Componente Práctico del Diplomado de Profundización CISCO CCNP

Ángela Natalia Arévalo González.

Universidad Nacional Abierta y a Distancia

Escuela de ciencias básicas, tecnología e ingeniería

Ingeniería en Telecomunicaciones

Octubre del 2018

Tabla de contenido

Resumen 3
Abstrac 3
Introducción 4
Objetivos 5
Objetivo General. 5
Objetivos específicos. 5
Configure and Verify Path Control Using PBR 6
Using the AS_PATH Attribute 24
Static VLANS, Trunking, and VTP 35
Synchronizing Campus Network Devices using Network Time Protocol (NTP) 60
Conclusiones. 76
Referencias Bibliográficas 77

Resumen.

El presente trabajo se desarrolla para aplicar de una manera práctica los conocimientos adquiridos en el Diplomado de Profundización CISCO CCNP, para lo cual se desarrollarán practicas empleando diferentes simuladores como Packet Tracer y GNS3. Los cuales permiten dar una idea clara y muy aproximada de la realidad sobre las diferentes configuraciones que se requieren a la hora de implementar diferentes tipos de red.

Abstrac.

This work is developed to apply in a practical way the knowledge acquired in the CISCO CCNP Diploma Course, for which practices will be developed using different simulators such as Packet Tracer and GNS3. Which allow to give a clear and very approximate idea of reality about the different configurations that are required when implementing different types of network.

Introducción.

Con la elaboración del presente trabajo se desea aplicar de una manera práctica los conocimientos adquiridos en el Diplomado de Profundización CISCO CCNP, para lo cual se desarrollarán practicas empleando diferentes simuladores como Packet Tracer y GNS3. Los cuales permiten dar una idea clara y muy aproximada de la realidad sobre las diferentes configuraciones que se requieren a la hora de implementar diferentes tipos de red.

Objetivos

Objetivo General.

Con la elaboración del presente trabajo se desea aplicar los conocimientos adquiridos en el Diplomado de Profundización CISCO CCNP, para lograr brindar soluciones en la construcción y modificación de redes inalámbricas y cableadas.

Objetivos específicos.

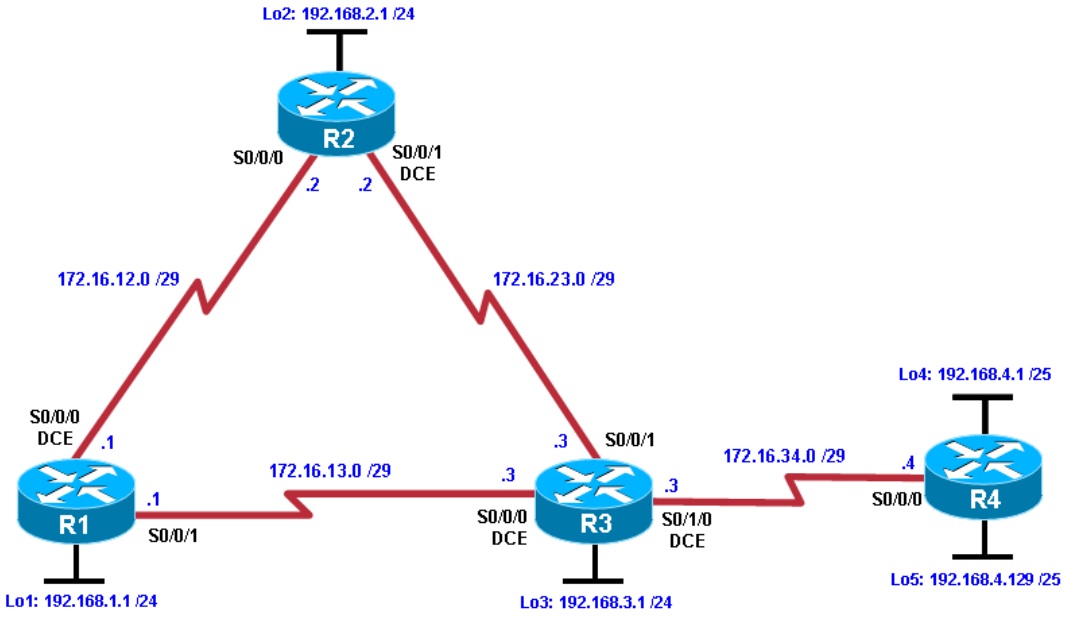
Identificar los escenarios donde se presentan los inconvenientes para plantear la solución adecuada.

Realizar el montaje de las redes empleando los simuladores de Packet Tracer y GSN3.

Configurar cada dispositivo para lograr su adecuado funcionamiento dentro de la red.

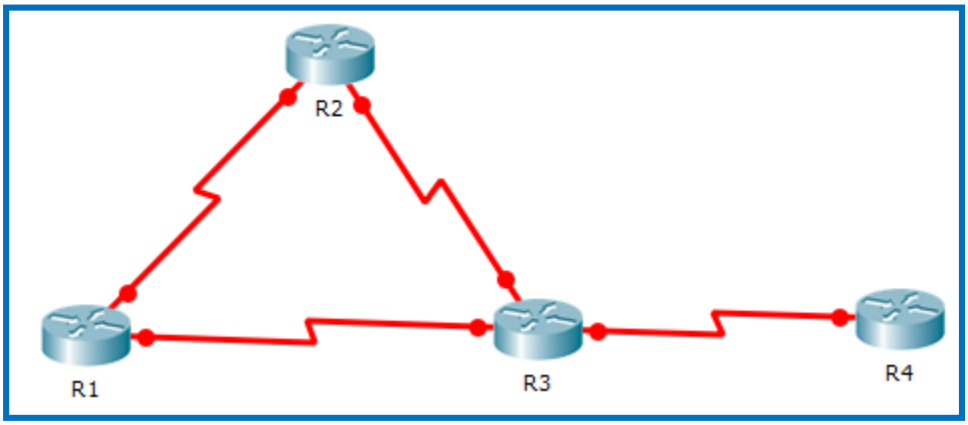
Configure and Verify Path Control Using PBR

Topology



Step 1: Configure loopbacks and assign addresses.

Cable the network as shown in the topology diagram. Erase the startup configuration, and reload each router to clear previous configurations.



Using the addressing scheme in the diagram, create the loopback interfaces and apply IP addresses to these and the serial interfaces on R1, R2, R3, and R4. On the serial interfaces connecting R1 to R3 and R3 to R4, specify the bandwidth as 64 Kb/s and set a clock rate on the DCE using the **clock rate 64000** command. On the serial interfaces connecting R1 to R2 and R2 to R3, specify the bandwidth as 128 Kb/s and set a clock rate on the DCE using the **clock rate 128000** command.

You can copy and paste the following configurations into your routers to begin.

Note: Depending on the router model, interfaces might be numbered differently than those listed. You might need to alter them accordingly.

Router R1

```
hostname R1
!
interface Lo1
 description R1 LAN
 ip address 192.168.1.1 255.255.255.0
!
interface Serial0/0/0
 description R1 --> R2
 ip address 172.16.12.1 255.255.255.248
 clock rate 128000
 bandwidth 128
 no shutdown
!
interface Serial0/0/1
 description R1 --> R3
 ip address 172.16.13.1 255.255.255.248
 bandwidth 64
 no shutdown
!
End
```

```
Router>en
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#hostname R1
R1(config)#interface Lo1

R1(config-if)#
%LINK-5-CHANGED: Interface Loopback1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback1, changed state to up

R1(config-if)#description R1 LAN
R1(config-if)#ip address 192.168.1.1 255.255.255.0
R1(config-if)#interface Serial0/0/0
R1(config-if)#description R1 --> R2
R1(config-if)#ip address 172.16.12.1 255.255.255.248
R1(config-if)#clock rate 128000
R1(config-if)#bandwidth 128
R1(config-if)#no shutdown

%LINK-5-CHANGED: Interface Serial0/0/0, changed state to down
R1(config-if)#interface Serial0/0/1
R1(config-if)#description R1 --> R3
R1(config-if)#ip address 172.16.13.1 255.255.255.248
R1(config-if)#bandwidth 64
R1(config-if)#no shutdown

%LINK-5-CHANGED: Interface Serial0/0/1, changed state to down
R1(config-if)#end
```

Router R2

```
hostname R2
!
interface Lo2
 description R2 LAN
 ip address 192.168.2.1 255.255.255.0
!
interface Serial0/0/0
 description R2 --> R1
 ip address 172.16.12.2 255.255.255.248
 bandwidth 128
 no shutdown

interface Serial0/0/1
 description R2 --> R3
 ip address 172.16.23.2 255.255.255.248
 clock rate 128000
 bandwidth 128
 no shutdown
!
End
```

```
Router>en
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#hostname R2
R2(config)#interface Lo2

R2(config-if)#
%LINK-5-CHANGED: Interface Loopback2, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback2, changed state to up

R2(config-if)#description R2 LAN
R2(config-if)#ip address 192.168.2.1 255.255.255.0
R2(config-if)#interface Serial0/0/0
R2(config-if)#description R2 --> R1
R2(config-if)#ip address 172.16.12.2 255.255.255.248
R2(config-if)#bandwidth 128
R2(config-if)#no shutdown

R2(config-if)#
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to up

R2(config-if)#interface Serial0/0/1
R2(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed state to up

R2(config-if)#description R2 --> R3
R2(config-if)#ip address 172.16.23.2 255.255.255.248
R2(config-if)#clock rate 128000
R2(config-if)#bandwidth 128
R2(config-if)#no shutdown

%LINK-5-CHANGED: Interface Serial0/0/1, changed state to down
R2(config-if)#end
```

Router R3

```
hostname R3
!
interface Lo3
 description R3 LAN
 ip address 192.168.3.1 255.255.255.0
!
interface Serial0/0/0
 description R3 --> R1
 ip address 172.16.13.3 255.255.255.248
 clock rate 64000
 bandwidth 64
 no shutdown
!
interface Serial0/0/1
 description R3 --> R2
 ip address 172.16.23.3 255.255.255.248
 bandwidth 128
 no shutdown
!
interface Serial0/1/0
 description R3 --> R4
 ip address 172.16.34.3 255.255.255.248
 clock rate 64000
 bandwidth 64
 no shutdown
!
End
```

```
Router>en
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#hostname R3
R3(config)#interface Lo3

R3(config-if)#
%LINK-5-CHANGED: Interface Loopback3, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback3, changed state to up

R3(config-if)#description R3 LAN
R3(config-if)#ip address 192.168.3.1 255.255.255.0
R3(config-if)#interface Serial0/0/0
R3(config-if)#description R3 --> R1
R3(config-if)#ip address 172.16.13.3 255.255.255.248
R3(config-if)#clock rate 64000
This command applies only to DCE interfaces
R3(config-if)#bandwidth 64
R3(config-if)#no shutdown

R3(config-if)#
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to up

R3(config-if)#interface Serial0/0/1
R3(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed state to up

R3(config-if)#description R3 --> R2
R3(config-if)#ip address 172.16.23.3 255.255.255.248
R3(config-if)#bandwidth 128
R3(config-if)#no shutdown
```

```
R3(config-if)#interface Serial0/1/0
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/1, changed state to up

R3(config-if)#description R3 --> R4
R3(config-if)#ip address 172.16.34.3 255.255.255.248
R3(config-if)#clock rate 64000
R3(config-if)#bandwidth 64
R3(config-if)#no shutdown

%LINK-5-CHANGED: Interface Serial0/1/0, changed state to down
R3(config-if)#end
```

Router R4

```
hostname R4
!
interface Lo4
 description R4 LAN A
 ip address 192.168.4.1 255.255.255.128
!
interface Lo5
 description R4 LAN B
 ip address 192.168.4.129 255.255.255.128
!
interface Serial0/0/0
 description R4 --> R3
 ip address 172.16.34.4 255.255.255.248
 bandwidth 64
 no shutdown
!
End
```

```
Router>en
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#hostname R4
R4(config)#interface Lo4

R4(config-if)#
%LINK-5-CHANGED: Interface Loopback4, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback4, changed state to up

R4(config-if)#description R4 LAN A
R4(config-if)#ip address 192.168.4.1 255.255.255.128
R4(config-if)#interface Lo5

R4(config-if)#
%LINK-5-CHANGED: Interface Loopback5, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback5, changed state to up

R4(config-if)#description R4 LAN B
R4(config-if)#ip address 192.168.4.129 255.255.255.128
R4(config-if)#interface Serial0/0/0
R4(config-if)#description R4 --> R3
R4(config-if)#ip address 172.16.34.4 255.255.255.248
R4(config-if)#bandwidth 64
R4(config-if)#no shutdown

R4(config-if)#
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to up

R4(config-if)#end
```

Verify the configuration with the **show ip interface brief**, **show protocols,** and **show interfaces description** commands. The output from router R3 is shown here as an example.

```
R3# show ip interface brief | include up
Serial0/0/0                172.16.13.3     YES manual up
up
Serial0/0/1                172.16.23.3     YES manual up
up
Serial0/1/0                172.16.34.3     YES manual up
up
Loopback3                  192.168.3.1     YES manual up
up
R3#
```

```
R3#show ip interface brief | include up
Serial0/0/0            172.16.13.3     YES manual up                    up
Serial0/0/1            172.16.23.3     YES manual up                    up
Serial0/1/0            172.16.34.3     YES manual up                    up
Loopback3              192.168.3.1     YES manual up                    up
R3#
```

```
R3# show protocols
Global values:
  Internet Protocol routing is enabled
Embedded-Service-Engine0/0 is administratively down, line protocol
is down
GigabitEthernet0/0 is administratively down, line protocol is down
GigabitEthernet0/1 is administratively down, line protocol is down
Serial0/0/0 is up, line protocol is up
  Internet address is 172.16.13.3/29
Serial0/0/1 is up, line protocol is up
  Internet address is 172.16.23.3/29
Serial0/1/0 is up, line protocol is up
  Internet address is 172.16.34.3/29
Serial0/1/1 is administratively down, line protocol is down
Loopback3 is up, line protocol is up
  Internet address is 192.168.3.1/24
R3#
```

```
R3#show protocols
Global values:
  Internet Protocol routing is enabled
GigabitEthernet0/0 is administratively down, line protocol is down
GigabitEthernet0/1 is administratively down, line protocol is down
Serial0/0/0 is up, line protocol is up
  Internet address is 172.16.13.3/29
Serial0/0/1 is up, line protocol is up
  Internet address is 172.16.23.3/29
Serial0/1/0 is up, line protocol is up
  Internet address is 172.16.34.3/29
Serial0/1/1 is administratively down, line protocol is down
Loopback3 is up, line protocol is up
  Internet address is 192.168.3.1/24
```

```
R3# show interfaces description | include up
Se0/0/0                        up              up       R3 --> R1
```

```
Se0/0/1                        up              up        R3 --> R2
Se0/1/0                        up              up        R3 --> R4
Lo3                            up              up        R3 LAN
R3#
```

```
R3# show interfaces description | include up
Se0/0/0                        up              up        R3 --> R1
Se0/0/1                        up              up        R3 --> R2
Se0/1/0                        up              up        R3 --> R4
Lo3                            up              up        R3 LAN
R3#
```

Step 3: Configure basic EIGRP.

a. Implement EIGRP AS 1 over the serial and loopback interfaces as you have configured it for the other EIGRP labs.

Advertise networks 172.16.12.0/29, 172.16.13.0/29, 172.16.23.0/29, 172.16.34.0/29, 192.168.1.0/24, 192.168.2.0/24, 192.168.3.0/24, and 192.168.4.0/24 from their respective routers.

You can copy and paste the following configurations into your routers.

Router R1

```
router eigrp 1
 network 192.168.1.0
 network 172.16.12.0 0.0.0.7
 network 172.16.13.0 0.0.0.7
 no auto-summary
```

```
R1(config)#router eigrp 1
R1(config-router)#network 192.168.1.0
R1(config-router)#network 172.16.12.0 0.0.0.7
R1(config-router)#network 172.16.13.0 0.0.0.7
R1(config-router)#no auto-summary
```

Router R2

```
router eigrp 1
 network 192.168.2.0
 network 172.16.12.0 0.0.0.7
 network 172.16.23.0 0.0.0.7
 no auto-summary
```

```
R2(config)#router eigrp 1
R2(config-router)#network 192.168.2.0
R2(config-router)#network 172.16.12.0 0.0.0.7
R2(config-router)#
%DUAL-5-NBRCHANGE: IP-EIGRP 1: Neighbor 172.16.12.1 (Serial0/0/0) is up: new adjacency

R2(config-router)#network 172.16.23.0 0.0.0.7
R2(config-router)#no auto-summary
```

Router R3

```
router eigrp 1
 network 192.168.3.0
 network 172.16.13.0 0.0.0.7
```

```
 network 172.16.23.0 0.0.0.7
 network 172.16.34.0 0.0.0.7
 no auto-summary
```

```
R3(config)#router eigrp 1
R3(config-router)#network 192.168.3.0
R3(config-router)#network 172.16.13.0 0.0.0.7
R3(config-router)#
%DUAL-5-NBRCHANGE: IP-EIGRP 1: Neighbor 172.16.13.1 (Serial0/0/0) is up: new adjacency

R3(config-router)#network 172.16.23.0 0.0.0.7
R3(config-router)#
%DUAL-5-NBRCHANGE: IP-EIGRP 1: Neighbor 172.16.23.2 (Serial0/0/1) is up: new adjacency

R3(config-router)#network 172.16.34.0 0.0.0.7
R3(config-router)#no auto-summary
```

Router R4

```
router eigrp 1
 network 192.168.4.0
 network 172.16.34.0 0.0.0.7
 no auto-summary
```

```
R4(config)#router eigrp 1
R4(config-router)#network 192.168.4.0
R4(config-router)#network 172.16.34.0 0.0.0.7
R4(config-router)#
%DUAL-5-NBRCHANGE: IP-EIGRP 1: Neighbor 172.16.34.3 (Serial0/0/0) is up: new adjacency

R4(config-router)#no auto-summary
```

You should see EIGRP neighbor relationship messages being generated.

Step 4: Verify EIGRP connectivity.

a. Verify the configuration by using the **show ip eigrp neighbors** command to check which routers have EIGRP adjacencies.

```
R1# show ip eigrp neighbors
EIGRP-IPv4 Neighbors for AS(1)
H   Address                 Interface         Hold Uptime   SRTT
RTO  Q  Seq
                                              (sec)         (ms)
Cnt Num
1   172.16.13.3             Se0/0/1             10 00:01:55   27
2340  0  9
0   172.16.12.2             Se0/0/0             13 00:02:07    8
1170  0  11
R1#
```

```
R1#show ip eigrp neighbors
IP-EIGRP neighbors for process 1
H   Address         Interface       Hold Uptime    SRTT   RTO   Q   Seq
                                    (sec)          (ms)       Cnt Num
0   172.16.12.2     Se0/0/0         13   00:14:11  40     1000  0   17
1   172.16.13.3     Se0/0/1         12   00:12:09  40     1000  0   24

R1#
```

```
R2# show ip eigrp neighbors
EIGRP-IPv4 Neighbors for AS(1)
H   Address                 Interface         Hold Uptime   SRTT
RTO  Q  Seq
                                              (sec)         (ms)
Cnt Num
1   172.16.23.3             Se0/0/1             12 00:02:15   12
1170  0  10
0   172.16.12.1             Se0/0/0             11 00:02:27    9
1170  0  13
R2#
```

```
R2#show ip eigrp neighbors
IP-EIGRP neighbors for process 1
H   Address         Interface       Hold Uptime    SRTT   RTO   Q   Seq
                                    (sec)          (ms)       Cnt Num
0   172.16.12.1     Se0/0/0         14   00:15:50  40     1000  0   15
1   172.16.23.3     Se0/0/1         11   00:13:39  40     1000  0   24

R2#
```

```
R3# show ip eigrp neighbors
EIGRP-IPv4 Neighbors for AS(1)
H   Address                 Interface         Hold Uptime   SRTT
RTO  Q  Seq
                                              (sec)         (ms)
Cnt Num
2   172.16.34.4             Se0/1/0             12 00:02:14   44
2340  0  3
1   172.16.23.2             Se0/0/1             11 00:02:23   10
1170  0  10
0   172.16.13.1             Se0/0/0             10 00:02:23 1031
5000  0  12
R3#
```

```
R3#show ip eigrp neighbors
IP-EIGRP neighbors for process 1
H   Address         Interface       Hold Uptime    SRTT   RTO   Q   Seq
                                    (sec)          (ms)       Cnt Num
0   172.16.13.1     Se0/0/0         14   00:14:53  40     1000  0   16
1   172.16.23.2     Se0/0/1         10   00:14:44  40     1000  0   18
2   172.16.34.4     Se0/1/0         11   00:09:11  40     1000  0   13

R3#
```

```
R4# show ip eigrp neighbors
EIGRP-IPv4 Neighbors for AS(1)
H   Address                 Interface         Hold Uptime   SRTT
RTO  Q  Seq
                                              (sec)         (ms)
Cnt Num
0   172.16.34.3             Se0/0/0             10 00:02:22   37
2340  0  11

R4#
```

```
R4#show ip eigrp neighbors
IP-EIGRP neighbors for process 1
H   Address         Interface      Hold Uptime    SRTT   RTO   Q   Seq
                                   (sec)          (ms)       Cnt  Num
0   172.16.34.3     Se0/0/0        13   00:10:17  40     1000  0   25

R4#
```

Did you receive the output you expected?

Si se recibió, con unas pequeñas variaciones.

b. Run the following Tcl script on all routers to verify full connectivity.

```
R1# tclsh

foreach address {
172.16.12.1
172.16.12.2
172.16.13.1
172.16.13.3
172.16.23.2
172.16.23.3
172.16.34.3
172.16.34.4
192.168.1.1
192.168.2.1
192.168.3.1
192.168.4.1
192.168.4.129
} { ping $address }
```

You should get ICMP echo replies for every address pinged. Make sure to run the Tcl script on each router.

Step 5: Verify the current path.

Before you configure PBR, verify the routing table on R1.

a. On R1, use the **show ip route** command. Notice the next-hop IP address for all networks discovered by EIGRP.

```
R1# show ip route | begin Gateway
Gateway of last resort is not set

      172.16.0.0/16 is variably subnetted, 6 subnets, 2 masks
C        172.16.12.0/29 is directly connected, Serial0/0/0
L        172.16.12.1/32 is directly connected, Serial0/0/0
```

```
C        172.16.13.0/29 is directly connected, Serial0/0/1
L        172.16.13.1/32 is directly connected, Serial0/0/1
D        172.16.23.0/29 [90/21024000] via 172.16.12.2, 00:07:22,
Serial0/0/0
D        172.16.34.0/29 [90/41024000] via 172.16.13.3, 00:07:22,
Serial0/0/1
      192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.1.0/24 is directly connected, Loopback1
L        192.168.1.1/32 is directly connected, Loopback1
D     192.168.2.0/24 [90/20640000] via 172.16.12.2, 00:07:22,
Serial0/0/0
D     192.168.3.0/24 [90/21152000] via 172.16.12.2, 00:07:22,
Serial0/0/0
      192.168.4.0/25 is subnetted, 2 subnets
D        192.168.4.0 [90/41152000] via 172.16.13.3, 00:07:14,
Serial0/0/1
D        192.168.4.128 [90/41152000] via 172.16.13.3, 00:07:14,
Serial0/0/1
R1#
```

```
R1#show ip route | begin Gateway
Gateway of last resort is not set

     172.16.0.0/16 is variably subnetted, 6 subnets, 2 masks
C       172.16.12.0/29 is directly connected, Serial0/0/0
L       172.16.12.1/32 is directly connected, Serial0/0/0
C       172.16.13.0/29 is directly connected, Serial0/0/1
L       172.16.13.1/32 is directly connected, Serial0/0/1
D       172.16.23.0/29 [90/21024000] via 172.16.12.2, 00:31:36, Serial0/0/0
D       172.16.34.0/29 [90/41024000] via 172.16.13.3, 00:29:27, Serial0/0/1
     192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C       192.168.1.0/24 is directly connected, Loopback1
L       192.168.1.1/32 is directly connected, Loopback1
D    192.168.2.0/24 [90/20640000] via 172.16.12.2, 00:31:47, Serial0/0/0
D    192.168.3.0/24 [90/21152000] via 172.16.12.2, 00:29:36, Serial0/0/0
     192.168.4.0/25 is subnetted, 2 subnets
D       192.168.4.0/25 [90/41152000] via 172.16.13.3, 00:24:03, Serial0/0/1
D       192.168.4.128/25 [90/41152000] via 172.16.13.3, 00:24:03, Serial0/0/1
```

On R4, use the **traceroute** command to the R1 LAN address and source the ICMP packet from R4 LAN A and LAN B.

Note: You can specify the source as the interface address (for example 192.168.4.1) or the interface designator (for example, Fa0/0).

```
R4# traceroute 192.168.1.1 source 192.168.4.1
Type escape sequence to abort.
Tracing the route to 192.168.1.1
VRF info: (vrf in name/id, vrf out name/id)
  1 172.16.34.3 12 msec 12 msec 16 msec
  2 172.16.23.2 20 msec 20 msec 20 msec
  3 172.16.12.1 24 msec *24 msec
R4#
R4# traceroute 192.168.1.1 source 192.168.4.129
Type escape sequence to abort.
Tracing the route to 192.168.1.1
```

```
VRF info: (vrf in name/id, vrf out name/id)
  1 172.16.34.3 12 msec 16 msec 12 msec
  2 172.16.23.2 28 msec 20 msec 16 msec
  3 172.16.12.1 24 msec *24 msec
R4#
```

Notice that the path taken for the packets sourced from the R4 LANs are going through R3 --> R2 --> R1.

Why are the R4 interfaces not using the R3 --> R1 path?

La serie entre los routers R1 y R3 se ha configurado en un ancho de banda de 64 Kb/s. Todas la interfaces usan 128 Kb/s. De esta manera R3 por menos métrica envía los paquetes a R2.

On R3, use the **show ip route** command and note that the preferred route from R3 to R1 LAN 192.168.1.0/24 is via R2 using the R3 exit interface S0/0/1.

```
R3# show ip route | begin Gateway
Gateway of last resort is not set

      172.16.0.0/16 is variably subnetted, 7 subnets, 2 masks
D        172.16.12.0/29 [90/21024000] via 172.16.23.2, 00:10:54,
Serial0/0/1
C        172.16.13.0/29 is directly connected, Serial0/0/0
L        172.16.13.3/32 is directly connected, Serial0/0/0
C        172.16.23.0/29 is directly connected, Serial0/0/1
L        172.16.23.3/32 is directly connected, Serial0/0/1
C        172.16.34.0/29 is directly connected, Serial0/1/0
L        172.16.34.3/32 is directly connected, Serial0/1/0
D     192.168.1.0/24 [90/21152000] via 172.16.23.2, 00:10:54,
Serial0/0/1
D     192.168.2.0/24 [90/20640000] via 172.16.23.2, 00:10:54,
Serial0/0/1
      192.168.3.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.3.0/24 is directly connected, Loopback3
L        192.168.3.1/32 is directly connected, Loopback3
      192.168.4.0/25 is subnetted, 2 subnets
D        192.168.4.0 [90/40640000] via 172.16.34.4, 00:10:47,
Serial0/1/0
D        192.168.4.128 [90/40640000] via 172.16.34.4, 00:10:47,
Serial0/1/0
R3#
```

```
R3#show ip route | begin Gateway
Gateway of last resort is not set

     172.16.0.0/16 is variably subnetted, 7 subnets, 2 masks
D       172.16.12.0/29 [90/21024000] via 172.16.23.2, 00:33:15, Serial0/0/1
C       172.16.13.0/29 is directly connected, Serial0/0/0
L       172.16.13.3/32 is directly connected, Serial0/0/0
C       172.16.23.0/29 is directly connected, Serial0/0/1
L       172.16.23.3/32 is directly connected, Serial0/0/1
C       172.16.34.0/29 is directly connected, Serial0/1/0
L       172.16.34.3/32 is directly connected, Serial0/1/0
D    192.168.1.0/24 [90/21152000] via 172.16.23.2, 00:33:15, Serial0/0/1
D    192.168.2.0/24 [90/20640000] via 172.16.23.2, 00:33:15, Serial0/0/1
     192.168.3.0/24 is variably subnetted, 2 subnets, 2 masks
C       192.168.3.0/24 is directly connected, Loopback3
L       192.168.3.1/32 is directly connected, Loopback3
     192.168.4.0/25 is subnetted, 2 subnets
D       192.168.4.0/25 [90/40640000] via 172.16.34.4, 00:27:42, Serial0/1/0
D       192.168.4.128/25 [90/40640000] via 172.16.34.4, 00:27:42, Serial0/1/0
```

On R3, use the **show interfaces serial 0/0/0** and **show interfaces s0/0/1** commands.

```
R3# show interfaces serial0/0/0
Serial0/0/0 is up, line protocol is up
  Hardware is WIC MBRD Serial
  Description: R3 --> R1
  Internet address is 172.16.13.3/29
  MTU 1500 bytes, BW 64 Kbit/sec, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
  Last input 00:00:01, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output
drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     399 packets input, 29561 bytes, 0 no buffer
     Received 186 broadcasts (0 IP multicasts)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     393 packets output, 29567 bytes, 0 underruns
     0 output errors, 0 collisions, 3 interface resets
     0 unknown protocol drops
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up

R3# show interfaces serial0/0/0 | include BW
  MTU 1500 bytes, BW 64 Kbit/sec, DLY 20000 usec,
R3# show interfaces serial0/0/1 | include BW
  MTU 1500 bytes, BW 128 Kbit/sec, DLY 20000 usec,
R3#
```

```
R3#show interfaces serial0/0/0
Serial0/0/0 is up, line protocol is up (connected)
  Hardware is HD64570
  Description: R3 --> R1
  Internet address is 172.16.13.3/29
  MTU 1500 bytes, BW 64 Kbit, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation HDLC, loopback not set, keepalive set (10 sec)
  Last input never, output never, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0 (size/max/drops); Total output drops: 0
  Queueing strategy: weighted fair
  Output queue: 0/1000/64/0 (size/max total/threshold/drops)
     Conversations  0/0/256 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 48 kilobits/sec
  5 minute input rate 102 bits/sec, 0 packets/sec
  5 minute output rate 102 bits/sec, 0 packets/sec
     520 packets input, 31057 bytes, 0 no buffer
     Received 4 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     460 packets output, 27621 bytes, 0 underruns
     0 output errors, 0 collisions, 1 interface resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

Notice that the bandwidth of the serial link between R3 and R1 (S0/0/0) is set to 64 Kb/s, while the bandwidth of the serial link between R3 and R2 (S0/0/1) is set to 128 Kb/s.

Confirm that R3 has a valid route to reach R1 from its serial 0/0/0 interface using the **show ip eigrp topology 192.168.1.0** command.

```
R3# show ip eigrp topology 192.168.1.0
EIGRP-IPv4 Topology Entry for AS(1)/ID(192.168.3.1) for
192.168.1.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is
21152000
  Descriptor Blocks:
  172.16.23.2 (Serial0/0/1), from 172.16.23.2, Send flag is 0x0
      Composite metric is (21152000/20640000), route is Internal
      Vector metric:
        Minimum bandwidth is 128 Kbit
        Total delay is 45000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
        Originating router is 192.168.1.1
  172.16.13.1 (Serial0/0/0), from 172.16.13.1, Send flag is 0x0
      Composite metric is (40640000/128256), route is Internal
      Vector metric:
        Minimum bandwidth is 64 Kbit
        Total delay is 25000 microseconds
        Reliability is 255/255
        Load is 1/255
```

```
          Minimum MTU is 1500
          Hop count is 1
          Originating router is 192.168.1.1
R3#
```

```
R3#show ip eigrp topology 192.168.1.0
IP-EIGRP (AS 1): Topology entry for 192.168.1.0/24
  State is Passive, Query origin flag is 1, 1 Successor(s), FD is 21152000
  Routing Descriptor Blocks:
  172.16.23.2 (Serial0/0/1), from 172.16.23.2, Send flag is 0x0
      Composite metric is (21152000/20640000), Route is Internal
      Vector metric:
        Minimum bandwidth is 128 Kbit
        Total delay is 45000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 2
  172.16.13.1 (Serial0/0/0), from 172.16.13.1, Send flag is 0x0
      Composite metric is (40640000/128256), Route is Internal
      Vector metric:
        Minimum bandwidth is 64 Kbit
        Total delay is 25000 microseconds
        Reliability is 255/255
        Load is 1/255
        Minimum MTU is 1500
        Hop count is 1
```

As indicated, R4 has two routes to reach 192.168.1.0. However, the metric for the route to R1 (172.16.13.1) is much higher (40640000) than the metric of the route to R2 (21152000), making the route through R2 the successor route.

Step 6: Configure PBR to provide path control.

Now you will deploy source-based IP routing by using PBR. You will change a default IP routing decision based on the EIGRP-acquired routing information for selected IP source-to-destination flows and apply a different next-hop router.

Recall that routers normally forward packets to destination addresses based on information in their routing table. By using PBR, you can implement policies that selectively cause packets to take different paths based on source address, protocol type, or application type. Therefore, PBR overrides the router's normal routing behavior.

Configuring PBR involves configuring a route map with **match** and **set** commands and then applying the route map to the interface.

The steps required to implement path control include the following:

- Choose the path control tool to use. Path control tools manipulate or bypass the IP routing table. For PBR, **route-map** commands are used.
- Implement the traffic-matching configuration, specifying which traffic will be manipulated. The **match** commands are used within route maps.
- Define the action for the matched traffic using **set** commands within route maps.
- Apply the route map to incoming traffic.

As a test, you will configure the following policy on router R3:

- All traffic sourced from R4 LAN A must take the R3 --> R2 --> R1 path.
- All traffic sourced from R4 LAN B must take the R3 --> R1 path.

a. On router R3, create a standard access list called **PBR-ACL** to identify the R4 LAN B network.

```
R3(config)# ip access-list standard PBR-ACL
R3(config-std-nacl)# remark ACL matches R4 LAN B traffic
R3(config-std-nacl)# permit 192.168.4.128 0.0.0.127
R3(config-std-nacl)# exit
R3(config)#
```

```
R3(config)#ip access-list standard PBR-ACL
R3(config-std-nacl)#remark ACL matches R4 LAN B traffic
R3(config-std-nacl)#permit 192.168.4.128 0.0.0.127
R3(config-std-nacl)#exit
```

Create a route map called **R3-to-R1** that matches PBR-ACL and sets the next-hop interface to the R1 serial 0/0/1 interface.

```
R3(config)# route-map R3-to-R1 permit
R3(config-route-map)# description RM to forward LAN B traffic to R1
R3(config-route-map)# match ip address PBR-ACL
R3(config-route-map)# set ip next-hop 172.16.13.1
R3(config-route-map)# exit
R3(config)#
```

Apply the R3-to-R1 route map to the serial interface on R3 that receives the traffic from R4. Use the **ip policy route-map** command on interface S0/1/0.

```
R3(config)# interface s0/1/0
R3(config-if)# ip policy route-map R3-to-R1
R3(config-if)# end
R3#
```

On R3, display the policy and matches using the **show route-map** command.

```
R3# show route-map
route-map R3-to-R1, permit, sequence 10
  Match clauses:
    ip address (access-lists): PBR-ACL
  Set clauses:
    ip next-hop 172.16.13.1
  Policy routing matches: 0 packets, 0 bytes
R3#
```

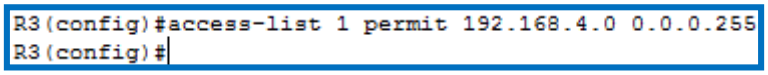
Note: There are currently no matches because no packets matching the ACL have passed through R3 S0/1/0.

Step 7: Test the policy.

Now you are ready to test the policy configured on R3. Enable the **debug ip policy** command on R3 so that you can observe the policy decision-making in action. To help filter the traffic, first create a standard ACL that identifies all traffic from the R4 LANs.

a. On R3, create a standard ACL which identifies all of the R4 LANs.

```
R3# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R3(config)# access-list 1 permit 192.168.4.0 0.0.0.255
R3(config)# exit
```



Enable PBR debugging only for traffic that matches the R4 LANs.

```
R3# debug ip policy ?
  <1-199>  Access list
  dynamic  dynamic PBR
  <cr>

R3# debug ip policy 1
Policy routing debugging is on for access list 1
```

Test the policy from R4 with the **traceroute** command, using R4 LAN A as the source network.

```
R4# traceroute 192.168.1.1 source 192.168.4.1

Type escape sequence to abort.
Tracing the route to 192.168.1.1

  1 172.16.34.3 0 msec 0 msec 4 msec
  2 172.16.23.2 0 msec 0 msec 4 msec
  3 172.16.12.1 4 msec 0 msec *
```

Notice the path taken for the packet sourced from R4 LAN A is still going through R3 --> R2 --> R1.

As the traceroute was being executed, router R3 should be generating the following debug output.

```
R3#
Jan 10 10:49:48.411: IP: s=192.168.4.1 (Serial0/1/0),
d=192.168.1.1, len 28, policy rejected -- normal forwarding
Jan 10 10:49:48.427: IP: s=192.168.4.1 (Serial0/1/0),
d=192.168.1.1, len 28, policy rejected -- normal forwarding
Jan 10 10:49:48.439: IP: s=192.168.4.1 (Serial0/1/0),
d=192.168.1.1, len 28, policy rejected -- normal forwarding
Jan 10 10:49:48.451: IP: s=192.168.4.1 (Serial0/1/0),
d=192.168.1.1, len 28, FIB policy rejected(no match) - normal
forwarding
Jan 10 10:49:48.471: IP: s=192.168.4.1 (Serial0/1/0),
d=192.168.1.1, len 28, FIB policy rejected(no match) - normal
forwarding
Jan 10 10:49:48.491: IP: s=192.168.4.1 (Serial0/1/0),
d=192.168.1.1, len 28, FIB policy rejected(no match) - normal
forwarding
Jan 10 10:49:48.511: IP: s=192.168.4.1 (Serial0/1/0),
d=192.168.1.1, len 28, FIB policy rejected(no match) - normal
forwarding
```

```
Jan 10 10:49:48.539: IP: s=192.168.4.1 (Serial0/1/0),
d=192.168.1.1, len 28, FIB policy rejected(no match) - normal
forwarding
Jan 10 10:49:51.539: IP: s=192.168.4.1 (Serial0/1/0),
d=192.168.1.1, len 28, FIB policy rejected(no match) - normal
forwarding
R3#
```

Why is the traceroute traffic not using the R3 --> R1 path as specified in the R3-to-R1 policy?

La LAN A no cumple con los criterios específicos en la lista de acceso PBR-ACL.

Test the policy from R4 with the **traceroute** command, using R4 LAN B as the source network.

```
R4# traceroute 192.168.1.1 source 192.168.4.129

Type escape sequence to abort.
Tracing the route to 192.168.1.1

  1 172.16.34.3 12 msec 12 msec 16 msec
  2 172.16.13.1 28 msec 28 msec *
```

Now the path taken for the packet sourced from R4 LAN B is R3 --> R1, as expected.

The debug output on R3 also confirms that the traffic meets the criteria of the R3-to-R1 policy.

```
R3#
R3#
Jan 10 10:50:04.283: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1, len 28, policy match
Jan 10 10:50:04.283: IP: route map R3-to-R1, item 10, permit
Jan 10 10:50:04.283: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1 (Serial0/0/0), len 28, policy routed
Jan 10 10:50:04.283: IP: Serial0/1/0 to Serial0/0/0 172.16.13.1
Jan 10 10:50:04.295: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1, len 28, policy match
Jan 10 10:50:04.295: IP: route map R3-to-R1, item 10, permit
Jan 10 10:50:04.295: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1 (Serial0/0/0), len 28, policy routed
Jan 10 10:50:04.295: IP: Serial0/1/0 to Serial0/0/0 172.16.13.1
Jan 10 10:50:04.311: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1, len 28, policy match
Jan 10 10:50:04.311: IP: route map R3-to-R1, item 10, permit
Jan 10 10:50:04.311: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1 (Serial0/0/0), len 28, policy routed
Jan 10 10:50:04.311: IP: Serial0/1/0 to Serial0/0/0 172.16.13.1
Jan 10 10:50:04.323: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1, len 28, FIB policy match
Jan 10 10:50:04.323: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1, len 28, PBR Counted
Jan 10 10:50:04.323: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1, g=172.16.13.1, len 28, FIB policy routed
Jan 10 10:50:04.351: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1, len 28, FIB policy match
Jan 10 10:50:04.351: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1, len 28, PBR Counted
```

```
Jan 10 10:50:04.351: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1, g=172.16.13.1, len 28, FIB policy routed
Jan 10 10:50:07.347: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1, len 28, FIB policy match
Jan 10 10:50:07.347: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1, len 28, PBR Counted
Jan 10 10:50:07.347: IP: s=192.168.4.129 (Serial0/1/0),
d=192.168.1.1, g=172.16.13.1, len 28, FIB policy routed
R3#
```

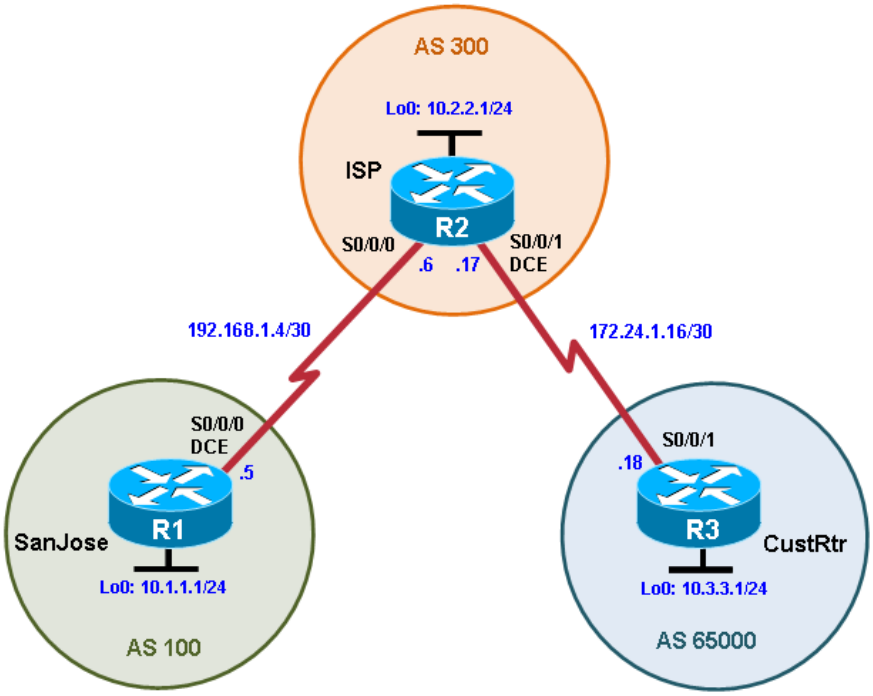
On R3, display the policy and matches using the **show route-map** command.

```
R3# show route-map
route-map R3-to-R1, permit, sequence 10
  Match clauses:
    ip address (access-lists): PBR-ACL
  Set clauses:
    ip next-hop 172.16.13.1
Nexthop tracking current: 0.0.0.0
172.16.13.1, fib_nh:0,oce:0,status:0

  Policy routing matches: 12 packets, 384 bytes
R3#
```

Note: There are now matches to the policy because packets matching the ACL have passed through R3 S0/1/0.

Using the AS_PATH Attribute

Topology



Step 0: Suggested starting configurations.

Apply the following configuration to each router along with the appropriate **hostname**. The **exec-timeout 0 0** command should only be used in a lab environment.

```
Router(config)# no ip domain-lookup
Router(config)# line con 0
Router(config-line)# logging synchronous
Router(config-line)# exec-timeout 0 0
```

Step 1: Configure interface addresses.

Using the addressing scheme in the diagram, create the loopback interfaces and apply IPv4 addresses to these and the serial interfaces on SanJose (R1), ISP (R2), and CustRtr (R3). The ISP loopbacks simulate real networks. Set a clock rate on the DCE serial interfaces.

```
SanJose(config)# interface Loopback0
SanJose(config-if)# ip address 10.1.1.1 255.255.255.0
SanJose(config-if)# exit
SanJose(config)# interface Serial0/0/0
SanJose(config-if)# ip address 192.168.1.5 255.255.255.252
SanJose(config-if)# clock rate 128000
SanJose(config-if)# no shutdown
SanJose(config-if)# end
SanJose#

ISP(config)# interface Loopback0
ISP(config-if)# ip address 10.2.2.1 255.255.255.0
ISP(config-if)# interface Serial0/0/0
ISP(config-if)# ip address 192.168.1.6 255.255.255.252
ISP(config-if)# no shutdown
ISP(config-if)# exit
ISP(config)# interface Serial0/0/1
ISP(config-if)# ip address 172.24.1.17 255.255.255.252
ISP(config-if)# clock rate 128000
ISP(config-if)# no shutdown
ISP(config-if)# end
ISP#

CustRtr(config)# interface Loopback0
CustRtr(config-if)# ip address 10.3.3.1 255.255.255.0
CustRtr(config-if)# exit
CustRtr(config)# interface Serial0/0/1
CustRtr(config-if)# ip address 172.24.1.18 255.255.255.252
CustRtr(config-if)# no shutdown
CustRtr(config-if)# end
CustRtr#
```

Use **ping** to test the connectivity between the directly connected routers.

Note: SanJose will not be able to reach either ISP's loopback (10.2.2.1) or CustRtr's loopback (10.3.3.1), nor will it be able to reach either end of the link joining ISP to CustRtr (172.24.1.17 and 172.24.1.18).

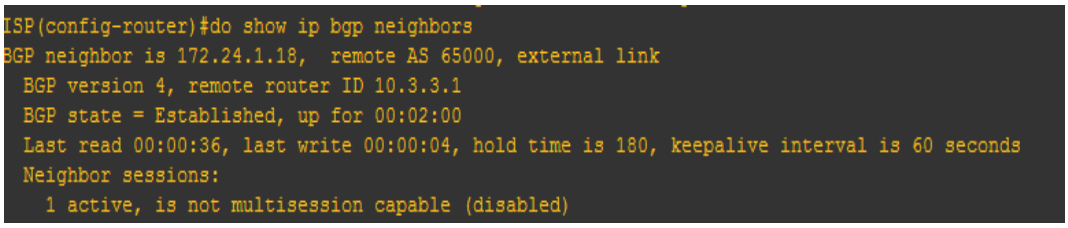
Step 2: Configure BGP.

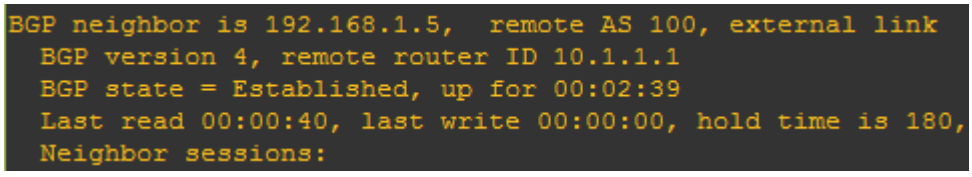
b. Configure BGP for normal operation. Enter the appropriate BGP commands on each router so that they identify their BGP neighbors and advertise their loopback networks.

```
SanJose(config)# router bgp 100
SanJose(config-router)# neighbor 192.168.1.6 remote-as 300
SanJose(config-router)# network 10.1.1.0 mask 255.255.255.0

ISP(config)# router bgp 300
ISP(config-router)# neighbor 192.168.1.5 remote-as 100
ISP(config-router)# neighbor 172.24.1.18 remote-as 65000
ISP(config-router)# network 10.2.2.0 mask 255.255.255.0

CustRtr(config)# router bgp 65000
CustRtr(config-router)# neighbor 172.24.1.17 remote-as 300
CustRtr(config-router)# network 10.3.3.0 mask 255.255.255.0
```

Verify that these routers have established the appropriate neighbor relationships by issuing the **show ip bgp neighbors** command on each router.

```
ISP# show ip bgp neighbors
BGP neighbor is 172.24.1.18,  remote AS 65000, external link
  BGP version 4, remote router ID 10.3.3.1
  BGP state = Established, up for 00:00:28
  Last read 00:00:28, last write 00:00:28, hold time is 180, keepalive
interval is 60 seconds
<output omitted>


BGP neighbor is 192.168.1.5,  remote AS 100, external link
  BGP version 4, remote router ID 10.1.1.1
  BGP state = Established, up for 00:01:34
  Last read 00:00:33, last write 00:00:06, hold time is 180, keepalive
interval is 60 seconds
<output omitted>
```





Step 3: Remove the private AS.

c. Display the SanJose routing table using the **show ip route** command. SanJose should have a route to both 10.2.2.0 and 10.3.3.0. Troubleshoot if necessary.

```
SanJose#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

Gateway of last resort is not set

      10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
C        10.1.1.0/24 is directly connected, Loopback0
L        10.1.1.1/32 is directly connected, Loopback0
B        10.2.2.0/24 [20/0] via 192.168.1.6, 00:04:22
B        10.3.3.0/24 [20/0] via 192.168.1.6, 00:03:14
      192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.1.4/30 is directly connected, Serial0/0/0
L        192.168.1.5/32 is directly connected, Serial0/0/0
```

```
Gateway of last resort is not set

      10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
C        10.1.1.0/24 is directly connected, Loopback0
L        10.1.1.1/32 is directly connected, Loopback0
B        10.2.2.0/24 [20/0] via 192.168.1.6, 00:04:29
B        10.3.3.0/24 [20/0] via 192.168.1.6, 00:03:58
      192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.1.4/30 is directly connected, Serial1/0
L        192.168.1.5/32 is directly connected, Serial1/0
SanJose(config)#
```

Ping the 10.3.3.1 address from SanJose.

Why does this fail?

Porque cuando R1, realiza un ping, lo hace a través de la interfaz s1/0, con dirección 192.168.1.5, cuando llega este paquete a R3, lo descarta ya que en su tabla de enrutamiento no existe esta red.

Ping again, this time as an extended ping, sourcing from the Loopback0 interface address.

```
SanJose# ping
Protocol [ip]:
Target IP address: 10.3.3.1
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: 10.1.1.1
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
```

```
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.3.3.1, timeout is 2 seconds:
Packet sent with a source address of 10.1.1.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max =
28/28/28 ms
SanJose#
```

Note: You can bypass extended ping mode and specify a source address using one of these commands:

```
SanJose# ping 10.3.3.1 source 10.1.1.1
```
or
```
SanJose# ping 10.3.3.1 source Lo0
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.3.3.1, timeout is 2 seconds:
Packet sent with a source address of 10.1.1.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/22/28 ms
SanJose#
```

Check the BGP table from SanJose by using the **show ip bgp** command. Note the AS path for the 10.3.3.0 network. The AS 65000 should be listed in the path to 10.3.3.0.

```
SanJose# show ip bgp
BGP table version is 5, local router ID is 10.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i -
internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f
  RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
  Origin codes: i - IGP, e - EGP, ? - incomplete
  RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  10.1.1.0/24      0.0.0.0                  0         32768 i
 *>  10.2.2.0/24      192.168.1.6              0             0 300
i
 *>  10.3.3.0/24      192.168.1.6                            0 300
65000 i
SanJose#
```

```
BGP table version is 4, local router ID is 10.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  10.1.1.0/24      0.0.0.0                  0         32768 i
 *>  10.2.2.0/24      192.168.1.6              0             0 300 i
 *>  10.3.3.0/24      192.168.1.6                            0 300 65000 i
SanJose#
```

Why is this a problem?

Por qué el router r3 con as privado 65000. No debe publicar su número de AS a los demás router, ya que provocaría un loop si se encontrara con otro router con el mismo AS privado.

Configure ISP to strip the private AS numbers from BGP routes exchanged with SanJose using the following commands.

```
ISP(config)# router bgp 300
ISP(config-router)# neighbor 192.168.1.5 remove-private-as
```

After issuing these commands, use the **clear ip bgp *** command on ISP to reestablish the BGP relationship between the three routers. Wait several seconds and then return to SanJose to check its routing table.

Note: The **clear ip bgp * soft** command can also be used to force each router to resend its BGP table.

```
ISP# clear ip bgp *
ISP#
*Sep  8 18:40:03.551: %BGP-5-ADJCHANGE: neighbor 172.24.1.18 Down
User reset
*Sep  8 18:40:03.551: %BGP_SESSION-5-ADJCHANGE: neighbor 172.24.1.18
IPv4 Unicast topology base removed from session  User reset
*Sep  8 18:40:03.551: %BGP-5-ADJCHANGE: neighbor 192.168.1.5 Down
User reset
*Sep  8 18:40:03.551: %BGP_SESSION-5-ADJCHANGE: neighbor 192.168.1.5
IPv4 Unicast topology base removed from session  User reset
*Sep  8 18:40:04.515: %BGP-5-ADJCHANGE: neighbor 172.24.1.18 Up
*Sep  8 18:40:04.519: %BGP-
ISP#5-ADJCHANGE: neighbor 192.168.1.5 Up
ISP#

 SanJose# show ip route
 <output omitted>

       10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
 C        10.1.1.0/24 is directly connected, Loopback0
 L        10.1.1.1/32 is directly connected, Loopback0
 B        10.2.2.0/24 [20/0] via 192.168.1.6, 00:00:20
 B        10.3.3.0/24 [20/0] via 192.168.1.6, 00:01:02
       192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
 C        192.168.1.4/30 is directly connected, Serial0/0/0
 L        192.168.1.5/32 is directly connected, Serial0/0/0
 SanJose#
```

```
Gateway of last resort is not set

      10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
C        10.1.1.0/24 is directly connected, Loopback0
L        10.1.1.1/32 is directly connected, Loopback0
B        10.3.3.0/24 [20/0] via 192.168.1.6, 00:00:06
      192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.1.4/30 is directly connected, Serial1/0
L        192.168.1.5/32 is directly connected, Serial1/0
SanJose#
```

Does SanJose still have a route to 10.3.3.0?

Si conoce la ruta 10.3.3.0/24 aprendida por la interfaz 192.168.1.6

SanJose should be able to ping 10.3.3.1 using its loopback 0 interface as the source of the ping.

```
SanJose# ping 10.3.3.1 source lo0

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.3.3.1, timeout is 2 seconds:
Packet sent with a source address of 10.1.1.1
   !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/28/32
ms
```

```
SanJose#ping 10.3.3.1 source lo0
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.3.3.1, timeout is 2 seconds:
Packet sent with a source address of 10.1.1.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 16/32/68 ms
SanJose#
```

Now check the BGP table on SanJose. The AS_ PATH to the 10.3.3.0 network should be AS 300. It no longer has the private AS in the path.

```
SanJose# show ip bgp
BGP table version is 9, local router ID is 10.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i
- internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f
RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  10.1.1.0/24      0.0.0.0                  0         32768 i
 *>  10.2.2.0/24      192.168.1.6              0             0 300
i
 *>  10.3.3.0/24      192.168.1.6                            0 300
i
SanJose#
```

```
SanJose#show ip bgp
BGP table version is 8, local router ID is 10.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  10.1.1.0/24      0.0.0.0                  0         32768 i
 *>  10.2.2.0/24      192.168.1.6              0             0 300 i
 *>  10.3.3.0/24      192.168.1.6                            0 300 i
```

Step 4: Use the AS_PATH attribute to filter routes.

As a final configuration, use the AS_PATH attribute to filter routes based on their origin. In a complex environment, you can use this attribute to enforce routing policy. In this case, the provider router, ISP, must be configured so that it does not propagate routes that originate from AS 100 to the customer router CustRtr.

AS-path access lists are read like regular access lists. The statements are read sequentially, and there is an implicit deny at the end. Rather than matching an address in each statement like a conventional access list, AS path access lists match on something called a regular expression. Regular expressions are a way of matching text patterns and have many uses. In this case, you will be using them in the AS path access list to match text patterns in AS paths.

d. Configure a special kind of access list to match BGP routes with an AS_PATH attribute that both begins and ends with the number 100. Enter the following commands on ISP.

```
ISP(config)# ip as-path access-list 1 deny ^100$
ISP(config)# ip as-path access-list 1 permit .*
```

The first command uses the **^** character to indicate that the AS path must begin with the given number 100. The **$** character indicates that the AS_PATH attribute must also end with 100. Essentially, this statement matches only paths that are sourced from AS 100. Other paths, which might include AS 100 along the way, will not match this list.

In the second statement, the **.** (period) is a wildcard, and the ***** (asterisk) stands for a repetition of the wildcard. Together, **.*** matches any value of the AS_PATH attribute, which in effect permits any update that has not been denied by the previous **access-list** statement.

For more details on configuring regular expressions on Cisco routers, see:

http://www.cisco.com/c/en/us/td/docs/ios/12_2/termserv/configuration/guide/ftersv_c/tcfaapre.html

http://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/13754-26.html

Apply the configured access list using the **neighbor** command with the **filter-list** option.

```
ISP(config)# router bgp 300
ISP(config-router)# neighbor 172.24.1.18 filter-list 1 out
```

The **out** keyword specifies that the list is applied to routing information sent to this neighbor.

Use the **clear ip bgp *** command to reset the routing information. Wait several seconds and then check the routing table for ISP. The route to 10.1.1.0 should be in the routing table.

Note: To force the local router to resend its BGP table, a less disruptive option is to use the **clear ip bgp * out** or **clear ip bgp * soft** command (the second command performs both outgoing and incoming route resync).

```
ISP# clear ip bgp *
ISP#
*Sep  8 18:48:04.915: %BGP-5-ADJCHANGE: neighbor 172.24.1.18 Down
User reset
*Sep  8 18:48:04.915: %BGP_SESSION-5-ADJCHANGE: neighbor
172.24.1.18 IPv4 Unicast topology base removed from session  User
reset
*Sep  8 18:48:04.915: %BGP-5-ADJCHANGE: neighbor 192.168.1.5 Down
User reset
*Sep  8 18:48:04.915: %BGP_SESSION-5-ADJCHANGE: neighbor
192.168.1.5 IPv4 Unicast topology base removed from session  User
reset
*Sep  8 18:48:04.951: %BGP-5-ADJCHANGE: neighbor 172.24.1.18 Up
*Sep  8 18:48:04.955: %BGP-
ISP#5-ADJCHANGE: neighbor 192.168.1.5 Up
ISP#

ISP# show ip route
<output omitted>

      10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
B        10.1.1.0/24 [20/0] via 192.168.1.5, 00:00:29
C        10.2.2.0/24 is directly connected, Loopback0
L        10.2.2.1/32 is directly connected, Loopback0
B        10.3.3.0/24 [20/0] via 172.24.1.18, 00:00:29
      172.24.0.0/16 is variably subnetted, 2 subnets, 2 masks
C        172.24.1.16/30 is directly connected, Serial0/0/1
L        172.24.1.17/32 is directly connected, Serial0/0/1
      192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.1.4/30 is directly connected, Serial0/0/0
L        192.168.1.6/32 is directly connected, Serial0/0/0
```

```
Gateway of last resort is not set

      10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
B        10.1.1.0/24 [20/0] via 192.168.1.5, 00:00:05
C        10.2.2.0/24 is directly connected, Loopback0
L        10.2.2.1/32 is directly connected, Loopback0
B        10.3.3.0/24 [20/0] via 172.24.1.18, 00:00:05
      172.24.0.0/16 is variably subnetted, 2 subnets, 2 masks
C        172.24.1.16/30 is directly connected, Serial1/1
L        172.24.1.17/32 is directly connected, Serial1/1
      192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.1.4/30 is directly connected, Serial1/0
L        192.168.1.6/32 is directly connected, Serial1/0
ISP#
```

Check the routing table for CustRtr. It should not have a route to 10.1.1.0 in its routing table.

```
CustRtr# show ip route
<output omitted>

      10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
B        10.2.2.0/24 [20/0] via 172.24.1.17, 00:00:32
C        10.3.3.0/24 is directly connected, Loopback0
L        10.3.3.1/32 is directly connected, Loopback0
      172.24.0.0/16 is variably subnetted, 2 subnets, 2 masks
C        172.24.1.16/30 is directly connected, Serial0/0/1
L        172.24.1.18/32 is directly connected, Serial0/0/1
```

```
Gateway of last resort is not set

      10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
B        10.2.2.0/24 [20/0] via 172.24.1.17, 00:00:09
C        10.3.3.0/24 is directly connected, Loopback0
L        10.3.3.1/32 is directly connected, Loopback0
      172.24.0.0/16 is variably subnetted, 2 subnets, 2 masks
C        172.24.1.16/30 is directly connected, Serial1/1
L        172.24.1.18/32 is directly connected, Serial1/1
CustRtr#
```

Return to ISP and verify that the filter is working as intended. Issue the **show ip bgp regexp ^100$** command.

```
ISP# show ip bgp regexp ^100$
BGP table version is 4, local router ID is 10.2.2.1
Status codes: s suppressed, d damped, h history, * valid, > best, i
- internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f
RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  10.1.1.0/24     192.168.1.5              0             0 100
i
```

```
ISP#show ip bgp regexp ^100$
BGP table version is 3, local router ID is 10.2.2.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  10.1.1.0/24     192.168.1.5              0             0 100 i
```

The output of this command shows all matches for the regular expressions that were used in the access list. The path to 10.1.1.0 matches the access list and is filtered from updates to CustRtr.

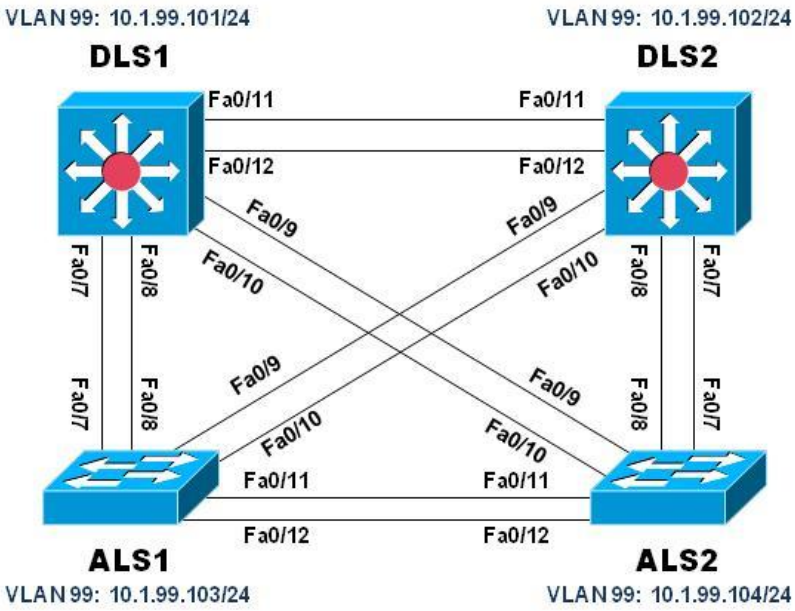
Run the following Tcl script on all routers to verify whether there is connectivity. All pings from ISP should be successful. SanJose should not be able to ping the CustRtr loopback 10.3.3.1 or the WAN link 172.24.1.16/30. CustRtr should not be able to ping the SanJose loopback 10.1.1.1 or the WAN link 192.168.1.4/30.

```
ISP# tclsh

foreach address {
10.1.1.1
10.2.2.1
10.3.3.1
192.168.1.5
192.168.1.6
172.24.1.17
172.24.1.18
} {
ping $address }
```

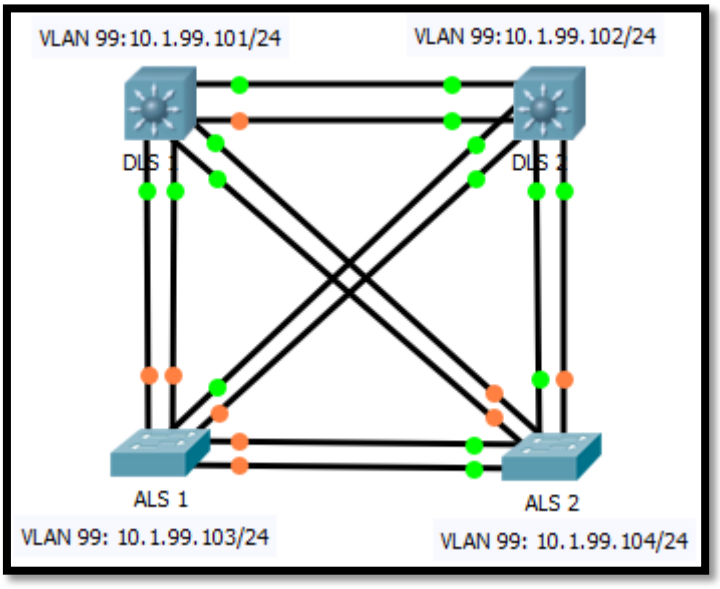
```
+>10.3.3.1
+>192.168.1.5
+>192.168.1.6
+>172.24.1.17
+>172.24.1.18
+>} {
+>ping $address }
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.1.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/9/16 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.2.2.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.3.3.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/16/28 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.1.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 8/10/16 ms
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.1.6, timeout is 2 seconds:
```

Static VLANS, Trunking, and VTP

Topology



All Switch-to-Switch connections are 802.1Q trunks



Prepare for the Lab

Prepare the switches for the lab

Use the **reset.tcl** script you created in Lab 1 “Preparing the Switch” to set your switches up for this lab. Then load the file BASE.CFG into the running-config with the command **copy flash:BASE.CFG running-config**. An example from DLS1:

```
DLS1# tclsh reset.tcl
Erasing the nvram filesystem will remove all configuration files! Continue?
[confirm]
[OK]
Erase of nvram: complete
Reloading the switch in 1 minute, type reload cancel to halt

Proceed with reload? [confirm]

*Mar  7 18:41:40.403: %SYS-7-NV_BLOCK_INIT: Initialized the geometry of
nvram
*Mar  7 18:41:41.141: %SYS-5-RELOAD: Reload requested by console. Reload
Reason: Reload command.
<switch reloads - output omitted>

Would you like to enter the initial configuration dialog? [yes/no]: n
Switch> en
*Mar  1 00:01:30.915: %LINK-5-CHANGED: Interface Vlan1, changed state to
administratively down
Switch# copy BASE.CFG running-config
Destination filename [running-config]?
184 bytes copied in 0.310 secs (594 bytes/sec)
DLS1#
```

Configure basic switch parameters.

Configure an IP address on the management VLAN according to the diagram. VLAN 1 is the default management VLAN, but following best practice, we will use a different VLAN. In this case, VLAN 99.

Enter basic configuration commands on each switch according to the diagram.

DLS1 example:

```
DLS1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
DLS1(config)# interface vlan 99
DLS1(config-if)# ip address 10.1.99.101 255.255.255.0
DLS1(config-if)# no shutdown
```

```
Switch>en
Switch#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#hostname DLS1
DLS1(config)#interface vlan 99
DLS1(config-if)#ip address 10.1.99.101 255.255.255.0
DLS1(config-if)#no shutdown
```

The interface VLAN 99 will not come up immediately, because the broadcast domain it is associated with (VLAN 99) doesn't exist on the switch. We will fix that in a few moments.

(Optional) On each switch, create an enable secret password and configure the VTY lines to allow remote access from other network devices.

DLS1 example:

```
DLS1(config)# enable secret class
DLS1(config)# line vty 0 15
DLS1(config-line)# password cisco
DLS1(config-line)# login
```

```
DLS1(config-if)#enable secret class
DLS1(config)#line vty 0 15
DLS1(config-line)#password cisco
DLS1(config-line)#login
```

Note: The passwords configured here are required for NETLAB compatibility only and are NOT recommended for use in a live environment.

Configure VTP Version 2, VLANs, and Trunking.

A VTP domain, also called a *VLAN management domain*, consists of trunked switches that are under the same administrative responsibility sharing the same VTP domain name. A switch can be in only one VTP domain, and VLAN database contents in the domain are globally synchronized. VLAN information is not propagated until a domain name is specified and trunks are set up between the devices.

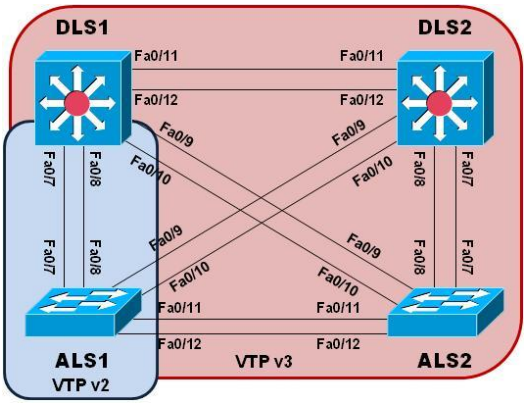
There are three versions of VTP available; Version 1 and 2 are able to support normal-range VLANs only, while version 3 can support normal- and extended-range VLANs, as well as the synchronization of other databases. Support for version 3 on the Catalyst platforms used in this lab was added in IOS version 12.2(52)SE. Older IOS versions do not generally support VTP version 3.

Switches operate in one of four VTP ***modes***. The default VTP mode for the 2960 and 3560 switches is server mode**, however our Lab 1 configuration changes this to transparent**.

VTP Mode	**Description**
VTP Server	You can create, modify, and delete VLANs and specify other configuration parameters, such as VTP version and VTP pruning, for the entire VTP domain. VTP servers advertise their VLAN configuration to other switches in the same VTP domain and synchronize their VLAN configuration with other switches based on advertisements received over trunk links. VTP server is the default mode.

	In VTP Server mode, VLAN configurations are only stored in the flash:vlan.dat file. While VLANs are manipulated in the configuration mode, the configuration commands do not appear in the running-config.
VTP Client	A VTP client behaves like a VTP server and transmits and receives VTP updates on its trunks, but you cannot create, change, or delete VLANs on a VTP client. VLANs are configured on another switch in the domain that is in server mode. In VTP Client mode, VLAN configurations are only stored in the flash:vlan.dat file. The configuration of VLANs does not appear in the running-config.
VTP Transparent	VTP transparent switches do not participate in VTP. A VTP transparent switch does not advertise its VLAN database nor synchronize its VLAN database based on received advertisements. However, transparent switches forward received VTP messages under two circumstances: either the VTP domain name of the transparent switch is empty (not yet configured), or it matches the domain name in the received VTP messages. In VTP Transparent mode, VLAN configurations are stored both in flash:vlan.dat file and also are present in the running-config. If extended range VLANs are used, however, they are stored in the flash:vlan.dat only if the switch is running VTP version 3.
VTP Off	A switch in VTP Off mode functions in the same manner as a VTP transparent switch, except that it does not forward VTP advertisements on trunks. VTP off is only available on switches that support VTP version 3 although it is not necessary to run VTP version 3 on the switch to be able to put it into the Off mode. In VTP Off mode, VLAN configurations are stored both in flash:vlan.dat file and also are present in the running-config. If extended range VLANs are used, however, they are stored in the flash:vlan.dat only if the switch is running VTP version 3.

In this lab we will demonstrate the configuration and operation of both VTP versions 2 and 3. We will do this by first configuring VTP version 2 between DLS1 and ALS1, and then configuring DLS1, DLS2 and ALS2 with VTP version 3



Verify VTP status

Issue the **`show vtp status`** command on DLS1

```
DLS1# show vtp status
VTP Version capable             : 1 to 3
VTP version running             : 1
VTP Domain Name                 :
VTP Pruning Mode                : Disabled
VTP Traps Generation            : Disabled
Device ID                       : 64a0.e72a.2200
Configuration last modified by 0.0.0.0 at 0-0-00 00:00:00

Feature VLAN:
--------------
VTP Operating Mode                : Transparent
Maximum VLANs supported locally   : 1005
Number of existing VLANs          : 5
Configuration Revision            : 0
MD5 digest                        : 0x57 0xCD 0x40 0x65 0x63 0x59 0x47 0xBD
                                    0x56 0x9D 0x4A 0x3E 0xA5 0x69 0x35 0xBC
```

```
DLS1#show vtp status
VTP Version capable             : 1 to 3
VTP version running             : 2
VTP Domain Name                 :
VTP Pruning Mode                : Disabled
VTP Traps Generation            : Disabled
Device ID                       : 0001.43E5.9590
Configuration last modified by 0.0.0.0 at 0-0-00 00:00:00
Local updater ID is 0.0.0.0 (no valid interface found)

Feature VLAN :
--------------
VTP Operating Mode                : Server
Maximum VLANs supported locally   : 1005
Number of existing VLANs          : 5
Configuration Revision            : 0
MD5 digest                        : 0x7D 0x5A 0xA6 0x0E 0x9A 0x72 0xA0 0x3A
                                    0xF0 0x58 0x10 0x6C 0x9C 0x0F 0xA0 0xF7
```

Because no VLAN configurations were made, all settings except the VTP mode that was changed in Lab 1 are the defaults. This switch is capable of running version 1, 2 or 3 of VTP and runs version 1 by default. All switches in the VTP domain must run the same VTP version. The VTP mode is set to transparent as a result of steps performed in Lab 1. The number of existing VLANs is the five built-in VLANs. Different switches in the Catalyst family support different numbers of local VLANs. The 3560 switch used in this lab supports a maximum of 1,005 VLANs locally, while the 2960 switch used in this lab supports at most 255 VLANs. Lastly, note that the configuration revision is 0.

As you should recall from CCNA, the configuration revision number is compared amongst VTPv1 or VTPv2 switches and the VLAN database from the switch with the highest revision number is adopted by all the other switches in the VLAN management domain. Every time VLAN information is modified and saved in the VLAN database (vlan.dat), the revision number is increased by one when the user exits from VLAN configuration mode.

In VTPv3, revision numbers are still used but they no longer determine the switch whose database is going to apply to the entire domain. Instead, a single designated switch in a VTP domain called the *primary server* is allowed to assert its database in the entire VTP domain, even if its own revision number is lower. Other switches that are not primary servers are not allowed to assert their databases even if their revision numbers are higher.

Multiple switches in the VTP domain can be in VTP server mode. In VTPv1 and VTPv2, any of these server switches can be used to manage all other switches in the VTP domain. In VTPv3, a single primary server for a particular VTP domain is designated to control where changes originate from in the switched network. This enables careful management and protection of the VLAN database.

Configure VTP on DLS1.

We will start off this lab by configuring DLS1 for VTP Server mode and setting the VTP domain name and VTP version 2. We will also set a VTP password, which provides some rudimentary protection against automatic VLAN database propagation. Because this password is set, VTPv2 will not allow ALS1 to automatically learn the domain name once trunks are installed.

```
DLS1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
DLS1(config)# vtp domain SWLAB
Changing VTP domain name from NULL to SWLAB
DLS1(config)# vtp version 2
DLS1(config)# vtp mode server
Setting device to VTP Server mode for VLANS.
DLS1(config)# vtp password cisco123
Setting device VTP password to cisco123
DLS1(config)#
*Mar  1 00:29:10.895: %SW_VLAN-6-VTP_DOMAIN_NAME_CHG: VTP domain name
changed to SWLAB.
```

```
DLS1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
DLS1(config)#vtp domain SWLAB
Changing VTP domain name from NULL to SWLAB
DLS1(config)#vtp version 2
DLS1(config)#vtp mode server
Device mode already VTP SERVER.
DLS1(config)#password cisco123
```

Verify these settings by using the **`show vtp status`** command again.

```
DLS1# show vtp status
VTP Version capable             : 1 to 3
VTP version running             : 2
VTP Domain Name                 : SWLAB
VTP Pruning Mode                : Disabled
VTP Traps Generation            : Disabled
Device ID                       : 64a0.e72a.2200
Configuration last modified by 0.0.0.0 at 0-0-00 00:00:00
Local updater ID is 0.0.0.0 (no valid interface found)

Feature VLAN:
--------------
```

```
VTP Operating Mode                  : Server
Maximum VLANs supported locally     : 1005
Number of existing VLANs            : 5
Configuration Revision              : 0
MD5 digest                          : 0xA7 0xE6 0xAF 0xF9 0xFE 0xA0 0x88 0x6B
                                      0x21 0x6D 0x70 0xEE 0x04 0x6D 0x90 0xF3
```

```
DLS1#show vtp status
VTP Version capable             : 1 to 3
VTP version running             : 2
VTP Domain Name                 : SWLAB
VTP Pruning Mode                : Disabled
VTP Traps Generation            : Disabled
Device ID                       : 0001.43E5.9590
Configuration last modified by 0.0.0.0 at 3-1-93 00:37:56
Local updater ID is 0.0.0.0 (no valid interface found)

Feature VLAN :
--------------
VTP Operating Mode                : Server
Maximum VLANs supported locally   : 1005
Number of existing VLANs          : 5
Configuration Revision            : 1
MD5 digest                        : 0xE5 0x3B 0xAD 0x37 0x0D 0x29 0xDD 0x3D
                                    0xF4 0xC9 0x20 0x20 0x44 0xBE 0x3F 0x8C
```

Configure VLANs on DLS1

Next configure the VLANs that will be required to support the network. There are two ways to create VLANs, either directly via the **vlan** command or by assigning an interface to a non-existent VLAN. For now, you will create the VLANs directly on the switch. Create:

VLAN 99 to enable the management interface.

VLAN 999 as a “parking lot” VLAN for unused access ports

Suspend this VLAN to prevent ports in the VLAN from every communicating with each other.

The VLANs required for network operations, which are VLANs 100, 110, and 120.

Suspending a VLAN deserves a special mention. Each VLAN has an operational state associated with it: it can be either active (the default state) or suspended. A suspended VLAN exists but it does not operate. Access ports assigned to a suspended VLAN drop all frames and are unable to communicate, similar to ports put into a nonexistent VLAN. Putting a suspended VLAN back into the active state reinstates normal communication on ports in that VLAN.

To globally *suspend* a VLAN, use the **state suspend** command in the VLAN configuration mode. This state is propagated by VTP to all other switches in the VTP domain if VTP is in use.

To locally *shut down* a VLAN, use the **shutdown** command in the VLAN configuration mode. This setting is not propagated through VTP.

Do not confuse the **shutdown** command in the VLAN configuration mode with the same command available under **interface Vlan** mode, which has a different and unrelated meaning. Further discussion on suspending and reactivating VLANs can be found in Part 3, Step 7 of this lab.

```
DLS1(config)# vlan 99
DLS1(config-vlan)# name MANAGEMENT
```

```
DLS1(config-vlan)# vlan 100
DLS1(config-vlan)# name SERVERS
DLS1(config-vlan)# vlan 110
DLS1(config-vlan)# name GUEST
DLS1(config-vlan)# vlan 120
DLS1(config-vlan)# name OFFICE
DLS1(config-vlan)# vlan 999
DLS1(config-vlan)# name PARKING_LOT
DLS1(config-vlan)# state suspend
DLS1(config-vlan)# vlan 666
DLS1(config-vlan)# name NATIVE_DO_NOT_USE
DLS1(config-vlan)# exit
```

```
DLS1(config)#vlan 99
DLS1(config-vlan)#
%LINK-5-CHANGED: Interface Vlan99, changed state to up

DLS1(config-vlan)#name MANAGEMENT
DLS1(config-vlan)#vlan 100
DLS1(config-vlan)#name SERVERS
DLS1(config-vlan)#vlan 110
DLS1(config-vlan)#name GUEST
DLS1(config-vlan)#vlan 120
DLS1(config-vlan)#name OFFICE
DLS1(config-vlan)#vlan 999
DLS1(config-vlan)#name PARKING_LOT
DLS1(config-vlan)#state suspend
DLS1(config-vlan)#vlan 666
DLS1(config-vlan)#name NATIVE_DO_NOT_USE
DLS1(config-vlan)#exit
```

The VLANs will not appear in the VLAN database until the **exit** command is issued.

After configuring the VLANs, issue the **show vtp status** command and you will see that the all-important configuration revision number has increased based on these changes to the VLAN database. Note that the revision number you have when performing this lab may be different.

```
DLS1#show vtp status | include Configuration Revision
Configuration Revision          : 6
```

Configure trunking on DLS1

VTP will only propagate information over trunks. Cisco switches support Dynamic Trunking Protocol (DTP), which allows automatic negotiation of trunks. The partial output here from ALS1 shows you the default trunking mode:

```
DLS1#show interface f0/7 switchport
Name: Fa0/7
Switchport: Enabled
Administrative Mode: dynamic auto
Operational Mode: down
Administrative Trunking Encapsulation: dot1q
Negotiation of Trunking: On
Access Mode VLAN: 1 (default)
```

```
Trunking Native Mode VLAN: 1 (default)
Administrative Native VLAN tagging: enabled
```

```
DLS1#show interface f0/7 switchport
Name: Fa0/7
Switchport: Enabled
Administrative Mode: dynamic auto
Operational Mode: static access
Administrative Trunking Encapsulation: negotiated
Operational Trunking Encapsulation: native
Negotiation of Trunking: On
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 1 (default)
Voice VLAN: none
```

Switches that are interconnected and have DTP enabled can form a trunk automatically if either end is in the dynamic desirable mode or static trunk mode on the condition that either both switches use the same VTP domain name or at least one of the switches does not yet have the VTP domain name configured.

The dynamic auto mode on both ends will prevent a trunk from automatically forming; however, this is not really a valid safeguard against unintentional trunk connections as the port can become a trunk if the other side changes to dynamic desirable or static trunk mode.

As a best practice you should configure each interface into either access or trunk mode and use the **`switchport nonegotiate`** interface configuration command to disable the propagation of DTP messages. Never leave ports to operate in the dynamic mode.

Configure the appropriate interfaces on DLS1 to be trunks.

Because DLS1 is a 3560, you must first specify the encapsulation protocol for the interface. Catalyst 3560 switches support ISL (Inter-Switch Link) and 802.1Q encapsulations for trunk interfaces. In the topology, all the trunks are 802.1Q trunks.

Change the native VLAN from the default of VLAN 1 to VLAN 666.

Set the interfaces to be in trunking mode only, and include the **`switchport nonegotiate`** command.

The **`no shutdown`** command is needed because the Lab 1 configuration has all interfaces shutdown.

```
DLS1(config)# interface range f0/7-12
DLS1(config-if-range)# switchport trunk encapsulation dot1q
DLS1(config-if-range)# switchport trunk native vlan 666
DLS1(config-if-range)# switchport mode trunk
DLS1(config-if-range)# switchport nonegotiate
DLS1(config-if-range)# no shutdown
DLS1(config-if-range)#
```

```
DLS1(config)# interface range f0/7-12
DLS1(config-if-range)# switchport trunk encapsulation dot1q
DLS1(config-if-range)# switchport trunk native vlan 666
DLS1(config-if-range)# switchport mode trunk
DLS1(config-if-range)# switchport nonegotiate
DLS1(config-if-range)# no shutdown
DLS1(config-if-range)#
```

By default, all VLANs are allowed on all trunks. You can explicitly control which VLANs are allowed on a trunk by using the **`switchport trunk allowed vlan`** *`vlan-id`* command on the interface at each end of the trunk.

There are several approaches to deciding what VLANs to allow or disallow to cross the trunk. Common practice is to disallow VLAN 1 and the PARKING_LOT vlan. You could go a step further and disallow any unused VLAN numbers, but you would then have to modify all the trunks should you later add a new VLAN to the network.

In this lab, disallowing the PARKING_LOT VLAN from all trunks is not really necessary since the VLAN has been suspended. Disallowing the VLAN can serve as an additional protection against inadvertent reactivation of this VLAN.

Disallowing VLAN 1, also referred to as VLAN 1 Minimization, excludes VLAN 1 from the trunk but does not restrict layer 2 management traffic (such as CDP, LLDP, VTP, STP, etc) from passing.

Since only these 2 VLANs are being disallowed, the **`except`** option of the command can be used:

```
DLS1(config-if-range)# switchport trunk allowed vlan ?
  WORD    VLAN IDs of the allowed VLANs when this port is in trunking mode
  add     add VLANs to the current list
  all     all VLANs
  except  all VLANs except the following
  none    no VLANs
  remove  remove VLANs from the current list

DLS1(config-if-range)# switchport trunk allowed vlan except 1,999
DLS1(config-if-range)#
```

Validate these settings by examining the switchport configuration for one of the trunk interfaces:

```
DLS1#show interface f0/7 switchport
Name: Fa0/7
Switchport: Enabled
Administrative Mode: trunk
Operational Mode: trunk
Administrative Trunking Encapsulation: dot1q
Operational Trunking Encapsulation: dot1q
Negotiation of Trunking: Off
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 666 (NATIVE_DO_NOT_USE)
```

```
Administrative Native VLAN tagging: enabled
<output omitted>
Trunking VLANs Enabled: 2-998,1000-4094
<output omitted>
```

Configure VTP and trunking on ALS1

Next configure VTP and trunking on ALS1. Configure ALS1 to be in VTP Client mode and then configure all of the appropriate trunk interfaces to use a native VLAN of 666 and to be in trunking mode only. The native VLAN number does not have to be the same across your network, but it must match between switches on a given connected trunk. Also, disallow VLANs 1 and 999.

```
ALS1(config)# vtp mode client
Setting device to VTP Client mode for VLANS.
ALS1(config)# interface range f0/7-12
ALS1(config-if-range)# switchport trunk native vlan 666
ALS1(config-if-range)# switchport mode trunk
ALS1(config-if-range)# switchport nonegotiate
ALS1(config-if-range)# switchport trunk allowed vlan except 1,999
ALS1(config-if-range)# no shutdown
ALS1(config-if-range)# exit
ALS1(config)#
```


```
ALS1(config)# vtp mode client
Setting device to VTP Client mode for VLANS.
ALS1(config)# interface range f0/7-12
ALS1(config-if-range)# switchport trunk native vlan 666
ALS1(config-if-range)# switchport mode trunk
ALS1(config-if-range)# switchport nonegotiate
ALS1(config-if-range)# switchport trunk allowed vlan except 1,999
ALS1(config-if-range)# no shutdown
ALS1(config-if-range)# exit
ALS1(config)#
```


After activating the interfaces, use the **`show interface trunk`** command to see the status of the trunks. You should see interfaces Fa0/7 and Fa0/8 in trunking mode.

```
ALS1# show interface trunk

Port        Mode             Encapsulation  Status        Native vlan
Fa0/7       on               802.1q         trunking      666
Fa0/8       on               802.1q         trunking      666

Port        Vlans allowed on trunk
Fa0/7       2-998,1000-4094
Fa0/8       2-998,1000-4094

Port        Vlans allowed and active in management domain
Fa0/7       none
```

```
Fa0/8       none

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/7       none
Fa0/8       none
ALS1#
```

```
ALS1# show interface trunk

Port        Mode             Encapsulation  Status        Native vlan
Fa0/7       on               802.1q         trunking      666
Fa0/8       on               802.1q         trunking      666

Port        Vlans allowed on trunk
Fa0/7       2-998,1000-4094
Fa0/8       2-998,1000-4094

Port        Vlans allowed and active in management domain
Fa0/7       none
Fa0/8       none

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/7       none
Fa0/8       none
ALS1#
```

Now if you look at the VTP status on ALS1, you will see the values are at their defaults, even though the trunk is operational. This is because of the VTP password.

```
ALS1# show vtp status
VTP Version capable             : 1 to 3
VTP version running             : 1
VTP Domain Name                 :
VTP Pruning Mode                : Disabled
VTP Traps Generation            : Disabled
Device ID                       : 64a0.e72a.2200
Configuration last modified by 0.0.0.0 at 0-0-00 00:00:00

Feature VLAN:
--------------
VTP Operating Mode                : Client
Maximum VLANs supported locally   : 255
Number of existing VLANs          : 5
Configuration Revision            : 0
MD5 digest                        : 0x57 0xCD 0x40 0x65 0x63 0x59 0x47 0xBD
                                    0x56 0x9D 0x4A 0x3E 0xA5 0x69 0x35 0xBC
```

Set the VTP password on ALS1 and the VLAN database will be synchronized. However, before you can set the password, the VTP domain name must be manually configured.

```
ALS1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
ALS1(config)# vtp domain SWLAB
Changing VTP domain name from NULL to SWLAB
ALS1(config)# vtp password cisco123
Setting device VTP password to cisco123
ALS1(config)# end
*Mar  1 00:27:21.902: %SW_VLAN-6-VTP_DOMAIN_NAME_CHG: VTP domain name
changed to SWLAB.
```

```
ALS1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
ALS1(config)#vtp domain SWLAB
Changing VTP domain name from NULL to SWLAB
ALS1(config)#vtp password cisco123
Setting device VLAN database password to cisco123
ALS1(config)#end
ALS1#
```

Now check the VTP status and you will see a revision number matching that of DLS1, and that VLANs 99, 100, 110, 120, 666 and 999 are all in the local VLAN database.

```
ALS1# show vtp status
VTP Version capable             : 1 to 3
VTP version running             : 2
VTP Domain Name                 : SWLAB
VTP Pruning Mode                : Disabled
VTP Traps Generation            : Disabled
Device ID                       : 64a0.e72a.2200
Configuration last modified by 0.0.0.0 at 3-1-93 00:04:37

Feature VLAN:
--------------
VTP Operating Mode                : Client
Maximum VLANs supported locally   : 255
Number of existing VLANs          : 11
Configuration Revision            : 6
MD5 digest                        : 0xF3 0x8A 0xEA 0xFA 0x9B 0x39 0x6D 0xF5
                                    0xA6 0x03 0x2F 0xB8 0x16 0xC1 0xE6 0x8C
```

```
ALS1#show vtp status
VTP Version                     : 2
Configuration Revision          : 0
Maximum VLANs supported locally : 255
Number of existing VLANs        : 5
VTP Operating Mode              : Client
VTP Domain Name                 : SWLAB
VTP Pruning Mode                : Disabled
VTP V2 Mode                     : Disabled
VTP Traps Generation            : Disabled
MD5 digest                      : 0x08 0x65 0xCA 0x33 0x22 0xCC 0x46 0xB0
Configuration last modified by 0.0.0.0 at 0-0-00 00:00:00
ALS1#
```

```
ALS1# show vlan brief | incl active
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
99   MANAGEMENT                       active
100  SERVERS                          active
110  GUEST                            active
120  OFFICE                           active
666  NATIVE_DO_NOT_USE                active
ALS1#
```

VLAN 999 will be missing from the filtered output above because it only includes VLANs in active state and VLAN 999 is suspended. Using the **`show vlan brief`** without filtering would show the VLAN 999.

You will also see that the configured VLANs except VLANs 1 and 999 are allowed over the trunks

```
ALS1#show interface trunk

Port        Mode             Encapsulation  Status        Native vlan
Fa0/7       on               802.1q         trunking      666
Fa0/8       on               802.1q         trunking      666

Port        Vlans allowed on trunk
Fa0/7       2-998,1000-4094
Fa0/8       2-998,1000-4094

Port        Vlans allowed and active in management domain
Fa0/7       99-100,110,120,666,999
Fa0/8       99-100,110,120,666,999

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/7       99-100,110,120,666
Fa0/8       99-100,110,120,666
ALS1#
```

```
ALS1#show interface trunk

Port        Mode             Encapsulation  Status        Native vlan
Fa0/7       on               802.1q         trunking      666
Fa0/8       on               802.1q         trunking      666

Port        Vlans allowed on trunk
Fa0/7       2-998,1000-4094
Fa0/8       2-998,1000-4094

Port        Vlans allowed and active in management domain
Fa0/7       99-100,110,120,666,999
Fa0/8       99-100,110,120,666,999

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/7       99-100,110,120,666
Fa0/8       99-100,110,120,666
ALS1#
```

You will also see that the state of interface VLAN 99 has changed to ‘up’.

```
*Mar  1 00:27:52.336: %LINEPROTO-5-UPDOWN: Line protocol on Interface
Vlan99, changed state to up
```

Because ALS1 is in VTP Client mode, local changes to the VLAN database cannot be made:

```
ALS1(config)# vlan 199
VTP VLAN configuration not allowed when device is in CLIENT mode.
ALS1(config)#
```

At this point, VTP version 2 is working and secured between DLS1 and ALS1. You should now be able to ping DLS1 from ALS1 and vice versa.

Park unused interfaces

On DLS1 and ALS1, place all of interfaces that will not be used into the PARKING_LOT VLAN and shut them down. For this lab, the interfaces being used on all switches are F0/6 through F0/12.

An example from DLS1:

```
DLS1(config)# interface range f0/1-5,f0/13-24,g0/1-2
DLS1(config-if-range)# switchport mode access
DLS1(config-if-range)# switchport nonegotiate
DLS1(config-if-range)# switchport access vlan 999
DLS1(config-if-range)# shutdown
DLS1(config-if-range)# exit
```

```
DLS1(config)# interface range f0/1-5,f0/13-24,g0/1-2
DLS1(config-if-range)# switchport mode access
DLS1(config-if-range)# switchport nonegotiate
DLS1(config-if-range)# switchport access vlan 999
DLS1(config-if-range)# shutdown
DLS1(config-if-range)# exit
```

Configure VTP version 3, VLANs, and Trunking

In this part of the lab you will configure VTP version 3 to operate across the rest of the switched network. VTP version 3 provides some significant benefits to the network administrator.

1. The concept of a primary server was added. In VTP versions 1 and 2, all VTP server switches are equal; any one of them may add/remove/rename VLANs and change their state. In VTP version 3, only the primary server can do this. There can be at most one *primary* server present in a VTP domain. The role of a *primary* server is a runtime state. It is not a part of the configuration; rather, this state is requested in privileged EXEC mode and is relinquished whenever another switch attempts to become the primary server or when the switch is reloaded.
2. VTP version 3 has the ability to hide the VTP password. On a VTP version 1 or 2 switch, issuing the command **`show vtp password`** will show the password to you in plain text. VTP version 3 allows you to specify that the password be hidden in the output, preventing the password from being inadvertently or maliciously divulged.
3. VTP version 3 can propagate information about extended-range VLANs; VLANs numbered between 1006 and 4094. To support these VLANs with VTP version 2, all switches had to be in transparent mode and the VLANs had to be configured manually on a switch-by-switch basis.
4. VTP version 3 only supports pruning for normal-range VLANs.
5. VTP version 3 supports propagating Private VLAN information. As with extended-range VLANs, the lack of PVLAN support in VTP version 2 required all switches to be transparent mode and manual configuration at each switch.
6. VTP version 3 added support for opaque databases. In other words, VTP version 3 can transport more than just the VLAN database between switches. The only option at this time is to share the Multiple Spanning Tree (MSTP) database, but room was left for expansion. We will cover MSTP in a later lab.

VTP version 3 is backwards compatible with VTP version 2; at the boundary of the two protocols, a VTP version 3 switch will send out both version 3 and version 2-compatible messages. Version 2 messages received by a version 3 switch are discarded.

Configure VTP on DLS1, DLS2, and ALS2

VTP version 3 cannot be configured unless a VTP domain name has been set, so for this step, setting the domain name is not needed on DLS1. Configure VTP version 3 on DLS1, DLS2, and ALS2 using the following parameters

VTP domain SWLAB (DLS2 and ALS2 only)

VTP version 3

VTP mode server (DLS2 and ALS2 only)

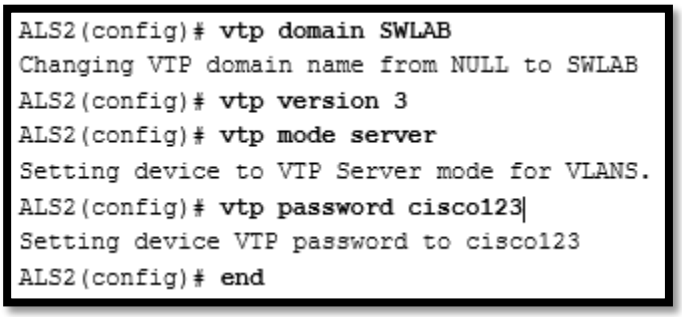
VTP password cisco123 (DLS2 and ALS2 only)

DLS1 Configuration:

```
DLS1(config)# vtp version 3
DLS1(config)#
*Mar  1 00:08:17.637: %SW_VLAN-6-OLD_CONFIG_FILE_READ: Old version 2 VLAN
configuration file detected and read OK.  Version 3
    files will be written in the future.
DLS1(config)# end
DLS1#
```

Example configuration on ALS2:

```
ALS2(config)# vtp domain SWLAB
Changing VTP domain name from NULL to SWLAB
ALS2(config)# vtp version 3
ALS2(config)# vtp mode server
Setting device to VTP Server mode for VLANS.
ALS2(config)# vtp password cisco123
Setting device VTP password to cisco123
ALS2(config)# end
*Mar  1 18:46:38.236: %SW_VLAN-6-VTP_DOMAIN_NAME_CHG: VTP domain name
changed to SWLAB.
*Mar  1 18:46:38.345: %SW_VLAN-6-OLD_CONFIG_FILE_READ: Old version 2 VLAN
configuration file detected and read OK.  Version 3
    files will be written in the future.
ALS2#
```



Configure trunking on DLS2 and ALS2

In steps 4 and 5 of part 1, we configured and activated all the trunk interfaces on DLS1 and ALS1. Now configure and activate all the trunk interfaces on DLS2 and ALS2.

Example from DLS2:

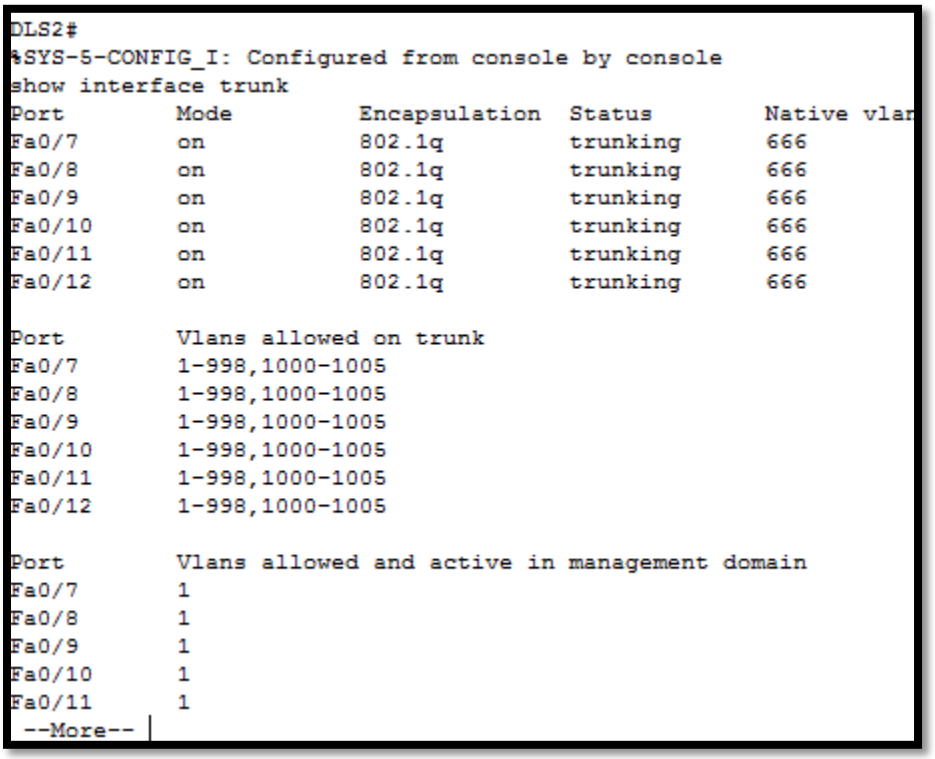
```
DLS2(config)# interface range f0/7-12
DLS2(config-if-range)# switchport trunk native vlan 666
DLS2(config-if-range)# switchport trunk encapsulation dot1q
DLS2(config-if-range)# switchport mode trunk
DLS2(config-if-range)# switchport nonegotiate
DLS2(config-if-range)# switchport trunk allowed vlan except 1,999
DLS2(config-if-range)# no shutdown
DLS2(config-if-range)# exit
DLS2(config)#
```

```
DLS2(config)# interface range f0/7-12
DLS2(config-if-range)# switchport trunk native vlan 666
DLS2(config-if-range)# switchport trunk encapsulation dot1q
DLS2(config-if-range)# switchport mode trunk
DLS2(config-if-range)# switchport nonegotiate
DLS2(config-if-range)# switchport trunk allowed vlan except 1,999
DLS2(config-if-range)# no shutdown
DLS2(config-if-range)# exit
```

Verify trunking and VTP on DLS2 and ALS2

Next verify that DLS2 and ALS2 trunking is operational. Here is an example from DLS2

```
DLS2# show interface trunk

Port        Mode             Encapsulation  Status        Native vlan
Fa0/7       on               802.1q         trunking      666
Fa0/8       on               802.1q         trunking      666
Fa0/9       on               802.1q         trunking      666
Fa0/10      on               802.1q         trunking      666
Fa0/11      on               802.1q         trunking      666
Fa0/12      on               802.1q         trunking      666

Port        Vlans allowed on trunk
Fa0/7       2-998,1000-4094
Fa0/8       2-998,1000-4094
Fa0/9       2-998,1000-4094
Fa0/10      2-998,1000-4094
Fa0/11      2-998,1000-4094
Fa0/12      2-998,1000-4094

Port        Vlans allowed and active in management domain
Fa0/7       none
Fa0/8       none
Fa0/9       none
Fa0/10      none
Fa0/11      none
Fa0/12      none

Port        Vlans in spanning tree forwarding state and not pruned
Fa0/7       none
Fa0/8       none
Fa0/9       none
Fa0/10      none
Fa0/11      none
Fa0/12      none
```



```
DLS2#
%SYS-5-CONFIG_I: Configured from console by console
show interface trunk
Port        Mode         Encapsulation  Status        Native vlan
Fa0/7       on           802.1q         trunking      666
Fa0/8       on           802.1q         trunking      666
Fa0/9       on           802.1q         trunking      666
Fa0/10      on           802.1q         trunking      666
Fa0/11      on           802.1q         trunking      666
Fa0/12      on           802.1q         trunking      666

Port        Vlans allowed on trunk
Fa0/7       1-998,1000-1005
Fa0/8       1-998,1000-1005
Fa0/9       1-998,1000-1005
Fa0/10      1-998,1000-1005
Fa0/11      1-998,1000-1005
Fa0/12      1-998,1000-1005

Port        Vlans allowed and active in management domain
Fa0/7       1
Fa0/8       1
Fa0/9       1
Fa0/10      1
Fa0/11      1
 --More--
```

Next, validate VTP is operational. Here is an example from DLS2:

```
DLS2# show vtp status
VTP Version capable             : 1 to 3
VTP version running             : 3
VTP Domain Name                 : SWLAB
VTP Pruning Mode                : Disabled
VTP Traps Generation            : Disabled
Device ID                       : e840.406f.7380

Feature VLAN:
--------------
VTP Operating Mode                : Server
Number of existing VLANs          : 5
Number of existing extended VLANs : 0
Maximum VLANs supported locally   : 1005
Configuration Revision            : 0
Primary ID                        : 0000.0000.0000
Primary Description               :
MD5 digest                        :


Feature MST:
--------------
VTP Operating Mode                : Transparent
```

```
Feature UNKNOWN:
--------------
VTP Operating Mode                : Transparent

DLS2#
```

Notice that the configuration revision number is zero and the number of local VLANs is the default of 5. There has been no update because DLS1's configuration revision number was reset to zero when the VTP version was changed, so at this point DLS2 and ALS2 will not learn about the configured VLANs because as far as they are concerned, they have the same database as DLS1.

```
DLS1# show vtp status | inc Configuration Revision
Configuration Revision            : 0
```

If we attempt to add VLANs at DLS1, or any of the other VTP version 3 switches, our attempt will not work and we will be told that we cannot add VLANs.

```
DLS1(config)# vlan 111
VTP VLAN configuration not allowed when device is not the primary server
for vlan database.
DLS1(config)#
```

Configure the Primary VTP Server

In a VTP version 3 domain, only the "Primary Server" can make changes to the VLAN database. Becoming the primary server requires the **`vtp primary`** privileged EXEC command be executed. When you issue that command, the switch checks to see if there is another switch acting as primary server already, and asks you to confirm that you want to continue.

In the output from DLS2 above, note that the Primary ID field equals 0000.0000.0000. That field will display the base MAC address of the primary server once a device is promoted into that role.

Also note that a separate primary server can be configured independently for each feature supported; VLAN or MST. If no feature is specified, the **`vlan`** feature is assumed.

Lastly, there is a **`force`** option which causes the switch not to check for conflicts in the identity of the primary server. If different switches in the VTP domain identify different switches as the primary server, there is a good chance there are inconsistencies in the VLAN database.

```
DLS1# vtp primary ?
  force  Do not check for conflicting devices
  mst    MST feature
  vlan   Vlan feature
  <cr>
```

```
DLS1# vtp primary ?
  force  Do not check for conflicting devices
  mst    MST feature
  vlan   Vlan feature
```

```
DLS1# vtp primary vlan
This system is becoming primary server for feature vlan
No conflicting VTP3 devices found.
Do you want to continue? [confirm]
DLS1#
*Mar  1 00:42:54.983: %SW_VLAN-4-VTP_PRIMARY_SERVER_CHG: e840.406f.7280 has become the primary server for the VLAN VTP feature
```

```
DLS1# vtp primary vlan
This system is becoming primary server for feature vlan
No conflicting VTP3 devices found.
Do you want to continue? [confirm]
DLS1#
```

Now verify the primary on DLS2 or ALS2:

```
DLS2# show vtp status | i Primary
Primary ID                      : e840.406f.7280
Primary Description             : DLS1
DLS2#
```

The promotion of DLS1 to primary increments its configuration revision number to 1, so the VLANs that were previously created on DLS1 are propagated to DLS2 and ALS2 automatically.

```
ALS2# sho vlan brief | inc active
1    default                          active    Fa0/1, Fa0/2, Fa0/3, Fa0/4
99   MANAGEMENT                       active
100  SERVERS                          active
110  GUEST                            active
120  OFFICE                           active
666  NATIVE_DO_NOT_USE                active
```

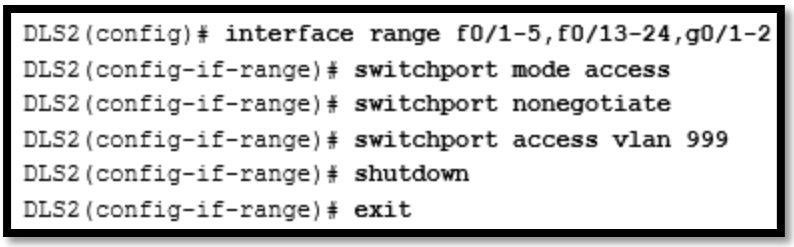
VLAN 999 will be missing from the filtered output above because it only includes VLANs in active state and VLAN 999 is suspended. Using the **`show vlan brief`** without filtering would show the VLAN 999.

Park unused interfaces.

On DLS2 and ALS2, place all of interfaces that will not be used into the PARKING_LOT VLAN and shut them down. For this lab, the interfaces being used on all switches are F0/6 through F0/12.

An example from DLS2:

```
DLS2(config)# interface range f0/1-5,f0/13-24,g0/1-2
DLS2(config-if-range)# switchport mode access
DLS2(config-if-range)# switchport nonegotiate
DLS2(config-if-range)# switchport access vlan 999
DLS2(config-if-range)# shutdown
DLS2(config-if-range)# exit
```



Verify VLAN management capability

DLS1 is able to create VLANs, including extended-range VLANs. Note that because ALS1 is running VTP version 2 and its revision number is 6, it will ignore any of the VTPv2 messages sent to it because they have a lower revision number. When a VTP message with an equal revision number but different MD5 checksum is received, ALS1 will report an error. Here we added seven VLANs and then remove 6 of them to push the revision number on DLS1 to 9.

```
DLS1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
DLS1(config)# vlan 510
DLS1(config-vlan)# name TEST510
DLS1(config-vlan)# exit
DLS1(config)# vlan 511
DLS1(config-vlan)# name TEST511
DLS1(config-vlan)# exit
DLS1(config)# vlan 512
DLS1(config-vlan)# name TEST512
DLS1(config-vlan)# exit
DLS1(config)# vlan 513
DLS1(config-vlan)# name TEST513
DLS1(config-vlan)# exit
DLS1(config)# vlan 514
DLS1(config-vlan)# name TEST514
DLS1(config-vlan)# exit
DLS1(config)# vlan 515
DLS1(config-vlan)# name TEST515
DLS1(config-vlan)# exit
DLS1(config)# vlan 1500
DLS1(config-vlan)# name TEST-EXT-1500
DLS1(config-vlan)# exit
DLS1(config)# no vlan 510-514
DLS1(config)# end
DLS1#
```

```
DLS1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
DLS1(config)# vlan 510
DLS1(config-vlan)# name TEST510
DLS1(config-vlan)# exit
DLS1(config)# vlan 511
DLS1(config-vlan)# name TEST511
DLS1(config-vlan)# exit
DLS1(config)# vlan 512
DLS1(config-vlan)# name TEST512
DLS1(config-vlan)# exit
DLS1(config)# vlan 513
DLS1(config-vlan)# name TEST513
DLS1(config-vlan)# exit
DLS1(config)# vlan 514
DLS1(config-vlan)# name TEST514
DLS1(config-vlan)# exit
DLS1(config)# vlan 515
DLS1(config-vlan)# name TEST515
DLS1(config-vlan)# exit
DLS1(config)# vlan 1500
DLS1(config-vlan)# name TEST-EXT-1500
DLS1(config-vlan)# exit
DLS1(config)# no vlan 510-514
DLS1(config)# end
DLS1#
```

```
ALS1# show vlan brief | i active
1    default                          active    Fa0/6
99   MANAGEMENT                       active
100  SERVERS                          active
110  GUEST                            active
120  OFFICE                           active
515  TEST515                          active
666  NATIVE_DO_NOT_USE                active
ALS1#

ALS2#show vlan brief | inc active
1    default                          active    Fa0/6
99   MANAGEMENT                       active
100  SERVERS                          active
110  GUEST                            active
120  OFFICE                           active
515  TEST515                          active
666  NATIVE_DO_NOT_USE                active
1002 fddi-default                     act/unsup
1003 trcrf-default                    act/unsup
```

```
1004 fddinet-default                act/unsup
1005 trbrf-default                  act/unsup
1500 TEST-EXT-1500                  active
ALS2#
```

At this point, you should be able to ping each switch from every other switch.

Change the VLAN status to deactivate ports.

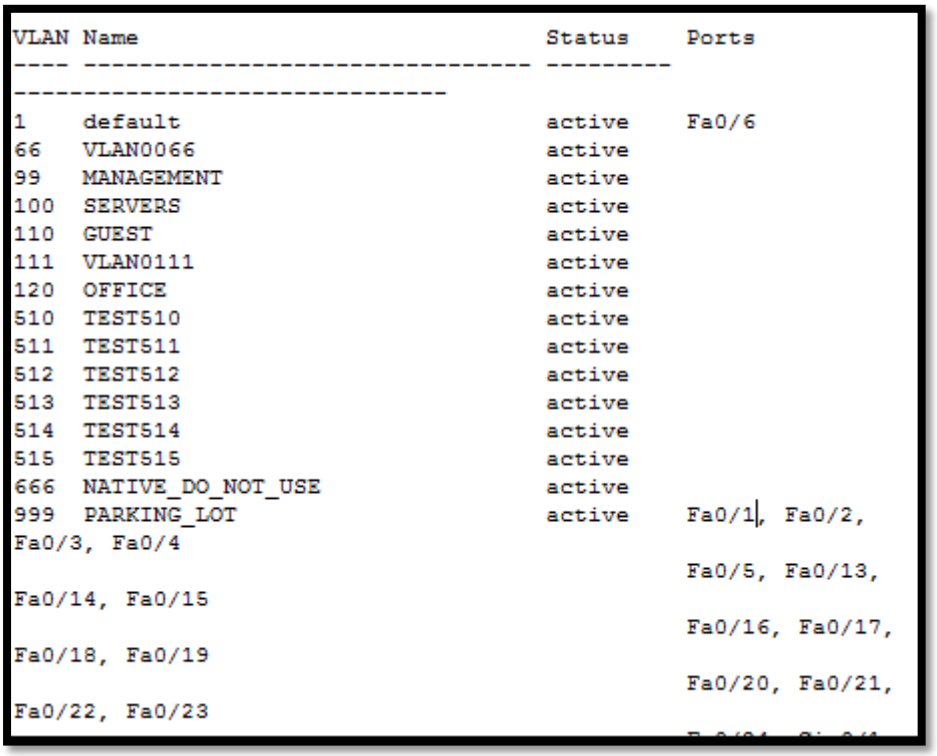
As already briefly discussed in Part 2, Step 3 of this lab, the default status of VLAN 1 and user-created VLANs is "active". A VLAN can be made locally inactive by entering the global configuration command **`shutdown vlan`** *`vlan-id`*, where *`vlan-id`* is the number of the VLAN to be shut down.

Alternatively, the VLAN can be shut down by issuing the **`shutdown`** command while in VLAN configuration mode and then exiting.

Both these options are equivalent, however, only the **`shutdown vlan`** command works in while in VTP client mode.

Shut down the Guest VLAN 110 on ALS1, wait a few moments, exit the configuration mode and then issue the **`show vlan brief`** command. The status should change to “act/lshut”.

```
ALS1(config)# shutdown vlan 110

ALS1# show vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
---
1    default                          active    Fa0/6
99   MANAGEMENT                       active
100  SERVERS                          active
110  GUEST                            act/lshut
515  TEST515                          active
120  OFFICE                           active
666  NATIVE_DO_NOT_USE                active
999  PARKING_LOT                      suspended Fa0/1, Fa0/2, Fa0/3, Fa0/4
                                                Fa0/5, Fa0/13, Fa0/14,
Fa0/15
                                                Fa0/16, Fa0/17, Fa0/18,
Fa0/19
                                                Fa0/20, Fa0/21, Fa0/22,
Fa0/23
                                                Fa0/24, Gi0/1, Gi0/2
1002 fddi-default                     act/unsup
1003 trcrf-default                    act/unsup
1004 fddinet-default                  act/unsup
1005 trbrf-default                    act/unsup
ALS1#
```



```
VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/6
66   VLAN0066                         active
99   MANAGEMENT                       active
100  SERVERS                          active
110  GUEST                            active
111  VLAN0111                         active
120  OFFICE                           active
510  TEST510                          active
511  TEST511                          active
512  TEST512                          active
513  TEST513                          active
514  TEST514                          active
515  TEST515                          active
666  NATIVE_DO_NOT_USE                active
999  PARKING_LOT                      active    Fa0/1, Fa0/2,
Fa0/3, Fa0/4
                                                Fa0/5, Fa0/13,
Fa0/14, Fa0/15
                                                Fa0/16, Fa0/17,
Fa0/18, Fa0/19
                                                Fa0/20, Fa0/21,
Fa0/22, Fa0/23
```

Reactivate all ports in ALS1 Guest VLAN 110 using the **no shutdown** command in VLAN configuration mode.

```
ALS1(config)# no shutdown vlan 110
```

As discussed and demonstrated in Part 2, Step 3 of this lab, you can put a VLAN into suspended status by using the **state suspend** command while in VLAN configuration mode on a VTPv2 server switch or on the VTPv3 primary server switch. In a mixed VTP version network, the suspension only works network-wide if it originates from the VTPv3 primary server. Suspending a VLAN causes all ports in that VLAN throughout the VTP domain to stop transferring data.

Suspend Guest VLAN 110 on DLS1, wait a few moments, exit VLAN configuration mode and then issue the **show vlan brief | include suspended** command. The status should change show the VLAN as suspended.

```
DLS1(config)# vlan 110
DLS1(config-vlan)# state ?
  active   VLAN Active State
  suspend  VLAN Suspended State

DLS1(config-vlan)# state suspend
DLS1(config-vlan)# exit
DLS1(config)#end
DLS1#

DLS1# sho vlan brief | include suspended
110  GUEST                            suspended
999  PARKING_LOT                      suspended Fa0/1, Fa0/2, Fa0/3, Fa0/4
DLS1#
```

Issue the **show vlan brief | include suspended** command on another switch in the network, and you will see that the VLAN status is suspended as well.

```
DLS2# show vlan brief | include suspended
110  GUEST                            suspended
999  PARKING_LOT                      suspended Fa0/1, Fa0/2, Fa0/3, Fa0/4
DLS2#
```

Reactivate VLAN 110 using the **state active** command in VLAN configuration mode.

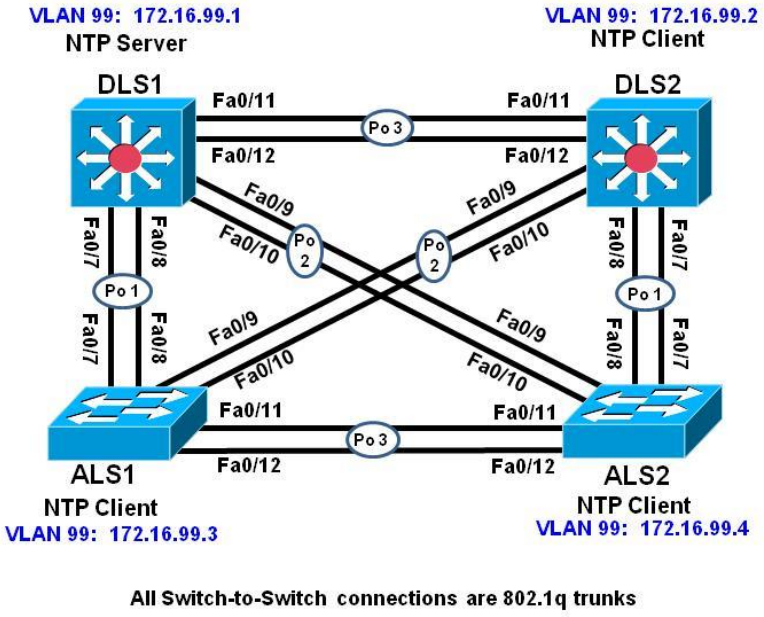
```
DLS1(config)# vlan 110
DLS1(config-vlan)# state active
DLS1(config-vlan)# exit
DLS1(config)#
```

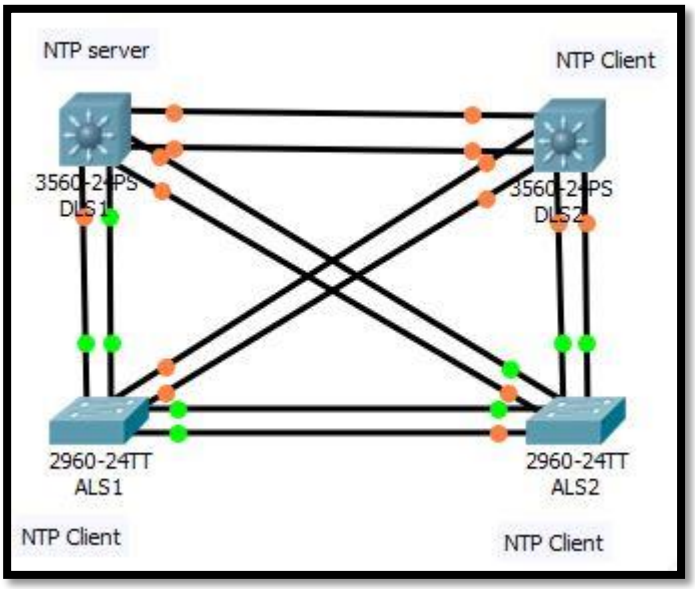
Issue the **show vlan brief | include suspended** command on another switch in the network, and you will see that the VLAN status is no longer listed.

```
DLS2# show vlan brief | include suspended
999  PARKING_LOT                      suspended Fa0/1, Fa0/2, Fa0/3, Fa0/4
DLS2#
```

Synchronizing Campus Network Devices using Network Time Protocol (NTP)

Topology





Prepare for the Lab

Prepare the switches for the lab

Use the **`reset.tcl`** script you created in Lab 1 “Preparing the Switch” to set your switches up for this lab. Then load the file BASE.CFG into the running-config with the command **`copy flash:BASE.CFG running-config`**. An example from DLS1:

```
DLS1# tclsh reset.tcl
Erasing the nvram filesystem will remove all configuration files!
Continue? [confirm]
[OK]
Erase of nvram: complete
Reloading the switch in 1 minute, type reload cancel to halt

Proceed with reload? [confirm]

*Mar  7 18:41:40.403: %SYS-7-NV_BLOCK_INIT: Initialized the
geometry of nvram
*Mar  7 18:41:41.141: %SYS-5-RELOAD: Reload requested by console.
Reload Reason: Reload command.
<switch reloads - output omitted>

Would you like to enter the initial configuration dialog? [yes/no]:
n
Switch> en
*Mar  1 00:01:30.915: %LINK-5-CHANGED: Interface Vlan1, changed
state to administratively down
Switch# copy BASE.CFG running-config
Destination filename [running-config]?
```

```
184 bytes copied in 0.310 secs (594 bytes/sec)
DLS1#
```

Configure basic switch parameters.

Configure an IP address on the management VLAN according to the diagram. VLAN 1 is the default management VLAN, but following best practice, we will use a different VLAN. In this case, VLAN 99.

a. Enter basic configuration commands on each switch according to the diagram.

DLS1 example:

```
DLS1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
DLS1(config)# vlan 99

DLS1(config-vlan)# name Management
DLS1(config-vlan)# exit
DLS1(config)# interface vlan 99
DLS1(config-if)# ip address 172.16.99.1 255.255.255.0
DLS1(config-if)# no shutdown
```

(Optional) On each switch, create an enable secret password and configure the VTY lines to allow remote access from other network devices.

DLS1 example:

```
DLS1(config)# enable secret class
DLS1(config)# line vty 0 15
DLS1(config-line)# password cisco
DLS1(config-line)# login
```

Note: The passwords configured here are required for NETLAB compatibility only and are NOT recommended for use in a live environment.

```
DLS1(config)#vlan 99
DLS1(config-vlan)#name Management
DLS1(config-vlan)#exit
DLS1(config)#interface vlan 99
DLS1(config-if)#
%LINK-5-CHANGED: Interface Vlan99, changed state to up

DLS1(config-if)#ip address 172.16.99.1 255.255.255.0
DLS1(config-if)#no shut
DLS1(config-if)#exit
DLS1(config)#enable secret class
DLS1(config)#line vty 0 15
DLS1(config-line)#password cisco
DLS1(config-line)#login
```

b. Configure DLS2, ALS1, and ALS2 to use DLS1 as their default gateway.

```
DLS2(config)# ip default-gateway 172.16.99.2
```

```
DLS2(config)#vlan 99
DLS2(config-vlan)#name Management
DLS2(config-vlan)#exit
DLS2(config)#int vlan 99
DLS2(config-if)#
%LINK-5-CHANGED: Interface Vlan99, changed state to up

DLS2(config-if)#ip address 172.16.99.2 255.255.255.0
DLS2(config-if)#no shut
DLS2(config-if)#exit
DLS2(config)#enable secret class
DLS2(config)#line vty 0 15
DLS2(config-line)#password cisco
DLS2(config-line)#login
DLS2(config-line)#exit
DLS2(config)#ip default-gateway 172.16.99.2
```

Configure trunks and EtherChannels between switches.

EtherChannel is used for the trunks because it allows you to utilize both Fast Ethernet interfaces that are available between each device, thereby doubling the bandwidth.

Note: It is good practice to shut down the interfaces on both sides of the link before a port channel is created and then re-enable them after the port channel is configured. In the BASE configuration, all interfaces are shut down, so you must remember to issue the **`no shutdown`** command.

e. Configure trunks and EtherChannels from DLS1 and DLS2 to the other three switches according to the diagram. The **switchport trunk encapsulation** {**isl** | **dot1q**} command is used because these switches also support ISL encapsulation. A sample configuration has been provided to assist you with the trunking and etherchannel configurations.

```
DLS1(config)# interface range fastEthernet 0/7 - 8
DLS1(config-if-range)# switchport trunk encapsulation dot1q
DLS1(config-if-range)# switchport mode trunk
DLS1(config-if-range)# switchport nonegotiate
DLS1(config-if-range)# channel-group 1 mode desirable
DLS1(config-if-range)# no shut
Creating a port-channel interface Port-channel 1
```

```
DLS1(config)#interface range fastEthernet 0/7 - 8
DLS1(config-if-range)#switchport trunk encapsulation dot1q
DLS1(config-if-range)#switchport mode trunk


DLS1(config-if-range)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/7, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/7, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan99, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/8, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/8, changed state to up

DLS1(config-if-range)#switchport nonegotiate
DLS1(config-if-range)#channel-group 1 mode desirable
DLS1(config-if-range)#
Creating a port-channel interface Port-channel 1

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/7, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/7, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/8, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/8, changed state to up

DLS1(config-if-range)#no shut
```

```
DLS2(config)#interface range fastEthernet 0/7 - 8
DLS2(config-if-range)#switchport trunk encapsulation dot1q
DLS2(config-if-range)#switchport mode trunk


DLS2(config-if-range)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/7, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/7, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan99, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/8, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/8, changed state to up

DLS2(config-if-range)#switchport nonegotiate
DLS2(config-if-range)#channel-group 1 mode desirable
DLS2(config-if-range)#
Creating a port-channel interface Port-channel 1

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/7, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/7, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/8, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/8, changed state to up

DLS2(config-if-range)#no shut
```

f. Configure the trunks and EtherChannel from ALS1 and ALS2 to the other switches. Notice that no encapsulation type is needed because the 2960 supports only 802.1q trunks. A sample configuration has been provided to assist you with the trunking and etherchannel configurations.

```
ALS1(config)# interface range fastEthernet 0/7 - 8
ALS1(config-if-range)# switchport mode trunk
ALS1(config-if-range)# switchport nonegotiate
ALS1(config-if-range)# channel-group 1 mode desirable
ALS1(config-if-range)# no shut
```

```
ALS1(config)#interface range fastEthernet 0/7 - 8
ALS1(config-if-range)#switchport mode trunk


ALS1(config-if-range)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/7, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/7, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan99, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/8, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/8, changed state to up

ALS1(config-if-range)#switchport nonegotiate
ALS1(config-if-range)#channel-group 1 mode desirable
ALS1(config-if-range)#
Creating a port-channel interface Port-channel 1

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/7, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/7, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/8, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/8, changed state to up

%LINK-5-CHANGED: Interface Port-channel1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Port-channel1, changed state to up

ALS1(config-if-range)#no shut
```

Verify trunking between DLS1, ALS1, and ALS2 using the **show interface trunk** command on all switches.

```
DLS1# show interface trunk

      Port        Mode             Encapsulation  Status        Native
      vlan
      Po1         on               802.1q         trunking      1
      Po2         on               802.1q         trunking      1
      Po3         on               802.1q         trunking      1

      Port        Vlans allowed on trunk
      Po1         1-4094
      Po2         1-4094
```

```
Po3         1-4094

Port        Vlans allowed and active in management domain
Po1         1
Po2         1
Po3         1

Port        Vlans in spanning tree forwarding state and not pruned
Po1         1
Po2         1
Po3         1
```

```
DLS1#show interface trunk
Port        Mode         Encapsulation  Status        Native vlan
Po1         on           802.1q         trunking      1
Po2         on           802.1q         trunking      1
Po3         on           802.1q         trunking      1

Port        Vlans allowed on trunk
Po1         1-1005
Po2         1-1005
Po3         1-1005

Port        Vlans allowed and active in management domain
Po1         1,99
Po2         1,99
Po3         1,99

Port        Vlans in spanning tree forwarding state and not
pruned
Po1         1,99
Po2         99
Po3         none
```

Issue the **`show etherchannel summary`** command on each switch to verify the EtherChannels. In the following sample output from ALS1, notice the three EtherChannels on the access and distribution layer switches.

```
ALS1# show etherchannel summary
      Flags:  D - down        P - bundled in port-channel
              I - stand-alone s - suspended
              H - Hot-standby (LACP only)
              R - Layer3      S - Layer2
              U - in use      f - failed to allocate aggregator

              M - not in use, minimum links not met
              u - unsuitable for bundling
              w - waiting to be aggregated
              d - default port


      Number of channel-groups in use: 3
      Number of aggregators:           3

      Group  Port-channel  Protocol    Ports
      ------+-------------+-----------+-----------------------------------
      -------------
```

```
1       Po1(SU)          PAgP       Fa0/7(P)     Fa0/8(P)
2       Po2(SU)          PAgP       Fa0/9(P)     Fa0/10(P)
3       Po3(SU)          PAgP       Fa0/11(P)    Fa0/12(P)
```

```
ALS1#show etherchannel summary
Flags:  D - down        P - in port-channel
        I - stand-alone s - suspended
        H - Hot-standby (LACP only)
        R - Layer3      S - Layer2
        U - in use      f - failed to allocate aggregator
        u - unsuitable for bundling
        w - waiting to be aggregated
        d - default port


Number of channel-groups in use: 3
Number of aggregators:           3

Group  Port-channel  Protocol    Ports
------+-------------+-----------+-----------------------------------------------

1      Po1(SU)           PAgP   Fa0/7(P) Fa0/8(P)
2      Po2(SU)           PAgP   Fa0/9(P) Fa0/10(P)
3      Po3(SU)           PAgP   Fa0/11(P) Fa0/12(P)
ALS1#
```

Which EtherChannel negotiation protocol is in use here?

PAgP

Configure the system clock .

g. The system clock can be set using a variety of methods. The system clock can be manually set, the time can be derived from an NTP source or from a subset of NTP (SNTP). It is important that all of your devices have accurate timestamps for use in systems reporting and for tracking validity of X.509 certificates used in Public Key Infrastructure and for event correlation in attack identification.

```
DLS1# show clock
*00:24:31.647 UTC Mon Mar 1 1993
```

h. DLS1#The show clock command displays what time is currently set on the device.

On DLS1, manually reconfigure the system clock using the **clock set** command from *privileged exec mode of operation.*

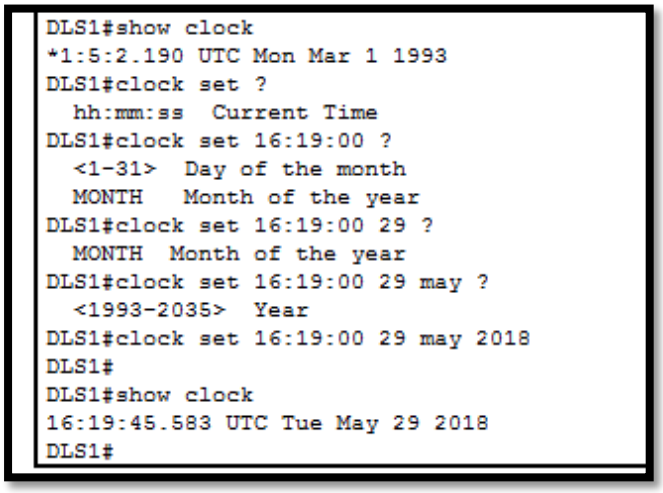
Note that the time you set should be the Coordinated Universal Time value.

```
DLS1# clock set ?
  hh:mm:ss  Current Time

DLS1# clock set 14:45:00 ?
  <1-31>  Day of the month
  MONTH   Month of the year

DLS1# clock set 14:45:00 29 ?
  MONTH  Month of the year

DLS1# clock set 14:45:00 29 July ?
```

```
  <1993-2035>  Year

DLS1# clock set 14:45:00 29 July 2015
DLS1#
*Jul 29 14:45:00.000: %SYS-6-CLOCKUPDATE: System clock has been
updated from 00:03:21 UTC Mon Mar 1 1993 to 14:45:00 UTC Wed Jul 29
2015, configured from console by console.
```

i. Verify that the system clock has been updated.

```
DLS1# show clock
14:45:33.755 UTC Wed Jul 29 2015
```



j. The default timezone is UTC. Change the default timezone to your current timezone and your standard time offset. For this example the system will be set for US Central Standard Time (CST) with a 6 hour negative offset. The -6 is the difference in hours from UTC for US Central Standard Time. Use clock timezone zone hours-offset command in ***global configuration***.

```
DLS1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
DLS1(config)# clock timezone ?
  WORD  name of time zone

DLS1(config)# clock timezone CST ?
  <-23 - 23>  Hours offset from UTC

DLS1(config)# clock timezone CST -6 ?
  <0-59>  Minutes offset from UTC
  <cr>

DLS1(config)# clock timezone CST -6
DLS1(config)#
Jul 29 14:46:26.620: %SYS-6-CLOCKUPDATE: System clock has been
updated from 14:46:26 UTC Wed Jul 29 2015 to 08:46:26 CST Wed Jul
29 2015, configured from console by console
```

```
DLS1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
DLS1(config)#clock timezone ?
  WORD  name of time zone
DLS1(config)#clock timezone CST ?
  <-23 - 23>  Hours offset from UTC
DLS1(config)#clock timezone CST -6 ?
  <0-59>  Minutes offset from UTC
  <cr>
DLS1(config)#clock timezone CST -6
DLS1(config)#
```

k. The command **clock summer-time** command can be used to automatically switch between standard time and daylight saving time. If you do not use this command, the system will default to using daylight savings rules for the United States.

```
DLS1(config)# clock summer-time ?
  WORD  name of time zone in summer

DLS1(config)# clock summer-time CDT ?
  date       Configure absolute summer time
  recurring  Configure recurring summer time

DLS1(config)# clock summer-time CDT recurring ?
  <1-4>  Week number to start
  first  First week of the month
  last   Last week of the month
  <cr>

DLS1(config)# clock summer-time CDT recurring
DLS1(config)#
Jul 29 14:47:20.249: %SYS-6-CLOCKUPDATE: System clock has been
updated from 08:47:20 CST Wed Jul 29 2015 to 09:47:20 CDT Wed Jul
29 2015, configured from console by console]
```

Verify clock settings using the **show clock** command with the keyword **detail**.

```
DLS1# show clock detail
09:48:11.679 CDT Wed Jul 29 2015
Time source is user configuration
Summer time starts 02:00:00 CST Sun Mar 8 2015
Summer time ends 02:00:00 CDT Sun Nov 1 2015
```

```
DLS1#
DLS1#show clock detail
10:25:26.747 CST Tue May 29 2018
Time source is user configuration
DLS1#
```

The clock setting should reflect the current local time, adjusted for daylight savings as appropriate.

Setting the clocks manually is not considered an accurate method of tracking time and events in networks. It is also not a scalable solution to manually configure time on all network devices. The

Network Time Protocol (NTP) allows the network device to poll an authoritative time source for synchronization.

Configure NTP.

NTP is used to make sure all network devices in the campus are synchronized. Time accuracy can be derived from three different external sources: Atomic clock, GPS receiver, and accurate time source. NTP synchronizes using UDP port 123. All of the devices must be configured with NTP.

a. Configure DLS1 as the authoritative time source in the campus network by using the **ntp master** command.

```
DLS1(config)# ntp master ?

 <1-15>  Stratum number

 <cr>
```

This command should only be used if you do not have a reliable external reference clock. We will use **ntp master** *stratum_number* command in global configuration mode. The stratum number should be configured with a high number in the event that there is more reliable NTP source available. A machine running NTP automatically chooses the machine with the lowest stratum number that is configured to communicate with using NTP as its time source. The lower the stratum number the more trustworthy the accuracy of the time source.

```
DLS1(config)# ntp master 10
```

b. Configure DLS2, ALS1, and ALS2 to synchronize to DLS1 using the **ntp server** *A.B.C.D (* IP address of peer) command. NTP synchronization should always refer to the most stable interface. Loopback interfaces are considered always up interfaces and therefore, the best choice for NTP synchronization. Also, the local time zone should be configured on each local device. Also, configure these devices with the same time zone and summer time configuration as on DLS1.

```
DLS2(config)# ntp server ?
  A.B.C.D     IP address of peer
  WORD        Hostname of peer
  X:X:X:X::X  IPv6 address of peer
  ip          Use IP for DNS resolution
  ipv6        Use IPv6 for DNS resolution
  vrf         VPN Routing/Forwarding Information
DLS2(config)# ntp server 172.16.99.1
DLS2(config)# clock timezone CST -6
*Jul 29 14:50:38.980: %SYS-6-CLOCKUPDATE: System clock has been
updated from 14:50:38 UTC Wed Jul 29 2015 to 08:50:38 CST Wed Jul
29 2015, configured from console by console.
DLS2(config)# clock summer-time CDT recurring
DLS2(config)#
Jul 29 14:50:54.247: %SYS-6-CLOCKUPDATE: System clock has been
updated from 08:50:54 CST Wed Jul 29 2015 to 09:50:54 CDT Wed Jul
29 2015, configured from console by console.
```

```
DLS2(config)#ntp ?
  authenticate        Authenticate time sources
  authentication-key  Authentication key for trusted time sources
  server              Configure NTP server
  trusted-key         Key numbers for trusted time sources
  update-calendar     Configure NTP to update the calendar.
DLS2(config)#ntp server ?
  A.B.C.D  IP address of peer
DLS2(config)#ntp server 172.16.99.1
DLS2(config)#clock timezone CST -6
DLS2(config)#clock summer-time CDT recurring
```

NOTE: Ensure that these commands are repeated on ALS1 and ALS2.

Verify NTP .

On DLS2, ALS1, and ALS2, use the **show ntp status** command to verify if these device clocks have synchronized to DLS1.

```
DLS2# show ntp status
Clock is synchronized, stratum 11, reference is 172.16.99.1
nominal freq is 119.2092 Hz, actual freq is 119.2092 Hz, precision is 2**17
reference time is D96366D1.B4DD4BA4 (09:50:57.706 CDT Wed Jul 29 2015)
clock offset is -0.2067 msec, root delay is 2.03 msec
root dispersion is 195.31 msec, peer dispersion is 1.55 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is -0.000000002
s/s
system poll interval is 64, last update was 165 sec ago
```

Outputs for ALS1 and ALS2 should be similar to the output displayed above for DLS2.

The stratum will be +1 from the stratum value used on the master in the network. In this lab scenario, we use ntp master 10. This indicates a stratum of 10 + 1 = 11. The stratum 11 is indicated in the show output.

NOTE: NTP may take up to 5 minutes to synchronize.

Secure NTP Operation using Access-Lists and Authentication

Secure NTP Operation using Access-Lists and Authentication

NTP operation can be secured using MD5 authentication. Authentication is enabled with the **ntp authenticate** command. The authentication keys are defined with **ntp authentication-key** command. The number specifies a unique NTP key. Valid keys are identified using the **ntp-trusted-key** command. It is important to note that NTP does not authenticate clients. NTP authenticates the source. Devices can still respond to unauthenticated requests. For this reason, access lists should be used in conjunction with NTP authentication to restrict NTP access.

a. Configure NTP authentication on DLS1 and DLS2.

```
DLS1(config)# ntp authenticate
DLS1(config)# ntp authentication-key 1 md5 P@55word
```

```
DLS1(config)# ntp trusted-key 1
```

Use the **show ntp status** command to verify that clock is still synchronized (note that it could take 5 minutes to resynchronize. The system clock is not reset, just the NTP relationship). If the clock shows unsynchronized, the client has not successfully authenticated.

```
DLS2# show ntp status

Clock is synchronized, stratum 11, reference is 172.16.99.1
nominal freq is 119.2092 Hz, actual freq is 119.2092 Hz, precision is 2**17
reference time is D9636A49.B36724F9 (10:05:45.700 CDT Wed Jul 29 2015)
clock offset is -0.3060 msec, root delay is 2.50 msec
root dispersion is 68.07 msec, peer dispersion is 0.10 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is -0.000000003
s/s
system poll interval is 128, last update was 12 sec ago.
```

```
DLS1(config)#ntp authenticate
DLS1(config)#ntp authentication-key 1 md5 P@55word
DLS1(config)#ntp trusted-key 1
DLS1(config)#
```

b. Repeat these commands on ALS1 and ALS2. Verify that the device clocks are synchronized using the appropriate show command.

Control NTP Access using Access-Lists

Time Servers can provide synchronization services in all directions to other devices in your network. A rogue NTP server to come in and falsify time on your network. Also, a rogue device could send a large number of bogus synchronization requests to your server preventing it from servicing legitimate devices. Configure an access-list so that polling can only come from members of the 172.16.0.0/16 network. NTP masters must allow "peer" access to source with IP address **127.127.x.1**. This IP address is the internal server address created by the NTP master command. The local router synchronizes using this IP. View the output for the **show ntp associations** command. If your device is configured as NTP master, then you must allow access to source IP of 127.127.x.1. This is because 127.127.x.1 is the internal server that is created by the **ntp master** command. The value of the third octet varies between platforms.

```
DLS1# show ntp associations

  address         ref clock       st   when   poll reach  delay  offset
disp
*~127.127.1.1     .LOCL.           9     12     16   377  0.000   0.000
0.226
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~
configured
DLS1#
```

Use the following commands to ensure that only devices on the 172.16.0.0 /16 network are able to poll or send requests to the NTP server. The **ntp access-group peer** command allows the

devices to synchronize itself to remote systems that pass the access-list. Time synchronization and control queries are allowed.

```
DLS1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
DLS1(config)# access-list 1 permit 127.127.1.1
DLS1(config)# access-list 2 permit 172.16.0.0 0.0.255.255
DLS1(config)#
DLS1(config)# ntp access-group ?
  peer        Provide full access
  query-only  Allow only control queries
  serve       Provide server and query access
  serve-only  Provide only server access

DLS1(config)# ntp access-group peer 1
DLS1(config)# ntp access-group serve-only 2
DLS1(config)# end
DLS1#
```

This command references the source address listed in **access-list 2** to determine if NTP services will rendered to the requesting device.

Verify NTP on all devices.

Use the **show ntp associations**, **show ntp status**, **show clock detail** commands to verify synchronization and configurations across the campus network.

```
DLS1# show ntp status
Clock is synchronized, stratum 10, reference is 127.127.1.1
nominal freq is 119.2092 Hz, actual freq is 119.2092 Hz, precision is 2**17
reference time is D963700D.38FAF326 (10:30:21.222 CDT Wed Jul 29 2015)
clock offset is 0.0000 msec, root delay is 0.00 msec
root dispersion is 0.48 msec, peer dispersion is 0.25 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000000000
s/s
system poll interval is 16, last update was 15 sec ago.

DLS1# show ntp associations

  address         ref clock       st   when   poll reach  delay  offset
disp
*~127.127.1.1     .LOCL.           9      0     16   377  0.000   0.000
0.244
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~
configured
DLS1#
DLS1# show clock detail
10:31:16.453 CDT Wed Jul 29 2015
Time source is NTP
Summer time starts 02:00:00 CST Sun Mar 8 2015
Summer time ends 02:00:00 CDT Sun Nov 1 2015
DLS1#
```

Compare the output on DLS2 to ALS1 and ALS2 devices.

```
DLS2# show ntp status
Clock is synchronized, stratum 11, reference is 172.16.99.1
nominal freq is 119.2092 Hz, actual freq is 119.2092 Hz, precision is 2**17
reference time is D9636FBE.B4569C10 (10:29:02.704 CDT Wed Jul 29 2015)
clock offset is 0.3609 msec, root delay is 1.99 msec
root dispersion is 7.82 msec, peer dispersion is 3.64 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is -0.000000003
s/s
system poll interval is 128, last update was 170 sec ago.
DLS2#
DLS2# show ntp associations

  address         ref clock       st   when   poll reach  delay  offset
disp
*~172.16.99.1     127.127.1.1     10    176    128   376  1.990   0.360
3.642
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~
configured
DLS2#
DLS2# show clock detail
10:32:02.545 CDT Wed Jul 29 2015
Time source is NTP
Summer time starts 02:00:00 CST Sun Mar 8 2015
Summer time ends 02:00:00 CDT Sun Nov 1 2015


ALS1# show ntp status
Clock is synchronized, stratum 11, reference is 172.16.99.1
nominal freq is 119.2092 Hz, actual freq is 119.2093 Hz, precision is 2**17
reference time is D9636FCD.C3C21CD9 (10:29:17.764 CDT Wed Jul 29 2015)
clock offset is -2.6199 msec, root delay is 2.16 msec
root dispersion is 13.73 msec, peer dispersion is 67.34 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is -0.000000145
s/s
system poll interval is 64, last update was 225 sec ago.
ALS1#
ALS1# show ntp associations

  address         ref clock       st   when   poll reach  delay  offset
disp
*~172.16.99.1     127.127.1.1     10    229     64   370  2.165  -2.619
67.347
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~
configured
ALS1#
ALS1# show clock detail
10:33:12.625 CDT Wed Jul 29 2015
Time source is NTP
Summer time starts 02:00:00 CST Sun Mar 8 2015
```

```
Summer time ends 02:00:00 CDT Sun Nov 1 2015
ALS1#

ALS2# show ntp status
Clock is synchronized, stratum 11, reference is 172.16.99.1
nominal freq is 119.2092 Hz, actual freq is 119.2093 Hz, precision is 2**17
reference time is D9636E71.F02399BE (10:23:29.938 CDT Wed Jul 29 2015)
clock offset is -5.3988 msec, root delay is 1.81 msec
root dispersion is 18.57 msec, peer dispersion is 195.04 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is -0.000000103
s/s
system poll interval is 64, last update was 624 sec ago.
ALS2#
ALS2# show ntp associations

  address         ref clock       st   when   poll reach  delay  offset
disp
*~172.16.99.1     127.127.1.1     10    306     64   360  1.811  -5.398
195.04
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~
configured
ALS2#
ALS2# show clock detail
10:34:04.084 CDT Wed Jul 29 2015
Time source is NTP
Summer time starts 02:00:00 CST Sun Mar 8 2015
Summer time ends 02:00:00 CDT Sun Nov 1 2015
ALS2#
```

Conclusiones.

Durante el trabajo se utilizó la configuración avanzada en routers con diferentes tipos de configuración tanto IPv4 y IPv6, con protocolos de enrutamiento como: SLA, NAT, BGP, INGP. Se implementa seguridad en los routers con clave cifrada.

Cuando se crean rutas estáticas se debe configurar con una dirección IP del siguiente salto que generalmente es la dirección IP del router del siguiente salto. También se aplica para rutas estáticas configuradas con una dirección del siguiente salto y una interfaz de salida.

OSPFv2 admite la autenticación de hash SHA usando cadenas de claves. Cisco se refiere a esto como la función de autenticación criptográfica OSPFv2. La función evita las actualizaciones de enrutamiento no autorizadas o no válidas en una red al autenticar los paquetes de protocolo OSPFv2 utilizando algoritmos HMAC-SHA.

La autenticación, autorización y contabilidad (AAA) es un marco basado en estándares que se puede implementar para controlar a quién se le permite acceder a una red (autenticar), qué pueden hacer en esa red (autorizar) y auditar lo que hicieron mientras accedían La red (contabilidad).

Dado que la sincronización no es el comportamiento predeterminado con IOS, es importante que los administradores de red aseguren la accesibilidad de IBGP entre todos los enrutadores en la ruta de tránsito.

Se observó por medio de práctica como el protocolo BGP (Protocolo de enlace de frontera), es un protocolo el cual se puede intercambiar información entre sistemas autónomos es decir una combinación entre protocolos de enrutamiento tanto internos como externo que en este caso es BGP.

Referencias Bibliográficas.

Teare, D., Vachon B., Graziani, R. (2015). CISCO Press (Ed). Path Control Implementation. Implementing Cisco IP Routing (ROUTE) Foundation Learning Guide CCNP ROUTE 300-101. Recuperado de https://1drv.ms/b/s!AmIJYei-NT1IlnMfy2rhPZHwEoWx

Teare, D., Vachon B., Graziani, R. (2015). CISCO Press (Ed). Implementing a Border Gateway Protocol (BGP) Solution for ISP Connectivity. Implementing Cisco IP Routing (ROUTE) Foundation Learning Guide CCNP ROUTE 300-101. Recuperado d https://1drv.ms/b/s!AmIJYei-NT1IlnMfy2rhPZHwEoWx

Teare, D., Vachon B., Graziani, R. (2015). CISCO Press (Ed). Implementing Routing Facilities for Branch Offices and Mobile Workers. Implementing Cisco IP Routing (ROUTE) Foundation Learning Guide CCNP ROUTE 300-101. Recuperado de https://1drv.ms/b/s!AmIJYei-NT1IlnMfy2rhPZHwEoWx

Teare, D., Vachon B., Graziani, R. (2015). CISCO Press (Ed). Implementing IPv6 in the Enterprise Network. Implementing Cisco IP Routing (ROUTE) Foundation Learning Guide CCNP ROUTE 300-101. Recuperado de https://1drv.ms/b/s!AmIJYei-NT1IlnMfy2rhPZHwEoWx

Froom, R., Frahim, E. (2015). CISCO Press (Ed). Fundamentals Review. Implementing Cisco IP Switched Networks (SWITCH) Foundation Learning Guide CCNP SWITCH 300-115. Recuperado de https://1drv.ms/b/s!AmIJYei-NT1IlnWR0hoMxgBNv1CJ

Froom, R., Frahim, E. (2015). CISCO Press (Ed). Campus Network Design Fundamentals. Implementing Cisco IP Switched Networks (SWITCH) Foundation Learning Guide CCNP SWITCH 300-115. Recuperado de https://1drv.ms/b/s!AmIJYei-NT1IlnWR0hoMxgBNv1CJ

Froom, R., Frahim, E. (2015). CISCO Press (Ed). Campus Network Architecture. Implementing Cisco IP Switched Networks (SWITCH) Foundation Learning Guide CCNP SWITCH 300-115. Recuperado de https://1drv.ms/b/s!AmIJYei-NT1IlnWR0hoMxgBNv1CJ

Froom, R., Frahim, E. (2015). CISCO Press (Ed). Spanning Tree Implementation. Implementing Cisco IP Switched Networks (SWITCH) Foundation Learning Guide CCNP SWITCH 300-115. Recuperado de https://1drv.ms/b/s!AmIJYei-NT1IlnWR0hoMxgBNv1CJ

Froom, R., Frahim, E. (2015). CISCO Press (Ed). InterVLAN Routing. Implementing Cisco IP Switched Networks (SWITCH) Foundation Learning Guide CCNP SWITCH 300-115. Recuperado de https://1drv.ms/b/s!AmIJYei-NT1IlnWR0hoMxgBNv1CJ