Student Name_____ Student Roll Num_____

LAB # 10: Inter-VLAN Routing (Using Router-on-Stick)

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

There are three inter-VLAN routing options:

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

Legacy Inter-VLAN Routing

The first inter-VLAN routing solution relied on using a router with multiple Ethernet interfaces. Each router interface was connected to a switch port in different VLANs. The router interfaces served as the default gateways to the local hosts on the VLAN subnet.

Legacy inter-VLAN routing using physical interfaces works, but it has a significant limitation. It is not reasonably scalable because routers have a limited number of physical interfaces. Requiring one physical router interface per VLAN quickly exhausts the physical interface capacity of a router.

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

Inter-VLAN Routing on a Layer 3 Switch

The modern method of performing inter-VLAN routing is to use Layer 3 switches and switched virtual interfaces (SVI). An SVI is a virtual interface that is configured on a Layer 3 switch which is also called multilayer switch because it operates at Layer 2 and Layer 3.

Router-on-a-Stick Inter-VLAN Routing

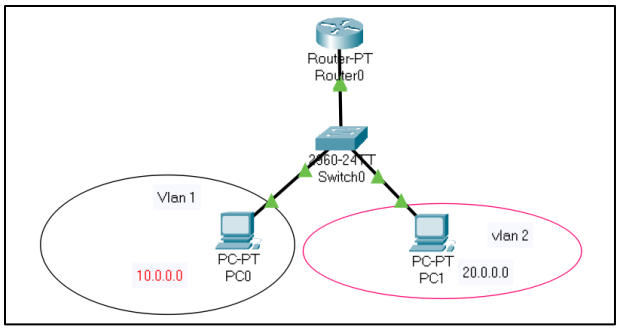
The "router-on-a-stick" inter-VLAN routing method overcomes the limitation of the legacy inter-VLAN routing method. It requires only one physical Ethernet interface to route traffic between multiple VLANs on a network.

A Cisco IOS router Ethernet interface is configured as an 802.1Q trunk and connected to a trunk port on a Layer 2 switch. Specifically, the router interface is configured using subinterfaces to identify routable VLANs.

The configured subinterfaces are software-based virtual interfaces. Each is associated with a single physical Ethernet interface. Subinterfaces are configured in software on a router. Each subinterface is independently configured with an IP address and VLAN assignment. Subinterfaces are configured for different subnets that correspond to their VLAN assignment. This facilitates logical routing.

When VLAN-tagged traffic enters the router interface, it is forwarded to the VLAN subinterface. After a routing decision is made based on the destination IP network address, the router determines the exit interface for the traffic. If the exit interface is configured as an 802.1Q subinterface, the data frames are VLAN-tagged with the new VLAN and sent back out the physical interface.

Following figure shows an example of router-on-a-stick inter-VLAN routing PC0 on VLAN 1 is communicating with PC1 on VLAN 2 through router Router0 using a single, physical router interface which is divided into sub-interfaces, which acts as a default gateway to their respective VLANs.



Configuration on switch:

Creating VLANs

Switch(config)# vlan 2
(default VLAN is VLAN 1 , so don't need to create)

Switch(config)# int range fa0/2-4
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 2
Switch(config-if)#exit

On switch0 we need to define the interface connected to the router as a **trunk link.** This will allow traffic from all VLANs to get to the router using that interface. (In our case it is fa0/1)

Switch(config)# int range fa0/1
Switch(config-if)# switchport mode trunk

Now, we will make sub-interface of fa0/0 as fa0/0.1 and fa0/0.2 and assign IP address as 10.0.0.1/8 and 20.0.0.1/8 respectively on router's logical ports.

Each subinterface is created using the interface interface_id.Subinterface_id in the global configuration mode. As shown below.

```
Router(config)#interface <interface_ID.Subinterface_ID>
```

NOTE: the "." Between the interface ID and the subinterface ID is a must. The subinterface ID is a logical number but ideally it should describe the VLAN ID.

Configuration on Router:

```
Router>en
Router#config t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface fa0/0.1
Router(config-subif)#encapsulation dot1q 1
Router(config-subif)#ip address 10.0.0.1 255.0.0.0
Router(config-subif)#no shut
Router(config-subif)#exit
Router(config)#interface fa0/0.2
Router(config-subif)#encapsulation dot1q 2
Router(config-subif)#ip address 20.0.0.1 255.0.0.0
Router(config-subif)#exit
Router(config)#interface fa0/0
Router(config-if)#no shut
```

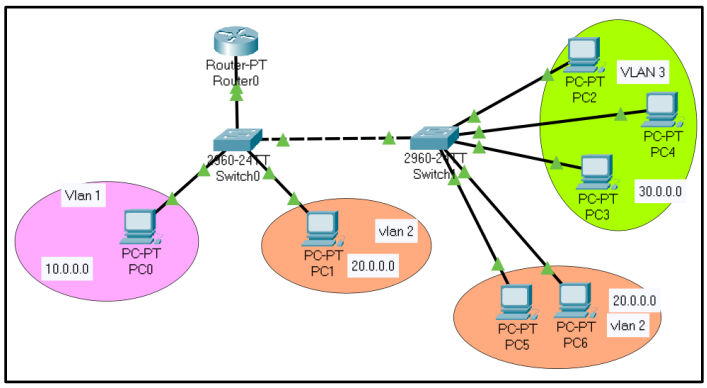
Use only FastEthernet 0/0 interface for both VLANs. We have created the Fa0/0.10 and Fa0/0.30 subinterfaces, specified the encapsulation type **dot1q** which is IEEE's 802.1Q, and the VLAN they belong to and we assigned an IP address. In this case, the physical interface, FastEthernet 0/0, does not need an IP address configuration, the only thing you must do is to use the **no shutdown** command so that the interfaces come up.

To display the information of fa0/0 interface after router configuration; use ***show interfaces*** command:

```
FastEthernet0/0.1 is up, line protocol is up (connected)
  Hardware is PQUICC_FEC, address is 00e0.8f2e.511e (bia 00e0.8f2e.511e)
  Internet address is 10.0.0.1/8
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation 802.1Q Virtual LAN, Vlan ID 1
  ARP type: ARPA, ARP Timeout 04:00:00,
  Last clearing of "show interface" counters never
FastEthernet0/0.2 is up, line protocol is up (connected)
  Hardware is PQUICC_FEC, address is 00e0.8f2e.511e (bia 00e0.8f2e.511e)
  Internet address is 20.0.0.1/8
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation 802.1Q Virtual LAN, Vlan ID 2
```

Lab Task:

1. Cable a network that is similar to the one in the given topology diagram.
2. Configure router and switches to perform inter-VLAN Routing using Router-on-stick.



3. What is 802.1Q Encapsulation? Briefly explain the purpose of the following command:
   encapsulation dot1q