# IP Routing

In the previous lab, you already setup a network with *static* routing. In this lab, you will learn about two types of dynamic routing: OSPF and BGP.

## Open Shortest Path First Routing

In practice, OSPF is used in larger installations where there are multiple routers used for internal connections. It is very difficult to model this in a few minutes (hours) in a single undergraduate networking lab! We’ll give it a shot:

### Scenario One

Boatsville University maintains three distinct zones in the campus network: administration, academic, and residence halls. Each zone has its own router. There is a main campus router in the computer center, and then a router for each of the individual zones. The campus has three publicly facing subnets for each zone: academics has 202.150.83.0/24, administration has 203.175.52.0/24, and the residence hall has 205.10.75.0/24.

**Task 1 – Setup the routers**

1. Place a “PT-Router” in the activity, change its name to “RMain”
2. Turn off its power, remove the two optical cards, and then drag three “PT-ROUTER-NM-1FFE” module into one of the empty slots. They’ll look almost identical, but you **should** end up with 100FX connections. Your device should like like: 
3. Turn power back on.
4. Place three “4321” routers into the activity, call one “RAdmin”, one “RAcad”, and one “RRes”.
5. In each of these 4321 routers, drag a “GLC-GE-100FX” optical SFP into the slow marked GE0/0  
   It should look like:  
   
6. **Do not use the automatic connection tool, rather use the orange fiber optic connection**, and connect the three zone routers to the main router, make sure you pick the right interfaces or you will not be able to complete this activity:

|  |  |  |  |
| --- | --- | --- | --- |
| Router | Interface | Router | Interface |
| RMain | Fa4/0 | RAcad | Gig0/0/0 |
| RMain | Fa5/0 | RAdmin | Gig0/0/0 |
| RMain | Fa6/0 | RRes | Gig0/0/0 |

**Task Two** **– Configure the Interfaces**

Each link between the routers needs to have a subnet, and Boatsville doesn’t want to waste is precious public IP addresses on a subnet that will use just two addresses. Instead, they use addresses from the 192.168.0.0/16 internal range.

In the previous Lab, in addition to configuring the IP address you also had to configure port descriptions, remote access, and security. We’re not going to do that in this lab to keep the focus on routing. You can do it if you like, but I won’t be checking for it in this lab. You do need to set the router’s name to match RMain, RAcad, RAdmin, or RRes.

|  |  |  |  |
| --- | --- | --- | --- |
| Router – Router | Subnet | RMain | Remote |
| RMain- RAcad | 192.168.16.0/24 | 192.168.16.1 | 192.168.16.2 |
| RMain – Radmin | 192.168.32.0/24 | 192.168.32.1 | 192.168.32.2 |
| RMain – Rres | 192.168.48.0/24 | 192.168.48.1 | 192.168.48.2 |

The 4321 routers use either the built-in RJ45 1000-BaseT network jack OR the 100Mbps Small Form Factor Port module that you added in a previous step. But, since the port can use one or the other, you need to configure which one you want the router to use. On each of the 4321 routers, we need to switch to 100Mbps, which can only be done when the media type is “rj45,” so:

RAcad(config-if)# media-type rj45

RAcad(config-if)# speed 100

RAcad(config-if)# media-type rj45

At this point, you should be able to test connectivity, and see that you can ping the other end of each subnet, but there is no routing happening.

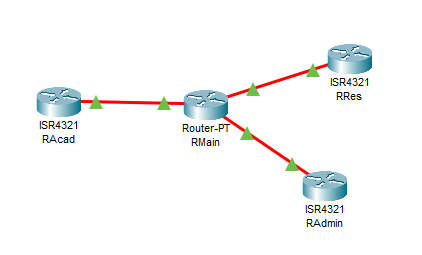


Figure Network with working links.

**Task 3 – Setup OSPF**

In the previous lab, we would now have to configure each of the individual routing tables to allow these subnets to communicate. We don’t need to do that now!

Since the RMain network controls the three routing subnets, RMain sets these:

On the RMain router:

1. Turn on OSPF by picking a *process number* for OSPF – you’ll need to use this same number if you ever want to reconfigure OSPF. Here we use 100, but that’s just a semi-random number:  
   router ospf 100
2. Issue network command to enable sharing routes on each of the three networks:  
   network 192.168.16.0 0.0.0.255 area 0  
   network 192.168.32.0 0.0.0.255 area 0  
   network 192.168.64.0 0.0.0.255 area 0

On each of the remote routers:

1. Turn on OSPF (here’s the process number again – this doesn’t have to match, but its good to be consistent):  
   RAcad(config)# router ospf 100
2. Issue network command to enable the one network (substitute the subnet for appropriate router):  
   RAcad(config-router)# network 192.168.XX.0 0.0.0.255 area 0
3. Finally, we issue a command that will redistribute future connected routes:  
   RAcad(config-router)# redistribute connected subnets

After several seconds of simulation time you should see a message indicating OSPF is done:  
Router#

01:23:44: %OSPF-5-ADJCHG: Process 100, Nbr 223.223.223.233 on FastEthernet0/0/0 from LOADING to FULL, Loading Done

Next, we need to verify that OSPF is working: On each router, run the command “show ip proto” and it should look similar to the following: This output shows that the OSPF database has populated the link state database from all of the routers! So, the next step is to verify that we can ping each of the devices:

Router# **show ip proto**

Routing Protocol is "ospf 100"

Outgoing update filter list for all interfaces is not set

Incoming update filter list for all interfaces is not set

Router ID 223.223.223.222

Number of areas in this router is 1. 1 normal 0 stub 0 nssa

Maximum path: 4

Routing for Networks:

192.168.16.0 0.0.0.255 area 0

Routing Information Sources:

Gateway Distance Last Update

223.223.223.220 110 00:03:11

223.223.223.221 110 00:02:06

223.223.223.222 110 00:00:55

223.223.223.233 110 00:00:55

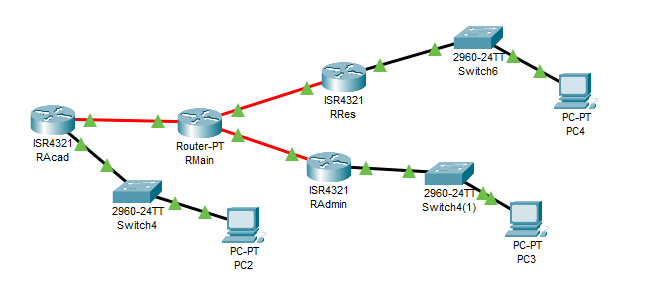
Distance: (default is 110)

Table You can use this table to help you check connectivity – record true or false here:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| From: | 192.168.16.1 | 192.168.16.2 | 192.168.32.1 | 192.168.32.2 | 192.168.48.1 | 192.168.48.2 |
| RMain: |  |  |  |  |  |  |
| RAcad: |  |  |  |  |  |  |
| RAdmin: |  |  |  |  |  |  |
| RRes: |  |  |  |  |  |  |

**Task 4 – Adding Public IP Subnets**

So far, you’re probably wondering why we even do OSPF in the first place. We’ve done a lot of work to get three subnets communicating – we did that in the first lab! This next step will give you that context (hopefully).

Check out my finished network for this step. You may want to read the next instructions first, take out a pencil and label the edges between routers with their CIDR subnets and the router/host IP addresses, as it will make configuration a little more sensible:  


Since these steps have been documented in Lab 1, I don’t repeat them here:

1. To each of the three ISR4321 routers connect a new 2960 switch (using ethernet) – note the interfaces (mine were on Fa01 on the router and Gig0/0/0 on the switches)
2. To each of the new switches, add a new PC, also connected over ethernet.
3. Try to connect the switch to the router – it should connect, but I had to pick it up and drop it down a couple of times until I was able to see it as “Gig0/0/1.” There seems to be a bug here, but with a little creative clickilation you should be able to get it setup.
4. On each of the three routers, configure the interface – give it the proper public class C public IP address from page 1 (RAcad: 202.150.83.0/24, RAdmin: 203.175.52.0/24, and RRes: 205.10.75.0/24). Make the interface’s address be the “.1”. So, RAcad’s ISR4321 router’s port connected to the new switch will be 202.10.75.1
5. Click on each of the new PCs, and give them a gateway of their subnet “.1”, and then their IP address will be “.5”, so the RAcad PC will be 202.150.83.5

**Task 5 – Connecting the Dots**In Lab 1, at this point, we would have to go to every router and manually add the routing information for each other route. Because of OSPF, all we needed to do was to configure each local router with its one and only route, and then that was broadcast throughout the rest of the network.

Verify that OSPF has done its job, go to the main router, and check the routes – and voila! The new subnets are in the main router, and even better, they will dynamically update as conditions on the network change!

1. Use the “show ip route” command:

Router>show ip route

Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP

D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area

N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2

E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP

i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area

\* - candidate default, U - per-user static route, o - ODR

P - periodic downloaded static route

Gateway of last resort is not set

C 192.168.16.0/24 is directly connected, FastEthernet4/0

C 192.168.32.0/24 is directly connected, FastEthernet5/0

C 192.168.48.0/24 is directly connected, FastEthernet6/0

O 202.150.83.0/24 [110/2] via 192.168.16.2, 00:00:23, FastEthernet4/0

O 203.175.52.0/24 [110/2] via 192.168.32.2, 00:01:51, FastEthernet5/0

O 205.10.75.0/24 [110/2] via 192.168.48.2, 00:01:28, FastEthernet6/0

223.223.223.0/32 is subnetted, 1 subnets

C 223.223.223.233 is directly connected, Loopback0

Router>

**When your Activity is complete, save and submit your work at this point.**

Now is a good time to build some of your own understanding of how OSPF works. Put packet tracer into Simulation mode, and observe the OSPF packets that get exchanged and look into the PDU’s that are sent to get an idea of what kind of information OSPF stores.

You can also try adding additional links or removing links and watching how OSPF updates itself.

# Scenario Two – A Hypercube

In this scenario, we will simulate a dense routing network. In this case, we use eight routers, numbered 0 through 7. The routers are connected using a *hypercube* architecture – each router sits on a vertex of a cube with *n* dimensions. Each router has *n* links between them. The example we’ll be working with is a cube with 3 dimensions, which has routers, and only twelve links between them. The advantage to this is that if any one router link goes down, there are still at least two routes through the network, giving high redundancy, and excellent performance. It also set a maximum on latency – In the worst case, we only need to go through a total of three routers to get between any machine.



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Source Router | Dest. Router | Network # | Src IP | Dst IP |
| 0 – g0/0 | 1 – g0/0 | 4 | 192.168.0.5 | 192.168.0.6 |
| 0 - g1/0 | 2 – g1/0 | 8 | 192.168.0.9 | 192.168.0.10 |
| 0 – g2/0 | 4 - g0/0 | 16 | 192.168.0.17 | 192.168.0.18 |
| 1 – g1/0 | 3 – g0/0 | 44 | 192.168.0.45 | 192.168.0.46 |
| 1 – g2/0 | 5 – g1/0 | 52 | 192.168.0.53 | 192.168.0.54 |
| 2 – g1/0 | 3 – g1/0 | 76 | 192.168.0.77 | 192.168.0.78 |
| 2 – g2/0 | 6 – g1/0 | 88 | 192.168.0.89 | 192.168.0.90 |
| 3 – g2/0 | 7 – g1/0 | 124 | 192.168.0.125 | 192.168.0.126 |
| 4 – g1/0 | 5 – g0/0 | 148 | 192.168.0.149 | 192.168.0.150 |
| 4 – g2/0 | 6 – g0/0 | 152 | 192.168.0.153 | 192.168.0.154 |
| 5 – g2/0 | 7 – g0/0 | 188 | 192.168.0.189 | 192.168.0.190 |
| 6 – g2/0 | 7 – g2/0 | 220 | 192.168.0.221 | 192.168.0.222 |

### Configure Routers

Configure the solution to match what I’ve done here:

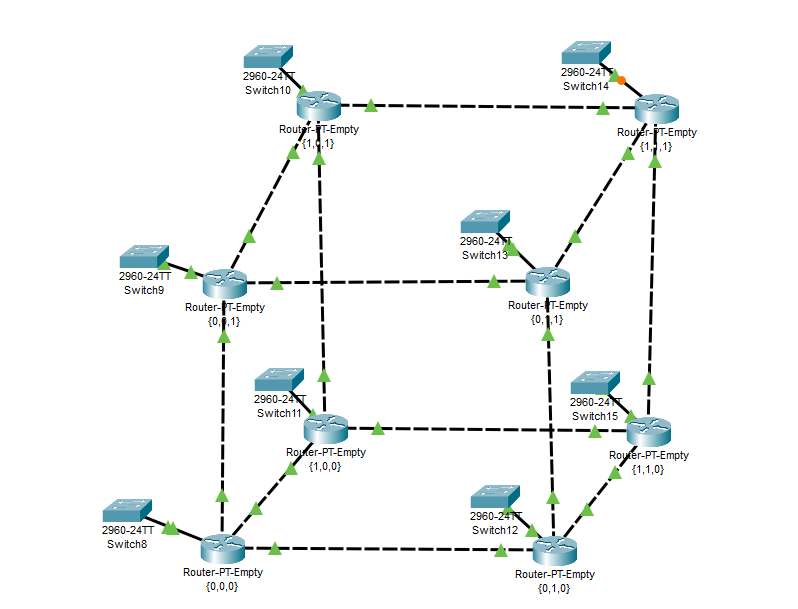
1. Place a PT-EMPTY router into your canvas
2. Open the configuration, turn it off, and add four PT-ROUTER-NM-1CGE cards into the right-most port, and turn it back on
3. Copy and paste that router to end up with a total of 8 routers.
4. Use the ethernet connections to make the links as shown in the diagram on the previous page. Use the routing information in the table to configure each of the subnets.
5. From each router, ping the adjacent router.

### Place and Configure Switches

Configure the solution to have eight 2960 switches:

1. Place one 2960 switch adjacent to each router (total of 8 switches)
2. Connect each switch to the router using the routers Gig3/0.
3. Configure the subnets on each of these subnets, router is the “.1” on this network, don’t forget the “no shut”:

|  |  |
| --- | --- |
| Router 0: 192.168.100.0/24 | Router 4: 192.168.104/24 |
| Router 1: 192.168.101.0/24 | Router 5: 192.168.105/24 |
| Router 2: 192.168.102.0/24 | Router 6: 192.168.106/24 |
| Router 3: 192.168.103.0/24 | Router 7: 192.168.107.24 |



### Configure Routes

Each of the routers needs to export its routes via OSPF, using the previous table, go to each router, and enable OSPF routing, and then select the addresses to export, and finally enable redistribution of connected networks. This is shown for Router 0, you’ll need to adjust the IP addresses for the other routers:

1. Enable OSPF  
   Router(config)# router ospf 100
2. Enable the 3 IP addresses for the router’s own interfaces:  
   Router(config-router)# network 192.168.0.5 0.0.0.0 area 0

Router(config-router)# network 192.168.0.9 0.0.0.0 area 0  
Router(config-router)# network 192.168.0.17 0.0.0.0 area 0

1. Enable redistribution of connected routes:  
   Router(config-router)# redistributed connected subnets

Wait for the message:

Router(config-router)#

00:05:27: %OSPF-5-ADJCHG: Process 132, Nbr 192.168.101.1 on GigabitEthernet0/0 from LOADING to FULL, Loading Done

Repeat on each of the other 7 routers. Again, you’ll need to adjust the IP addresses. Do you remember how to list the IP addresses configured on a machine? That can help here.

Upon completion of this step, you should now be able to ping the router/switch IP address from every router to every router, e.g. every router should now be able to ping 192.168.100.1, .101.1, .102.1, …

### Add a PC & Server

Add a PC to the switch connected to router 0, make its IP address 192.168.100.2 and its gateway 192.168.100.1. Add a server to the switch connected to router 7, make its IP address 192.168.107.1, and its gateway 192.168.107.1.

Try pinging from the PC to the server and vice versa – make sure this works.

Then, use the *tracert* command, and trace the path from the PC to the server – is it the shortest path? Sometimes it takes a few seconds of simulated time for OSPF to “converge” (remember the Dijkstra algorithm). If its not, wait a little and see if the OSPF state goes from LOADING to FULL again, and then check it. This is just an opportunity to explore the behavior of this algorithm.

### Fault Tolerance

Use the PC and do a traceroute to “192.168.101.1” – it ***should*** be two hops.

Put the network into “simulation” mode, use the event filters to only capture OSPF. Then, *delete* the link between router 0 and 1 (that’s OK, its easy to put back). Watch OSPF renegotiate. Then, traceroute from the PC to 192.168.101.1 again – how many hops did it become? Did it find a new shortest path?

Put the network link back. Watch OSPF renegotiate again. Make some observations about how many messages it took to bring the network back into balance. Do the traceroute one more time and see if we’re back to the fastest route (two hops). How much time did it take to get back to the fastest route?

# Part 3 - Multiple Areas

In the previous example, OSPF generated a lot of traffic between the nodes to restore its state. This is a known limitation to the algorithm – the number of messages grows *quadratically* with the number of routers: Needed messages for *n* machines.

One way to “fix” this is to put groups of routers into different areas and then route between them. This is a classic Divide & Conquer strategy – if we split the network up into a group of *m* and *o* routers with *p* links between them, then I need to have messages. Seems worse, until we do the math. In the previous, *n=8*, so we needed messages. If we group routers into two groups of 4, with 2 links between them, then I need to exchange: messages, a savings of

Start with the part 3 activity. This was the same set of routers that we used in part 2. I’ve gone through and reset the OSPF routing configuration by running the following command on each router:

1. Router(config)# no route ospf 100

## Configure Area 1 and 2.

Area one consists of routers 0, 1, 2 and 3. Using the same subnetting scheme as before, reconfigure OSPF, using process 100, but this time using area 1. Don’t forget to redistribute the directly connected subnets like you did in part 2. Remember that we need to *this router’s* IP address for each of the local links that are shared, and that there aren’t any links to the other area.

Do the same thing for routers 4,5,6, and 7, only this time, we use area 2.

Both networks should eventually converge. When they do, you should be able see the routes *within* an area, but there aren’t any links *between* areas, so you won’t be able to go between them.

You can use *ping* and *traceroute* to verify that the routes are working correctly. Double check to make sure that each router is using the correct # of steps – e.g. going from router 6 to router 5 should take two jumps (either 6 -> 4 -> 5 or 6 -> 7 -> 5).

## Configure Area 0 – The Network Backbone

OSPF reserves “area 0” to be the network backbone. Routers connected to this are know as Area Border Routers, and are expected to route between multiple area.

Lets start by making a single connection between the two areas:

1. Add a network connection from router 3 to router 7
2. Configure the same IP addresses as you used in Part 2.
3. Re-enter OSPF route configuration in both routers 3 & 7.
4. Add the network between them as you have been doing, but use area 0 for just this link.

Wait for the network to converge.

Verify that we can reach between the networks using *ping* and *traceroute*.

### Types of Routers

Use the command “show ip ospf” on either router 3 or 7, and then use it on router 1 or 4. Note the differences in the output:

Routers 3 & 7 have become a different *type* of router: they are now an area border router because they join these two areas together.

Supports opaque LSA

It is an autonomous system boundary router

It is an area border router

### 

### Routing Distance

One of the advantages of the using the *hypercube* is that every router is reachable in *d-1* hops, where *d* is the dimension of the hypercube. So, for a *d=3* network (a cube), everything is 0, 1, or 2 hops away. But, now that we have two squares connected by a single link, we have a “worst case” of 4 hops away. For example, 1 -> 2 -> 3 -> 7 -> 6.

We can reduce this by adding another network links. Start by filling in the table with the current “hop counts” from each router to another:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Source Router* | *Destination Router* | | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 | 0 |  |  |  |  |  |  |  |
| 1 |  | 0 |  |  |  |  |  |  |
| 2 |  |  | 0 |  |  |  |  |  |
| 3 |  |  |  | 0 |  |  |  |  |
| 4 |  |  |  |  | 0 |  |  |  |
| 5 |  |  |  |  |  | 0 |  |  |
| 6 |  |  |  |  |  |  | 0 |  |
| 7 |  |  |  |  |  |  |  | 0 |

If we are only willing to have a hop count of 3 – if we add ***one more*** network links, we can achieve this – figure out what network you need to add and add it to the network and configure OSPF for this new link.