IPv6 and OSPF Networks Lab

By: Jeremy Watson

Senior Project Advisor: Hugh Smith
Computer Engineering
California Polytechnic State University, San Luis Obispo
Spring 2016

I. Acronyms

Open Shortest Path First - OSPF
Internet Protocol - IP
Cisco Certified Network Associate - CCNA
Cisco Certified Internet Expert - CCIE
Internet Protocol Version 6 - IPv6
Area Border Router - ABR
Autonomous System - AS
Multiprotocol Label Switching - MPLS

II. Introduction

Purpose

To create an OSPF lab assignment for the Introduction to Computer Networks class and append it to the existing IPv6 lab assignment.

Background

The goal of the computer networks class is to give students a general understanding of how the Internet works. At the end of the class, students should be able to answer the following question in a great amount of detail: How is it that when I type *www.google.com* into my web browser, I get a web page? One of the best things about computers and the internet is that they are, for the most part, easy to use and are getting even easier to use. And with the development of mobile phones, tablets, wearables, etc., the internet is becoming more accessible to everyone. And with the size of the internet increasing, speed and efficiency are key metrics to learn and pay attention to. Which is where OSPF comes in.

The goal of this senior project is to teach students about the OSPF protocol, which is a routing protocol used to find the shortest route between two points in an IP network. Currently, the CPE 464 lab teaches students how to configure network equipment to learn about the network(s) that it is currently connected to, but it does not teach students how to configure OSPF or any other kind of routing optimization. OSPF is commonly used by large enterprise networks and is a key topic when studying for network engineer certifications like the CCNA and CCIE. So this is still a relevant protocol to learn because it is commonly used and asked about.

The OSPF assignment is currently being appended to the IPv6 lab because it is the shortest of the current lab assignments. The original goal was to write the OSPF lab so that it used IPv6, but due to certain roadblocks this was not implemented and will be explained further in a later section.

Slight edits were made to the original IPv6 assignment. This was done because the original wording of the network setup was misleading, which resulted in none of the students or assistants being able to connect to the internet using IPv6.

III. Technical Details

General Overview

OSPF is a link-state routing protocol which is meant to run inside of a single AS. This protocol actually serves a few different purposes inside of an AS. The first, is that it actually learns the topology of the network. The second, is that OSPF calculates the shortest path for a packet as it receives them. The third, is that OSPF learns the AS as changes happen to it. For example, if new routes are introduced or if interfaces fail, then OSPF will learn about these changes and then adjust the routing table as appropriate. Together these three pieces of functionality are what make OSPF a reliable and efficient routing protocol. It not only learns the network once, it continually learns it over and over again when changes occur and recalculates routes dynamically.

How OSPF Works

OSPF works by learning the network topology and then calculating routes as it learns. When the process first starts up, it will attempt to form adjacencies with its neighbor routers. For example, **Figure 1** shows an OSPF area with several routers inside of it. Router A is neighbors with Routers B and C. When Router A is brought up and the OSPF process is configured, it will note who its neighbors are, and then form adjacencies with them. When two routers form an adjacency they share/advertise their link-state database with each other. Once the router has established its adjacencies, the router will then advertise its own link-state database with them and vice-versa. This happens when a timer expires and can be configured differently depending on the needs of the AS. As this propagates, the router will then learn about what networks its neighbors are connected to, and what its neighbors neighbors are connected to, and so on. This gives the router a complete view of the OSPF area that it is connected to. Something to note, is that a router will only learn about its own OSPF area. It will not learn about neighboring areas, this prevents the topology database from becoming overwhelmingly large. Once the link-state database is formed, the router then creates a shortest path tree out of the graph of routers in its database. After the tree is created, the OSPF process then updates the router’s routing table and traffic is able to be forwarded to the appropriate places. If the OSPF process discovers that the tree has multiple routes to the same network, it will pick the lowest-cost path and only update the routing table with that path. And if there are multiple paths with the same cost, it will update the table with all of them.

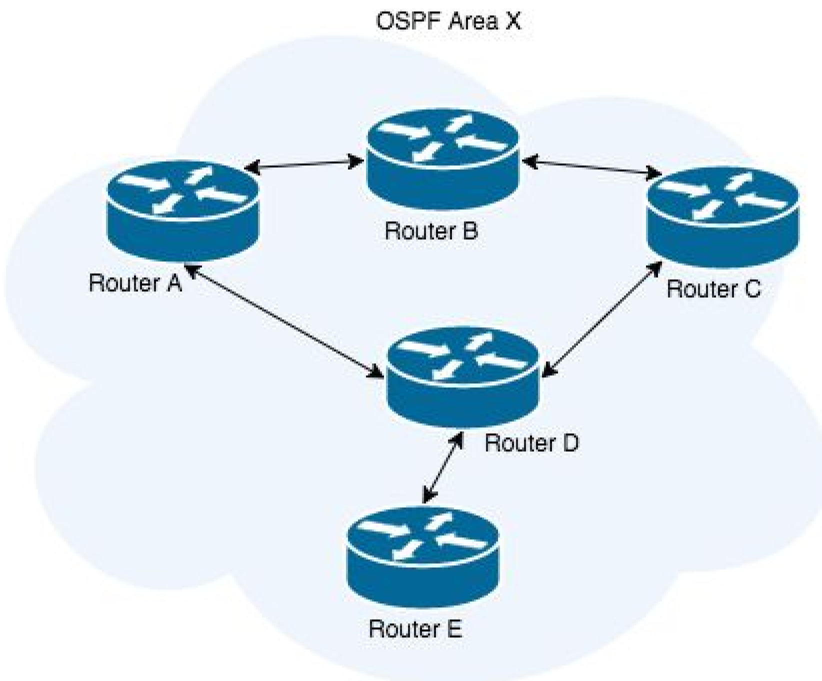


Figure 1: OSPF Neighbor Routers

OSPF Areas

One of the benefits of OSPF is that it is very scalable. This becomes apparent when OSPF Areas are introduced. As mentioned before, the link-state database is shared between routers based on a timer. When the timer expires, the database is advertised to the router's adjacencies. If the database is massive and the timer goes off frequently, the database becomes cumbersome to the network as a whole because the OSPF process begins to take up a lot of the network's bandwidth. So in order to fix that, we use OSPF areas. OSPF areas are groupings of networks, routers, and links that have the same OSPF identification number. The OSPF process running on the routers will only learn about its own OSPF area, thereby reducing the size of each router's link-state database.

Inter-area Routing

Since each OSPF area only learns about the networks and routers inside of its own area, how does communication between areas occur? There needs to be some sort of mechanism which shares information between areas without inflating the link-state database too much. That is where the backbone area comes in. Area 0 is the OSPF area which all traffic passes through in order to reach other areas. It contains route summaries from all areas and

can forward traffic to the appropriate ABRs as necessary. ABRs are routers that, by definition, are connected to the backbone area and another area. They are the routers which handle routing between areas. **Figure 2** shows an example of inter-area routing. So how routing works from one area to another is like this: Each router in an OSPF area learns about its own area. So according to the figure below, Area X will only learn about Area X, Area Y will only learn about Area Y. In each area, there is at least one ABR which, by definition, is connected to area 0. For example, the ABR in Area X will summarize all of the networks inside of Area X and then advertise them all to its neighbor in Area 0. The same thing occurs in Area Y. Now, once traffic needs to go from Area X to Area Y, the routers in Area X will know to forward the traffic to their ABR. The ABR will send the traffic to the backbone area, which will then send it to the ABR of Area Y. Once in Area Y, the traffic will be forwarded to its appropriate place.

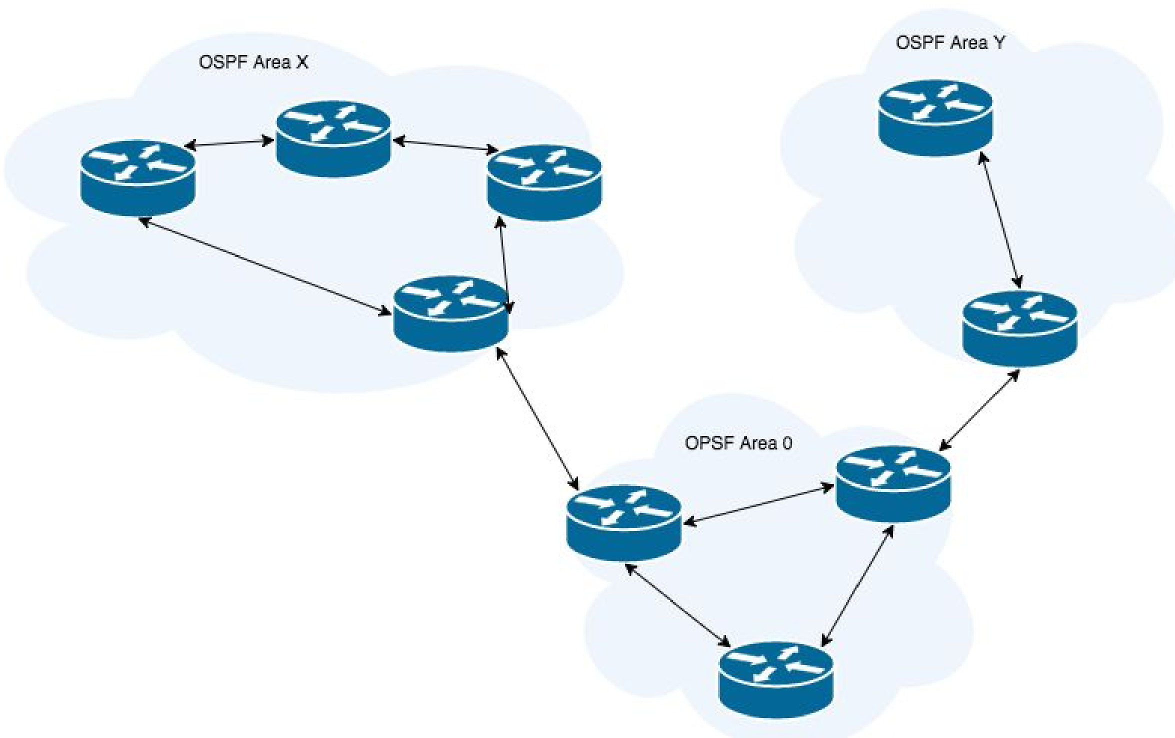


Figure 2: OSPF Inter-Area Routing Example

IV. Lab Assignment Content

This section will describe the lab assignment in Appendix B, and the different topics that are covered in it.

Overview

At the beginning of the OSPF section of the lab, each group will get into a team of four. Each team of four will use the same network configurations in order to keep things simple. In each team, they will pair off into two groups of two. In the first part of the lab, they will learn about how to configure the loopback address, how to configure the OSPF process, and equal-cost load balancing. Afterwards, they will re-join with the entire team and learn about the backbone area and inter-area routing.

Loopback Addresses

In order to properly configure OSPF (and to make things a little easier) a loopback interface is required. A loopback interface is a virtual interface. It is not tied to any specific piece of hardware on a router and is always up if the router is running. In industry, the physical interfaces' IP addresses can change depending on what network it is connected to; there is no reason for the loopback address to change, so it is normally used to identify routers. When the user runs the command below, it will display its neighbors as their respective loopback addresses. This will make it easier to distinguish who owns which router.

OSPF Frame Types

OSPF has five different types of frames that it uses to establish itself and to maintain the link-state database as changes occur. The students will setup a port monitoring session and will start a Wireshark capture in order to observe the different frames. This will happen after configuring the OSPF process and establishing an adjacency with their partner workgroup. After monitoring for a few minutes, they should be able to see all of the different packet types. Below is the header structure for an OSPF packet. Each type of OSPF packet will start with this header below in **Figure 3.** In the figures following, you will notice that the "Type" field is replaced with a number. This number represents the the OSPF packet type, which is unique for each of the OSPF frames. The "Router ID" and "Area ID" fields are used to determine which router the packet is coming from, and also which field it is from. This is used to keep areas separate from each other, so if a router gets a packet from Area 10, but is actually located in Area 20, the router would ignore the packet.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Version #   |     Type      |         Packet length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Router ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Area ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Checksum            |             AuType            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Authentication                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Authentication                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: OSPF Packet Header [1]

The first type of frame is called an Hello Packet. These packets are used to start and maintain neighbor relationships. When the OSPF process first starts, establishing who the host router's neighbors are is the first step in initializing the process. The Hello Packets are sent periodically based on a timer to maintain these neighbor relationships in case new neighbors are connected or interfaces fail. **Figure 4** shows how the full OSPF Hello Packet is structured.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Version #   |       1       |         Packet length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Router ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Area ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Checksum            |             AuType            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Authentication                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Authentication                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Network Mask                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         HelloInterval         |    Options    |    Rtr Pri    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     RouterDeadInterval                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Designated Router                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Backup Designated Router                    |
```

Figure 4: OSPF Hello Packet [1]

The second is called a Database Description Packet. This frame is used when adjacencies are first being established. This frame gives information about the topological database and how the network is setup. Something to keep in mind, is that when a router sends out this packet, it is communicating what its current topological database looks like. So if a router is just being brought up, or if networks are being joined, this can sometimes take several packet exchanges to do before it is complete. The function of this packet is part of what makes OSPF scalable. **Figure 5** shows how the packet is structured and what data is contained inside of it.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Version #   |       2       |         Packet length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Router ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Area ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Checksum            |             AuType            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Authentication                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Authentication                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Interface MTU         |    Options    |0|0|0|0|0|I|M|MS
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     DD sequence number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-                                                             -+
|                                                               |
+-                      An LSA Header                          -+
|                                                               |
+-                                                             -+
|                                                               |
+-                                                             -+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              ...                              |
```

Figure 5: OSPF Database Description Packet [1]

The next is called the Link State Request Packet. This packet type is used in the final stage of adjacency initialization. After a router has received its neighbors' topological database, it might find that parts of its own database are out of date. The router will then use this packet

type to request pieces of its neighbors database that are more up to date. **Figure 6** shows the structure of the packet.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Version #   |       3       |         Packet length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Router ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Area ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Checksum            |             AuType            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Authentication                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Authentication                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          LS type                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Link State ID                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Advertising Router                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              ...                              |
```

Figure 6: OSPF Link State Request Packet [1]

The fourth type is the Link State Update Packet, which is used for link state advertisement flooding. These advertisements are flooded throughout the OSPF area and this is how routes are shared between routers. **Figure 7** shows how the packet is structured.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Version #   |       4       |         Packet length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Router ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Area ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Checksum            |             AuType            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Authentication                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Authentication                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            # LSAs                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-                                                            +-+
|                             LSAs                              |
+-                                                            +-+
|                              ...                              |
```

Figure 7: OSPF Link State Update Packet [1]

The final type is the Link State Acknowledgement Packet. This packet is simply sent in response to the Link State Update Packet to confirm that it has been received. This helps maintain synchronization between routers. **Figure 8** shows the structure of the packet.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Version #   |       5       |         Packet length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Router ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Area ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Checksum            |             AuType            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Authentication                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Authentication                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-                                                             -+
|                                                               |
+-                         An LSA Header                       -+
|                                                               |
+-                                                             -+
```

Figure 8: OSPF Link Stack Acknowledgement Packet [1]

Equal Cost Load Balancing

Another benefit of OSPF is that it provides functionality for load balancing. This event occurs when there are multiple routes discovered to the same destination network with the same route cost. When this happens, the sending router will distribute the traffic to the destination network among the various routes. This avoids congesting one route or link to the point where other traffic is unable to use that route or link. The cost of a route is determined by the amount of bandwidth used on a particular interface.

The students will perform this in the lab as well. They will establish 3 equal cost paths between themselves and their partner workgroup. Next, they will run a traceroute command which will be load balanced over all of their routes. After they do this, they will then configure what is called the reference bandwidth and discover that not all of their routes are actually equal. This is because one of their three established routes will be using a 100Mb/s port while the remaining routes are actually 1Gb/s ports. **Figure 9** below shows how students will be setting up their network with their partner workgroup to investigate equal cost load balancing.

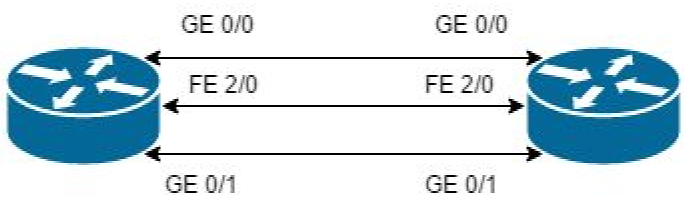


Figure 9: OSPF Equal Cost Load Balancing Setup

Areas and Inter-Area Communication

Students will explore this area of OSPF at the end of the lab assignment. They will come together with their other team and then use the backbone area in order to share routes between areas. Once the backbone area is configured, the lab has them ping every interface on the network and then get a student assistant to sign them off. They will then examine their routing table and try to understand the special route label that is used to get to the other OSPF area. **Figure 10** below shows the setup that the students will be using when setting up this portion of their lab assignment.

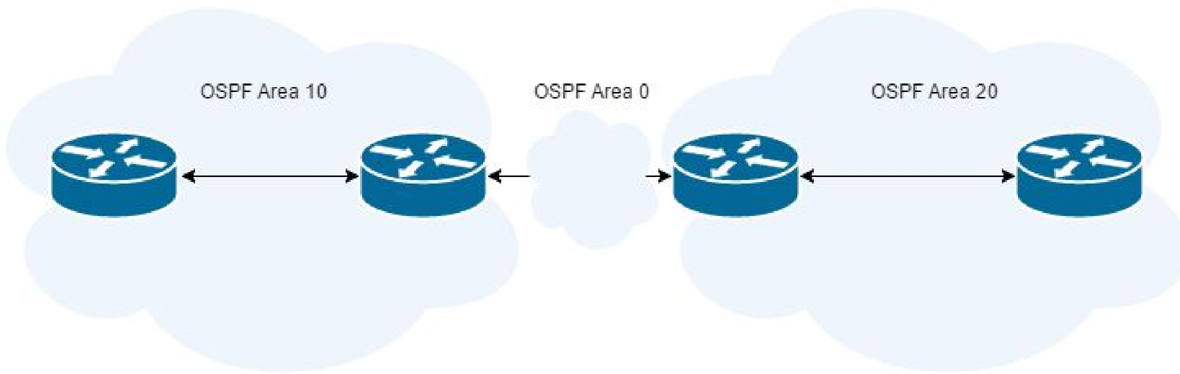


Figure 10: OSPF Inter-Area Routing

V. Issues/Roadblocks

Like every project, there were definitely issues and roadblocks that were hit along the way. Unfortunately, one of them could not be overcome due to scheduling conflicts. At the beginning of this project, the original goal was to create an OSPF and MPLS extension to the IPv6 lab. MPLS is a traffic engineering protocol which allows routers to determine an entire route for a data packet at once rather than recalculate it at each router. Pairing these two protocols together makes sense because MPLS requires a peer-to-peer protocol to run on top of; OSPF is what is commonly used in industry. However, after digging through Cisco's documentation and tutorials, the resulting traffic engineered tunnel was not working. It seems to be an issue with the link-state database not recognizing certain links as being up, so when these links are used in the traffic engineered path, the OSPF process determines that this tunnel is not valid and defaults to the routing table. After much troubleshooting, I tried to set-up a meeting with a Cisco network engineer. However, due to scheduling conflicts, this was not able to happen.

VI. Future Work

As I learned more about the technology, I realized that there were more topics about OSPF that I could add to the assignment, but simply did not have the time for. Below are a few ideas for extensions to the lab assignment and restructures to the existing labs.

Lab Extensions

As discussed in prior sections and as implemented in the lab assignment, OSPF areas must be directly connected to the backbone area in order to learn about the rest of the AS.

However, there are times where it is not possible to directly connect an area to the backbone. This is where virtual links are used. I was not able to do much research on this topic, but it could be beneficial for students to learn about.

This was mentioned earlier, because the software on our current routers does not support IPv6 with MPLS, I was made to shift my assignment into an IPv4 lab. However, our routers do support OSPF with IPv6. So this current lab could be edited to use IPv6 instead, this would allow the lab to seem more unified with the first part of the lab.

One of the things that is inefficient about routing is that each packet’s next hop is calculated at the router that it is currently at. This involves a lengthy lookup process at each router. Multiprotocol Label Switching is a protocol which allows traffic engineers to setup traffic-engineered paths through an OSPF area. This was the other part of the lab assignment that I could not get working unfortunately.

Assignment Restructuring

In a previous lab, students use the RIP protocol to have their routers learn the networks that they are connected to. Both RIP and OSPF have to do with network learning, so it could make sense to join them into one lab assignment. The assignment could point out the differences between the two and how in certain situations one is preferred over the other. This could result in having another lab assignment just for OSPF and MPLS.

VII. Conclusion

Overall, this was an enjoyable project to work on. The central challenge of this project was learning about networking protocols, and then learning them both well enough to teach other students. Something that students do not consider is how time consuming and difficult it can be to create assignments for students. Especially when they must convey complicated topics simply and concisely. Unfortunately, not everything could be implemented in time, but overall the assignment is interesting and should teach the students a lot about basic OSPF configurations. I’d like to personally thank Dr. Bellardo for helping me troubleshoot the issues with the original IPv6 lab. And I’d like to give a special thank you to Dr. Smith for being incredibly patient and understanding with me throughout my project. Educators like him are why I have an interest in teaching in the first place.

VIII. Appendix

A.Fact Sources

[1] Internet Engineering Task Force, “OSPF Version 2”, www.ietf.org. [Online]
Available: https://www.ietf.org/rfc/rfc2328.txt.

[2] Cisco Systems, Inc. “What Are OSPF Areas and Virtual Links?”, www.cisco.com. [Online]
Available:
http://www.cisco.com/c/en/us/support/docs/ip/open-shortest-path-first-ospf/13703-8.html.

[3] Cisco Systems, Inc. “OSPF Virtual Link”, www.cisco.com. [Online]
Available:
http://www.cisco.com/c/en/us/support/docs/ip/open-shortest-path-first-ospf/47866-ospfdb7.html

[4] Microsoft, “OSPF Areas”, technet.microsoft.com. [Online]
Available: https://technet.microsoft.com/en-us/library/cc957882.aspx

B. Video Sources

[5] networkingstepbystep, “OSPF Video Cheat Sheet: Enable OSPF on an IP Network”, YouTube.com, June 5, 2010
Available: https://www.youtube.com/watch?v=jbntiEAF6MQ.

[6] Barker, Keith, “OSPF Load Balancing and Tweaking”, YouTube.com, June 16, 2011.
Available: https://www.youtube.com/watch?v=97Ukh4cPur8

[7] networkingstepbystep, “MPLS Video Cheat Sheet: MPLS Traffic Engineering - Preparing the Network”, YouTube.com, June 13, 2010.
Available: https://www.youtube.com/watch?v=8FSgtEkoZ0g

[8] networkingstepbystep, “MPLS Video Cheat Sheet: MPLS Traffic Engineering - MPLS Traffic Engineering Tunnel”, YouTube.com, June 13, 2010.
https://www.youtube.com/watch?v=cQ9Qe1dJFI4

C.Lab Assignment

Name: ____ Lab Section: 9am Noon 3pm

CPE 464 – Computer Networks
Lab 8 - IPv6 and OSPF

A. Overview

This is a two-part lab. In the first part of the lab, you will explore the current state of IPv6 use. To do this you will create an IPv4 only network for the left PC, and an IPv6 only network for the right PC. You will use the lab RIP network to get both IPv4 and IPv6 service to your router. For this lab your left PC should be using Windows and your right PC should be using Linux. Remember to refer to the separate "Configuring IP Addresses" document for IPv6 specific versions of the IPv4 commands we have been using in lab.

In the second part, you will play around with OSPF using IPv4 and learn about some of the metrics it uses to calculate the cost of a certain path. You will also learn about OSPF areas and how to connect areas together.

Stage I

B. Network Wiring Setup

Connect your bench as follows:

- Connect the FastEthernet 1/0 interface to port 11 on your patch panel.
- Connect port 11 on your patch panel to the backbone switch.
  - Ask your TA which ports you are allowed to connect to.
- Connect the left PC on the bench to the GigabitEthernet 0/0 interface on the router.
- **<u>DO NOT</u>** connect the right PC on the bench until instructed to do so (it will eventually be connected to the GigabitEthernet 0/1 interface on the router, so get the needed cable now).

C. Configure IPv4 service to the left PC

Use the table at the end of this lab to determine your bench's assigned RIP vlan number and IP addresses. These numbers can also be found in the RIP lab and on the desktop background in Windows. You will need to use one of your bench IPv4 subnets for the connection between the router and the left PC. The following commands configure the router:

```
RouterX# configure term
RouterX(config)# interface FastEthernet 1/0
RouterX(config-if)# no shut
RouterX(config-if)# exit
```

```
RouterX(config)# interface FastEthernet 1/0.<vlan-id>
RouterX(config-subif)# encapsulation dot1q <vlan-id>
RouterX(config-subif)# ip address <RIP-IP-address> <RIP-subnet-mask>
RouterX(config-subif)# exit

RouterX(config)# router rip
RouterX(config-router)# version 2
RouterX(config-router)# redistribute connected
RouterX(config-router)# network 208.94.62.0
RouterX(config-router)# no auto-summary
RouterX(config-router)# exit

RouterX(config)# ip route 0.0.0.0 0.0.0.0 <Backbone-RIP-IP-address>

RouterX(config)# interface GigabitEthernet 0/0
RouterX(config-if)# no shut
RouterX(config-if)# ip address <Bench-Subnet-A-address> <Bench-Subnet-netmask>
RouterX(config-if)# end
```

Configure your left PC to join subnet A. Ensure the left PC can get online before continuing.

D. Configure IPv6 service to the Router

First, a note on IPv6 address syntax. The 16 byte address are written as eight two byte groups separated by colons:

2620:0000:0dd0:2000:0000:0000:0000:0002

This is long, cumbersome, and can lead to many typos. Hence a shorthand is used. Leading zeros within a single group can be omitted:

2620:0:dd0:2000:0:0:0:2 == 2620:0000:0dd0:2000:0000:0000:0000:0002

Consecutive groups of zero can be compressed using a double colon (::):

2620:0:dd0:2000::2 == 2620:0:dd0:2000:0:0:0:2

Only one such group of zeros can be compressed with ::, otherwise it would be ambiguous. For instance, **2620::dd0:2000::2** is an invalid IPv6 address.

The first step is to configure IPv6 service with the backbone router via the RIP vlan. The commands to enable this are as follows:

```
RouterX# configure term
RouterX(config)# ipv6 unicast-routing

RouterX(config)# interface fastEthernet 1/0.<vlan-id>
RouterX(config-subif)# ipv6 enable
RouterX(config-subif)# ipv6 address <RIP-IPv6-address>/64
RouterX(config-subif)# ipv6 rip v6rip enable
RouterX(config-subif)# exit

RouterX(config)# ipv6 router rip v6rip
RouterX(config-rtr)# redistribute connected
```

```
RouterX(config-rtr)# exit
RouterX(config)# ipv6 route 0::0/0 <Backbone-RIP-IPv6-address>
RouterX(config)# end
```

Notice the similarities between IPv4 and IPv6 configuration. Everything we have worked with thus far for v4 applies to v6, although the command syntax and address format is slightly different.

At this point you should have IPv6 service to your router. Verify connectivity by running the following two commands:

```
RouterX# ping <Backbone-RIP-IPv6-address>
RouterX# ping 2620:0:dd0:2000::2
```

If either ping fails, stop and fix the problem now. Do not continue until it is resolved. The first ping ensures your direct connection to the lab router is working. If the first ping fails, double-check the IPv6 address you have assigned to your network interface. The second ping tests multi-hop connectivity to a server in the front of the lab. If the second ping fails check that ripV6 is setup correctly.

E. Configure IPv6 service to GigabitEthernet 0/1

Pick a /64 subnet to use from your bench's allocated /56 subnet. By convention the router is configured with address ::1 on the subnet. To configure GigabitEthernet 0/1 with the IPv6 address, use the following commands:

```
RouterX# configure term
RouterX(config)# interface GigabitEthernet 0/1
RouterX(config-if)# no shut
RouterX(config-if)# ipv6 enable
RouterX(config-if)# ipv6 address <Router-IPv6-subnet>::1/64
RouterX(config-if)# end
```

F. Configure the right PC

Boot the right PC into Linux by selecting "Ubuntu 14.04, kernel *-generic" at the boot prompt. The username and password match the Windows username and password. See the end of lab for a summary of Linux commands that might be useful when troubleshooting network problems. After you log in, open up a terminal and get a root shell by running the command:

```
$ sudo su
```

Enter the password when prompted. Because the network isn't configured properly yet, it might take 1-2 minutes to see the password prompt. Don't close the root shell. It will take another 1-2 minutes to reopen. If you need to use a serial program in Linux to configure the router, it is "minicom".

Capture the initial IPv6 network traffic using Wireshark. To get the initial traffic, leave your network cable to the right PC unplugged during the next steps. Ensure the Ethernet card by running the following command in the root shell:

```
# ifconfig eth0 up
```

Start Wireshark from the root shell by running the following command:

```
# wireshark &
```

Start a capture, with no filters, on eth0. Once the capture is running attach the PC to the router. At this point IPv6 should auto-configure the host's IPv6 and default gateway address. Verify you can ping the router on your bench:

```
$ ping6 <Router-IPv6-subnet>::1
```

If you can't ping, fix the problem before continuing. If the ping was unsuccessful you need to restart the Wireshark capture after the problem is resolved.

Stop and analyze the Wireshark capture. What is similar and what is different than a ping in IPv4?

How does ICMP differ?

How does ARP differ?

Look at the router advertisement packets. What do you think they are used for (remember the left PC can still reach Google)?

Verify you can ping the backbone router. This tests multi-hop IPv6 routing:

```
$ ping6 <Backbone-RIP-IPv6-address>
```

If you can't ping, stop and fix the problem before continuing.

Verify you can ping the lab's 2 IPv6 DNS servers:

```
$ ping6 2620:0:dd0:2000::2
$ ping6 2620:0:dd0:2001::2
```

If you can't ping, stop and fix the problem before continuing.

Configure DNS so you can resolve host names. Edit the file "/etc/resolv.conf" from your root shell:

```
# vi /etc/resolv.conf
```

Add the following 2 lines to the file, save, and exit vim:

```
nameserver  2620:0:dd0:2000::2
nameserver  2620:0:dd0:2001::2
```

Verify the name servers are working correctly by resolving google.com:

```
$ dig www.google.com
```

If you can't resolve Google, stop and fix the problem before continuing.

Open Firefox and browse to http://www.kame.net/ on both the left and right PCs. What is the difference between the sites? Why?

Have a TA initial here before continuing (show them kame.net works as expected).

____

G. Compare the IPv4 Internet with the IPv6 Internet

Spend a few minutes browsing the IPv4 and IPv6 Internets. At a minimum, visit google.com and microsoft.com on both PCs. Summarize your experience in the space below.

H. Add IPv4 to the right PC

Lets make the right PC a first-class Internet citizen and give it an IPv4 address. You need to configure GigabitEthernet 0/1 with an IPv4 address from your bench's B subnet:

```
RouterX# configure term
RouterX(config)# interface GigabitEthernet 0/1
RouterX(config-if)# ip address <router-subnet-B-address> <subnet-B-netmask>
RouterX(config-if)# end
```

Once your router has an IP address, add one to the Linux PC by running the following commands in the root shell:

```
# ifconfig eth0 up <host-subnet-B-address> netmask <subnet-B-netmask>
# ip route add default via <router-subnet-B-address>
```

Verify IPv4 connectivity by pinging the router address and the lab IPv4 DNS servers:

```
$ ping <router-subnet-B-address>
$ ping 208.94.63.1
$ ping 208.94.63.9
```

If you can't ping, stop and fix the problem before continuing.

Add the lab's two IPv4 DNS servers into the /etc/resolv.conf file. Edit the file in your root shell:

```
# vi /etc/resolv.conf
```

Add the following 2 lines to the file (without removing the IPv6 lines), save, and exit vim:

```
nameserver  208.94.63.1
nameserver  208.94.63.9
```

Verify the name servers are working correctly by resolving www.microsoft.com:

```
$ dig www.microsoft.com
```

Re-browse the web sites visited in part F. At a minimum, visit google.com and microsoft.com on both PCs. How did the Internet experience change in Linux running a dual-stack Internet?

H. Clean up

You need to clean up your Linux DNS settings so you don't leave the machine in a bad state for other students. You do this by deleting the /etc/resolv.conf file. In your root shell run:

```
# rm /etc/resolv.conf
```

After that, power-cycle your workbench. This will remove all prior configurations so that you can start fresh for the second part of the lab.

Stage II

A. Group Setup

First, get into a group of four workbenches. Among the four of you, fill out the table at the end of this lab titled: "OSPF Network Setup". Out of your group of four, you will pick an initial partner workgroup. Make sure that you and your partner workgroup are on the same team specified on the table. Use those networks and IP addresses assigned.

Local Network

You will need
- 4x short straight-through cables
- 2x long straight-through cables
- 1x crossover cable

Connect your bench as follows:
- Connect the left and right PCs to your switch
- Connect FastEthernet 1/0 and GigabitEthernet 0/0 interface on your router to your switch
- Configure your equipment with the IP info on the table
- **Figure 1** below shows the configuration

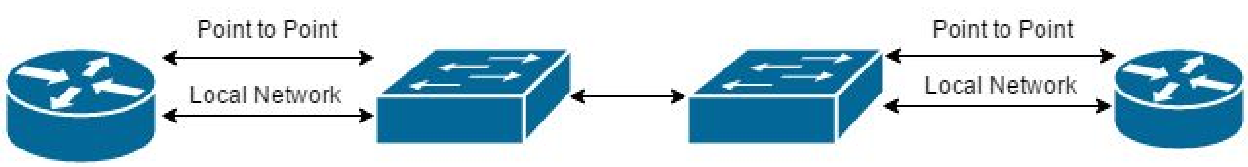


Figure 1: Basic Network Setup

Loopback Address

You will now configure your loopback address. A loopback interface is a virtual interface. It is not tied to any specific piece of hardware on your router and is always up if your router is running. In industry, the physical interfaces' IP addresses can change depending on what network it is connected to; there is no reason for the loopback address to change, so it is normally used to identify routers. See the table at the end of this lab to see what your loopback address will be.

```
RouterX# configure terminal
RouterX(config)# interface loopback 0
RouterX(config-if)# ip address <loopback address>
RouterX(config-if)# end
```

B. Port Sniffing OSPF Packets

Now we will be sniffing OSPF packets through the switch. From now on, all pinging will be done by the **Right PC** and all of the sniffing will be done by the **Left PC**.

Follow the instructions below to turn on port monitoring:

```
SwitchX# configure terminal
SwitchX(config)# monitor session 1 dest int gi <Left PC Port>
SwitchX(config)# monitor session 1 source int gi <Right PC Port>
SwitchX(config)# end
SwitchX# show monitor session 1
```

You should now be able to start a capture. Start it now.

C. Configuring the OSPF Process

Normally when we configure the router, we are configuring the interfaces on the router. This time, we are going to configure the OSPF process that will run in the background and will learn the network. Let's get started!

Now pick a process ID and then write it in the space below, we will need this for future reference.

OSPF Process ID: ______

Now follow these steps:

```
RouterX# configure terminal
RouterX(config)# router ospf <PID>
RouterX(config-router)# router-id <Loopback 0 address>
RouterX(config-router)# network <Group Network> 0.0.255.255 area <Area>
RouterX(config-router)# end
RouterX# show ip ospf neighbor
```

At this point, you should be able to ping every interface in the network.

Use a filter on the capture and see how many different types of frames you see here. Give it about 3 minutes.

How often are the OSPF Hello Packets arriving?

Now remove the point-to-point connection from your router. How long does it take for the OSPF process to show that the neighbor is down? (Keep running **show ip ospf neighbor**)

D. OSPF Equal Cost Load-Balancing and Metrics

Now connect your FastEthernet 2/0 and GigabitEthernet 0/1 interfaces to your switch, use the IP config info in the table mentioned before. Make sure that you can ping all interfaces in your network. **Figure 2** below shows the configuration that you should have setup. It omits the switches; this is because it is no longer necessary to have every link go through the switch. You can connect directly router to (patch panel to) router.

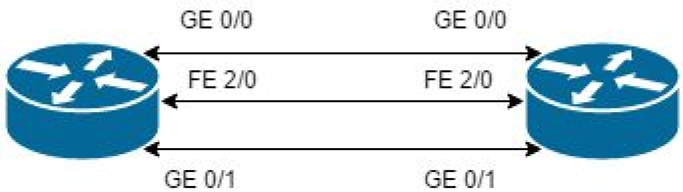


Figure 2: Equal Cost Load Balancing Setup

Now that we have 3 different routes to our neighbor, let's inspect it a little bit.

`RouterX# show ip ospf interface brief`

How many entries are there here?

What is the cost of these links? Does this seem correct?

If you spotted something wrong, good; don't fix it yet! If not, don't worry about it too much, we'll come back and fix it in a few minutes.

One of the great things about OSPF, is that it will give you an estimated cost of each link and when sending traffic, it will usually pick the least cost path each time. But what happens when there are several links with the same cost? Let's find out.

`RouterX# traceroute` *`<neighbor's IP>`* **`probe 4`**

What is happening here? What are the benefits of this?

Now, back to what I mentioned about something being wrong.

`RouterX# show ip ospf interface brief`

The cost of each link is 1, why should this not be so?

Let’s fix it!

```
RouterX# configure terminal
RouterX(config)# router ospf <PID>
RouterX(config-router)# auto-cost reference-bandwidth 1000
RouterX(config-router)# end
RouterX# show ip ospf interface brief
```

Which interfaces have a different cost now? Why does this make sense?

Why did we use 1000 as the reference-bandwidth? Should this always be the case?

Now before proceeding, do you think this changed the routing table at all? Why?

```
RouterX# show ip route ospf
```

What has changed? Was this what you expected?

What does this imply about how the OSPF process works with the routing table?

What would happen if OSPF put all of the routes it knew about inside of the table?

E. Inter-Area Communication

In this section, you will form a group of four and learn about how to communicate between OSPF areas. Before beginning, make sure that your partner workgroup is at this point too! **Figure 3** shows a diagram of what you will be setting up with your partner team. They have been working in one OSPF area and you have been working in a separate one. Now, we will connect the two together!

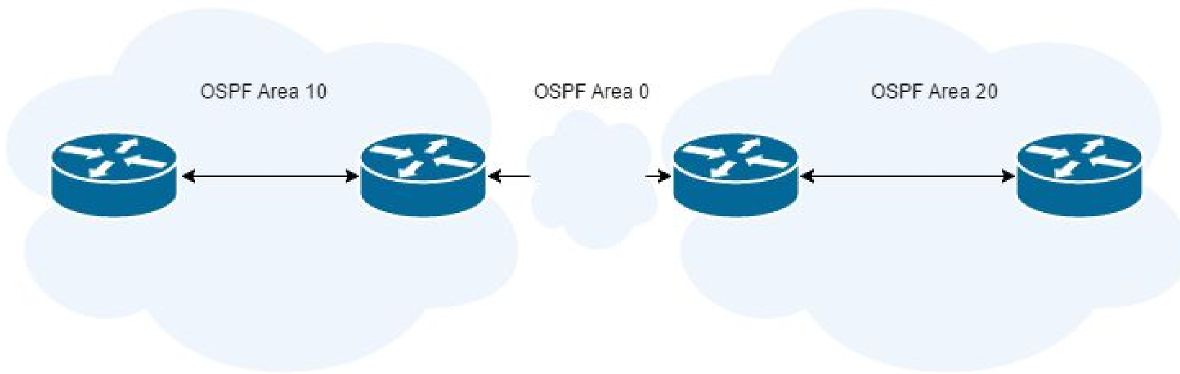


Figure 3: Inter-Area Routing Setup

Now get into your group of 4 work benches and connect each of your networks together using the network diagram at the end of this lab.

IMPORTANT: Notice that the diagram is slightly different, only one workbench from each group will connect to each other.

TIP: Have all four groups fill out the diagram the same way. For example, if you are groups A, B, C, and D, then on everyone's diagrams Group A will be in the same place, same for Group B, Group C, and Group D.

Now configure a single point-to-point connection between the two pairs of groups.

ALRIGHT, WE'RE ALMOST THERE.

From your PC, try to ping someone from the other OSPF area.
Why does this fail?

Okay, so we've got two OSPF areas that we need to connect. The whole point of OSPF areas is to keep parts of the network separate from each other, but what about when we want inter-area communication to happen? We could use a gateway of last resort or a static route, but that's not really scalable when we start adding more and more OSPF areas. This is where the backbone area comes in.

For the two groups bridging the two teams together, follow the steps below. Everyone else, just follow along.

```
RouterX# configure terminal
RouterX(config)# router ospf <PID>
RouterX(config-router)# network <PointToPoint Network> 0.0.0.3 area 0
RouterX(config-router)# end
RouterX# show ip route ospf
```

What does the IA tag mean?

How many routes were added to your routing table? Does this make sense?

At this point, all groups should be able to ping each other. Verify this and then get your TA to initial below.

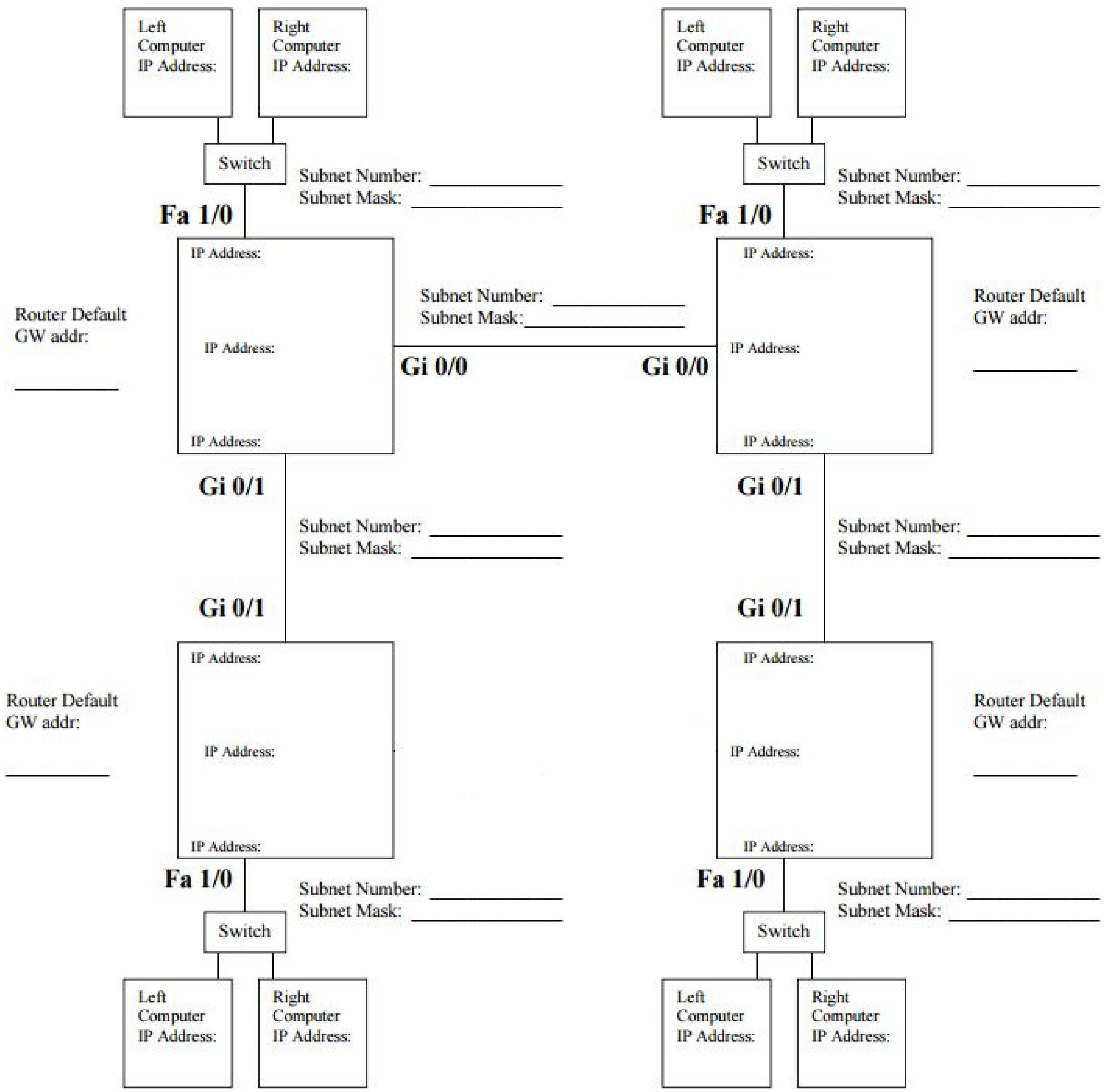
Have a TA initial here before continuing (They will ask you to ping interfaces).

____

F. Workstation Clean-up

You’re done! Shut down your workbench and both of your computers. Return all cables to their appropriate places. Thank you!

Network Diagram



RIP IP Address Subnet Assignments

Workbench	VLAN	Network	FastEthernet 1/0 IP Address	Backbone IP Address
A	800	208.94.62.128/30 2620:0000:0DD0:0120::/64	208.94.62.130 2620:0000:0DD0:0120::2	208.94.62.129 2620:0000:0DD0:0120::1
B	801	208.94.62.132/30 2620:0000:0DD0:0121::/64	208.94.62.134 2620:0000:0DD0:0121::2	208.94.62.133 2620:0000:0DD0:0121::1
C	802	208.94.62.136/30 2620:0000:0DD0:0122::/64	208.94.62.138 2620:0000:0DD0:0122::2	208.94.62.137 2620:0000:0DD0:0122::1
D	803	208.94.62.140/30 2620:0000:0DD0:0123::/64	208.94.62.142 2620:0000:0DD0:0123::2	208.94.62.141 2620:0000:0DD0:0123::1
E	804	208.94.62.144/30 2620:0000:0DD0:0124::/64	208.94.62.146 2620:0000:0DD0:0124::2	208.94.62.145 2620:0000:0DD0:0124::1
F	805	208.94.62.148/30 2620:0000:0DD0:0125::/64	208.94.62.150 2620:0000:0DD0:0125::2	208.94.62.149 2620:0000:0DD0:0125::1
G	806	208.94.62.152/30 2620:0000:0DD0:0126::/64	208.94.62.154 2620:0000:0DD0:0126::2	208.94.62.153 2620:0000:0DD0:0126::1
H	807	208.94.62.156/30 2620:0000:0DD0:0127::/64	208.94.62.158 2620:0000:0DD0:0127::2	208.94.62.157 2620:0000:0DD0:0127::1
I	808	208.94.62.160/30 2620:0000:0DD0:0128::/64	208.94.62.162 2620:0000:0DD0:0128::2	208.94.62.161 2620:0000:0DD0:0128::1
J	809	208.94.62.164/30 2620:0000:0DD0:0129::/64	208.94.62.166 2620:0000:0DD0:0129::2	208.94.62.165 2620:0000:0DD0:0129::1
K	810	208.94.62.168/30 2620:0000:0DD0:012A::/64	208.94.62.170 2620:0000:0DD0:012A::2	208.94.62.169 2620:0000:0DD0:012A::1

L	811	208.94.62.172/30 2620:0000:0DD0:012B::/64	208.94.62.174 2620:0000:0DD0:012B::2	208.94.62.173 2620:0000:0DD0:012B::1
M	812	208.94.62.176/30 2620:0000:0DD0:012C::/64	208.94.62.178 2620:0000:0DD0:012C::2	208.94.62.177 2620:0000:0DD0:012C::1
N	813	208.94.62.180/30 2620:0000:0DD0:012D::/64	208.94.62.182 2620:0000:0DD0:012D::2	208.94.62.181 2620:0000:0DD0:012D::1
O	814	208.94.62.184/30 2620:0000:0DD0:012E::/64	208.94.62.186 2620:0000:0DD0:012E::2	208.94.62.185 2620:0000:0DD0:012E::1
P	815	208.94.62.188/30 2620:0000:0DD0:012F::/64	208.94.62.190 2620:0000:0DD0:012F::2	208.94.62.189 2620:0000:0DD0:012F::1

Table 1: RIP IP Address Subnet Assignments

Bench Subnet Address Assignments

Workbench	IPv4 Subnet A	IPv4 Subnet B	IPv6 Subnet
A	208.94.61.0/29	208.94.61.8/29	2620:0000:0DD0:1000::/56
B	208.94.61.16/29	208.94.61.24/29	2620:0000:0DD0:1100::/56
C	208.94.61.32/29	208.94.61.40/29	2620:0000:0DD0:1200::/56
D	208.94.61.48/29	208.94.61.56/29	2620:0000:0DD0:1300::/56
E	208.94.61.64/29	208.94.61.72/29	2620:0000:0DD0:1400::/56
F	208.94.61.80/29	208.94.61.88/29	2620:0000:0DD0:1500::/56
G	208.94.61.96/29	208.94.61.104/29	2620:0000:0DD0:1600::/56
H	208.94.61.112/29	208.94.61.120/29	2620:0000:0DD0:1700::/56
I	208.94.61.128/29	208.94.61.136/29	2620:0000:0DD0:1800::/56
J	208.94.61.144/29	208.94.61.152/29	2620:0000:0DD0:1900::/56
K	208.94.61.160/29	208.94.61.168/29	2620:0000:0DD0:1A00::/56

L	208.94.61.176/29	208.94.61.184/29	2620:0000:0DD0:1B00::/56
M	208.94.61.192/29	208.94.61.200/29	2620:0000:0DD0:1C00::/56
N	208.94.61.208/29	208.94.61.216/29	2620:0000:0DD0:1D00::/56
O	208.94.61.224/29	208.94.61.232/29	2620:0000:0DD0:1E00::/56
P	208.94.61.240/29	208.94.61.248/29	2620:0000:0DD0:1F00::/56

Table 2: Bench Subnet Address Assignments

OSPF Configuration Table

Workbench Letter				
	Group 1	Group 2	Group 3	Group 4
	Team A		Team B	
OSPF Area	10		20	
Group Network	10.1.0.0/16		10.2.0.0/16	
Subnet Address	10.1.0.0/24	10.1.1.0/24	10.2.0.0/24	10.2.1.0/24
Left PC	10.1.0.1/24	10.1.1.1/24	10.2.0.1/24	10.2.1.1/24
Right PC	10.1.0.2/24	10.1.1.2/24	10.2.0.2/24	10.2.1.2/24
FastEthernet 1/0	10.1.0.3/24	10.1.1.3/24	10.2.0.3/24	10.2.1.3/24
GigabitEthernet 0/0	10.1.2.1/24	10.1.2.2/24	10.2.2.1/24	10.2.2.2/24
FastEthernet 2/0	10.1.3.1/24	10.1.3.2/24	10.2.3.1/24	10.2.3.2/24
GigabitEthernet 0/1	10.1.4.1/24	10.1.4.2/24	10.2.4.1/24	10.2.4.2/24
Inter-Team Connection	10.3.0.1/30		10.3.0.2/30	
Loopback Addresses	172.16.1.1/32	172.16.2.2/32	172.16.3.3/32	172.16.4.4/32

Table 3: Bench Loopback Addresses
Note: All groups of 4 will use the same IP addresses