Project – (250 Points Total):

This is an extremely detail oriented assignment! Please spend the time necessary to complete this assignment as thoroughly as you possibly can since it will be most beneficial to you in the long run!

There are a ten topics here. Each topic is worth twenty (25) points and may include multiple technologies within the topic. You may need to research certain topics and others you may have within your class notes from the labs and lectures. For each topic you should:

1. Provide a detailed description and/or explanation of what the topic is (in your own words). If there are multiple technologies in use, make sure you describe each one individually.
2. Following the description/explanation you will create an example topology that will be used for this question. (Yes, you may utilize this same topology on other questions – only if it directly relates to other questions.) Make sure you specify interfaces in use, ip addresses and subnet masks within your topology. Your topology should pertain to the topic and should not have more technologies configured within the topology than is necessary to demonstrate your mastery of the topic discussed.
3. Next, you will provide a configuration for the topic listed based upon your example topology.
   1. For the first few questions, you should provide detailed steps as to where you are within the router or switch when issuing a specific command so you demonstrate basic knowledge and manipulation of the Cisco IOS. Throughout all of the topics you should specify what each command does. (A brief example is listed below.)
4. Provide show commands for each topic. What are you looking for in order to verify this technology is working properly?
5. Provide debug commands for each topic. And, explain how you would go about troubleshooting this technology if something is not working correctly.

Please keep in mind… No assumptions are made! Some of the items have multiple steps so be sure to include **all** the steps in the configuration, etc...

**For example** *(please keep in mind this is abbreviated example):*

If you are asked to ***assign an IP Address to a router interface***, you may want to specify the following:

Create a sample topology, then write…

An IP Address is a Layer 3 logical address that is assigned to each host within a TCP/IP network. The IP Address assigned to the host is determined by the network administrator and is dependent upon the network it is directly connected to. An IP Address is a hierarchical address that consists of 4 octets, 8 bits each, 32 bits in total. An IP Address is typically written in dotted decimal notation:

* + - 1. is an example of a Class A IP Address

To assign an IP Address to a router interface you would proceed in the following manner:

|  |  |
| --- | --- |
| Router> | This prompt specifies the router is in USER mode |
| Router>***enable*** | this command is entered to go from USER mode to PRIVILEGE mode |
| Router# | You are now in PRIVILEGE mode |
| Router# ***configure terminal*** | used to enter GLOBAL CONFIGURATION mode from PRIVILEGE mode. This is needed to configure all parameters within the router |
| Router(config)# | You are now in GLOBAL CONFIGURATION mode |
| Router(config)#***interface fastethernet0/0*** | This is entered to go into INTERFACE CONFIGURATION mode to configure the fastethernet0/0 interface within the router |
| Router(config-if)# | You are now in INTERFACE CONFIGURATION mode |
| Router(config-if)#***ip address 1.1.1.1 255.0.0.0*** | This sets the IP Address to 1.1.1.1 with a subnet mask of 255.0.0.0 which is the default subnet mask for a Class A IP Address |
| Router(config-if)#***no shutdown*** | After the IP Address is set on the interface, it’s always a good idea to enable the interface and test connectivity. |

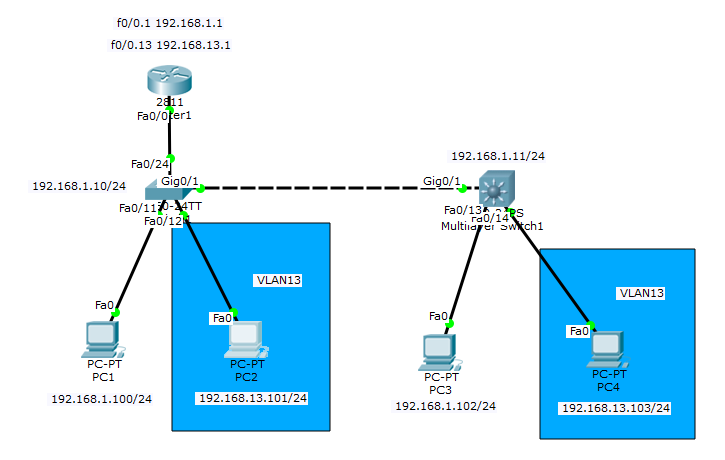
Specify the show commands, test commands to verify the configuration is working properly and move onto the next topic.

This assignment should be **printed and turned in by 9:00am, Saturday, April 29, 2017.** **In addition to your printed copy, please submit a softcopy in Microsoft Word (.docx or .doc) format via email to robert.cannistra@marist.edu.** Make sure you cite your resources and do not plagiarize! Your citations should be placed at the end of each question, not at the end of the project. **All work should be completed individually!!!** **There will not be any extensions given for this assignment**, so make sure you **SUBMIT IT ON TIME otherwise you will receive a grade of ZERO!!!**

If you have any questions or concerns, please let me know!

Topics to research, explain, provide detail and provide examples for: (make sure you provide as much detail as possible in a concise manner and make it as legible as possible! Once this is graded and turned back to you, it will provide you with a detailed study guide for your final practical.) Please feel free to copy and paste the table from above and use it as a template for each question. Please do not remove the numbers or questions/statements below. Use this as a template and add your answers under each numbered question/statement on each page.

1. VLANs, VTP, and IEEE 802.1q (NOTE: Make sure you accomplish this ***without*** going into VLAN Database mode and show the syntax for both 2960 and 3560 switches!)

To configure the question above I used 2 switches, a router, and 4 PCs. VLANs are making groups of PCs into a subnetwork so the PCs in one VLAN can only communicate with those in the same VLAN group. To accomplish the question at hand we assigned IPs to each computer, and interface needed as well as used sub interfaces on f0/0 of the router so we can have multiple IP addresses pointed towards that one port. The VTP and IEEE 802.1q comes into play when configuring the switches. The 2960 switch used was configured as the server in the VTP configuration. Additioanlly, the 2960 switch does not needed to be configured to 802.1q because it has the configuration already on it. The 3560 switch needed the IEEE configuration added and was treated as the VTP client in this topology.

|  |  |  |
| --- | --- | --- |
| Basic Commands | Command | Description |
|  | Router>enable | Enters a user into privileged mode |
|  | Router#Configure Terminal | Enters a privileged user into the configuration mode |
|  | Router(config)#hostname Router1 | Sets the hostname of the Router to Router1 |
|  | Router(config)#enable secret cisco | Sets an encrypted password when a user tries to enter Privileged mode |
|  | Router(config)#line con 0 | Enters into the configuring of the lines when booting into CLI |
|  | Router(config-line)#exec-timeout 0 0 | Sets the timeout to never timeout |
|  | Router(config-line)#password cisco | Sets the password at the base line in CLI |
|  | Router(config-line)#logging synchronous | This prevents every logging output from interrupting your console session |
|  | Router(config-line)#login | Enables a login prompt at base line of CLI |
|  | Router(config)#line vty 0 4 | Enters into the configuring of lines 0-4 in TELNET |
|  | Router(config-line)#exec-timeout 60 0 | Sets the timeout to 60 minutes |
|  | Router(config)#int f0/0.1 | Enters selected physical interface or subinterface |
|  | Router(config-if)# [ip address] [Subnet Mask] | Sets the ip address of the interface selected |
|  | Router(config-if)# no [ip address] | Deletes current ip address or ip address selected |
|  | Router(config-if)# no shutdown | Turns on the selected interface |
| VTP Commands | Switch(config)# vtp mode [server/client] | Sets the VTP mode |
|  | Switch(config)# vtp domain [INETLAB] | Sets the VTP domain |
|  | Switch(config)# vtp password cisco | Sets the VTP password |
| VLAN Commands | Switch(config-if)# switchport mode access | Enables device to carry traffic in a VLAN |
|  | Switch(config-if)#switchport access vlan # | Sets the interface to specified VLAN |
|  | Switch(config)int vlan # | Enters specified VLAN |
|  | Switch(config-if)#ip address [ip address] [subnet mask] | Sets ip address of VLAN |
|  | Switch(config-if)#no shutdown | Enables the VLAN |
|  | Switch(config)#vlan # | Enters VLAN Group |
|  | Switch(config-vlan)#name BLUE | Sets VLAN name to BLUE |
| IEEE 802.1q | Switch(config-if)switchport trunk encapsulation dot1q | Sets Trunking encapsulation to 802.1q standards |
|  | Switch(config-if)switchport mode trunk | Enables the current interface to trunk |

|  |  |  |
| --- | --- | --- |
| Show Commands | Commands | Explanation |
|  | Switch#show ip int brief | Shows current interfaces enabled and IP addresses set to those interfaces |
|  | Switch#show vlan | Shows table of VLANs, current interfaces in each, and names |
|  | Switch#show run | Shows current running configuration for the device |
|  | Ping [ip address] | Pings selected ip address |

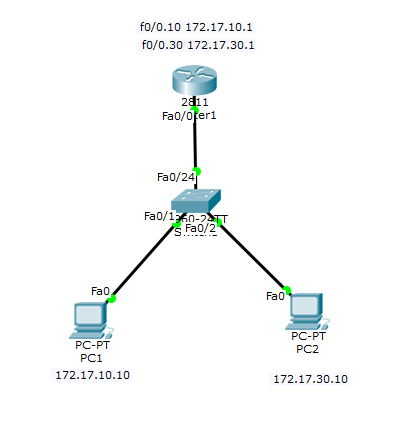
Using these show commands I was able to verify each interface had the correct IP address as well as each VLAN had the correct interfaces associated with them. Ping was the final reveal if the VLANs worked. Pinging from PC1 to PC3 shows connectivity, but pinging from PC1 to PC2 or PC4 should not show connectivity and vice versa for PC2 and PC4.

**Debug Syntax:**

Show run: Show run was a huge debugging, one of my errors was I didn’t put an interface into trunking mode which lead to the whole network not being able to trunk. Once that was resolved it worked like a charm.

Show vlan: Show vlan was used in debugging if my vlans were created and named properly as well as making sure the interfaces I wanted were shown in the VLAN.

1. InterVLAN Routing with a Router-on-a-stick and SVIs

In this problem we utilized a router, switch, and 2 PCs. As we saw from problem 1, the goal of a VLAN is to separate networks of PCs to only be able to communicate with one another. The problem here is we want to have InterVLAN connectivity. The router on a stick below shows perfectly how to do it. The use of sub interfaces on the router makes it so even if the PCs are separated into VLANs they can still connect to each other. Instead of setting an ip address onto a VLAN group, we set an ip address on the sub interface and made it so it would trunk to the specific VLAN we wanted. To finish the connection, we had to make sure the actual switch was turned onto trunking on f0/24. Once that was configured we have full connectivity.

|  |  |  |
| --- | --- | --- |
| Vlan commands | Command | Description |
|  | Switch(config-if)#switchport mode access | Enables an interface to use VLAN |
|  | Switch(config-if)#switchport access vlan # | Assigns an interface to a VLAN |
|  | Switch(config-if)#switchport mode trunk | Enables an interface to trunk |
|  | Router(config)#int f0/0.# | Enables and enters into a sub interface of a physical interface. |
|  | Router(config-if)#encapsulation dot1q [vlan #] | Sets the sub interface into dot1q and assigns it to a vlan # |

|  |  |  |
| --- | --- | --- |
| Show Commands | Commands | Explanation |
|  | Switch/Router#show ip int brief | Shows current interfaces enabled and IP addresses set to those interfaces |
|  | Switch#show vlan | Shows table of VLANs, current interfaces in each, and names |
|  | Switch#show run | Shows current running configuration for the device |
|  | Ping [ip address] | Pings selected ip address |

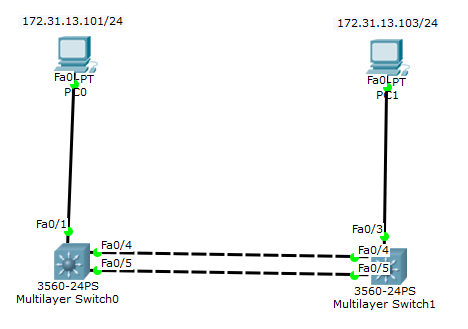
In testing this topology the ultimate goal was getting connectivity from PC1 to PC2 having them in separate VLAN groups. To test this I would ping either address from the PC, if it failed there was something wrong. Once I verified everything was good, the ping went through and came back.

**Debug Syntax:**

**Show vlan:** I used this command a lot to see if my VLANs were grouped correctly and to make sure the correct interfaces were assigned to the groups they needed to be.

**Show run:** This was a major one in seeing if the configurations went through because a big problem was that I could not see what the sub interfaces were configured to, using show run I could see the sub interfaces and how they were configured so I could make changes accordingly.

1. First Hop Redundancy Protocol with HSRP using Link Aggregation between two 3560 multilayer switches. Demonstrate your knowledge of LACP and PAgP.

In this problem we are focusing on HSRP and the channel protocols LACP and PAgP. To complete this I used 2 3560 switches and 2 PCs. The end goal of this is to be able to ping to each PC. To get to this goal we utilized First Hop Redundancy Protocols LACP and PAgP. Using these protocols and HSRP we achieved full connectivity.

|  |  |  |
| --- | --- | --- |
| PAgP Commands | Command | Description |
|  | Switch(config-if)#channel-protocol pacp | Declares the protocol |
|  | Switch(config-if)#channel-group # mode desirable | Command assigns the interface to an etherchannel and declares what protocol will be used. |
|  | Switch(config)#int port-channel # | Creates a portchannel is one does not exist, and enters into that channel |
|  | Switch(config-if)#switchport trunk encap dot1q | Assigns 802.1q to portchannel |
|  | Switch(config-if)#switchport mode trunk | Sets etherchannel to trunking |
|  | Switch(config-if)#switchport native vlan # | Sets the native vlan to # specified |
| LACP Commands | Switch(config-if)#channel-protocol lacp | Declares the protocol |
|  | Switch(config-if)#channel-group # mode active | Command assigns the interface to an etherchannel and declares what protocol will be used |
|  | Switch(config)#spanning-tree vlan # root [primary/secondary] | Enables the spanning tree to be primary or secondary based on the switch being configured. |
| HSRP Commands | Switch(config)#ip routing | Enables routing on switch |
|  | Switch(config)int vlan # | Enters interface of the VLAN |
|  | Switch(config-if)#ip address [ip address] | Sets ip address for interface |
|  | Switch(config-if)#standby [vlan #] ip [ip address] | Sets IP address for standy, default gateway |
|  | Switch(config-if)#standby [vlan #] priority [105/115] | Sets standby priority by number listed |
|  | Switch(config-if)#standby [vlan #] preempt | Enables the HSRP with the highest priority to immediately become the active router |

|  |  |  |
| --- | --- | --- |
| HSRP, LACP, PAGP | Commands | Explanation |
|  | Switch#Show etherchannel sum | Shows summary of all etherchannels on the switch and state of devices in the channels. |
|  | Switch#Show standy | Shows status of standby for HSRP |
|  | Switch#Show int trunk | Shows what portchannel or interfaces are trunking |

To see the end result of this problem I had to be able to ping from PC1 to PC2. Through this we could swap out PAGP or LACP interchangeably with a simple change in commands both giving full connectivity. I used all the commands above to make sure my configuration was working and proper in state. All were used and the ping was successful after a few tweaks were made.

**Debug Syntax**

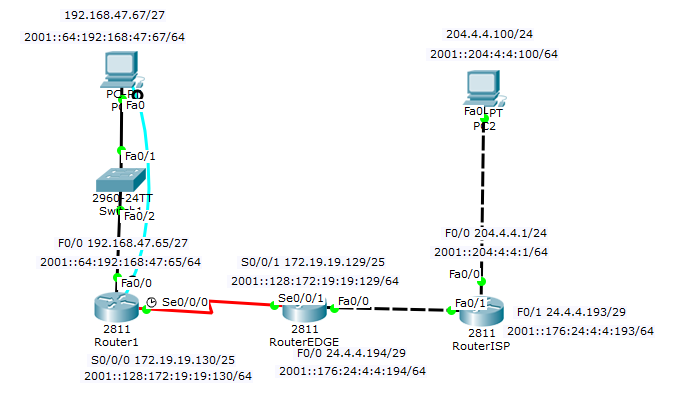
**Ping:**  ultimate goal here was to be able to ping to the other PC, we could have used either PAGP or LACP. To ping, it tests if we can or not and if not we can trouble shoot accordingly.

**Show etherchannel sum:** This command allowed me to see what the etherchannels had assigned and what state the devices in them were in. This allowed me to make changes to the interfaces in the etherchannels.

**Sh run:** Show run was used to make sure my switches were matching and the configuration was the same. This helps a lot with the fact if one of the VLANs is off, or IPs are different the connectivity could be broken.

1. IPv4 and IPv6 Static routing (NOTE: With AND Without the use of a default static route.)

The topic in this question is IPv4 and IPv6 Static routing. This is the method of having connectivity between computers and routers so different networks can be reached. These static routes make sure the routers can connect to networks that aren’t the ones they are in so a PC can essentially ping to a different network than its own. A default static route is used for any connection that doesn’t have a learned or static route it gets pushed to the default static route. To accomplish this, we needed 2 separate networks with a standard network containing a PC, switch, router and another network that contained a router in the middle as well as a router and PC. This enables us to have default static routes between the 2 routers and PC shown on the right of the topology and static routes to the router in the middle known as RouterEDGE. This topology enables us to use everything asked in the question above.



|  |  |  |
| --- | --- | --- |
| IP Routing | Commands | Explanation |
|  | Ipv6 unicast routing | Enables ipv6 addressing on the router at hand. |
|  | Ip route [dest. Network] [dest. SM] [next hop address] | Sets a static route using the syntax listed |
|  | Ipv6 route [dest address/SM] [Outgoing Int] | Sets ipv6 route using the syntax listed |
|  | Ip route 0.0.0.0 0.0.0.0 24.4.4.193 | Sets a default static route to address listed in last octet of syntax |
|  | Ipv6 route ::/0 [outgoing int] | Sets default static route to interface listed. |

|  |  |  |
| --- | --- | --- |
| Show Commands | Commands | Explanation |
|  | Show ip route | Shows current routes on the router |
|  | Show ipv6 route | Shows current ipv6 routes on the router |

Using these show commands you are will see all the routes for the router you are configuring. On here you should see the S marker for a static route listing the network it is reaching with that static route. Additionally, you should see S\* for a default static. In the topology from above the Edge and ISP router both have default static routes.

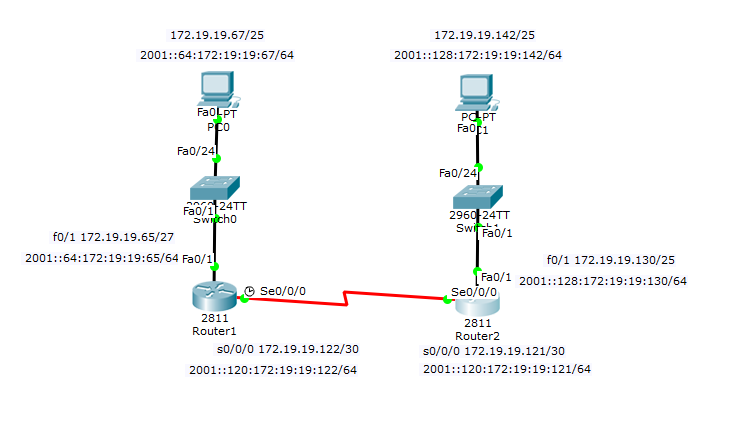
**Debug Syntax:**

**Ping**: Pinging in this situation is very useful being static routes are mainly used for connectivity between networks. If my PC was not able to ping my Edge router then the static routes I configured was not correct. Using this command I was able to troubleshoot and see a mistake in my ip address I typed in when configuring routing.

**Tracert**: Tracert is a command that is used to show the flow of connectivity through a ping. It can show where a connection is being dropped which is very helpful when configuring static routing. Using this command a user can see where the connection goes bad and debug their static routes starting there.

In addition to these two, the two show commands were also used in debugging to see if there were typos in address or to see what was routed and what was not.

1. EIGRP and EIGRP for IPv6 (NOTE: Make sure you utilize VLSM in your topology and configuration without the use of automatic summarization.)

In this problem we are faced with configuring a dynamic routing protocol, EIGRP. We are also asked to program EIGRP for IPv6. In this topology I used 2 PCs, 2 Switches, and 2 Routers. The goal is getting the PCs to ping each other even though they are in different networks. We accomplished this in a previous question using static routes which we have to manually enter. With EIGRP, you have to enter the network addresses of neighboring interfaces and the protocol configures the rest. To piggyback onto ipv6, it was rather simple applying it as well with just a couple more syntax lines added. The end result

was being able to ping from PC1 to PC2 using IPv4 and IPv6.

|  |  |  |
| --- | --- | --- |
| EIGRP commands | Command | Description |
|  | Router (config)#router eirgrp 307 |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| Show Commands | Commands | Explanation |
|  | Router#show run | Shows running config on file |
|  | Router#show ip route | Shows ipv4 routes |
|  | Router#show ipv6 route | Shows ipv6 routes |
|  | Ping [ip address] | Pings selected ip address |

The end goal of this problem was to have connectivity between PC1 and PC2. To test this we would need to be able to ping from PC1 to PC2. Additionally, we would need to be able to ping both ipv4 and ipv6 addresses. To check through my configs, I used show run multiple times to make sure both routers were identical with most of the configuration aside from some ip addresses. In addition to this, I checked both ipv4 and ipv6 routes to make sure all were in EIGRP protocol. Once I took these steps and had connectivity the problem was solved.

**Debug Syntax:**

**Show Run:** Show run was a big factor in this problem. I used it to find out which interface was missing ipv6 eigrp 307. In addition to this, I used it frequently to quickly check my network addresses listed in eigrp 307 to make sure they were correct and made sense. Once all IP addresses and networks were set I would use the next syntax.

**Show ip/ipv6 route:** I used this trouble shooting syntax to make sure my network addresses input were transferred into EIGRP routes. If one was not listed here I had to re check its syntax and conversion to make sure it showed on this list.

1. OSPFv2 and OSPFv3 (NOTE: Make sure you utilize loopback interfaces within your topology and configuration…and make sure you specify why you are utilizing them.)
2. Extended ACLs and Extended Named ACLs for IPv4 and IPv6 (NOTE: how, where and why you applying these…)
3. Extended ACLs and Extended Named ACLs are used for controlling traffic to and from servers. This is used in firewalls a lot because it can control bad traffic and cancel the request or let the request through if it is thorough. The technologies used in the example are servers, an Edge router, an ISP router, and a PC with a switch. The servers are used for sending and holding requests. For example, if a pc is trying to reach a webpage held on Server 2 it needs to go through server 1 and send out a request to access server 2 going through the Edge and ISP router. The switch makes it easy for a pc to be added to the scheme. The routers are where the ACLs are going to be stored, they are the ones that process and either deny or permit traffic through to the servers.
4. Static NAT and Dynamic NAT/PAT for IPv4 (NOTE: Make sure you show a configuration with static NAT for a web server in the DMZ, then PAT using a pool of multiple addresses for internal hosts connecting to the Internet. You should also demonstrate your knowledge of using just the interface IP Address when this is the only available address presented to you.)
5. DHCP Server on the Router and a local DHCP client and a remote DHCP client (ie: using DHCP Relay.)
6. PPP using PPP CHAP