Post 10

Thursday, January 24, 2019 11:13 PM

INTER-VLAN ROUTING P1

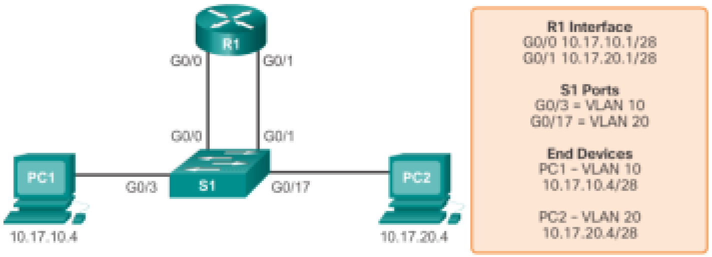
Inter-VLAN Routing P1

Trunk: Info from VLANs between devices: L3 process required for traffic to move from 1 network segment to other

Inter-VLAN routing: Process of fwding traffic from 1 VLAN to another using routing

VLANS	○ Segments switched networks: L2 switch [4096 VLANs]: Limited IPv4/6 function ○ L2 switch: NO routing function: No dynamic routing [only static] ○ VLANs: Broadcast domains: Separate VLANs: Unable to communicate w/out routing device ○ L3 multilayer switches/routers perform functions

Legacy Inter-VLAN Routing



R1 Interface
G0/0 10.17.10.1/28
G0/1 10.17.20.1/28

S1 Ports
G0/3 = VLAN 10
G0/17 = VLAN 20

End Devices
PC1 - VLAN 10
10.17.10.4/28

PC2 - VLAN 20
10.17.20.4/28

1st solution: Not implemented anymore in switched networks: As # of VLANs increases: Exhausts physical int capacity

- Router w/multiple physical ints: Each int connected to separate network/config'd w/distinct subnet
- Routing performed by connecting different physical router ints to different physical switch ports

Switch ports connected? Access mode: Each physical int assigned different VLAN

- Each router int: Accept traffic from VLAN associated w/switch connected
- Traffic: Router to other VLANs connected to other ints

More

- Multiple physical ints: Routes by having each physical int connected to unique VLAN
- Each int config'd w/IP for subnet associated w/a VLAN
- Config IP on physical ints: Devices connect to each VLAN: Communicates w/router using physical int connected on same vlan
- Network devices: Can use router as gateway to access devices connected to other VLANs

How?	▪ Source device: Determines if dest local/remote to local subnet: Compares source/dest IP against mask When destination IP determined to be on remote network ▪ Source ID's where to fwd packet to reach destination ▪ Source examines routing table: Determines where data sent ▪ Devices use gateway as L2 destination for all traffic ▪ IP of router int on local subnet acts as gateway for sending **Gateway:** Route devices use when no defined route to destination network When source determines packet must travel through local router int on connected VLAN: ▪ Source sends ARP request to determine MAC of local router int ▪ Router sends ARP reply back: Source uses MAC to finish framing packet before sending unicast If eth0 frame has destination MAC of router int: ▪ Switch knows port to fwd unicast traffic out of to reach router int for VLAN When frame arrives at router: ▪ Router removes source/dest MAC info to examine dest IP of packet ▪ Router compares dest to entries in routing table to determine where to fwd

	data for final destination If router determines network locally connected [w/inter-vlan routing]: ○ Router sends ARP request out int physically connected to destination VLAN ○ Dest device responds to router w/MAC: Router uses to frame packet ○ Router sends unicast to switch: Fwds out port where destination device connected

Switch Config:
Create VLANs vlan vlan_id [*global config*]
Assign switch ports to appropriate VLANs switchport access vlan vlan_id [*int config*]

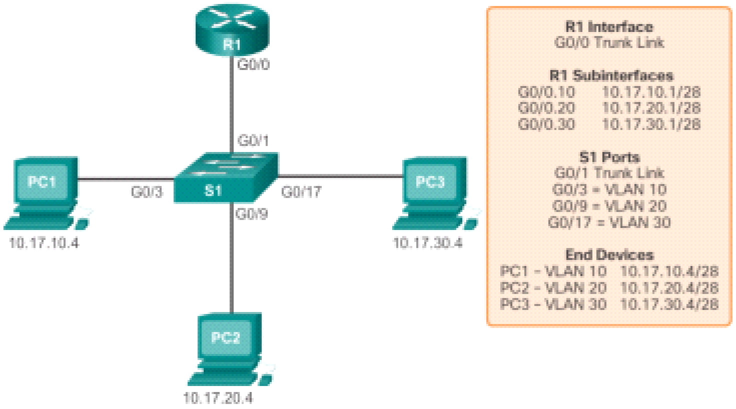
```
switch(config)# vlan 10
switch(config-vlan)# vlan 30
switch(config-vlan)# int f0/11
switch(config-if)# switchport access vlan 10
switch(config-if)# int f0/4
switch(config-if)# switchport access vlan 10
switch(config-if)# int f0/6
switch(config-if)# switchport access vlan 30
switch(config-if)# int f0/5
switch(config-if)# switchport access vlan 30
switch(config-if)# end
```

Router config: Similar to VLAN ints on switches: Repeat for all router ints: Each int must be assigned to unique subnet

```
router(config)# int g0/0
router(config-if)# ip address 172.17.10.1 255.255.255.0
router(config-if)# no shutdown
router(config-if)# int g0/1
router(config-if)# ip address 172.17.30.1 255.255.255.0
router(config-if)# no shutdown
router(config-if)# end
```

Check routing table show ip route
C Indicates route local for connected int: Also ID'd in route entry

Router-on-a-Stick: More common implementation of inter-VLAN routing



R1 Interface
G0/0 Trunk Link

R1 Subinterfaces
G0/0.10 10.17.10.1/28
G0/0.20 10.17.20.1/28
G0/0.30 10.17.30.1/28

S1 Ports
G0/1 Trunk Link
G0/3 = VLAN 10
G0/9 = VLAN 20
G0/17 = VLAN 30

End Devices
PC1 - VLAN 10 10.17.10.4/28
PC2 - VLAN 20 10.17.20.4/28
PC3 - VLAN 30 10.17.30.4/28

- SW may permit config router int as trunk link: Only 1 physical int on router/switch to route packets between multiple VLANs

Single physical int routes traffic between multiple VLANs on network

- Router int config'd to act as trunk link: Connected to switch port config'd in trunk mode
- Router performs inter-VLAN by accepting VLAN-tagged traffic on trunk int coming from adjacent switch
- Internally routes between VLANs using subinterfaces: Fwds traffic
- VLAN-tagged for destination VLAN: Out of same physical int used to receive traffic

Subinterfaces: SW based virtual ints: Associated w/single physical int

- Each subint independently config'd w/IP/VLAN assignment
- Config'd for diff subnets w/VLAN assignment: Logical routing
- After decision made based on destination VLAN: Data frames VLAN-tagged: Sent back out physical int

Advantages: Using trunk links/subints decreases # of router/switch ports used: Saves money/reduces complexity/scalable

How?	VLAN trunking with subints: • Allows single physical router int to route traffic for multiple VLANs • Uses virtual subints on router to overcome HW limits: SW-based virtual ints • Assigned to physical ints • Each subint config'd independently w/own IP/mask • Allows single physical int to simultaneously be part of multiple logical networks When using router-on-stick: • Physical int of router must be connected to trunk link on adjacent switch • On router: Subints created for each unique VLAN • Each subint assigned IP/specific to its subnet/VLAN/to tag frames for that VLAN • Router can keep traffic from each subint separated as traverses trunk link back to switch

Config Switch

- Start by enabling trunking on switch port connected to router

If router doesn't support DTP (Dynamic Trunking Protocol):

- Following cmds can't be used: switchport mode dynamic auto | switchport mode dynamic desirable

```
switch(config)# vlan 10
switch(config-vlan)# vlan 30
switch(config-vlan)# int f0/5
switch(config-if)# switchport mode trunk
switch(config-if)# end
```

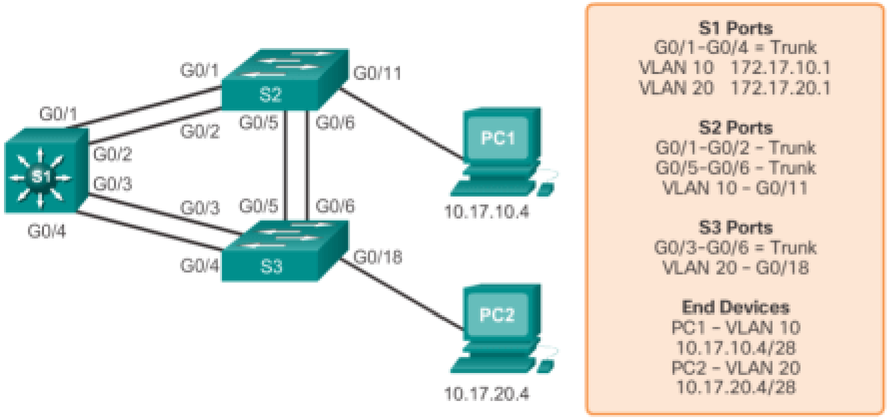
Config Router: Different compared to legacy

- Each subint created using int int_id subint_id [*global config*]
- Subint # Configurable: Reflects VLAN #
- Before assigning IP to subint: encapsulation dot1q vlan_id
- Native keyword option: Can appended to cmd to set IEEE 802.1Q native VLAN
- Each router subint must be assigned IP on unique subnet
- Physical int disabled? All subints disabled

```
router(config)# int g0/0.10
router(config-subif)# encapsulation dot1q 10
router(config-subif)# ip address 172.17.10.1 255.255.255.0
router(config-subif)# int g0/0.30
router(config-subif)# encapsulation dot1q 30
router(config-subif)# ip address 172.17.30.1 255.255.255.0
router(config)# int g0/0
router(config)# no shutdown
```

```
show vlan
show vlan br
show route
show ip route
```

Multilayer Switch



S1 Ports
G0/1-G0/4 = Trunk
VLAN 10   172.17.10.1
VLAN 20   172.17.20.1

S2 Ports
G0/1-G0/2 - Trunk
G0/5-G0/6 - Trunk
VLAN 10 - G0/11

S3 Ports
G0/3-G0/6 = Trunk
VLAN 20 - G0/18

End Devices
PC1 - VLAN 10
10.17.10.4/28
PC2 - VLAN 20
10.17.20.4/28

Router-on-stick: Requires 1 physical int on router: 1 int on switch: Simple cabling

- Other implementations don't require dedicated router
- Multilayer switch: Can perform L2/3 functions
- Supports dynamic routing: Inter-VLAN routing

To enable switch to perform routing functions: Switch must have IP routing enabled

- More scalable: Routers have limited #'s of ports to connect to
- For ints config'd as trunk line: Limited amts of traffic accommodated on line at once

Traffic routed internally to switch device: Packets not filtered down single trunk to obtain new VLAN tag info

- Doesn't replace router (they don't implement security measures/controls routers do)
- L2 device w/some routing capabilities

2960 Catalyst switch: Enable routing functionality
switch(config)# sdm prefer lanbase-routing [Catalyst 3560 supports EIGRP/OSPF/BGP]

Ping / tracert

Ping	○ Sends ICMP echo requests to destination address ▪ Host receives ICMP echo request ▪ Responds w/ICMP echo reply to confirm received ○ Calculates time using difference between time echo request sent/reply received ○ Elapsed time tests latency
Tracert	○ Good for confirming routed path taken between devices ○ Displays list of ingress router ints ICMP echo requests reached ▪ **UNIX:** traceroute ○ ICMP determines path w/specific TTL values defined on frame ▪ TTL value: Determines how many hops ICMP echo can reach ▪ 1st ICMP: Sent w/TTL value that expires at 1st router on destination ▪ ICMP echo request times out 1st route ▪ ICMP msg sent from router to originating device ▪ Device records response from router ▪ Sends another ICMP echo request: Greater TTL ▪ Allows request to bypass 1st router to reach 2nd device ▪ Rinse/Repeat