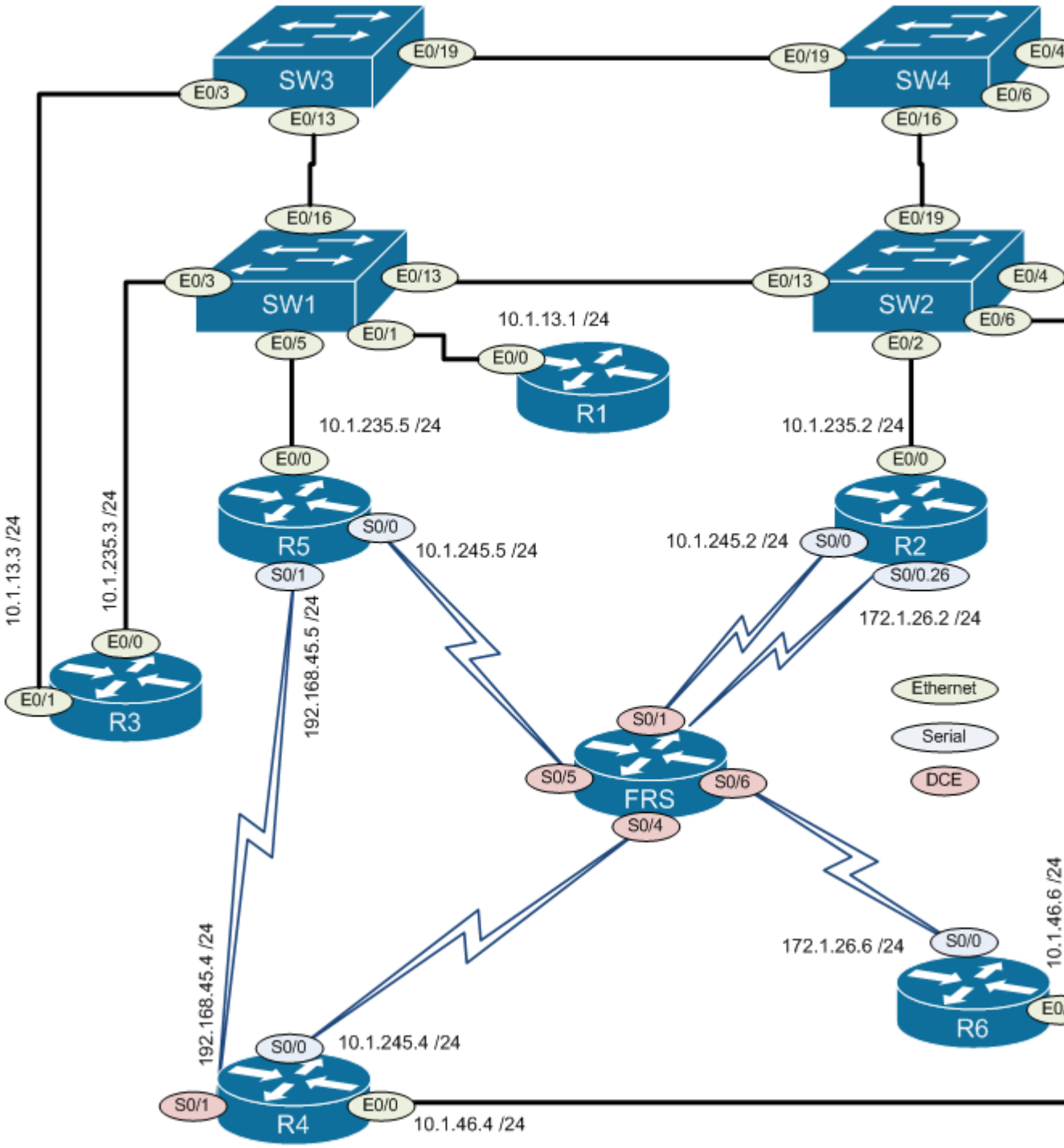
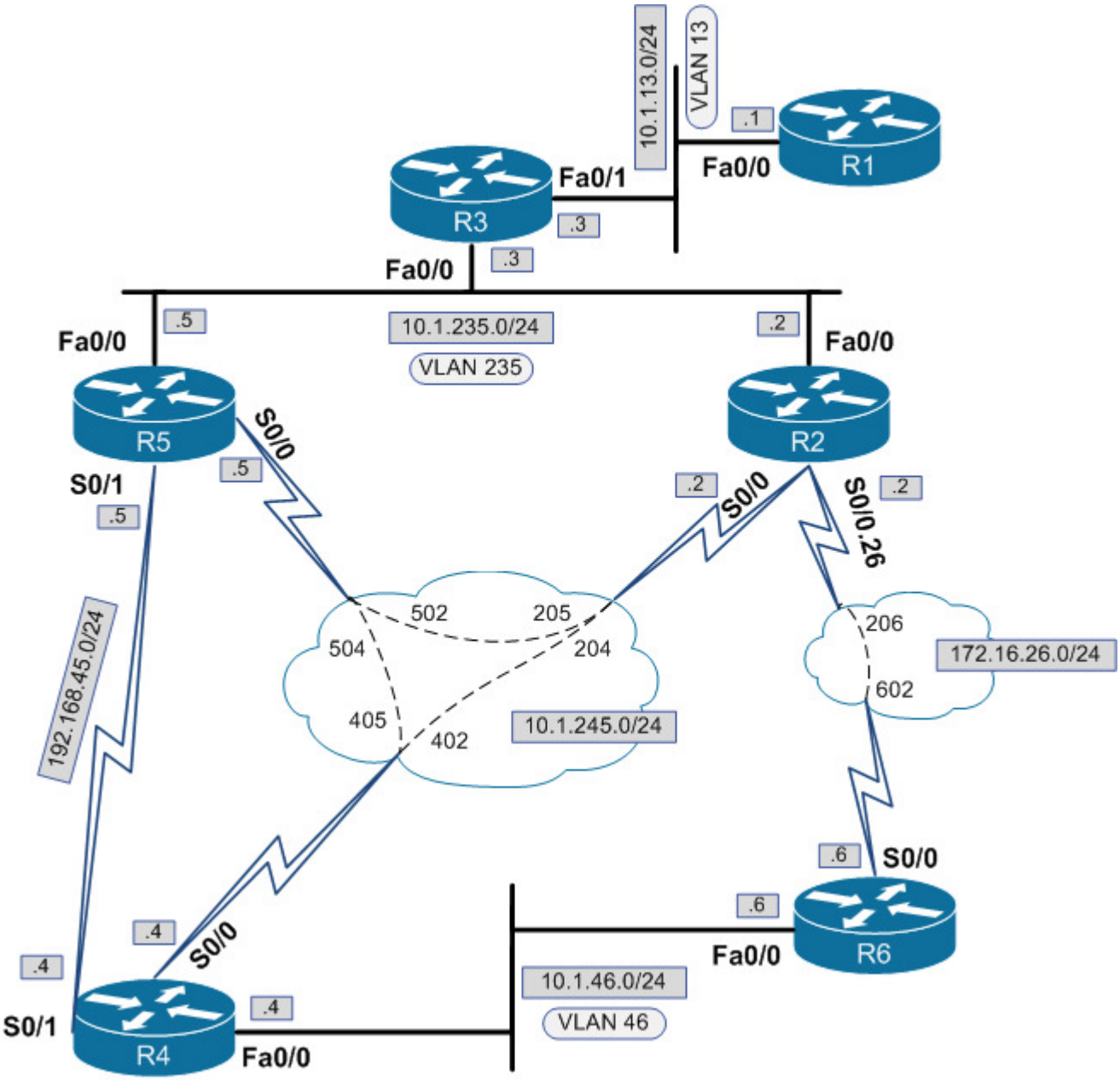
Technologies Covered

- Connect to your POD
- Base Configuration
- Interface Configuration
- Virtual LAN (VLAN) Configuration [Access & Trunk Ports]
- IP Addressing
- Frame Relay [Dynamic]
- RIP Routing
- TCL Scripts / Lab Verification


SW3
SW4
SW1
SW2
R1
R2
R3
R4
R5
R6
FRS
E0/19
E0/3
E0/13
E0/16
E0/4
E0/6
E0/5
E0/1
E0/0
E0/2
10.1.13.1 /24
10.1.235.5 /24
10.1.235.2 /24
10.1.13.3 /24
10.1.235.3 /24
10.1.245.5 /24
10.1.245.2 /24
S0/0
S0/1
S0/0.26
172.1.26.2 /24
192.168.45.5 /24
192.168.45.4 /24
S0/5
S0/6
S0/4
172.1.26.6 /24
10.1.46.6 /24
10.1.245.4 /24
10.1.46.4 /24
Ethernet
Serial
DCE



The purpose of this lab is to help you understand basic router and switch configuration, as well as the purpose of these devices. Your goal for this lab is to configure a fully routed network so that each device can communicate with every other device.

Do not worry if you do not fully understand how everything works for this lab. Topics covered in this lab may not be covered in class for several weeks. This lab is to give you a general overview of how networking works, and how the devices are configured. All of this material will be covered at least once more in future labs.

Important: When configuring passwords on any lab, on any device, always use: `ccie-1824`.

These Labs are designed to challenge your knowledge of the material. All of these commands can be found from the reading material. Your best bet is the CCNA Portable Command Guide. Should you not be able to find a command there, the Cisco documentation is the next best solution. Ask your TA for assistance using the Cisco documentation.

Connect to your POD

Connect to a POD using any of the methods mentioned in Lab 0.

Base Configuration

When starting a lab, there are some basic things that will need to be done on every device. The following should be done at the start of every lab:

- Configure the device hostname
- Disable domain-name lookup
- Configure passwords for console and virtual terminal lines and privileged exec mode
- Configure password encryption
- Set console timeout
- Configure "logging synchronous" on the console line

Interface Configuration

When a router boots with no configuration for the first time, all of its interfaces are shut down by default. When a switch boots with no configuration, all interfaces are ***not*** shut down by default. In order to facilitate proper communication between devices the interfaces used in this lab will be enabled; all other interfaces should be shutdown.

To start, shut down all interfaces on the switches. This is good practice as you should only utilize necessary interfaces. As you may have noticed, your switches have 24 fastEthernet interfaces. It would be tedious to shut down each one individually. Luckily with the `interface range` command, you can issue the same commands to multiple interfaces, simultaneously.

In Cisco IOS, commands are negated by pre-pending a command with `no`. Please look at the physical diagram and turn on the necessary interfaces on the routers and switches.

To bring up serial interfaces, additional commands are required. A serial interface represents a point to point link between two devices. Each side of a serial link is different; one side of the cable has a pin that provides a clock signal for the other device, the *Data Circuit-Terminating Equipment* (**DCE**). The other side is called the Data Terminal Equipment or **DTE**. To make serial interfaces work, you must configure a clock rate on the DCE side. On the pod diagrams, the DCE is denoted by a red circle. Please look at your physical diagram and set the clock rate. As a general rule, we will use a value of **64000** in our labs as that is the highest common clock rate on our equipment.

There are two ways to verify that an interface is not shutdown: 1 - A console message will appear.

> *Aug 11 20:38:41.019: %LINK-3-UPDOWN: Interface FastEthernet0/1, changed state to up

*Aug 11 20:38:42.131: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state to up

OR

*Aug 11 21:03:03.195: %LINK-3-UPDOWN: Interface Serial0/0/0, changed state to up

*Aug 11 21:03:04.195: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed state to up

2 - The `show ip interface brief` command. This command will tell you the state of each interface. Pay close attention to the *Status* and *Protocol* fields.

Status / Protocol

Administratively down / down This state indicates that the interface is shutdown.

down / down This state indicates that the interface is not shutdown. There may be a cable malfunction or no directly connected device.

up / down This state indicates that the interface is not shutdown. However, there is a problem with the protocol. There could be an encapsulation or configuration error.

up / up This state indicates that the interface is not shutdown. This is the desired state for a functional interface. However, other configuration errors may still be present.

Once you have all of the appropriate interfaces turned on, you can check to see if each device sees it's neighbors. Cisco devices use a proprietary protocol to allow devices to sense directly connected devices, **Cisco Discovery Protocol**. The **show cdp neighbors** command will show information about any directly connected devices that are sending and receiving **CDP Packets**. This will include hostnames and locally connected interfaces.

Virtual LAN (VLAN) Configuration [Access & Trunk Ports]

PLEASE NOTE: VLANS ARE NOT SAVED WITH THE SAVE ALL SCRIPT.

If you want to save the vlans save the vlan config that you write into a document and paste it into the console every lab.

In order for two devices on an Ethernet network to communicate, they must be on the same Local Area Network (LAN). On most switches that you may be familiar with, this means connecting devices to the same switch. This works great if you want every device to be able to talk with each other. Sometimes you may need to have two devices talking to each other, and another two devices talking to each other, but not be able to talk between the two groups. For instance, you have an accounting and a marketing department, and you want them to be able to communicate within their individual departments, but not between departments, for security reasons. You could buy another switch, put one department on one switch, and the other department on the other switch, or you could use Virtual Local

Area Networks (VLANs). By configuring ports for one department in one VLAN, and ports for the other department in another VLAN, you are separating the computers as if they were plugged into different switches. Devices in one VLAN cannot talk to devices in another VLAN without first going through a router. This would, for example, allow you to put security measures in place on the router to allow only approved traffic between the computers in different departments.

Now let's say this company is in a multi-story building. Unfortunately, the marketing and accounting departments are dispersed between two floors, and there is a switch on each floor. We could run two cables between them, one for the accounting VLAN, and one for the marketing VLAN, but this is not very scalable. Instead, we can use one cable between the two switches and setup VLAN trunking. What VLAN trunking allows us to do is to use one cable between two switches to transmit data from different VLANs while maintaining separation between those VLANs. The way this works is that when a switch sends data out of an interface configured as a trunk, it puts a special tag in the frame indicating what VLAN that data belongs to. When the adjacent switch receives this data frame, it looks at this tag to determine what VLAN it belongs to.

VLANs and switches operate at layer 2 of the OSI model. MAC addresses are layer 2 addresses used for communicating Ethernet LANs. There are several VLANs set up by default on switches.

VLANs are configured on a per interface basis on switches. Interfaces can be set up as either an *access port* or *trunk port*.

Access Ports A switch interface configured as an access port will only permit one VLAN. If you configure an interface to be an access port for VLAN X (where X is an arbitrary number), the switch will automatically create that VLAN; if it does not already exist. You will be notified of this by a console message.

Trunk Ports A switch interface configured as a trunk port will function just like an access port. However, this configuration is required for permitting multiple VLANs. Unlike an access port, trunk port configuration *WILL NOT* automatically create VLANs. Trunk ports are used for switch-to-switch connections, as well as switch-to-router connections that utilize sub-interfaces.

Trunk ports will need to configured with an encapsulation type. There are two types of encapsulation, ISL and dot1q. For the purposes of this lab, configure the trunk link between Switch1 and Switch2 with dot1q encapsulation. Allow only the necessary VLANs across this trunk link.

Please review the logical diagram and configure the switch interfaces appropriately.

After configuration, verify that all necessary interfaces are set up properly. Get into the habit of verifying everything. It is extremely important as it will help you troubleshoot and problems.

VLAN verification can be done with the following commands: **show vlan brief** and **show interfaces trunk**... **show vlan brief** will list all existing VLANs, their corresponding name, and the interfaces/ports they are associated with. **show interfaces trunk** will show the VLANs allowed on trunked ports as well as encapsulation type.

IP Addressing

IP addresses are hierarchical addresses used for communication between devices on different Layer 2 segments. Each Layer 2 segment is assigned a subnetwork of IP addresses, the size of these subnetworks is determined by the subnet mask. For the purposes of this lab all subnet masks will be set to 255.255.255.0 or a /24.

IP address are assigned per interface. Each interface can only have one IPv4 address. If you try to assign a second IP address, the previous one will get overwritten. Look at the diagram and assign IP addresses to all of the interfaces. Do not configure IP addresses over Frame Relay yet.

Once IP addresses are configured, you should verify Layer 2 connectivity. Use show commands to verify that the IP addresses are configured on the correct interfaces. Remember, at this point, you have configured VLANs and IP addresses. You should be able to ping IP addresses that are on the same VLAN. Routing Protocols have not been configured yet so you WILL NOT be able to ping across VLANs. Verify this with pings.

Frame Relay [Dynamic]

Frame Relay is a Layer 2 protocol used for Wide Area Networking. Companies with multiple offices will often use frame-relay in their Wide Area Network (WAN).

For addressing, Frame Relay uses a field called the Data Link Connection Identifier (**DLCI**). On the logical diagram, the numbers in the clouds represent the DLCI. For example, for R2 to send data to R4 it uses DLCI 204, and to send data to R5 it uses DLCI 205. For R5 to send data back to R2, it uses DLCI 502. The network between R4, R2 and R5 is called a full mesh network, as each router has a Permanent Virtual Circuit (**PVC**) to every other device. The network between R2 and R6 is called a point to point network as there is only one link between the two devices.

There are two forms of Frame Relay, Static and Dynamic. This Lab will cover Dynamic Frame Relay Switching. Dynamic mode is much simpler in terms of configuration. It also automatically maps out paths for frames to take.

There are two parts to Frame Relay [Dynamic] Configuration: The Frame Relay Switch(FRS) & the devices connecting to the FRS cloud. For visualization purposes, associate the cloud with the FRS. The devices connecting to the FRS cloud do not handle any of the actual Frame Relay Switching.

Start with the Frame Relay Switch's configuration. Then move on to configuring the connected devices.

Frame Relay Switch Configuration: Enable frame relay switching in *global configuration mode* (`frame-relay ?`); Encapsulate frame relay in *interface configuration mode* (`encapsulation ?`); Configure ROUTE statements (`frame-relay route local-DLCI interface opposite-interface other-DLCI`) in *interface configuration*

Devices connected to FRS Cloud Encapsulate frame relay in *interface configuration mode* (`encapsulation ?`)

***The above configuration is for Frame Relay on *PHYSICAL* interfaces. For Frame Relay on sub-interfaces, there are slight changes: Enable Frame Switching on the *PHYSICAL* interface Create a sub-interface [aka *LOGICAL* interface] and set it to "point-to-point" Assign an IP address on the *LOGICAL* interface. Assign it a DLCI using "frame-relay interface-dlci " command.

To verify, use **show frame-relay route** on the FRS and **show frame-relay map** on the connected devices. This will display the DLCI's available routes and statuses. Make sure all addresses on a common subnet can ping each other. You will not be able to ping yourself.

RIP Routing

At this point in the lab, each router can send data to routers that are adjacent to itself. In order for routers to know how to get to all of the networks, you must configure a routing protocol. A routing protocol advertises what networks that router knows how to get to. Other routers then learn this information, and are able to send data to those networks. In this lab we will be configuring the RIP routing protocol.

Configure RIP Version 2 on all routers. Make sure each router advertises all of its known networks. Do not summarize these networks. Be careful when advertising networks. Fat-fingering an incorrect network will not present an immediately noticeable error. Routing protocols will just add it to their advertisements. Be sure to verify and remove any incorrect networks. There are numerous ways to verify that the routing protocol is functioning properly. Besides looking at your running configuration, you can use **show ip route**. This command displays the router's routing table. Read the legend to see what routes have been learned. ***Networks that are *directly connected* will not be learned through a routing protocol.

TCL Scripts / Lab Verification

At the end of each lab, you will need to verify that everything is working. This is called the "Golden Moment", the point when you have universal connectivity. Every IP address should be able to ping all other IP addresses. It is not practical to do this by hand as longer Labs will be unmanageable. We have a handy tool called TCL (pronounced tickle) scripts.

The basic syntax for this command is:

```
tclsh
foreach address {
    10.1.13.1
    10.1.13.3
    10.1.235.3
    10.1.235.2
    10.1.235.5
    10.1.245.5
    10.1.245.2
    172.16.26.6
```

```
    10.1.245.4
    172.16.26.2
    192.168.45.5
    192.168.45.4
    10.1.46.4
    10.1.46.6} {
    ping $address
    }
  tclquit
```

It is recommended that you copy this template into notepad. Cisco IOS is finicky with the formatting and you may end up telnetting into each address if you mess up the formatting. You can use **show ip alias** to help you get the appropriate addresses.

For any ping that fails, go back and recheck the previous sections of the lab. Do the best you can to troubleshoot. If you cannot figure it out, call over a TA to help you.