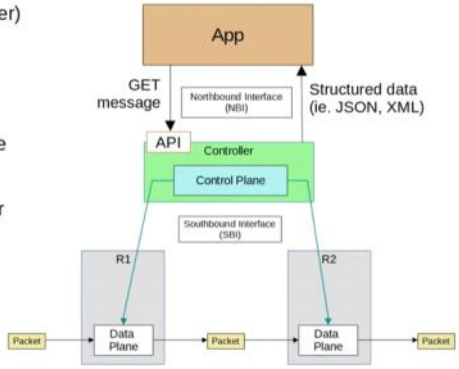
Southbound Interface (SBI)

- The SBI is used for communications between the controller and the network devices it controls.
- It typically consists of a communication protocol and API (Application Programming Interface).
- APIs facilitate data exchanges between programs.
  - → Data is exchanged between the controller and the network devices.
  - → An API on the network devices allows the controller to access information on the devices, control their data plane tables, etc.
- Some examples of SBIs:
  - → OpenFlow
  - → Cisco OpFlex
  - → Cisco onePK (Open Network Environment Platform Kit)
  - → NETCONF



Northbound Interface (NBI)

- Using the SBI, the controller communicates with the managed devices and gathers information about them:
  - → The devices in the network
  - → The topology (how the devices are connected together)
  - → The available interfaces on each device
  - → Their configurations
- The **Northbound Interface (NBI)** is what allows us to interact with the controller, access the data it gathers about the network, program it, and make changes in the network via the SBI.
- A *REST API* is used on the controller as an interface for apps to interact with it.
  - → REST = Representational State Transfer
- Data is sent in a structured (*serialized*) format such as JSON or XML.



Automation in Traditional Networks vs SDN

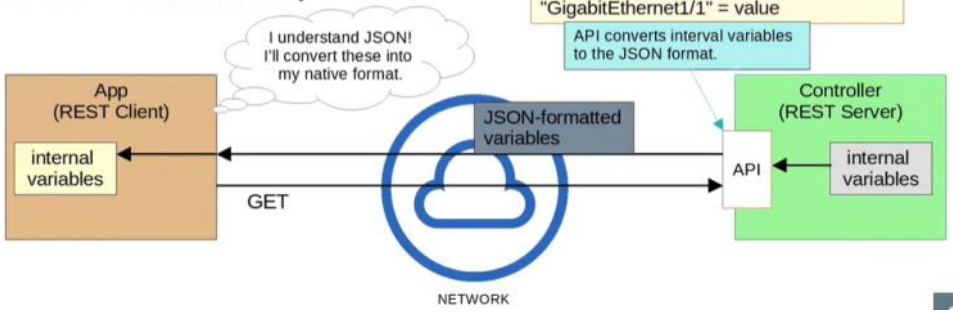
- Networking tasks can be automated in traditional networks:
  - → Scripts can be written (ie. using Python) to push commands to many devices at once.
  - → Python with good use of Regular Expressions can parse through **show** commands to gather information about the network devices.
- However, the robust and centralized data collected by SDN controllers greatly facilitates these functions.
  - → The controller collects information about all devices in the network.
  - → Northbound APIs allow apps to access information in a format that is easy for programs to understand (ie. JSON, XML).
  - → The centralized data facilitates network-wide analytics.
- SDN tools can provide the benefits of automation without the requirement of third-party scripts & apps.
  - → You don't need expertise in automation to make use of SDN tools.
  - → However, APIs allow third-party applications to interact with the controller, which can be very powerful.

Although SDN and automation aren't the same thing, the SDN architecture greatly facilitates the automation of various tasks in the network via the SDN controller and APIs.

Data serialization languages allow us to represent *variables* with text.

```
{"interface_name": "GigabitEthernet1/1",
 "status": "up",
 "ip_address": "192.168.1.1",
 "netmask": "255.255.255.0"}
```

Variables are containers that store values.
"interface_name" = container
"GigabitEthernet1/1" = value



JSON

- **JSON** (JavaScript Object Notation) is an open standard **file format** and **data interchange format** that uses human-readable text to store and transmit data objects.
- It is standardized in RFC 8259 (https://datatracker.ietf.org/doc/html/rfc8259).

Take the time to read it!

- It was derived from JavaScript, but it is language-independent and many modern programming languages are able to generate and read JSON data.
  → REST APIs often use JSON.
- *Whitespace* is insignificant.
- JSON can represent four 'primitive' data types:
  → string
  → number
  → boolean
  → null
- JSON also has two 'structured' data types:
  → object
  → array

JSON

```
R1#show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0         192.168.1.1     YES manual up                    up
GigabitEthernet0/1         unassigned      YES unset  administratively down down
```

```
{
    "ip_interfaces": [
            {
            "Interface":"GigabitEthernet0/0",
            "IP-Address": "192.168.1.1",
            "OK?": "YES",
            "Method": "manual",
            "Status": "up",
            "Protocol": "up"
            },
            {
            "Interface":"GigabitEthernet0/1",
            "IP-Address": "unassigned",
            "OK?": "YES",
            "Method": "unset",
            "Status": "administratively down",
            "Protocol": "down"
            }
    ]
}
```

XML

- **XML** (Extensible Markup Language) was developed as a markup language, but is now used as a general data serialization language.
  → markup languages (ie. HTML) are used to format text (font, size, color, headings, etc.)
- XML is generally less human-readable than JSON.
- Whitespace is insignificant.
- Often used by REST APIs.
- <key>value</key>

```
R1#show ip interface brief | format
<?xml version="1.0" encoding="UTF-8"?>
  <ShowIpInterfaceBrief xmlns="ODM://built-in//show_ip_interface_brief">
    <SpecVersion>built-in</SpecVersion>
    <IPInterfaces>
      <entry>
        <Interface>GigabitEthernet0/0</Interface>
        <IP-Address>192.168.1.1</IP-Address>
        <OK>YES</OK>
        <Method>manual</Method>
        <Status>up</Status>
        <Protocol>up</Protocol>
      </entry>
      <entry>
        <Interface>GigabitEthernet0/1</Interface>
        <OK>YES</OK>
        <Method>unset</Method>
        <Status>administratively down</Status>
        <Protocol>down</Protocol>
      </entry>
    </IPInterfaces>
  </ShowIpInterfaceBrief>
```

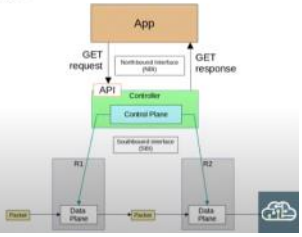
YAML

- **YAML** originally meant *Yet Another Markup Language*, but to distinguish its purpose as a data-serialization language rather than a markup language, it was repurposed to *YAML Aint Markup Language*.
- YAML is used by the network automation tool Ansible (we'll cover that later!).
- YAML is very human-readable.
- Whitespace **is significant** (unlike JSON and XML).
  → Indentation is very important.
- YAML files start with --- .
- - is used to indicate a list.
- Keys and values are represented as key:value.

```
---
ip_interfaces:
- Interface: GigabitEthernet0/0
  IP-Address: 192.168.1.1
  OK?: 'YES'
  Method: manual
  Status: up
  Protocol: up
- Interface: GigabitEthernet0/1
  IP-Address: unassigned
  OK?: 'YES'
  Method: unset
  Status: administratively down
  Protocol: down
```

APIs

- An API (Application Programming Interface) is a software interface that allows two applications to communicate with each other.
- APIs are essential not just for network automation, but for all kinds of applications.
- In SDN architecture, APIs are used to communicate between apps and the SDN controller (via the NBI), and between the SDN controller and the network devices (via the SBI).
- The NBI typically uses REST APIs.
- NETCONF and RESTCONF are popular southbound APIs.



Purpose	CRUD Operation	HTTP Verb
Create new variable	**Create**	**POST**
Retrieve value of variable	**Read**	**GET**
Change the value of variable	**Update**	**PUT, PATCH**
Delete variable	**Delete**	**DELETE**

HTTP Request

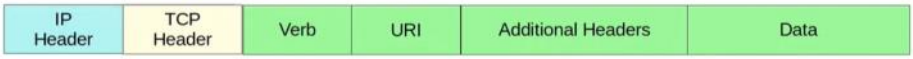
- When an HTTP client sends a request to an HTTP server, the HTTP header includes information like this:
  → An HTTP Verb (ie. GET)
  → A URI (Uniform Resource Identifier), indicating the resource it is trying to access.

- Here's an example of a URI (which we will use in the demonstration later):

https://sandboxdnac.cisco.com/dna/intent/api/v1/network-device
scheme authority path

HTTP Request

- The HTTP request can include additional headers which pass additional information to the server.
  → Check the list at https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers



- An example would be an **Accept** header, which informs the server about the type(s) of data that can be sent back to the client.
  → ie. **`Accept: application/json`** or **`Accept: application/xml`**

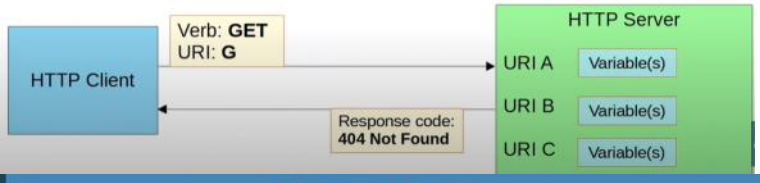
- You can also view standard HTTP header fields with some examples at https://en.wikipedia.org/wiki/List_of_HTTP_header_fields

- When a REST client makes an API call (request) to a REST server, it will send an HTTP request like the one above.
  *REST APIs don't *have* to use HTTP for communication, although HTTP is the most common choice.

HTTP Response

- The server's response will include a status code indicating if the request succeeded or failed, as well other details.

- The first digit indicates the class of the response:
  → **1xx** *informational* – the request was received, continuing process
  → **2xx** *successful* – the request was successfully received, understood, and accepted
  → **3xx** *redirection* – further action needs to be taken in order to complete the request
  → **4xx** *client error* – the request contains bad syntax or cannot be fulfilled



HTTP Response

Here are some examples of each HTTP Response class:

- **1xx** *Informational*
  → **102 Processing** indicates that the server has received the request and is processing it, but the response is not yet available.

- **2xx** *Successful*
  → **200 OK** indicates that the request succeeded.
  → **201 Created** indicates that the request succeeded and a new resource was created (ie. in response to POST)

- **3xx** *Redirection*
  → **301 Moved Permanently** indicates that the requested resource has been moved, and the server indicates its new location.

- **4xx** *Client Error*
  → **403 Unauthorized** means the client must authenticate to get a response.
  → **404 Not Found** means the requested resource was not found.

- **5xx** *Server Error*

REST

- REST stands for Representational State Transfer.

- **REST APIs** are also known as **REST-based APIs** or **RESTful APIs**.
  → REST isn't a specific API. Instead, it describes a set of rules about how the API should work.

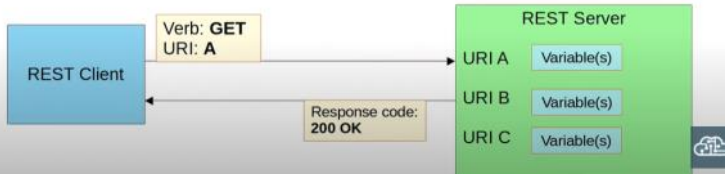
- The six constraints of RESTful architecture are:
  → Uniform Interface
  → Client-server
  → Stateless
  → Cacheable or non-cacheable
  → Layered system
  → Code-on-demand (optional)

REST: Client-server

- **REST APIs** use a client-server architecture.

- The client uses API calls (HTTP requests) to access the resources on the server.

- The separation between the client and server means they can both change and evolve independently of each other.
  → When the client application changes or the server application changes, the interface between them must not break.



REST: Stateless

- **REST APIs** exchanges are stateless.
- This means that each API exchange is a separate event, independent of all past exchanges between the client and server.
  → The server does not store information about previous requests from the client to determine how it should respond to new requests.
- If authentication is required, this means that the client must authenticate with the server for each request it makes.
- TCP is an example of a stateful protocol.
- UDP is an example of a stateless protocol.

*Although REST APIs use HTTP, which uses TCP (stateful) as its Layer 4 protocol, HTTP and REST APIs themselves aren't stateful. The functions of each layer are separate!

REST: Cacheable or Non-Cacheable

- **REST APIs** must support caching of data.
- *Caching* refers to storing data for future use.
  → For example, your computer might cache many elements of a web page so that it doesn't have to retrieve the entire page every time you visit it.
  → This improves performance for the client and reduces the load on the server.
- Not all resources have to be cacheable, but cacheable resources MUST be declared as cacheable.

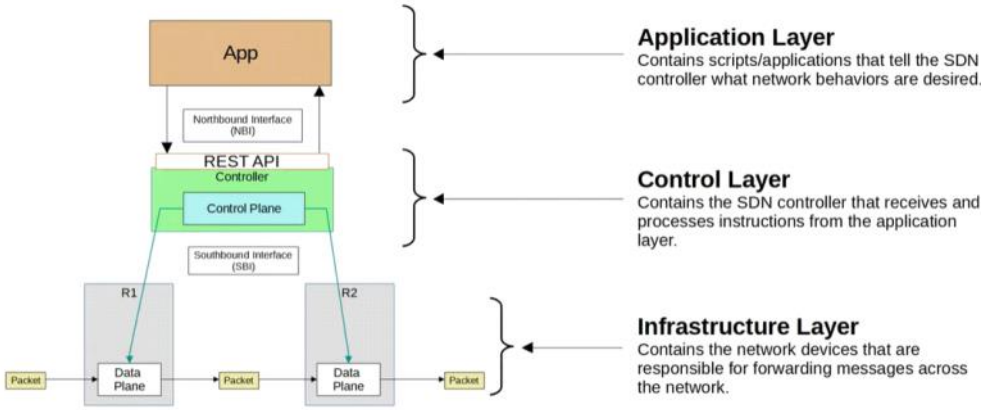
Cisco DevNet

- "Cisco DevNet is Cisco's developer program to help developers and IT professionals who want to write applications and develop integrations with Cisco products, platforms, and APIs."
- DevNet offers lots of free resources such as courses, tutorials, labs, sandboxes, documentation, etc. to learn about automation and develop your skills.
- There is also a DevNet certification track that you can pursue if you're interested in automation.
- We will use their Cisco **DNA Center** Sandbox to send a REST API call using **Postman**.
  → DNA Center is one of Cisco's SDN controllers. We will cover it in more detail in the next video!
  → Postman is a platform for building and using APIs.
- To start:
  → Make an account on developer.cisco.com
  → Make an account on postman.com + download the desktop app (https://www.postman.com/downloads/).

https://developer.cisco.com/docs/dna-center/#!getting-started

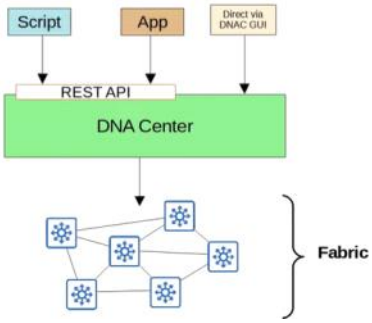
SDN Review

- **Software-Defined Networking (SDN)** is an approach to networking that centralizes the control plane into an application called a *controller.*
- Traditional control planes use a distributed architecture.
- An SDN controller centralizes control plane functions like calculating routes.
- The controller can interact programmatically with the network devices using APIs.
- The **SBI** is used for communications between the controller and the network devices it controls.
- The **NBI** is what allows us to interact with the controller with our scripts and applications.



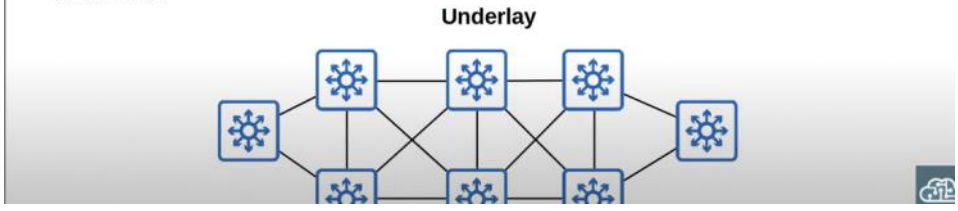
SD-Access

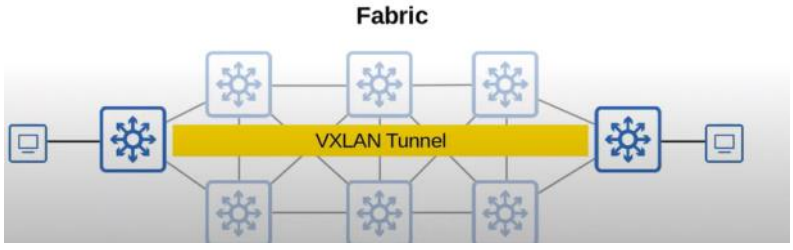
- Cisco **SD-Access** is Cisco's SDN solution for automating campus LANs.
  → ACI (Application Centric Infrastructure) is their SDN solution for automating data center networks.
  → SD-WAN is their SDN solution for automating WANs.
- Cisco **DNA (Digital Network Architecture) Center** is the controller at the center of SD-Access.



SD-Access

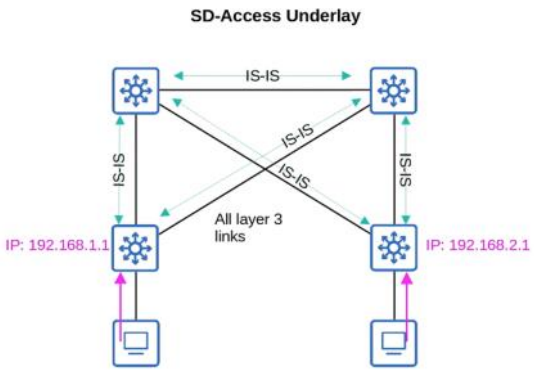
- The **underlay** is the underlying physical network of devices and connections (including wired and wireless) which provide IP connectivity (ie. using IS-IS).
  → Multilayer switches and their connections.
- The **overlay** is the virtual network built on top of the physical underlay network.
  → SD-Access uses VXLAN (Virtual Extensible LAN) to build tunnels.
- The **fabric** is the combination of the overlay and underlay; the physical and virtual network as a whole.

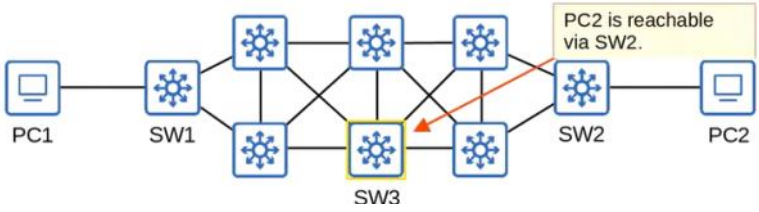


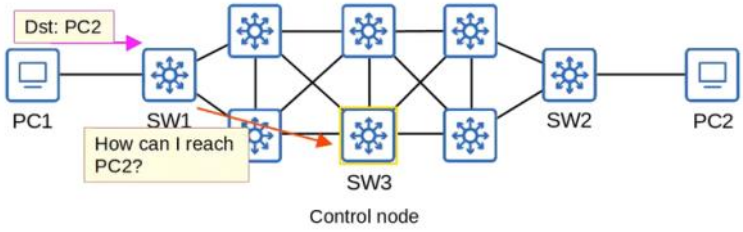


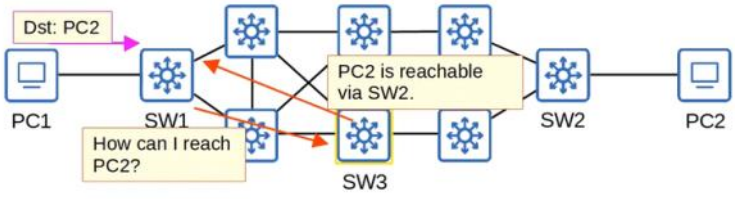
SD-Access Underlay

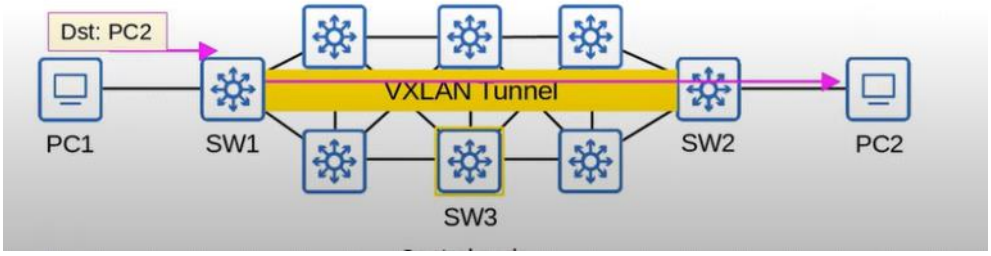
- The underlay's purpose is to support the VXLAN tunnels of the overlay.
- There are three different roles for switches in SD-Access:
  → **Edge nodes**: Connect to end hosts
  → **Border nodes**: Connect to devices outside of the SD-Access domain, ie. WAN routers.
  → **Control nodes**: Use LISP (Locator ID Separation Protocol) to perform various control plane functions.
- You can add SD-Access on top of an existing network *(brownfield deployment)* if your network hardware and software supports it.
  → Google 'Cisco SD-Access compatibility matrix' if you're curious.
  → In this case DNA Center won't configure the underlay.
- A new deployment *(greenfield deployment)* will be configured by DNA Center to use the optimal SD-Access underlay:
  → All switches are Layer 3 and use IS-IS as their routing protocol.
  → All links between switches are routed ports. This means STP is not needed.
  → Edge nodes (access switches) act as the default gateway of end hosts *(routed access layer)*.





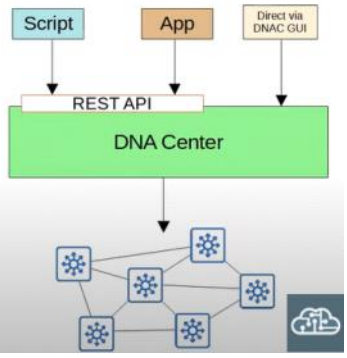






Cisco DNA Center

- Cisco DNA Center has two main roles:
  - → The SDN controller in SD-Access
  - → A network manager in a traditional network (non-SD-Access)
- DNA Center is an application installed on Cisco UCS server hardware.
- It has a REST API which can be used to interact with DNA center.
- The SBI supports protocols such as NETCONF and RESTCONF (as well as traditional protocols like Telnet, SSH, SNMP).
- DNA Center enables *Intent-Based Networking* (IBN).
  - → More buzzwords! Yay!
  - → The goal is to allow the engineer to communicate their intent for network behavior to DNA Center, and then DNA Center will take care of the details of the actual configurations and policies on devices.



Cisco DNA Center

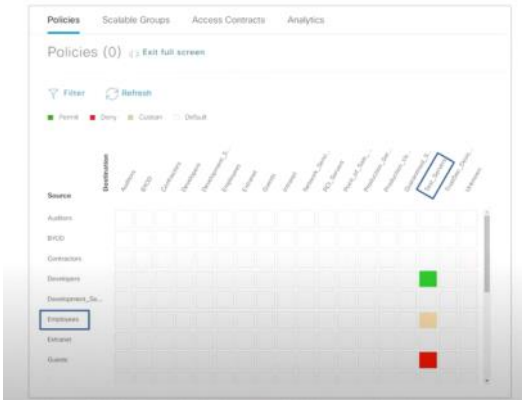
- Traditional security policies using ACLs can become VERY

Cisco DNA Center

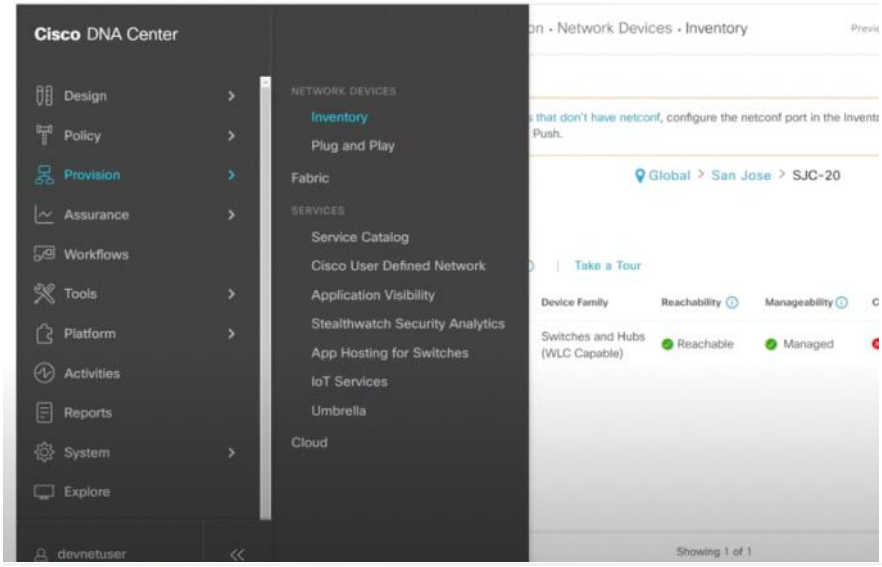
- Traditional security policies using ACLs can become VERY cumbersome.
  → ACLs can have **thousands** of entries.
  → The intent of entries is forgotten with time and as engineers leave and new engineers take over.
  → Configuring and applying the ACLs correctly across a network is cumbersome and leaves room for error.

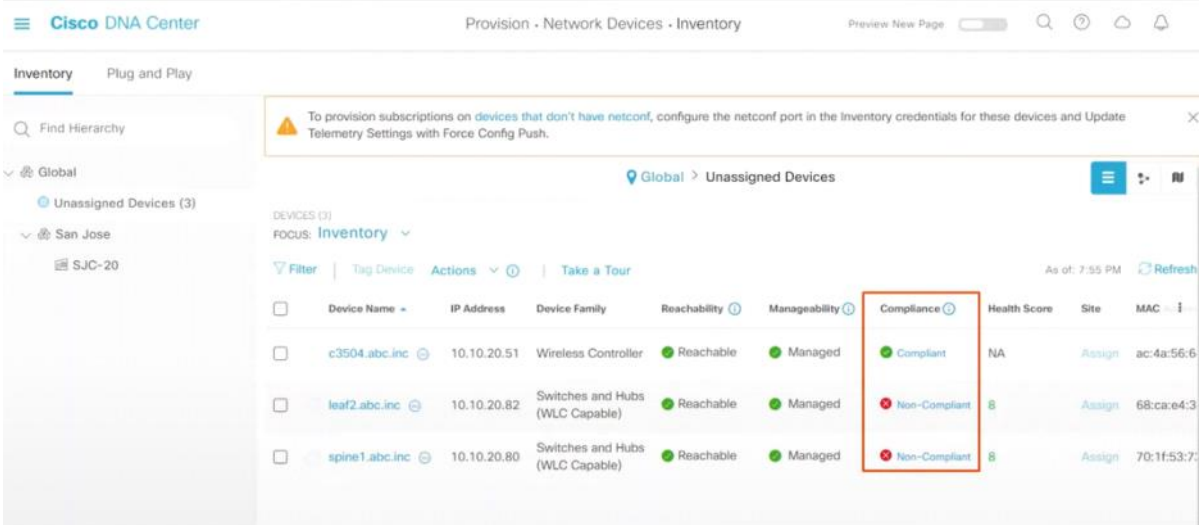
- DNA Center allows the engineer to specify the intent of the policy (this group of users can't communicate with this group, this group can access this server but not that server, etc.), and DNA Center will take care of the exact details of implementing the policy.

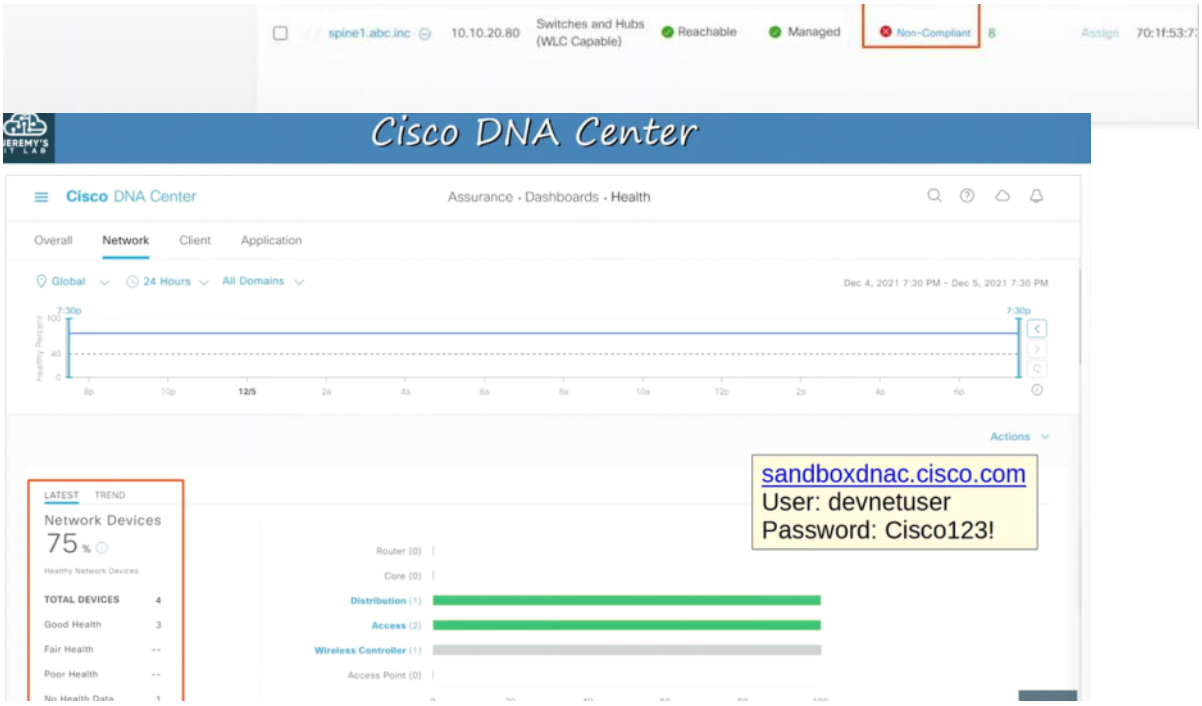
Cisco DNA Center



Cisco DNA Center







DNA Center vs Traditional Network Management

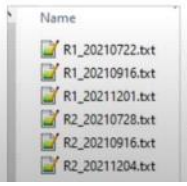
- Traditional network management:
  - → Devices are configured one-by-one via SSH or console connection.
  - → Devices are manually configured via console connection before being deployed.
  - → Configurations and policies are managed per-device. (distributed)
  - → New network deployments can take a long time due to the manual labor required.
  - → Errors and failures are more likely due to increased manual effort.

- DNA Center-based network management:
  - → Devices are centrally managed and monitored from the DNA Center GUI or other applications using its REST API.
  - → The administrator communicates their intended network behavior to DNA Center, which changes those intentions into configurations on the managed network devices.
  - → Configurations and policies are centrally managed.
  - → Software versions are also centrally managed. DNA Center can monitor cloud servers for new versions and then update the managed devices.
  - → New network deployments are much quicker. New devices can automatically receive their configurations from DNA Center without manual configuration.

Configuration Drift

- *Configuration drift* is when individual changes made over time cause a device's configuration to deviate from the standard/correct configurations as defined by the company.
  - → Although each device will have unique parts of its configuration (IP addresses, host name, etc), most of a device's configuration is usually defined in standard templates designed by the network architects/engineers of the company.
  - → As individual engineers make changes to devices (for example to troubleshoot and fix network issues, test configurations, etc.), the configuration of a device can drift away from the standard.
  - → Records of these individual changes and their reasons aren't kept.
  - → This can lead to future issues.

- Even without automation tools, it is best to have standard *configuration management* practices.
  - → When a change is made, save the config as a text file and place it in a shared folder.



  - → A standard naming system like *hostname_yyyymmdd* might be used.
  - → There are flaws to this system, as an engineer might forget to place the new config in the folder after making changes. Which one should be considered the 'correct' config?
  - → Even if configurations are properly saved like this, it doesn't guarantee that the configurations actually match the standard.

Configuration Provisioning

- *Configuration provisioning* refers to how configuration changes are applied to devices.
  → This includes configuring new devices, too.
- Traditionally, configuration provisioning is done by connecting to devices one-by-one via SSH.
  → This is not practical in large networks.
- Configuration management tools like Ansible, Puppet, and Chef allow us to make changes to devices on a mass scale with a fraction of the time/effort.
- Two essential components: *templates* and *variables*

```
hostname {{hostname}}
!
interface GigabitEthernet0/0
ip address {{address}} {{mask}}
ip ospf {{process_id}} area {{area}}
```

```
---
hostname: R1
address: 192.168.1.1
mask: 255.255.255.0
process_id: 1
area: 0
```

Template + Variables →

Config

```
hostname R1
!
interface GigabitEthernet0/0
ip address 192.16.1.1 255.255.255.0
ip ospf 1 area 0
```

Configuration Management Tools

- *Configuration management tools* are network automation tools that facilitate the centralized control of large numbers of network devices.
- The options you need to be aware of for the CCNA are Ansible, Puppet, and Chef.
- These tools were originally developed after the rise of VMs, to enable server system admins to automate the process of creating, configuring, and removing VMs.
  → However, they are also widely used to manage network devices.
- These tools can be used to perform tasks such as:
  → Generate configurations for new devices on a large scale.
  → Perform configuration changes on devices (all devices in your network, or a certain subset of devices).
  → Check device configurations for compliance with defined standards.
  → Compare configurations between devices, and between different versions of configurations on the same device.

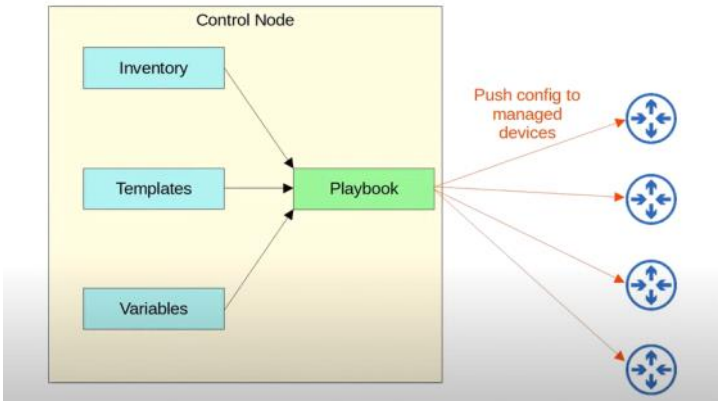
Ansible

- Ansible is a configuration management tool owned by Red Hat.
- Ansible itself is written in Python.
- Ansible is *agentless*.
  → It doesn't require any special software to run on the managed devices.

ANSIBLE

- Ansible uses SSH to connect to devices, make configuration changes, extract information, etc.
- Ansible uses a *push* model. The Ansible server (Control node) uses SSH to connect to managed devices and *push* configuration changes to them.
  → Puppet and Chef use a *pull* model.
- After installing Ansible itself, you must create several text files:
  → **Playbooks**: These are files are 'blueprints of automation tasks'. They outline the logic and actions of the tasks that Ansible should do. Written in YAML.
  → **Inventory**: These files list the devices that will be managed by Ansible, as well as characteristics of each device such as their device role (access switch, core switch, WAN router, firewall, etc). Written in INI, YAML, or other formats.
  → **Templates**: These files represent a device's configuration file, but specific values for variables are not provided. Written in Jinja2 format.
  → **Variables**: These files list variables and their values. These values are substituted into the templates to create complete configuration files. Written in YAML.

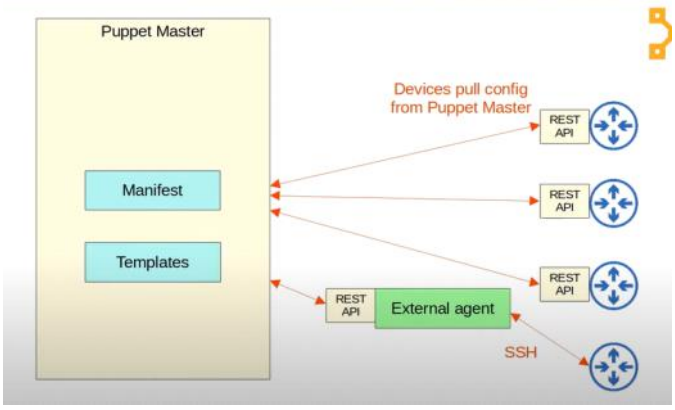
Ansible



Puppet

- Puppet is a configuration management tool written in Ruby.
- Puppet is typically agent-based.
  - → Specific software must be installed on the managed devices.
  - → Not all Cisco devices support a Puppet agent.
- It can be run agentless, in which a proxy agent runs on an external host, and the proxy agent uses SSH to connect to the managed devices and communicate with them.
- The Puppet server is called the 'Puppet master'.
- Puppet uses a pull model (clients 'pull' configurations from the Puppet master).
  - → Clients use TCP 8140 to communicate with the Puppet master.
- Instead of YAML, it uses a proprietary language for files.
- Text files required on the Puppet master include:
  - → **Manifest**: This file defines the desired configuration state of a network device.
  - → **Templates**: Similar to Ansible templates. Used to generate Manifests.

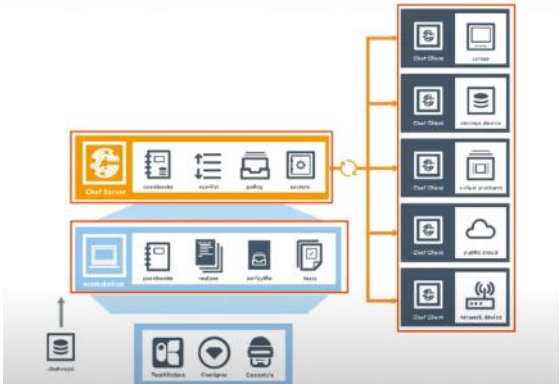
Puppet



Chef

- Chef is a configuration management tool written in Ruby.
- Chef is agent-based.
  - → Specific software must be installed on the managed devices.
  - → Not all Cisco devices support a Chef agent.
- Chef uses a pull model.
- The server uses TCP 10002 to send configurations to clients.
- Files use a DSL (Domain-Specific Language) based on Ruby.
- Text files used by Chef include:
  - → **Resources**: The 'ingredients' in a recipe. Configuration objects managed by Chef.
  - → **Recipes**: The 'recipes' in a cookbook. Outline the logic and actions of the tasks performed on the resources.
  - → **Cookbooks**: A set of related recipes grouped together.
  - → **Run-list**: An ordered list of recipes that are run to bring a device to the desired configuration state.

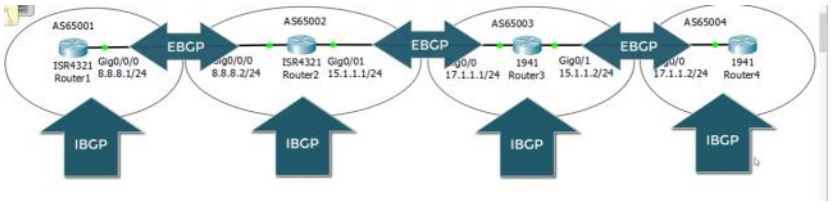
Chef



	Ansible	Puppet	Chef
Key Files defining actions	Playbook	Manifest	Recipe, Run-list
Communication Protocol	SSH	HTTPS (via REST API)	HTTPS (via REST API)
Key Port	22 (SSH port)	8140	10002
Agent-based/ Agentless	Agentless	Agent-based (or Agentless)	Agent-based
Push/Pull	Push	Pull	Pull

PRACTICING

Wednesday, March 8, 2023 2:20 PM



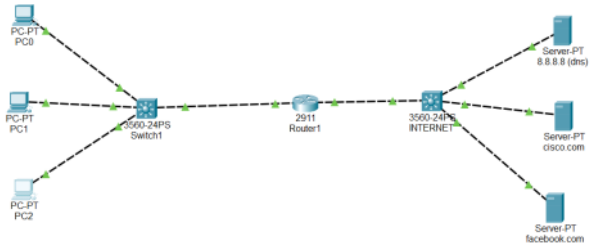
R1(config)# router bgp 65001(asn number)
R1(config-router)# neighbor 8.8.8.2 remote-as 65002
R1(config-router)# network 1.1.1.1 mask 255.255.255.255(not an inverse mask)
R1(config-router)# network 8.8.8.0 mask 255.255.255.0

R2(config)# router bgp 65002(asn number)
R2(config-router)# neighbor 8.8.8.1 remote-as 65001

Show ip bgp neighbors/summary(BGP STATE: ESTABLISHED)

R2(config-router)# network 2.2.2.2 mask 255.255.255.255(not an inverse mask)
R2(config-router)# network 8.8.8.0 mask 255.255.255.0
R1(config-router)# network 15.1.1.0 mask 255.255.255.0

Manually connect them one by one statically



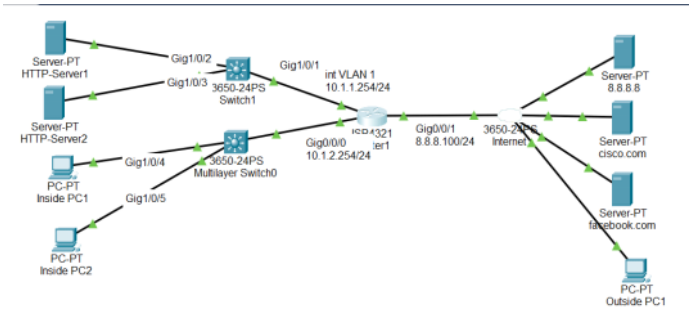
For nat address translation we need to put router1's left side as inside and other on eas outside
PORT ADDRESS TRANSLATIONS
R1# int g0/0/0
Ip nat inside
R1# int g0/0/1
Ip nat outside

Ip nat inside source list 1 interface gigabitethernet 0/0/1 overload

(ip nat inside source static tcp 10.1.1.100(insidelocal(eg:pc address or server adddress)) 80 8.8.8.200(insideglobal(public address ned to be purchased)) 80(in real world router cisco adds ectendable keywork at the end of cmd))
(ip nat inside source static 10.1.1.101 8.8.8.201 (full nat translation)(any traffic to 8.8.8.201 will go to 10.1.1.101)
clear ip nat translation *

Access-list 1 permit any

R1# sh nat ip translations



Access-list 100 permit tcp host 10.1.2.101 host 10.1.1.100 eq 80
Access-list 100 deny ip 10.1.2.0 0.0.0.255 10.1.1.0 0.0.0.255 (in real world log can be given as a word in last)(since this line comes on bottom of the priority this will be checked only if the above list is not matched)

Show access-lists

Ip access-list extended 100(same as the first line but it enters like a group instead of one line at a time)
20 permit tcp host 10.1.2.101 host 10.1.1.100 eq 80 (20 is the priority number)

Int g0/0/0
Ip access-group 100 in (this will apply it inbound to that interface)

Reload in 10(device will reload in 10 minutes)

POWER CYCLE DEVICES option in packet tracer will reboot all devices and confiuration wil be lost
When clicking the button in the router it goes to ROMMON mode
The ROM Monitor (ROMMON) is a bootstrap program that initializes the hardware and boots the Cisco IOS XE software when you power on or reload a router

Rommon 1>help(to see available commands)
Rommon 2>confreg 0x2122(will be available in show version command in normal mode)(boots into ROM if initial boot fails)
We can see available config register in internet

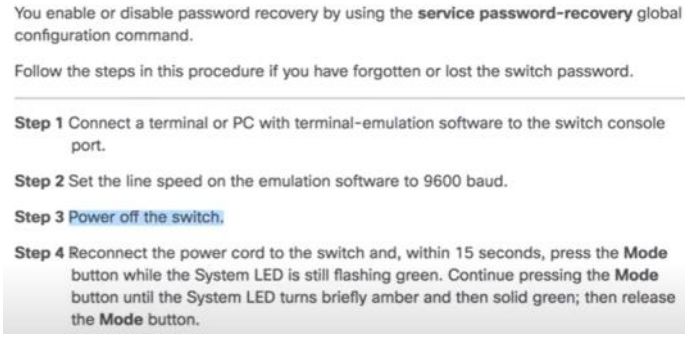
R1(config)# config-register registernumber(for initializing when normal mode)
If we specify 0x2120 configreg and reload it will go to rommon mode
0x2120 will enter into rommon mode
0x2142 ignores the NVRAM contents(startup config)

Dir command lists the files in flash:/

For password recovery for routers we need to do something in rommon mode so when booting press ctrl+c to stop and goto rommon mode

1) We can change the configreg as 0x2142 (bypasses NVRAM) and reset and copy the start to run config (enable password will be shown in startup config)
2) If the password/secret is encrypted then after copying and wrting the start ti run config we can override them by giving enable secret cisco to change the password

For switch password recovery


You enable or disable password recovery by using the **service password-recovery** global configuration command.

Follow the steps in this procedure if you have forgotten or lost the switch password.

Step 1 Connect a terminal or PC with terminal-emulation software to the switch console port.

Step 2 Set the line speed on the emulation software to 9600 baud.

Step 3 Power off the switch.

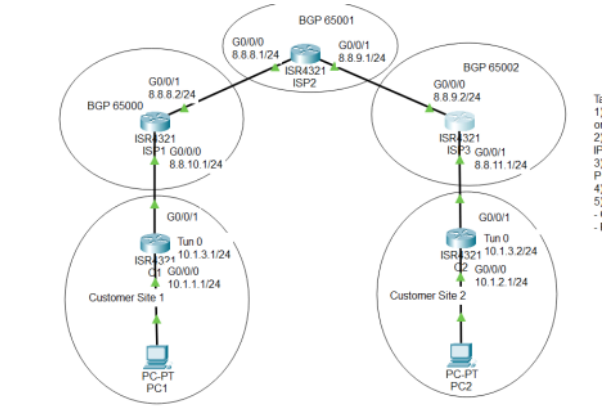
Step 4 Reconnect the power cord to the switch and, within 15 seconds, press the **Mode** button while the System LED is still flashing green. Continue pressing the **Mode** button until the System LED turns briefly amber and then solid green; then release the **Mode** button.

After pressing the mode button we get into rommon mode in switch
Switch: flash_init
Switch: load_helper(if there)
Switch: dir flash: (to see contents in flash)
Switch: rename flash:config.text flash:config.text.old
Switch: boot

S1> en
S1# rename flash:config.text.old flash:config.text
S1# copy flash:config.text system:running-config
S1(config)# enable secret cisco
Copy run start

In packet tracer we can boot the switch and erase the nvram contents and after booting up it will not ask password since it is erased and using the renamed config we can copy to run config as shown in above steps

GRE TUNNELING:



C1 SIDE:

Int tunnel 0
Ip add 10.1.3.2 255.255.255.0
Tunnel source ggabitethernet g0/0/1(publicfacing)
Tunnel destination 8.8.11.2 (public facing in c2 router)

Router eigrp 100
Network 10.0.0.0
No auto-summary
C2 SIDE:

Int tunnel 0
Ip add 10.1.3.1 255.255.255.0
Tunnel source ggabitethernet g0/0/1(publicfacing)
Tunnel destination 8.8.10.2 (public facing in c1 router)

Router eigrp 100
Network 10.0.0.0
No auto-summary
After this configuration only the ip address of c1 and c2 is visible to pc's not public/isp adddresses

IF OS IS DELETED IN THE ROUTER THEN WE CAN USE ROMMON MODE TO DOWNLOAD THE OS FILE FORM TFTP SERVER BY SETTING ENV VARIABLES(IP_ADDDRESS,IP_SUBNET_MASK,DEFAULT_GATEWAY,TFTP_SERVER,TFTP_FILE) and use tftpdnld , reset/boot

BACK AND RESTORE:

copy running-config tftp
Address or name of remote host []? 10.1.1.100
Destination filename [R1-confg]? R1-runconfig

Boot system flash filename (to boot using another os image)(in conft mode)
Write
Reload

Ppp is the encapsulation we can set in interfaces and it should match or support in both receiving and sending end and also can provide authentication strings like passwords or byh automatic handskae auth

HUB has one collision domain since each time packets are flooded onto every interface but for switch each interfaces acts as a separate collission domain since all packets can be received and sent at the same time

ARP is used only in ethernet links not in serial console links

QUALITY OF SERVICE:

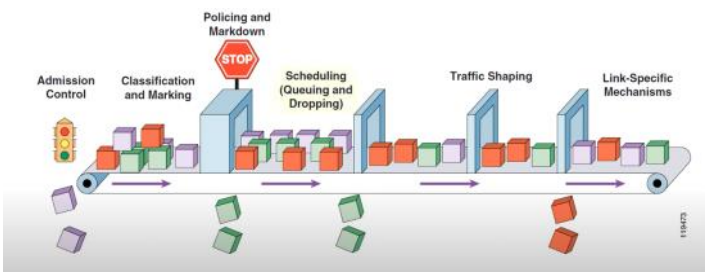
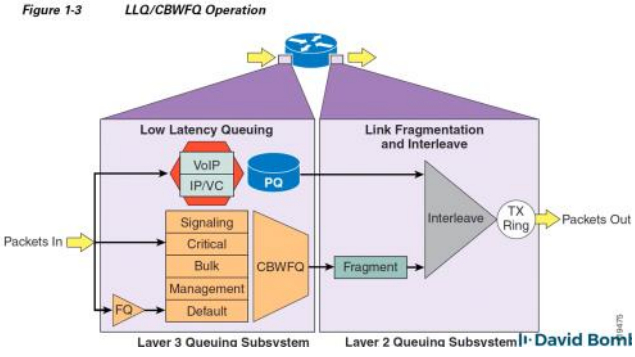
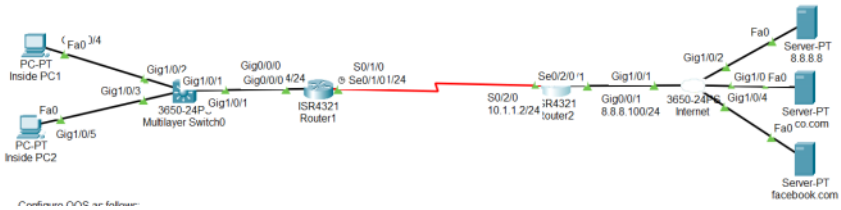


Figure 1-3 LLQ/CBWFQ Operation





NETWORK BASED APPLICATION RECOGNITION(NBAR)
R1(config)#class-map voice(defaults to match-all)
R1(config-cmap)#match protocol rtp

R1(config)#class-map http(defaults to match-all)
R1(config-cmap)#match protocol http

R1(config)#class-map icmp(defaults to match-all)
R1(config-cmap)#match protocol icmp

R1(config)policy-map mark
R1(config-pmap)# class voice
R1(config-pmap-c)# set ip dscp ef
R1(config-pmap-c)# priority 100(priority bandwidth of 100kbps)
R1(config-pmap)# class http
R1(config-pmap-c)# set ip dscp af31
R1(config-pmap-c)# bandwidth 50(minimim bandwidth 50kbps)
R1(config-pmap)# class icmp
R1(config-pmap-c)# set ip dscp af11
R1(config-pmap-c)# bandwidth 25

R1(config)# int s0/1/0
R1(config-if)# service-policy output(or input) mark(policymarkname)

R2(config)#class-map voice(defaults to match-all)
R2(config-cmap)#match ip dscp ef

R2(config)#class-map http(defaults to match-all)
R2(config-cmap)#match ip dscp af31

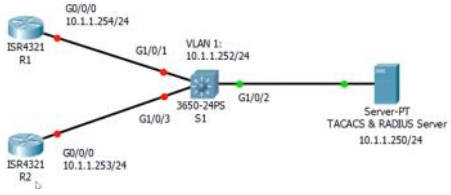
R2(config)#class-map icmp(defaults to match-all)
R2(config-cmap)#match ip dscp af11

R2(config)policy-map remark
R2(config-pmap)# class voice
R2(config-pmap-c)# set precedence 5
R2(config-pmap)# class http
R2(config-pmap-c)# set precedence 3
R2(config-pmap)# class icmp
R2(config-pmap-c)# set precedence 0

R2(config)# int s0/2/0
R2(config-if)# service-policy input(or output) remark(policymarkname)

Show policy-map
Show policy map int s0/1/0(shows the packets transferred through this map)(matches)

authentication, authorization, and accounting (AAA)



R1(config)# aaa new-model
R1(config)# username backup password cisco
R1(config)# aaa authentication login default group tactacs+ local
R1(config)# aaa authentication enable default group tactacs+ local

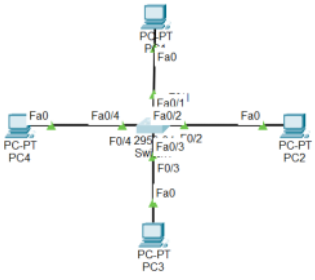
R1(config)#tactacs-server host 10.1.1.250 key cisco

R2(config)# aaa new-model
R2(config)# username backup password cisco
R2(config)# aaa authentication login default group radius local
R2(config)# aaa authentication enable default group radius local

R2(config)#radius server somename
R2(config-radius-server)#address ipv4 10.1.1.250
R2(config-radius-server)key cisco

(Port will be 1645 which will be default in radius server side so it will be default put in the config)
The backup password we created will be used when we cant access server

LOCAL SPAN(PORT MONITORING ETC..)
SPAN (Switched Port Analyzer) is a dedicated port on a switch that takes a mirrored copy of network traffic from within the switch to be sent to a destination



(for devices in the same vlan in the switch)
S1(config)# monitor session 1 source interface fastEthernet 0/1 both(tx,rx)

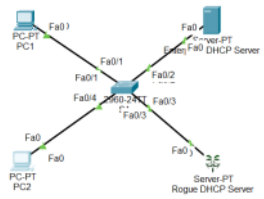
S1(config)# monitor session 1 destination interface fastEthernet 0/2
(for devices each in different vlan)
S1(config)# monitor session 2 source interface fastEthernet 0/1 , fastEthernet 0/3 , fastEthernet 0/4
S1(config)# monitor session 2 destination interface fastEthernet 0/2

REMOTE SPAN
S1(config)# monitor session 1 source interface fastEthernet 0/1 both
S1(config)# monitor session 1 destination remote vlan 99 reflector-port fa0/1

S2(config)# monitor session 1 source remote vlan 99
S2(config)# monitor session 1 destination interface fastethernet 0/1

(acccordingly in therir respective lan's configure the access,trunk ports to make this work)

DHCP SNOOPING:

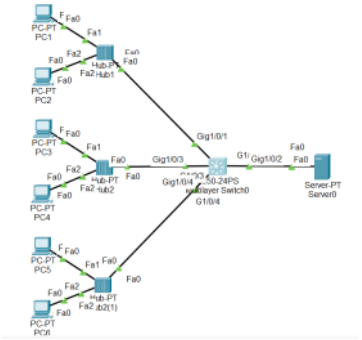


SW1(config)# ip dhcp snooping
SW1# show ip dhcp snooping
SW1# show ip dhcp snooping binding
SW1# show ip dhcp snooping database

SW1# debug ip dhcp snooping packet(debugs the packet flows)

SW1(config)# ip dhcp snooping vlan 1
SW1(conifg)# int f0/2
SW1(config-if)#ip dhcp snooping trust

PORT-SECURITY:



SW1(config)# int g1/0/1
SW1(config-if)# switchport mode access
SW1(config-if)# switchport port-security

SW1# show port-security
SW1# show port-security int g1/0/1
SW1# show port-security address
(when pc1 sends dhcp request it will get processed and since therer is only one mac address allowed pc1 is allowed and when pc2 send the request the violation occurs and interface is disabled)
Vice versa can happen if pc2 starts

SW1# show port-security maximum 2

SW1(config)# int g1/0/3
SW1(config-if)# switchport mode access
SW1(config-if)# switchport port-security mac-address sticky
SW1(config-if)# switchport port-security
(here the max address learnt in the first request will be saved to runconfiglike switchport port-security mac-address sticky macaddressofpc and if it is saved and we reload the router the sticky macaddress command will be there and only that particular macaddress isallowed)

SW1(config)# int g1/0/4
SW1(config-if)# switchport mode access
SW1(config-if)# switchport port-security mac-address pcmacaddress
SW1(config-if)# switchport port-security
SW1(config-if)# switchport port-security violation restrict(/shutdown/protect)(this will increment the violation counter)

PORTFAST:

Int range g0/1/2-g0/1/3
Spanning-tree portfast

Arp -a (lists the mac-ip table)
Arp -d(clears the table)(then broadcast since not there)

R1(conig)# Logging host 10.1.1.250(syslog server)
R1(conig)# Service timestamps log datetime msec(to log with timestamps)

R1(conig)# ntp server 10.1.1.250(ntp server)

RIP:

R1(config)# router rip
R1(config-router)# network 10.1.1.10
R1(config-router)# network 10.2.1.10
R1(config-router)# version 2

Show ip rip database

SW1# ip default-gateway 192.168.1.254

If we are not disabling the automatic summary in RIP then for eg in a router a loopback having 172.168.2.2 and in another far away network with 172.168.1.0 the router will think that both are on the same network and construct a single routee with two next hops(to go to this network we can go to this or this like that). With automatic summary disabled it will have a specific route to that particular networks

R1(config-router)#Default-information originate (to advertise the default routes)

Ip name-server 8.8.8.8(to tell router to which dns dserver to use)

The abstract “0 – 4” means that the device can allow 5 simultaneous virtual connections which may be Telnet or SSH.The virtual terminal or “VTY” lines are virtual lines that allow connecting to the device using telnet or Secure Shell (SSH). Cisco devices can have up to 16 VTY lines

Line vty 0 4
Login
Password cisco

Enable password cisco

Debug ip packet (logs all the ip packet movement in very detail)
Un all(turns off it)

Debug eigrp packets(debugging packets of eigrp)
SAME AUTONOMOUS SYSTEM NU MBER SHOULD BE CONFIGURED IN ALL ROUTERS USING EIGRP
ROUTER EIGRP 100

Logging console (to log changes in the console port)
Metric weights 1 1 1 1 1 1(5-K values setting)(should match in all routers)

'

IP SUBNETTING:

192.168.1.0/24 -> 4 subnets

2^n -> Subnets 2^2 = 4

1111 1111.1111 1111.1111 1111.0000 0000 = 24
255 . 255 .255 .0

Network Host
192.168.1.0000 0000

Subnet = Stealing

Network Subnet Host
192.168.1.00 00 0000 = 192.168.1.0/26
192.168.1.01 00 0000 = 192.168.1.64/26
192.168.1.10 00 0000 = 192.168.1.128/26
192.168.1.11 00 0000 = 192.168.1./26

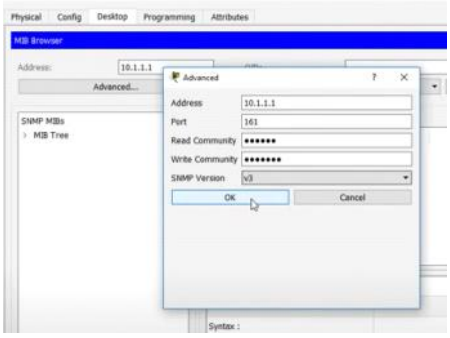
10000000 = 128

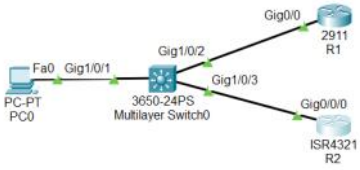
Network Subnet Host
192.168.1.00 00 0000 = 192.168.1.0/26
192.168.1.01 00 0000 = 192.168.1.64/26
192.168.1.10 00 0000 = 192.168.1.128/26
192.168.1.11 00 0000 = 192.168.1.192/26

Here after stealing the bits we have to give the corresponding bits so that when converting the last octet(host portion) into decimal we get the correct subnet address

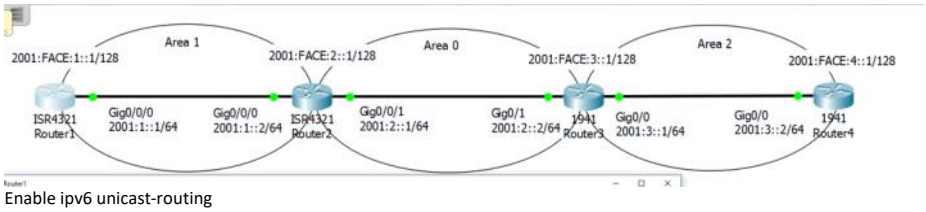
Here n is the number of bits we are stealing form host portion
If we steal 2 bits in host portion then in the 8 bits only 6 bits will be remaining

SNMP :





R1(config)# snmp-server community public ro
R1(config)# snmp-server community private rw
R2(config)# snmp-server community public ro
R2(config)# snmp-server community private rw



Enable ipv6 unicast-routing

R1#Ipv6 router ospf 1
R1#Router-id 1.1.1.1

R1#Show ipv6 protocols(to see routing protocols for ipv6)

R1#Int g0/0/0
R1#Ipv6 ospf 1 area 1
R1# int l0
R1# ipv6 ospf 1 area 1

R2#Int g0/0/0
R2#Ipv6 ospf 1 area 1
R2# int g0/0/1
R2# ipv6 ospf 1 area 0

Specify the area accordig to the area in the diagram in spcific interfaces

To identify the first usable host we need to put a AND operation between the subnetmask and the ip address in binary format