UNIVERSITY OF
SOUTH CAROLINA

MPLS AND ADVANCED BGP TOPICS LAB SERIES

Book Version: 07-09-2021

Principal Investigator: Jorge Crichigno

Award 1829698
"CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers"

Contents

MPLS AND ADVANCED BGP TOPICS

Lab 1: Configuring Multiprotocol BGP

Document Version: 12-25-2020

Award 1829698
"CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers"

Contents

Overview

This lab introduces Multiprotocol Border Gateway Protocol (MP-BGP), an extension of BGP that enables the distribution of routing information to multiple network layers and address families. In an environment where IPv4 and IPv6 are both configured, BGP routers become neighbors using IPv4 addresses and exchange IPv6 prefixes. The lab aims to configure MP-BGP to advertise IPv4 and IPv6 prefixes across Autonomous Systems (ASes).

This lab is based on the Cisco Certified Network Professional (CCNP) ROUTE Version 7 training, laboratory 7-5[10]. While this laboratory material uses FRR as the router (which includes the routing protocols) and Linux as the control plane operating system, vendors' operating systems such as Cisco have similar functionality.

Objectives

By the end of this lab, you should be able to:

1. Understand the concept of MP-BGP.
2. Understand Intradomain and Interdomain routing concept.
3. Verify OSPFv2 and OSPFv3.
4. Verify IBGP and EBGP configuration.
5. Use MP-BGP to distribute IPv4 and IPv6 in parallel.

Lab settings

The information in Table 1 provides the credentials to access Client machine.

Table 1. Credentials to access Client machine.

Device	Account	Password
Client	admin	password

Lab roadmap

This lab is organized as follows:

1. Section 1: Introduction.
2. Section 2: Lab topology.
3. Section 3: Enable and verify OSPF configuration.
4. Section 4: Configure and verify BGP for IPv4 networks.
5. Section 5: Configure and verify BGP for IPv6 networks.
6. Section 6: Verify BGP configuration.

1 Introduction

1.1 IPv4 and IPv6 addresses

The IP transmits blocks of data called datagrams from the source to the destination that are identified by a fixed-length address. IPv4 is the dominant protocol of the Internet that identifies devices on a network. It uses a 32-bit address space which allows to store more than 4 billion addresses[1].

As the Internet evolves, more IP addresses are required than the IPv4 offers. Thus, IPv6 was designed to fulfill the need for more Internet addresses. IPv6 uses a 64-bit address space which allows to store a huge number of addresses (more than 340 undecillion)[2].

IPv6 has multiple address types. The link local address is used on a single link, or within a Local Area Network (LAN). Link local addresses do not have to be globally unique, thus, they are not routable. The prefix of the link local address is fe80::/10. Global unicast address is globally unique, and it is used to identify a single interface on the Internet. The prefix of the global unicast address is 2000::/3 [3].

Consider Figure 1. In IPv6, the link local address is used between routers that are directly connected, whereas the global address is used between routers that do not share a common link.

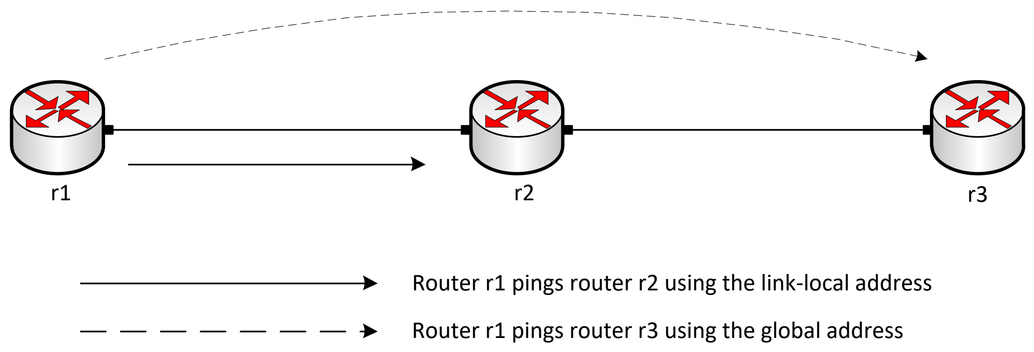

r1
r2
r3
Router r1 pings router r2 using the link-local address
Router r1 pings router r3 using the global address

Figure 1. IPv6 link local and global addresses.

1.2 Intradomain and Interdomain routing protocols

The Internet consists of many independent administrative domains, referred to as ASes. ASes are operated by different organizations, which can run their internal routing protocols. A routing protocol that runs within an AS is referred to as intradomain routing protocol. One of the most widely used intradomain protocols is OSPF. Since an AS may be large and nontrivial to manage, OSPF allows an AS to be divided into numbered areas[4]. An area is a logical collection of networks, routers, and links. All routers in the same area have detailed information of the topology within their area[5]. Traditionally, OSPF only supported IPv4 addresses (OSPFv2); however, it was redesigned to support IPv6 as well (OSPFv3)[6].

A routing protocol that runs between ASes is referred to as interdomain routing protocol. ASes may use different intradomain routing protocols; however, they must use the same interdomain routing protocol, i.e., BGP. BGP allows the enforcement of different routing policies on the traffic from one AS to another. For example, a security policy can prevent the dissemination of routing information from one AS to another[4]. BGP has been extended to carry routing information for multiple protocols, such as IPv6 [7].

BGP is referred to as EBGP when it is running between different ASes, whereas it is referred to as IBGP when it is running within an AS[4]. IBGP is usually used to distribute the EBGP learned routes among the routers within the same AS[4].

Consider Figure 2. The intradomain routing protocol within AS 100 is OSPF, and the interdomain routing protocol between AS 100 and AS 200 is BGP (EBGP). Routers within the same AS advertise their EBGP learned routes among each other through IBGP.

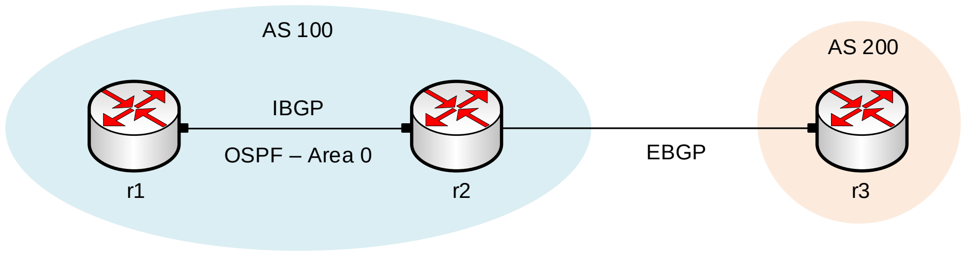


Figure 2. Routers that exchange information within the same AS use OSPF and IBGP, while routers that exchange information between different ASes use EBGP.

1.3 MP-BGP

In an IPv4 environment, BGP establishes sessions using IPv4, i.e., BGP peers are configured with IPv4 addresses. The routes that are advertised by BGP also have IPv4 addresses[8]. MP-BGP was introduced to make BGP available for other network layer protocols. MP-BGP supports IPv4 and IPv6 unicast, IPv4 and IPv6 multicast and also VPN labels that are used in MPLS-VPN.

Consider Figure 3. The BGP session established between router r1 and router r2 is done using IPv4. Using MP-BGP, router r1 can advertise the address of the attached IPv6 LAN through the current BGP session.

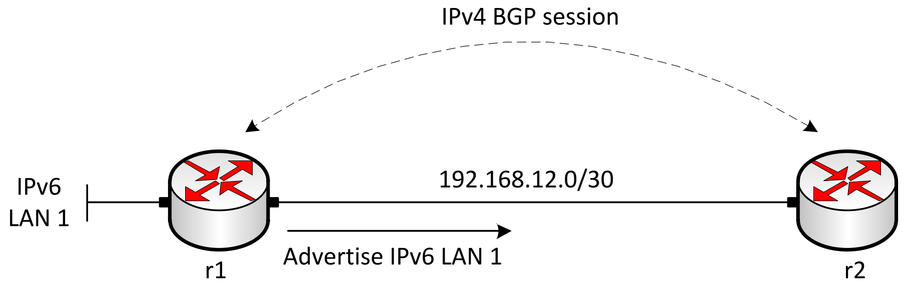


Figure 3. Advertising an IPv6 LAN address through a BGP IPv4 session.

Similarly, consider Figure 4. The BGP session established between router r1 and router r2 is done using IPv6. Using MP-BGP, router r1 can advertise the address of the attached IPv4 LAN through the current BGP session.

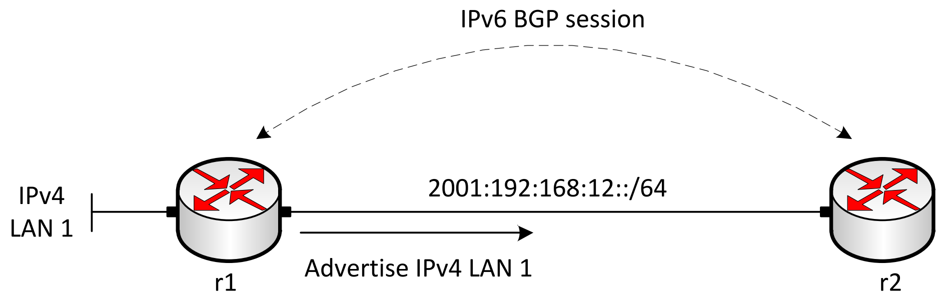


Figure 4. Advertising an IPv4 LAN address through a BGP IPv6 session.

2 Lab topology

Consider Figure 5. The topology consists of two ASes. The Internet Service Provider (ISP), i.e., router r1, provides Internet service to the Campus network (routers r2 and r3). The ASN assigned to the ISP and the Campus network is 100 and 200, respectively. The ISP communicates with the Campus via EBGP routing protocol, and the routers within the Campus network communicate using IBGP and OSPF.

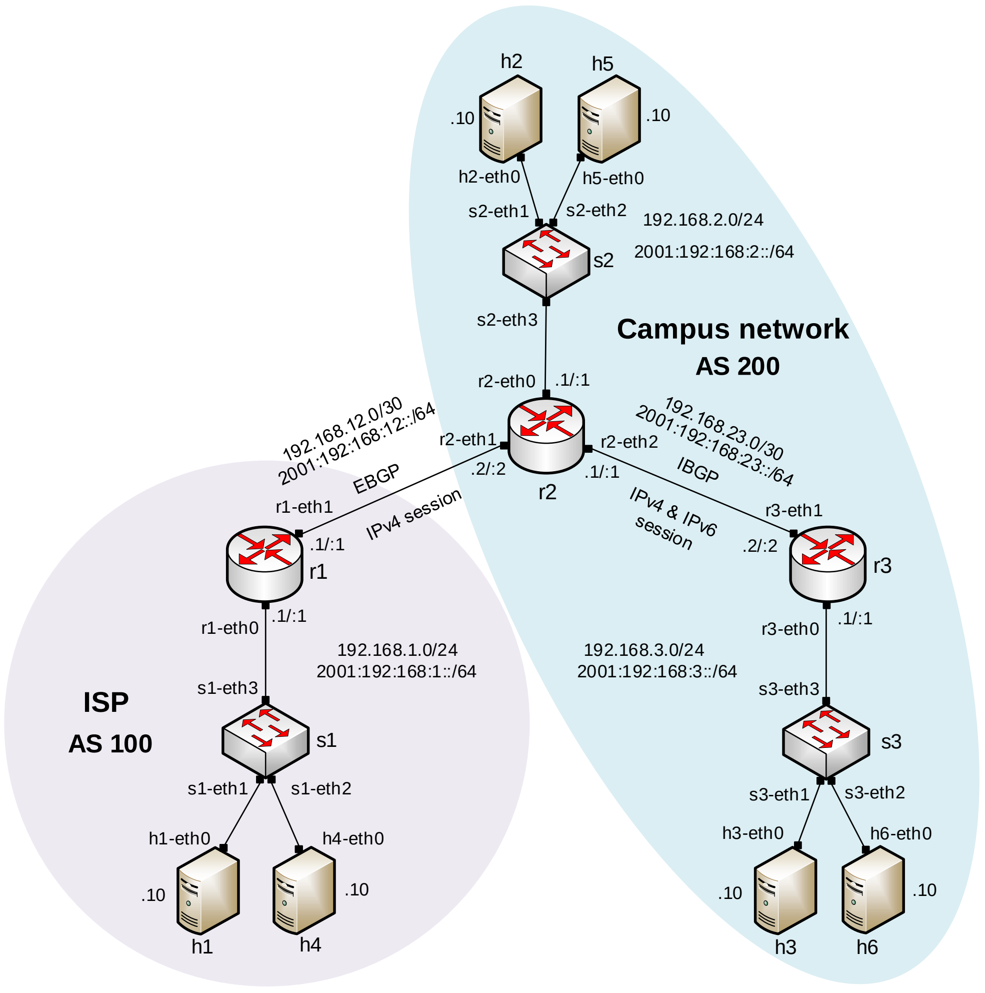


Figure 5. Lab topology.

2.1 Lab settings

Routers and hosts are already configured according to the IP addresses shown in Table 2.

Table 2. Topology information.

Device	Interface	IPV4 Address	IPV6 Address	Default gateway
r1 (ISP)	r1-eth0	192.168.1.1/24	2001:192:168:1::1/64	N/A
	r1-eth1	192.168.12.1/30	2001:192:168:12::1/64	N/A
	lo	192.168.11.1/32	2001:192:168:11::1/64	N/A
	r2-eth0	192.168.2.1/24	2001:192:168:2::1/64	N/A
	r2-eth1	192.168.12.2/30	2001:192:168:12::2/64	N/A

r2 (Campus network)	r2-eth2	192.168.23.1/30	2001:192:168:23::1/64	N/A
	lo	192.168.22.1/32	2001:192:168:22::1/64	N/A
r3 (Campus network)	r3-eth0	192.168.3.1/24	2001:192:168:3::1/64	N/A
	r3-eth1	192.168.23.2/30	2001:192:168:23::2/64	N/A
	lo	192.168.33.1/32	2001:192:168:33::1/64	N/A
h1	h1-eth0	192.168.1.10/24	N/A	192.168.1.1
h2	h2-eth0	192.168.2.10/24	N/A	192.168.2.1
h3	h3-eth0	192.168.3.10/24	N/A	192.168.3.1
h4	h4-eth0	N/A	2001:192:168:1::10/64	2001:192:168:1::1
h5	h5-eth0	N/A	2001:192:168:2::10/64	2001:192:168:2::1
h6	h6-eth0	N/A	2001:192:168:3::10/64	2001:192:168:3::1

2.2 Open topology and load the configuration

Step 1. Start by launching MiniEdit by clicking on desktop's shortcut. When prompted for a password, type `password`.

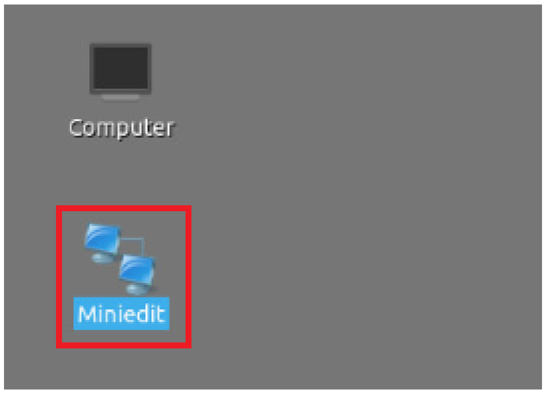


Figure 6. MiniEdit shortcut.

Step 2. On MiniEdit's menu bar, click on *File* then *open* to load the lab's topology. Locate the *lab1.mn* topology file in the default directory, */home/frr/MPLS_advanced_BGP/lab1* and click on *Open*.

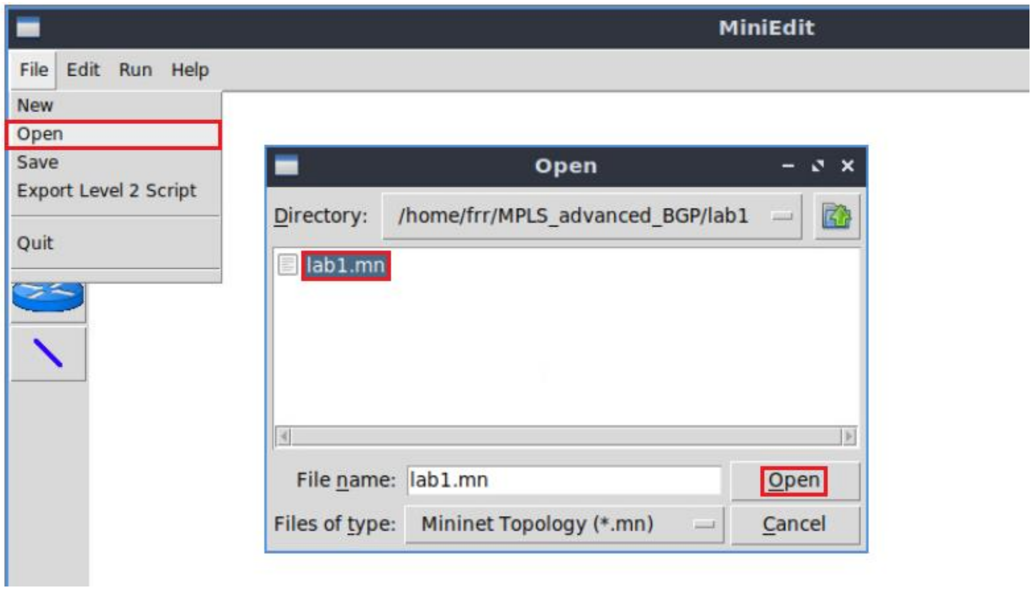


Figure 7. MiniEdit's Open dialog.

At this point, the topology is loaded with all the required network components. You will execute a script that will load the configuration of the routers.

Step 3. Open the Linux Terminal.

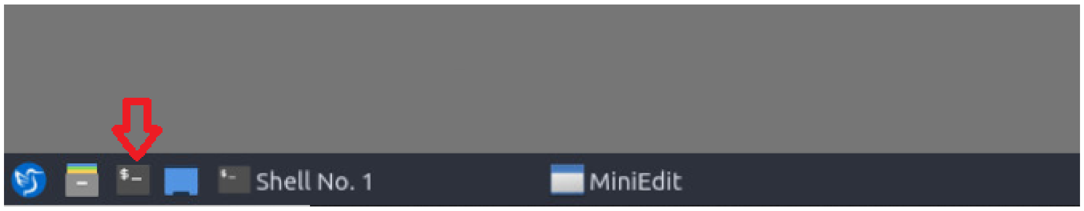
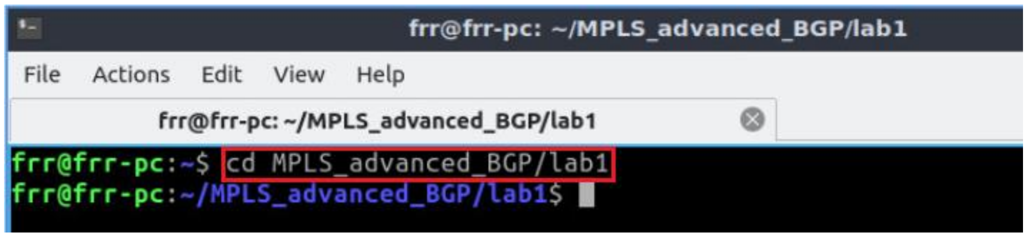


Figure 8. Opening Linux terminal

Step 4. Click on the Linux terminal and navigate into *MPLS_advanced_BGP/lab1* directory by issuing the following command. This folder contains a configuration file, and the script is responsible for loading the configuration. The configuration file will assign the IP addresses to the routers' interfaces. The `cd` command is short for change directory followed by an argument that specifies the destination directory.

```
cd MPLS_advanced_BGP/lab1
```

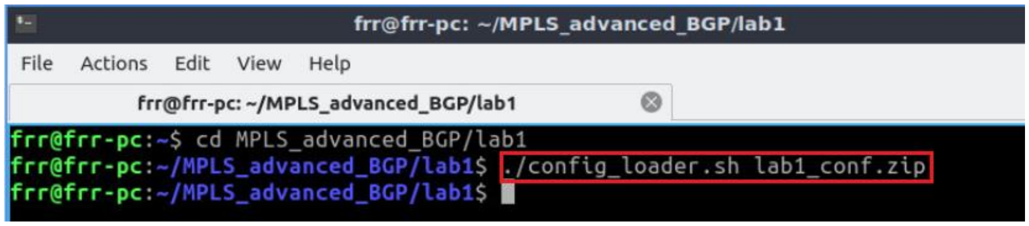

frr@frr-pc: ~/MPLS_advanced_BGP/lab1
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab1

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab1
frr@frr-pc:~/MPLS_advanced_BGP/lab1$
```

Figure 9. Entering to the *MPLS_advanced_BGP/lab1* directory.

Step 5. To execute the shell script, type the following command. The argument of the program corresponds to the configuration zip file that will be loaded in all the routers in the topology.

```
./config_loader.sh lab1_conf.zip
```


frr@frr-pc: ~/MPLS_advanced_BGP/lab1
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab1

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab1
frr@frr-pc:~/MPLS_advanced_BGP/lab1$ ./config_loader.sh lab1_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab1$
```

Figure 10. Executing the shell script to load the configuration.

Step 6. At this point, hosts h1, h2, and h3 interfaces are configured. To proceed with the emulation, click on the *Run* button located on the lower left-hand side.

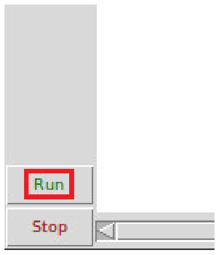


Figure 11. Starting the emulation.

Step 7. Click on Mininet's terminal, i.e., the one launched when MiniEdit was started.

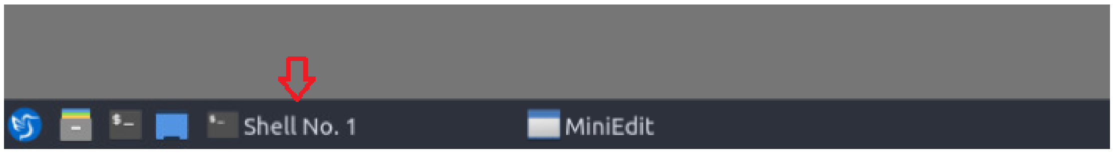
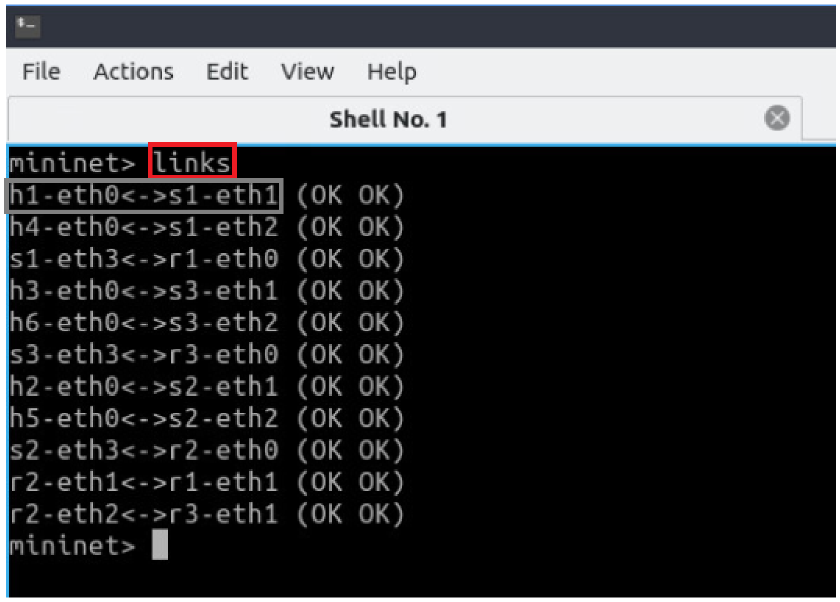


Figure 12. Opening Mininet's terminal.

Step 8. Issue the following command to display the interface names and connections.

```
links
```



```
mininet> links
h1-eth0<->s1-eth1 (OK OK)
h4-eth0<->s1-eth2 (OK OK)
s1-eth3<->r1-eth0 (OK OK)
h3-eth0<->s3-eth1 (OK OK)
h6-eth0<->s3-eth2 (OK OK)
s3-eth3<->r3-eth0 (OK OK)
h2-eth0<->s2-eth1 (OK OK)
h5-eth0<->s2-eth2 (OK OK)
s2-eth3<->r2-eth0 (OK OK)
r2-eth1<->r1-eth1 (OK OK)
r2-eth2<->r3-eth1 (OK OK)
mininet> 
```

Figure 13. Displaying network interfaces.

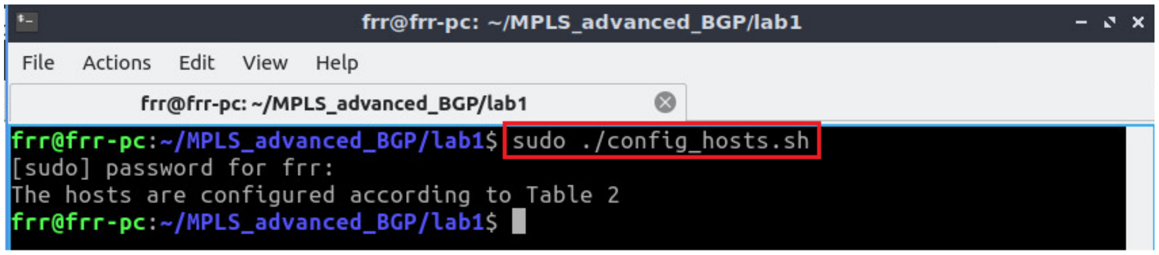
In Figure 13, the link displayed within the gray box indicates that interface eth0 of host h1 connects to interface eth1 of switch s1 (i.e., *h1-eth0<->s1-eth1*).

2.3 Configure and verify the hosts

You will verify the IPv4 addresses of each host (h1, h2, and h3) following Table 2. Additionally, you will verify IP addresses and the default gateway for IPv6 hosts (h4, h5, and h6).

Step 1. In the Linux terminal, type the following command to configure IP addresses in the hosts according to Table 2. Type `password` as the password. A message will indicate that the configuration was applied correctly.

```
sudo ./config_hosts.sh
```


frr@frr-pc: ~/MPLS_advanced_BGP/lab1
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab1

```
frr@frr-pc:~/MPLS_advanced_BGP/lab1$ sudo ./config_hosts.sh
[sudo] password for frr:
The hosts are configured according to Table 2
frr@frr-pc:~/MPLS_advanced_BGP/lab1$ 
```

Figure 14. Setting hosts configuration.

Step 2. Hold right-click on host h1 and select *Terminal*. This opens the terminal of host h1 and allows the execution of commands on that host.

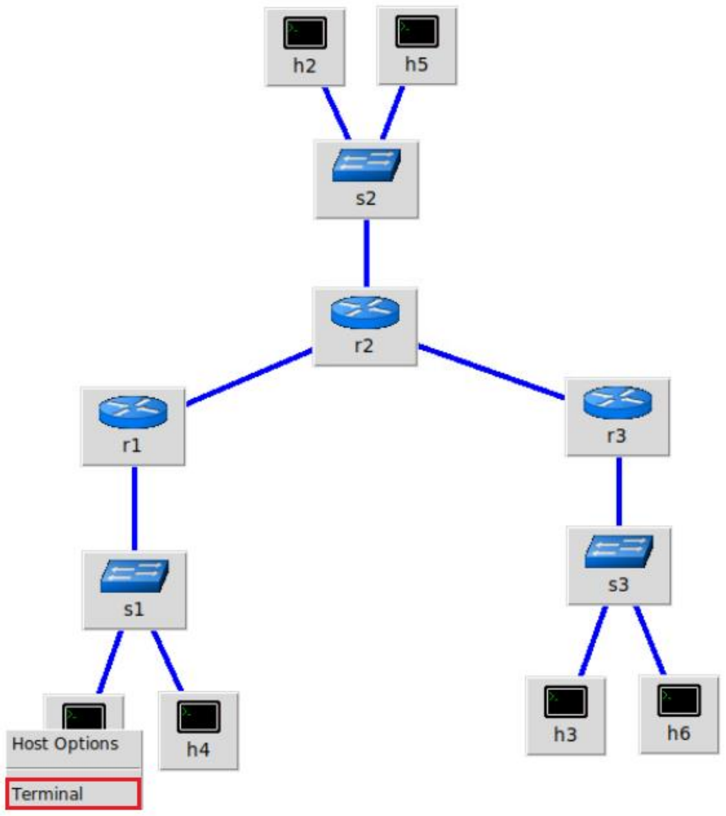
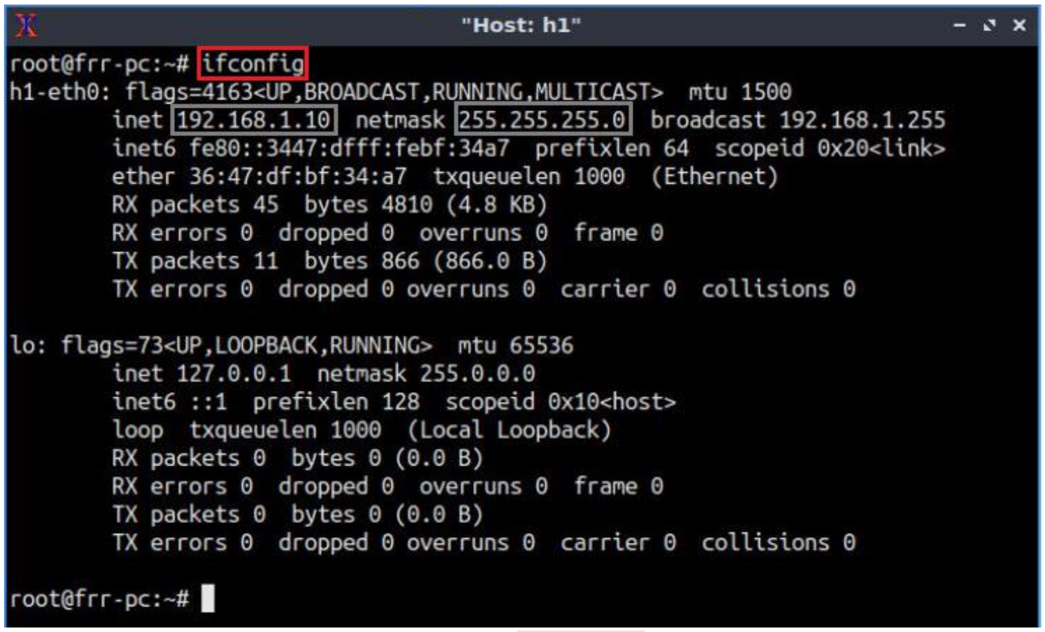


Figure 15. Opening terminal on host h1.

Step 3. In host h1 terminal, type the command shown below to verify that the IP address was assigned successfully. You will verify that interface *h1-eth0* is configured with IP address 192.168.1.10 and the subnet mask 255.255.255.0.

```
ifconfig
```

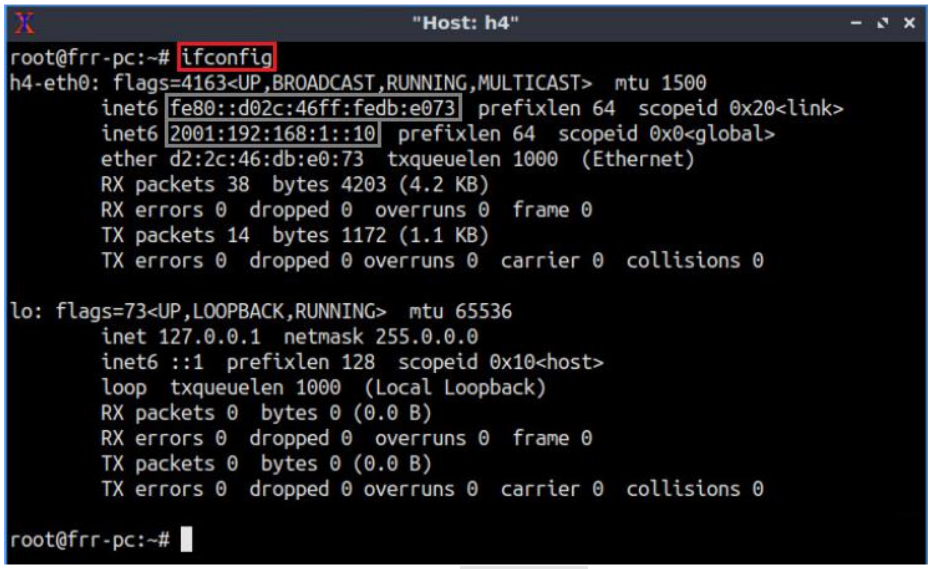


```
"Host: h1"
root@frr-pc:~# ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.10  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::3447:dfff:febf:34a7  prefixlen 64  scopeid 0x20<link>
        ether 36:47:df:bf:34:a7  txqueuelen 1000  (Ethernet)
        RX packets 45  bytes 4810 (4.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 11  bytes 866 (866.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@frr-pc:~#
```

Figure 16. Output of `ifconfig` command.

Step 4. In host h4 terminal, type the following command to verify IPv6 addresses assigned to the interface *h4-eth0*.

```
ifconfig
```



```
"Host: h4"
root@frr-pc:~# ifconfig
h4-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::d02c:46ff:fedb:e073  prefixlen 64  scopeid 0x20<link>
        inet6 2001:192:168:1::10  prefixlen 64  scopeid 0x0<global>
        ether d2:2c:46:db:e0:73  txqueuelen 1000  (Ethernet)
        RX packets 38  bytes 4203 (4.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 14  bytes 1172 (1.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@frr-pc:~#
```

Figure 17. Output of `ifconfig` command.

Consider the figure above. There are two IPv6 addresses assigned to host h4. IPv6 addresses fe80::d02c:46ff:fedb:e073 and 2001:192:168:1::10 are assigned as link local address and global address, respectively. Link local address is used to communicate with directly connected networks.

You may notice a different link local address since they are assigned randomly in FRR.

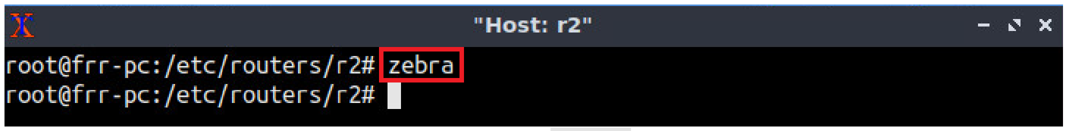
3 Enable daemons and verify OSPF configuration

In this section, the user will enable the OSPF daemon that will allow router r2 and router r3 to run OSPFv2 and OSPFv3. Then, the user will verify that the configuration is according to Table 2. Finally, IPv6 forwarding operation will be enabled in the routers.

3.1 Enable OSPFv2 and OSPFv3 daemons

Step 1. In router r2 terminal, you will start zebra daemon, which is a multi-server routing software that provides TCP/IP based routing protocols. The configuration will not be working if you do not enable zebra daemon initially. In order to start the zebra, type the following command:

```
zebra
```

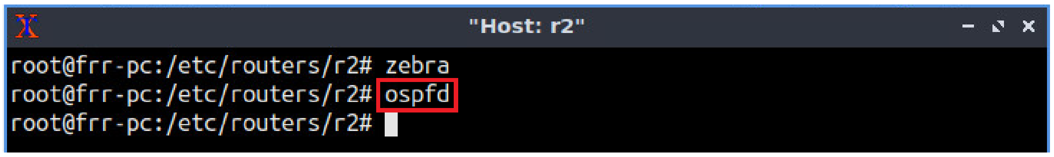


```
root@frr-pc:/etc/routers/r2# zebra
root@frr-pc:/etc/routers/r2#
```

Figure 18. Starting `zebra` daemon.

Step 2. Type the following command in router r2 to enable OSPF daemon for IPv4 networks:

```
ospfd
```

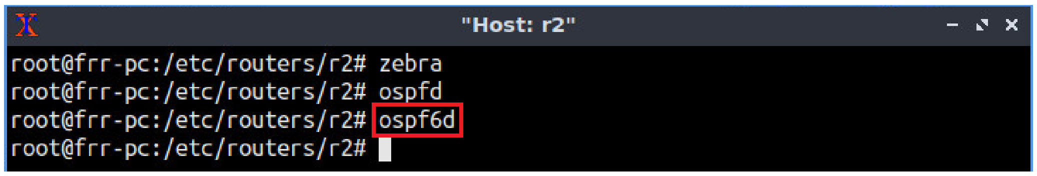


```
root@frr-pc:/etc/routers/r2# zebra
root@frr-pc:/etc/routers/r2# ospfd
root@frr-pc:/etc/routers/r2#
```

Figure 19. Starting OSPF daemon for IPv4 networks.

Step 3. Type the following command in router r2 to enable OSPF daemon for IPv6 networks:

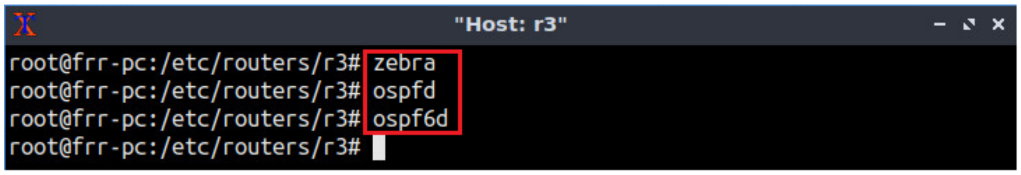
```
ospf6d
```



```
root@frr-pc:/etc/routers/r2# zebra
root@frr-pc:/etc/routers/r2# ospfd
root@frr-pc:/etc/routers/r2# ospf6d
root@frr-pc:/etc/routers/r2#
```

Figure 20. Starting OSPF daemon for IPv6 networks.

Step 4. Proceed similarly from step 1 to step 3 in router r3. The steps are summarized in the figure below.



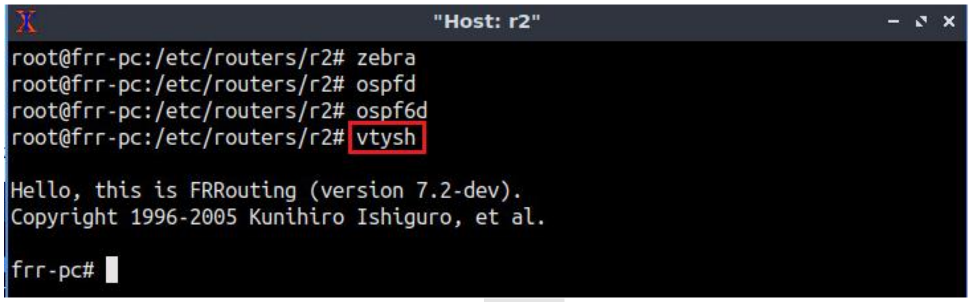
```
root@frr-pc:/etc/routers/r3# zebra
root@frr-pc:/etc/routers/r3# ospfd
root@frr-pc:/etc/routers/r3# ospf6d
root@frr-pc:/etc/routers/r3#
```

Figure 21. Steps to enable daemons in router r3.

3.2 Verify routing tables

Step 1. In order to enter to router r2 terminal, issue the following command. *vtysh* should be started in order to provide all the CLI commands defined by the daemons. To proceed, issue the following command:

```
vtysh
```

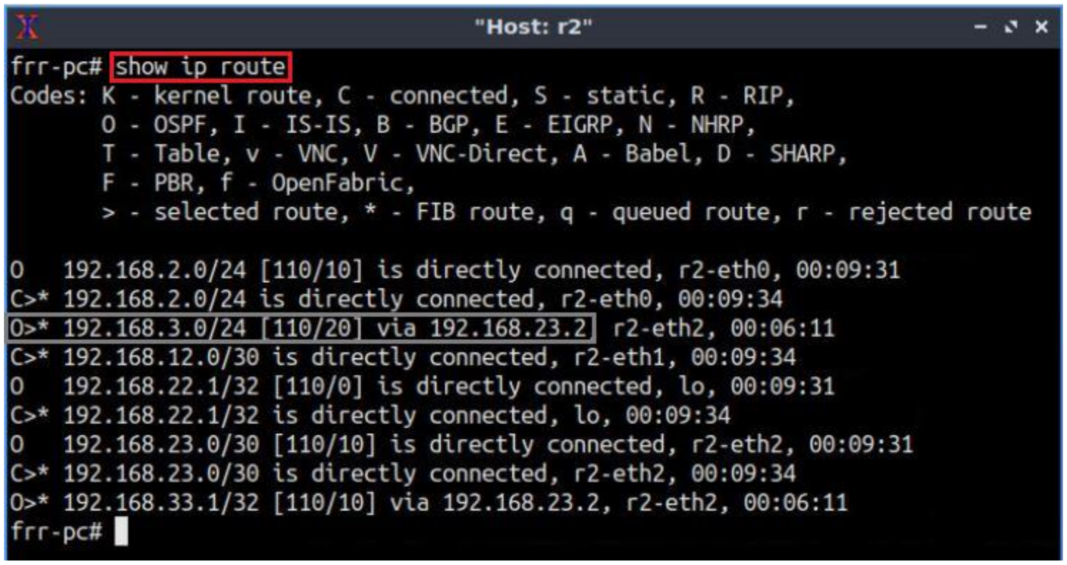


```
root@frr-pc:/etc/routers/r2# zebra
root@frr-pc:/etc/routers/r2# ospfd
root@frr-pc:/etc/routers/r2# ospf6d
root@frr-pc:/etc/routers/r2# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 22. Starting `vtysh` in router r2.

Step 2. Type the following command to display the IPv4 routing table in router r2.

```
show ip route
```



```
frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

O   192.168.2.0/24 [110/10] is directly connected, r2-eth0, 00:09:31
C>* 192.168.2.0/24 is directly connected, r2-eth0, 00:09:34
O>* 192.168.3.0/24 [110/20] via 192.168.23.2, r2-eth2, 00:06:11
C>* 192.168.12.0/30 is directly connected, r2-eth1, 00:09:34
O   192.168.22.1/32 [110/0] is directly connected, lo, 00:09:31
C>* 192.168.22.1/32 is directly connected, lo, 00:09:34
O   192.168.23.0/30 [110/10] is directly connected, r2-eth2, 00:09:31
C>* 192.168.23.0/30 is directly connected, r2-eth2, 00:09:34
O>* 192.168.33.1/32 [110/10] via 192.168.23.2, r2-eth2, 00:06:11
frr-pc#
```

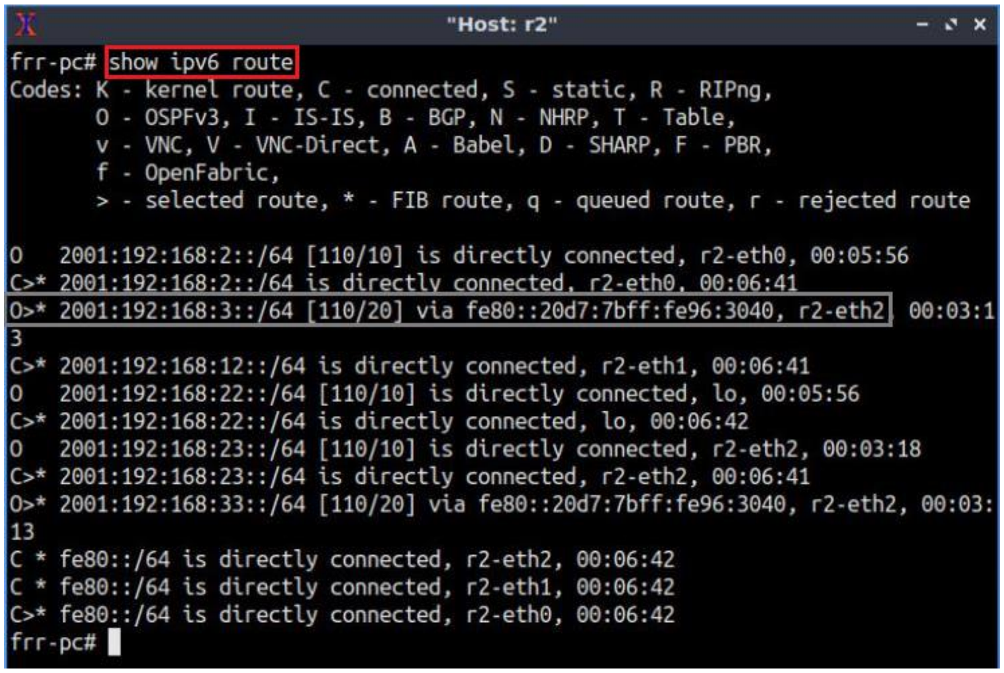
Figure 23. Displaying IPv4 routing table in router r2.

Consider the figure above. Router r2 has a route to the network 192.168.3.0/24 via IP address 192.168.23.2 (interface *r2-eth2*). Networks 192.168.2.0/24 and 192.168.23.0/30 have two available paths from router r2. The administrative distance (AD) of the paths advertised through OSPF is 110. The AD is a value used by routers to select the best path when there are multiple available routes to the same destination. A smaller AD is always preferable to the routers. The characters `>*` indicate that the following path is used to reach a specific network. Router r2 prefers directly connected networks over OSPF since the former has a lower AD than the latter.

It may take a while to show all the routes.

Step 3. To show the IPv6 routing table, type the following command.

```
show ipv6 route
```



```
"Host: r2"
frr-pc# show ipv6 route
Codes: K - kernel route, C - connected, S - static, R - RIPng,
       O - OSPFv3, I - IS-IS, B - BGP, N - NHRP, T - Table,
       v - VNC, V - VNC-Direct, A - Babel, D - SHARP, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

O   2001:192:168:2::/64 [110/10] is directly connected, r2-eth0, 00:05:56
C>* 2001:192:168:2::/64 is directly connected, r2-eth0, 00:06:41
O>* 2001:192:168:3::/64 [110/20] via fe80::20d7:7bff:fe96:3040, r2-eth2, 00:03:1
3
C>* 2001:192:168:12::/64 is directly connected, r2-eth1, 00:06:41
O   2001:192:168:22::/64 [110/10] is directly connected, lo, 00:05:56
C>* 2001:192:168:22::/64 is directly connected, lo, 00:06:42
O   2001:192:168:23::/64 [110/10] is directly connected, r2-eth2, 00:03:18
C>* 2001:192:168:23::/64 is directly connected, r2-eth2, 00:06:41
O>* 2001:192:168:33::/64 [110/20] via fe80::20d7:7bff:fe96:3040, r2-eth2, 00:03:
13
C * fe80::/64 is directly connected, r2-eth2, 00:06:42
C * fe80::/64 is directly connected, r2-eth1, 00:06:42
C>* fe80::/64 is directly connected, r2-eth0, 00:06:42
frr-pc#
```

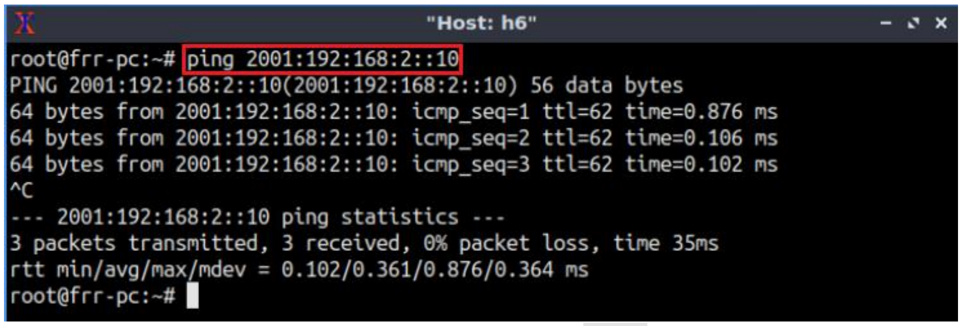
Figure 24. Displaying IPv6 routing table in router r2.

Consider the figure above. Router r2 has a route to the network 2001:192:168:3::/64 via interface *r2-eth2*.

Step 4. Proceed similarly from step 1 to step 3 in router r3.

Step 5. In host h6 terminal, perform a connectivity test between host h6 and host h5 by issuing the following command. To stop the test, press `Ctrl+c`. The result will show a successful connectivity test.

```
ping 2001:192:168:2::10
```



```
"Host: h6"
root@frr-pc:~# ping 2001:192:168:2::10
PING 2001:192:168:2::10(2001:192:168:2::10) 56 data bytes
64 bytes from 2001:192:168:2::10: icmp_seq=1 ttl=62 time=0.876 ms
64 bytes from 2001:192:168:2::10: icmp_seq=2 ttl=62 time=0.106 ms
64 bytes from 2001:192:168:2::10: icmp_seq=3 ttl=62 time=0.102 ms
^C
--- 2001:192:168:2::10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 35ms
rtt min/avg/max/mdev = 0.102/0.361/0.876/0.364 ms
root@frr-pc:~#
```

Figure 25. Connectivity test using `ping` command.

4 Configure and verify BGP for IPv4 networks

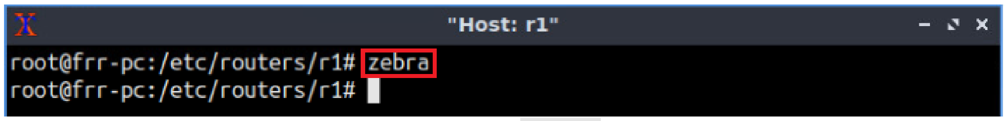
In this section, you will configure BGP in all routers. Routers r2 and r3 communicate with router r1 through EBGP, while router r2 communicates with router r3 through IBGP. You will assign BGP neighbors to allow the routers to exchange BGP routes.

You will configure EBGP so that router r1 uses IPv4 as the BGP transport for IPv4 sessions. For IBGP, you will configure router r2 so that IPv4 routing information is transported by IPv4 TCP sessions.

4.1 Configure and verify EBGP in router r1

Step 1. In order to start the zebra daemon in router r1, type the following command:

```
zebra
```

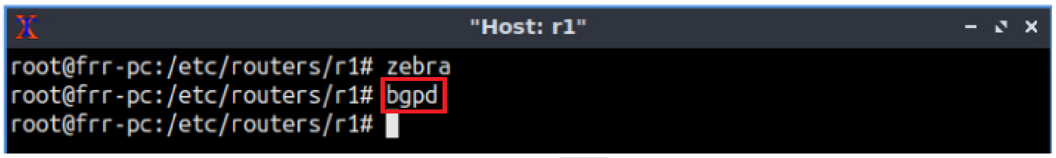


```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1#
```

Figure 26. Starting `zebra` daemon.

Step 2. Type the following command in router r1 terminal to enable and start BGP routing protocol.

```
bgpd
```

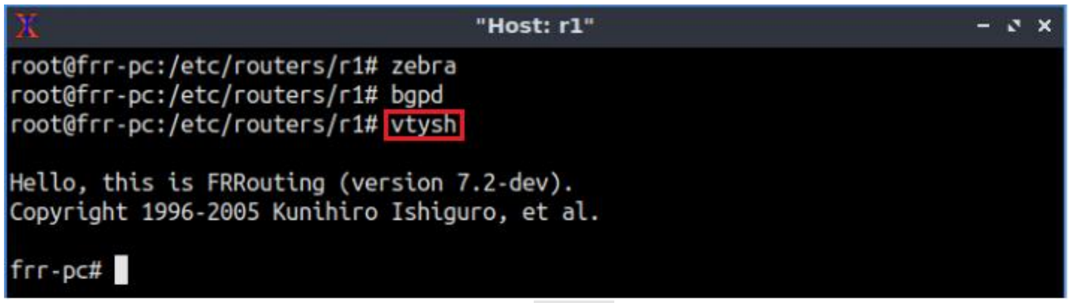


```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1#
```

Figure 27. Starting `BGP` daemon.

Step 3. In order to enter to router r1 terminal, type the following command:

```
vtysh
```

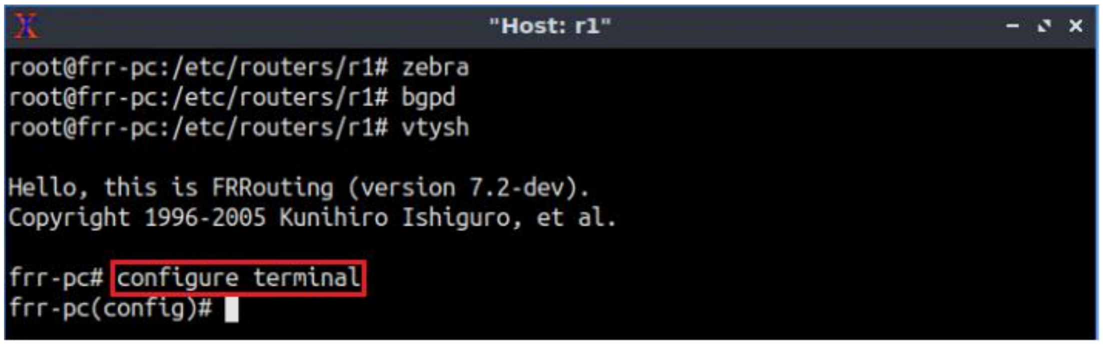


```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 28. Starting `vtysh` in router r1.

Step 4. To enable router r1 configuration mode, issue the following command:

```
configure terminal
```

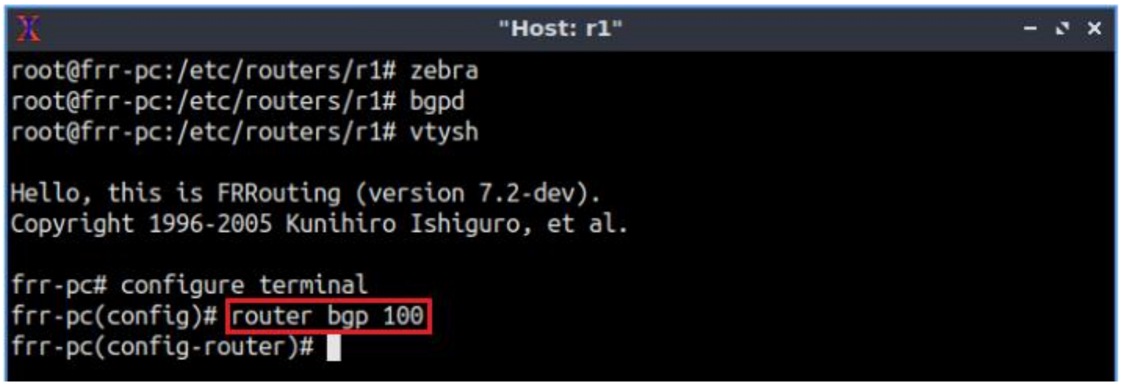


```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)#
```

Figure 29. Enabling configuration mode in router r1.

Step 5. The ASN assigned for router r1 is 100. In order to enter into the configuration mode, type the following command:

```
router bgp 100
```

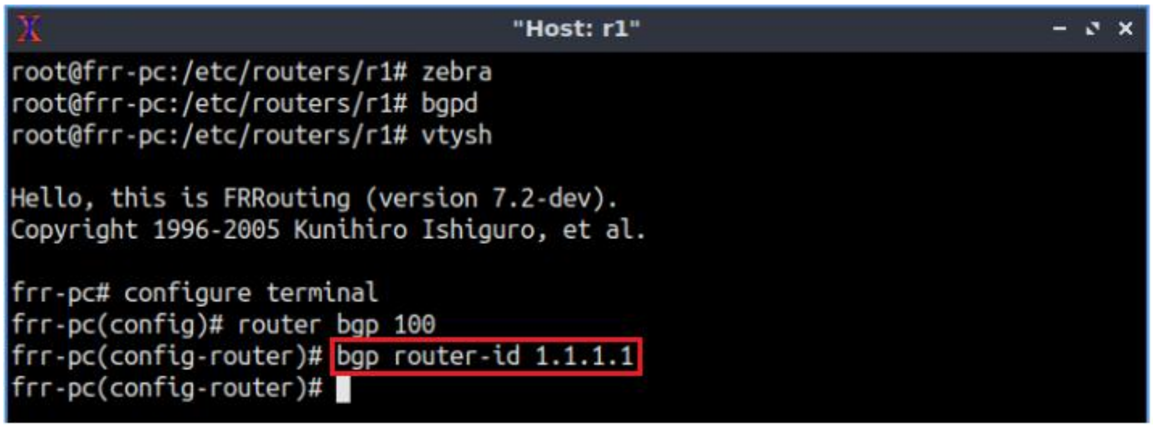


```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)#
```

Figure 30. Configuring BGP in router r1.

Step 6. Assign a router ID to router r1 by issuing the following command.

```
bgp router-id 1.1.1.1
```

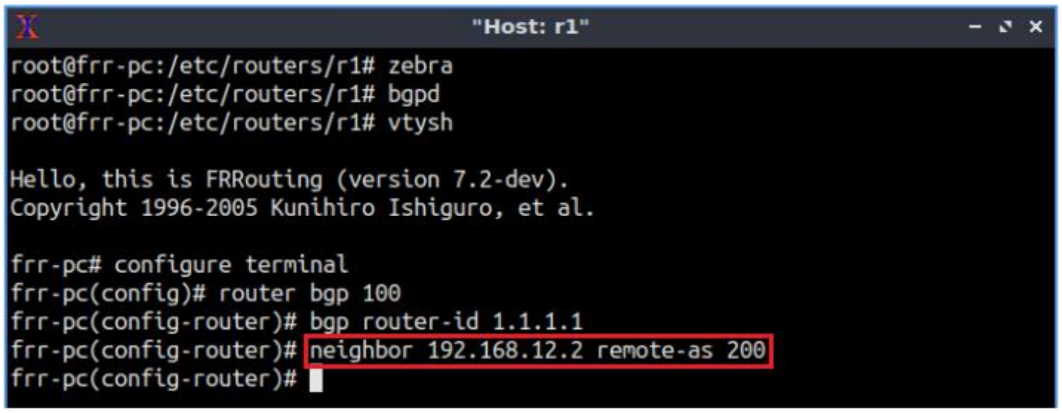


```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 1.1.1.1
frr-pc(config-router)#
```

Figure 31. Assigning a router ID in router r1.

Step 7. To configure a BGP neighbor to router r1 (AS 100), type the following command. This command specifies the neighbor IP address (192.168.12.2) and the ASN of the remote BGP peer (AS 200). This neighbor will act as the BGP transport for both IPv4 and IPv6 networks.

```
neighbor 192.168.12.2 remote-as 200
```

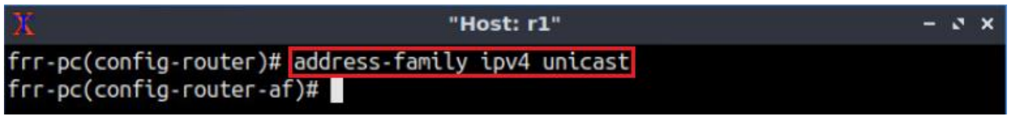


```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 1.1.1.1
frr-pc(config-router)# neighbor 192.168.12.2 remote-as 200
frr-pc(config-router)#
```

Figure 32. Assigning BGP neighbor to router r1.

Step 8. Type the following command to enter address-family mode where you can configure routing sessions that use standard IPv4 address prefixes.

```
address-family ipv4 unicast
```

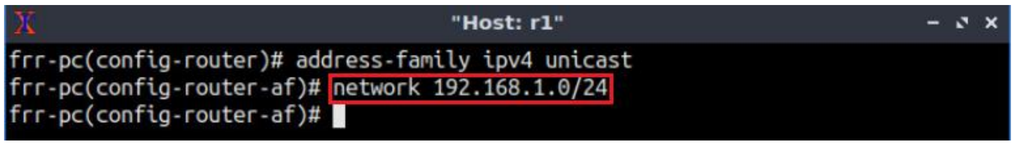


```
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)#
```

Figure 33. Enabling address-family IPv4 configuration mode in router r1.

Step 9. In this step, router r1 will advertise the LAN 192.168.1.0/24 to its BGP neighbor. To do so, issue the following command:

```
network 192.168.1.0/24
```

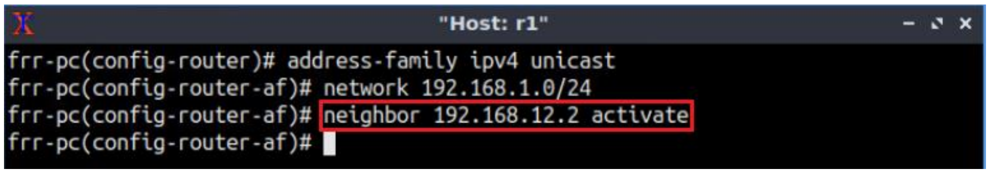


```
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# network 192.168.1.0/24
frr-pc(config-router-af)#
```

Figure 34. Advertising IPv4 LAN in router r1.

Step 10. Type the following command to activate the neighbor 192.168.12.2 so that router r1 uses this neighbor to advertise the IPv4 LAN.

```
neighbor 192.168.12.2 activate
```

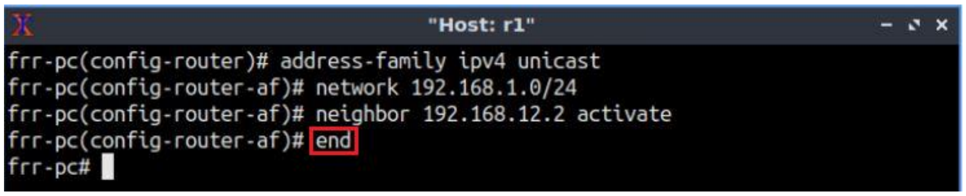


```
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# network 192.168.1.0/24
frr-pc(config-router-af)# neighbor 192.168.12.2 activate
frr-pc(config-router-af)#
```

Figure 35. Activating neighbor to advertise IPv4 network.

Step 11. Type the following command to exit from configuration mode.

```
end
```

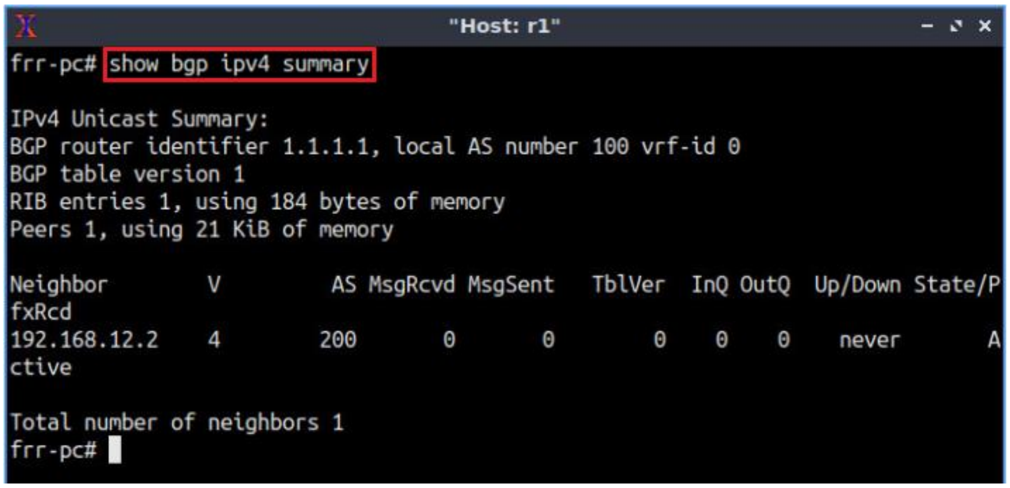


```
"Host: r1"
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# network 192.168.1.0/24
frr-pc(config-router-af)# neighbor 192.168.12.2 activate
frr-pc(config-router-af)# end
frr-pc#
```

Figure 36. Exiting from configuration mode.

Step 12. Type the following command in router r1 to verify IPv4 peering information with router r2. You will notice that BGP connectivity for IPv4 is over an IPv4 BGP transport session, using the neighbor address 192.168.12.2.

```
show bgp ipv4 summary
```

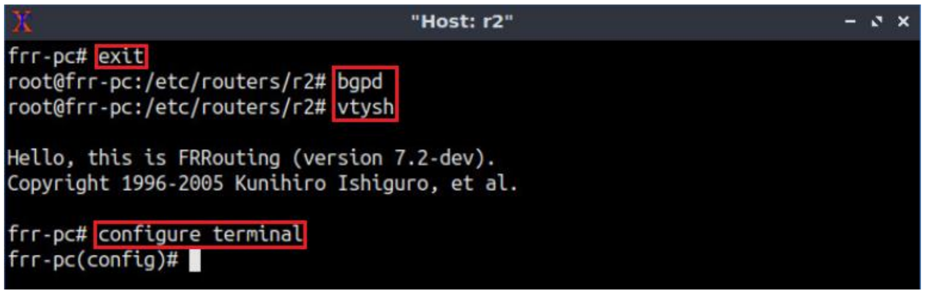


```
"Host: r1"
frr-pc# show bgp ipv4 summary

IPv4 Unicast Summary:
BGP router identifier 1.1.1.1, local AS number 100 vrf-id 0
BGP table version 1
RIB entries 1, using 184 bytes of memory
Peers 1, using 21 KiB of memory

Neighbor        V         AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down State/P
fxRcd
192.168.12.2    4        200       0       0        0    0    0    never       A
ctive

Total number of neighbors 1
frr-pc#
```

Figure 37. Verifying IPv4 BGP summary in router r1.

4.2 Configure and verify EBGP and IBGP in router r2

Step 1. In router r2, exit vtysh session and enable the BGP daemon. Enable router configuration mode to configure BGP in router r2. All the steps are summarized in the following figure.

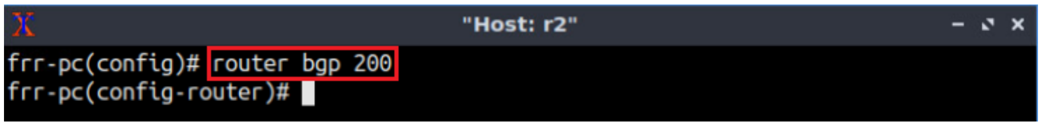


```
"Host: r2"
frr-pc# exit
root@frr-pc:/etc/routers/r2# bgpd
root@frr-pc:/etc/routers/r2# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)#
```

Figure 38. Starting BGP daemon in router r2.

Step 2. The ASN assigned for router r2 is 200. In order to configure BGP, type the following command:

```
router bgp 200
```

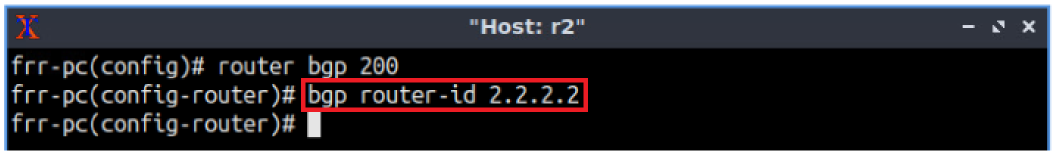


```
frr-pc(config)# router bgp 200
frr-pc(config-router)#
```

Figure 39. Configuring BGP in router r2.

Step 3. Assign a router ID to router r2 by issuing the following command.

```
bgp router-id 2.2.2.2
```

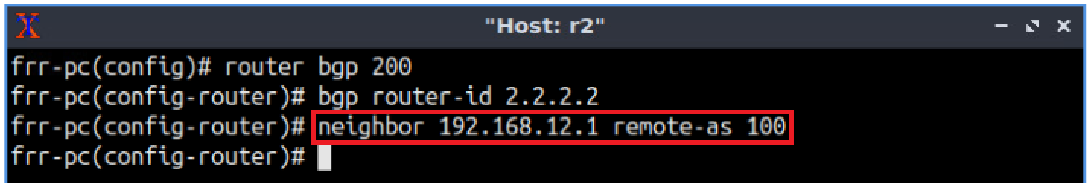


```
frr-pc(config)# router bgp 200
frr-pc(config-router)# bgp router-id 2.2.2.2
frr-pc(config-router)#
```

Figure 40. Assigning a router ID in router r2.

Step 4. To configure EBGP neighbor to router r2 (AS 200), type the command shown below. This command specifies the neighbor IP address (192.168.12.1) and the ASN of the remote BGP peer (AS 100). This neighbor will act as the BGP transport for both IPv4 and IPv6 networks.

```
neighbor 192.168.12.1 remote-as 100
```

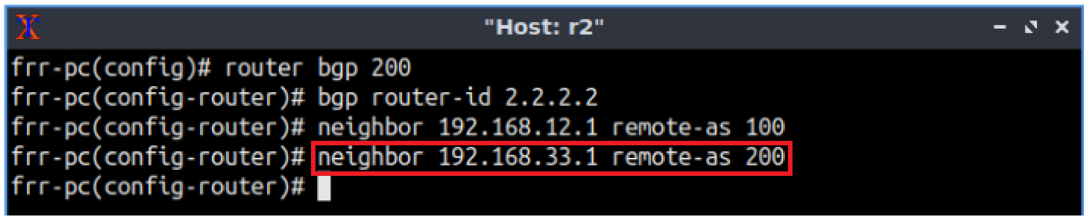


```
frr-pc(config)# router bgp 200
frr-pc(config-router)# bgp router-id 2.2.2.2
frr-pc(config-router)# neighbor 192.168.12.1 remote-as 100
frr-pc(config-router)#
```

Figure 41. Assigning EBGP neighbor to router r2.

Step 5. Type the following command to assign the IPv4 neighbor so that IPv4 network uses IPv4 BGP transport while communicating with router r3. For IBGP peering between router r2 and router r3, assign the loopback address of router r3 as the neighbor of router r2.

```
neighbor 192.168.33.1 remote-as 200
```

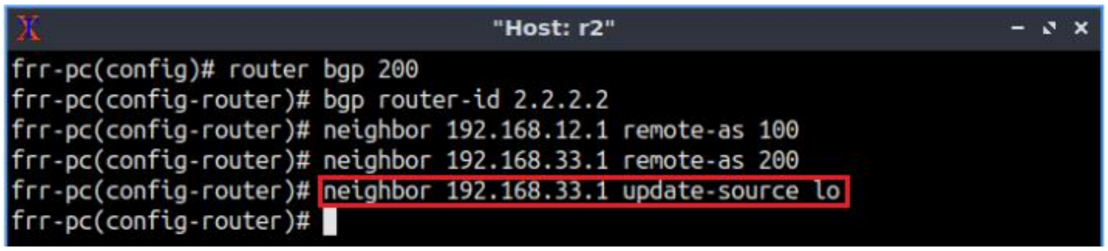


```
frr-pc(config)# router bgp 200
frr-pc(config-router)# bgp router-id 2.2.2.2
frr-pc(config-router)# neighbor 192.168.12.1 remote-as 100
frr-pc(config-router)# neighbor 192.168.33.1 remote-as 200
frr-pc(config-router)#
```

Figure 42. Assigning IBGP neighbor to router r2 for IPv4 network.

Step 6. Type the following command to assign *lo* as the source IP in router r2.

```
neighbor 192.168.33.1 update-source lo
```

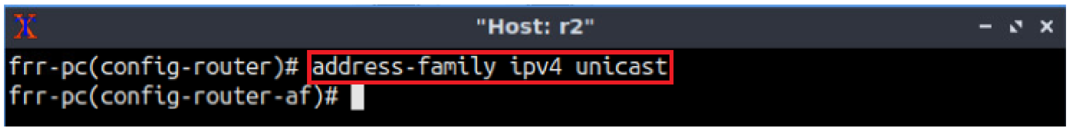


```
frr-pc(config)# router bgp 200
frr-pc(config-router)# bgp router-id 2.2.2.2
frr-pc(config-router)# neighbor 192.168.12.1 remote-as 100
frr-pc(config-router)# neighbor 192.168.33.1 remote-as 200
frr-pc(config-router)# neighbor 192.168.33.1 update-source lo
frr-pc(config-router)#
```

Figure 43. Assigning loopback as source IP for the neighbor 192.168.33.1.

Step 7. Type the following command to enter address-family mode where you can configure routing sessions that use standard IPv4 address prefixes.

```
address-family ipv4 unicast
```

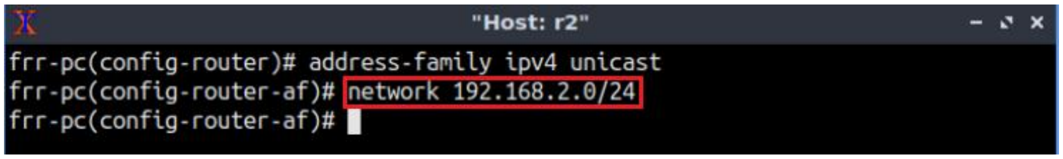


```
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)#
```

Figure 44. Enabling address-family IPv4 configuration mode in router r2.

Step 8. In this step, router r2 will advertise the LAN 192.168.2.0/24 to its BGP peers. To do so, issue the following command:

```
network 192.168.2.0/24
```

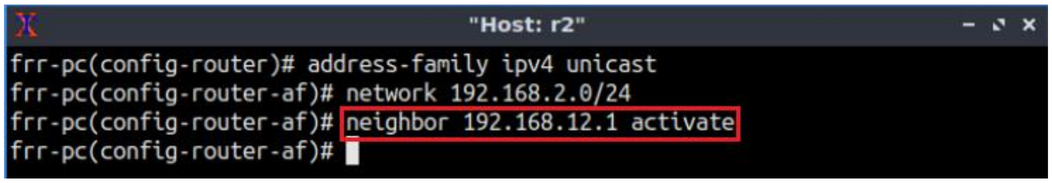


```
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# network 192.168.2.0/24
frr-pc(config-router-af)#
```

Figure 45. Advertising IPv4 LAN in router r2.

Step 9. Type the following command to activate the neighbor 192.168.12.1 so that this neighbor is used to exchange IPv4 routes with router r1.

```
neighbor 192.168.12.1 activate
```

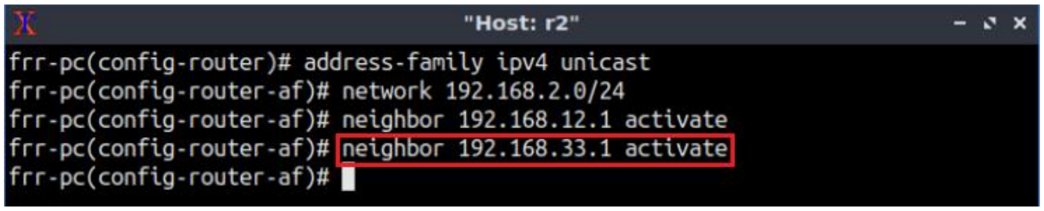


```
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# network 192.168.2.0/24
frr-pc(config-router-af)# neighbor 192.168.12.1 activate
frr-pc(config-router-af)#
```

Figure 46. Activating EBGP neighbor to advertise IPv4 network.

Step 10. Type the following command to activate the neighbor 192.168.33.1 so that this neighbor is used to exchange IPv4 routes with router r3.

```
neighbor 192.168.33.1 activate
```

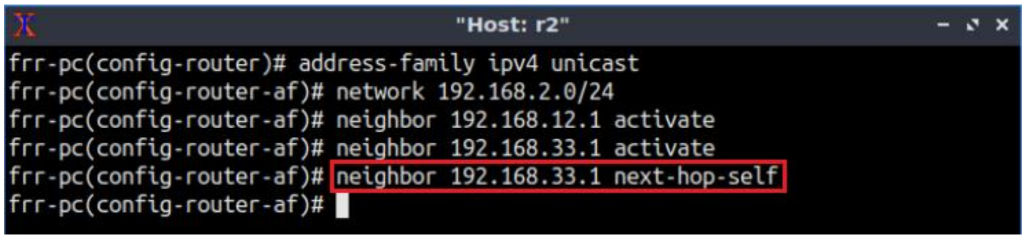


```
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# network 192.168.2.0/24
frr-pc(config-router-af)# neighbor 192.168.12.1 activate
frr-pc(config-router-af)# neighbor 192.168.33.1 activate
frr-pc(config-router-af)#
```

Figure 47. Activating IBGP neighbor to advertise IPv4 network.

Step 11. Type the following command in router r2 so that the interface lo is used as the next hop address of router r2. It will allow router r3 to receive the route to router r1 since the next hop address (192.168.22.1) is known to router r3.

```
neighbor 192.168.33.1 next-hop-self
```

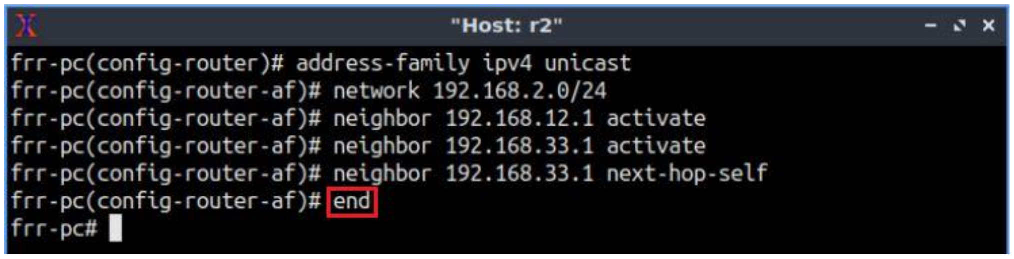


```
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# network 192.168.2.0/24
frr-pc(config-router-af)# neighbor 192.168.12.1 activate
frr-pc(config-router-af)# neighbor 192.168.33.1 activate
frr-pc(config-router-af)# neighbor 192.168.33.1 next-hop-self
frr-pc(config-router-af)#
```

Figure 48. Assigning next hop address in router r2.

Step 12. Type the following command to exit from configuration mode.

```
end
```

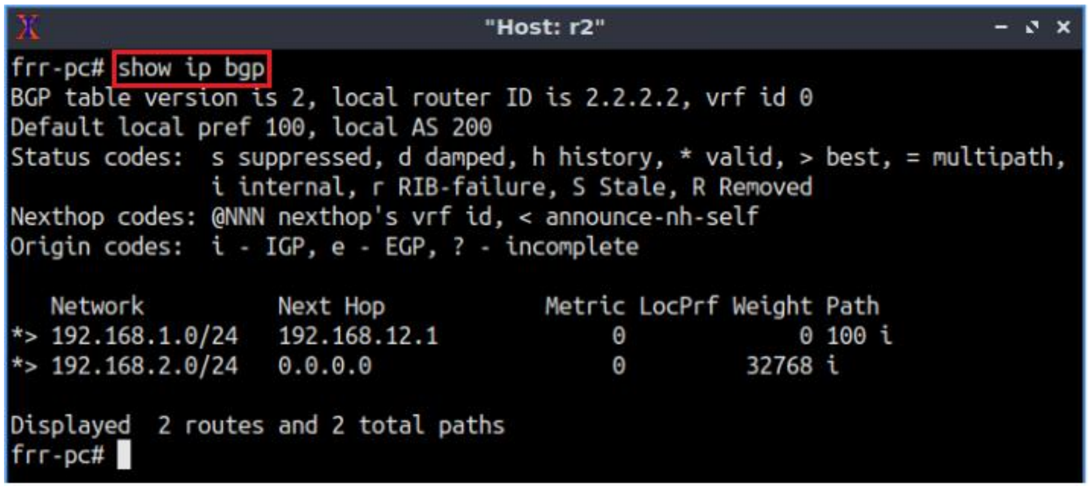


```
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# network 192.168.2.0/24
frr-pc(config-router-af)# neighbor 192.168.12.1 activate
frr-pc(config-router-af)# neighbor 192.168.33.1 activate
frr-pc(config-router-af)# neighbor 192.168.33.1 next-hop-self
frr-pc(config-router-af)# end
frr-pc#
```

Figure 49. Exiting from configuration mode.

Step 13. Type the following command to verify BGP networks. You will observe the LAN network 192.168.1.0/24 advertised by router r1.

```
show ip bgp
```



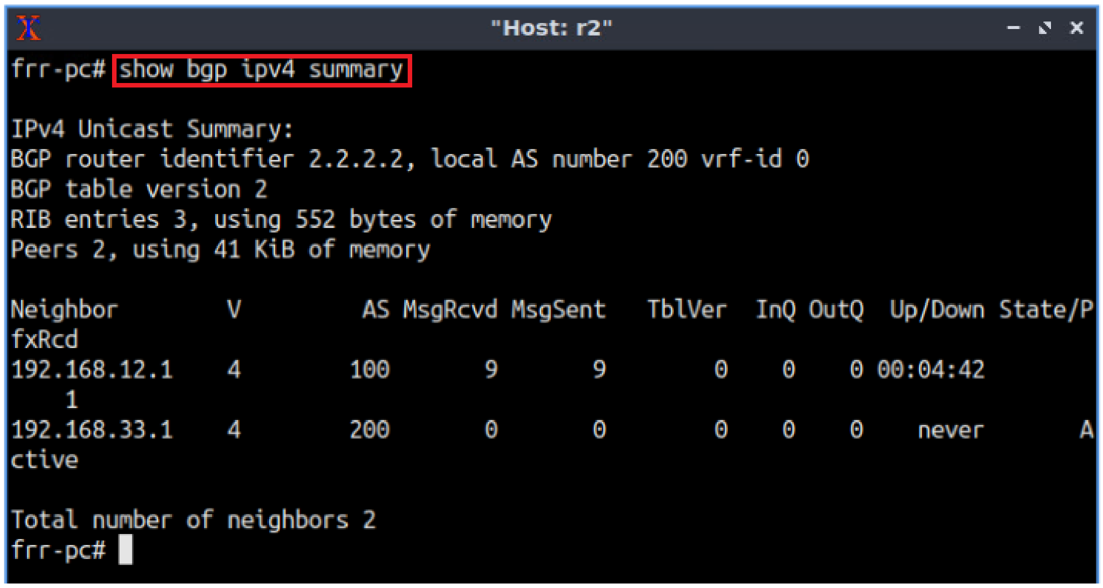
```
"Host: r2"
frr-pc# show ip bgp
BGP table version is 2, local router ID is 2.2.2.2, vrf id 0
Default local pref 100, local AS 200
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 192.168.1.0/24   192.168.12.1             0             0 100 i
*> 192.168.2.0/24   0.0.0.0                  0         32768 i

Displayed  2 routes and 2 total paths
frr-pc# 
```

Figure 50. Verifying BGP networks in router r2.

Consider the figure above. You will observe the LAN network 192.168.1.0/24 advertised by router r1.

Step 14. Type the following command in router r2 to verify IPv4 peering information with routers r1 and r3.

```
show bgp ipv4 summary
```



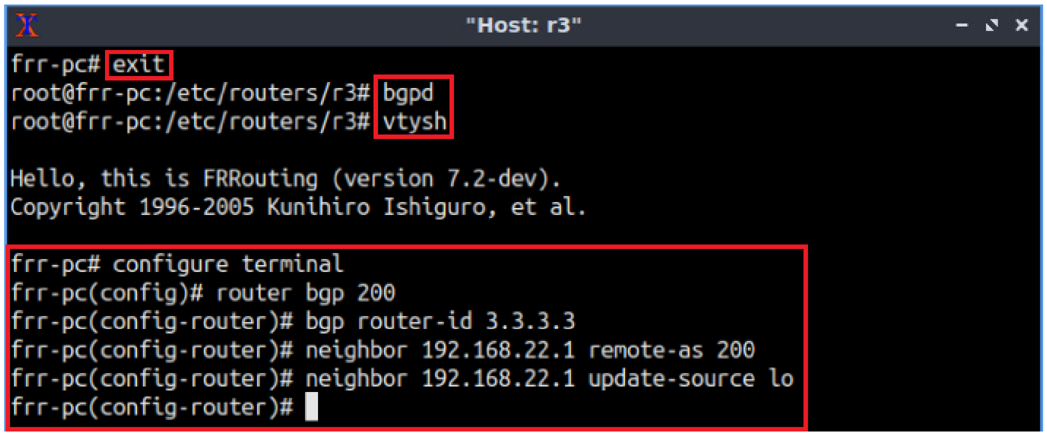
```
"Host: r2"
frr-pc# show bgp ipv4 summary

IPv4 Unicast Summary:
BGP router identifier 2.2.2.2, local AS number 200 vrf-id 0
BGP table version 2
RIB entries 3, using 552 bytes of memory
Peers 2, using 41 KiB of memory

Neighbor        V         AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down State/P
fxRcd
192.168.12.1    4        100       9       9        0    0    0 00:04:42
    1
192.168.33.1    4        200       0       0        0    0    0    never       A
ctive

Total number of neighbors 2
frr-pc# 
```

Figure 51. Verifying IPv4 BGP summary in router r2.

Consider the figure above. You will notice that BGP connectivity for IPv4 is over IPv4 BGP transport session, using the neighbor address 192.168.12.1 and 192.168.33.1 respectively.

4.3 Configure and verify IBGP in router r3

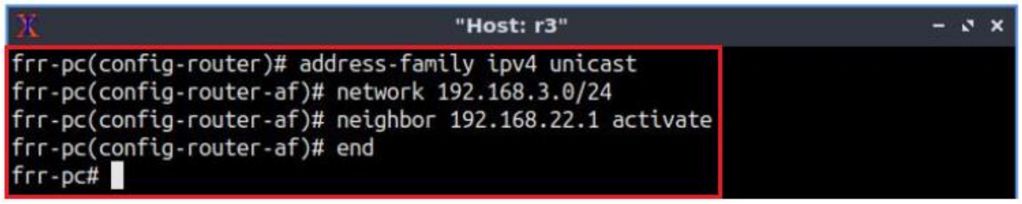
Step 1. Router r3 is configured similarly to router r2 but, with different metrics in order to establish IBGP peering with router r2. All the steps are summarized in the following figure.



```
frr-pc# exit
root@frr-pc:/etc/routers/r3# bgpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 200
frr-pc(config-router)# bgp router-id 3.3.3.3
frr-pc(config-router)# neighbor 192.168.22.1 remote-as 200
frr-pc(config-router)# neighbor 192.168.22.1 update-source lo
frr-pc(config-router)#
```

Figure 52. Configuring BGP in router r3.

Step 2. Configure IPv4 address-family in router r3. There is no need to assign the next hop when configuring BGP, since router r3 is not participating in any EBGP session. All the steps are summarized in the following figure.

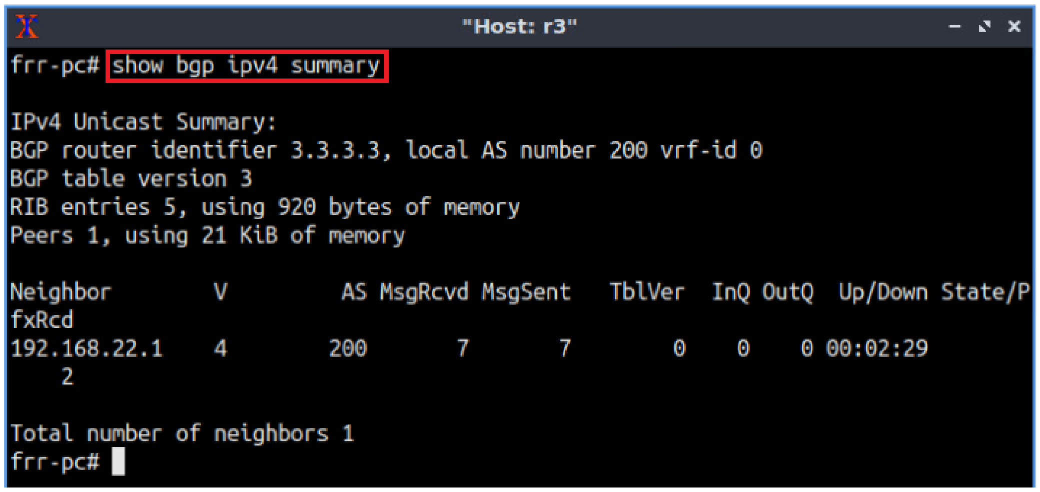


```
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# network 192.168.3.0/24
frr-pc(config-router-af)# neighbor 192.168.22.1 activate
frr-pc(config-router-af)# end
frr-pc#
```

Figure 53. Configuring IPv4 address-family in router r3.

Step 3. Type the following command in router r3 to verify IPv4 peering information with router r2. You will notice that BGP connectivity for IPv4 is over an IPv4 BGP transport session, using the neighbor address 192.168.22.1.

```
show bgp ipv4 summary
```



```
frr-pc# show bgp ipv4 summary

IPv4 Unicast Summary:
BGP router identifier 3.3.3.3, local AS number 200 vrf-id 0
BGP table version 3
RIB entries 5, using 920 bytes of memory
Peers 1, using 21 KiB of memory

Neighbor        V         AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down State/P
fxRcd
192.168.22.1    4        200       7       7        0    0    0 00:02:29
    2

Total number of neighbors 1
frr-pc#
```

Figure 54. Verifying IPv4 BGP summary in router r3.

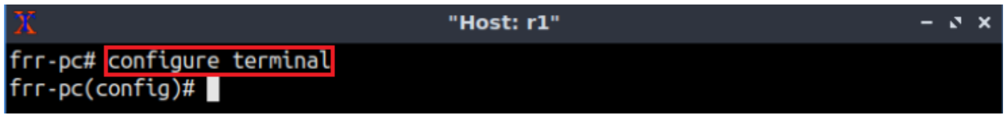
5 Configure and verify BGP for IPv6 networks

In this section, you will configure EBGP so that router r1 uses IPv4 as the BGP transport for IPv6 sessions. Create a route-map next-hop-IPv6 to attach to the BGP neighbor in the outbound direction so that the next-hop parameter overwrites with the appropriate IPv6 next-hop address. For IBGP, you will configure router r2 so that IPv6 routing information is transported by IPv6 TCP sessions.

5.1 Configure and verify EBGP in router r1

Step 1. To enable router r1 into configuration mode, issue the following command:

```
configure terminal
```

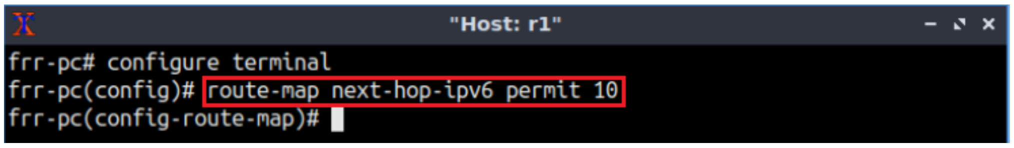


```
frr-pc# configure terminal
frr-pc(config)#
```

Figure 55. Enabling configuration mode in router r1.

Step 2. Type the following command to create a route-map named next-hop-IPv6 with permit clause. The permit clause will allow BGP to use the route map policy. The sequence number allows the identification and editing of multiple statements. You will use default sequence number which is 10. You will be entering the configuration mode where you can set the route-map policy.

```
route-map next-hop-ipv6 permit 10
```

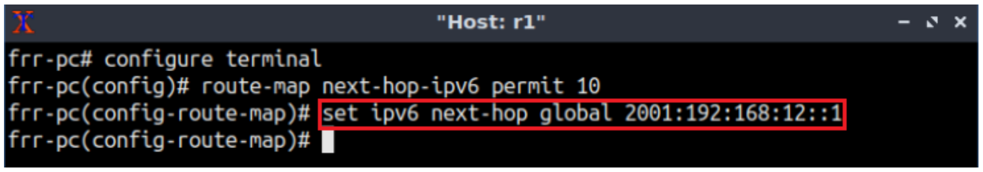


```
frr-pc# configure terminal
frr-pc(config)# route-map next-hop-ipv6 permit 10
frr-pc(config-route-map)#
```

Figure 56. Creating next-hop-IPv6 route-map.

Step 3. Type the following command to set the route-map policy. As the IPv6 address will be transported by IPv4 TCP session, you need an IPv6 address so that the next-hop parameter overwrites with the appropriate IPv6 next-hop address. By the route-map policy, you will set the global IPv6 address 2001:192:168:12::1 for router r1 which will be the next hop address for router r2 and the link local address will appear as the next hop address in the BGP table of router r2.

```
set ipv6 next-hop global 2001:192:168:12::1
```

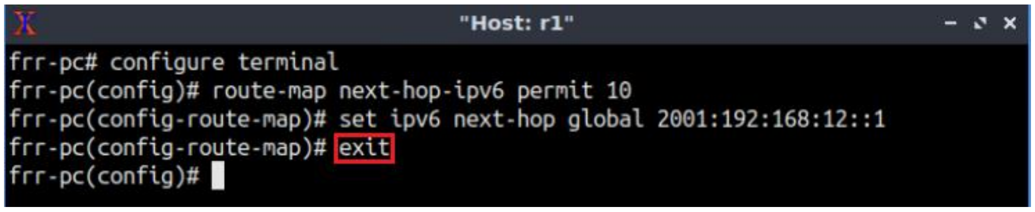


```
frr-pc# configure terminal
frr-pc(config)# route-map next-hop-ipv6 permit 10
frr-pc(config-route-map)# set ipv6 next-hop global 2001:192:168:12::1
frr-pc(config-route-map)#
```

Figure 57. Setting route-map policy in router r1.

Step 4. Type the following command to exit from the configuration mode.

```
exit
```

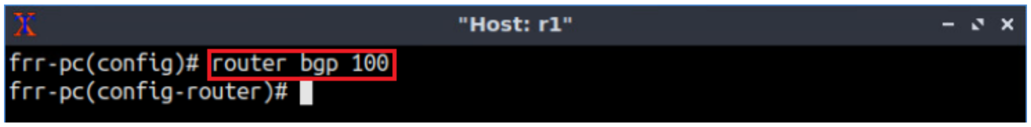


```
"Host: r1"
frr-pc# configure terminal
frr-pc(config)# route-map next-hop-ipv6 permit 10
frr-pc(config-route-map)# set ipv6 next-hop global 2001:192:168:12::1
frr-pc(config-route-map)# exit
frr-pc(config)#
```

Figure 58. Exiting from route-map mode.

Step 5. The ASN assigned for router r1 is 100. In order to apply the configuration, type the following command:

```
router bgp 100
```

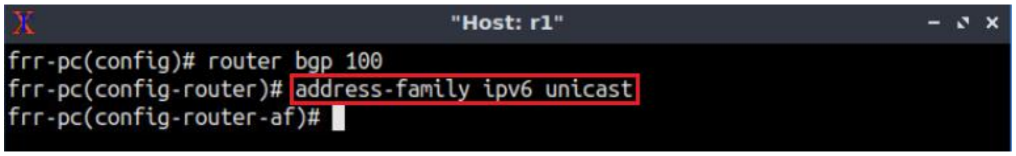


```
"Host: r1"
frr-pc(config)# router bgp 100
frr-pc(config-router)#
```

Figure 59. Configuring BGP in router r1.

Step 6. Type the following command to enter address-family mode where you can configure routing sessions that use standard IPv6 address prefixes.

```
address-family ipv6 unicast
```

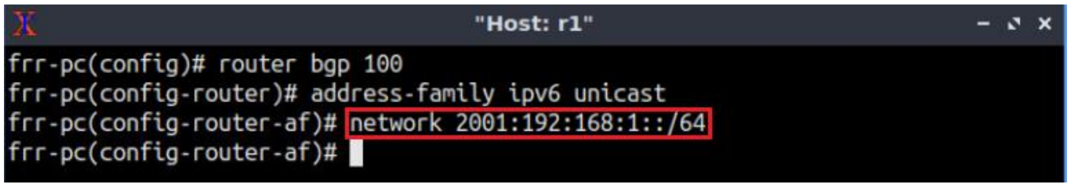


```
"Host: r1"
frr-pc(config)# router bgp 100
frr-pc(config-router)# address-family ipv6 unicast
frr-pc(config-router-af)#
```

Figure 60. Enabling address-family IPv6 configuration mode in router r1.

Step 7. In this step, router r1 will advertise the IPv6 LAN 2001:192:168:1::/64 to its BGP peers. To do so, issue the following command:

```
network 2001:192:168:1::/64
```

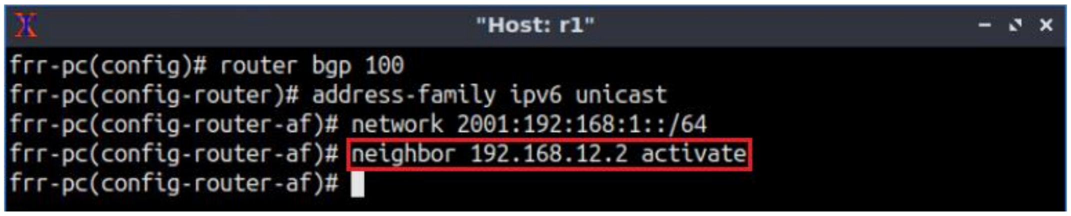


```
"Host: r1"
frr-pc(config)# router bgp 100
frr-pc(config-router)# address-family ipv6 unicast
frr-pc(config-router-af)# network 2001:192:168:1::/64
frr-pc(config-router-af)#
```

Figure 61. Advertising IPv6 LAN in router r1.

Step 8. Since you are using IPv4 neighbor as BGP transport, you will activate the IPv4 neighbor within the IPv6 address-family. To do so, type the following command.

```
neighbor 192.168.12.2 activate
```

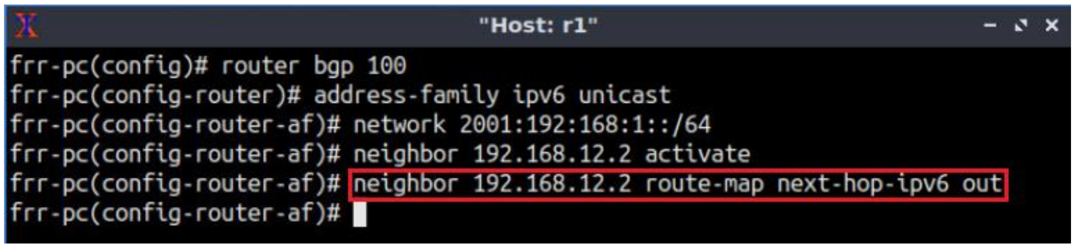


```
frr-pc(config)# router bgp 100
frr-pc(config-router)# address-family ipv6 unicast
frr-pc(config-router-af)# network 2001:192:168:1::/64
frr-pc(config-router-af)# neighbor 192.168.12.2 activate
frr-pc(config-router-af)#
```

Figure 62. Activating neighbor to advertise IPv6 network.

Step 9. Type the following command to attach the route-map to BGP neighbor in the outbound direction. Outbound (`out`) direction means that this information in the route-map will be applied to IPv6 BGP updates whenever they are sent to router r2. In the BGP table of router r2, 2001:192:168:12::1 will be used as next-hop address. The next-hop address will be the link local address of 2001:192:168:12::1 because link local addresses are used as next-hop addresses by default in FRR.

```
neighbor 192.168.12.2 route-map next-hop-ipv6 out
```

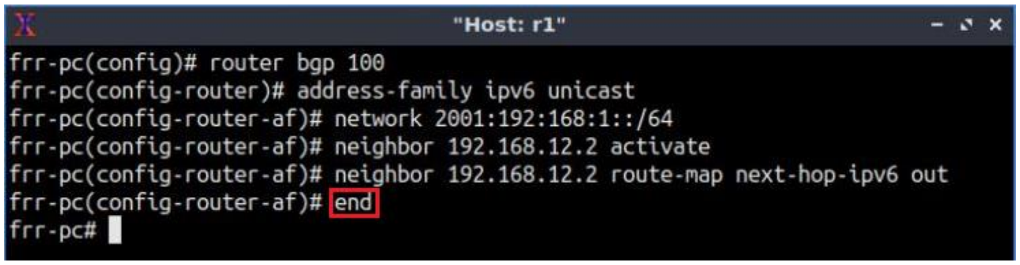


```
frr-pc(config)# router bgp 100
frr-pc(config-router)# address-family ipv6 unicast
frr-pc(config-router-af)# network 2001:192:168:1::/64
frr-pc(config-router-af)# neighbor 192.168.12.2 activate
frr-pc(config-router-af)# neighbor 192.168.12.2 route-map next-hop-ipv6 out
frr-pc(config-router-af)#
```

Figure 63. Attaching the route-map to BGP neighbor.

Step 10. Type the following command to exit from configuration mode.

```
end
```

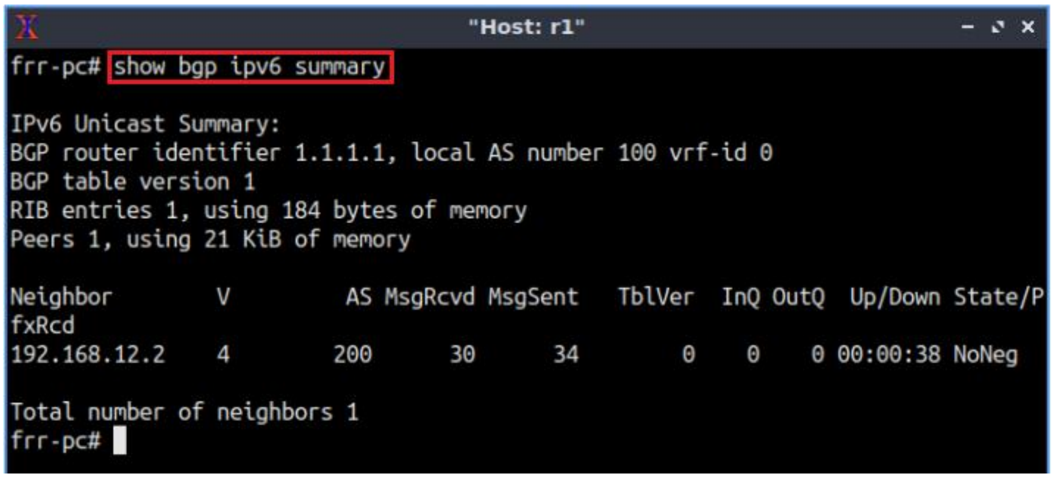


```
frr-pc(config)# router bgp 100
frr-pc(config-router)# address-family ipv6 unicast
frr-pc(config-router-af)# network 2001:192:168:1::/64
frr-pc(config-router-af)# neighbor 192.168.12.2 activate
frr-pc(config-router-af)# neighbor 192.168.12.2 route-map next-hop-ipv6 out
frr-pc(config-router-af)# end
frr-pc#
```

Figure 64. Exiting from configuration mode.

Step 11. Type the following command in router r1 to verify IPv6 peering information with router r2. You will notice that BGP connectivity for IPv6 is over an IPv4 BGP transport session, using the neighbor address 192.168.12.2.

```
show bgp ipv6 summary
```

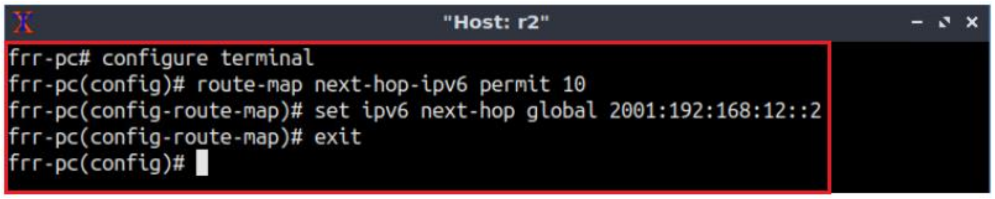


```
"Host: r1"
frr-pc# show bgp ipv6 summary

IPv6 Unicast Summary:
BGP router identifier 1.1.1.1, local AS number 100 vrf-id 0
BGP table version 1
RIB entries 1, using 184 bytes of memory
Peers 1, using 21 KiB of memory

Neighbor        V         AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down State/P
fxRcd
192.168.12.2    4        200      30      34        0    0    0 00:00:38 NoNeg

Total number of neighbors 1
frr-pc#
```

Figure 65. Verifying IPv6 BGP summary in router r1.

5.2 Configure and verify EBGP and IBGP in router r2

Step 1. Enable BGP daemon and create a route-map so that you can attach the route-map to the BGP neighbor of router r2. All the steps are summarized in the figure below.

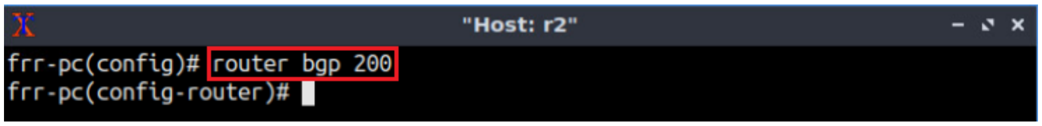


```
"Host: r2"
frr-pc# configure terminal
frr-pc(config)# route-map next-hop-ipv6 permit 10
frr-pc(config-route-map)# set ipv6 next-hop global 2001:192:168:12::2
frr-pc(config-route-map)# exit
frr-pc(config)#
```

Figure 66. Creating next-hop-IPv6 route-map in router r2.

Step 2. The ASN assigned for router r2 is 200. In order to configure BGP, type the following command:

```
router bgp 200
```

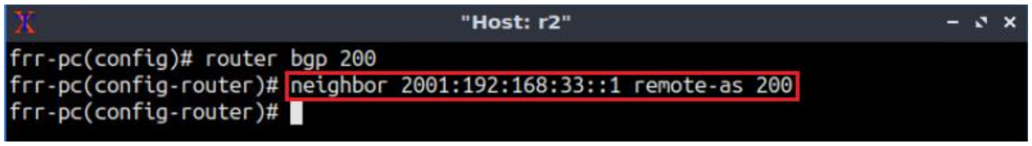


```
"Host: r2"
frr-pc(config)# router bgp 200
frr-pc(config-router)#
```

Figure 67. Configuring BGP in router r2.

Step 3. In this step, you will configure IBGP neighbor to router r2. Type the following command to assign the IPv6 neighbor so that IPv6 network uses IPv6 BGP transport. For IBGP peering between router r2 and router r3, assign the loopback address of router r3 as the neighbor of router r2.

```
neighbor 2001:192:168:33::1 remote-as 200
```

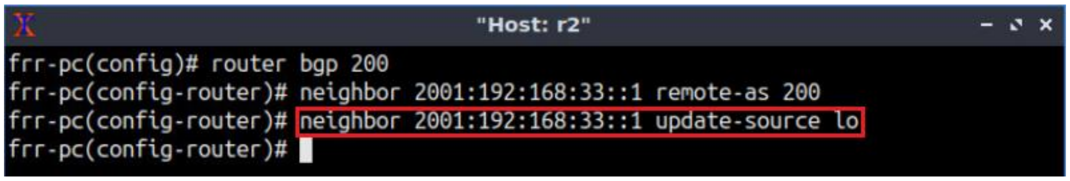


```
"Host: r2"
frr-pc(config)# router bgp 200
frr-pc(config-router)# neighbor 2001:192:168:33::1 remote-as 200
frr-pc(config-router)#
```

Figure 68. Assigning IBGP neighbor to router r2 for IPv6 network.

Step 4. In BGP, the source IP address of BGP packets sent by the router must be the same as neighbor IP address set on the neighboring router. As you are assigning the loopback as neighbor address, you must use loopback address as the source of BGP packets sent to the neighbor. Type the following command to assign *lo* as source IP in router r2.

```
neighbor 2001:192:168:33::1 update-source lo
```

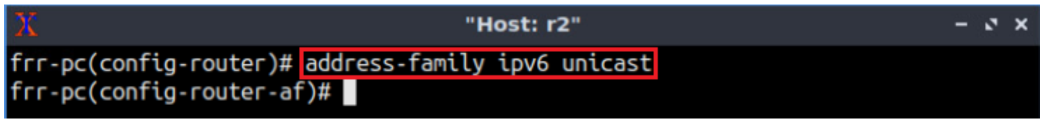


```
"Host: r2"
frr-pc(config)# router bgp 200
frr-pc(config-router)# neighbor 2001:192:168:33::1 remote-as 200
frr-pc(config-router)# neighbor 2001:192:168:33::1 update-source lo
frr-pc(config-router)#
```

Figure 69. Assigning loopback as source IP for the neighbor 2001:192:168:33::1.

Step 5. Type the following command to enter address-family mode where you can configure routing sessions that use standard IPv6 address prefixes.

```
address-family ipv6 unicast
```

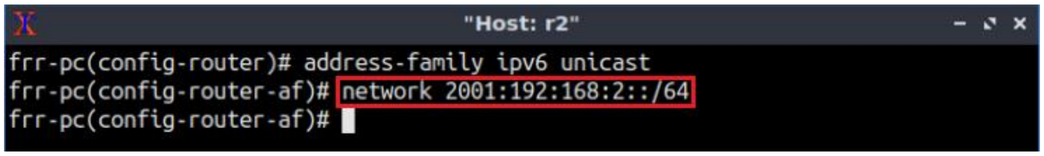


```
"Host: r2"
frr-pc(config-router)# address-family ipv6 unicast
frr-pc(config-router-af)#
```

Figure 70. Enabling address-family IPv6 configuration mode in router r2.

Step 6. In this step, router r2 will advertise the LAN 2001:192:168:2::/64 to its BGP peers. To do so, issue the following command:

```
network 2001:192:168:2::/64
```

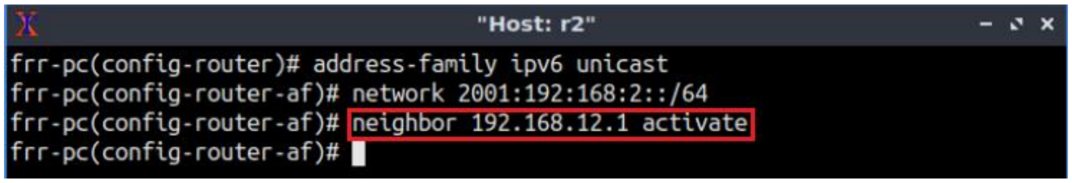


```
"Host: r2"
frr-pc(config-router)# address-family ipv6 unicast
frr-pc(config-router-af)# network 2001:192:168:2::/64
frr-pc(config-router-af)#
```

Figure 71. Advertising IPv6 LAN in router r2.

Step 7. Type the following command to activate the neighbor 192.168.12.1 so that router r2 uses this neighbor to exchange IPv6 routes with router r1.

```
neighbor 192.168.12.1 activate
```

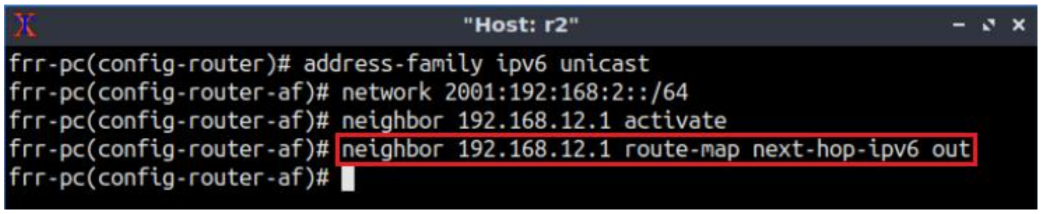


```
"Host: r2"
frr-pc(config-router)# address-family ipv6 unicast
frr-pc(config-router-af)# network 2001:192:168:2::/64
frr-pc(config-router-af)# neighbor 192.168.12.1 activate
frr-pc(config-router-af)#
```

Figure 72. Activating neighbor to advertise IPv6 network.

Step 8. Type the following command to attach the route-map to BGP neighbor in the outbound direction. Outbound direction means that this information in the route-map will be applied to IPv6 BGP updates as they are sent to router r1. In the BGP table of router r1, 2001:192:168:12::2 will be used as next-hop address. The next-hop address will be the link local address of 2001:192:168:12::2 because FRR always uses link local address as next-hop address.

```
neighbor 192.168.12.1 route-map next-hop-ipv6 out
```

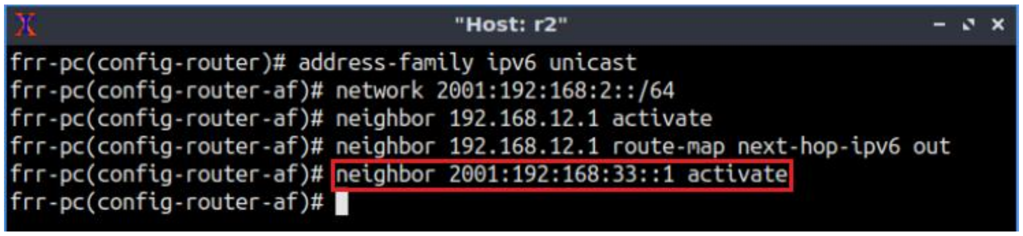


```
frr-pc(config-router)# address-family ipv6 unicast
frr-pc(config-router-af)# network 2001:192:168:2::/64
frr-pc(config-router-af)# neighbor 192.168.12.1 activate
frr-pc(config-router-af)# neighbor 192.168.12.1 route-map next-hop-ipv6 out
frr-pc(config-router-af)#
```

Figure 73. Attaching the route-map to BGP neighbor of router r2.

Step 9. Type the following command to activate the neighbor 2001:192:168:33::1 so that router r2 uses this neighbor to exchange IPv6 routes with router r3.

```
neighbor 2001:192:168:33::1 activate
```

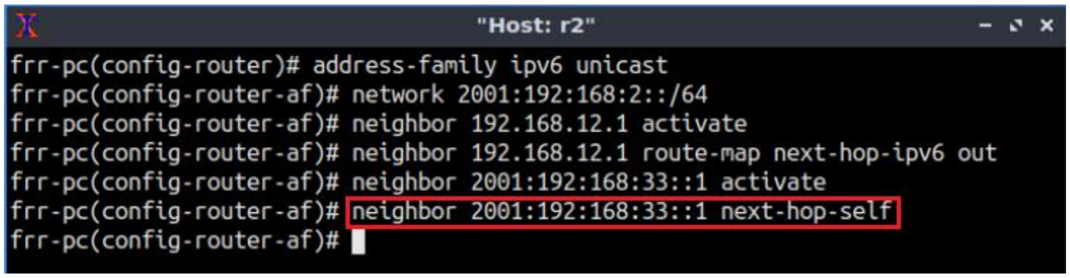


```
frr-pc(config-router)# address-family ipv6 unicast
frr-pc(config-router-af)# network 2001:192:168:2::/64
frr-pc(config-router-af)# neighbor 192.168.12.1 activate
frr-pc(config-router-af)# neighbor 192.168.12.1 route-map next-hop-ipv6 out
frr-pc(config-router-af)# neighbor 2001:192:168:33::1 activate
frr-pc(config-router-af)#
```

Figure 74. Activating neighbor to advertise IPv6 network.

Step 10. Type the following command in router r2 so that the interface lo is used as the next hop address of router r2. It will allow router r3 to receive the route to router r1 as the next hop address (2001:192:168:22::1) is known to router r3.

```
neighbor 2001:192:168:33::1 next-hop-self
```

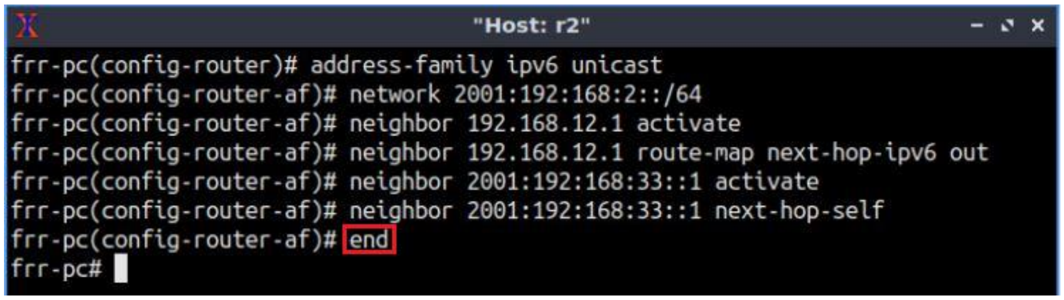


```
frr-pc(config-router)# address-family ipv6 unicast
frr-pc(config-router-af)# network 2001:192:168:2::/64
frr-pc(config-router-af)# neighbor 192.168.12.1 activate
frr-pc(config-router-af)# neighbor 192.168.12.1 route-map next-hop-ipv6 out
frr-pc(config-router-af)# neighbor 2001:192:168:33::1 activate
frr-pc(config-router-af)# neighbor 2001:192:168:33::1 next-hop-self
frr-pc(config-router-af)#
```

Figure 75. Assigning next hop address in router r2.

Step 11. Type the following command to exit from configuration mode.

```
end
```

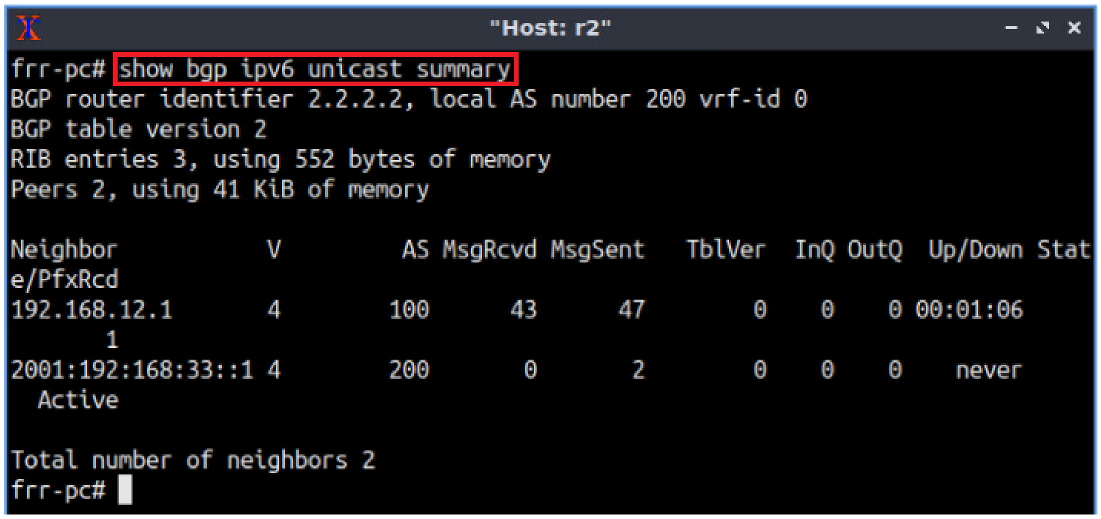


```
frr-pc(config-router)# address-family ipv6 unicast
frr-pc(config-router-af)# network 2001:192:168:2::/64
frr-pc(config-router-af)# neighbor 192.168.12.1 activate
frr-pc(config-router-af)# neighbor 192.168.12.1 route-map next-hop-ipv6 out
frr-pc(config-router-af)# neighbor 2001:192:168:33::1 activate
frr-pc(config-router-af)# neighbor 2001:192:168:33::1 next-hop-self
frr-pc(config-router-af)# end
frr-pc#
```

Figure 76. Exiting from configuration mode.

Step 12. Type the following command in router r2 to verify IPv6 neighbors.

```
show bgp ipv6 unicast summary
```



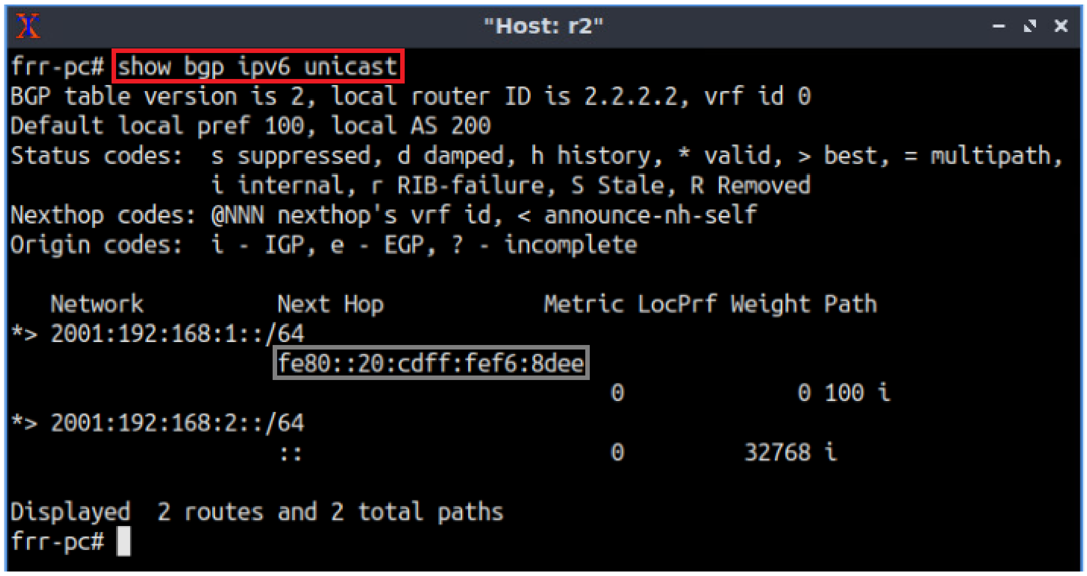
```
frr-pc# show bgp ipv6 unicast summary
BGP router identifier 2.2.2.2, local AS number 200 vrf-id 0
BGP table version 2
RIB entries 3, using 552 bytes of memory
Peers 2, using 41 KiB of memory

Neighbor        V         AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down Stat
e/PfxRcd
192.168.12.1    4        100      43      47        0    0    0 00:01:06
       1
2001:192:168:33::1 4     200       0       2        0    0    0    never
  Active

Total number of neighbors 2
frr-pc#
```

Figure 77. Verifying IPv6 BGP neighbors in router r2.

Consider the figure above. The BGP table shows that router r2 communicates with router r1 through IPv4 neighbor (192.168.12.1) and communicates with router r3 via IPv6 neighbor (2001:192:168:33::1).

Step 13. Type the following command in router r2 to verify IPv6 routes. You will notice the link local address of 2001:192:168:12::2 as the next hop address for the network 2001:192:168:1::/64 which was used in the route-map.

```
show bgp ipv6 unicast
```

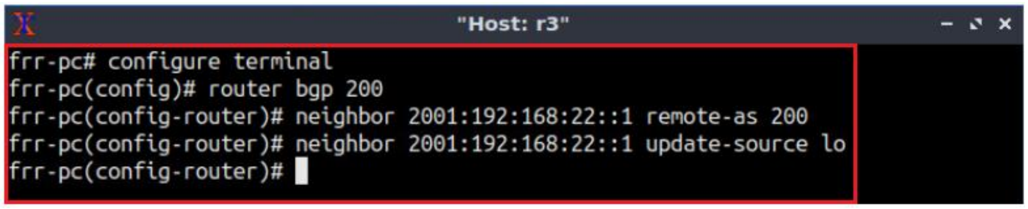


```
frr-pc# show bgp ipv6 unicast
BGP table version is 2, local router ID is 2.2.2.2, vrf id 0
Default local pref 100, local AS 200
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 2001:192:168:1::/64
                    fe80::20:cdff:fef6:8dee
                                             0             0 100 i
*> 2001:192:168:2::/64
                    ::                       0         32768 i

Displayed  2 routes and 2 total paths
frr-pc#
```

Figure 78. Verifying IPv6 routes in router r2.

5.3 Configure and verify IBGP in router r3

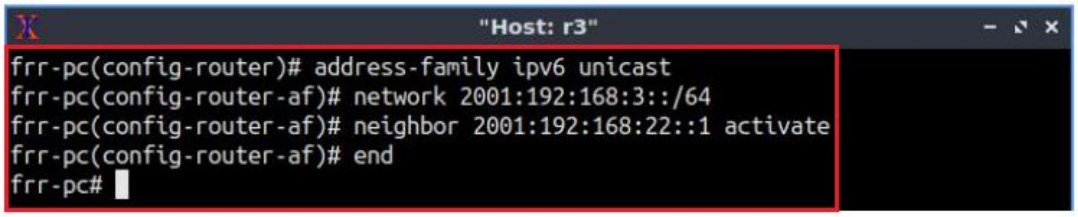
Step 1. Router r3 is configured similarly to router r2 but, with different metrics in order to establish IBGP peering with router r2. All the steps are summarized in the following figure.



```
frr-pc# configure terminal
frr-pc(config)# router bgp 200
frr-pc(config-router)# neighbor 2001:192:168:22::1 remote-as 200
frr-pc(config-router)# neighbor 2001:192:168:22::1 update-source lo
frr-pc(config-router)#
```

Figure 79. Configuring BGP in router r3.

Step 2. Configure IPv6 address-family in router r3. All the steps are summarized in the following figure.

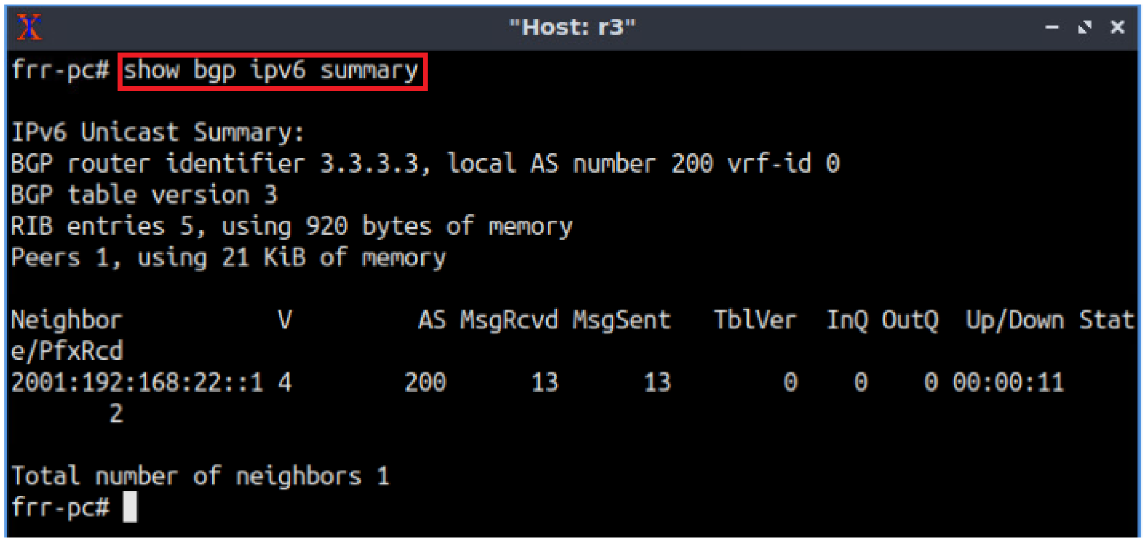


```
frr-pc(config-router)# address-family ipv6 unicast
frr-pc(config-router-af)# network 2001:192:168:3::/64
frr-pc(config-router-af)# neighbor 2001:192:168:22::1 activate
frr-pc(config-router-af)# end
frr-pc#
```

Figure 80. Configuring IPv6 address-family in router r3.

Step 3. Type the following command in router r3 to verify IPv6 neighbors. The BGP table shows that router r3 communicates with router r2 through IPv6 neighbor (2001:192:168:22::1).

```
show bgp ipv6 summary
```



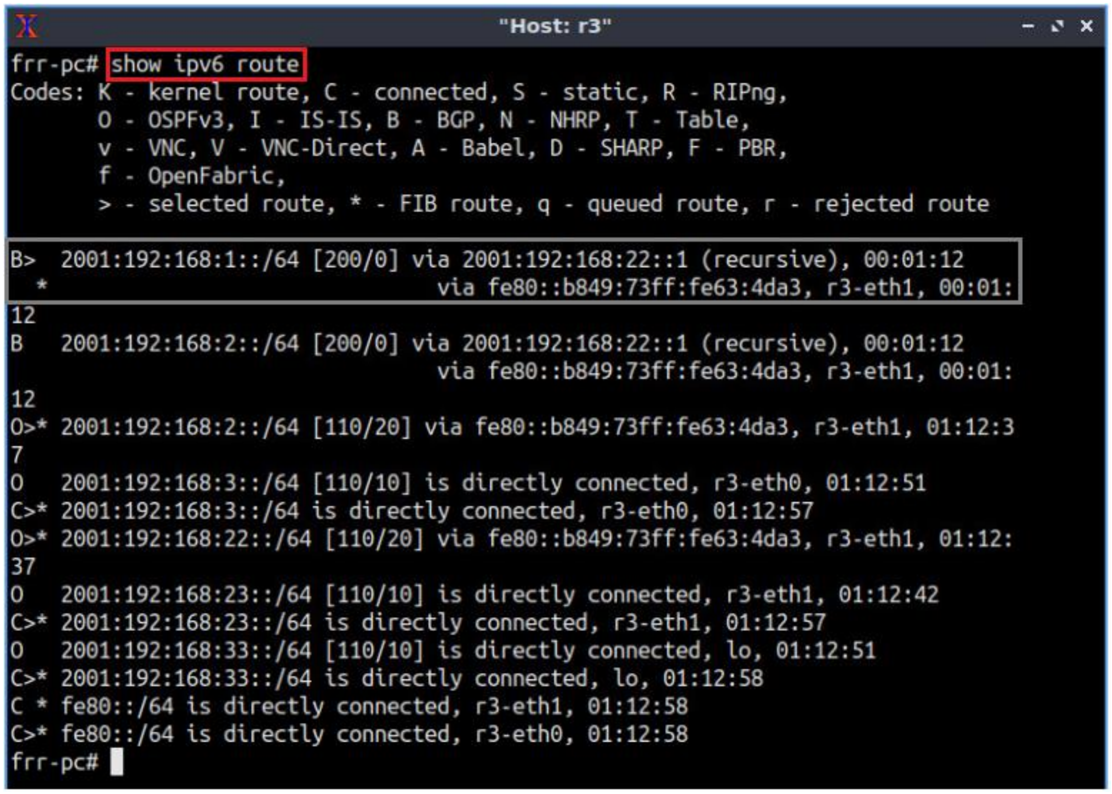
```
frr-pc# show bgp ipv6 summary

IPv6 Unicast Summary:
BGP router identifier 3.3.3.3, local AS number 200 vrf-id 0
BGP table version 3
RIB entries 5, using 920 bytes of memory
Peers 1, using 21 KiB of memory

Neighbor            V         AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down Stat
e/PfxRcd
2001:192:168:22::1 4        200      13      13        0    0    0 00:00:11
       2

Total number of neighbors 1
frr-pc#
```

Figure 81. Verifying IPv6 BGP neighbors in router r3.

6 Verify BGP configuration

Step 1. Type the following command to verify the routing table of router r3.

```
show ipv6 route
```



```
frr-pc# show ipv6 route
Codes: K - kernel route, C - connected, S - static, R - RIPng,
       O - OSPFv3, I - IS-IS, B - BGP, N - NHRP, T - Table,
       v - VNC, V - VNC-Direct, A - Babel, D - SHARP, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

B>  2001:192:168:1::/64 [200/0] via 2001:192:168:22::1 (recursive), 00:01:12
  *                              via fe80::b849:73ff:fe63:4da3, r3-eth1, 00:01:
12
B   2001:192:168:2::/64 [200/0] via 2001:192:168:22::1 (recursive), 00:01:12
                                 via fe80::b849:73ff:fe63:4da3, r3-eth1, 00:01:
12
O>* 2001:192:168:2::/64 [110/20] via fe80::b849:73ff:fe63:4da3, r3-eth1, 01:12:3
7
O   2001:192:168:3::/64 [110/10] is directly connected, r3-eth0, 01:12:51
C>* 2001:192:168:3::/64 is directly connected, r3-eth0, 01:12:57
O>* 2001:192:168:22::/64 [110/20] via fe80::b849:73ff:fe63:4da3, r3-eth1, 01:12:
37
O   2001:192:168:23::/64 [110/10] is directly connected, r3-eth1, 01:12:42
C>* 2001:192:168:23::/64 is directly connected, r3-eth1, 01:12:57
O   2001:192:168:33::/64 [110/10] is directly connected, lo, 01:12:51
C>* 2001:192:168:33::/64 is directly connected, lo, 01:12:58
C * fe80::/64 is directly connected, r3-eth1, 01:12:58
C>* fe80::/64 is directly connected, r3-eth0, 01:12:58
frr-pc#
```

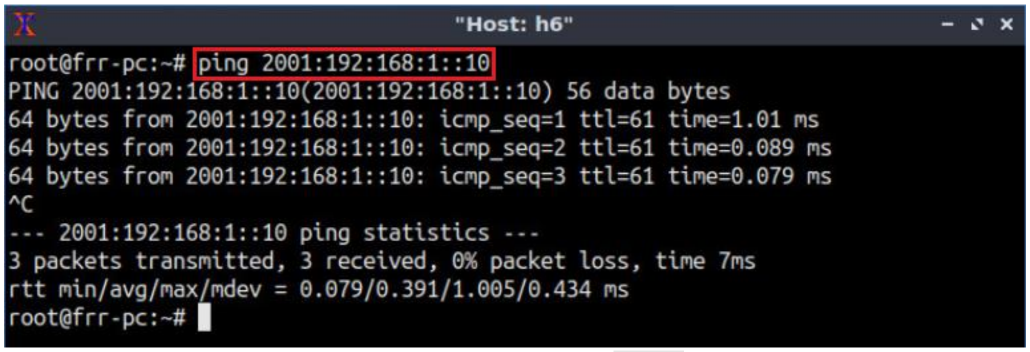
Figure 82. Verifying IPv6 routes in router r3.

Consider the figure above. To reach the network 2001:192:168:1::/64, router r3 uses the link local address of the interface *r3-eth1* to communicate with router r2 via interface *r3-*

eth1 since they are directly connected. Then, router r2 (2001:192:168:22::1) uses IPv4 BGP transport to reach the destination, 2001:192:168:1::/64.

Step 2. In host h6 terminal, perform a connectivity between host h6 and host h4 by issuing the command shown below. To stop the test, press `Ctrl+c`. The result will show a successful connectivity test.

```
ping 2001:192:168:1::10
```



```
root@frr-pc:~# ping 2001:192:168:1::10
PING 2001:192:168:1::10(2001:192:168:1::10) 56 data bytes
64 bytes from 2001:192:168:1::10: icmp_seq=1 ttl=61 time=1.01 ms
64 bytes from 2001:192:168:1::10: icmp_seq=2 ttl=61 time=0.089 ms
64 bytes from 2001:192:168:1::10: icmp_seq=3 ttl=61 time=0.079 ms
^C
--- 2001:192:168:1::10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 7ms
rtt min/avg/max/mdev = 0.079/0.391/1.005/0.434 ms
root@frr-pc:~#
```

Figure 83. Connectivity test using `ping` command.

This concludes Lab 1. Stop the emulation and then exit out of MiniEdit.

References

1. J. Postel, “*Internet protocol*”, 1981. [Online]. Available: https://tools.ietf.org/html/rfc791
2. S. Deering, R. Hinden, “*Internet protocol version 6 (IPv6) specification*”, 1998. [Online]. Available: https://tools.ietf.org/html/rfc2460
3. Cisco, “*IPv6 addressing and basic connectivity configuration guide, Cisco IOS release 15M&T*”, 2013. [Online]. Available: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipv6_basic/configuration/15-mt/ip6b-15-mt-book.pdf
4. A. Tanenbaum, D. Wetherall, “*Computer networks*”, 5th Edition, Pearson, 2012.
5. Cisco, “*What are OSPF areas and virtual links?*”, 2016. [Online]. Available: https://www.cisco.com/c/en/us/support/docs/ip/open-shortest-path-first-ospf/13703-8.html
6. R. Coltun, D. Ferguson, J. Moy, A. Lindem, “*OSPF for IPv6*”, 2008.
7. T. Bates, R. Chandra, D. Katz, Y. Rekhter, “*Multiprotocol extensions for BGP-4*”, 1998. [Online]. Available: https://tools.ietf.org/html/rfc2283
8. Cisco, “*Implementing Cisco IP routing (ROUTE) foundation learning guide*”, Pearson, 2015.
9. D. Teare, B. Vachon, R. Graziani, "Implementing cisco IP Routing (Route), foundation learning guide," CCNP ROUTE 300-101.
10. CCNP, Lab 7-5, “*Configuring MP-BGP*”.

MPLS AND ADVANCED BGP TOPICS

Lab 2: IP Spoofing and Mitigation Techniques

Document Version: 3-5-2020

Award 1829698
"CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers"

Contents

Overview

This lab introduces Internet Protocol (IP) address spoofing that occurs on the Internet between routers running Border Gateway Protocol (BGP). In this lab, a compromised host will spoof the IP address and launch a Denial of Service (DoS) on a victim, each in a different Autonomous System (AS). The goal of this lab is to configure the Internet Service Provider (ISP) to mitigate IP spoofing attacks by applying the appropriate filters on the network traffic of its customers.

Objectives

By the end of this lab, students should be able to:

1. Configure BGP as the main protocol between ASes.
2. Understand and configure IP spoofing and DoS attack.
3. Understand IP spoofing mitigation techniques.
4. Apply route filters to mitigate IP spoofing.

Lab settings

The information in Table 1 provides the credentials to access Client machine.

Table 1**.** Credentials to access Client machine.

Device	**Account**	**Password**
Client	admin	password

Lab roadmap

This lab is organized as follows:

1. Section 1: Introduction.
2. Section 2: Lab topology.
3. Section 3: Configure BGP on routers.
4. Section 4: Perform IP spoofing and DoS attack.
5. Section 5: Mitigate DDoS attack by using IP source filtering.

1 Introduction

1.1 BGP overview

BGP is an exterior gateway protocol designed to exchange routing and reachability information among ASes on the Internet. BGP is relevant to network administrators of large organizations which connect to one or more ISPs, as well as to ISPs who connect to other network providers. In terms of BGP, an AS is referred to as a routing domain, where all networked systems operate common routing protocols and are under the control of a single administration[1].

BGP is a form of distance vector protocol. It requires each router to maintain a table, which stores the distance and the output interface (i.e., vector) to remote networks. BGP makes routing decisions based on paths, network policies, or rule set configured by a network administrator and is involved in making core routing decisions[1].

Two routers that establish a BGP connection are referred to as BGP peers or neighbors**.** BGP sessions run over Transmission Control Protocol (TCP). If a BGP session is established between two neighbors in different ASes, the session is referred to as an External BGP (EBGP) session. If the session is established between two neighbors in the same AS, the session is referred to as Internal BGP (IBGP)[1]. Figure 1 shows a network running BGP protocol. Routers that exchange information within the same AS use IBGP, while routers that exchange information between different ASes use EBGP.

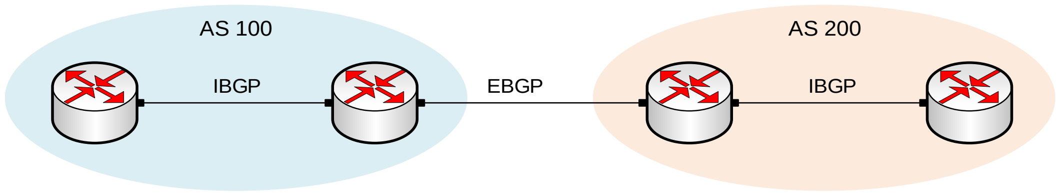


Figure 1. Routers that exchange information within the same AS use IBGP, while routers that exchange information between different ASes use EBGP.

1.2 IP Spoofing and DoS attacks

IP source address spoofing is the process of originating IP packets with source addresses other than those assigned to the origin host. An attacker that spoofs source IP addresses appears as the to be another host[2]. IP spoofing can be exploited in several ways, mainly to launch DoS attacks. The latter is an attack that can exhaust the computing and communication resources of its victim within a short period of time[3].

Consider Figure 2. Host A (attacker) spoofs the source IP address of host C (victim) and request host B to send 100 GB of data. Host B will receive the request and sends the data to the spoofed source address, i.e., to host C. Thus, consuming the resources of the victim.

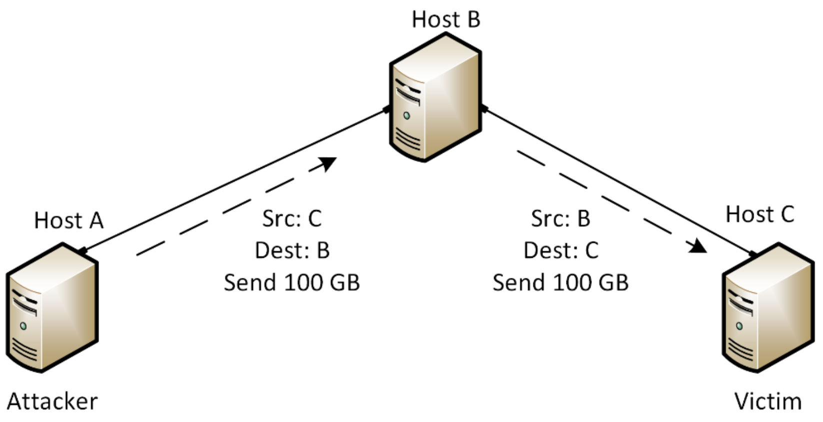


Figure 2. Host A performs DoS attack on host C by spoofing its IP address.

1.3 Anti-Spoofing techniques

Mutually Agreed Norms for Routing Security (MANRS) is a global initiative, supported by the Internet Society, that provides crucial fixes to reduce the most common routing threats. MANRS as many recommendations to prevent IP spoofing by ingress filtering, e.g., checking the source addresses of IP datagrams[4].

1.3.1 Unicast Reverse Path Forwarding (uRPF)

uRPF is one effective method to prevent IP spoofing. uRPF has multiple modes of operation, among them is the uRPF strict mode, in which the router accepts incoming packets on a specific interface if two conditions satisfy[4]:

1. The source IP address of the incoming packet has an entry in the routing table.
2. The router uses the same interface to reach this source as where it received the packet on.

Consider Figure 3. Router r1 uses uRPF strict mode. Incoming packets on interface *r1-eth0* that have source IP address of Host B will be dropped, since router r1 uses interface *r1-eth1* to reach host B.

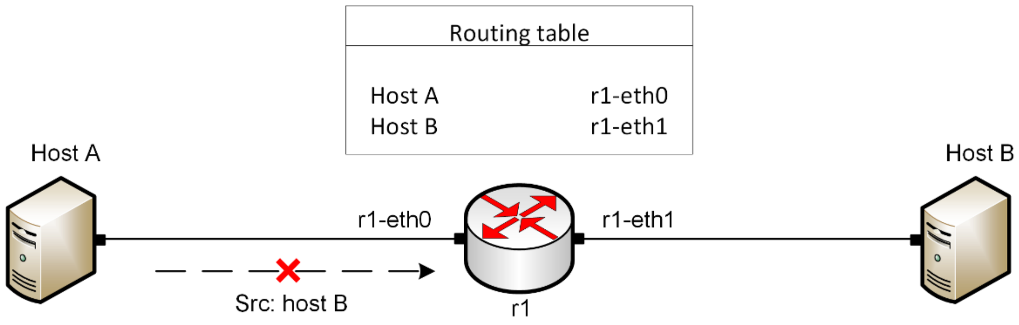


Figure 3. Router r1 uses uRPF to prevent spoofed IP packets.

1.3.2 Route filtering

Route filtering is a method for selectively identifying routes that are advertised or received from neighbor routers. Route filtering may be used to manipulate traffic flows, reduce memory utilization, or to improve security[5].

Network operators should apply route filters to prevent spoofed IP packets from their customers. Consider Figure 4. The ISP (router r2) filters inbound network traffic of its customers, i.e., traffic sent from routers r1 and r3, based on their assigned IP addresses. Thus, each customer can't generate network traffic with spoofed IP addresses.

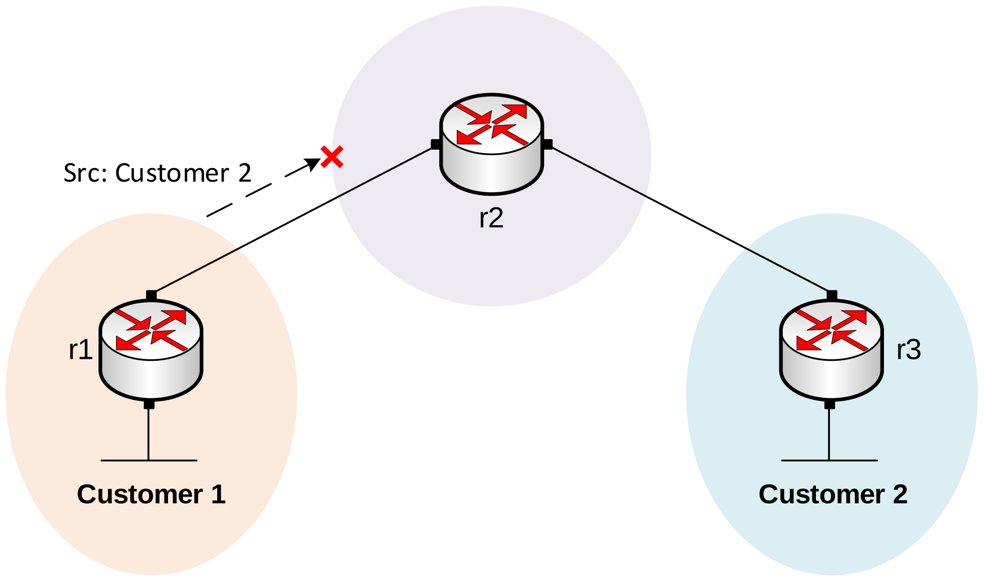


Figure 4. The ISP drops network traffic sent from Customer 1, since their source IP address corresponds to Customer 2.

In this lab, we will apply the route filters to prevent IP spoofing using netfilter. The latter is a framework for packet filtering built in Linux kernel[6].

2 Lab topology

Consider Figure 5. The topology consists of three ASes. The ISP, consisting of routers r2 and r3, provides Internet service to the Campus-1 (router r1) and Campus-2 (router r4) networks. The Autonomous System Numbers (ASNs) assigned to Campus-1, ISP, and Campus-2 are 100, 200, and 300, respectively. The ISP communicates with the Campus networks via EBGP routing protocol, and the routers within the ISP communicate using IBGP. Host h1 in Campus-1 spoofs the IP address of host h4 in Campus-2. Consequently, host h1 launches a DoS attack on host h4 using hosts h2 and h3. To mitigate IP spoofing, the ISP (router r2) applies the appropriate route filters on the network traffic generated from Campus-1.

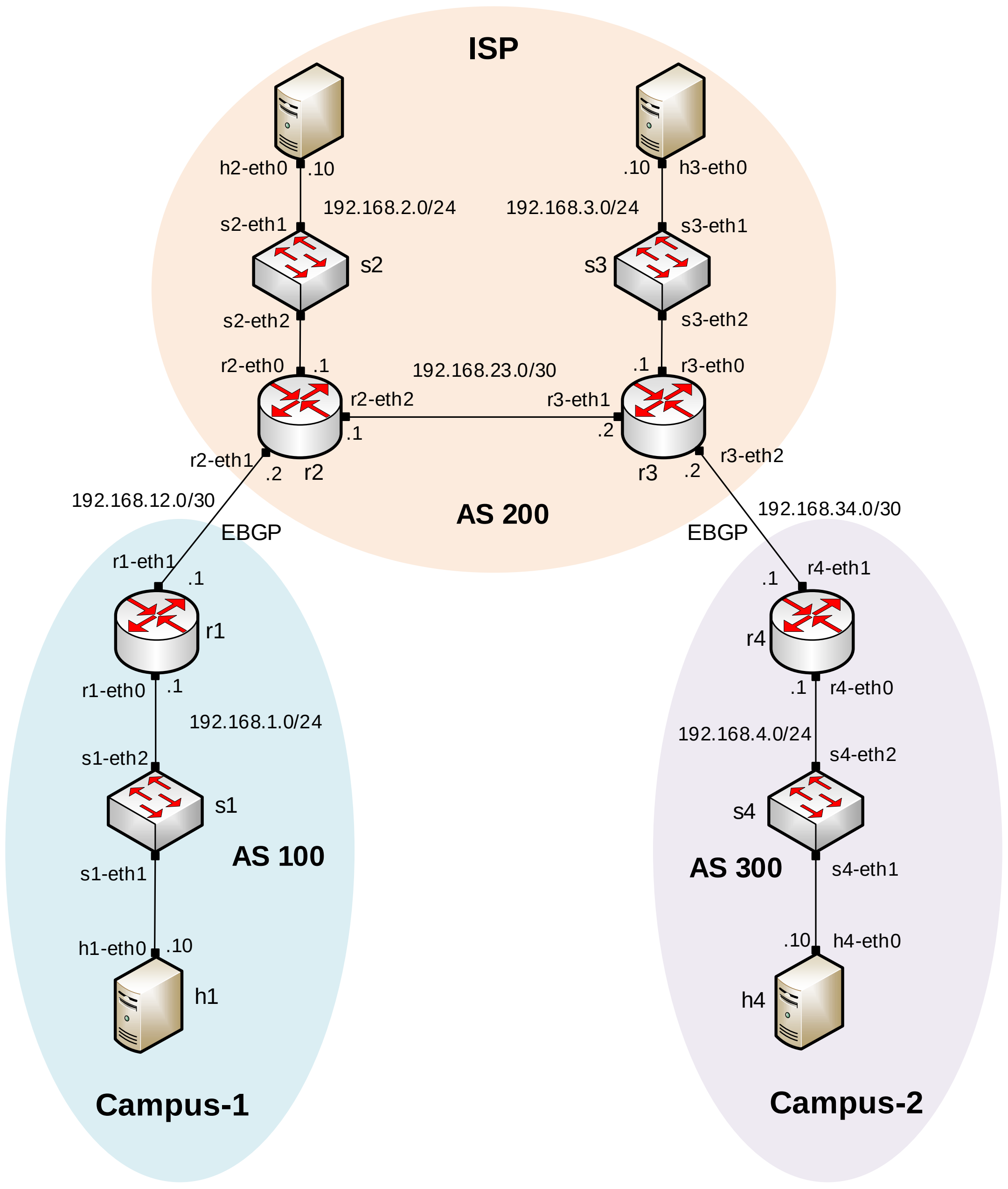


Figure 5. Lab topology.

2.1 Lab settings

Routers and hosts are already configured according to the IP addresses shown in Table 2.

Table 2. Topology information.

Device	Interface	IPV4 Address	Subnet	Default gateway
r1 (Campus-1)	r1-eth0	192.168.1.1	/24	N/A
	r1-eth1	192.168.12.1	/30	N/A

r2 (ISP)	r2-eth0	192.168.2.1	/24	N/A
	r2-eth1	192.168.12.2	/30	N/A
	r2-eth2	192.168.23.1	/30	N/A
r3 (ISP)	r3-eth0	192.168.3.1	/24	N/A
	r3-eth1	192.168.23.2	/30	N/A
	r3-eth2	192.168.34.1	/30	N/A
r4 (Campus-2)	r4-eth0	192.168.4.1	/24	N/A
	r4-eth1	192.168.34.2	/30	N/A
h1	h1-eth0	192.168.1.10	/24	192.168.1.1
h2	h2-eth0	192.168.2.10	/24	192.168.2.1
h3	h3-eth0	192.168.3.10	/24	192.168.3.1
h4	h4-eth0	192.168.4.10	/24	192.168.4.1

2.2 Open topology and load the configuration

Step 1. Start by launching Miniedit by clicking on Desktop's shortcut. When prompted for a password, type `password`.

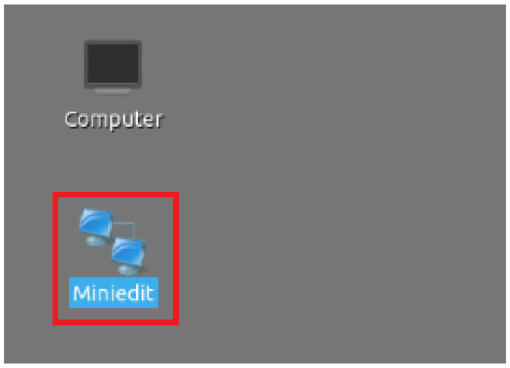


Figure 6. MiniEdit shortcut.

Step 2. On Miniedit's menu bar, click on *File* then *open* to load the lab's topology. Locate the *Lab2.mn* topology file in the default directory, */home/frr/MPLS_advanced_BGP/lab2* and click on *Open*.

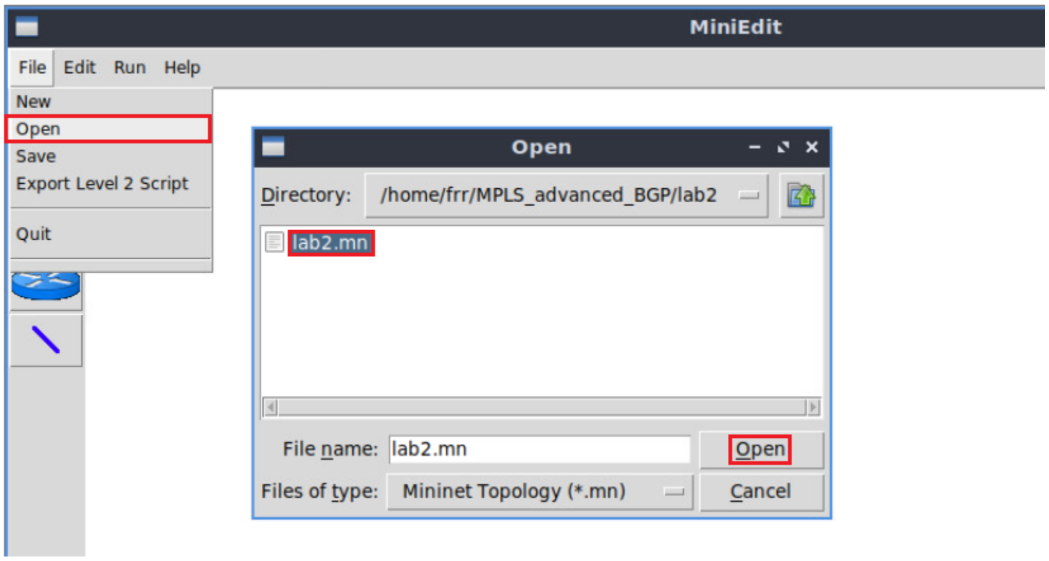


Figure 7. MiniEdit's Open dialog.

At this point the topology is loaded with all the required network components. You will execute a script that will load the configuration of the routers.

Step 3. Open the Linux terminal.

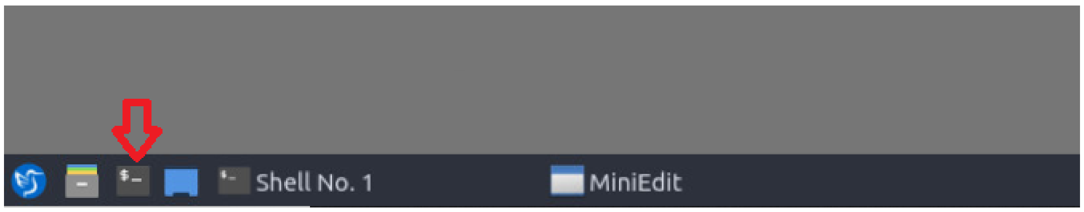
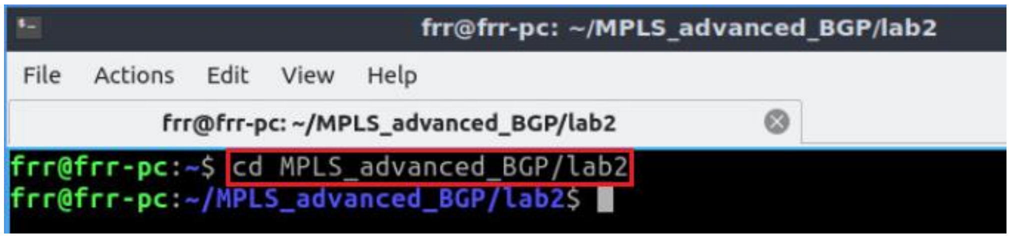


Figure 8. Opening Linux terminal

Step 4. Click on the Linux's terminal and navigate into *MPLS_advanced_BGP/lab2* directory by issuing the following command. This folder contains a configuration file and the script responsible for loading the configuration. The configuration file will assign the IP addresses to the routers' interfaces. The `cd` command is short for change directory followed by an argument that specifies the destination directory.

```
cd MPLS_advanced_BGP/lab2
```

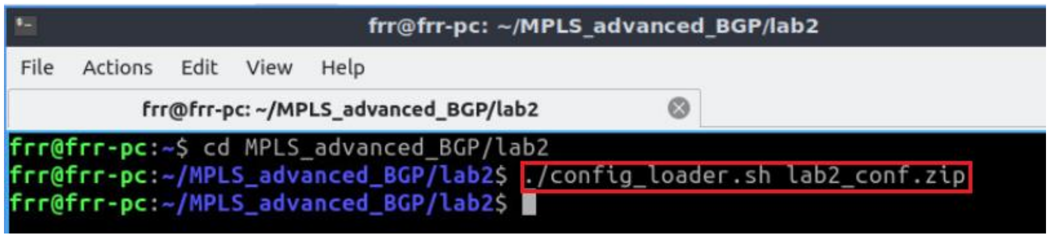

frr@frr-pc: ~/MPLS_advanced_BGP/lab2
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab2

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab2
frr@frr-pc:~/MPLS_advanced_BGP/lab2$
```

Figure 9. Entering to the *MPLS_advanced_BGP/lab2* directory.

Step 5. To execute the shell script, type the following command. The argument of the program corresponds to the configuration zip file that will be loaded in all the routers in the topology.

```
./config_loader.sh lab2_conf.zip
```

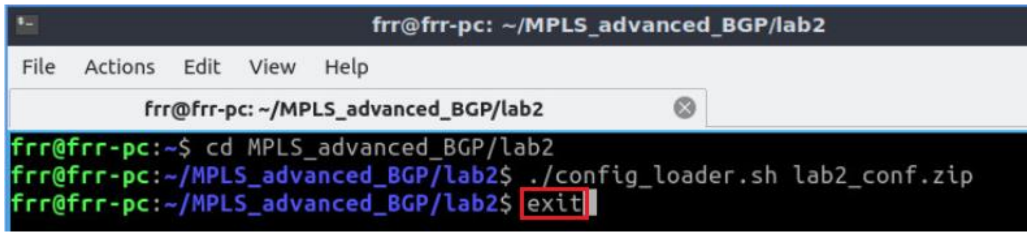

frr@frr-pc: ~/MPLS_advanced_BGP/lab2
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab2

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab2
frr@frr-pc:~/MPLS_advanced_BGP/lab2$ ./config_loader.sh lab2_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab2$
```

Figure 10. Executing the shell script to load the configuration.

Step 6. Type the following command to exit the Linux terminal.

```
exit
```


frr@frr-pc: ~/MPLS_advanced_BGP/lab2
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab2

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab2
frr@frr-pc:~/MPLS_advanced_BGP/lab2$ ./config_loader.sh lab2_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab2$ exit
```

Figure 11. Exiting from the terminal.

Step 7. At this point hosts h1, h2, h3 and h4 interfaces are configured. To proceed with the emulation, click on the *Run* button located in lower left-hand side.

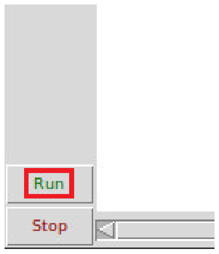


Figure 12. Starting the emulation.

Step 8. Click on Mininet's terminal, i.e., the one launched when MiniEdit was started.

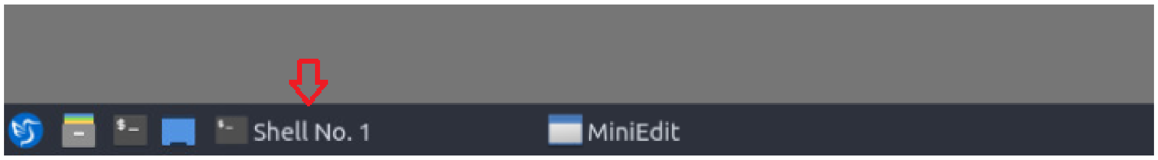
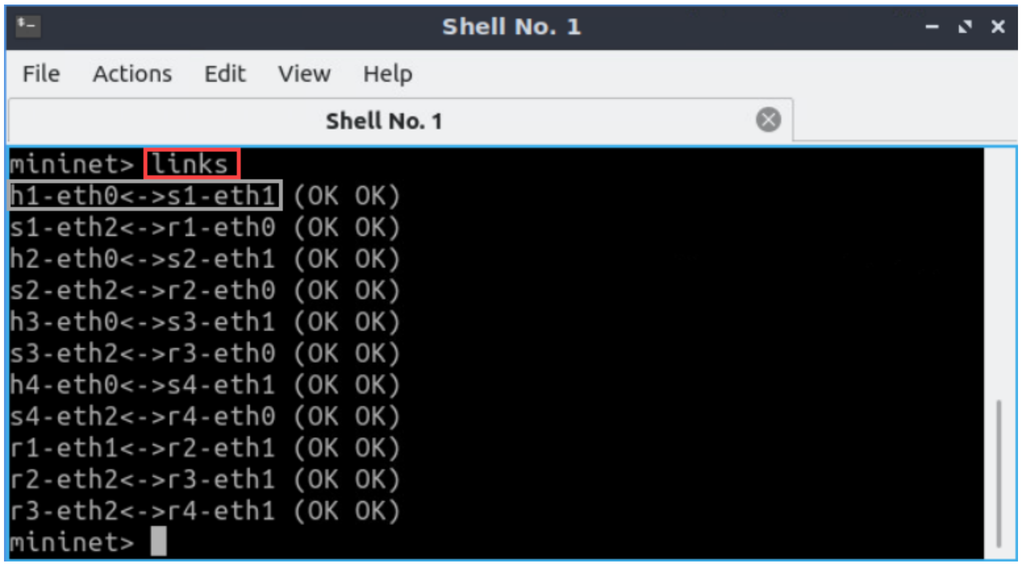


Figure 13. Opening Mininet's terminal.

Step 9. Issue the following command to display the interface names and connections.

```
links
```



```
mininet> links
h1-eth0<->s1-eth1 (OK OK)
s1-eth2<->r1-eth0 (OK OK)
h2-eth0<->s2-eth1 (OK OK)
s2-eth2<->r2-eth0 (OK OK)
h3-eth0<->s3-eth1 (OK OK)
s3-eth2<->r3-eth0 (OK OK)
h4-eth0<->s4-eth1 (OK OK)
s4-eth2<->r4-eth0 (OK OK)
r1-eth1<->r2-eth1 (OK OK)
r2-eth2<->r3-eth1 (OK OK)
r3-eth2<->r4-eth1 (OK OK)
mininet>
```

Figure 14. Displaying network interfaces.

In Figure 14, the link displayed within the gray box indicates that interface *eth0* of host h1 connects to interface *eth1* of switch s1 (i.e., *h1-eth0<->s1-eth1*).

2.3 Load zebra daemon and Verify IP addresses

You will verify the IP addresses listed in Table 2 and inspect the routing table of routers r1, r2, r3 and r4.

Step 1. Hold right-click on host h1 and select *Terminal*. This opens the terminal of host h1 and allows the execution of commands on that host.

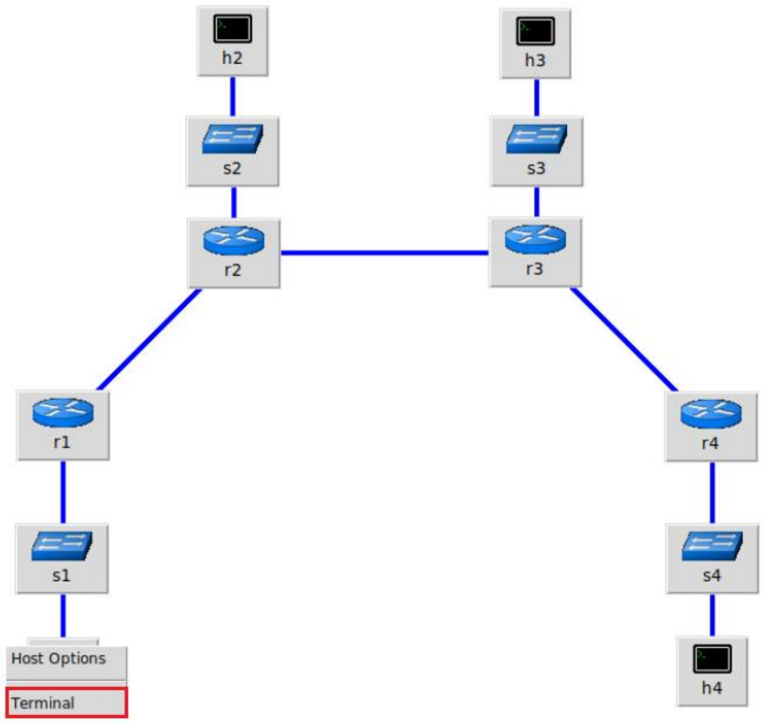
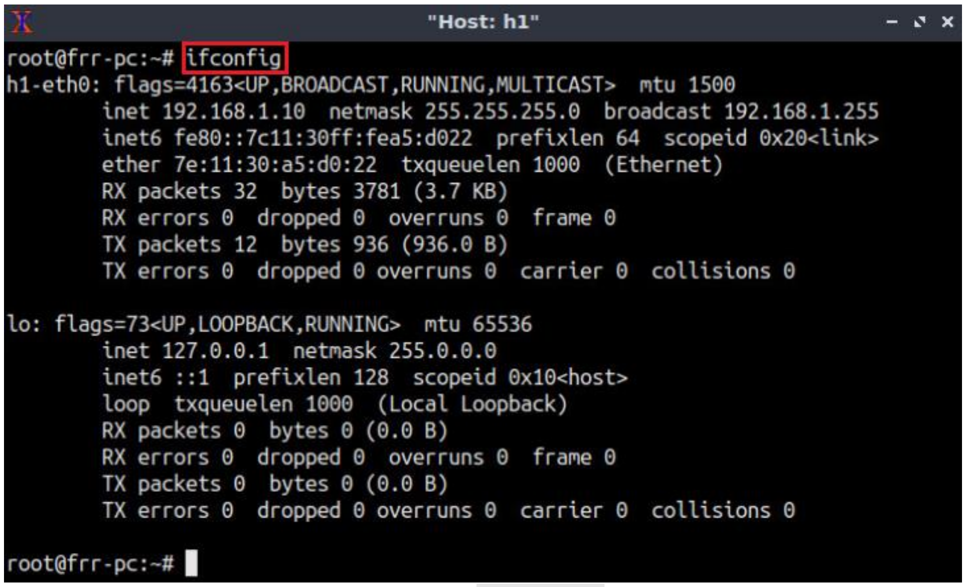


Figure 15. Opening terminal on host h1.

Step 2. On host h1 terminal, type the command shown below to verify that the IP address was assigned successfully. You will verify that host h1 has an interface, *h1-eth0* configured with the IP address 192.168.1.10 and the subnet mask 255.255.255.0.

```
ifconfig
```

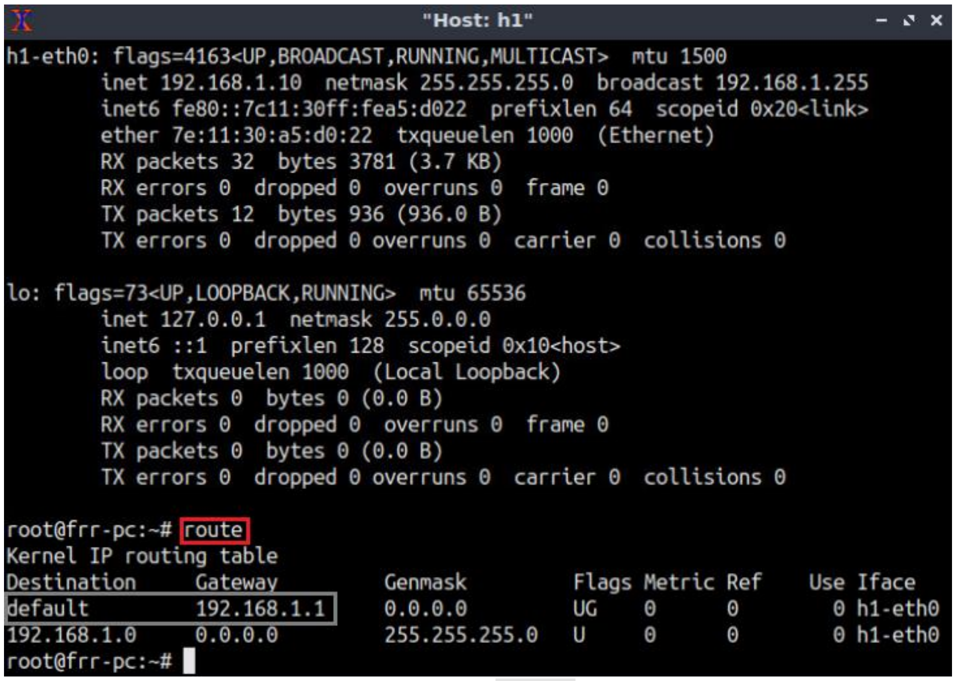


```
"Host: h1"
root@frr-pc:~# ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.10  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::7c11:30ff:fea5:d022  prefixlen 64  scopeid 0x20<link>
        ether 7e:11:30:a5:d0:22  txqueuelen 1000  (Ethernet)
        RX packets 32  bytes 3781 (3.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12  bytes 936 (936.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@frr-pc:~#
```

Figure 16. Output of `ifconfig` command.

Step 3. On host h1 terminal, type the command shown below to verify that the default gateway IP address is 192.168.1.1.

```
route
```



```
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.10  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::7c11:30ff:fea5:d022  prefixlen 64  scopeid 0x20<link>
        ether 7e:11:30:a5:d0:22  txqueuelen 1000  (Ethernet)
        RX packets 32  bytes 3781 (3.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12  bytes 936 (936.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@frr-pc:~# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         192.168.1.1     0.0.0.0         UG    0      0        0 h1-eth0
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 h1-eth0
root@frr-pc:~#
```

Figure 17. Output of `route` command.

Step 4. In order to verify hosts h2, h3 and h4, proceed similarly by repeating from step 1 to step 3 on host h2, h3 and h4 terminals. Similar results should be observed.

Step 5. You will validate that the router interfaces are configured correctly according to Table 2. In order to verify router r1, hold right-click on router r1 and select Terminal.

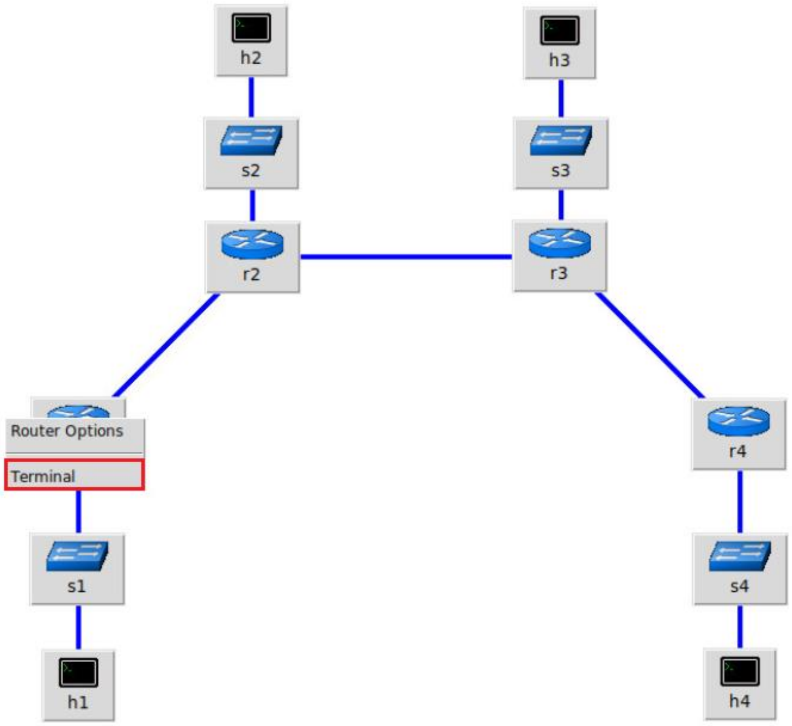
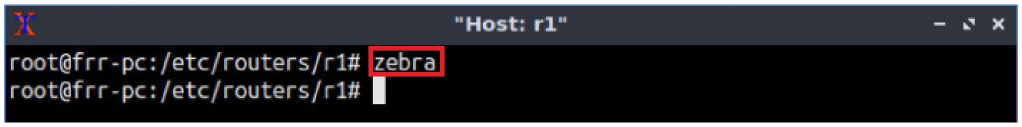


Figure 18. Opening terminal on router r1.

Step 6. Start zebra daemon, which is a multi-server routing software that provides TCP/IP based routing protocols. The configuration will not be working if you do not enable zebra daemon initially. In order to start the zebra, type the following command:

```
zebra
```

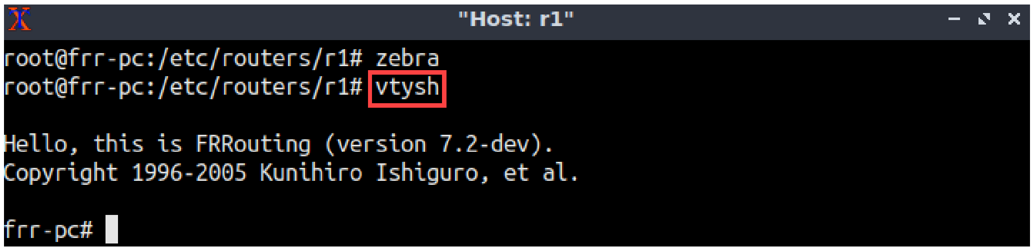


```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1#
```

Figure 19. Starting zebra daemon.

Step 7. After initializing zebra, vtysh should be started in order to provide all the CLI commands defined by the daemons. To proceed, issue the following command:

```
vtysh
```

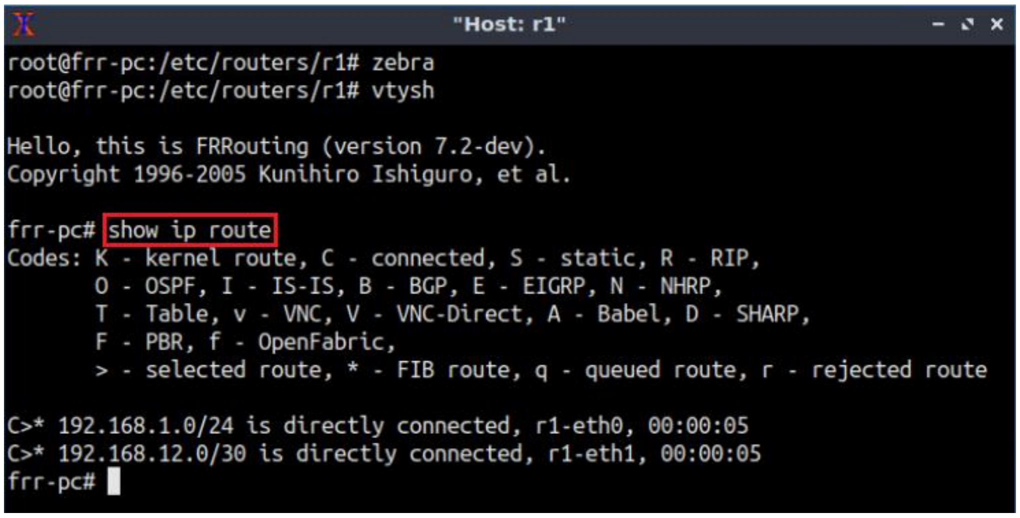


```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 20. Starting vtysh on router r1.

Step 8. Type the following command on router r1 terminal to verify the routing table of router r1. It will list all the directly connected networks. The routing table of router r1 does not contain any route to the networks attached to routers r2 (192.168.2.0/24), r3 (192.168.3.0/24) and r4 (192.168.4.0/24) as there is no routing protocol configured yet.

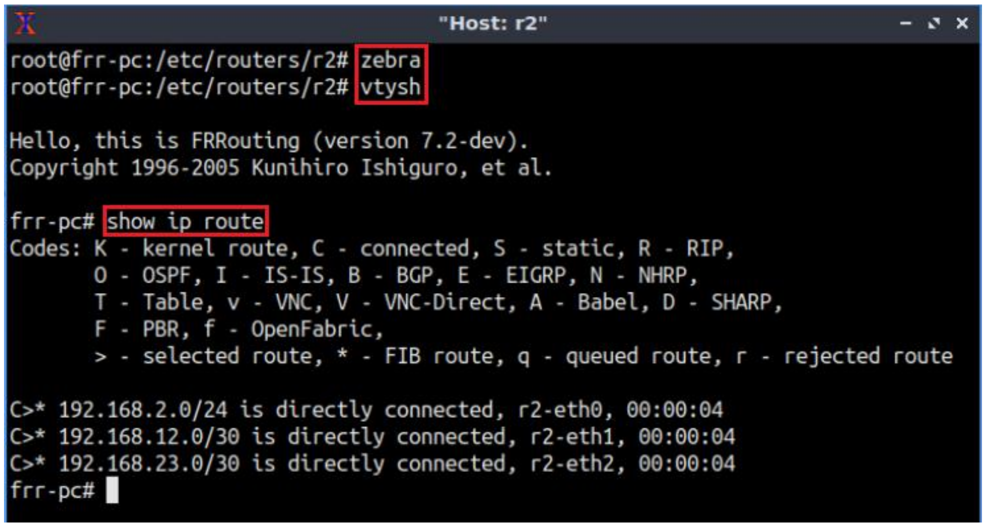
```
show ip route
```



```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.1.0/24 is directly connected, r1-eth0, 00:00:05
C>* 192.168.12.0/30 is directly connected, r1-eth1, 00:00:05
frr-pc#
```

Figure 21. Displaying the routing table of router r1.

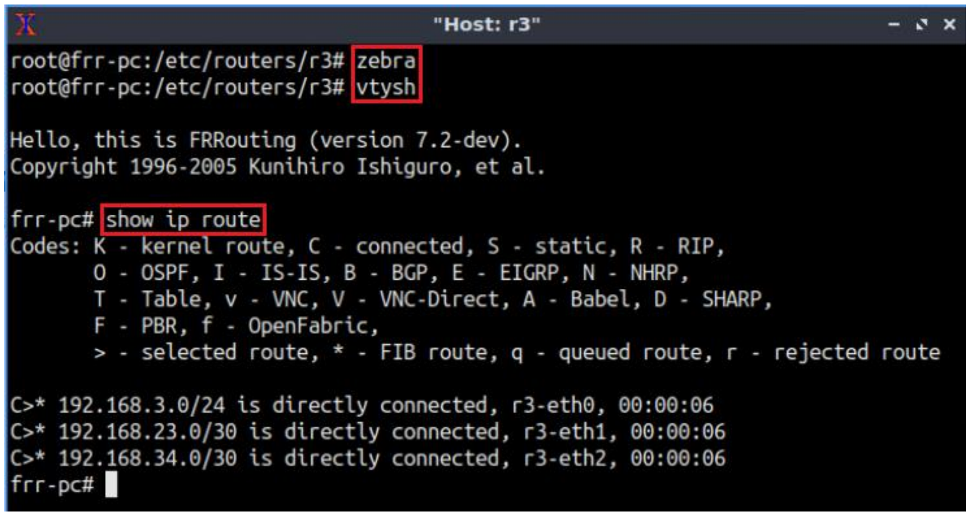
Step 9. Router r2 is configured similarly to router r1 but, with different IP addresses (see Table 2). Those steps are summarized in the following figure. To proceed, in router r2 terminal issue the commands depicted below. At the end, you will verify all the directly connected networks of router r2.



```
root@frr-pc:/etc/routers/r2# zebra
root@frr-pc:/etc/routers/r2# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.2.0/24 is directly connected, r2-eth0, 00:00:04
C>* 192.168.12.0/30 is directly connected, r2-eth1, 00:00:04
C>* 192.168.23.0/30 is directly connected, r2-eth2, 00:00:04
frr-pc#
```

Figure 22. Displaying the routing table of router r2.

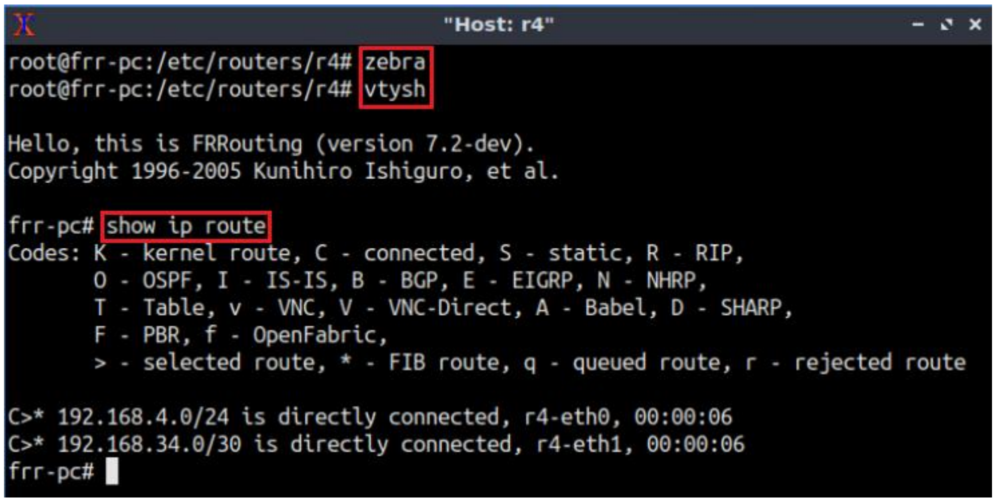
Step 10. Router r3 is configured similarly to router r1 but, with different IP addresses (see Table 2). Those steps are summarized in the following figure. To proceed, in router r3 terminal issue the commands depicted below. At the end, you will verify all the directly connected networks of router r3.



```
root@frr-pc:/etc/routers/r3# zebra
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.3.0/24 is directly connected, r3-eth0, 00:00:06
C>* 192.168.23.0/30 is directly connected, r3-eth1, 00:00:06
C>* 192.168.34.0/30 is directly connected, r3-eth2, 00:00:06
frr-pc#
```

Figure 23. Displaying the routing table of router r3.

Step 11. Router r4 is configured similarly to router r1 but, with different IP addresses (see Table 2). Those steps are summarized in the following figure. To proceed, in router r4 terminal issue the commands depicted below. At the end, you will verify all the directly connected networks of router r4.



```
"Host: r4"
root@frr-pc:/etc/routers/r4# zebra
root@frr-pc:/etc/routers/r4# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.4.0/24 is directly connected, r4-eth0, 00:00:06
C>* 192.168.34.0/30 is directly connected, r4-eth1, 00:00:06
frr-pc#
```

Figure 24. Displaying the routing table of router r4.

3 Configure BGP on routers

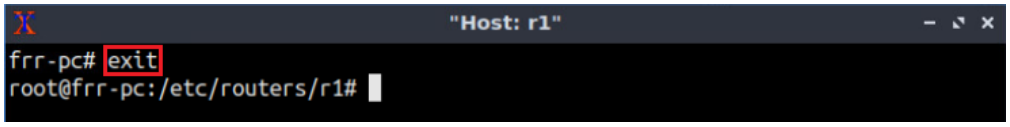
In this section, you will configure BGP on the routers that are hosted in different ASes. You will assign BGP neighbors to allow the routers to exchange BGP routes. Furthermore, all routers will advertise their Local Area Networks (LANs) via BGP.

3.1 Configure EBGP on routers

In this section, you will configure EBGP on all routers.

Step 1. To configure BGP routing protocol, you need to enable the BGP daemon first. In router r1 terminal, type the following command to exit the vtysh session:

```
exit
```

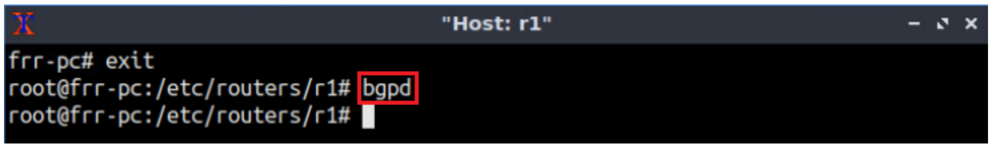


```
"Host: r1"
frr-pc# exit
root@frr-pc:/etc/routers/r1#
```

Figure 25. Exiting the vtysh session.

Step 2. Type the following command on r1 terminal to enable and start BGP routing protocol.

```
bgpd
```

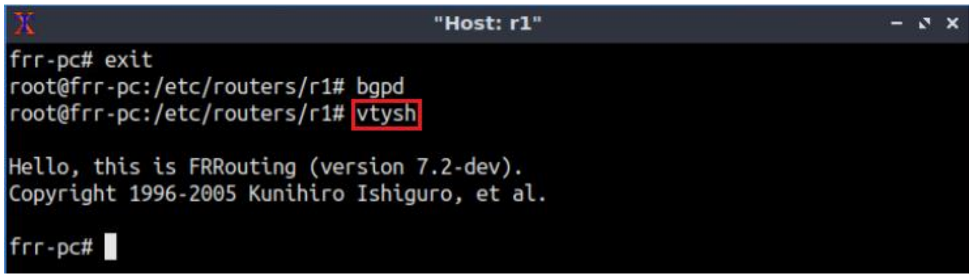


```
"Host: r1"
frr-pc# exit
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1#
```

Figure 26. Starting BGP daemon.

Step 3. In order to enter to router r1 terminal, type the following command:

```
vtysh
```



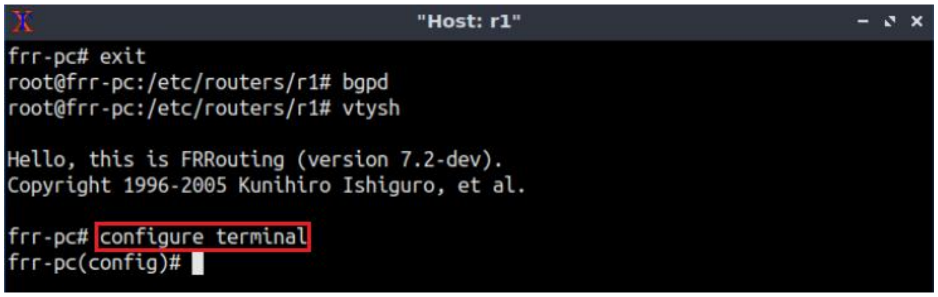
"Host: r1"

```
frr-pc# exit
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 27. Starting vtysh in router r1.

Step 4. To enable router r1 into configuration mode, issue the following command:

```
configure terminal
```



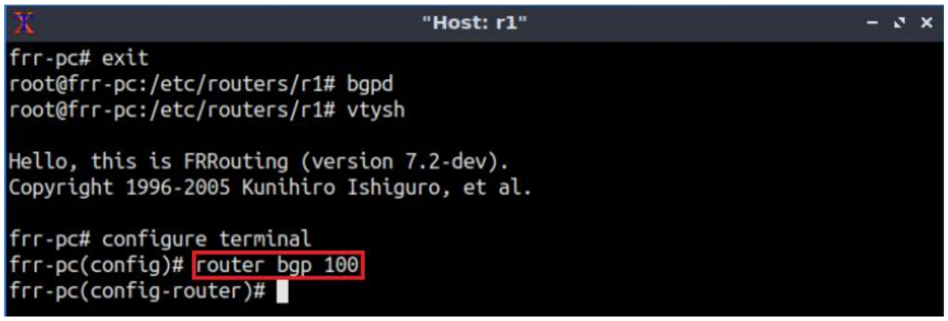
"Host: r1"

```
frr-pc# exit
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)#
```

Figure 28. Enabling configuration mode in router r1.

Step 5. The ASN assigned for router r1 is 100. In order to configure BGP, type the following command:

```
router bgp 100
```



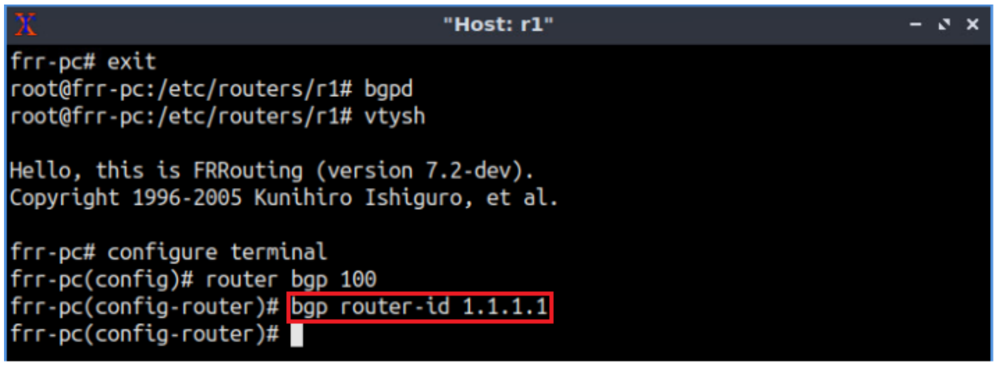
"Host: r1"

```
frr-pc# exit
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)#
```

Figure 29. Configuring BGP on router r1.

Step 6. Assign a router ID to router r1 by issuing the following command.

```
bgp router-id 1.1.1.1
```

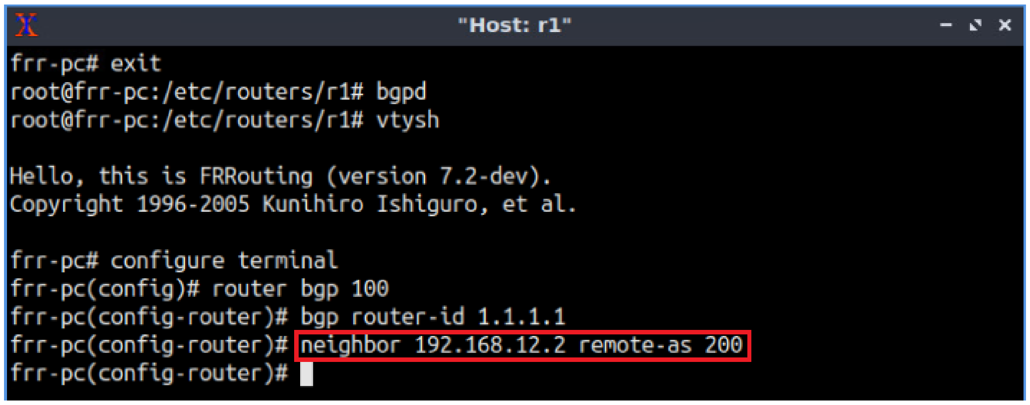


```
frr-pc# exit
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 1.1.1.1
frr-pc(config-router)#
```

Figure 30. Assigning a router ID in router r1.

Step 7. To configure a BGP neighbor to router r1 (AS 100), type the command shown below. This command specifies the neighbor IP address (192.168.12.2) and the ASN of the remote BGP peer (AS 200).

```
neighbor 192.168.12.2 remote-as 200
```

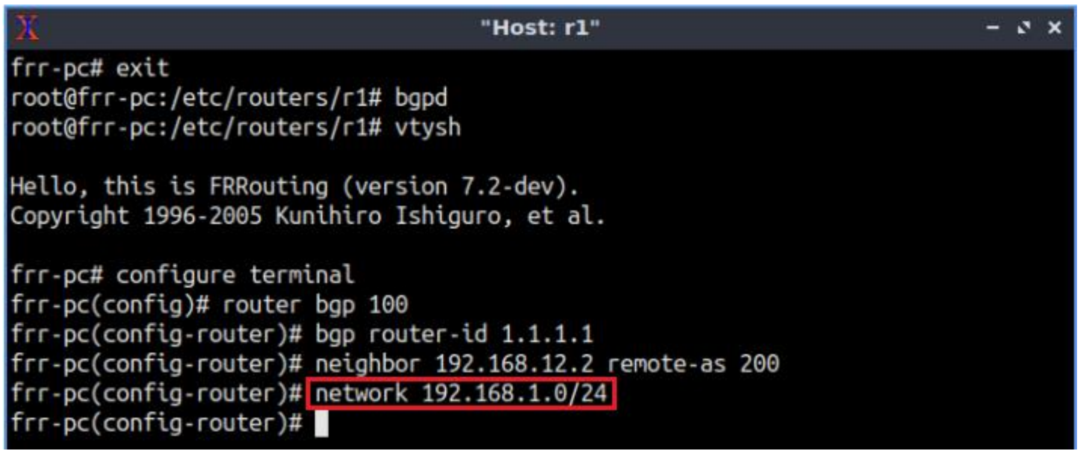


```
frr-pc# exit
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 1.1.1.1
frr-pc(config-router)# neighbor 192.168.12.2 remote-as 200
frr-pc(config-router)#
```

Figure 31. Assigning BGP neighbor to router r1.

Step 8. In this step, router r1 will advertise the LAN 192.168.1.0/24 to its BGP peers. To do so, issue the following command:

```
network 192.168.1.0/24
```

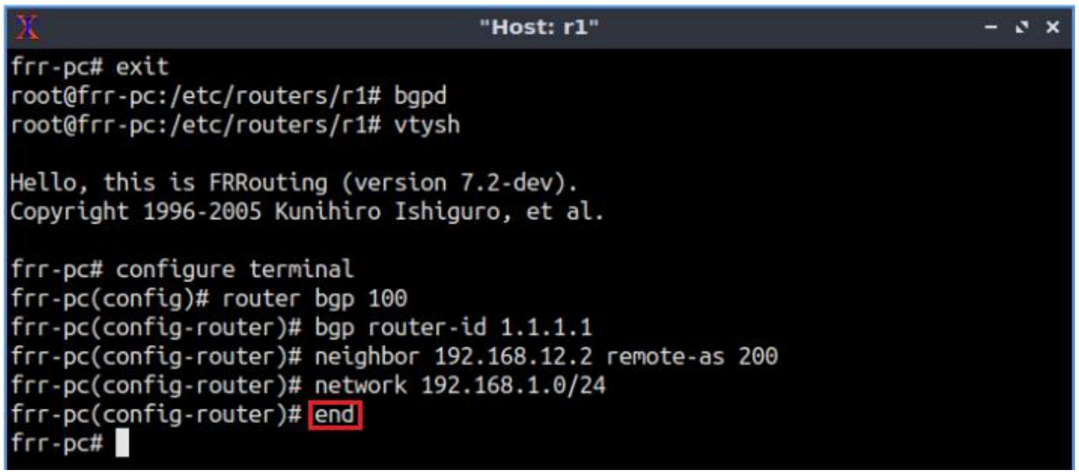


```
frr-pc# exit
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 1.1.1.1
frr-pc(config-router)# neighbor 192.168.12.2 remote-as 200
frr-pc(config-router)# network 192.168.1.0/24
frr-pc(config-router)#
```

Figure 32. Advertising local network in router r1.

Step 9. Type the following command to exit from configuration mode.

```
end
```

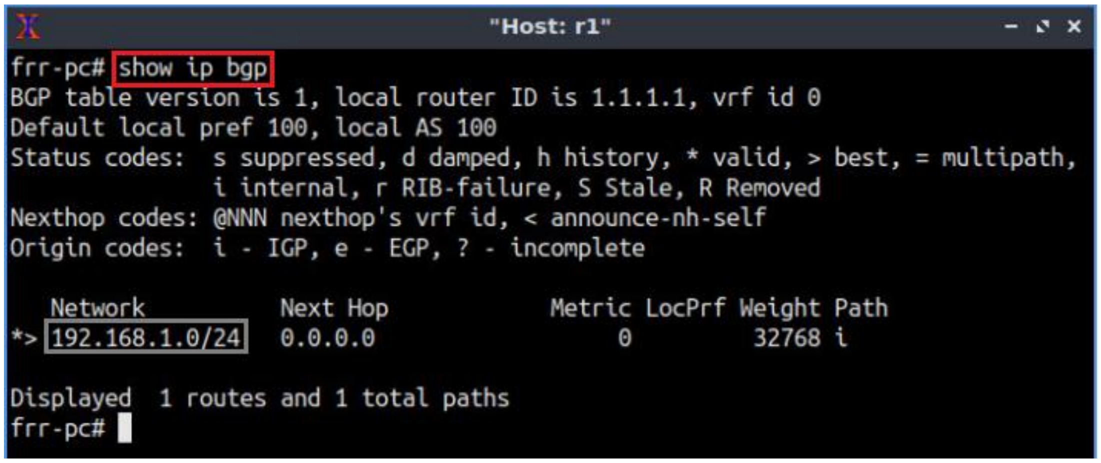


```
frr-pc# exit
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 1.1.1.1
frr-pc(config-router)# neighbor 192.168.12.2 remote-as 200
frr-pc(config-router)# network 192.168.1.0/24
frr-pc(config-router)# end
frr-pc#
```

Figure 33. Exiting from configuration mode.

Step 10. Type the following command to verify BGP networks. You will observe the LAN of router r1.

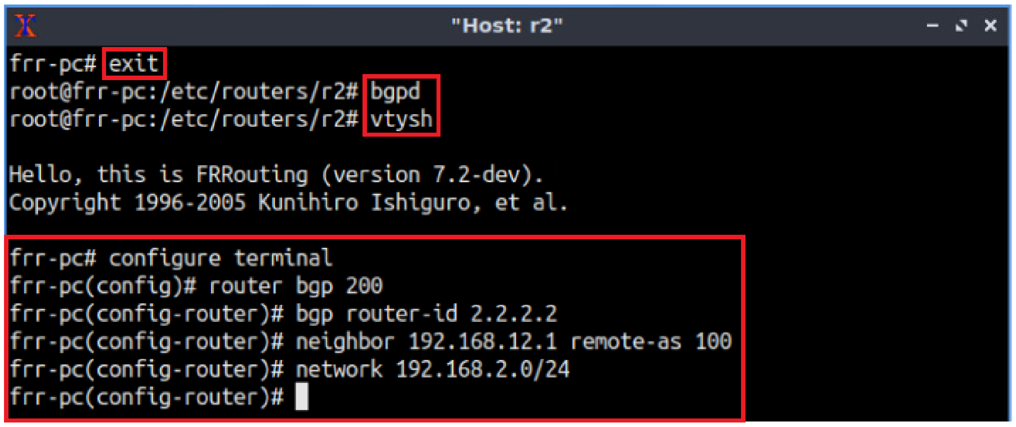
```
show ip bgp
```



```
frr-pc# show ip bgp
BGP table version is 1, local router ID is 1.1.1.1, vrf id 0
Default local pref 100, local AS 100
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 192.168.1.0/24   0.0.0.0                  0         32768 i

Displayed  1 routes and 1 total paths
frr-pc#
```

Figure 34. Verifying BGP networks in router r1.

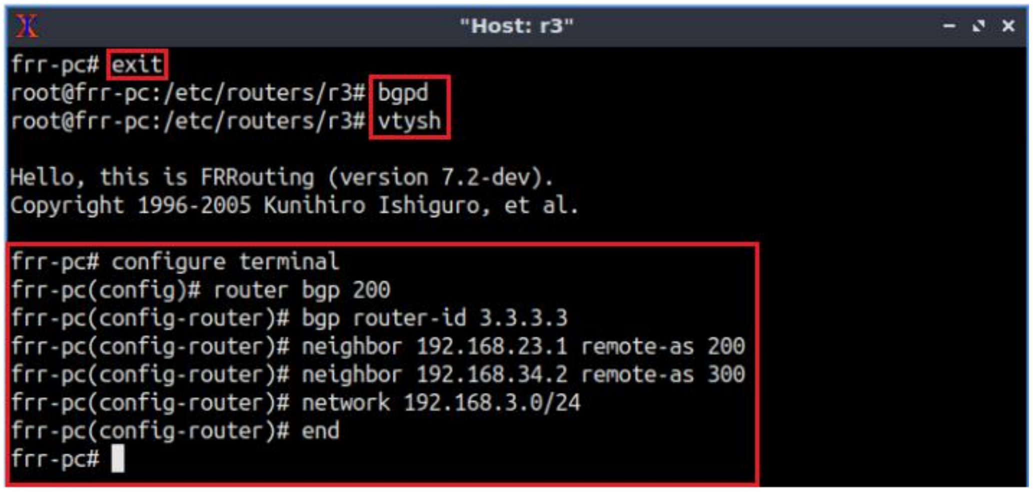
Step 11. Follow from step 1 to step 8 but with different metrics in order to configure BGP on router r2. All the steps are summarized in the following figure.



```
"Host: r2"
frr-pc# exit
root@frr-pc:/etc/routers/r2# bgpd
root@frr-pc:/etc/routers/r2# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 200
frr-pc(config-router)# bgp router-id 2.2.2.2
frr-pc(config-router)# neighbor 192.168.12.1 remote-as 100
frr-pc(config-router)# network 192.168.2.0/24
frr-pc(config-router)#
```

Figure 35. Configuring BGP on router r2.

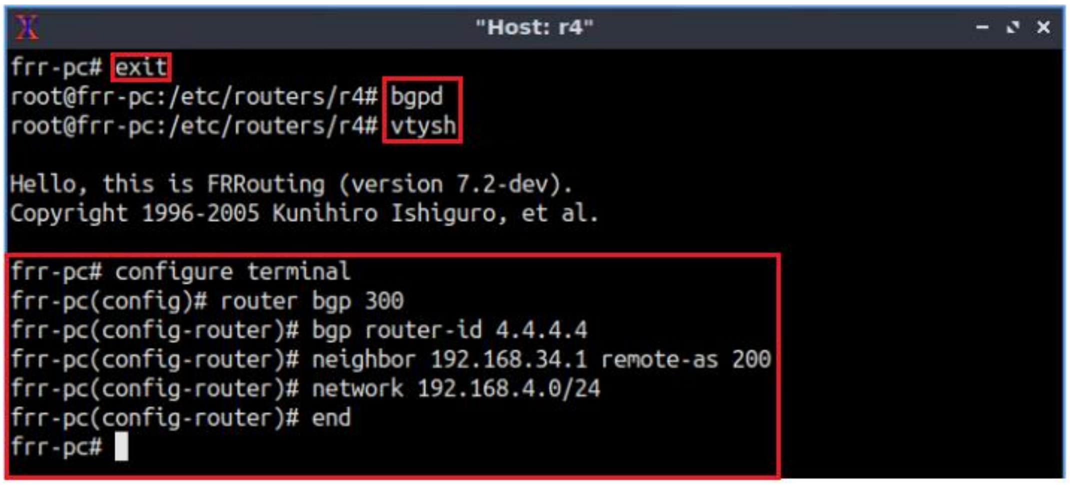
Step 11. Follow from step 1 to step 9 but with different metrics in order to configure EBGP on router r3 to establish BGP peering with routers r2 and r4. All the steps are summarized in the following figure.



```
"Host: r3"
frr-pc# exit
root@frr-pc:/etc/routers/r3# bgpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 200
frr-pc(config-router)# bgp router-id 3.3.3.3
frr-pc(config-router)# neighbor 192.168.23.1 remote-as 200
frr-pc(config-router)# neighbor 192.168.34.2 remote-as 300
frr-pc(config-router)# network 192.168.3.0/24
frr-pc(config-router)# end
frr-pc#
```

Figure 36. Configuring EBGP on router r3.

Step 12. Follow from step 1 to step 9 but with different metrics in order to configure EBGP on router r4. All the steps are summarized in the following figure.

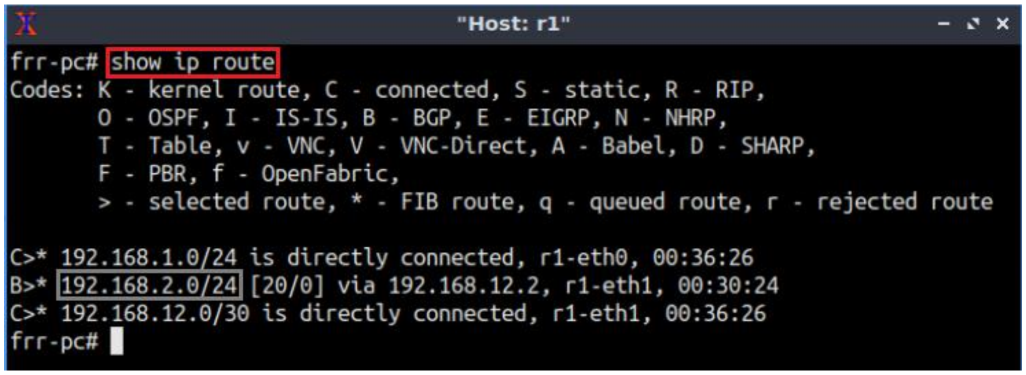


```
"Host: r4"
frr-pc# exit
root@frr-pc:/etc/routers/r4# bgpd
root@frr-pc:/etc/routers/r4# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 300
frr-pc(config-router)# bgp router-id 4.4.4.4
frr-pc(config-router)# neighbor 192.168.34.1 remote-as 200
frr-pc(config-router)# network 192.168.4.0/24
frr-pc(config-router)# end
frr-pc#
```

Figure 37. Configuring BGP on router r4.

Step 13. Type the following command to verify the routing table of router r1. Router r1 has a route to the network 192.168.2.0/24 only since IBGP is not configured between routers r2 and r3 yet.

```
show ip route
```



```
"Host: r1"
frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.1.0/24 is directly connected, r1-eth0, 00:36:26
B>* 192.168.2.0/24 [20/0] via 192.168.12.2, r1-eth1, 00:30:24
C>* 192.168.12.0/30 is directly connected, r1-eth1, 00:36:26
frr-pc#
```

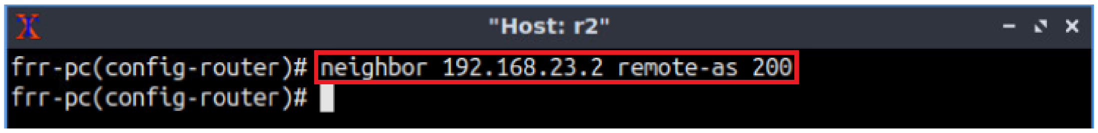
Figure 38. Displaying the routing table of router r1.

3.2 Configure IBGP on routers r2 and r3

In this section, you will configure IBGP on routers r2 and r3. Furthermore, you will configure BGP next-hop-self on routers r2 and r3 so that these routers have valid routes to the EBGP routes that are advertised by their IBGP neighbors.

Step 1. Type the following command on r2 terminal to establish IBGP peering with router r3.

```
neighbor 192.168.23.2 remote-as 200
```

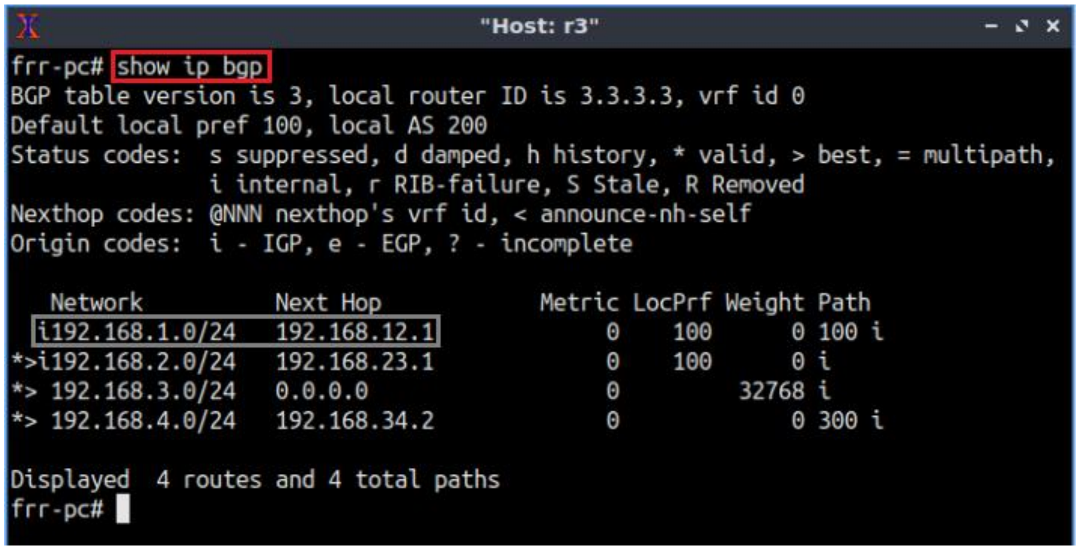


```
"Host: r2"
frr-pc(config-router)# neighbor 192.168.23.2 remote-as 200
frr-pc(config-router)#
```

Figure 39. Assigning BGP neighbor to router r2.

Step 2. Type the following command to verify the BGP table of router r3. Router r3 can't reach the network 192.168.1.0/24 since the next hop address (192.168.12.1) is not known to router r3.

```
show ip bgp
```

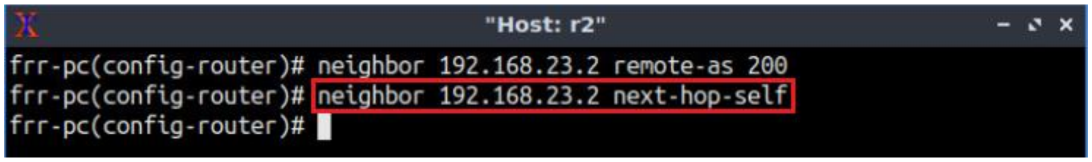


```
frr-pc# show ip bgp
BGP table version is 3, local router ID is 3.3.3.3, vrf id 0
Default local pref 100, local AS 200
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
  i192.168.1.0/24   192.168.12.1             0    100      0 100 i
*>i192.168.2.0/24   192.168.23.1             0    100      0 i
*> 192.168.3.0/24   0.0.0.0                  0         32768 i
*> 192.168.4.0/24   192.168.34.2             0             0 300 i

Displayed  4 routes and 4 total paths
frr-pc#
```

Figure 40. Verifying BGP networks on router r3.

Step 3. Router r2 will configure BGP *next-hop-self* so that the neighbor 192.168.23.2 (router r3) can reach the EBGP routes advertised by router r2, such as 192.168.1.0/24, through router r2. To do so, type the following command on router r2 terminal.

```
neighbor 192.168.23.2 next-hop-self
```

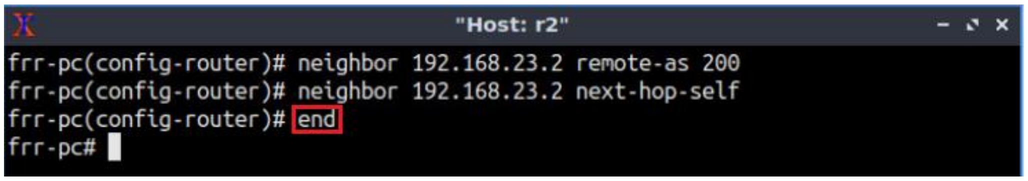


```
frr-pc(config-router)# neighbor 192.168.23.2 remote-as 200
frr-pc(config-router)# neighbor 192.168.23.2 next-hop-self
frr-pc(config-router)#
```

Figure 41. Changing BGP next hop in router r2.

Step 4. Type the following command to exit from configuration mode.

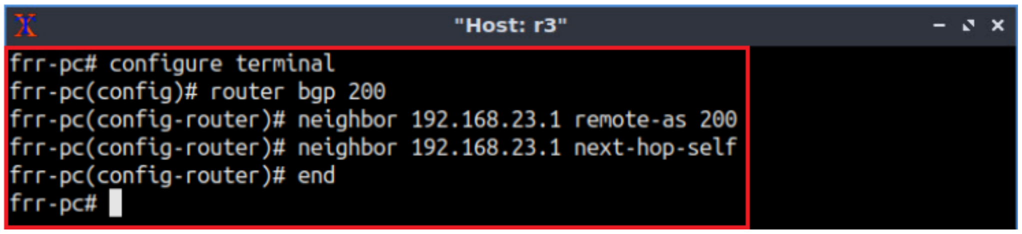
```
end
```



```
frr-pc(config-router)# neighbor 192.168.23.2 remote-as 200
frr-pc(config-router)# neighbor 192.168.23.2 next-hop-self
frr-pc(config-router)# end
frr-pc#
```

Figure 42. Exiting from configuration mode.

Step 5. In router r3 terminal, configure IBGP to peer with router r3. All the steps are summarized in the following figure.

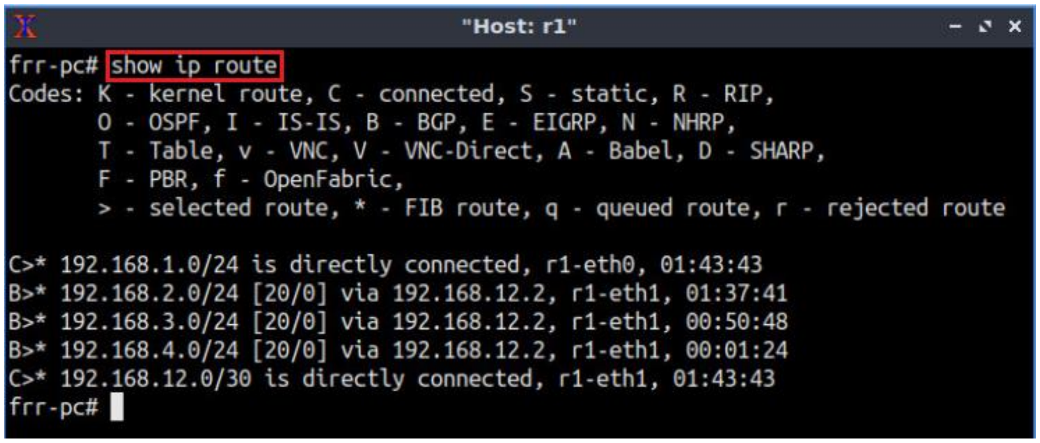


```
frr-pc# configure terminal
frr-pc(config)# router bgp 200
frr-pc(config-router)# neighbor 192.168.23.1 remote-as 200
frr-pc(config-router)# neighbor 192.168.23.1 next-hop-self
frr-pc(config-router)# end
frr-pc#
```

Figure 43. Configuring IBGP on router r3.

Step 6. Type the following command to verify the routing table of router r1. Router r1 has routes to the networks 192.168.3.0/24 and 192.168.4.0/24.

```
show ip route
```

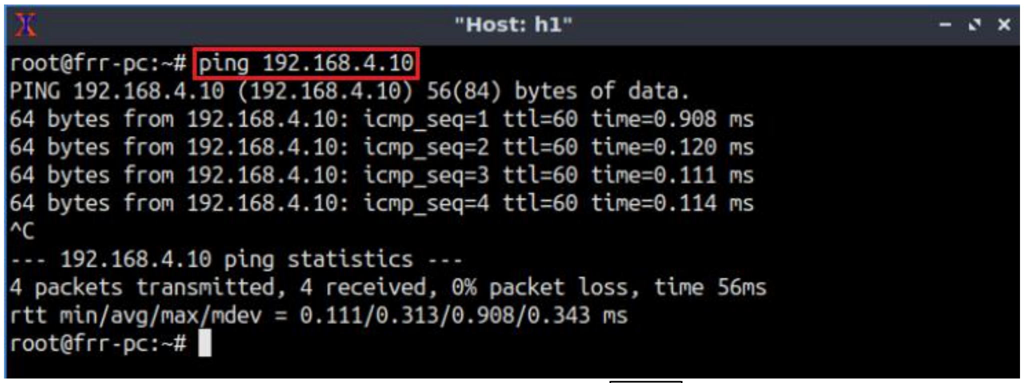


```
frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.1.0/24 is directly connected, r1-eth0, 01:43:43
B>* 192.168.2.0/24 [20/0] via 192.168.12.2, r1-eth1, 01:37:41
B>* 192.168.3.0/24 [20/0] via 192.168.12.2, r1-eth1, 00:50:48
B>* 192.168.4.0/24 [20/0] via 192.168.12.2, r1-eth1, 00:01:24
C>* 192.168.12.0/30 is directly connected, r1-eth1, 01:43:43
frr-pc#
```

Figure 44. Displaying the routing table of router r1.

Step 7. In host h1 terminal, perform a connectivity between host h1 and host h4 by issuing the command shown below. To stop the test, press `Ctrl+c`. The result will show a successful connectivity test.

```
ping 192.168.4.10
```



```
root@frr-pc:~# ping 192.168.4.10
PING 192.168.4.10 (192.168.4.10) 56(84) bytes of data.
64 bytes from 192.168.4.10: icmp_seq=1 ttl=60 time=0.908 ms
64 bytes from 192.168.4.10: icmp_seq=2 ttl=60 time=0.120 ms
64 bytes from 192.168.4.10: icmp_seq=3 ttl=60 time=0.111 ms
64 bytes from 192.168.4.10: icmp_seq=4 ttl=60 time=0.114 ms
^C
--- 192.168.4.10 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 56ms
rtt min/avg/max/mdev = 0.111/0.313/0.908/0.343 ms
root@frr-pc:~#
```

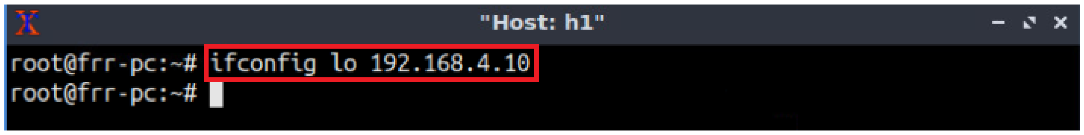
Figure 45. Connectivity test using `ping` command.

4 Perform IP spoofing and DoS attack

In this section, host h1 will spoof the IP address of host h4 and perform a DoS attack on it. Host h1 will send network traffic to hosts h2 and h3 with the source IP address of host h4. Thus, when hosts h2 and h3 receive the network traffic, they will reply to the source, i.e., host h4.

Step 1. Type the following command on h1 terminal. Host h1 is compromised and it will spoof the IP address of host h4 to perform a DoS against it. To do so, h1 sets an interface to the IP address of h4 first.

```
ifconfig lo 192.168.4.10
```

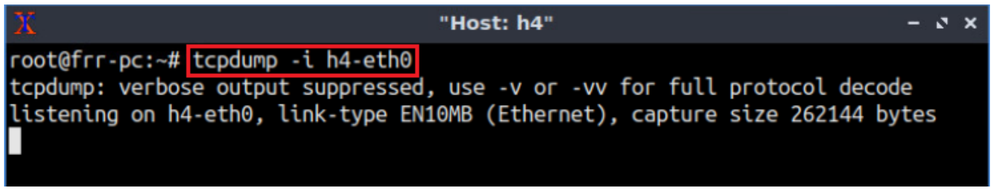


```
"Host: h1"
root@frr-pc:~# ifconfig lo 192.168.4.10
root@frr-pc:~# 
```

Figure 46. Assigning an IP address to the loopback interface.

Step 2. Type the following command on host h4 terminal. The command `tcpdump` allows you to capture the network traffic. The `-i` option allows you to specify the interface to be monitored (*h4-eth0*).

```
tcpdump -i h4-eth0
```



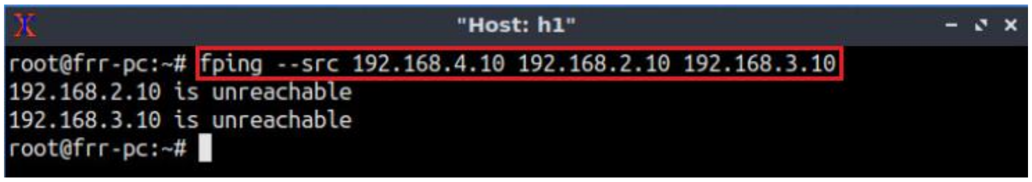
```
"Host: h4"
root@frr-pc:~# tcpdump -i h4-eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on h4-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

Figure 47. Capturing packets on interface h4-eth0.

Consider Figure 46. Currently, there is no network traffic directed on interface *eth0* of host h4. After launching the DoS attack from host h1, you will notice the network traffic redirected to host h4 using the *tcpdump* command.

Step 3. In host h1 terminal, issue the command shown below. The command used is `fping`. This command allows host h1 to ping multiple hosts. The `--src` option is followed by the source IP address. In this case, host h4 is configured with the source IP address (192.168.4.10). Then, the destinations IP addresses are specified, i.e. host h2 (192.168.2.10) and host h3 (192.168.3.10).

```
fping --src 192.168.4.10 192.168.2.10 192.168.3.10
```

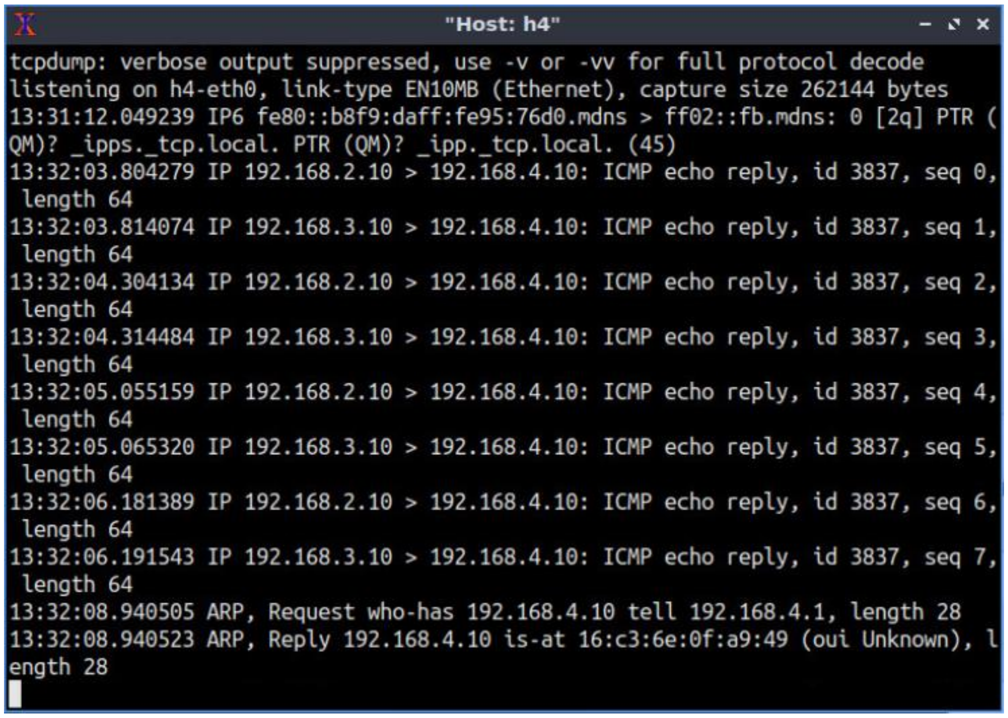


```
"Host: h1"
root@frr-pc:~# fping --src 192.168.4.10 192.168.2.10 192.168.3.10
192.168.2.10 is unreachable
192.168.3.10 is unreachable
root@frr-pc:~# 
```

Figure 48. Pinging hosts h2 and h3 from host h1 via the source IP address 192.168.4.10.

Consider Figure 47. The two hosts h2 and h3 are unreachable since they will not reply to host h1. Instead, they will reply to the source IP address 192.168.4.10 which is host h4. This is how host h1 performs a DoS attack using different hosts.

Step 4. In host h4 terminal, observe the network traffic redirected from host h2 (192.168.2.10) and host h3 (192.168.3.10) towards host h4 (192.168.4.10).



```
"Host: h4"
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on h4-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
13:31:12.049239 IP6 fe80::b8f9:daff:fe95:76d0.mdns > ff02::fb.mdns: 0 [2q] PTR (
QM)? _ipps._tcp.local. PTR (QM)? _ipp._tcp.local. (45)
13:32:03.804279 IP 192.168.2.10 > 192.168.4.10: ICMP echo reply, id 3837, seq 0,
 length 64
13:32:03.814074 IP 192.168.3.10 > 192.168.4.10: ICMP echo reply, id 3837, seq 1,
 length 64
13:32:04.304134 IP 192.168.2.10 > 192.168.4.10: ICMP echo reply, id 3837, seq 2,
 length 64
13:32:04.314484 IP 192.168.3.10 > 192.168.4.10: ICMP echo reply, id 3837, seq 3,
 length 64
13:32:05.055159 IP 192.168.2.10 > 192.168.4.10: ICMP echo reply, id 3837, seq 4,
 length 64
13:32:05.065320 IP 192.168.3.10 > 192.168.4.10: ICMP echo reply, id 3837, seq 5,
 length 64
13:32:06.181389 IP 192.168.2.10 > 192.168.4.10: ICMP echo reply, id 3837, seq 6,
 length 64
13:32:06.191543 IP 192.168.3.10 > 192.168.4.10: ICMP echo reply, id 3837, seq 7,
 length 64
13:32:08.940505 ARP, Request who-has 192.168.4.10 tell 192.168.4.1, length 28
13:32:08.940523 ARP, Reply 192.168.4.10 is-at 16:c3:6e:0f:a9:49 (oui Unknown), l
ength 28
```

Figure 49. Monitoring network traffic on host h4.

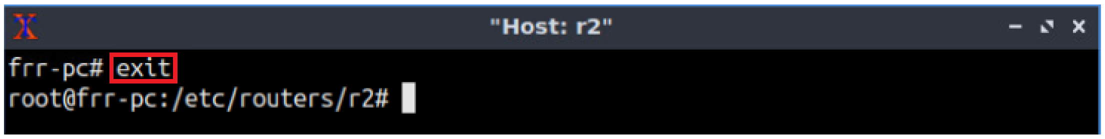
To interrupt capturing the network traffic on interface *eth0* of host h4 press `Ctrl+c`.

5 Mitigate DDoS attack by using IP source filtering

In this section, you will configure the ISP (router r2) to filter network traffic of Campus-1 based on their source IP addresses. Thus, mitigating IP spoofing and its possible attacks, such as DoS. To filter network traffic based on the source IP address, iptables utility will be used as FRR doesn’t support this feature. Iptables is a command line program used to configure packet filtering on Linux operating systems.

Step 1. In router r2 terminal, type the following command to exit the vtysh session:

```
exit
```

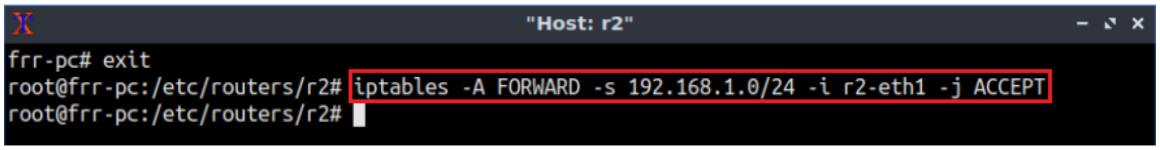


```
"Host: r2"
frr-pc# exit
root@frr-pc:/etc/routers/r2#
```

Figure 50. Exiting the vtysh session.

Step 2. Type the following command on router r2 terminal to add a filtering rule. The option `-A FORWARD` specifies that the rule added corresponds to incoming connections that are not being delivered locally. The option `-s` is used to specify the source IP address of the traffic. The option `-i` is used to specify the input interface receiving the traffic (*r2-eth1*). The option `-j` is to specify what to do if the packet matches. Briefly, in this command we are inserting a rule to accept all incoming packets on interface *r2-eth1* (facing Campus-1) having a source IP address that belongs to the subnet 192.168.1.0/24.

```
iptables -A FORWARD -s 192.168.1.0/24 -i r2-eth1 -j ACCEPT
```

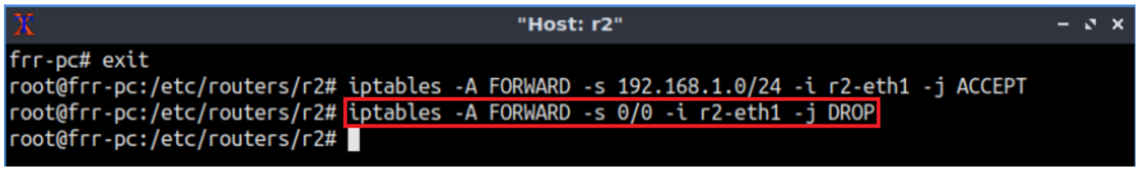


```
frr-pc# exit
root@frr-pc:/etc/routers/r2# iptables -A FORWARD -s 192.168.1.0/24 -i r2-eth1 -j ACCEPT
root@frr-pc:/etc/routers/r2# 
```

Figure 51. Adding a filtering rule on router r2.

Step 3. Similarly, to add another filtering rule on router r2, type the following command. The `-s 0/0` option matches all IP addresses. Briefly, we are dropping all incoming packets on interface *r2-eth1*.

```
iptables -A FORWARD -s 0/0 -i r2-eth1 -j DROP
```



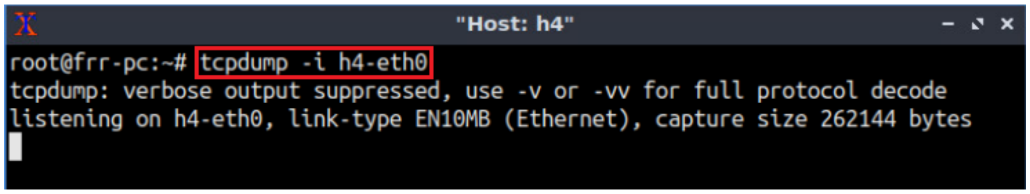
```
frr-pc# exit
root@frr-pc:/etc/routers/r2# iptables -A FORWARD -s 192.168.1.0/24 -i r2-eth1 -j ACCEPT
root@frr-pc:/etc/routers/r2# iptables -A FORWARD -s 0/0 -i r2-eth1 -j DROP
root@frr-pc:/etc/routers/r2# 
```

Figure 52. Adding a filtering rule on router r2.

After adding two filters on router r2, all incoming packets on interface *r2-eth1* will be permitted if their IP address belongs to the subnet 192.168.1.0/24. Otherwise, the packets will be dropped and not forwarded to their destination.

Step 4. We will launch another DoS attack from host h1 on host h4 and validate that the attack is mitigated. On host h4 terminal, type the following command to capture the network traffic on interface *h4-eth0*.

```
tcpdump -i h4-eth0
```

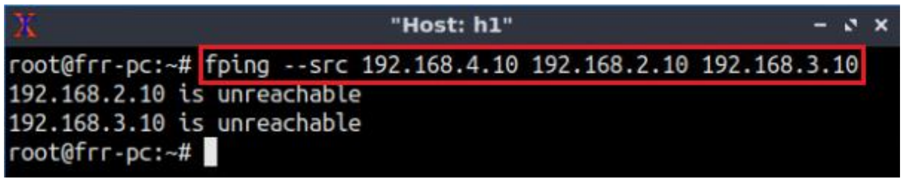


```
root@frr-pc:~# tcpdump -i h4-eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on h4-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

Figure 53. Capturing packets on interface h4-eth0.

Step 5. On host h1 terminal, use the command `fping` to ping hosts h2 (192.168.2.10) and h3 (192.168.3.10) from the source IP address 192.168.4.10.

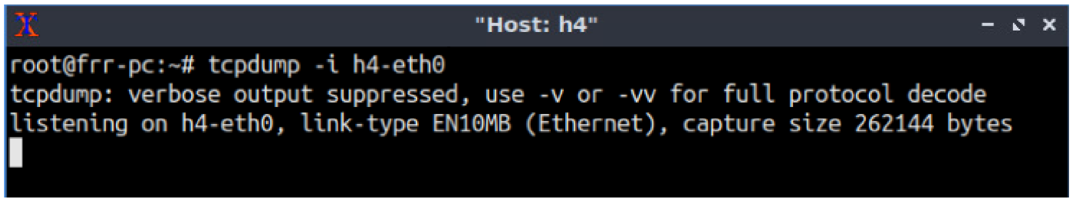
```
fping --src 192.168.4.10 192.168.2.10 192.168.3.10
```



```
root@frr-pc:~# fping --src 192.168.4.10 192.168.2.10 192.168.3.10
192.168.2.10 is unreachable
192.168.3.10 is unreachable
root@frr-pc:~# 
```

Figure 54. Pinging hosts h2 and h3 from host h1 via the source IP address 192.168.4.10.

Step 6. In host h4 terminal, observe that even after the DoS attack was performed from host h1, host h4 did not receive any packet. Thus, the ISP (router r2) was able to filter the network traffic based on the source IP address and mitigate IP spoofing attacks.



```
root@frr-pc:~# tcpdump -i h4-eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on h4-eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

Figure 55. Monitoring network traffic on host h4.

To interrupt capturing the network traffic on interface *eth0* of host h4 press `Ctrl+c`.

This concludes Lab 2. Stop the emulation and then exit out of MiniEdit.

References

1. A. Tanenbaum, D. Wetherall, "Computer networks", 5th Edition, Pearson, 2012.
2. MANRS, "Anti-Spoofing". [Online]. Available: https://www.manrs.org/isps/guide/antispoofing/
3. C. Douligeris, A. Mitrokotsa, "DDoS attacks and defense mechanisms: classification and state-of-the-art", [Online]. Available: https://reader.elsevier.com/reader/sd/pii/S1389128603004250?token=825EB28028CD30A735627A835E87DF874474EF4072D273D6C9F0B0B58712597778BFDF230132C2FA8A8E082D370CAE51
4. Ciscopress, "BGP Fundamentals". [Online]. Available: https://www.ciscopress.com/articles/article.asp?p=2756480&seqNum=6
5. Netfilter, "The netfilter.org project". [Online]. Available: https://netfilter.org/

UNIVERSITY OF SOUTH CAROLINA

MPLS AND ADVANCED BGP TOPICS

Lab 3: BGP Hijacking

Document Version: 03-12-2020

Award 1829698
“CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers”

Contents

Overview

This lab presents Border Gateway Protocol (BGP) hijacking attack that occurs on the Internet between different Autonomous Systems (ASes). In this lab, a compromised host will hijack the Internet Protocol (IP) address of a victim and advertise this address to its BGP routers. Thus, the network traffic destined to the victim will be rerouted to the attacker. The goal of this lab is to configure the Internet Service Provider (ISP) to mitigate BGP hijacking attacks by applying IP prefix filters on the network traffic of its customers.

Objectives

By the end of this lab, students should be able to:

1. Configure BGP as the main protocol between ASes.
2. Understand and perform BGP hijacking.
3. Understand BGP hijacking mitigation techniques.
4. Apply IP prefix filters to counter BGP hijacking.

Lab settings

The information in Table 1 provides the credentials to access Client machine.

Table 1**.** Credentials to access Client machine.

Device	**Account**	**Password**
Client	admin	password

Lab roadmap

This lab is organized as follows:

1. Section 1: Introduction.
2. Section 2: Lab topology.
3. Section 3: Configure BGP on routers.
4. Section 4: BGP hijacking.
5. Section 5: BGP hijacking mitigation using IP prefix filtering.

1 Introduction

1.1 BGP overview

BGP is an exterior gateway protocol designed to exchange routing and reachability information among ASes on the Internet. BGP is relevant to network administrators of large organizations which connect to one or more ISPs, as well as to ISPs who connect to other network providers. In terms of BGP, an AS is referred to as a routing domain, where all networked systems operate common routing protocols and are under the control of a single administration[1].

BGP is a form of distance vector protocol. It requires each router to maintain a table, which stores the distance and the output interface (i.e., vector) to remote networks. BGP makes routing decisions based on paths, network policies, or rule set configured by a network administrator and is involved in making core routing decisions[1].

Two routers that establish a BGP connection are referred to as BGP peers or neighbors**.** BGP sessions run over Transmission Control Protocol (TCP). If a BGP session is established between two neighbors in different ASes, the session is referred to as an External BGP (EBGP) session. If the session is established between two neighbors in the same AS, the session is referred to as Internal BGP (IBGP)[1]. Figure 1 shows a network running BGP protocol. Routers that exchange information within the same AS use IBGP, while routers that exchange information between different ASes use EBGP.

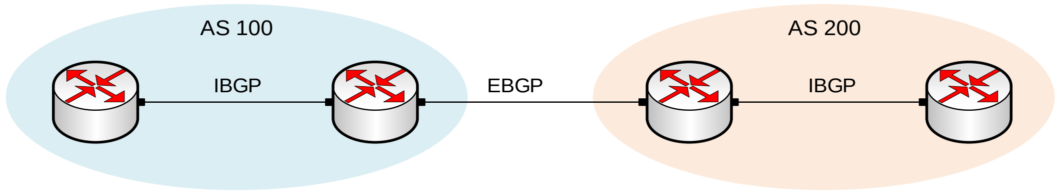


Figure 1. Routers that exchange information within the same AS use IBGP, while routers that exchange information between different ASes use EBGP.

1.2 BGP hijacking

BGP exchanges routing and reachability information among ASes. By default, when routers that have established BGP peering relationship trust each other. Consequently, any IP prefix announced by a router is accepted by its neighbors. However, the Internet is not always ideal, unauthorized network can originate IP prefix owned by other networks to divert traffic for those prefixes towards the unauthorized network[2]. This process is known as BGP hijacking.

Consider Figure 2. The IP address 192.168.3.10 matches the networks 192.168.3.0/25 advertised by router r1, and 192.168.3.0/24 advertised by router 3. To ping the network 192.168.3.10, router r2 prefers the route advertised by router r1 (hijacking router) over router r3 (legitimate router). This decision is made since router r1 advertises a more specific announcement (/25) than router r3 (/24).

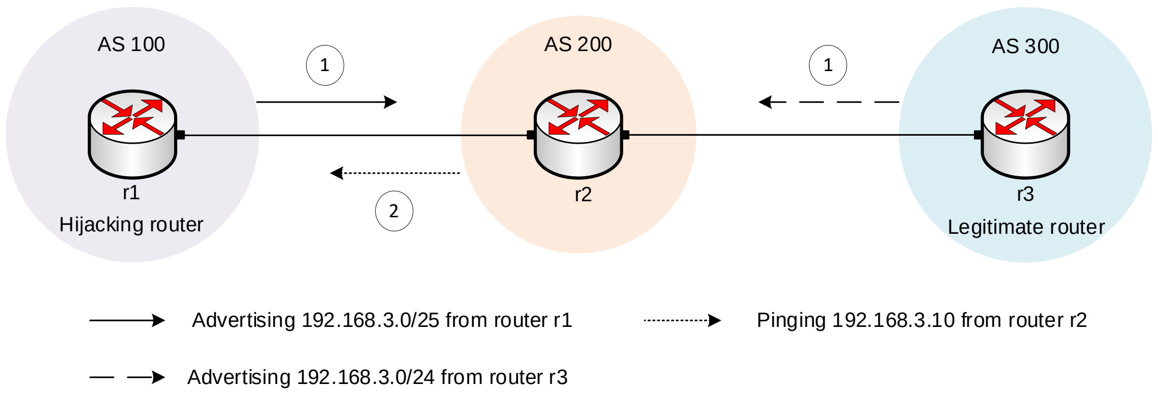

AS 100
r1
Hijacking router
AS 200
r2
AS 300
r3
Legitimate router
1
1
2
Advertising 192.168.3.0/25 from router r1
Pinging 192.168.3.10 from router r2
Advertising 192.168.3.0/24 from router r3

Figure 2. BGP hijacking using specific IP prefix advertisement.

Consider Figure 3. Router r1 (hijacking router) and router r4 (legitimate router) advertise the same network (192.168.4.0/24). To ping the IP address 192.168.4.10, router r2 prefers the route advertised by router r1 (hijacking router) over router r4 (legitimate router). This decision is made since the path to router r1 is shorter than that to router r4.

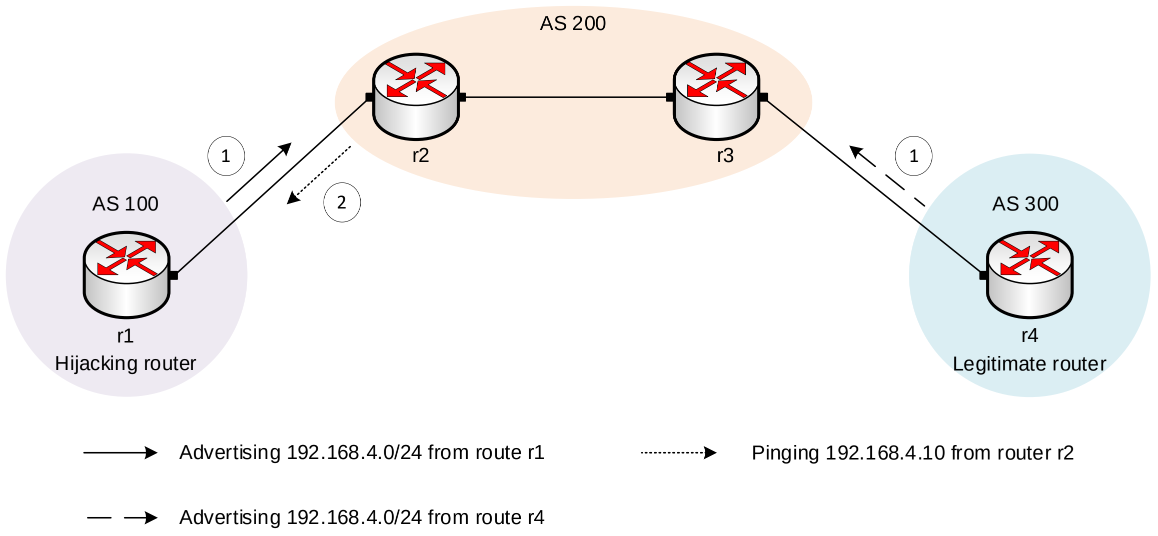

AS 200
r2
r3
AS 100
r1
Hijacking router
AS 300
r4
Legitimate router
1
2
1
Advertising 192.168.4.0/24 from route r1
Pinging 192.168.4.10 from router r2
Advertising 192.168.4.0/24 from route r4

Figure 3. BGP hijacking using shorter path advertisement.

1.3 BGP hijacking mitigation techniques

Mutually Agreed Norms for Routing Security (MANRS) is a global initiative, supported by the Internet Society, that provides crucial fixes to reduce the most common routing threats. MANRS has many recommendations to prevent propagation of incorrect routing information, such as Resource Public Key Infrastructure (RPKI)[3].

1.3.1 Using RPKI to validate route origins

Since any route can be originated and announced by any random network, there needs to be a method to manage BGP advertisements. RPKI is a cryptographic method to support improved security of Internet routing. It enables an entity to verifiably assert that it is the legitimate holder of a set of IP addresses or a set of AS numbers[4].

Consider Figure 4. When router r2 receives route advertisements (1), it contacts the RPKI-enabled server to validate route advertisements. This server in turns sends if the received routes are valid or not based on stored cryptographic information about each entity (2). After receiving the response from the RPKI-enabled server, router r2 will ping the network 192.168.3.10 via the legitimate router (router r3).

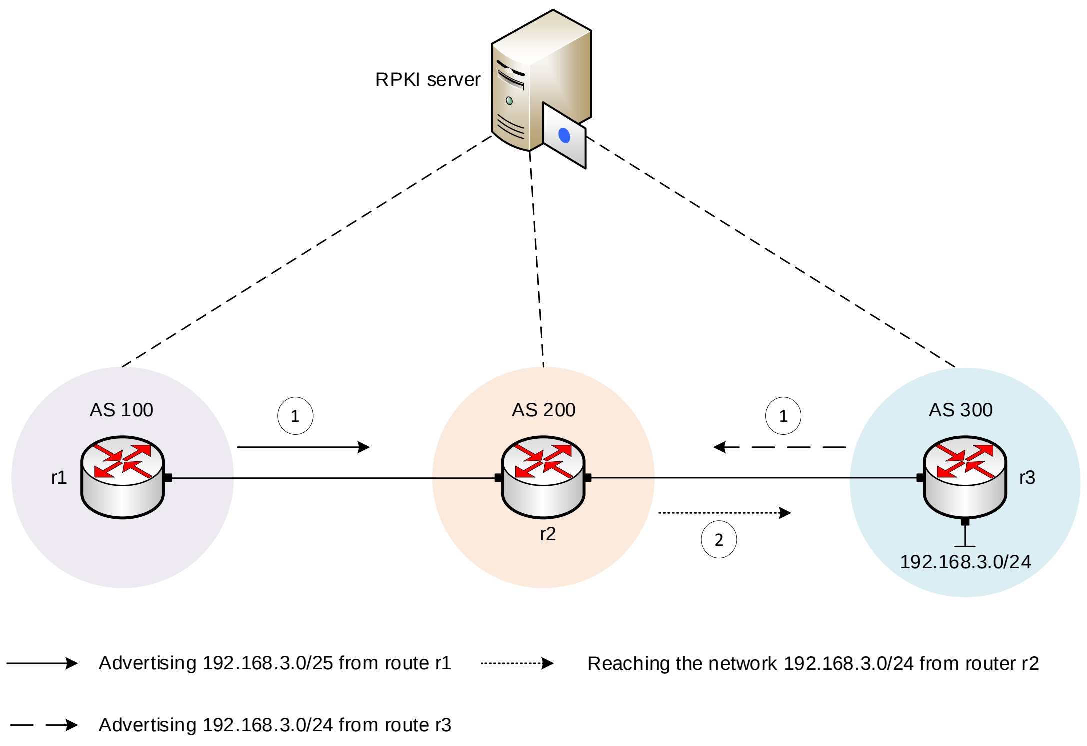

RPKI server
AS 100
r1
1
AS 200
r2
1
2
AS 300
r3
192.168.3.0/24
Advertising 192.168.3.0/25 from route r1
Reaching the network 192.168.3.0/24 from router r2
Advertising 192.168.3.0/24 from route r3

Figure 4. Using RPKI to validate IP prefix advertisements

1.3.2 BGP prefix filters

A router can limit the number of BGP route advertisements by configuring IP prefix filters. Prefix filtering can be applied to inbound and outbound advertisements.

Consider Figure 5. The network 192.168.1.0/24 belongs to its Customer in AS 100. The ISP applies BGP prefix filters to allow only the advertisement of this network. Thus, if router r1 tries to hijack other IP prefixes, the ISP will prevent such action.

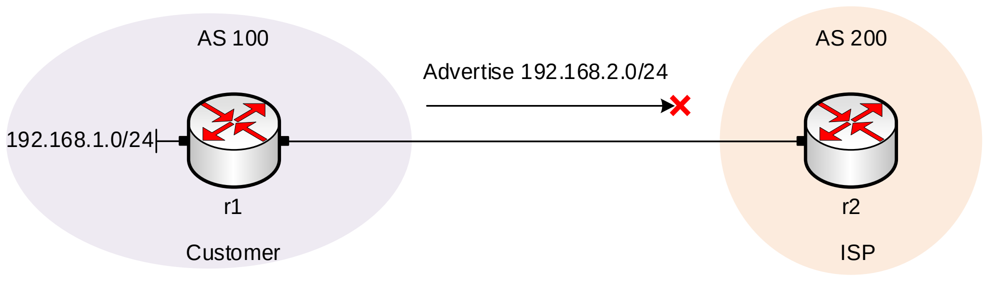


Figure 5. Applying IP prefix filters to prevent hijacked IP prefixes.

Although using RPKI is highly recommended to validate advertised IP prefixes, in this lab, we will apply BGP prefix filters as they are more feasible and easier to be implemented.

2 Lab topology

Consider Figure 6. The topology consists of three ASes. The ISP, consisting of routers r3 and r4, provides Internet service to the Campus-1 (router r1) and Campus-2 (router r2) networks. The Autonomous System Numbers (ASNs) assigned to Campus-1, ISP, and Campus-2 are 100, 200, and 300, respectively. The ISP communicates with the Campus networks via EBGP routing protocol, and the routers within the ISP communicate using IBGP. Host h1 in Campus-1 hijacks the network address assigned to Campus-2. Thus, all the traffic destined to Campus-2 and passign through router r3 will be rerouted to router r1.

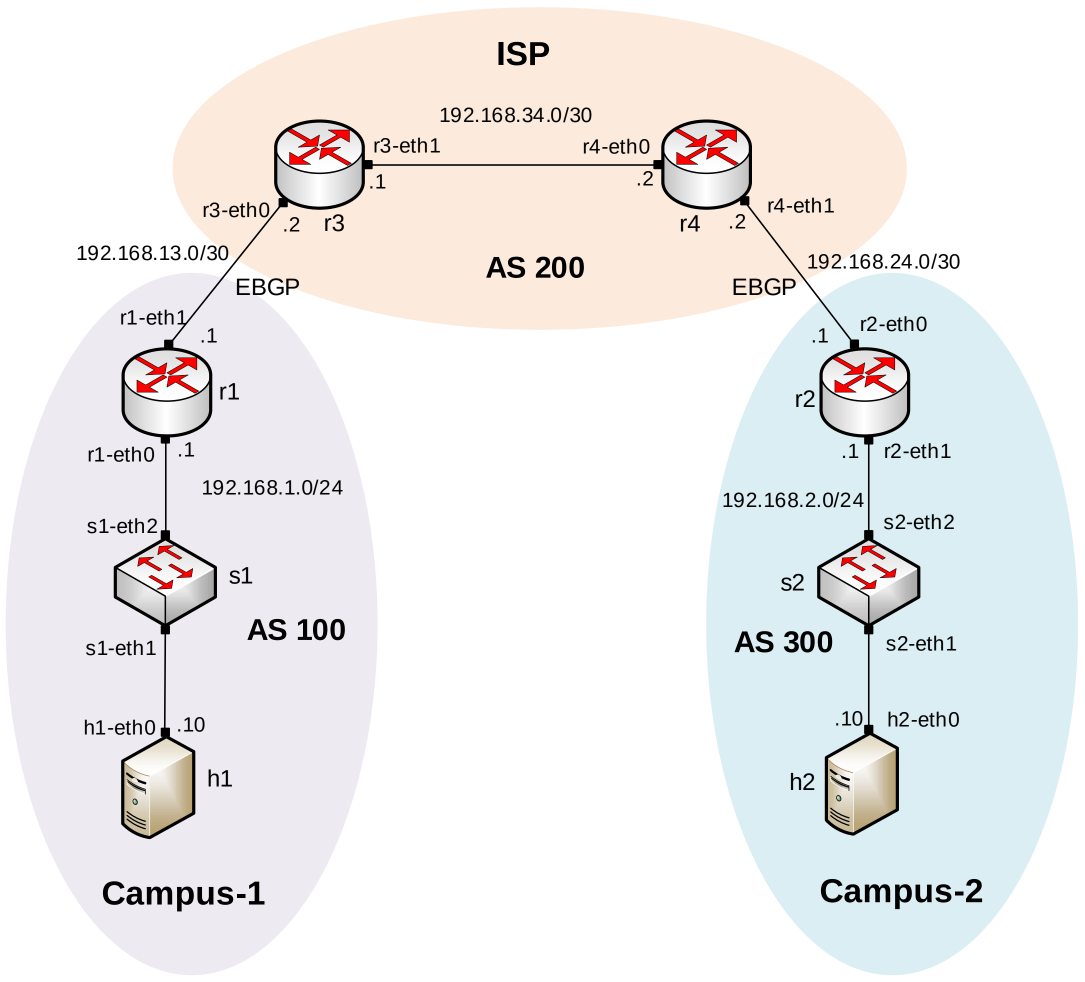


Figure 6. Lab topology.

2.1 Lab settings

Routers and hosts are already configured according to the IP addresses shown in Table 2.

Table 2. Topology information.

Device	Interface	IPV4 Address	Subnet	Default gateway
r1 (Campus-1)	r1-eth0	192.168.1.1	/24	N/A
	r1-eth1	192.168.13.1	/30	N/A
r2 (Campus-2)	r2-eth0	192.168.2.1	/24	N/A
	r2-eth1	192.168.24.1	/30	N/A
r3 (ISP)	r3-eth0	192.168.13.2	/30	N/A
	r3-eth1	192.168.34.1	/30	N/A
	r4-eth0	192.168.34.2	/30	N/A

r4 (ISP)	r4-eth1	192.168.24.2	/30	N/A
h1	h1-eth0	192.168.1.10	/24	192.168.1.1
h2	h2-eth0	192.168.2.10	/24	192.168.2.1

2.2 Open the topology

Step 1. Start by launching Miniedit by clicking on Desktop's shortcut. When prompted for a password, type `password`.

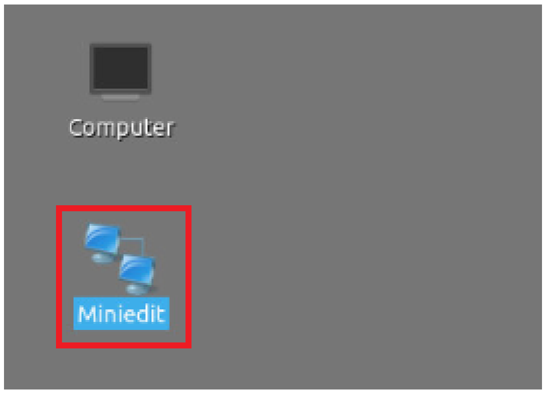


Figure 7. Miniedit shortcut.

Step 2. On Miniedit's menu bar, click on *File* then *open* to load the lab's topology. Locate the *Lab3.mn* topology file in the default directory, */home/frr/MPLS_advanced_BGP/lab3* and click on *Open*.

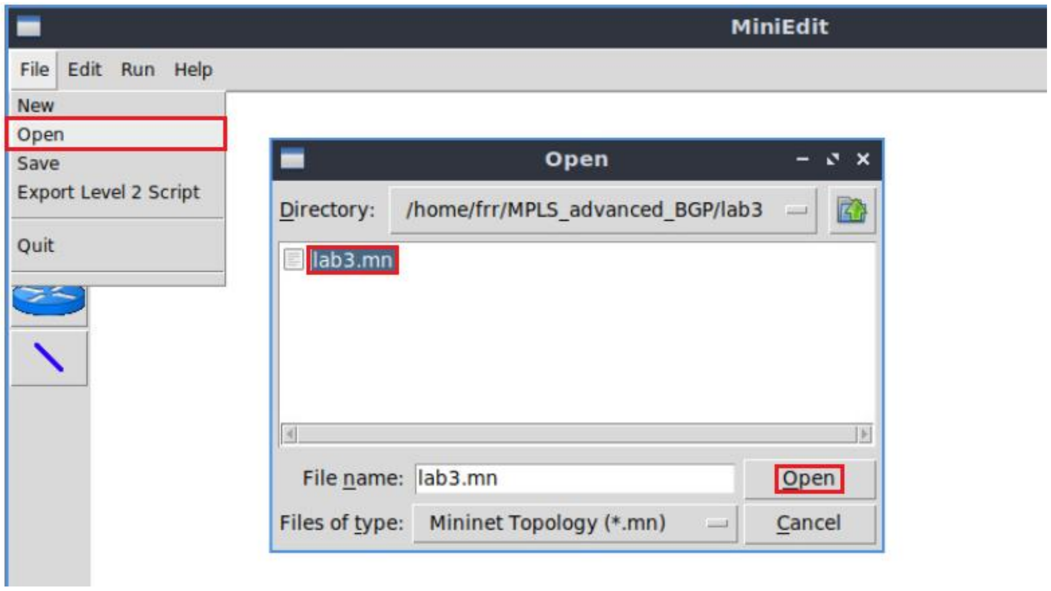


Figure 8. MiniEdit's open dialog.

At this point the topology is loaded with all the required network components. You will execute a script that will load the configuration of the routers.

Step 3. Open the Linux terminal.

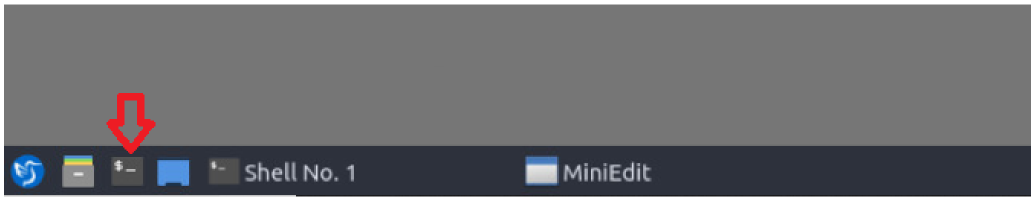
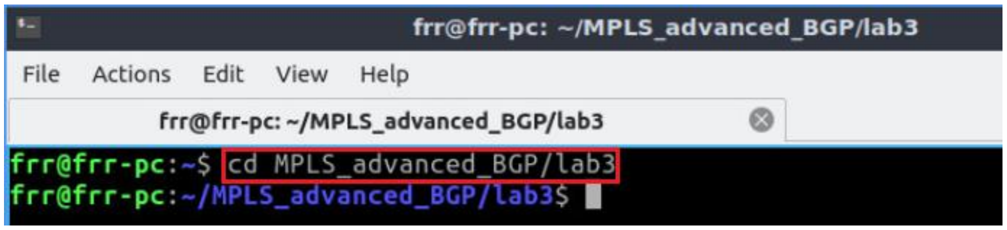


Figure 9. Opening Linux terminal.

Step 4. Click on the Linux's terminal and navigate into *MPLS_advanced_BGP/lab3* directory by issuing the following command. This folder contains a configuration file and the script responsible for loading the configuration. The configuration file will assign the IP addresses to the routers' interfaces. The `cd` command is short for change directory followed by an argument that specifies the destination directory.

```
cd MPLS_advanced_BGP/lab3
```

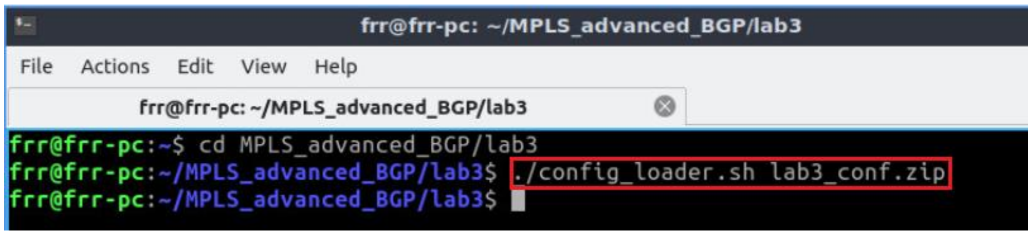

frr@frr-pc: ~/MPLS_advanced_BGP/lab3
File
Actions
Edit
View
Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab3

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab3
frr@frr-pc:~/MPLS_advanced_BGP/lab3$
```

Figure 10. Entering to the *MPLS_advanced_BGP/lab3* directory.

Step 5. To execute the shell script, type the following command. The argument of the program corresponds to the configuration zip file that will be loaded in all the routers in the topology.

```
./config_loader.sh lab3_conf.zip
```

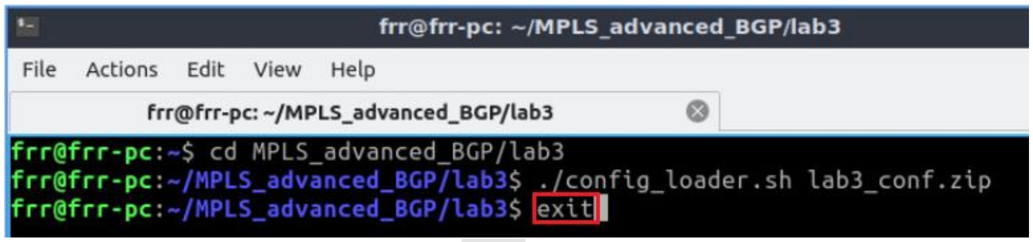

frr@frr-pc: ~/MPLS_advanced_BGP/lab3
File
Actions
Edit
View
Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab3

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab3
frr@frr-pc:~/MPLS_advanced_BGP/lab3$ ./config_loader.sh lab3_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab3$
```

Figure 11. Executing the shell script to load the configuration.

Step 6. Type the following command to exit the Linux terminal.

```
exit
```


frr@frr-pc: ~/MPLS_advanced_BGP/lab3
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab3

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab3
frr@frr-pc:~/MPLS_advanced_BGP/lab3$ ./config_loader.sh lab3_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab3$ exit
```

Figure 12. Using `exit` to exit the terminal.

Step 7. At this point hosts h1 and h2 interfaces are configured. To proceed with the emulation, click on the *Run* button located in lower left-hand side.

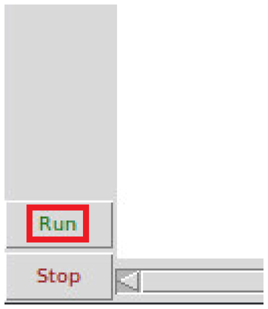


Figure 13. Starting the emulation.

Step 8. In Mininet's terminal, i.e., the one launched when MiniEdit was started.

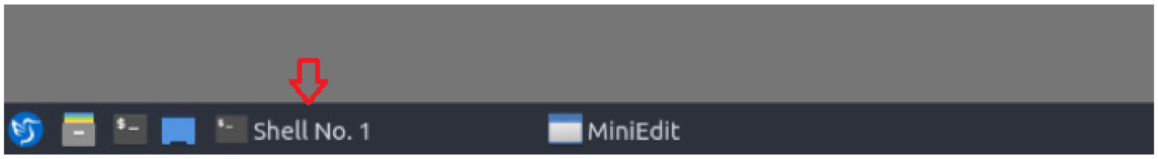
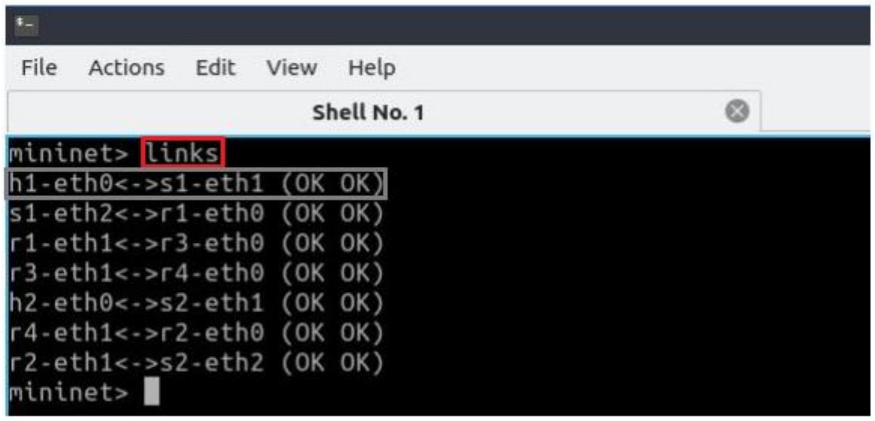


Figure 14. Opening Mininet's terminal.

Step 9. Issue the following command to display the interface names and connections.

```
links
```



```
mininet> links
h1-eth0<->s1-eth1 (OK OK)
s1-eth2<->r1-eth0 (OK OK)
r1-eth1<->r3-eth0 (OK OK)
r3-eth1<->r4-eth0 (OK OK)
h2-eth0<->s2-eth1 (OK OK)
r4-eth1<->r2-eth0 (OK OK)
r2-eth1<->s2-eth2 (OK OK)
mininet>
```

Figure 15. Displaying network interfaces.

In Figure 15, the link displayed within the gray box indicates that interface *eth0* of host h1 connects to interface *eth1* of switch s1 (i.e., *h1-eth0<->s1-eth1*).

2.3 Load zebra daemon and verify configuration

You will verify that the IP addresses listed in Table 2 and inspect the routing table of routers r1, r2, r3, and r4.

Step 1. Hold right-click on host h1 and select *Terminal*. This opens the terminal of host h1 and allows the execution of commands on that host.

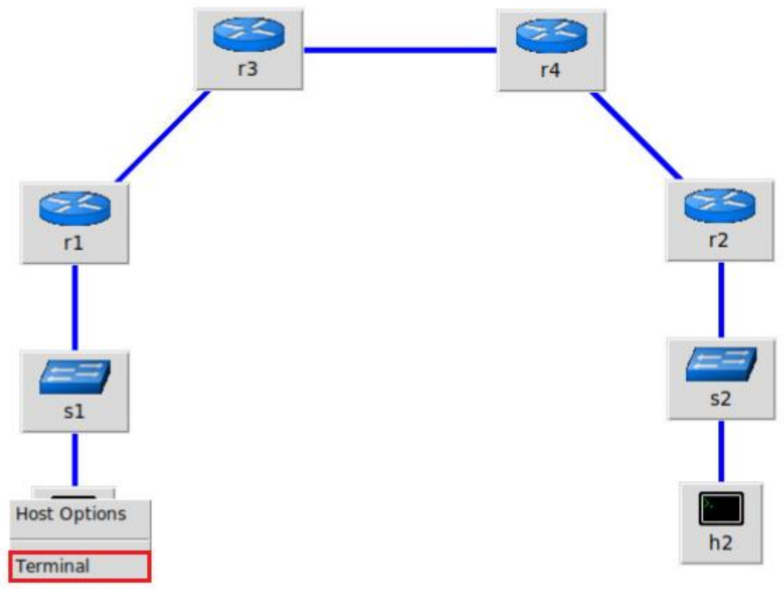
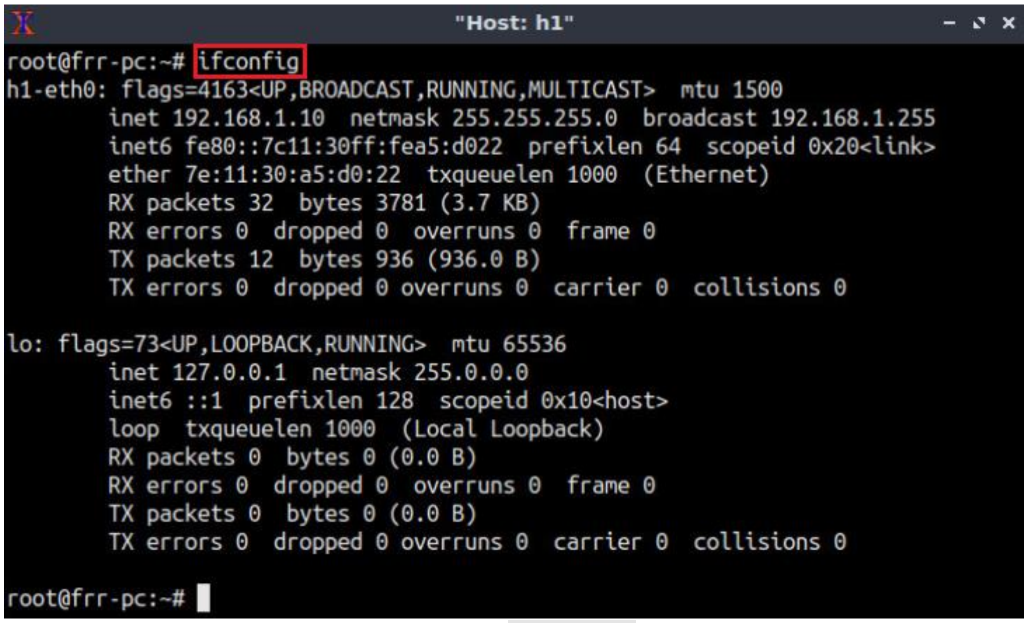


Figure 16. Opening a terminal on host h1.

Step 2. On h1 terminal, type the command shown below to verify that the IP address was assigned successfully. You will verify that host h1 has two interfaces, *h1-eth0* configured with the IP address 192.168.1.10 and the subnet mask 255.255.255.0.

```
ifconfig
```

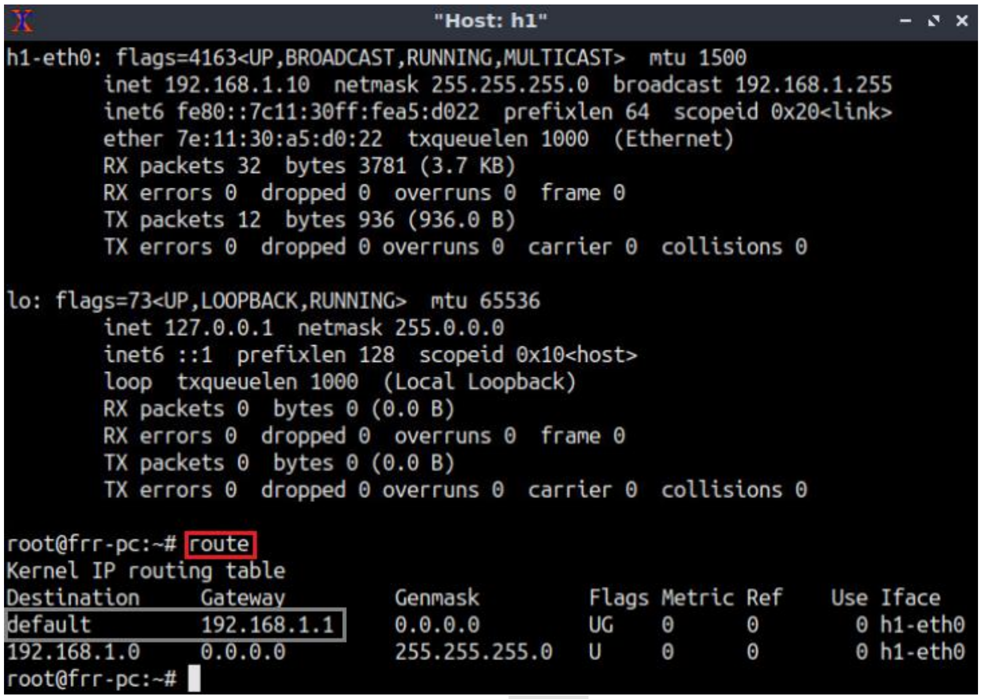


```
root@frr-pc:~# ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.10  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::7c11:30ff:fea5:d022  prefixlen 64  scopeid 0x20<link>
        ether 7e:11:30:a5:d0:22  txqueuelen 1000  (Ethernet)
        RX packets 32  bytes 3781 (3.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12  bytes 936 (936.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@frr-pc:~#
```

Figure 17. Output of `ifconfig` command.

Step 3. On host h1 terminal, type the command shown below to verify that the default gateway IP address is 192.168.1.1.

```
route
```



```
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.10  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::7c11:30ff:fea5:d022  prefixlen 64  scopeid 0x20<link>
        ether 7e:11:30:a5:d0:22  txqueuelen 1000  (Ethernet)
        RX packets 32  bytes 3781 (3.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12  bytes 936 (936.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@frr-pc:~# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         192.168.1.1     0.0.0.0         UG    0      0        0 h1-eth0
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 h1-eth0
root@frr-pc:~#
```

Figure 18. Output of `route` command.

Step 4. In order to verify host h2, proceed similarly by repeating from step 1 to step 3 on host h2 terminal. Similar results should be observed.

Step 5. You will validate that the router interfaces are configured correctly according to Table 2. To proceed, hold right-click on r1 and select *Terminal*.

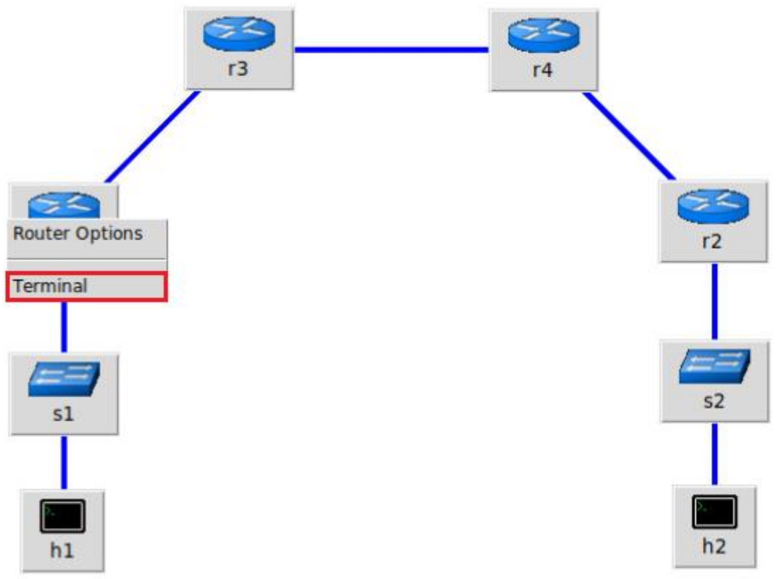
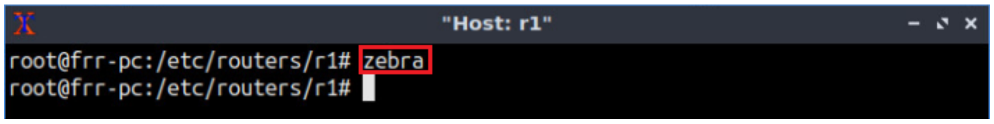


Figure 19. Opening a terminal on router r1.

Step 6. In this step, you will start zebra daemon, which is a multi-server routing software that provides TCP/IP based routing protocols. The configuration will not be working if you do not enable zebra daemon initially. In order to start the zebra, type the following command:

```
zebra
```

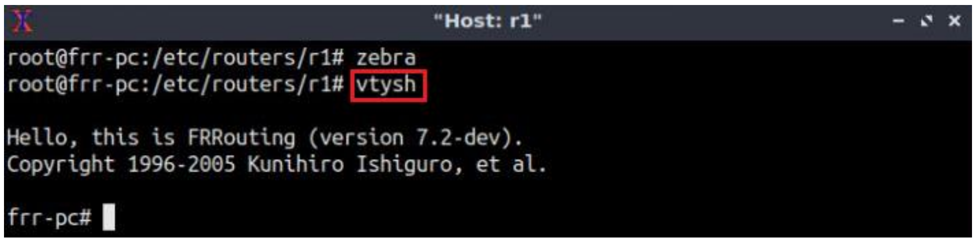


```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# 
```

Figure 20. Starting zebra daemon.

Step 7. After initializing zebra, vtysh is started in order to provide all the CLI commands defined by the daemons in a single shell. To proceed, issue the following command:

```
vtysh
```

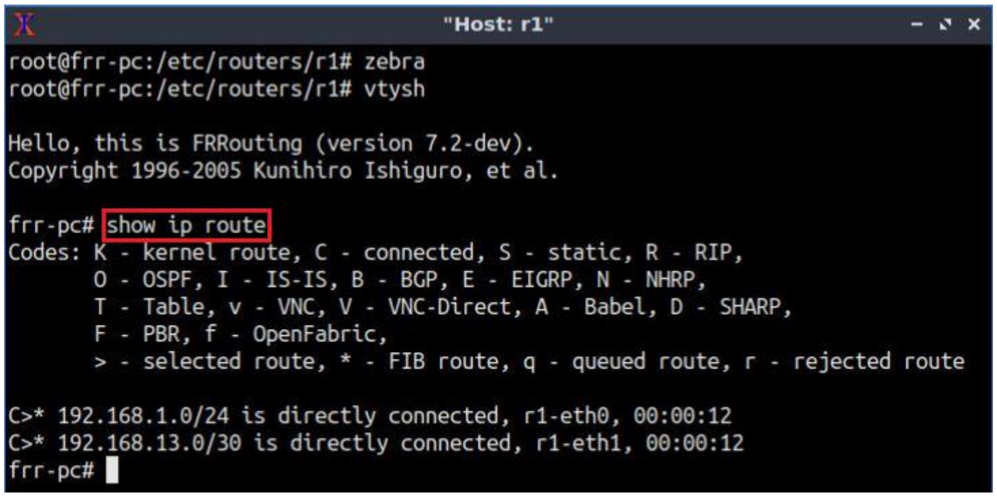


```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# 
```

Figure 21. Starting vtysh on router r1.

Step 8. Type the following command on router r1 terminal to verify the routing table of router r1. It will list all the directly connected networks. The routing table of router r1 does not contain any route to the network attached to router r2 (192.168.2.0/24) and router r4 (192.168.24.0/30, 192.168.34.0/30) as there is no routing protocol configured yet.

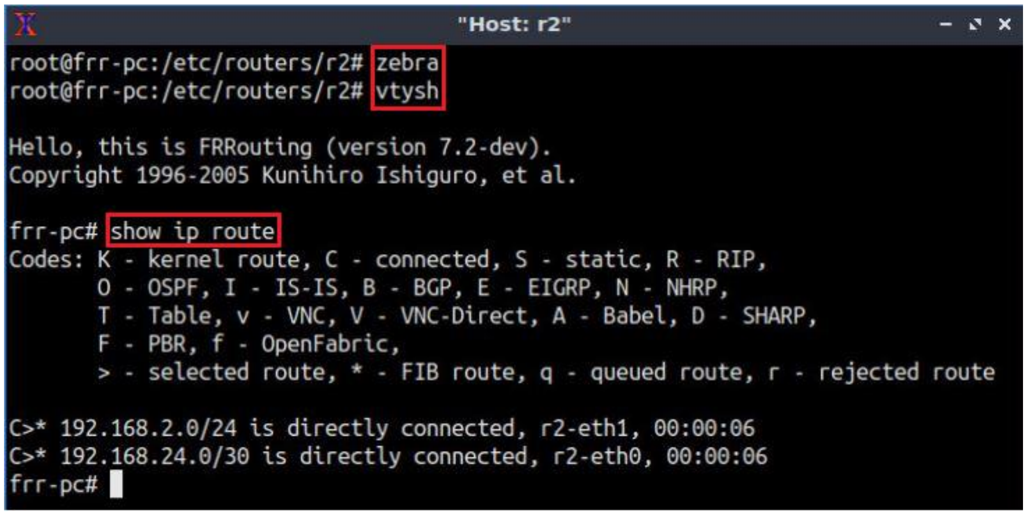
```
show ip route
```



```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.1.0/24 is directly connected, r1-eth0, 00:00:12
C>* 192.168.13.0/30 is directly connected, r1-eth1, 00:00:12
frr-pc#
```

Figure 22. Displaying routing table of router r1.

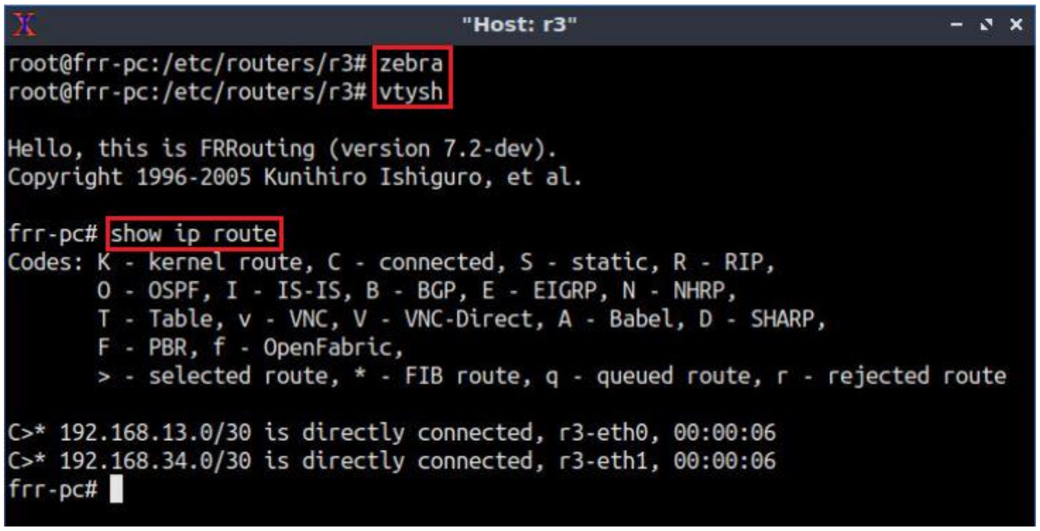
Step 9. Router r2 is configured similarly to router r1 but, with different IP addresses (see Table 2). Those steps are summarized in the following figure. To proceed, in router r2 terminal issue the commands depicted below. At the end, you will verify all the directly connected networks of router r2.



```
"Host: r2"
root@frr-pc:/etc/routers/r2# zebra
root@frr-pc:/etc/routers/r2# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.2.0/24 is directly connected, r2-eth1, 00:00:06
C>* 192.168.24.0/30 is directly connected, r2-eth0, 00:00:06
frr-pc#
```

Figure 23. Displaying routing table of router r2.

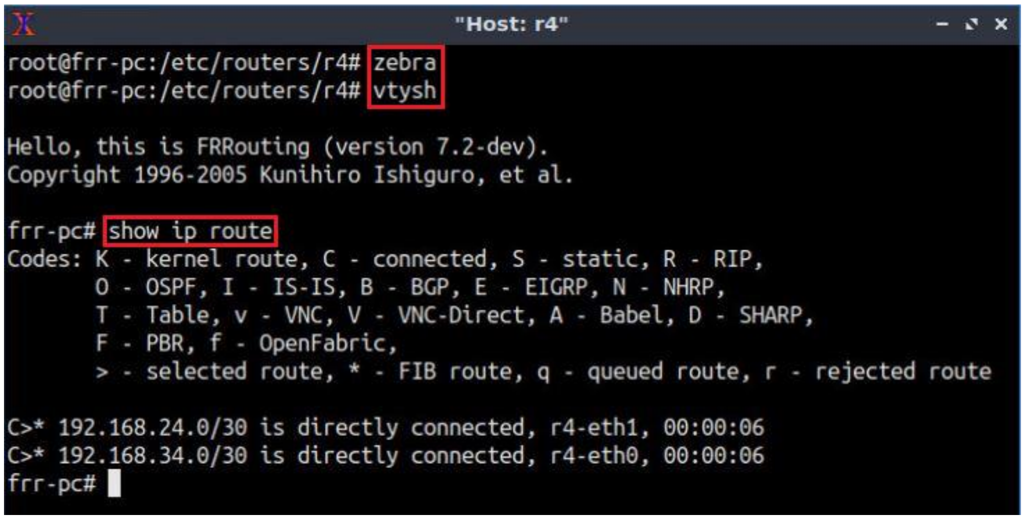
Step 10. Router r3 is configured similarly to router r1 but, with different IP addresses (see Table 2). Those steps are summarized in the following figure. To proceed, in router r3 terminal, issue the commands depicted below. At the end, you verify all the directly connected networks of router r3.



```
root@frr-pc:/etc/routers/r3# zebra
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.13.0/30 is directly connected, r3-eth0, 00:00:06
C>* 192.168.34.0/30 is directly connected, r3-eth1, 00:00:06
frr-pc#
```

Figure 24. Displaying routing table of router r3.

Step 11. Router r4 is configured similarly to router r1 but, with different IP addresses (see Table 2). Those steps are summarized in the following figure. To proceed, in router r4 terminal, issue the commands depicted below. At the end, you verify all the directly connected networks of router r4.



```
root@frr-pc:/etc/routers/r4# zebra
root@frr-pc:/etc/routers/r4# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.24.0/30 is directly connected, r4-eth1, 00:00:06
C>* 192.168.34.0/30 is directly connected, r4-eth0, 00:00:06
frr-pc#
```

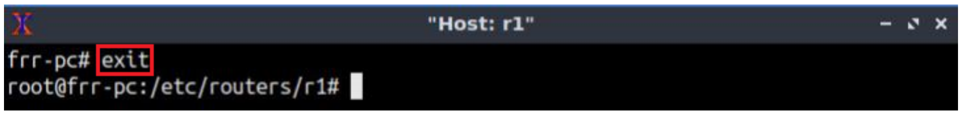
Figure 25. Displaying routing table of router r4.

3 Configure BGP on routers

In this section, you will configure EBGP on the routers that are hosted in different ASes. You will assign BGP neighbors to allow the routers to exchange BGP routes. Furthermore, routers r1 and r2 will advertise their Local Area Networks (LANs) via BGP. Therefore, router r3 and router r4 will receive route information about LAN 192.168.1.0/24 and 192.168.2.0/24, respectively.

Step 1. To configure BGP routing protocol, you need to enable the BGP daemon first. In router r1, type the following command to exit the vtysh session:

```
exit
```

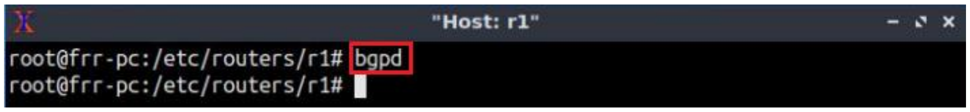


```
"Host: r1"
frr-pc# exit
root@frr-pc:/etc/routers/r1#
```

Figure 26. Exiting the vtysh session.

Step 2. Type the following command on router r1 terminal to enable and to start BGP routing protocol.

```
bgpd
```

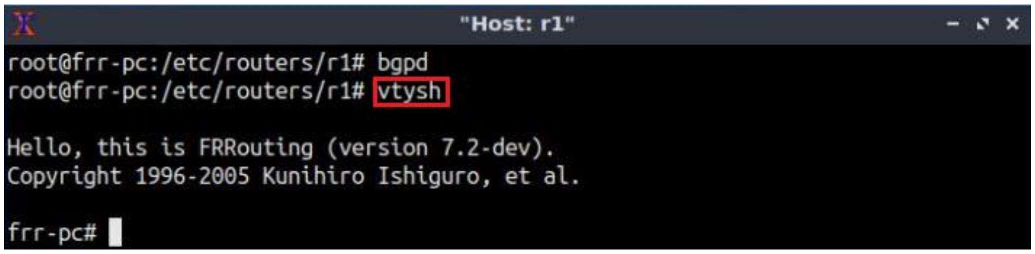


```
"Host: r1"
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1#
```

Figure 27. Starting BGP daemon.

Step 3. In order to enter to router r1 terminal, type the following command:

```
vtysh
```

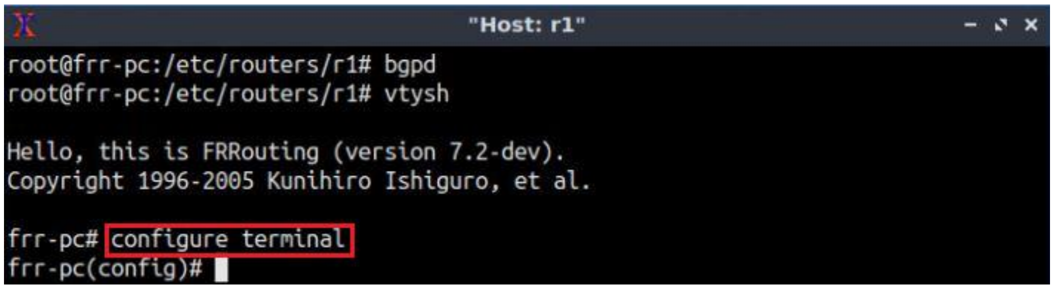


```
"Host: r1"
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 28. Starting vtysh on router r1.

Step 4. To enable router r1 configuration mode, issue the following command:

```
configure terminal
```

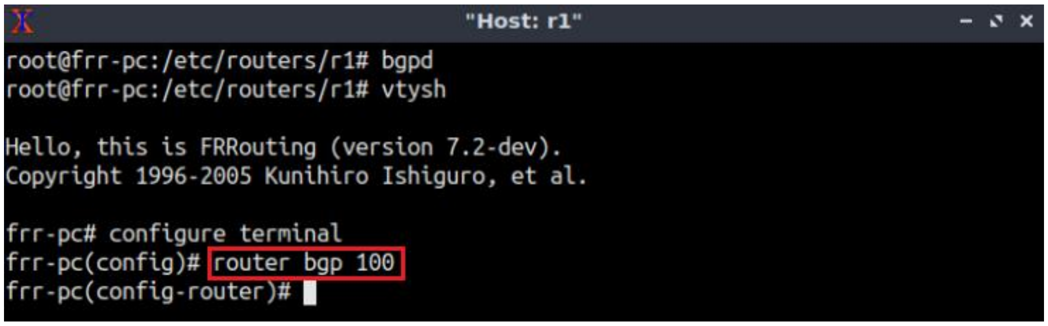


```
"Host: r1"
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)#
```

Figure 29. Enabling configuration mode on router r1.

Step 5. The ASN assigned for router r1 is 100. In order to configure BGP, type the following command:

```
router bgp 100
```

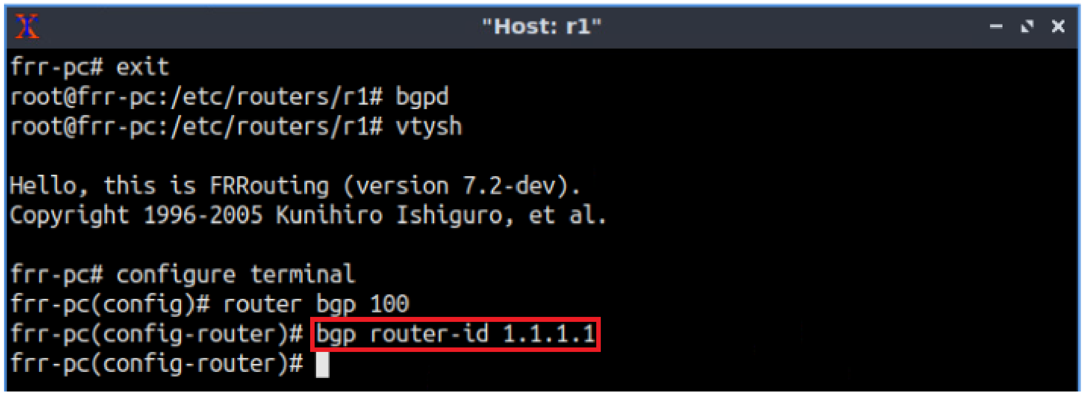


```
"Host: r1"
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)#
```

Figure 30. Configuring BGP on router r1.

Step 6. Assign a router ID to router r1 by issuing the following command.

```
bgp router-id 1.1.1.1
```

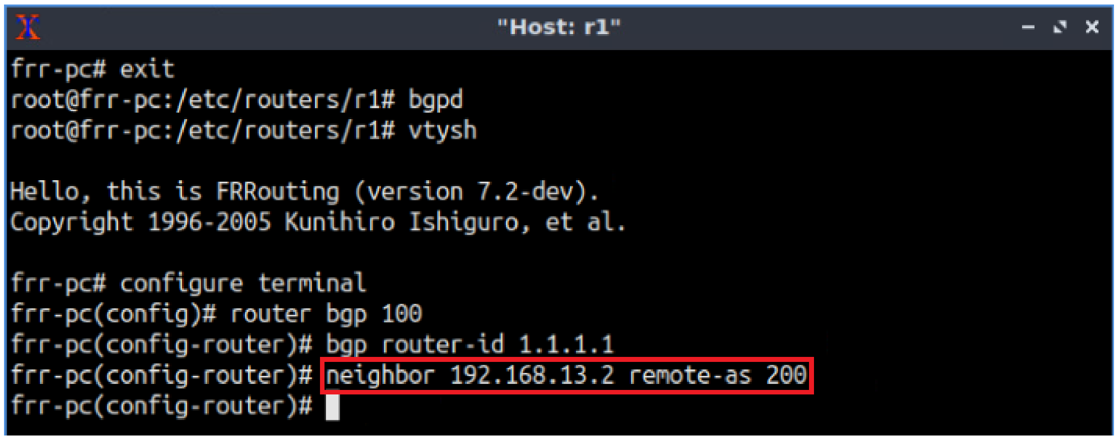


```
"Host: r1"
frr-pc# exit
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 1.1.1.1
frr-pc(config-router)#
```

Figure 31. Assigning a router ID in router r1.

Step 7. To configure a BGP neighbor to router r1 (AS 100), type the command shown below. This command specifies the neighbor IP address (192.168.13.2) and ASN of the remote BGP peer (AS 200).

```
neighbor 192.168.13.2 remote-as 200
```

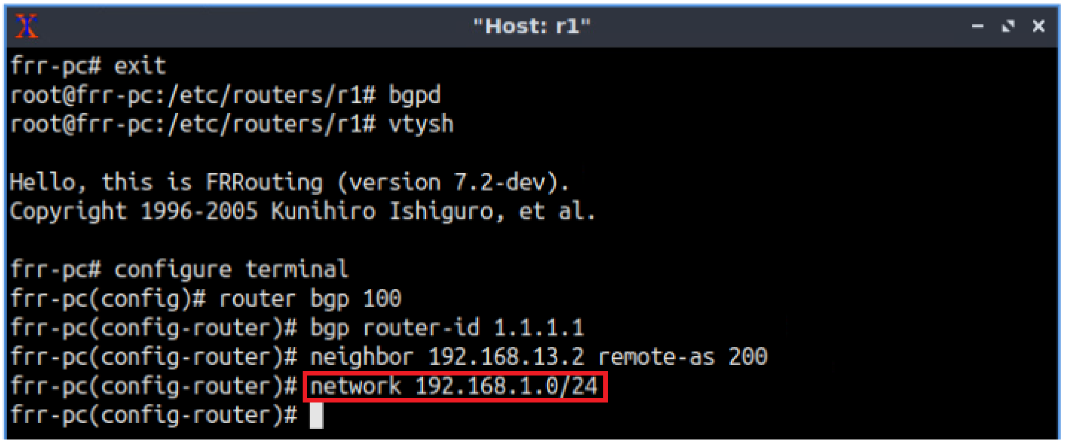


```
"Host: r1"
frr-pc# exit
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 1.1.1.1
frr-pc(config-router)# neighbor 192.168.13.2 remote-as 200
frr-pc(config-router)#
```

Figure 32. Assigning BGP neighbor to router r1.

Step 8. In this step, router r1 will advertise the LAN 192.168.1.0/24 to router r3 through EBGP. To do so, issue the following command:

```
network 192.168.1.0/24
```

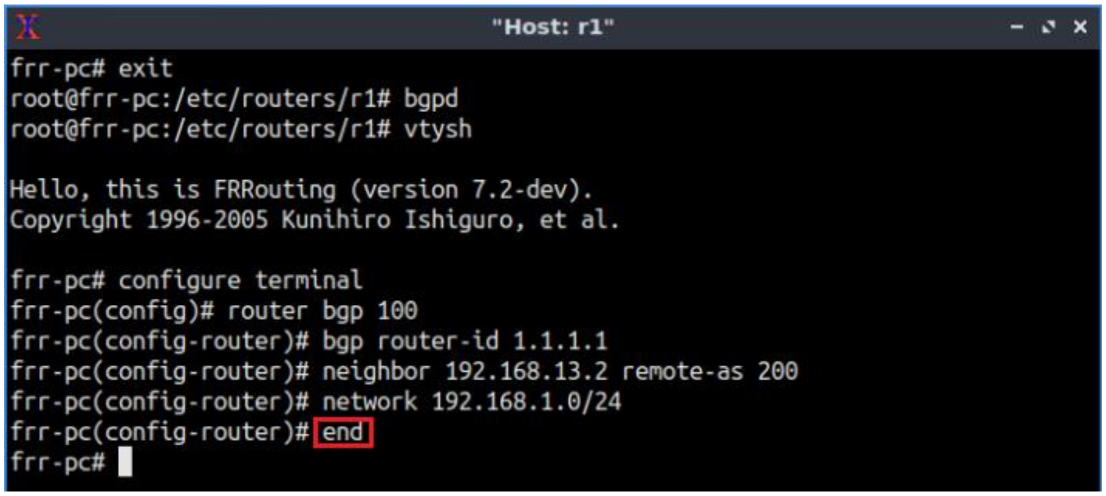


```
frr-pc# exit
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 1.1.1.1
frr-pc(config-router)# neighbor 192.168.13.2 remote-as 200
frr-pc(config-router)# network 192.168.1.0/24
frr-pc(config-router)#
```

Figure 33. Advertising a network on router r1.

Step 9. Type the following command to exit from the configuration mode.

```
end
```

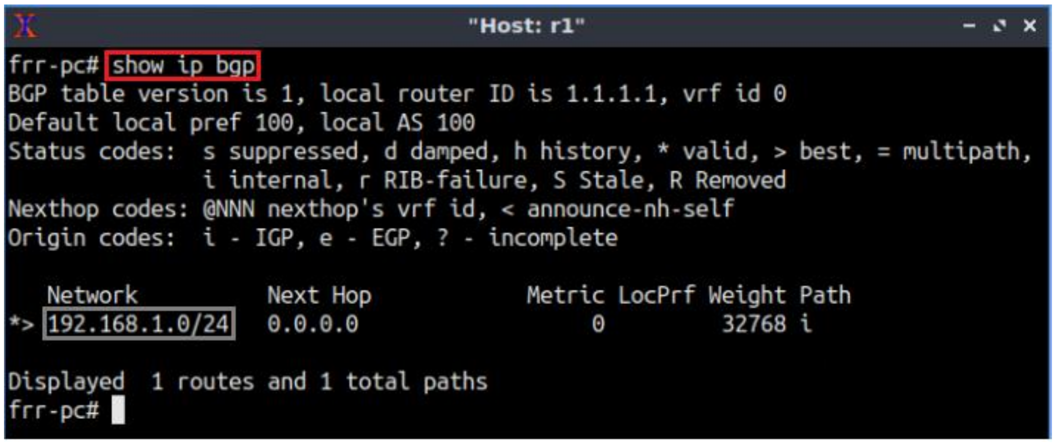


```
frr-pc# exit
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 1.1.1.1
frr-pc(config-router)# neighbor 192.168.13.2 remote-as 200
frr-pc(config-router)# network 192.168.1.0/24
frr-pc(config-router)# end
frr-pc#
```

Figure 34. Exiting from configuration mode.

Step 10. Type the following command to verify BGP networks. You will observe the LAN network of router r1.

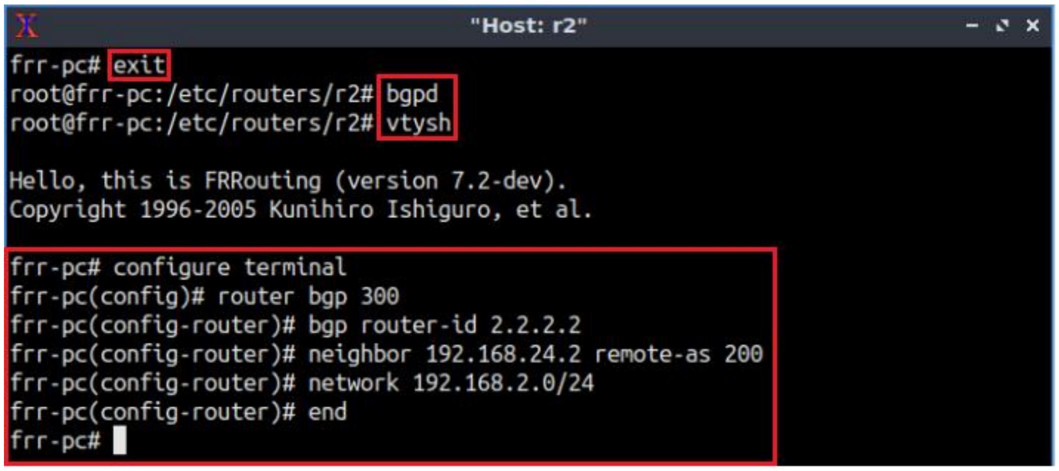
```
show ip bgp
```



```
"Host: r1"
frr-pc# show ip bgp
BGP table version is 1, local router ID is 1.1.1.1, vrf id 0
Default local pref 100, local AS 100
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 192.168.1.0/24   0.0.0.0                  0         32768 i

Displayed  1 routes and 1 total paths
frr-pc#
```

Figure 35. Verifying BGP networks on router r1.

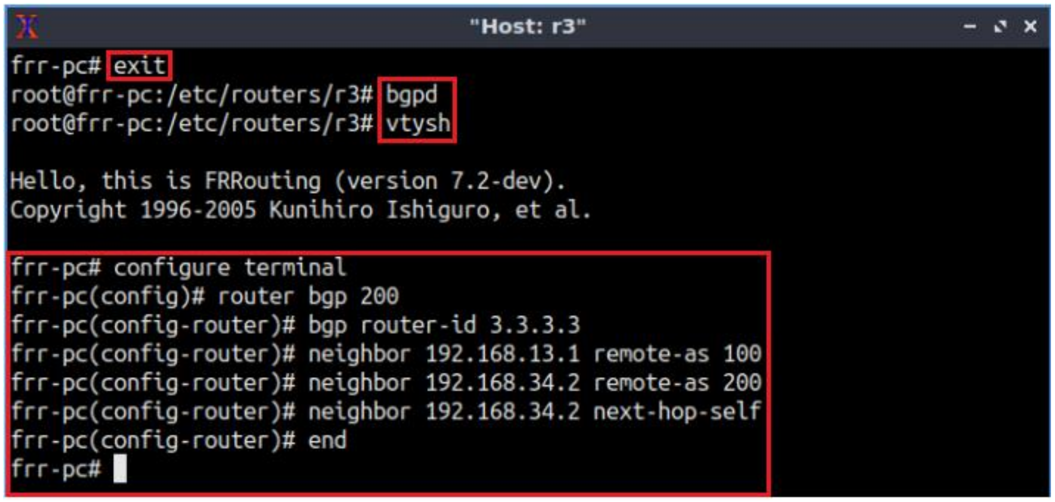
Step 11. Follow from step 1 to step 9 but with different metrics in order to configure BGP in router r2. All these steps are summarized in the following figure.



```
"Host: r2"
frr-pc# exit
root@frr-pc:/etc/routers/r2# bgpd
root@frr-pc:/etc/routers/r2# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 300
frr-pc(config-router)# bgp router-id 2.2.2.2
frr-pc(config-router)# neighbor 192.168.24.2 remote-as 200
frr-pc(config-router)# network 192.168.2.0/24
frr-pc(config-router)# end
frr-pc#
```

Figure 36. Configuring BGP on router r2.

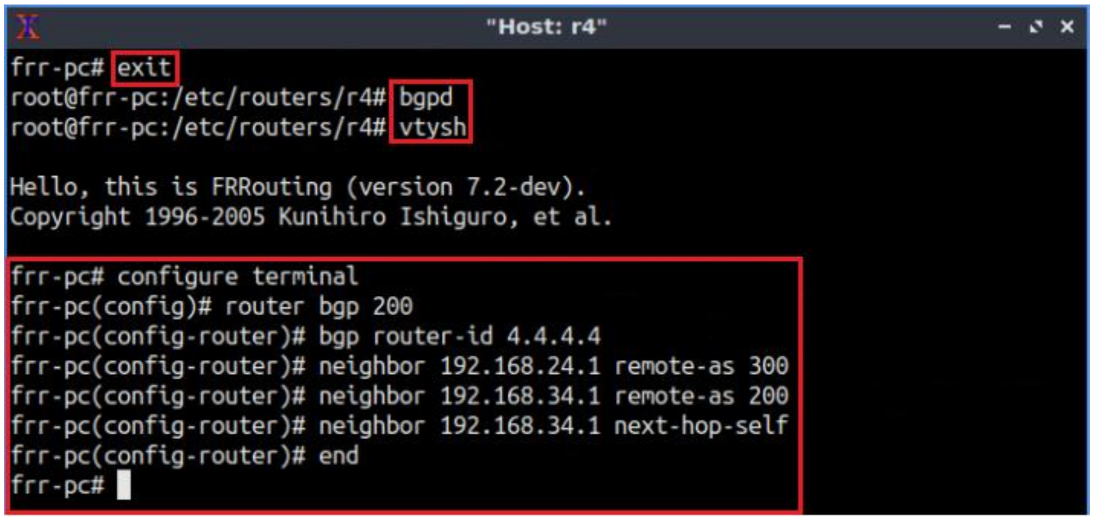
Step 12. Follow from step 1 to step 9 but with different metrics in order to configure BGP in router r3. Additionally, router r3 will configure BGP *next-hop-self* so that the neighbor 192.168.34.2 (router r4) can reach the EBGP routes advertised by router r3, such as 192.168.1.0/24, through router r3. All these steps are summarized in the following figure.



```
"Host: r3"
frr-pc# exit
root@frr-pc:/etc/routers/r3# bgpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 200
frr-pc(config-router)# bgp router-id 3.3.3.3
frr-pc(config-router)# neighbor 192.168.13.1 remote-as 100
frr-pc(config-router)# neighbor 192.168.34.2 remote-as 200
frr-pc(config-router)# neighbor 192.168.34.2 next-hop-self
frr-pc(config-router)# end
frr-pc#
```

Figure 37. Configuring BGP on router r3.

Step 13. Follow from step 1 to step 9 but with different metrics in order to configure BGP in router r4. Additionally, router r4 will configure BGP *next-hop-self* so that the neighbor 192.168.34.1 (router r3) can reach the EBGP routes advertised by router r4, such as 192.168.2.0/24, through router r4. All these steps are summarized in the following figure.

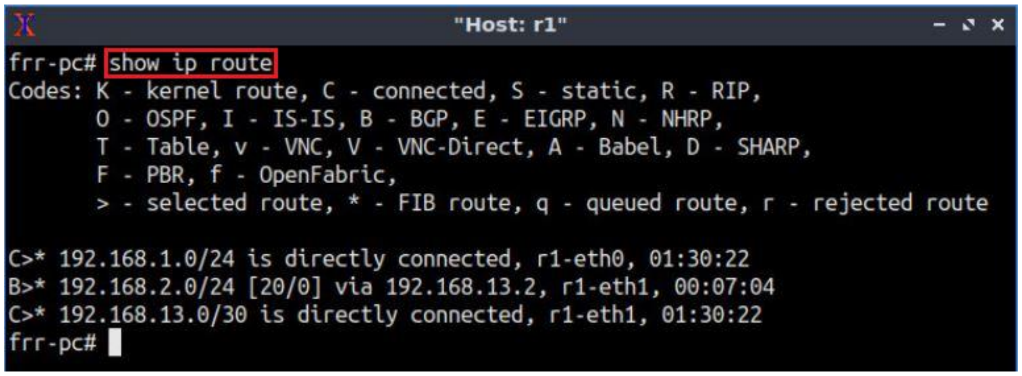


```
"Host: r4"
frr-pc# exit
root@frr-pc:/etc/routers/r4# bgpd
root@frr-pc:/etc/routers/r4# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 200
frr-pc(config-router)# bgp router-id 4.4.4.4
frr-pc(config-router)# neighbor 192.168.24.1 remote-as 300
frr-pc(config-router)# neighbor 192.168.34.1 remote-as 200
frr-pc(config-router)# neighbor 192.168.34.1 next-hop-self
frr-pc(config-router)# end
frr-pc#
```

Figure 38. Configuring BGP on router r4.

Step 14. Type the following command to verify the routing table of router r1. The LAN of router r2 network (192.168.2.0/24) is advertised to router r1 through EBGP.

```
show ip route
```



```
frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.1.0/24 is directly connected, r1-eth0, 01:30:22
B>* 192.168.2.0/24 [20/0] via 192.168.13.2, r1-eth1, 00:07:04
C>* 192.168.13.0/30 is directly connected, r1-eth1, 01:30:22
frr-pc#
```

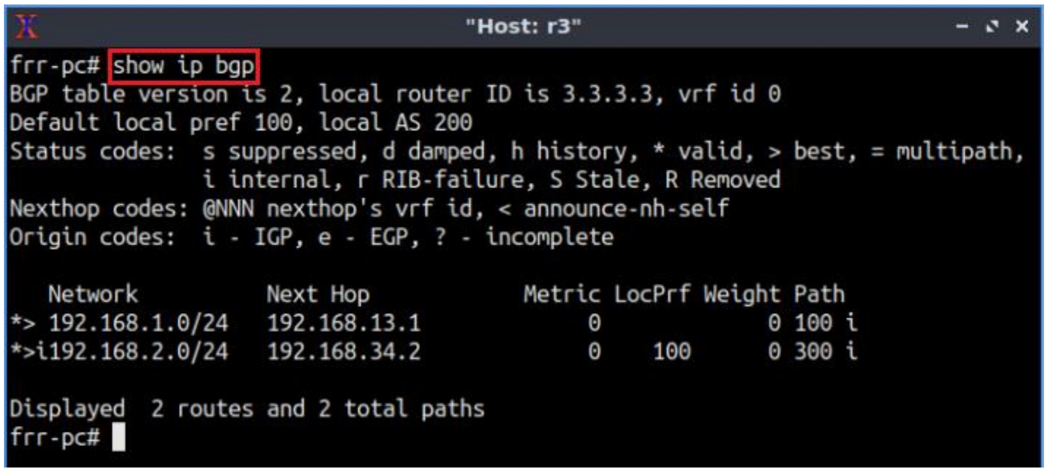
Figure 39. Verifying the routing table of router r1.

4 Perform BGP hijacking

In this section, you will configure router r1 to hijack the network 192.168.2.0/24 corresponding to Campus-2 by advertising it to its BGP neighbors. Thus, router r3 will have a route to the network 192.168.2.0/24 through router r1 and will use this route to send all network traffic destined to Campus-2.

Step 1. On router r3 terminal, type the following command to verify BGP networks.

```
show ip bgp
```



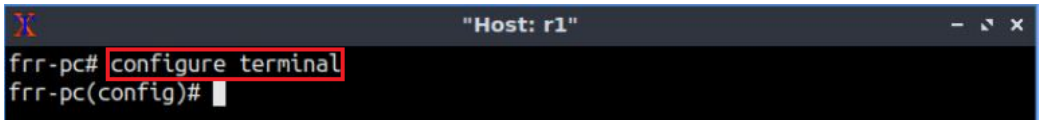
```
frr-pc# show ip bgp
BGP table version is 2, local router ID is 3.3.3.3, vrf id 0
Default local pref 100, local AS 200
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 192.168.1.0/24   192.168.13.1             0             0 100 i
*>i192.168.2.0/24   192.168.34.2             0    100      0 300 i

Displayed  2 routes and 2 total paths
frr-pc#
```

Figure 40. Verifying BGP networks in router r3.

Consider the figure above. Router r3 can reach the networks 192.168.1.0/24 and 192.168.2.0/24 through the next hops 192.168.13.1 and 192.168.34.2, respectively.

Step 2. To enable router r1 into configuration mode, issue the following command.

```
configure terminal
```

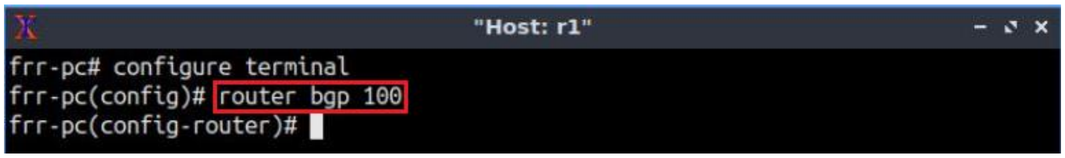


```
frr-pc# configure terminal
frr-pc(config)# 
```

Figure 41. Enabling configuration mode in router r1.

Step 3. You will advertise the network 192.168.2.0/24 of Campus-2 from router r1 to all BGP neighbors. Type the following command to enter BGP configuration mode.

```
router bgp 100
```

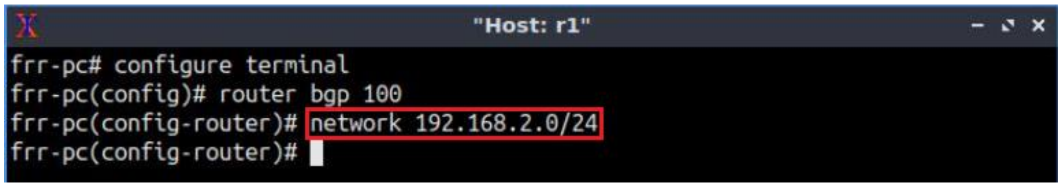


```
frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# 
```

Figure 42. Configuring BGP on router r1.

Step 4. In this step, router r1 will hijack the network 192.168.2.0/24 of Campus-2 by advertising this network to all its BGP neighbors.

```
network 192.168.2.0/24
```

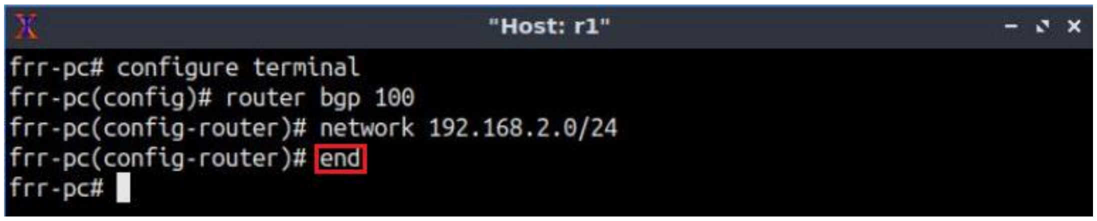


```
frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# network 192.168.2.0/24
frr-pc(config-router)# 
```

Figure 43. Hijacking a network on router r1.

Step 5. Type the following command to exit from the configuration mode.

```
end
```

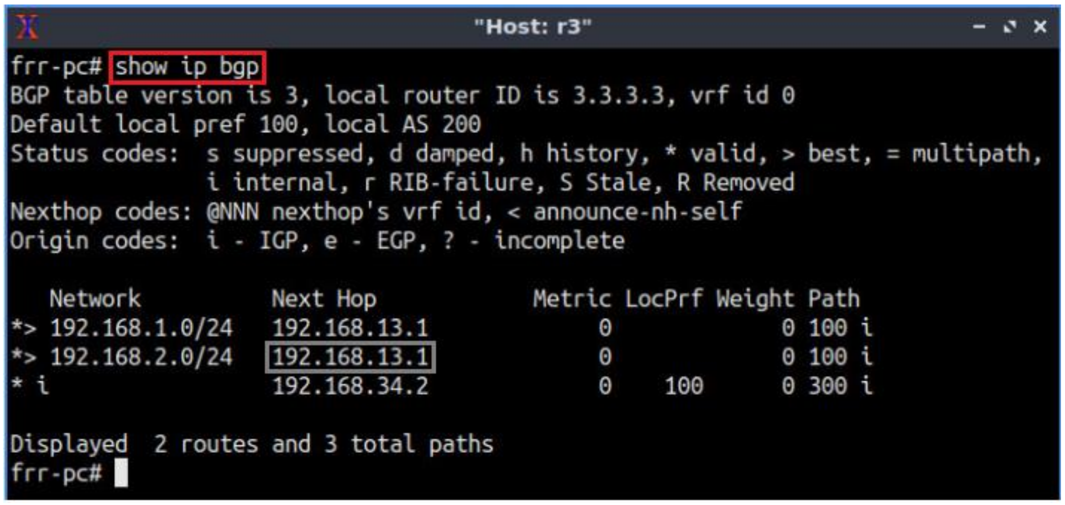


```
frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# network 192.168.2.0/24
frr-pc(config-router)# end
frr-pc# 
```

Figure 44. Exiting from configuration mode.

Step 6. On router r3 terminal, type the following command to verify BGP networks.

```
show ip bgp
```



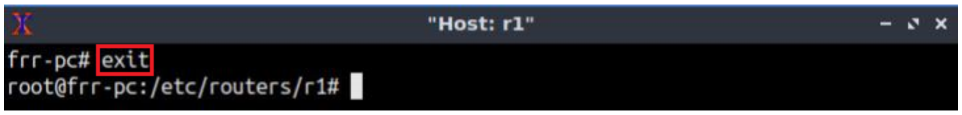
```
"Host: r3"
frr-pc# show ip bgp
BGP table version is 3, local router ID is 3.3.3.3, vrf id 0
Default local pref 100, local AS 200
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 192.168.1.0/24   192.168.13.1             0             0 100 i
*> 192.168.2.0/24   192.168.13.1             0             0 100 i
* i                 192.168.34.2             0    100      0 300 i

Displayed  2 routes and 3 total paths
frr-pc#
```

Figure 45. Verifying BGP networks in router r3.

Consider the figure above. Router r3 can reach the network 192.168.2.0/24 through the next hops 192.168.13.1 (router r1) and 192.168.34.2 (router r4). However, router r3 prefers to use the next hop 192.168.13.1 over 192.168.34.2 in its route to 192.168.2.0/24. This can be inferred from the characters `>*` next to the network address 192.168.2.0/24, which means that the corresponding next hop (192.168.13.1) is the best route to reach the network.

Step 7. On router r1 terminal, type the following command to exit from the configuration mode.

```
exit
```

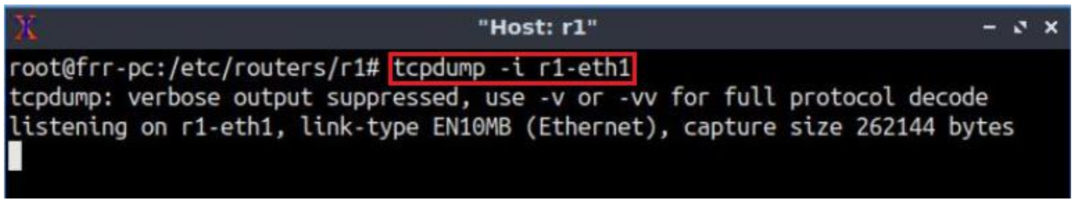


```
"Host: r1"
frr-pc# exit
root@frr-pc:/etc/routers/r1#
```

Figure 46. Exiting the vtysh session.

Step 8. Type the following command on router r1 terminal. The command `tcpdump` allows you to capture the network traffic. The `-i` option allows you to specify the interface to be monitored (*r1-eth1*).

```
tcpdump -i r1-eth1
```

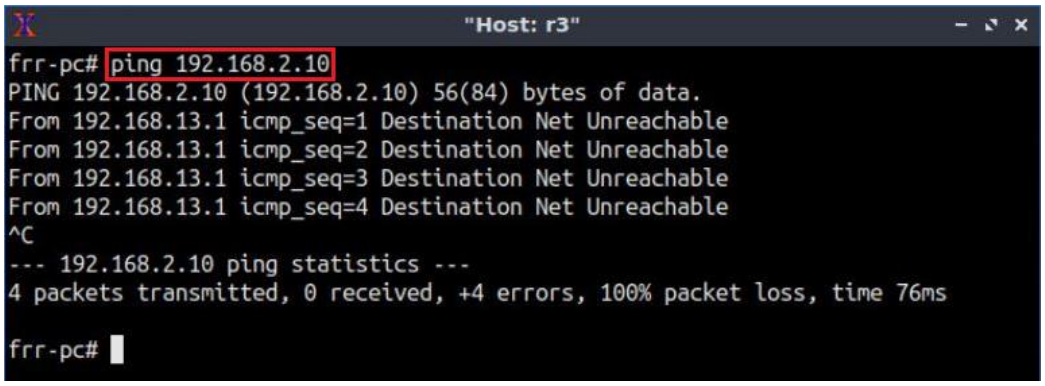


```
"Host: r1"
root@frr-pc:/etc/routers/r1# tcpdump -i r1-eth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on r1-eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
```

Figure 47. Capturing packets on interface *r1-eth1*.

Step 9. On router r3 terminal, type the following command to ping the IP address 192.168.2.10 corresponding to host h2.

```
ping 192.162.2.10
```

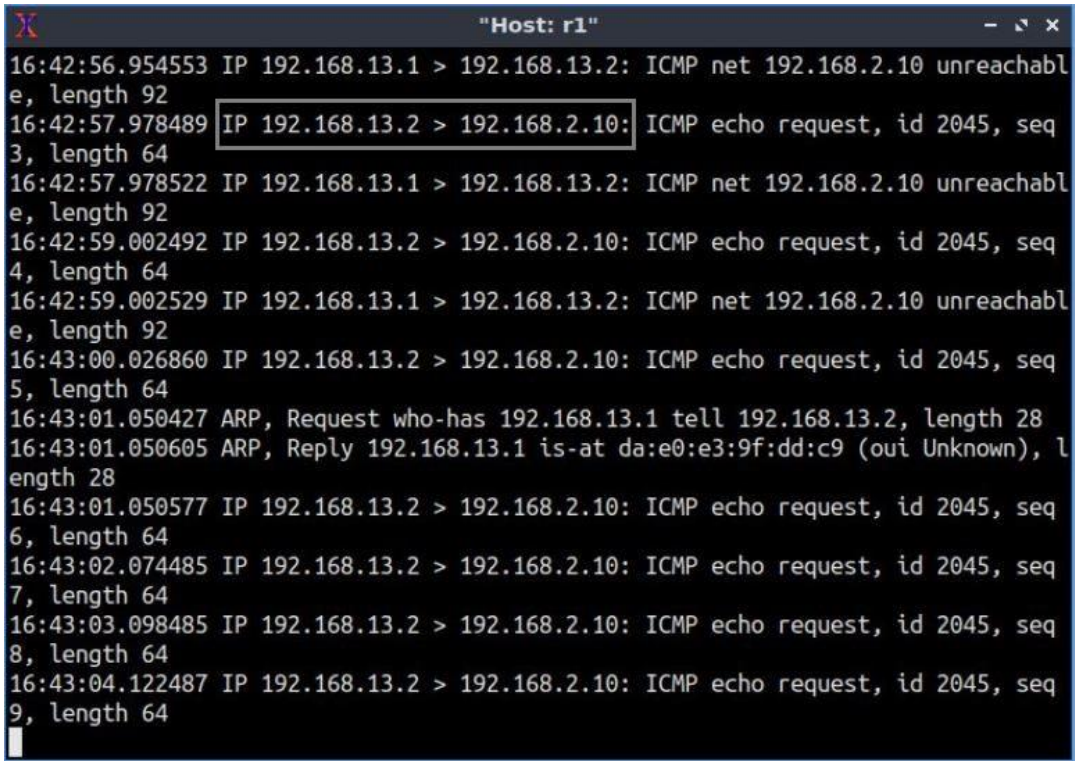


```
frr-pc# ping 192.168.2.10
PING 192.168.2.10 (192.168.2.10) 56(84) bytes of data.
From 192.168.13.1 icmp_seq=1 Destination Net Unreachable
From 192.168.13.1 icmp_seq=2 Destination Net Unreachable
From 192.168.13.1 icmp_seq=3 Destination Net Unreachable
From 192.168.13.1 icmp_seq=4 Destination Net Unreachable
^C
--- 192.168.2.10 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 76ms

frr-pc#
```

Figure 48. Pinging host h2 from router r3.

Consider the figure above. Router r3 is pinging a host within the network 192.168.2.0/24. If the hijacking was successful, then the network traffic must be redirected to router r1, rather than router r4 (and eventually host h2). Press `Ctrl+c` to stop the test.

Step 10. On router r1, notice how the network traffic that is sent from the IP addresses 192.168.13.2 (router r3) to 192.168.2.10 (host h2) is captured on router r1.



```
16:42:56.954553 IP 192.168.13.1 > 192.168.13.2: ICMP net 192.168.2.10 unreachabl
e, length 92
16:42:57.978489 IP 192.168.13.2 > 192.168.2.10: ICMP echo request, id 2045, seq
3, length 64
16:42:57.978522 IP 192.168.13.1 > 192.168.13.2: ICMP net 192.168.2.10 unreachabl
e, length 92
16:42:59.002492 IP 192.168.13.2 > 192.168.2.10: ICMP echo request, id 2045, seq
4, length 64
16:42:59.002529 IP 192.168.13.1 > 192.168.13.2: ICMP net 192.168.2.10 unreachabl
e, length 92
16:43:00.026860 IP 192.168.13.2 > 192.168.2.10: ICMP echo request, id 2045, seq
5, length 64
16:43:01.050427 ARP, Request who-has 192.168.13.1 tell 192.168.13.2, length 28
16:43:01.050605 ARP, Reply 192.168.13.1 is-at da:e0:e3:9f:dd:c9 (oui Unknown), l
ength 28
16:43:01.050577 IP 192.168.13.2 > 192.168.2.10: ICMP echo request, id 2045, seq
6, length 64
16:43:02.074485 IP 192.168.13.2 > 192.168.2.10: ICMP echo request, id 2045, seq
7, length 64
16:43:03.098485 IP 192.168.13.2 > 192.168.2.10: ICMP echo request, id 2045, seq
8, length 64
16:43:04.122487 IP 192.168.13.2 > 192.168.2.10: ICMP echo request, id 2045, seq
9, length 64
```

Figure 49. Monitoring network traffic on router r1.

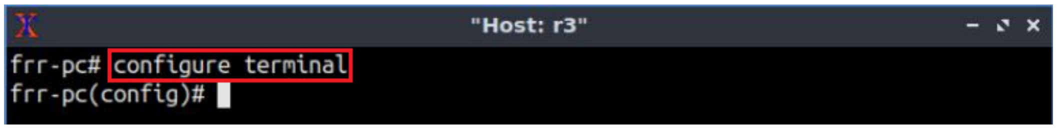
To interrupt capturing the network traffic on interface *eth1* of router r1 press `Ctrl+c`.

5 Mitigate BGP hijacking by using IP prefix filtering

In this section, you will configure IP prefix lists on routers r3 and r4 to restrict route advertisements from Campus-1 and Campus-2, respectively. Thus, mitigating BGP hijacking attacks.

Step 1. To enable router r3 into configuration mode, issue the following command.

```
configure terminal
```



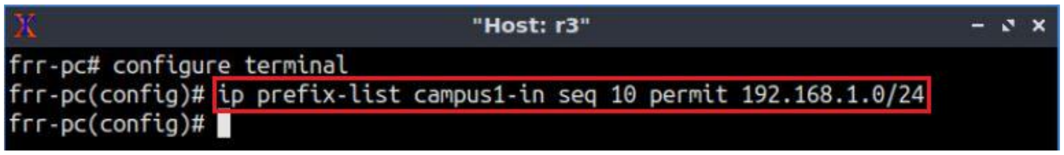
"Host: r3"

```
frr-pc# configure terminal
frr-pc(config)#
```

Figure 50. Enabling configuration mode on router r1.

Step 2. In this step, you will create an IP prefix list that permits the network 192.168.1.0/24. An IP prefix list must have a name (campus1-in), a permit or deny clause to allow or reject the packets that match the prefix list, and the network IP address to match on (192.168.1.0/24).

```
ip prefix-list campus1-in seq 10 permit 192.168.1.0/24
```



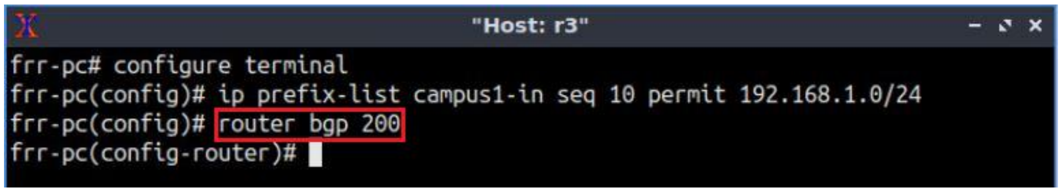
"Host: r3"

```
frr-pc# configure terminal
frr-pc(config)# ip prefix-list campus1-in seq 10 permit 192.168.1.0/24
frr-pc(config)#
```

Figure 51. Creating an IP prefix list.

Step 3. You will filter the route updates that are advertised by neighbor router r1 to router r3. Type the following command to enter BGP configuration mode:

```
router bgp 200
```



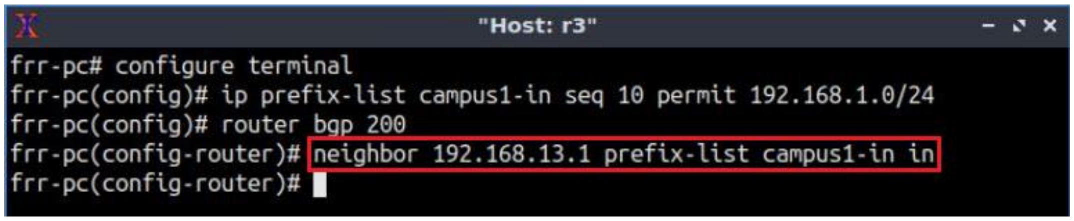
"Host: r3"

```
frr-pc# configure terminal
frr-pc(config)# ip prefix-list campus1-in seq 10 permit 192.168.1.0/24
frr-pc(config)# router bgp 200
frr-pc(config-router)#
```

Figure 52. Configuring BGP on router r3.

Step 4. Router r3 will apply the prefix list *campus1-in* to its neighbor 192.168.13.1 (router r1). Thus, only BGP advertisements corresponding to 192.168.1.0/24 will be permitted since they match the IP prefix list assigned. The option `in` corresponds to inbound traffic, i.e., the traffic coming to router r3.

```
neighbor 192.168.13.1 prefix-list campus1-in in
```

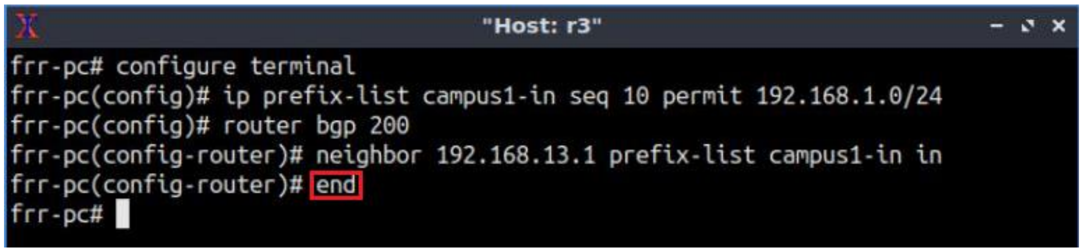


```
"Host: r3"
frr-pc# configure terminal
frr-pc(config)# ip prefix-list campus1-in seq 10 permit 192.168.1.0/24
frr-pc(config)# router bgp 200
frr-pc(config-router)# neighbor 192.168.13.1 prefix-list campus1-in in
frr-pc(config-router)#
```

Figure 53. Applying the IP prefix list to router r3 neighbor.

Step 5. Type the following command to exit from the configuration mode.

```
end
```

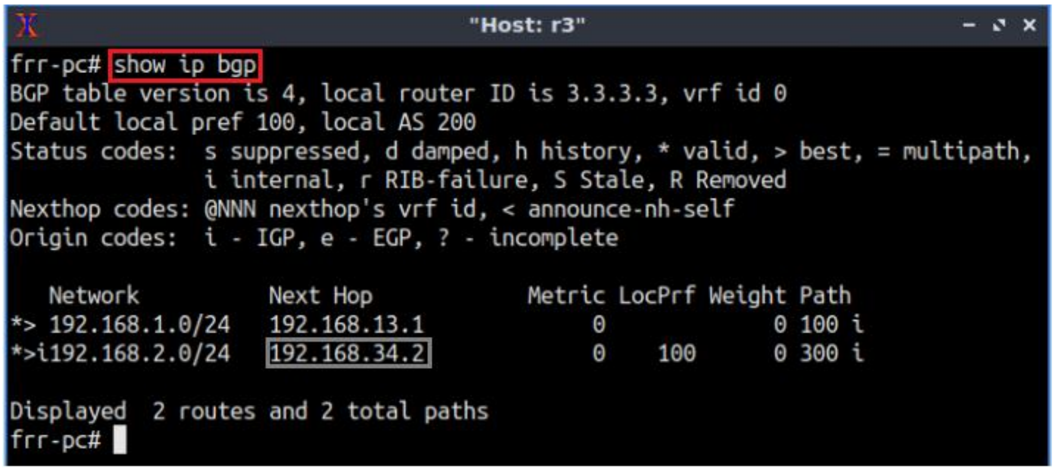


```
"Host: r3"
frr-pc# configure terminal
frr-pc(config)# ip prefix-list campus1-in seq 10 permit 192.168.1.0/24
frr-pc(config)# router bgp 200
frr-pc(config-router)# neighbor 192.168.13.1 prefix-list campus1-in in
frr-pc(config-router)# end
frr-pc#
```

Figure 54. Exiting from configuration mode.

Step 6. On router r3 terminal, type the following command to verify BGP networks.

```
show ip bgp
```

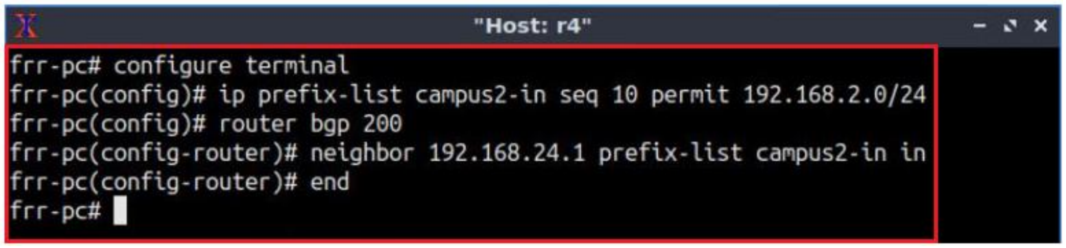


```
"Host: r3"
frr-pc# show ip bgp
BGP table version is 4, local router ID is 3.3.3.3, vrf id 0
Default local pref 100, local AS 200
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 192.168.1.0/24   192.168.13.1             0             0 100 i
*>i192.168.2.0/24   192.168.34.2             0    100      0 300 i

Displayed  2 routes and 2 total paths
frr-pc#
```

Figure 55. Verifying BGP networks in router r3.

Consider the figure above. Router r3 has a route to the network 192.168.2.0/24 through the next hop 192.168.34.2 (router r4) only. Even though router r1 is still advertising the network 192.168.2.0/24, router r3 mitigates BGP hijacking through the configure IP prefix list filters.

Step 7. Follow from step 1 to step 5 to configure IP prefix list on router r4. Router r4 will configure the IP prefix list campus2-in to only permit advertising the network 192.168.2.0/24 from router r2.



```
frr-pc# configure terminal
frr-pc(config)# ip prefix-list campus2-in seq 10 permit 192.168.2.0/24
frr-pc(config)# router bgp 200
frr-pc(config-router)# neighbor 192.168.24.1 prefix-list campus2-in in
frr-pc(config-router)# end
frr-pc#
```

Figure 56. Configuring IP prefix list on router r4.

This concludes Lab 3. Stop the emulation and then exit out of MiniEdit.

References

1. A. Tanenbaum, D. Wetherall, “Computer networks,” 5th Edition, Pearson, 2012.
2. V. Khare, Q. Ju, B. Zhang, “Concurrent Prefix Hijacks: Occurrence and Impacts,” 2012. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/2398776.2398780
3. MANRS, “Filtering,” [Online]. Available: https://www.manrs.org/isps/guide/filtering/
4. M. Lepinski, S. Kent, “An infrastructure to support secure internet routing,” [Online]. Available: https://tools.ietf.org/html/rfc6480
5. Ciscopress, “CCNP Routing and Switching Portable Command Guide: Configuration of Redistribution,” 2015. [Online]. Available: https://www.ciscopress.com/articles/article.asp?p=2273507&seqNum=11

MPLS AND ADVANCED BGP TOPICS

Lab 4: Introduction to MPLS

Document Version: 07-04-2021

Award 1829698
"CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers"

Contents

Overview

This lab presents an introduction to Multiprotocol Label Switching (MPLS). This protocol allows routers to forward packets based on fixed-length labels rather than the destination IP addresses. In this lab, static MPLS will be configured and verified between two hosts that are required to exchange routes.

Objectives

By the end of this lab, you should be able to:

1. Understand the concept of MPLS.
2. Explore MPLS label distribution protocols.
3. Configure static MPLS in routers.
4. Verify MPLS labels by capturing the traffic between routers.

Lab settings

The information in Table 1 provides the credentials to access Client machine.

Table 1**.** Credentials to access Client machine.

Device	Account	Password
Client	admin	password

Lab roadmap

This lab is organized as follows:

1. Section 1: Introduction.
2. Section 2: Lab topology.
3. Section 3: Configuring static MPLS from router r1 to router r2.
4. Section 4: Configuring static MPLS from router r2 to router r1.
5. Section 5: Verifying the configuration.

1 Introduction

1.1 Introduction to MPLS

In traditional IP routing, each router must look up the destination IP address of the packet to make the forwarding decision. MPLS eliminates the look up process and ensures transmission between end nodes with short path labels. The predetermined paths that make MPLS work are called label-switched paths (LSPs). A router that operates at the edge of an MPLS network and acts as the entry and exit point for the network is called Label Edge Router (LER). The packet enters the edge of the MPLS backbone is examined and forwarded to the next hop in the pre-set Label Switched Path (LSP). As the packet travels that path, each router on the path uses the label – not other information, such as the IP. An MPLS router that performs routing based only on the label is called a label switch router (LSR). Each LSR has the ability to do three main things: push, pop, or swap labels from a packet. To push a label simply means to add a label to a packet, to pop is to remove a label from a packet, and to swap is to remove and add an alternative label to the packet[1]. A packet can have multiple labels attached which are arranged in a stack and are considered in the order from the most recent label to the least recent label.

However, within each router, the incoming label is examined, and its next hop is matched with a new label. The old label is replaced with the new label for the packet's next destination, and then the freshly labeled packet is sent to the next router. Each router repeats the process until the packet reaches the exit router. The label information is removed at either the last hop or the exit router so that the packet goes back to being identified by an IP header instead of an MPLS label.

The ingress LER is the first to insert an MPLS header and label on a packet. The egress LER is the last point before leaving the network and removes all the MPLS labels and header. Both the ingress and egress LER's are considered as Provider Edge (PE) routers. LSR's are considered as Provider (P) routers.

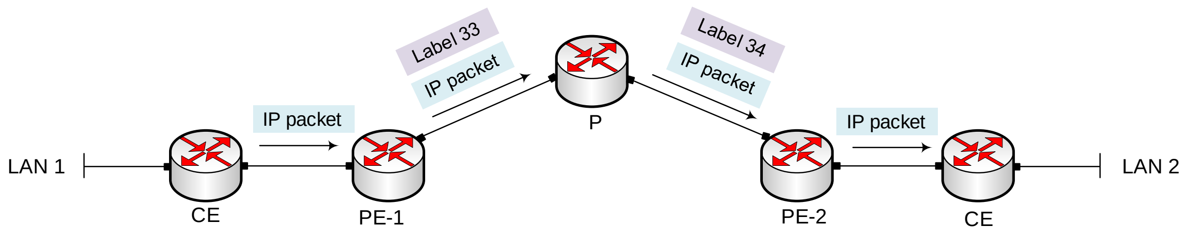


Figure 1. MPLS label forwarding.

Consider Figure 1. Customer Edge (CE) of Local Area Network (LAN) 1 will forward an IP packet to the provider Edge (PE). PE will push label 33 along with the IP packet to the provider (P). P will replace the label with label 34 which will be forwarded to the Provider Edge (PE). PE will pop the label and the IP packet will be delivered to Customer Edge (CE) of LAN 2 which is the destination router.

1.2 Label distribution protocols

Label distribution protocol is a set of procedures that provides the information MPLS uses to create the forwarding tables in each LSR in the MPLS domain[2]. Followings are the most widely used label distribution protocols in MPLS.

Static MPLS: MPLS entries can be configured statically, handling MPLS consists of pushing, swapping, or popping labels to IP packets.

LDP: LDP is a protocol that automatically generates and exchanges labels between routers. LDP enables LSRs to discover potential peers and to establish LDP sessions. It depends on the network's Interior Gateway Protocol (IGP) to determine the path an LSP must take.

Resource Reservation Protocol-Traffic Engineering (RSVP-TE): RSVP-TE is used to establish MPLS transport LSPs when there are traffic engineering requirements. RSVP is a signaling protocol that handles bandwidth allocation and true traffic engineering across an MPLS network. When RSVP and MPLS are combined, a flow or session can be defined with greater flexibility and generality.

1.3 MPLS header architecture

MPLS operates at a layer that is generally considered to lie between OSI Layer 2 (data link layer) and Layer 3 (network layer), often referred to as a layer 2.5 protocol.

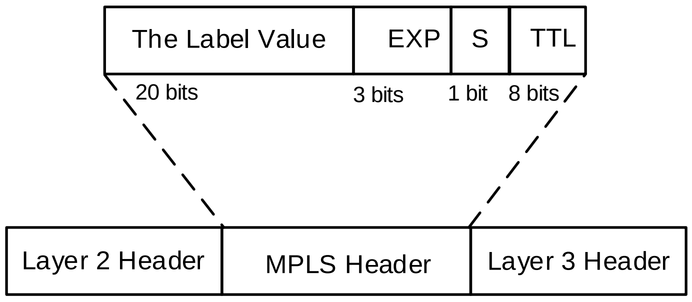


Figure 2. MPLS header architecture.

Consider Figure 2. MPLS works by prefixing packets with an MPLS header, containing one or more labels. This is called a label stack. Each entry in the label stack contains four fields.

1. **The Label Value:** The first 20 bits are the label value. This value can be between 0 and 1,048,575. The first 16 values are exempted from normal use.

2. **EXP:** Three bits are the experimental (EXP) bits. These bits are used solely for quality of service (QoS) purposes, which represents the set of techniques necessary to manage network bandwidth, delay, jitter, and packet loss.

3. **S:** 1 bit is the Bottom of Stack (BoS) bit. The stack is the collection of labels that are found on top of the packet. The BoS bit is set to 1 if this is the bottom label in the stack. Otherwise, the BoS bit remains 0.

4. **TTL:** Last 8 bits are used for Time-to-Live (TTL). This TTL has the same function as the TTL found in the IP header. It is simply decreased by 1 at each hop, and its main function is to avoid a packet being stuck in a routing loop.

2 Lab topology

Consider Figure 3. The topology consists of three routers, two switches and two end hosts. Customer 1 (h1) and Customer 2 (h2) are allowed to exchange routes using static MPLS.

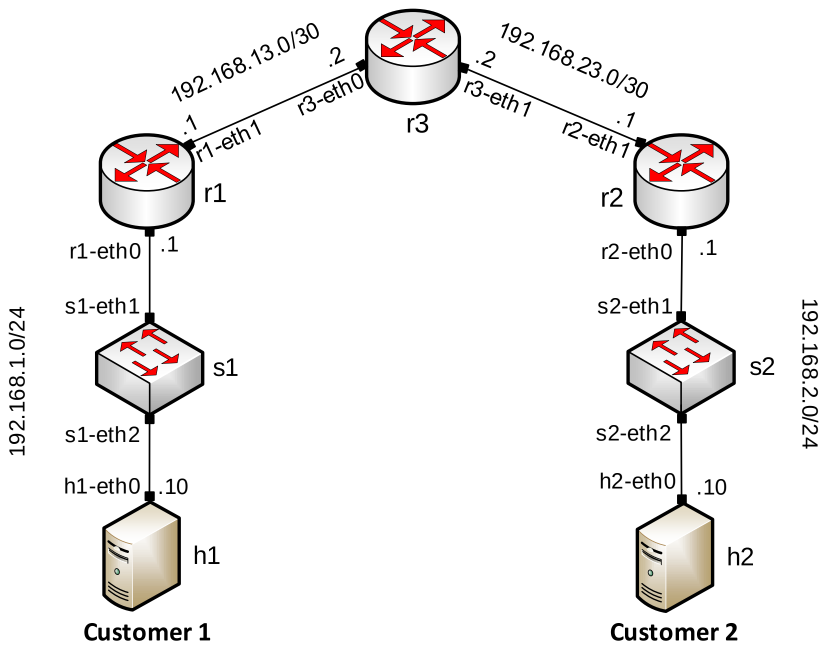


Figure 3. Lab topology.

2.1 Lab settings

Routers and hosts are already configured according to Table 2.

Table 2. Topology information.

Device	Interface	IPV4 Address	Subnet	Default gateway
r1	r1-eth0	192.168.1.1	/24	N/A
	r1-eth1	192.168.13.1	/30	N/A
r2	r2-eth0	192.168.2.1	/24	N/A
	r2-eth1	192.168.23.1	/30	N/A
r3	r3-eth0	192.168.13.2	/30	N/A
	r3-eth1	192.168.23.2	/30	N/A
h1	h1-eth0	192.168.1.10	/24	192.168.1.1
h2	h2-eth0	192.168.2.10	/24	192.168.2.1

2.2 Open the topology and load the configuration

Step 1. Start by launching MiniEdit by clicking on the desktop shortcut. When prompted for a password, type `password`.

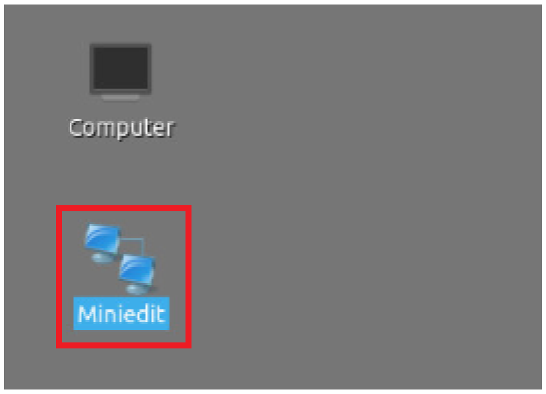


Figure 4. MiniEdit shortcut.

Step 2. On MiniEdit's menu bar, click on *File* then *open* to load the lab's topology. Locate the *lab4.mn* topology file in the default directory, */home/frr/MPLS_advanced_BGP/lab4* and click on *Open*.

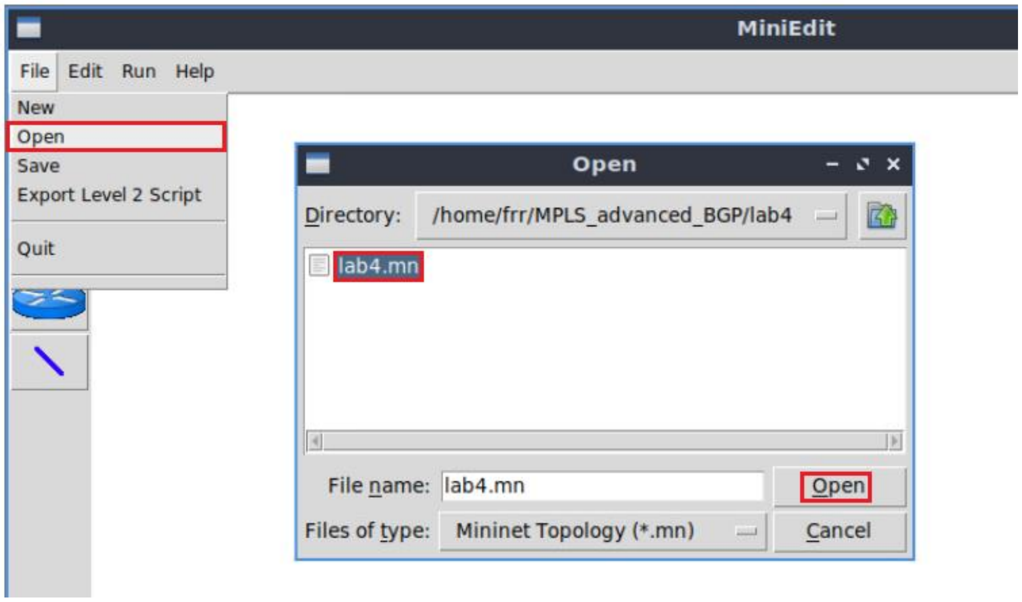


Figure 5. MiniEdit's Open dialog.

At this point the topology is loaded with all the required network components. You will execute a script that will load the configuration of the routers.

Step 3. Open the Linux terminal.

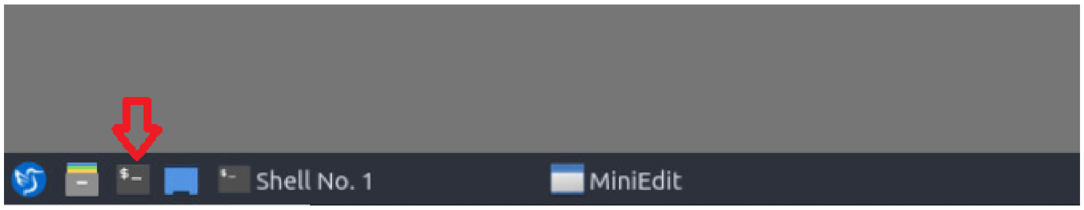
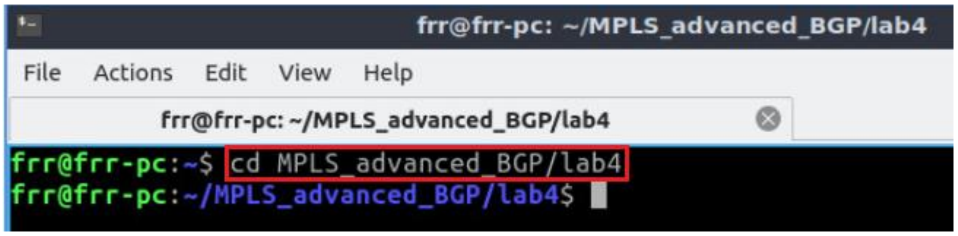


Figure 6. Opening Linux terminal.

Step 4. Click on the Linux's terminal and navigate into *MPLS_advanced_BGP/lab4* directory by issuing the following command. This folder contains a configuration file and the script responsible for loading the configuration. The configuration file will assign the IP addresses to the routers' interfaces. The `cd` command is short for change directory followed by an argument that specifies the destination directory.

```
cd MPLS_advanced_BGP/lab4
```

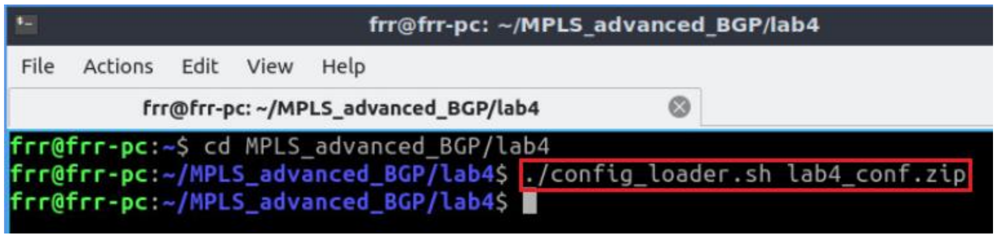

frr@frr-pc: ~/MPLS_advanced_BGP/lab4
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab4

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab4
frr@frr-pc:~/MPLS_advanced_BGP/lab4$
```

Figure 7. Navigating into the lab's directory.

Step 5. To execute the shell script, type the following command. The argument of the program corresponds to the configuration file that will be loaded in all routers.

```
./config_loader.sh lab4_conf.zip
```


frr@frr-pc: ~/MPLS_advanced_BGP/lab4
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab4

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab4
frr@frr-pc:~/MPLS_advanced_BGP/lab4$ ./config_loader.sh lab4_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab4$
```

Figure 8. Executing the shell script to load the configuration.

Step 6. At this point the interfaces of hosts h1 and h2 are configured. To proceed with the emulation, click on the *Run* button located on lower left-hand side.

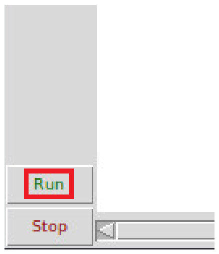


Figure 9. Starting the emulation.

Step 7. Click on Mininet's terminal, i.e., the one that launched when MiniEdit was started.

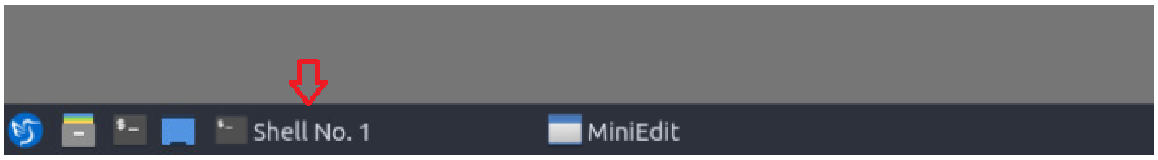
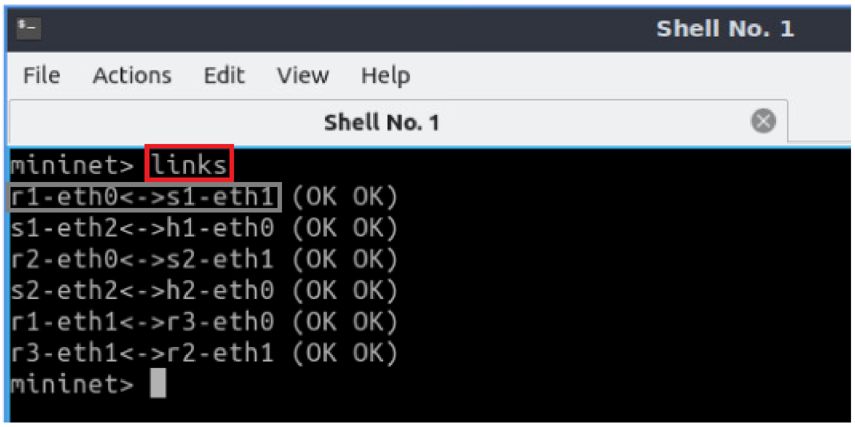


Figure 10. Opening Mininet's terminal.

Step 8. Issue the following command to display the interface names and connections.

```
links
```



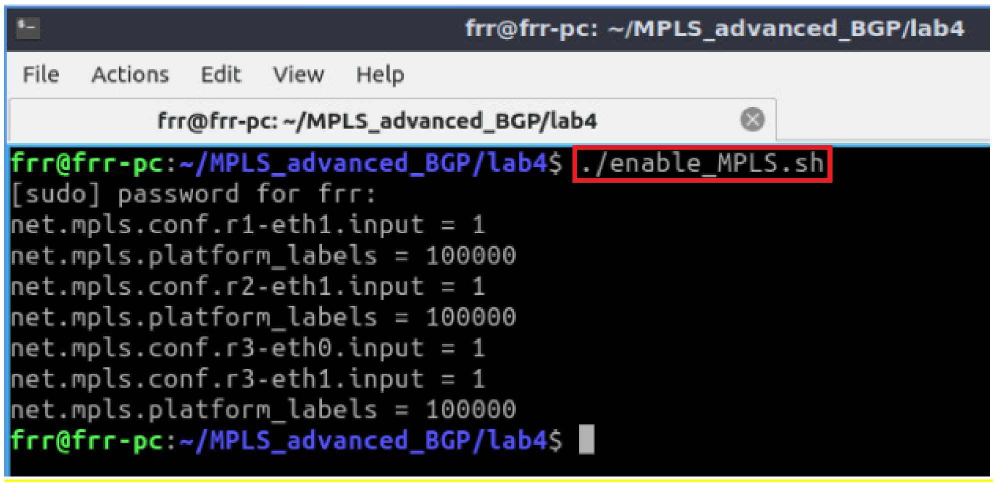
```
mininet> links
r1-eth0<->s1-eth1 (OK OK)
s1-eth2<->h1-eth0 (OK OK)
r2-eth0<->s2-eth1 (OK OK)
s2-eth2<->h2-eth0 (OK OK)
r1-eth1<->r3-eth0 (OK OK)
r3-eth1<->r2-eth1 (OK OK)
mininet>
```

Figure 11. Displaying network interfaces.

In the figure above, the link displayed within the gray box indicates that interface eth0 of router r1 connects to interface eth1 of switch s1 (i.e., *r1-eth0<->s1-eth1*).

Step 9. In order to enable MPLS forwarding in all the router's interfaces, type the following command in the opened Linux terminal. If a password is required, type `password`.

```
./enable_MPLS.sh
```


frr@frr-pc: ~/MPLS_advanced_BGP/lab4
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab4

```
frr@frr-pc:~/MPLS_advanced_BGP/lab4$ ./enable_MPLS.sh
[sudo] password for frr:
net.mpls.conf.r1-eth1.input = 1
net.mpls.platform_labels = 100000
net.mpls.conf.r2-eth1.input = 1
net.mpls.platform_labels = 100000
net.mpls.conf.r3-eth0.input = 1
net.mpls.conf.r3-eth1.input = 1
net.mpls.platform_labels = 100000
frr@frr-pc:~/MPLS_advanced_BGP/lab4$
```

Figure 12. Enabling MPLS forwarding in all the routers.

Consider the figure above. The command executes a shell script that enables MPLS forwarding in all routers. All the router interfaces assign labels that are used to forward packets. Value 1 is assigned to all the router interfaces so that they participate in the label processing. Router interfaces connected to the host do not perform label processing. *Platform_labels* is the table that recognizes all the assigned labels and participates in label forwarding. Value 100000 (maximum value for label forwarding) is assigned to *platform_labels* in order to enable label forwarding.

2.3 Load zebra daemon and verify the configuration

You will verify that IP addresses listed in Table 2 and inspect the routing table of the routers.

Step 1. Hold right-click on host h1 and select *Terminal*. This opens the terminal of host h1 and allows the execution of commands in that host.

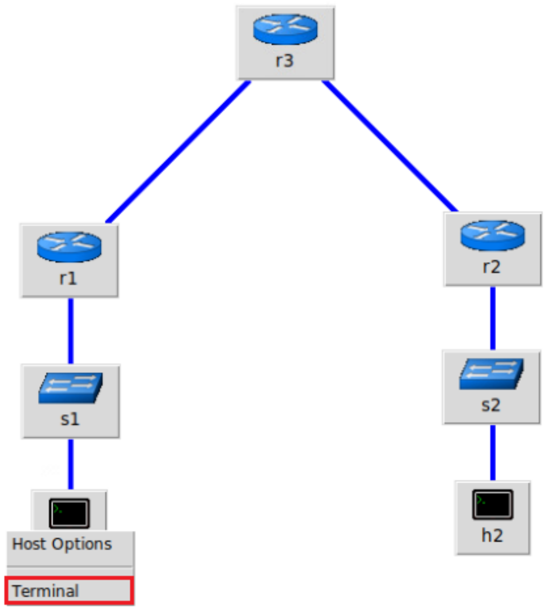
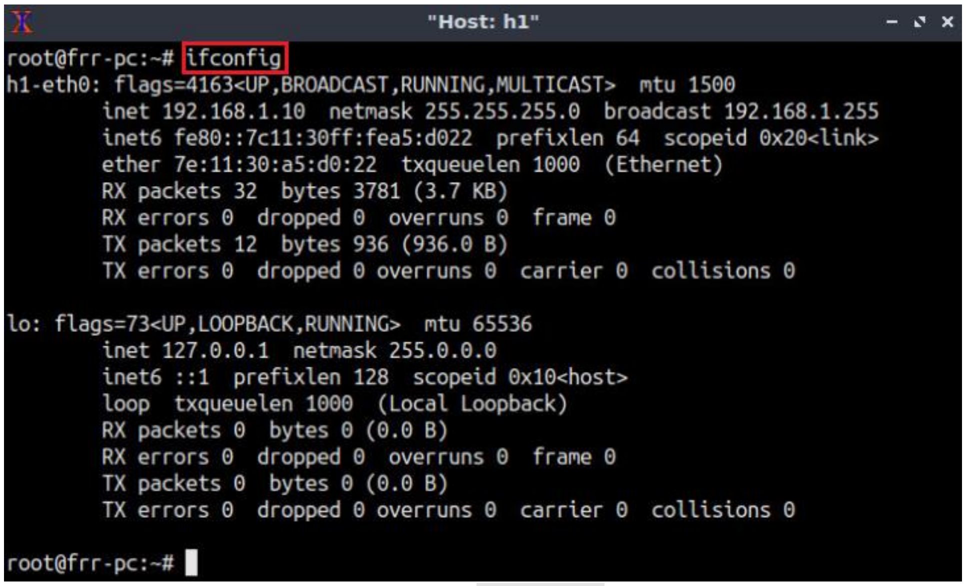


Figure 13. Opening host h1's terminal.

Step 2. In host h1 terminal, type the command shown below to verify that the IP address was assigned successfully. You will verify that interface *h1-eth0* is configured with IP address 192.168.1.10 and subnet mask 255.255.255.0.

```
ifconfig
```

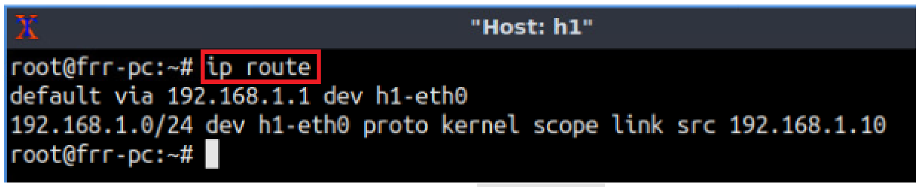


```
root@frr-pc:~# ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.10  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::7c11:30ff:fea5:d022  prefixlen 64  scopeid 0x20<link>
        ether 7e:11:30:a5:d0:22  txqueuelen 1000  (Ethernet)
        RX packets 32  bytes 3781 (3.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12  bytes 936 (936.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@frr-pc:~#
```

Figure 14. Output of `ifconfig` command.

Step 3. In host h1 terminal, type the following command to verify the default gateway IP address, 192.168.1.1.

```
ip route
```



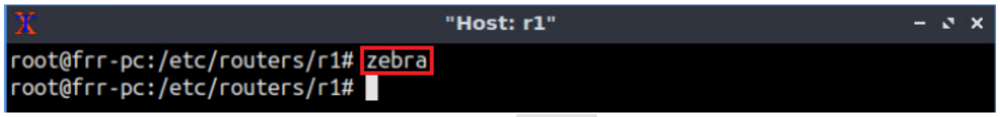
```
root@frr-pc:~# ip route
default via 192.168.1.1 dev h1-eth0
192.168.1.0/24 dev h1-eth0 proto kernel scope link src 192.168.1.10
root@frr-pc:~#
```

Figure 15. Output of the `ip route` command.

Step 4. In order to verify host h2, proceed similarly by repeating from step 1 to step 3 in host h2 terminal. Similar results should be observed.

Step 5. In router r1's terminal, you will start zebra daemon, which is a multi-server routing software that provides TCP/IP based routing protocols. The configuration will not be working if you do not enable zebra daemon initially. In order to start the zebra, type the following command:

```
zebra
```

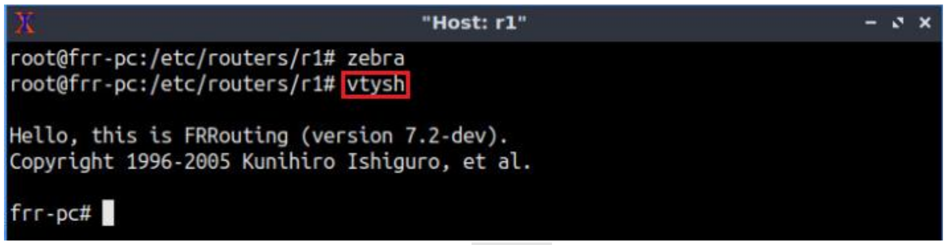


```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1#
```

Figure 16. Starting `zebra` daemon.

Step 6. After initializing `zebra`, `vtysh` should be started in order to provide all the CLI commands defined by the daemons. To proceed, issue the following command:

```
vtysh
```

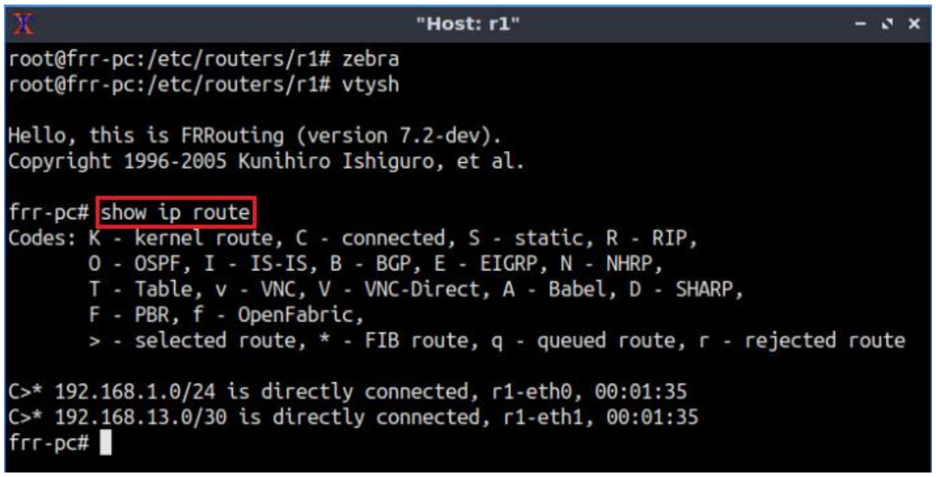


```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 17. Starting `vtysh` in router r1.

Step 7. Type the following command in router r1's terminal to verify the routing table of router r1. It will list all the directly connected networks. The routing table of router r1 does not contain any route to the network attached to router r2 (192.168.2.0/24) as there is no routing protocol configured yet.

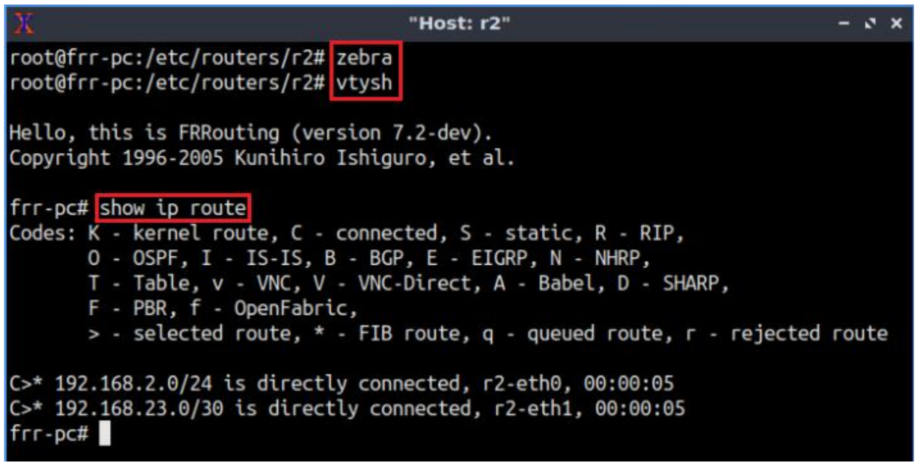
```
show ip route
```



```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.1.0/24 is directly connected, r1-eth0, 00:01:35
C>* 192.168.13.0/30 is directly connected, r1-eth1, 00:01:35
frr-pc#
```

Figure 18. Displaying routing table of router r1.

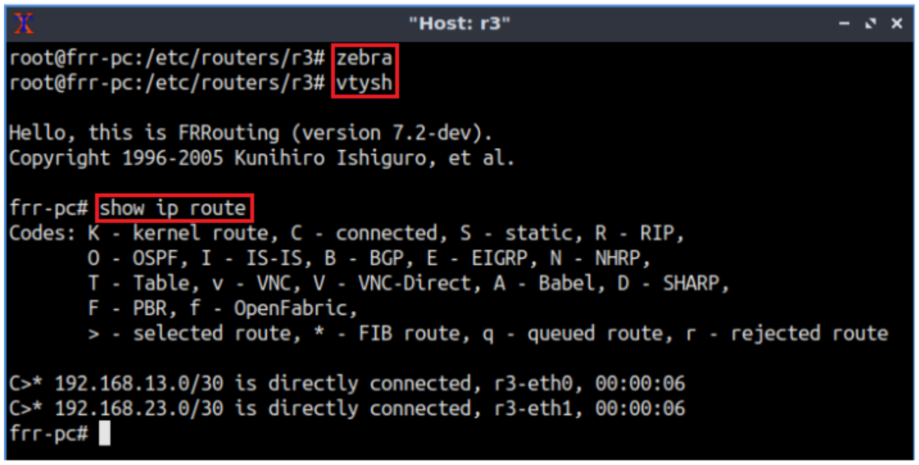
Step 8. Router r2 is configured similarly to router r1 but, with different IP addresses (see Table 2). Those steps are summarized in the following figure. To proceed, in router r2's terminal, issue the commands depicted below. At the end, you will verify all the directly connected networks of router r2.



```
"Host: r2"
root@frr-pc:/etc/routers/r2# zebra
root@frr-pc:/etc/routers/r2# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.2.0/24 is directly connected, r2-eth0, 00:00:05
C>* 192.168.23.0/30 is directly connected, r2-eth1, 00:00:05
frr-pc#
```

Figure 19. Displaying routing table of router r2.

Step 9. Router r3 is configured similarly to router r1 but, with different IP addresses (see Table 2). Those steps are summarized in the following figure. To proceed, in router r3's terminal, issue the commands depicted below. At the end, you will verify all the directly connected networks of router r3.



```
root@frr-pc:/etc/routers/r3# zebra
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.13.0/30 is directly connected, r3-eth0, 00:00:06
C>* 192.168.23.0/30 is directly connected, r3-eth1, 00:00:06
frr-pc#
```

Figure 20. Displaying routing table of router r3.

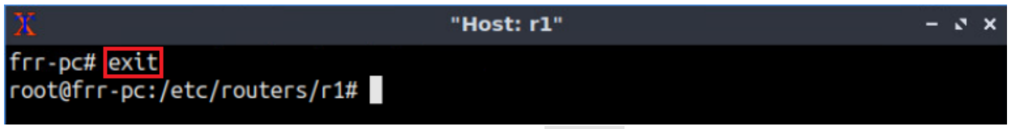
3 Configuring static MPLS from router r1 to router r2

In this section, you will configure static MPLS in routers. Router r1 will push a label to router r3. Routers r3 will swap label and the data packet will reach to router r2. Router r2 will pop the label and the data packet will be delivered to the destination host h2.

3.1 Push labels

Step 1. At this point, router r1 can reach the directly connected network 192.168.13.0/30. To communicate with other networks, you will configure static routing in router r1. To configure static routing, you need to enable the static daemon first. In router r1, type the following command to exit the `vtysh` session:

```
exit
```

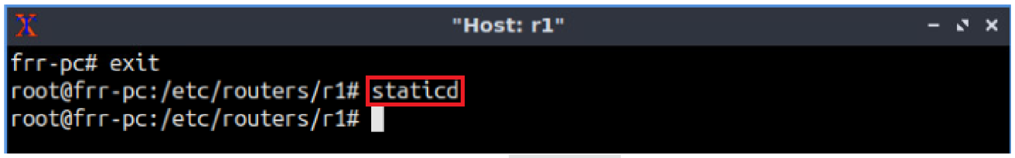


```
frr-pc# exit
root@frr-pc:/etc/routers/r1#
```

Figure 21. Exiting the `vtysh` session.

Step 2. Type the following command to enable the static routing daemon in router r1.

```
staticd
```

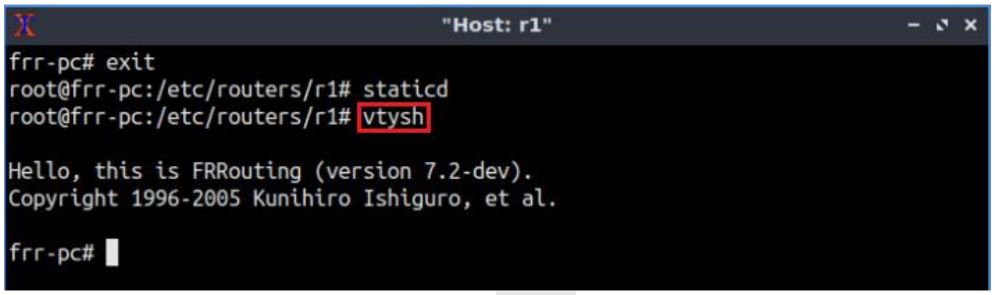


```
frr-pc# exit
root@frr-pc:/etc/routers/r1# staticd
root@frr-pc:/etc/routers/r1#
```

Figure 22. Starting `staticd` daemon.

Step 3. In order to enter router r1's terminal, issue the following command:

```
vtysh
```

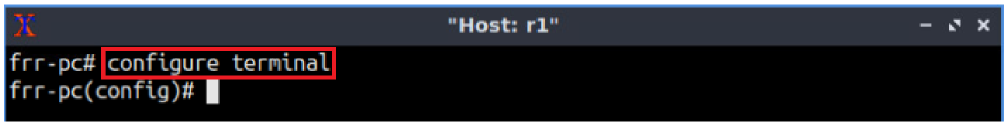


```
frr-pc# exit
root@frr-pc:/etc/routers/r1# staticd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 23. Starting `vtysh` in router r1.

Step 4. To enable router r1's configuration mode, issue the following command:

```
configure terminal
```

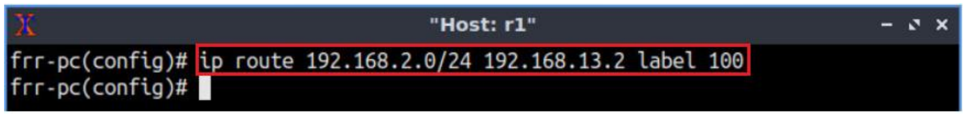


```
frr-pc# configure terminal
frr-pc(config)#
```

Figure 24. Enabling configuration mode in router r1.

Step 5. Type the following command to configure a static route to the network 192.168.2.0/24. Router r1 will assign label 100 to forward the traffic to router r3, interface *r3-eth0* (192.168.13.2).

```
ip route 192.168.2.0/24 192.168.13.2 label 100
```

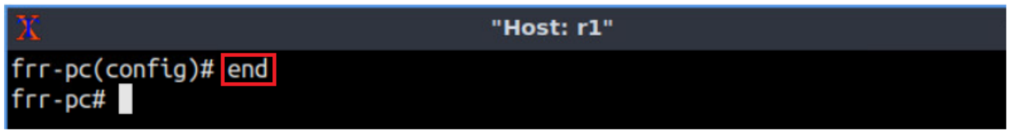


```
frr-pc(config)# ip route 192.168.2.0/24 192.168.13.2 label 100
frr-pc(config)#
```

Figure 25. Pushing label for the network 192.168.2.0/24.

Step 6. Type the following command to exit from the configuration mode.

```
end
```

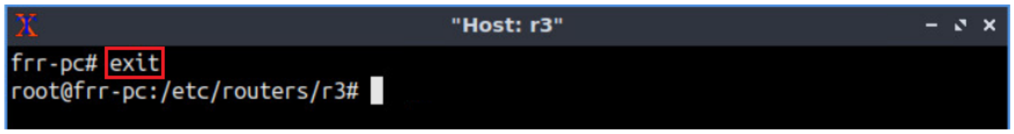


```
frr-pc(config)# end
frr-pc#
```

Figure 26. Exiting from configuration mode.

Step 7. In router r3 terminal, type the following command to exit from `vtysh`.

```
exit
```

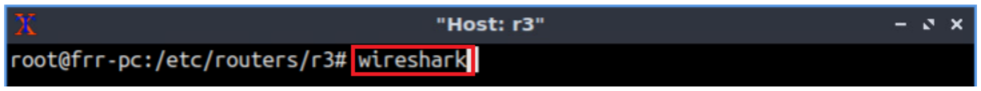


```
"Host: r3"
frr-pc# exit
root@frr-pc:/etc/routers/r3#
```

Figure 27. Exiting from vtysh.

Step 8. Type the following command to start Wireshark packet analyzer. A new window will emerge.

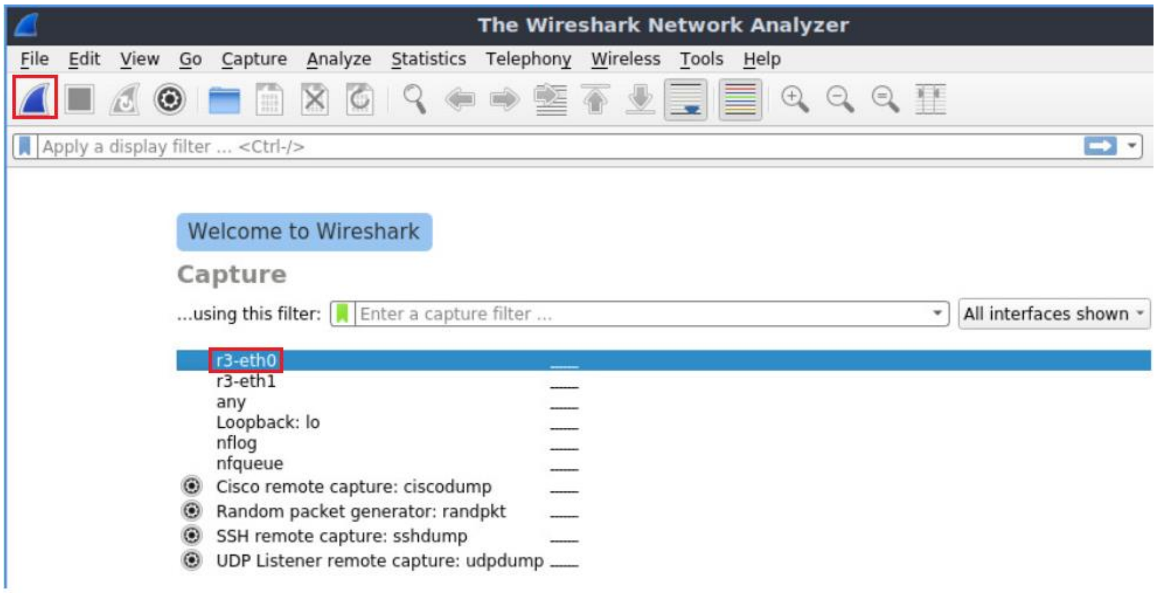
```
wireshark
```



```
"Host: r3"
root@frr-pc:/etc/routers/r3# wireshark
```

Figure 28. Starting Wireshark packet analyzer.

Step 9. Select interface *r3-eth0* and click on the icon located on the upper left-hand side to start capturing packets.

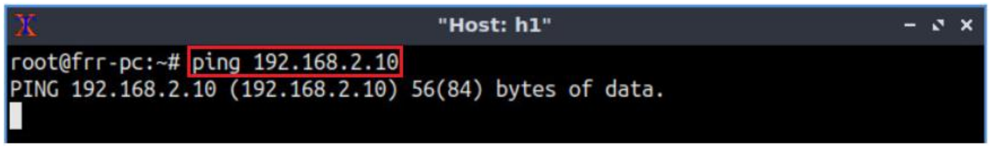


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Welcome to Wireshark
Capture
…using this filter:
Enter a capture filter …
All interfaces shown
r3-eth0
r3-eth1
any
Loopback: lo
nflog
nfqueue
Cisco remote capture: ciscodump
Random packet generator: randpkt
SSH remote capture: sshdump
UDP Listener remote capture: udpdump

Figure 29. Starting packet capture.

Step 10. Test the connectivity between hosts h1 and h2 using the `ping` command. In router r1, type the command specified below.

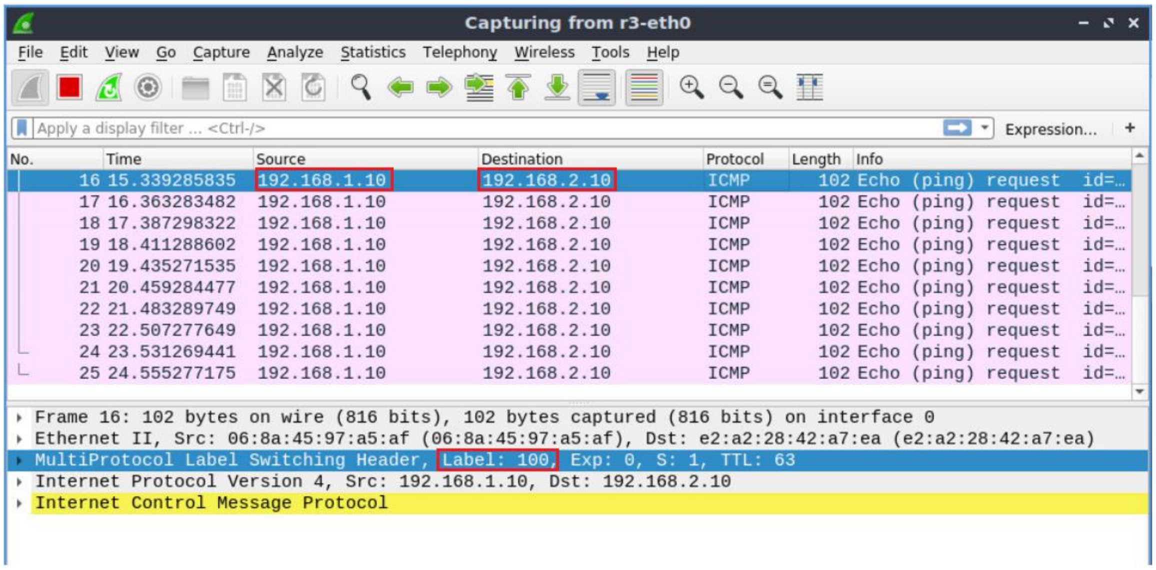
```
ping 192.168.2.10
```



```
"Host: h1"
root@frr-pc:~# ping 192.168.2.10
PING 192.168.2.10 (192.168.2.10) 56(84) bytes of data.
```

Figure 30. Output of the `ping` command in host h1.

Step 11. In Wireshark, click on any ICMP packet to see the MPLS label.


Capturing from r3-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Expression…

```
No.  Time          Source        Destination   Protocol Length Info
  16 15.339285835  192.168.1.10  192.168.2.10  ICMP     102 Echo (ping) request  id=…
  17 16.363283482  192.168.1.10  192.168.2.10  ICMP     102 Echo (ping) request  id=…
  18 17.387298322  192.168.1.10  192.168.2.10  ICMP     102 Echo (ping) request  id=…
  19 18.411288602  192.168.1.10  192.168.2.10  ICMP     102 Echo (ping) request  id=…
  20 19.435271535  192.168.1.10  192.168.2.10  ICMP     102 Echo (ping) request  id=…
  21 20.459284477  192.168.1.10  192.168.2.10  ICMP     102 Echo (ping) request  id=…
  22 21.483289749  192.168.1.10  192.168.2.10  ICMP     102 Echo (ping) request  id=…
  23 22.507277649  192.168.1.10  192.168.2.10  ICMP     102 Echo (ping) request  id=…
  24 23.531269441  192.168.1.10  192.168.2.10  ICMP     102 Echo (ping) request  id=…
  25 24.555277175  192.168.1.10  192.168.2.10  ICMP     102 Echo (ping) request  id=…

Frame 16: 102 bytes on wire (816 bits), 102 bytes captured (816 bits) on interface 0
Ethernet II, Src: 06:8a:45:97:a5:af (06:8a:45:97:a5:af), Dst: e2:a2:28:42:a7:ea (e2:a2:28:42:a7:ea)
MultiProtocol Label Switching Header, Label: 100, Exp: 0, S: 1, TTL: 63
Internet Protocol Version 4, Src: 192.168.1.10, Dst: 192.168.2.10
Internet Control Message Protocol
```

Figure 31. Verifying MPLS label.

Consider the figure above. You will notice that MPLS label 100 has been assigned. Router r1 uses label 100 to forward traffic to interface *r3-eth0* (192.168.13.2).

Close Wireshark after verifying the label. Select '*Stop and Quit without saving*' option for unsaved packets.

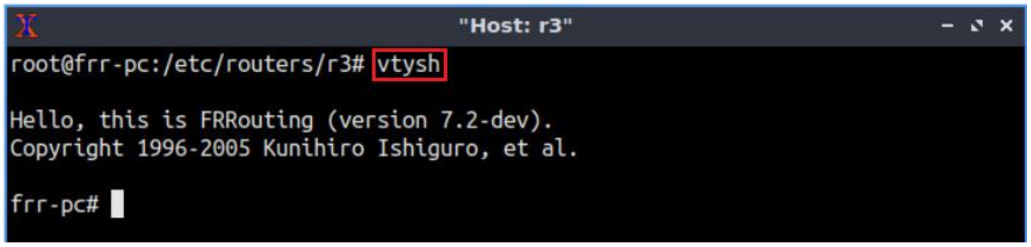
Step 12. In host h1's terminal, press `Ctrl+c` to stop the test.

3.2 Swap labels

At this point, router r1 will use label 100 to reach router r3. Router r3 will swap the label with a new label to forward the packet.

Step 1. Type the following command to enable vtysh:

```
vtysh
```

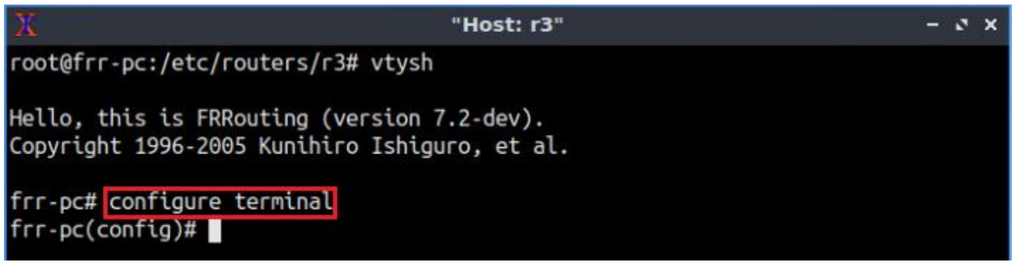


```
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 32. Enabling vtysh in router r3.

Step 2. To enable router r3 configuration mode, issue the following command:

```
configure terminal
```

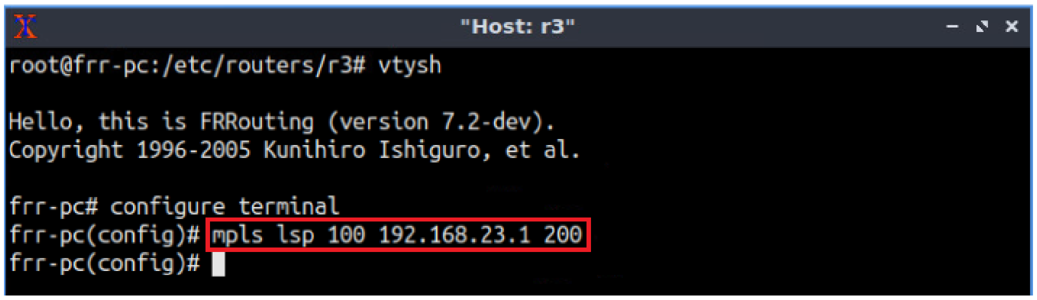


```
"Host: r3"
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)#
```

Figure 33. Enabling configuration mode in router r3.

Step 3. In this step, you will configure label swapping in router r3. Router r3 will receive label 100 from router r1 and swap the label with label 200 to forward the traffic to router r2. Type the following command to enable label swapping in router r3.

```
mpls lsp 100 192.168.23.1 200
```

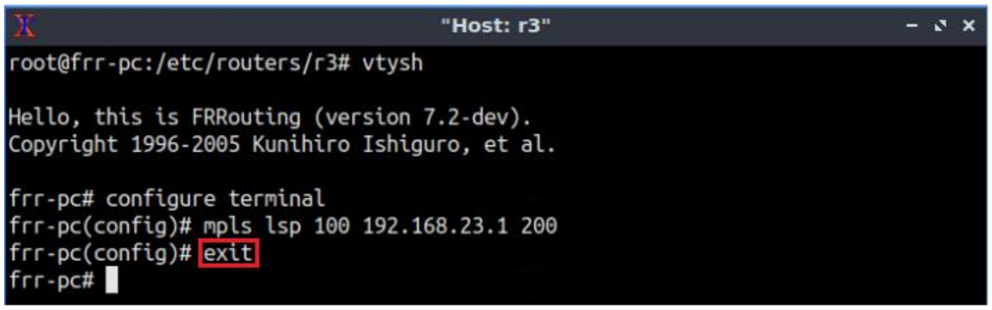


```
"Host: r3"
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# mpls lsp 100 192.168.23.1 200
frr-pc(config)#
```

Figure 34. Enabling label swapping in router r3.

Step 4. Type the following command to exit from the configuration mode.

```
exit
```



```
"Host: r3"
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# mpls lsp 100 192.168.23.1 200
frr-pc(config)# exit
frr-pc#
```

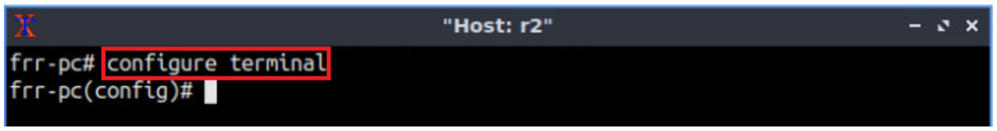
Figure 35. Exiting from configuration mode.

3.3 Pop labels

At this point, router r3 will use label 200 to reach router r2. Router r2 will pop the label and the IP packet will be delivered to the destination host h2.

Step 1. To enable configuration mode in router r2, issue the following command:

```
configure terminal
```

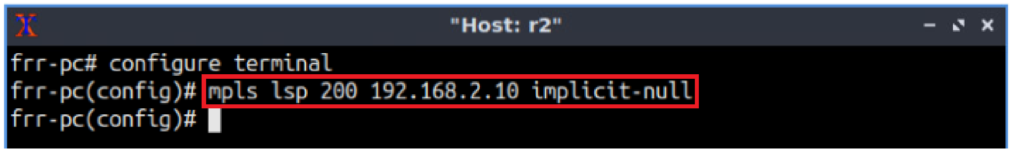


```
"Host: r2"
frr-pc# configure terminal
frr-pc(config)#
```

Figure 36. Enabling configuration mode in router r2.

Step 2. Issue the following command in router r2. Router r2 will pop the label (200) and the IP packet will be delivered to the destination host, h2 (192.168.2.10).

```
mpls lsp 200 192.168.2.10 implicit-null
```



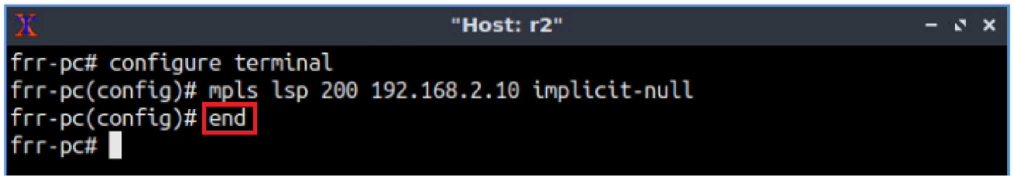
```
"Host: r2"
frr-pc# configure terminal
frr-pc(config)# mpls lsp 200 192.168.2.10 implicit-null
frr-pc(config)#
```

Figure 37. Enabling label popping in router r2.

The keyword `implicit-null` indicates that the router will perform a pop and the IP packet will be delivered to the destination.

Step 3. Type the following command to exit from the configuration mode.

```
end
```

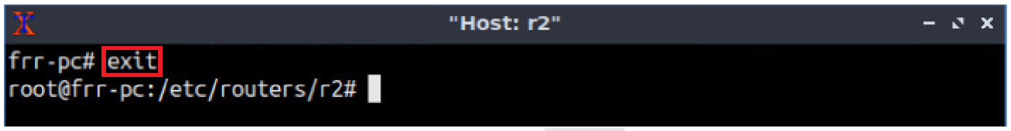


```
"Host: r2"
frr-pc# configure terminal
frr-pc(config)# mpls lsp 200 192.168.2.10 implicit-null
frr-pc(config)# end
frr-pc#
```

Figure 38. Exiting configuration mode.

Step 4. Type the following command to exit the `vtysh` session:

```
exit
```

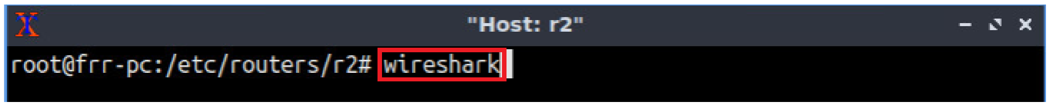


```
"Host: r2"
frr-pc# exit
root@frr-pc:/etc/routers/r2#
```

Figure 39. Exiting the `vtysh` session.

Step 5. Type the following command to start Wireshark packet analyzer. A new window will emerge.

```
wireshark
```



```
"Host: r2"
root@frr-pc:/etc/routers/r2# wireshark
```

Figure 40. Starting Wireshark packet analyzer.

Step 6. Select interface *r2-eth0* and click on the icon located on the upper left-hand side to start capturing packets.

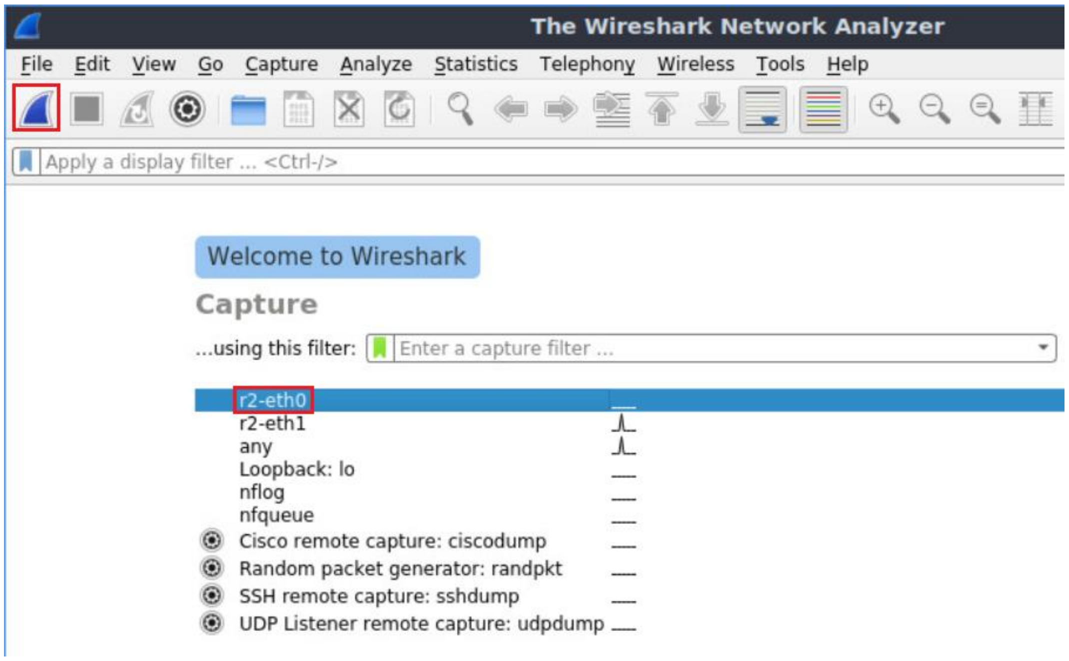
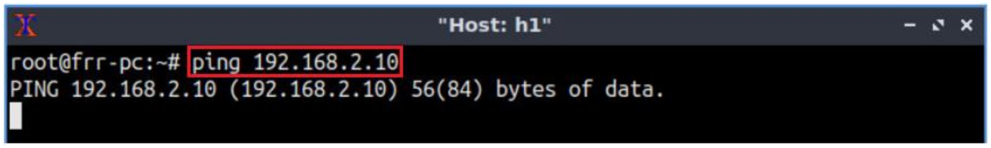

The Wireshark Network Analyzer
File
Edit
View
Go
Capture
Analyze
Statistics
Telephony
Wireless
Tools
Help
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Capture
...using this filter:
Enter a capture filter ...
r2-eth0
r2-eth1
any
Loopback: lo
nflog
nfqueue
Cisco remote capture: ciscodump
Random packet generator: randpkt
SSH remote capture: sshdump
UDP Listener remote capture: udpdump

Figure 41. Starting packet capture.

Step 7. Test the connectivity between hosts h1 and h2 using the `ping` command. In host h1, type the command specified below.

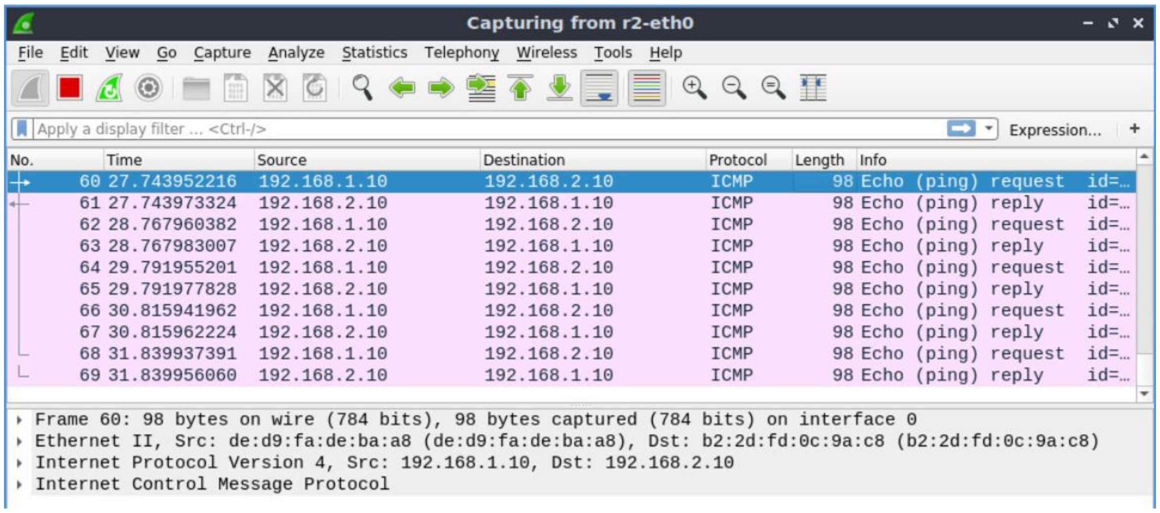
```
ping 192.168.2.10
```



```
"Host: h1"
root@frr-pc:~# ping 192.168.2.10
PING 192.168.2.10 (192.168.2.10) 56(84) bytes of data.
```

Figure 42. Output of the `ping` command in host h1.

Step 8. Click on any of the ICMP packets and verify MPLS label in Wireshark.


Capturing from r2-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.   Time          Source         Destination    Protocol  Length Info
   60 27.743952216  192.168.1.10   192.168.2.10   ICMP        98 Echo (ping) request  id=…
   61 27.743973324  192.168.2.10   192.168.1.10   ICMP        98 Echo (ping) reply    id=…
   62 28.767960382  192.168.1.10   192.168.2.10   ICMP        98 Echo (ping) request  id=…
   63 28.767983007  192.168.2.10   192.168.1.10   ICMP        98 Echo (ping) reply    id=…
   64 29.791955201  192.168.1.10   192.168.2.10   ICMP        98 Echo (ping) request  id=…
   65 29.791977828  192.168.2.10   192.168.1.10   ICMP        98 Echo (ping) reply    id=…
   66 30.815941962  192.168.1.10   192.168.2.10   ICMP        98 Echo (ping) request  id=…
   67 30.815962224  192.168.2.10   192.168.1.10   ICMP        98 Echo (ping) reply    id=…
   68 31.839937391  192.168.1.10   192.168.2.10   ICMP        98 Echo (ping) request  id=…
   69 31.839956060  192.168.2.10   192.168.1.10   ICMP        98 Echo (ping) reply    id=…

Frame 60: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
Ethernet II, Src: de:d9:fa:de:ba:a8 (de:d9:fa:de:ba:a8), Dst: b2:2d:fd:0c:9a:c8 (b2:2d:fd:0c:9a:c8)
Internet Protocol Version 4, Src: 192.168.1.10, Dst: 192.168.2.10
Internet Control Message Protocol
```

Figure 43. Verifying MPLS label.

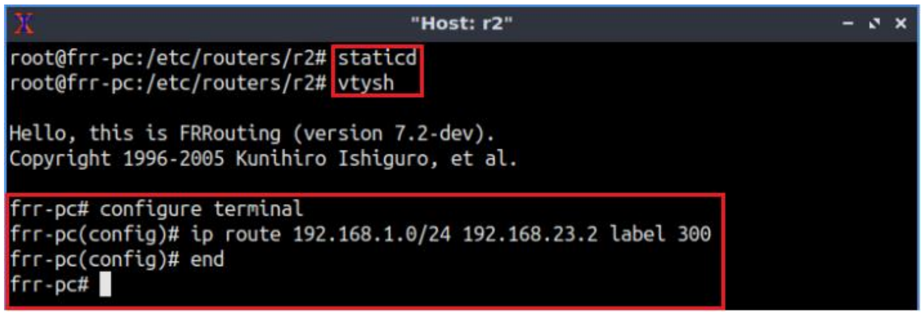
Consider the figure above. You will notice there is no MPLS label attached to the IP packet as router r2 popped the label and delivered the IP packet to its destination. You will notice that host h2 (192.168.2.10) is generating a reply for the destination host h1 (192.168.1.10) but router r2 does not have a route back to router r1.

Step 9. In host h1's terminal, press `Ctrl+c` to stop the test and close Wireshark.

4 Configuring static MPLS from router r2 to router r1

In this section, you will configure static MPLS from router r2 to router r1 so that both the hosts, h1 and h2 can ping each other.

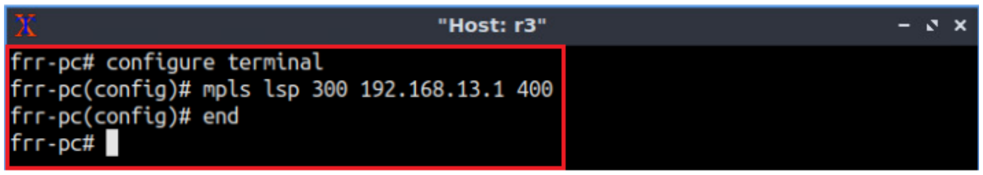
Step 1. Type the following command to create static route for router r1 network (192.168.1.0/24). Assign label 300 for the next hop 192.168.23.2. The necessary steps are summarized in the following figure.



```
root@frr-pc:/etc/routers/r2# staticd
root@frr-pc:/etc/routers/r2# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# ip route 192.168.1.0/24 192.168.23.2 label 300
frr-pc(config)# end
frr-pc#
```

Figure 44. Pushing labels in router r2.

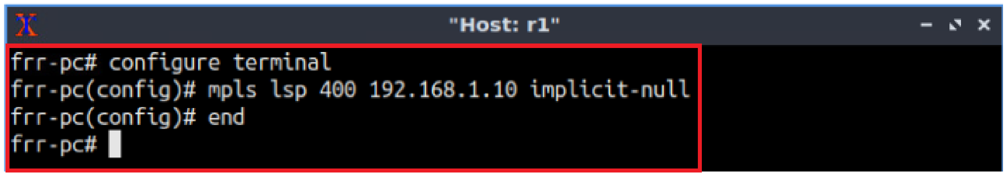
Step 2. Type the following commands in router r3. Router r3 will receive label 300, swap the label with 400 to forward the traffic to router r1 (192.168.13.1). All the commands are summarized in the following figure.



```
frr-pc# configure terminal
frr-pc(config)# mpls lsp 300 192.168.13.1 400
frr-pc(config)# end
frr-pc#
```

Figure 45. Enabling label swapping in router r3.

Step 3. Type the following commands to pop the label in router r1 which was received from router r3. Router r1 will pop the label (400) and the IP packet will be delivered to the destination host, h1 (192.168.1.10).



```
frr-pc# configure terminal
frr-pc(config)# mpls lsp 400 192.168.1.10 implicit-null
frr-pc(config)# end
frr-pc#
```

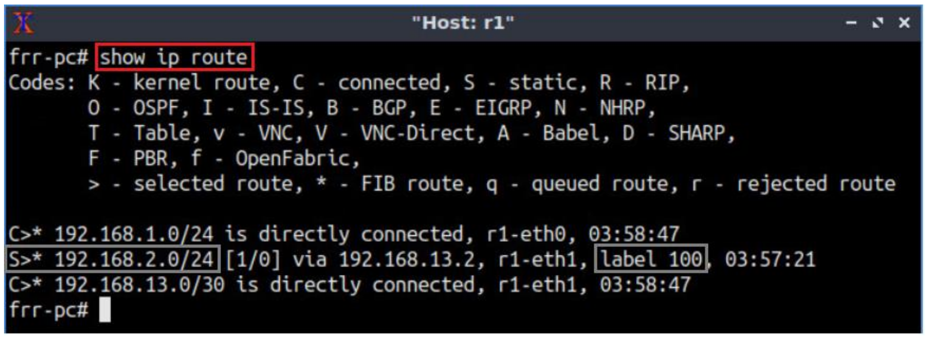
Figure 46. Popping label in router r1.

5 Verifying the configuration

In this section, you will verify the MPLS table and the connectivity between the two hosts.

Step 1. Type the following command to verify the routing table in router r1. You will notice static routes and the labels assigned to router r1.

```
show ip route
```

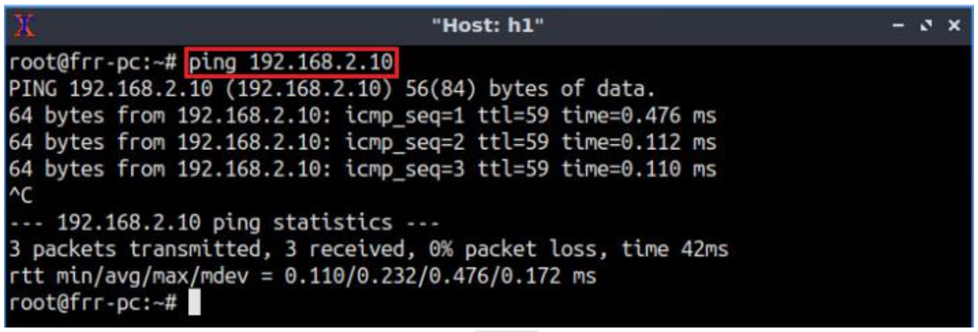


```
frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.1.0/24 is directly connected, r1-eth0, 03:58:47
S>* 192.168.2.0/24 [1/0] via 192.168.13.2, r1-eth1, label 100, 03:57:21
C>* 192.168.13.0/30 is directly connected, r1-eth1, 03:58:47
frr-pc#
```

Figure 47. Verifying routing table in router r1.

Step 2. In host h1, type the following command to test the connectivity between host h1 and host h2 using the `ping` command. To stop the test, press `Ctrl+c`. The figure below shows a successful connectivity test.

```
ping 192.168.2.10
```

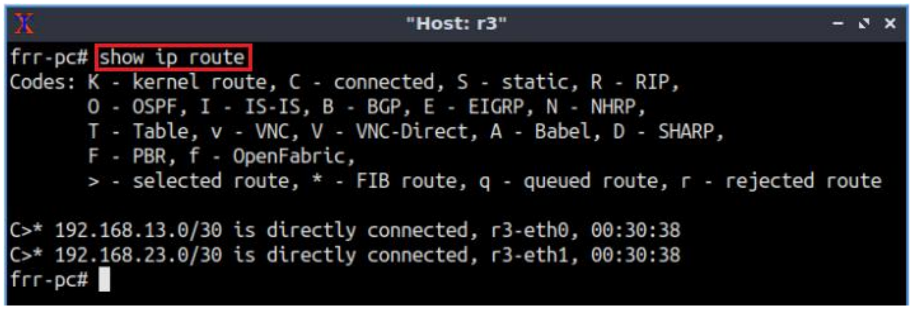


```
"Host: h1"
root@frr-pc:~# ping 192.168.2.10
PING 192.168.2.10 (192.168.2.10) 56(84) bytes of data.
64 bytes from 192.168.2.10: icmp_seq=1 ttl=59 time=0.476 ms
64 bytes from 192.168.2.10: icmp_seq=2 ttl=59 time=0.112 ms
64 bytes from 192.168.2.10: icmp_seq=3 ttl=59 time=0.110 ms
^C
--- 192.168.2.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 42ms
rtt min/avg/max/mdev = 0.110/0.232/0.476/0.172 ms
root@frr-pc:~#
```

Figure 48. Output of the `ping` command in host h2.

Step 3. Type the following command to verify routing table in router r3.

```
show ip route
```



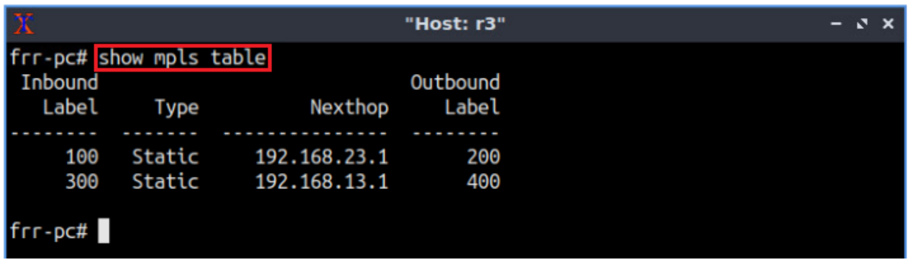
```
"Host: r3"
frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 192.168.13.0/30 is directly connected, r3-eth0, 00:30:38
C>* 192.168.23.0/30 is directly connected, r3-eth1, 00:30:38
frr-pc#
```

Figure 49. Verifying routing table in router r3.

Consider the figure above. Router r3 only knows about directly connected networks. Hosts h1 and h2 can communicate with each other since routers r3 uses MPLS labels to perform packet forwarding.

Step 4. Type the following command to verify the MPLS table in router r3.

```
show mpls table
```



```
"Host: r3"
frr-pc# show mpls table
 Inbound                              Outbound
   Label     Type          Nexthop       Label
 --------  -------  ---------------  --------
     100   Static     192.168.23.1        200
     300   Static     192.168.13.1        400

frr-pc#
```

Figure 50. Verifying MPLS table in router r3.

Consider the figure above. Router r3 forwards packets based on the labels only. If the router receives packet with label 100, it will forward the traffic to router r2 (192.168.23.1), if the receiving label is 300, the nexthop is 192.168.13.1.

This concludes Lab 4. Stop the emulation and then exit out of MiniEdit.

References

1. Luc De Ghein, “*MPLS fundamentals*”, 1st Edition, Pearson. 2006.
2. Juniper, “*MPLS label distribution protocols overview*”, 2018. [Online]. Available: https://www.juniper.net/documentation/en_US/junose15.1/topics/concept/mpls-label-distribution-protocols.html
3. Greek University, “*Static route*”, [Online]. Available: https://geek-university.com/ccna/static-routes/#:~:text=Static%20routes%20are%20manually%20added,to%20one%20of%20its%20interfaces.
4. Lydia Parziale, David T. Britt, Chuck Davis, Jason Forrester, Wei Liu, Carolyn Matthews, Nicolas Rosselot, “*TCP/IP tutorial and technical overview*”, 8th Edition, Pearson, 2006.
5. Linux foundation collaborative projects, “*FRR routing documentation*”, 2017. [Online]. Available: http://docs.frrouting.org/en/latest/zebra.html#mpls-commands
6. Juniper, “*Understanding MPLS label operations*”, 2020. [Online]. Available: https://www.juniper.net/documentation/en_US/junos/topics/concept/mpls-label-operations-qfx-series.html

UNIVERSITY OF SOUTH CAROLINA

MPLS AND ADVANCED BGP TOPICS

Lab 5: Label Distribution Protocol (LDP)

Document Version: 03-04-2021

Award 1829698
“CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers”

Contents

Overview

The lab discusses the concept of Label Distribution Protocol (LDP) that automatically generates and exchanges Multiprotocol Label Switching (MPLS) labels between routers. The lab aims to configure and verify LDP between two hosts that are required to exchange routes.

Objectives

By the end of this lab, students should be able to:

1. Explain the concept of MPLS.
2. Understand the concept of LDP.
3. Configure LDP in routers.
4. Verify configuration using *Wireshark*.

Lab settings

The information in Table 1 provides the credentials to access Client machine.

Table 1**.** Credentials to access Client machine.

Device	**Account**	**Password**
Client	admin	password

Lab roadmap

This lab is organized as follows:

1. Section 1: Introduction.
2. Section 2: Lab topology.
3. Section 3: Configure static route on routers.
4. Section 4: Configure OSPF on routers.
5. Section 5: Configure LDP on routers.
6. Section 6: Verifying configuration.

1 Introduction

1.1 Introduction to MPLS

MPLS is a networking technology that allows routers to forward traffic based on label-to-label mapping. The labels are attached to IP packets and routers can forward traffic by looking at the labels instead of IP address[2]. It works on top of an Interior Gateway Protocol (IGP) to create a predetermined path to forward traffic to the destination.

The predetermined paths that make MPLS work are called label-switched paths (LSPs). A router that operates at the edge of an MPLS network and acts as the entry and exit points for the network is called Label Edge Router (LER). The packet enters the edge of the MPLS backbone is examined and forwarded to the next hop in the pre-set LSP. As the packet travels that path, each router on the path uses the label – not other information. An MPLS router that performs routing based only on the label is called a label switch router (LSR). Each LSR has the ability to do three main things: push, pop, or swap labels from a packet. To push a label simply means to add a label to a packet, to pop is to remove a label from a packet and to swap is to remove and add an alternative label to the packet[2]. It is possible for a packet to have multiple labels attached which are arranged in a stack and are considered in the order from the most recent label to the least recent label.

1.2 Label Distribution Protocol (LDP)

LDP protocol allows each router to generate labels for its prefixes and the label values are advertised to its neighbors. It allows LSRs to discover potential peers to establish neighbor adjacency for the purpose of exchanging label information. LSPs are set hop-by-hop, and labels can be distributed between neighbors independently.

Each LSR creates a local label and then distributes this label to all its LDP neighbors. These received labels are called remote labels. The neighbors then store these remote and local labels in a special table known as Label Information Base (LIB). Each LSR has only one local label per prefix, but multiple remote labels since it usually contains more than one adjacent LSR[2].

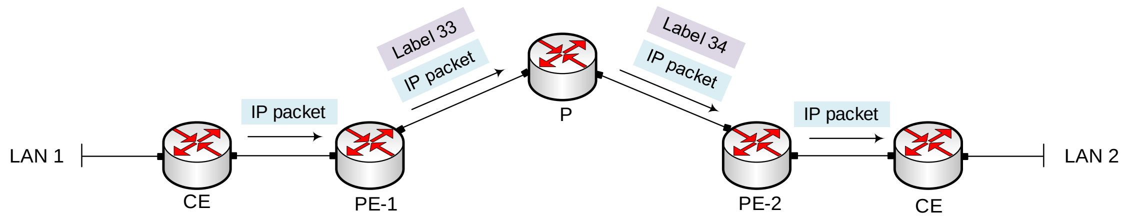


Figure 1. MPLS LDP forwarding.

Consider Figure 1. Customer Edge (CE) of Local Area Network (LAN) 1 will forward IP packet to the provider Edge (PE). Provider Edge router, PE-1 will push label 33 along with the IP packet to the provider router (P). Router P will swap the label with label 34 which will be forwarded to router PE-2. PE-2 will pop the label and the IP packet will be delivered to the destination router, CE of LAN 2.

2 Lab topology

Consider Figure 2. The topology consists of three networks, Customer 1, ISP and Customer 2. Customers (r1, r2) are communicating with ISP through static route. OSPF is running within the ISP network. MPLS LDP is configured within ISP routers to make the forwarding faster.

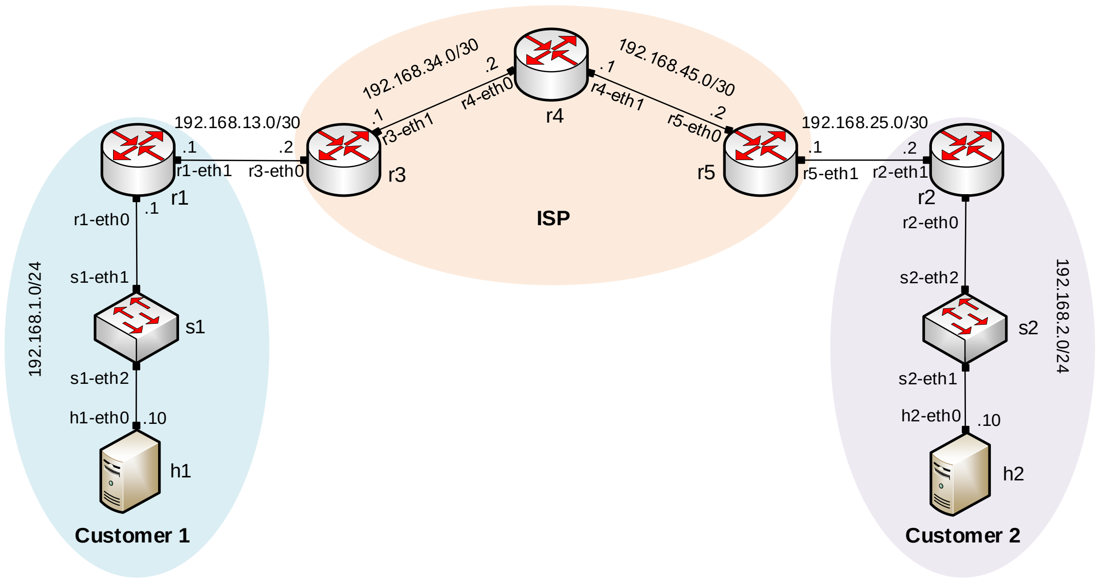


Figure 2. Lab topology.

2.1 Lab settings

Routers and hosts are already configured according to the IP addresses shown in Table 2.

Table 2. Topology information.

Device	Interface	IPV4 Address	Subnet	Default gateway
r1	r1-eth0	192.168.1.1	/24	N/A
	r1-eth1	192.168.13.1	/30	N/A
r2	r2-eth0	192.168.2.1	/24	N/A
	r2-eth1	192.168.25.1	/30	N/A
r3	r3-eth0	192.168.13.2	/30	N/A
	r3-eth1	192.168.34.1	/30	N/A
r4	r4-eth0	192.168.34.2	/30	N/A
	r4-eth1	192.168.45.1	/30	N/A
r5	r5-eth0	192.168.45.2	/30	N/A
	r5-eth1	192.168.25.2	/30	N/A

h1	h1-eth0	192.168.1.10	/24	192.168.1.1
h2	h2-eth0	192.168.2.10	/24	192.168.2.1

2.2 Open topology and load the configuration

Step 1. Start by launching MiniEdit by clicking on desktop's shortcut. When prompted for a password, type `password`.

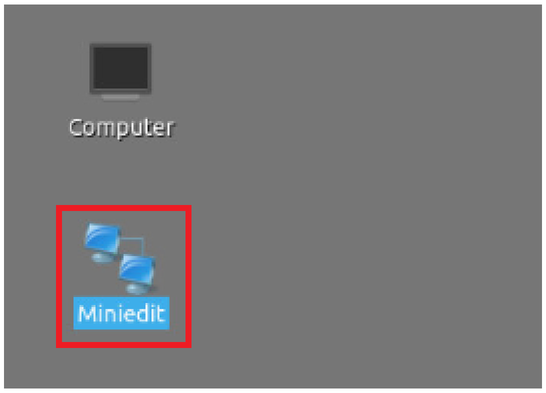
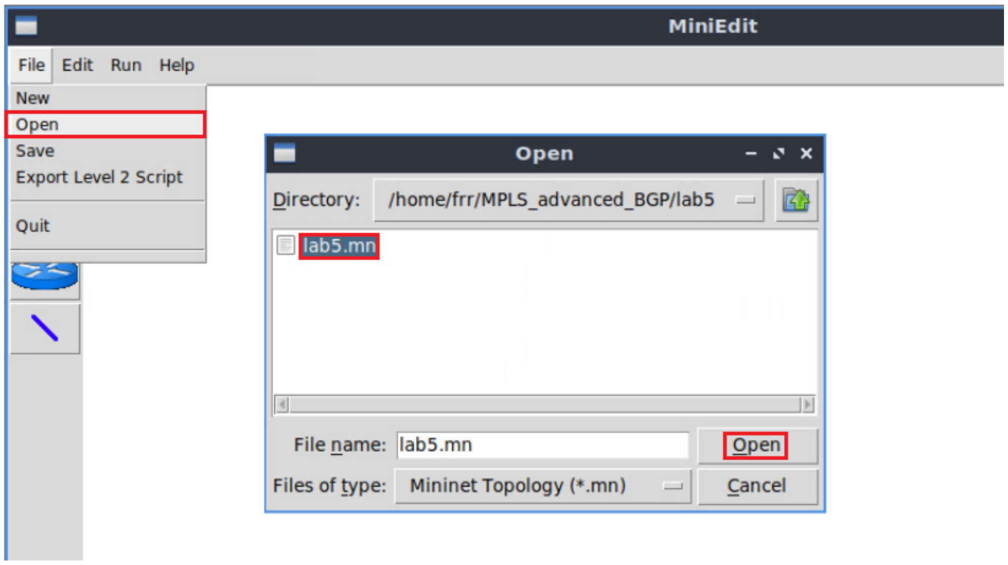


Figure 3. MiniEdit shortcut.

Step 2. On MiniEdit's menu bar, click on *File* then *open* to load the lab's topology. Locate the *lab5.mn* topology file in the default directory, */home/frr/MPLS_advanced_BGP/lab5* and click on *Open*.



Directory: /home/frr/MPLS_advanced_BGP/lab5

lab5.mn

File name: lab5.mn

Files of type: Mininet Topology (*.mn)

Figure 4. MiniEdit's Open dialog.

At this point the topology is loaded with all the required network components. You will execute a script that will load the configuration of the routers.

Step 3. Open the Linux terminal.

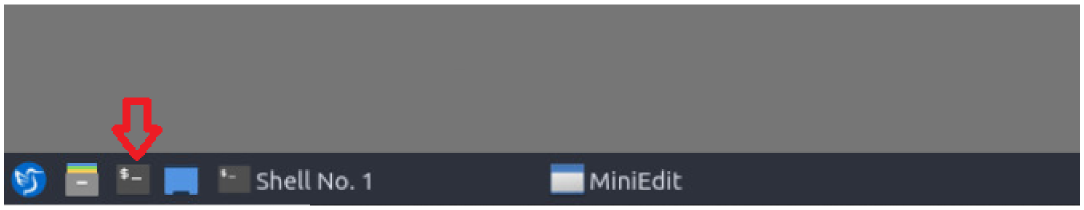
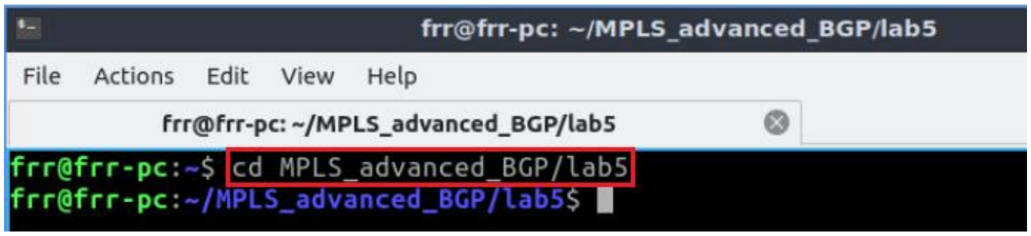


Figure 5. Opening Linux terminal.

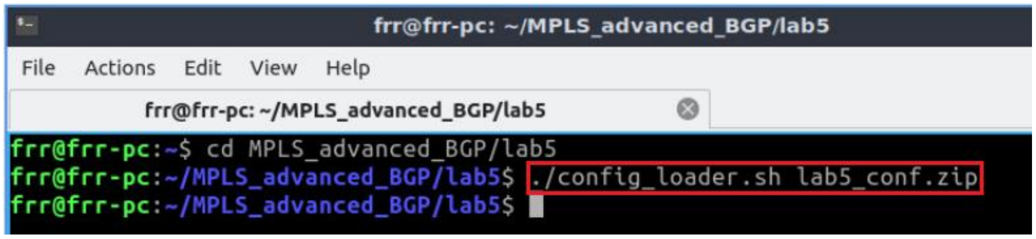
Step 4. Click on the Linux's terminal and navigate into *MPLS_advanced_BGP/lab5* directory by issuing the following command. This folder contains a configuration file and the script responsible for loading the configuration. The configuration file will assign the IP addresses to the routers' interfaces. The `cd` command is short for change directory followed by an argument that specifies the destination directory.

```
cd MPLS_advanced_BGP/lab5
```


frr@frr-pc: ~/MPLS_advanced_BGP/lab5
File
Actions
Edit
View
Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab5
```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab5
frr@frr-pc:~/MPLS_advanced_BGP/lab5$
```
Figure 6. Entering to the *MPLS_advanced_BGP/lab5* directory.

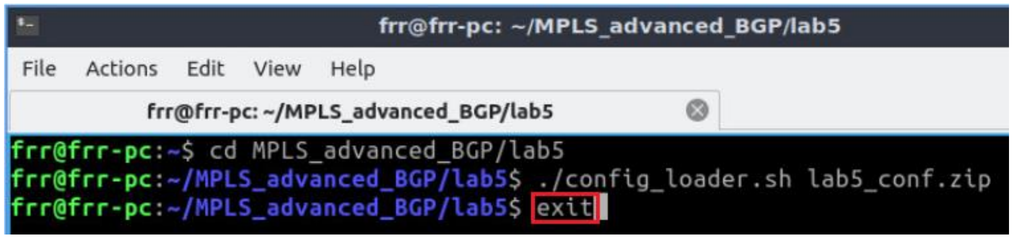
Step 5. To execute the shell script, type the following command. The argument of the program corresponds to the configuration zip file that will be loaded in all the routers in the topology.

```
./config_loader.sh lab5_conf.zip
```


frr@frr-pc: ~/MPLS_advanced_BGP/lab5
File
Actions
Edit
View
Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab5
```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab5
frr@frr-pc:~/MPLS_advanced_BGP/lab5$ ./config_loader.sh lab5_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab5$
```
Figure 7. Executing the shell script to load the configuration.

Step 6. Type the following command to exit the Linux terminal.

```
exit
```



```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab5
frr@frr-pc:~/MPLS_advanced_BGP/lab5$ ./config_loader.sh lab5_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab5$ exit
```

Figure 8. Exiting from the terminal.

Step 7. At this point hosts h1 and h2 interfaces are configured. To proceed with the emulation, click on the *Run* button located in lower left-hand side.

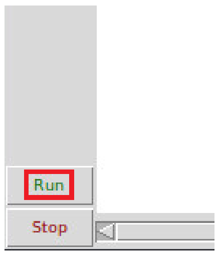


Figure 9. Starting the emulation.

Step 8. Click on Mininet's terminal, i.e., the one launched when MiniEdit was started.

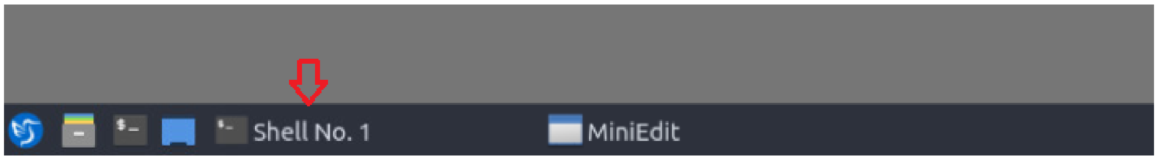
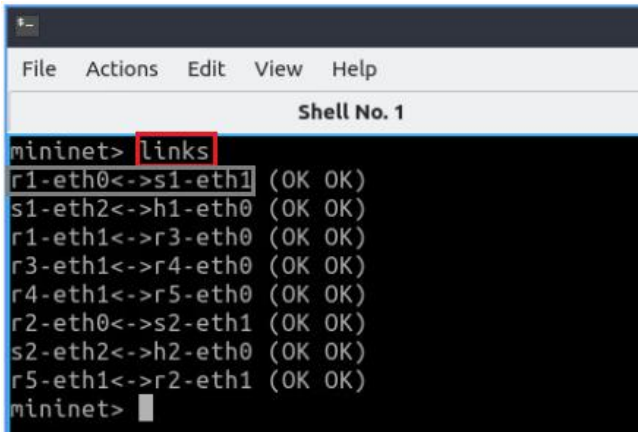


Figure 10. Opening Mininet's terminal.

Step 9. Issue the following command to display the interface names and connections.

```
links
```

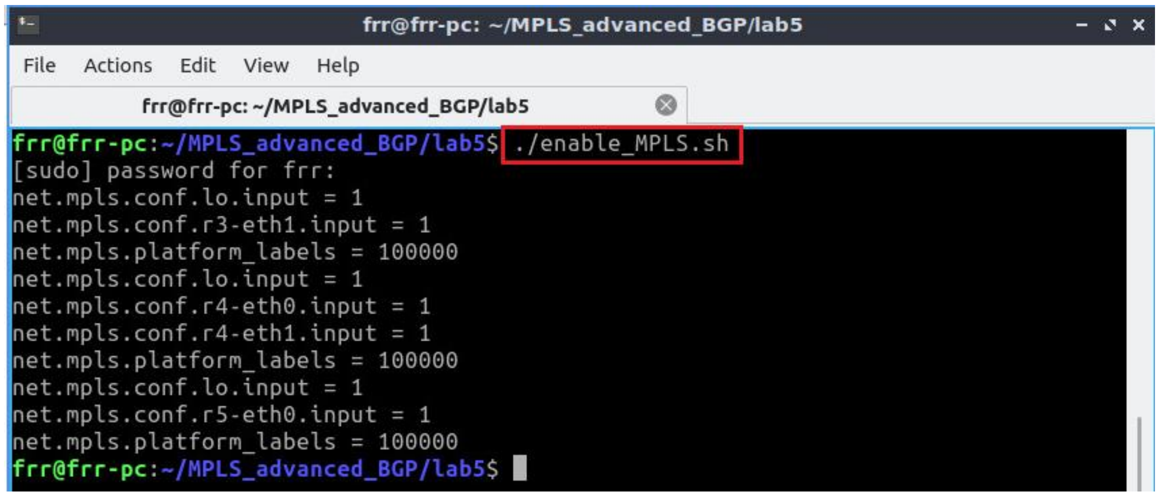


```
mininet> links
r1-eth0<->s1-eth1 (OK OK)
s1-eth2<->h1-eth0 (OK OK)
r1-eth1<->r3-eth0 (OK OK)
r3-eth1<->r4-eth0 (OK OK)
r4-eth1<->r5-eth0 (OK OK)
r2-eth0<->s2-eth1 (OK OK)
s2-eth2<->h2-eth0 (OK OK)
r5-eth1<->r2-eth1 (OK OK)
mininet>
```

Figure 11. Displaying network interfaces.

In Figure 11, the link displayed within the gray box indicates that interface eth1 of router r1 connects to interface eth1 of switch s1 (i.e., *r1-eth0<->s1-eth1*).

Step 10. Type the following command to enable MPLS forwarding in routers r3, r4 and r5. If a password is required, type `password`.


frr@frr-pc: ~/MPLS_advanced_BGP/lab5
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab5

```
frr@frr-pc:~/MPLS_advanced_BGP/lab5$ ./enable_MPLS.sh
[sudo] password for frr:
net.mpls.conf.lo.input = 1
net.mpls.conf.r3-eth1.input = 1
net.mpls.platform_labels = 100000
net.mpls.conf.lo.input = 1
net.mpls.conf.r4-eth0.input = 1
net.mpls.conf.r4-eth1.input = 1
net.mpls.platform_labels = 100000
net.mpls.conf.lo.input = 1
net.mpls.conf.r5-eth0.input = 1
net.mpls.platform_labels = 100000
frr@frr-pc:~/MPLS_advanced_BGP/lab5$
```

Figure 12. Enabling MPLS forwarding in routers r3, r4 and r5.

Consider the figure above. The command executes a shell script that enables MPLS forwarding in all routers. All the router interfaces assign labels that are used to forward packets. Value 1 is assigned to all the router interfaces so that they participate in the label processing. *Platform_labels* is the table which recognizes all the assigned labels and participates in label forwarding. Value 100000 (maximum value for label forwarding) is assigned to *platform_labels* in order to enable label forwarding.

Routers within ISP network will participate in label forwarding. Router interfaces connected to customer routers do not perform label forwarding.

2.3 Load zebra daemon and verify connectivity between routers

In this section, you will verify that IP addresses listed according to Table 2. You will also verify the routing table of the routers.

Step 1. Hold right-click on host h1 and select *Terminal*. This opens the terminal of host h1 and allows the execution of commands on that host.

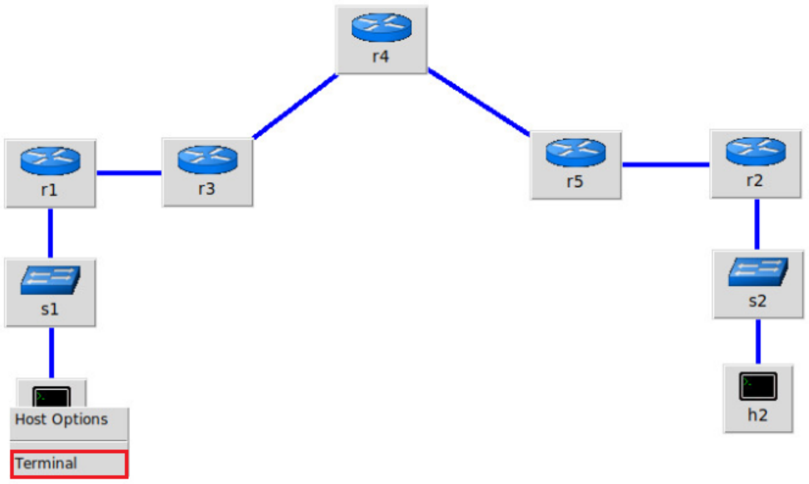
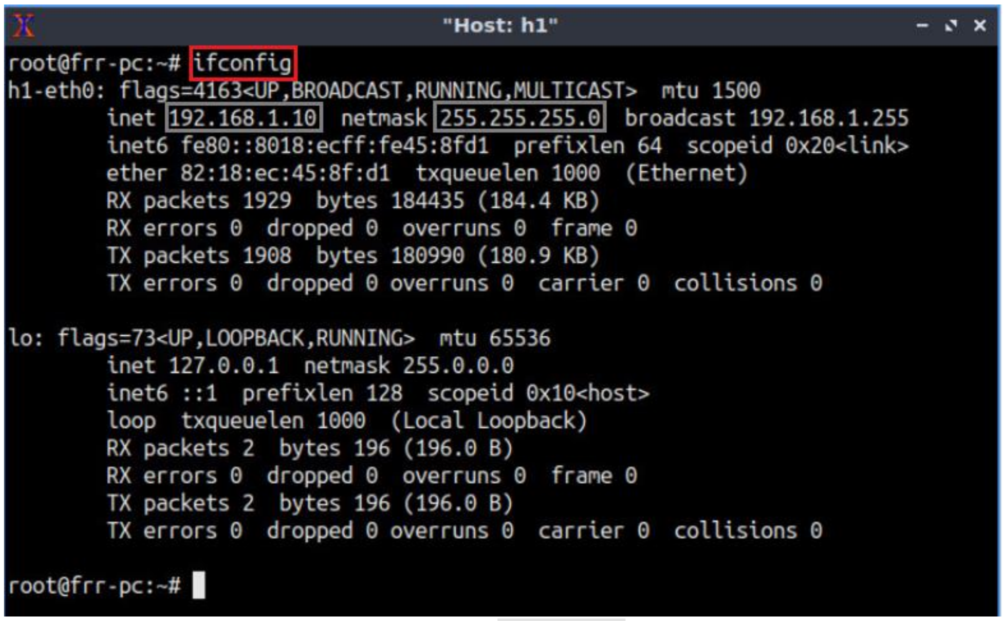


Figure 13. Opening a terminal on host h1.

Step 2. In host h1 terminal, type the following command to verify that the IP address was assigned successfully. You will verify that host h1 interface *h1-eth0* is configured with IP address 192.168.1.10 and subnet mask 255.255.255.0.

```
ifconfig
```



```
root@frr-pc:~# ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.10  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::8018:ecff:fe45:8fd1  prefixlen 64  scopeid 0x20<link>
        ether 82:18:ec:45:8f:d1  txqueuelen 1000  (Ethernet)
        RX packets 1929  bytes 184435 (184.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1908  bytes 180990 (180.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 2  bytes 196 (196.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2  bytes 196 (196.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@frr-pc:~# 
```

Figure 14. Output of `ifconfig` command.

Step 3. In order to verify host h2, proceed similarly by repeating step 1 and step 2 in host h2 terminal. Similar results should be observed.

Step 4. In order to open router r3 terminal, hold right-click on router r3 and select Terminal.

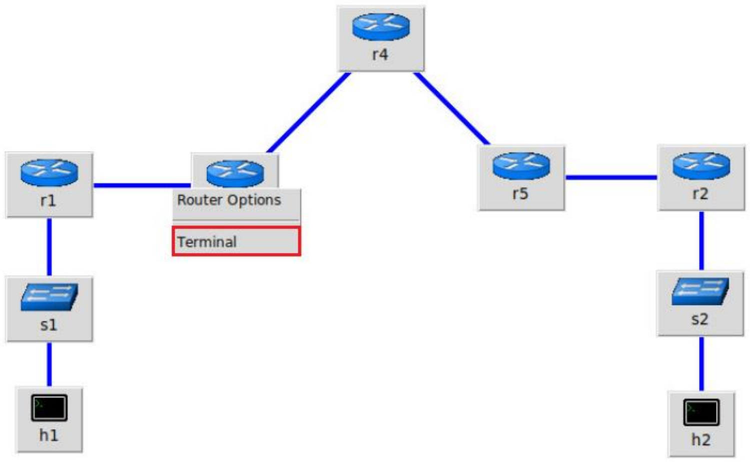
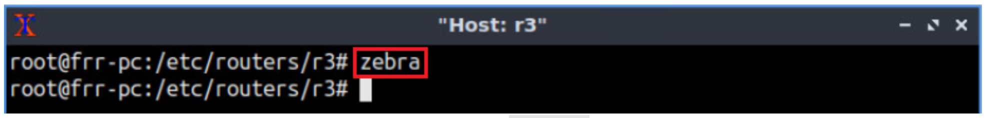


Figure 15. Opening a terminal on router r3.

Step 5. In router r3 terminal, you will start zebra daemon, which is a multi-server routing software that provides TCP/IP based routing protocols. The configuration will not be working if you do not enable zebra daemon initially. In order to start the zebra, type the following command:

```
zebra
```

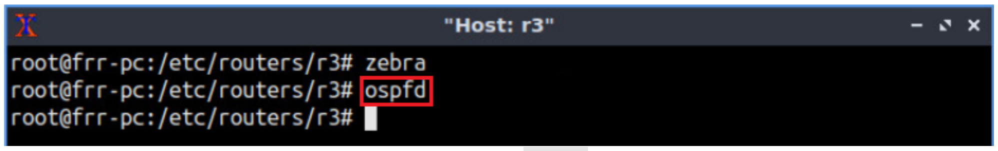


```
"Host: r3"
root@frr-pc:/etc/routers/r3# zebra
root@frr-pc:/etc/routers/r3#
```

Figure 16. Starting `zebra` daemon.

Step 6. Type the following command in router r3 terminal to enable OSPF daemon.

```
ospfd
```

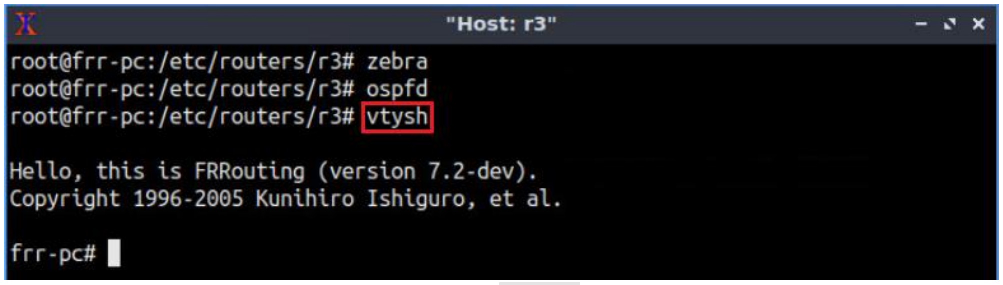


```
"Host: r3"
root@frr-pc:/etc/routers/r3# zebra
root@frr-pc:/etc/routers/r3# ospfd
root@frr-pc:/etc/routers/r3#
```

Figure 17. Starting `OSPF` daemon.

Step 7. After initializing zebra, vtysh should be started in order to provide all the CLI commands defined by the daemons. To proceed, issue the following command:

```
vtysh
```

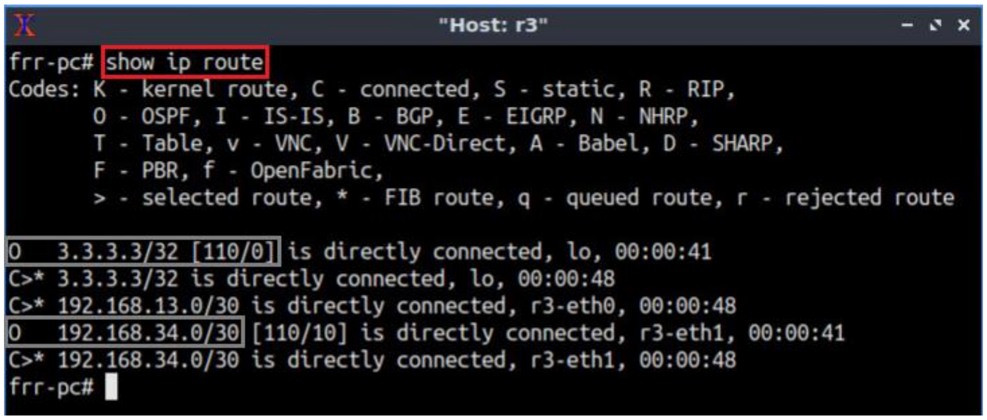


```
"Host: r3"
root@frr-pc:/etc/routers/r3# zebra
root@frr-pc:/etc/routers/r3# ospfd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 18. Starting `vtysh` in router r3.

Step 8. Type the following command in router r3 terminal to verify the routing table.

```
show ip route
```

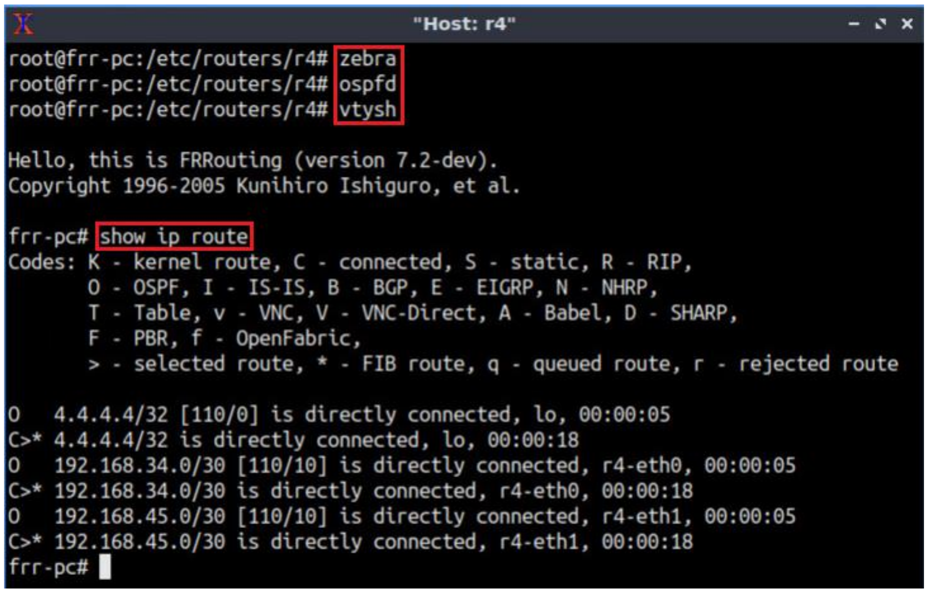


```
"Host: r3"
frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

O   3.3.3.3/32 [110/0] is directly connected, lo, 00:00:41
C>* 3.3.3.3/32 is directly connected, lo, 00:00:48
C>* 192.168.13.0/30 is directly connected, r3-eth0, 00:00:48
O   192.168.34.0/30 [110/10] is directly connected, r3-eth1, 00:00:41
C>* 192.168.34.0/30 is directly connected, r3-eth1, 00:00:48
frr-pc#
```

Figure 19. Displaying routing table of router r3.

Consider the figure above. The routing table contains OSPF networks. Router r3 will advertise the networks once OSPF is loaded in other routers.

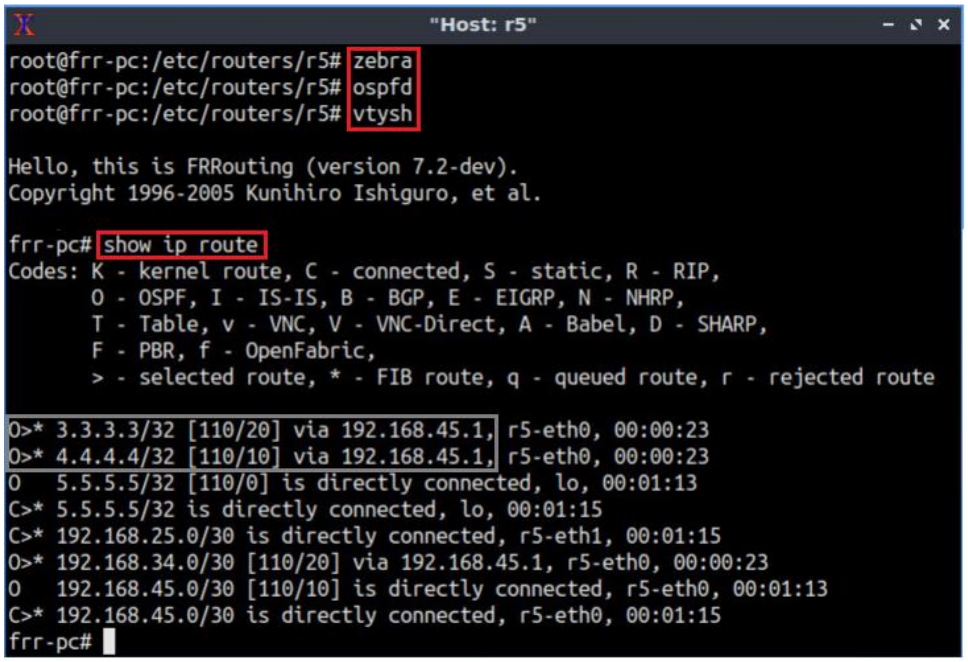
Step 9. Router r4 is configured similarly as router r3. Repeat from step 6 to step 9 to verify the routing table of router r4.



```
"Host: r4"
root@frr-pc:/etc/routers/r4# zebra
root@frr-pc:/etc/routers/r4# ospfd
root@frr-pc:/etc/routers/r4# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

O   4.4.4.4/32 [110/0] is directly connected, lo, 00:00:05
C>* 4.4.4.4/32 is directly connected, lo, 00:00:18
O   192.168.34.0/30 [110/10] is directly connected, r4-eth0, 00:00:05
C>* 192.168.34.0/30 is directly connected, r4-eth0, 00:00:18
O   192.168.45.0/30 [110/10] is directly connected, r4-eth1, 00:00:05
C>* 192.168.45.0/30 is directly connected, r4-eth1, 00:00:18
frr-pc#
```

Figure 20. Displaying routing table of router r4.

Step 10. Router r5 is configured similarly as router r3. Repeat from step 6 to step 9 to verify the routing table of router r5.



```
"Host: r5"
root@frr-pc:/etc/routers/r5# zebra
root@frr-pc:/etc/routers/r5# ospfd
root@frr-pc:/etc/routers/r5# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

O>* 3.3.3.3/32 [110/20] via 192.168.45.1, r5-eth0, 00:00:23
O>* 4.4.4.4/32 [110/10] via 192.168.45.1, r5-eth0, 00:00:23
O   5.5.5.5/32 [110/0] is directly connected, lo, 00:01:13
C>* 5.5.5.5/32 is directly connected, lo, 00:01:15
C>* 192.168.25.0/30 is directly connected, r5-eth1, 00:01:15
O>* 192.168.34.0/30 [110/20] via 192.168.45.1, r5-eth0, 00:00:23
O   192.168.45.0/30 [110/10] is directly connected, r5-eth0, 00:01:13
C>* 192.168.45.0/30 is directly connected, r5-eth0, 00:01:15
frr-pc#
```

Figure 21. Displaying routing table of router r5.

Routers will advertise their networks at this point since OSPF is loaded in all routers. The figure above shows that router r5 can communicate with loopback addresses of routers r3 (3.3.3.3) and r4 (4.4.4.4) are reachable through OSPF.

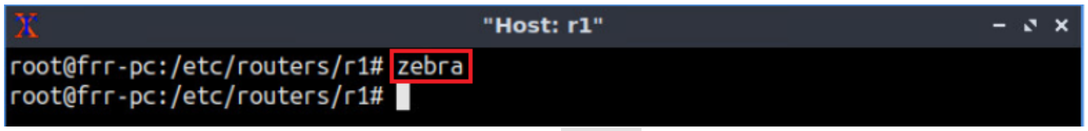
3 Configure static route in routers

Customer routers are not aware of OSPF and LDP configuration running on ISP routers. Routers r1 and r2 will be connected to ISP routers through static route.

In this section, you will configure default route in customer routers (r1, r2) so that all the customers connected to the ISP can communicate with each other. Additionally, configure static route to the CE routers (r3, r5) to provide connectivity between the customer routers and the ISP.

Step 1. Type the following command to start *zebra* daemon in router r1.

```
zebra
```

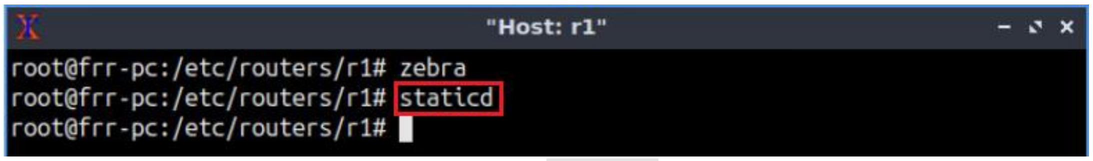


```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1#
```

Figure 22. Starting `zebra` daemon.

Step 2. Type the following command in router r1 terminal to enable *staticd* daemon.

```
staticd
```

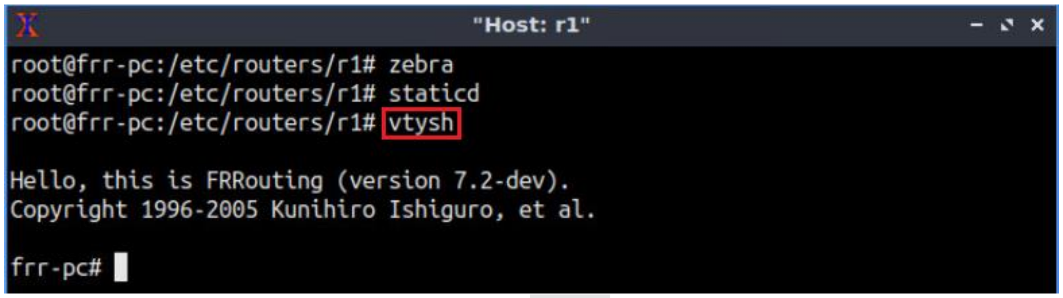


```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# staticd
root@frr-pc:/etc/routers/r1#
```

Figure 23. Starting `staticd` daemon.

Step 3. In order to enter to router r1 terminal, issue the following command:

```
vtysh
```

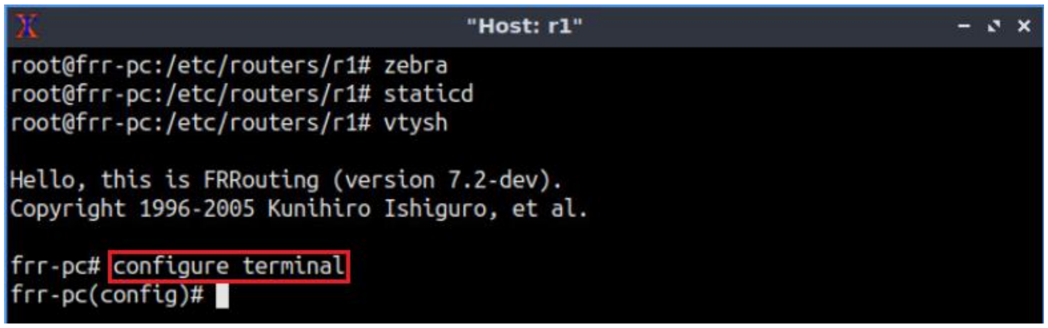


```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# staticd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 24. Starting `vtysh` in router r1.

Step 4. To enable router r1 configuration mode, issue the following command:

```
configure terminal
```

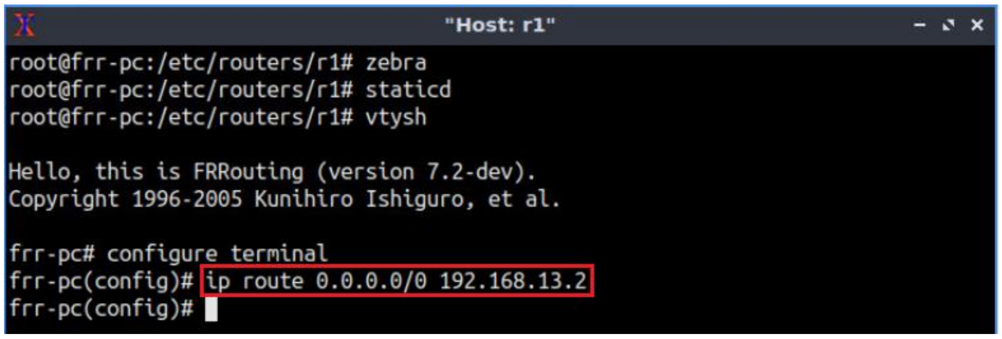


```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# staticd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)#
```

Figure 25. Enabling configuration mode in router r1.

Step 5. Type the following command to configure default route in router r1.

```
ip route 0.0.0.0/0 192.168.13.2
```



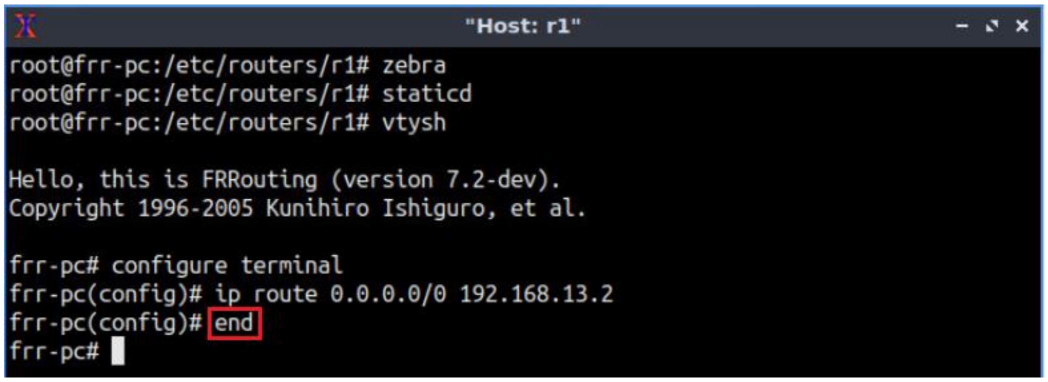
```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# staticd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# ip route 0.0.0.0/0 192.168.13.2
frr-pc(config)#
```

Figure 26. Configuring default route in router r1.

By configuring default route, router r1 will allow all the outgoing traffic via 192.168.13.2.

Step 6. Type the following command to exit from the configuration mode.

```
end
```

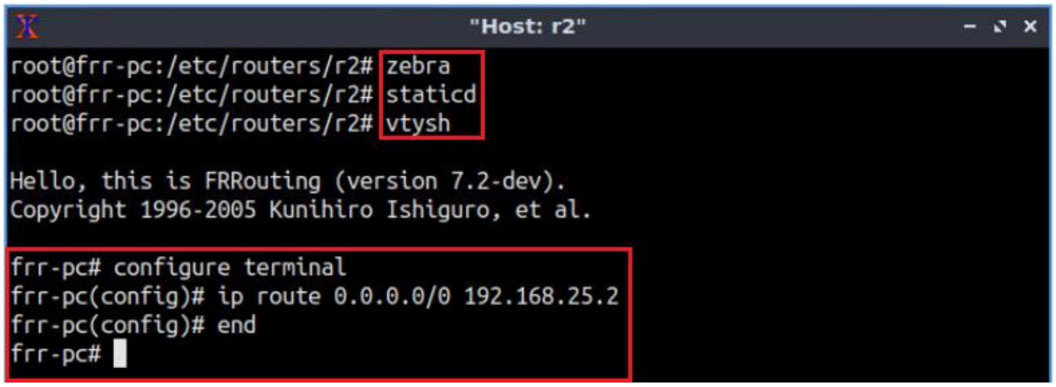


```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# staticd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# ip route 0.0.0.0/0 192.168.13.2
frr-pc(config)# end
frr-pc#
```

Figure 27. Exiting from configuration mode.

Step 7. Enable *staticd* daemon in router r2 and type the following command to configure default route. All the steps are summarized in the following figure.

```
ip route 0.0.0.0/0 192.168.25.2
```



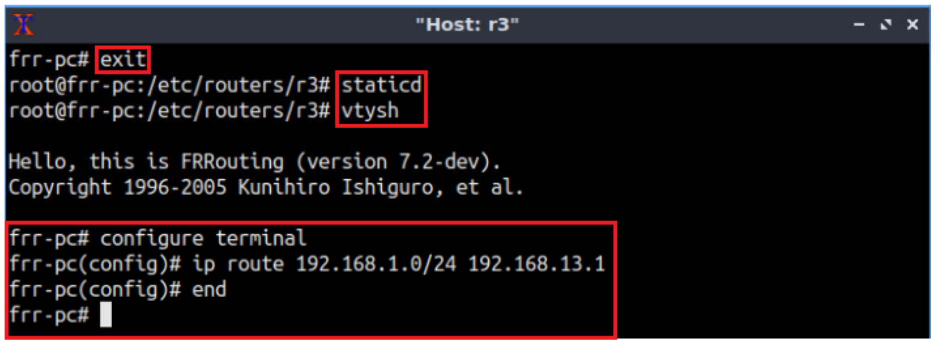
```
root@frr-pc:/etc/routers/r2# zebra
root@frr-pc:/etc/routers/r2# staticd
root@frr-pc:/etc/routers/r2# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# ip route 0.0.0.0/0 192.168.25.2
frr-pc(config)# end
frr-pc#
```

Figure 28. Configuring default route in router r2.

By configuring default route, router r2 will allow all the outgoing traffic via 192.168.25.2.

Step 8. Enable *staticd* daemon in router r3 and type the following command to configure static route. All the steps are summarized in the following figure.

```
ip route 192.168.1.0/24 192.168.13.1
```



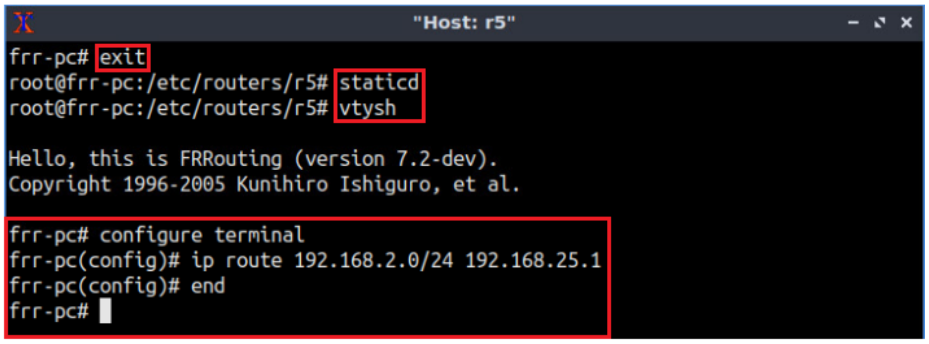
```
"Host: r3"
frr-pc# exit
root@frr-pc:/etc/routers/r3# staticd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# ip route 192.168.1.0/24 192.168.13.1
frr-pc(config)# end
frr-pc#
```

Figure 29. Configuring static route in router r3.

Consider the figure above. Router r3 can reach router r1 (192.168.1.0/24) using next-hop address, 192.168.13.1.

Step 9. Enable *staticd* daemon in router r5 and type the following command to configure static route. Router r5 can reach router r2 (192.168.2.0/24) using next-hop address 192.168.25.1. All the steps are summarized in the following figure.

```
ip route 192.168.2.0/24 192.168.25.1
```



```
"Host: r5"
frr-pc# exit
root@frr-pc:/etc/routers/r5# staticd
root@frr-pc:/etc/routers/r5# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# ip route 192.168.2.0/24 192.168.25.1
frr-pc(config)# end
frr-pc#
```

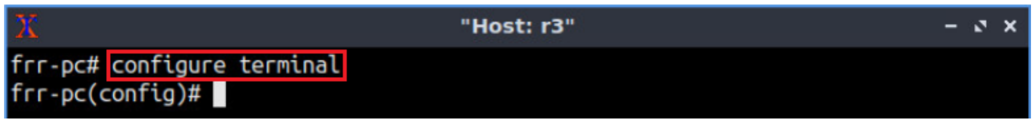
Figure 30. Configuring static route in router r5.

4 Redistribute routes in routers r3 and r5

In his section, you will inject static routes into OSPF in order to advertise networks 192.168.1.0/24 and 192.168.2.0/24 to other routers running OSPF.

Step 1. To enable router r3 configuration mode, issue the following command:

```
configure terminal
```

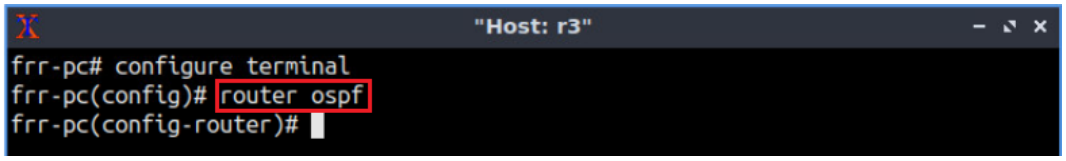


```
"Host: r3"
frr-pc# configure terminal
frr-pc(config)#
```

Figure 31. Enabling configuration mode in router r3.

Step 2. In order to configure OSPF routing protocol, type the following command. This command enables OSPF configuration mode where you advertise the networks directly connected to router r3.

```
router ospf
```

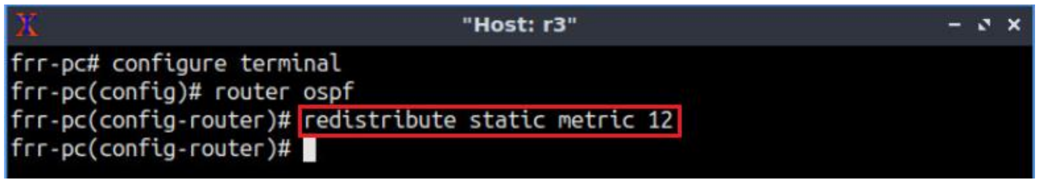


```
"Host: r3"
frr-pc# configure terminal
frr-pc(config)# router ospf
frr-pc(config-router)#
```

Figure 32. Configuring OSPF in router r3.

Step 3. You will inject static route (192.168.1.0/24) into OSPF in order to advertise the network 192.168.1.0/24 to other routers. Type the following command to redistribute static route.

```
redistribute static metric 12
```



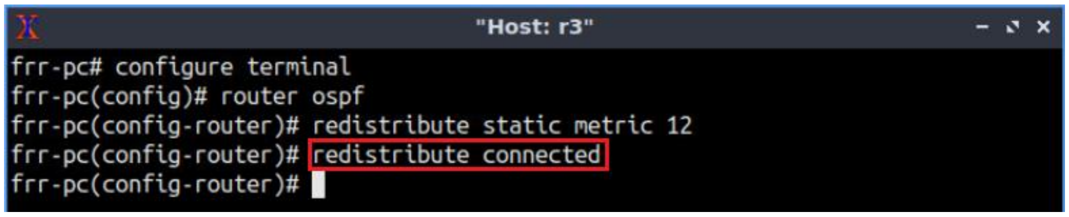
```
"Host: r3"
frr-pc# configure terminal
frr-pc(config)# router ospf
frr-pc(config-router)# redistribute static metric 12
frr-pc(config-router)#
```

Figure 33. Redistributing static route.

In order to redistribute static routes, a specific metric is required. For the purpose of this lab, you will specify the metric `12`.

Step 4. You will redistribute connected networks in order to advertise network 192.168.13.0/30 to other routers. Type the following command to redistribute connected networks.

```
redistribute connected
```

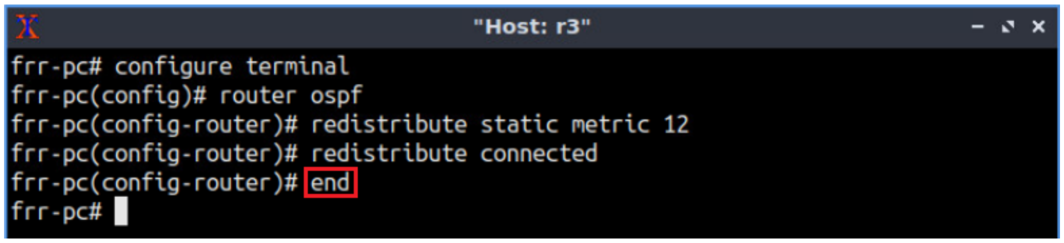


```
"Host: r3"
frr-pc# configure terminal
frr-pc(config)# router ospf
frr-pc(config-router)# redistribute static metric 12
frr-pc(config-router)# redistribute connected
frr-pc(config-router)#
```

Figure 34. Redistributing connected route.

Step 5. Type the following command to exit from the configuration mode.

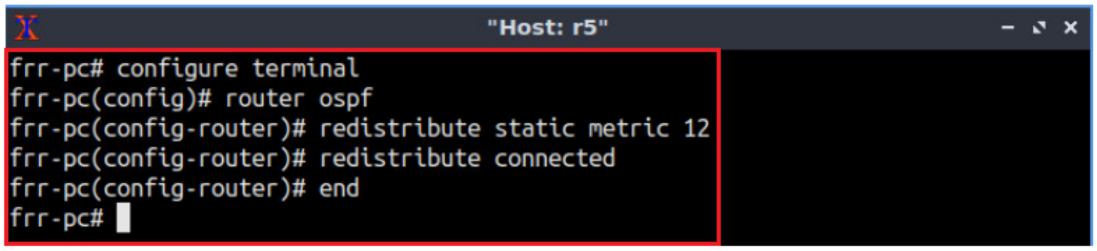
```
end
```



```
frr-pc# configure terminal
frr-pc(config)# router ospf
frr-pc(config-router)# redistribute static metric 12
frr-pc(config-router)# redistribute connected
frr-pc(config-router)# end
frr-pc#
```

Figure 35. Exiting from configuration mode.

Step 6. Router r5 is configured similarly to router r3 but, with different IP addresses (see Table 2). Those steps are summarized in the following figure. To proceed, in router r5's terminal, issue the commands depicted below.

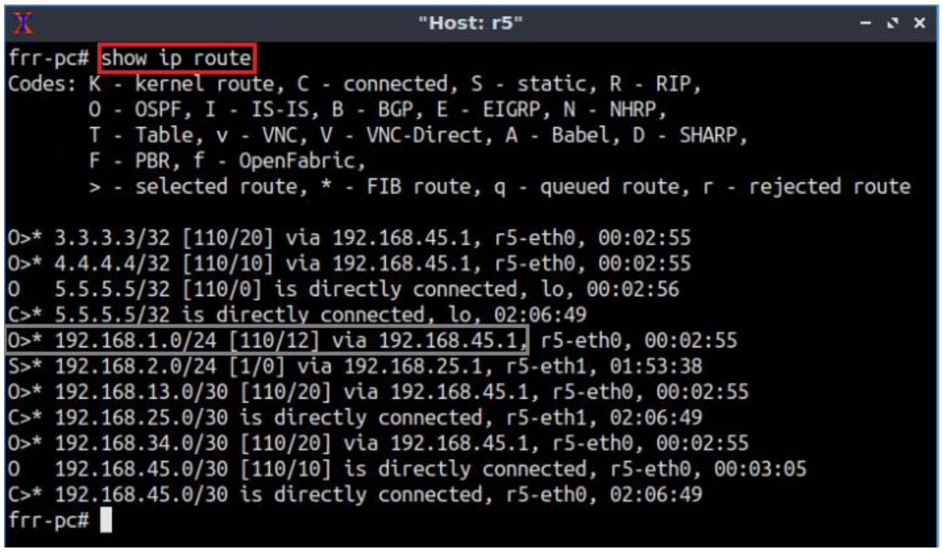


```
frr-pc# configure terminal
frr-pc(config)# router ospf
frr-pc(config-router)# redistribute static metric 12
frr-pc(config-router)# redistribute connected
frr-pc(config-router)# end
frr-pc#
```

Figure 36. Redistributing routes in router r5.

Step 7. Type the following command to verify the routing table of router r5.

```
show ip route
```



```
frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

O>* 3.3.3.3/32 [110/20] via 192.168.45.1, r5-eth0, 00:02:55
O>* 4.4.4.4/32 [110/10] via 192.168.45.1, r5-eth0, 00:02:55
O   5.5.5.5/32 [110/0] is directly connected, lo, 00:02:56
C>* 5.5.5.5/32 is directly connected, lo, 02:06:49
O>* 192.168.1.0/24 [110/12] via 192.168.45.1, r5-eth0, 00:02:55
S>* 192.168.2.0/24 [1/0] via 192.168.25.1, r5-eth1, 01:53:38
O>* 192.168.13.0/30 [110/20] via 192.168.45.1, r5-eth0, 00:02:55
C>* 192.168.25.0/30 is directly connected, r5-eth1, 02:06:49
O>* 192.168.34.0/30 [110/20] via 192.168.45.1, r5-eth0, 00:02:55
O   192.168.45.0/30 [110/10] is directly connected, r5-eth0, 00:03:05
C>* 192.168.45.0/30 is directly connected, r5-eth0, 02:06:49
frr-pc#
```

Figure 37. Verifying the routing table of router r5.

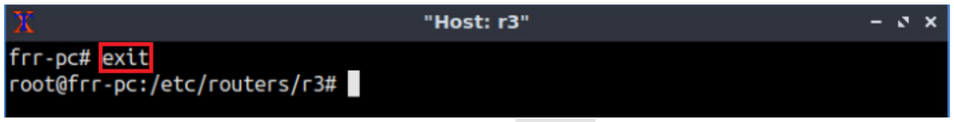
Consider the figure above. Router r5 has a route to network 192.168.1.0/24 which was learned by OSPF through redistribution.

5 Configure LDP in ISP routers

In this section, you will configure LDP within ISP routers. All the router interfaces must be reachable in order to participate in the label processing. LDP will automatically generate and exchange labels between routers.

Step 1. To configure LDP, you need to enable the LDP daemon first. In router r3, type the following command to exit the vtysh session:

```
exit
```

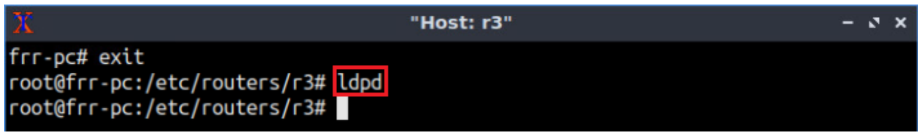


```
frr-pc# exit
root@frr-pc:/etc/routers/r3# 
```

Figure 38. Exiting the `vtysh` session.

Step 2. Type the following command in router r3 terminal to enable LDP daemon.

```
ldpd
```

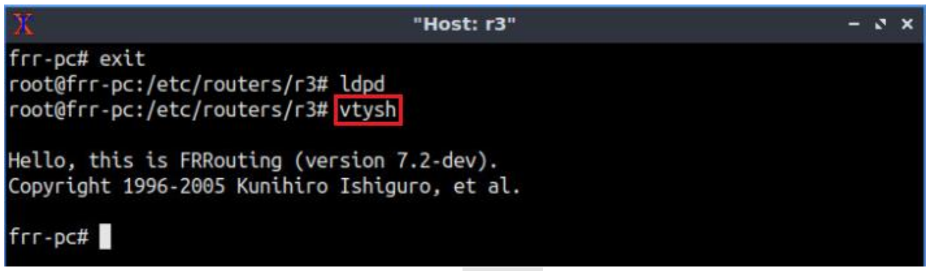


```
frr-pc# exit
root@frr-pc:/etc/routers/r3# ldpd
root@frr-pc:/etc/routers/r3# 
```

Figure 39. Starting LDP daemon.

Step 3. In order to enter to router r3's terminal, issue the following command:

```
vtysh
```

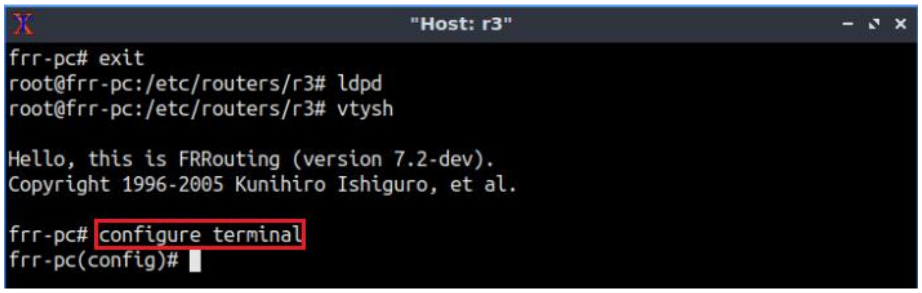


```
frr-pc# exit
root@frr-pc:/etc/routers/r3# ldpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# 
```

Figure 40. Starting `vtysh` in router r3.

Step 4. To enable router r3 configuration mode, issue the following command:

```
configure terminal
```

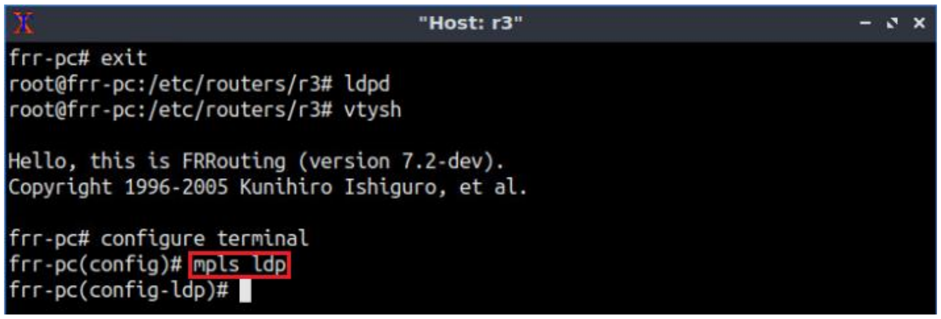


```
"Host: r3"
frr-pc# exit
root@frr-pc:/etc/routers/r3# ldpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)#
```

Figure 41. Enabling configuration mode in router r3.

Step 5. Type the following command to enable LDP configuration mode.

```
mpls ldp
```

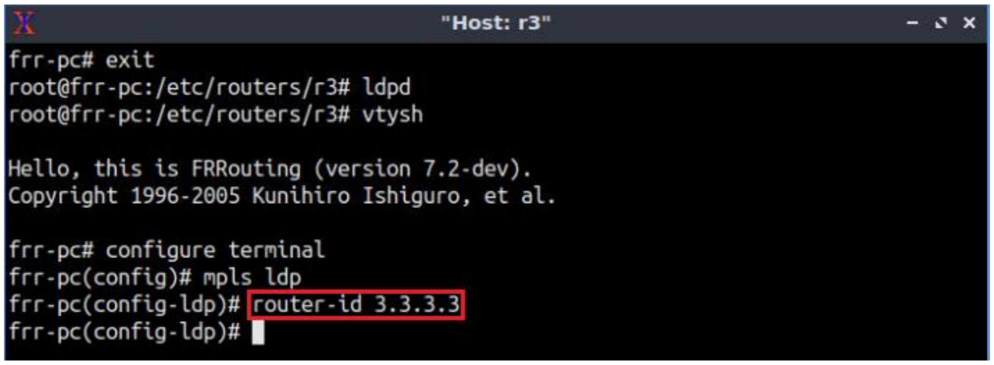


```
"Host: r3"
frr-pc# exit
root@frr-pc:/etc/routers/r3# ldpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# mpls ldp
frr-pc(config-ldp)#
```

Figure 42. Enabling LDP in router r3.

Step 6. Type the following command to set the router-id. You will use loopback address (3.3.3.3) as router-id.

```
router-id 3.3.3.3
```

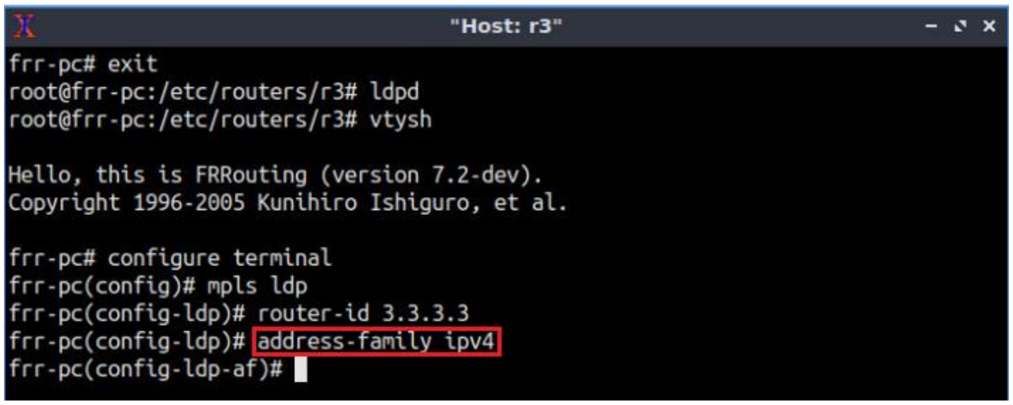


```
"Host: r3"
frr-pc# exit
root@frr-pc:/etc/routers/r3# ldpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# mpls ldp
frr-pc(config-ldp)# router-id 3.3.3.3
frr-pc(config-ldp)#
```

Figure 43. Assigning router-id in router r3.

Step 7. Type the following command in order to configure LDP for ipv4 networks.

```
address-family ipv4
```

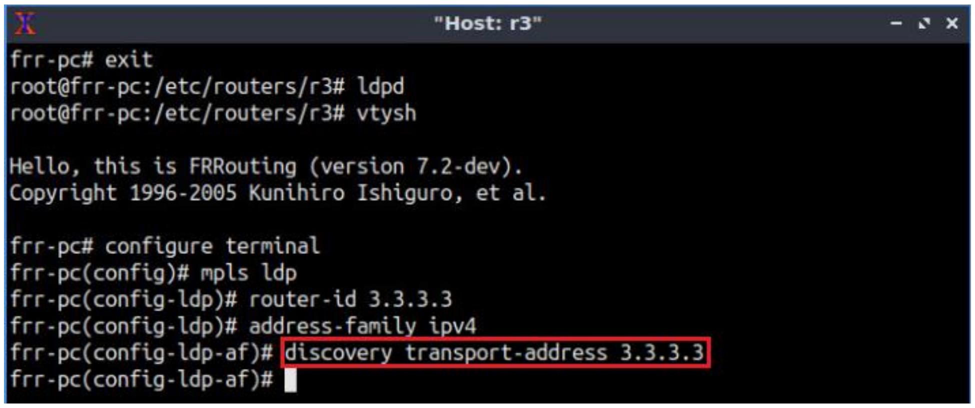


```
frr-pc# exit
root@frr-pc:/etc/routers/r3# ldpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# mpls ldp
frr-pc(config-ldp)# router-id 3.3.3.3
frr-pc(config-ldp)# address-family ipv4
frr-pc(config-ldp-af)#
```

Figure 44. Enabling address-family for IPV4 networks in router r3.

Step 8. To establish the TCP session, each router must learn the other router's transport address. The transport address is an IP address used to identify the TCP session over which the LDP session will run. Type the following command to advertise the transport address (loopback) in router r3.

```
discovery transport-address 3.3.3.3
```

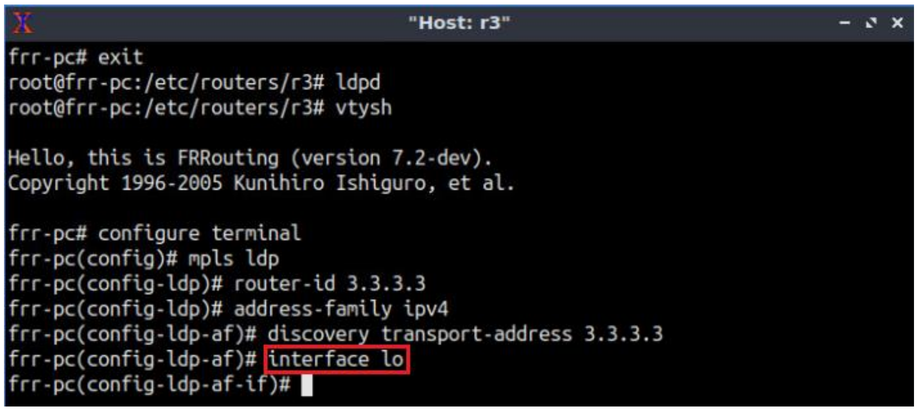


```
frr-pc# exit
root@frr-pc:/etc/routers/r3# ldpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# mpls ldp
frr-pc(config-ldp)# router-id 3.3.3.3
frr-pc(config-ldp)# address-family ipv4
frr-pc(config-ldp-af)# discovery transport-address 3.3.3.3
frr-pc(config-ldp-af)#
```

Figure 45. Assigning transport-address in router r3.

Step 9. Type the following command to enable interface *lo* in order to participate in the LDP session.

```
interface lo
```

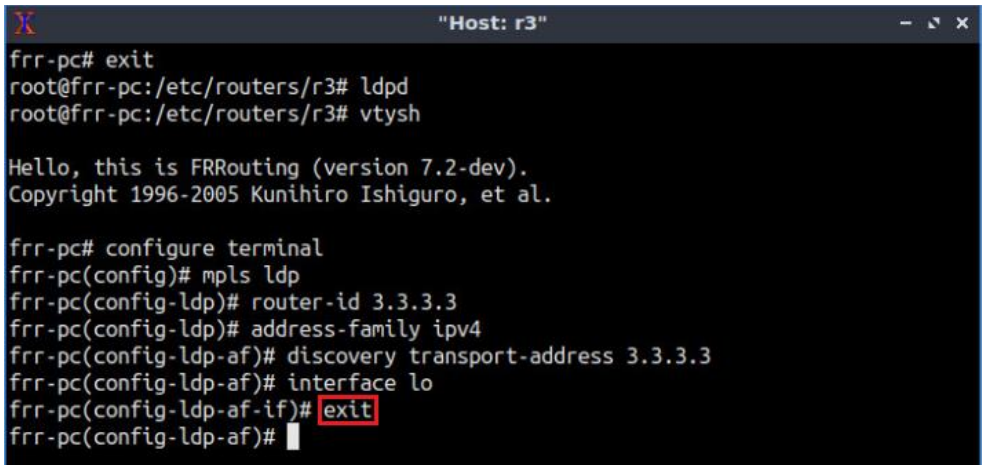


```
frr-pc# exit
root@frr-pc:/etc/routers/r3# ldpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# mpls ldp
frr-pc(config-ldp)# router-id 3.3.3.3
frr-pc(config-ldp)# address-family ipv4
frr-pc(config-ldp-af)# discovery transport-address 3.3.3.3
frr-pc(config-ldp-af)# interface lo
frr-pc(config-ldp-af-if)#
```

Figure 46. Assigning interface *lo* to participate in the LDP session.

Step 10. Type the following command to exit from the interface mode.

```
exit
```



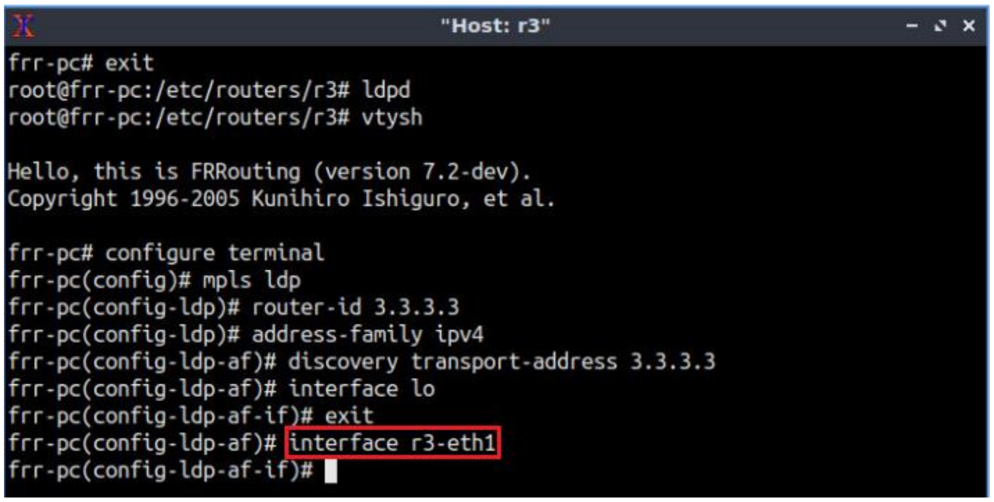
"Host: r3"

```
frr-pc# exit
root@frr-pc:/etc/routers/r3# ldpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# mpls ldp
frr-pc(config-ldp)# router-id 3.3.3.3
frr-pc(config-ldp)# address-family ipv4
frr-pc(config-ldp-af)# discovery transport-address 3.3.3.3
frr-pc(config-ldp-af)# interface lo
frr-pc(config-ldp-af-if)# exit
frr-pc(config-ldp-af)#
```

Figure 47. Exiting from *lo* interface.

Step 11. Type the following command to enable interface *r3-eth1* in order to participate in the LDP session.

```
interface r3-eth1
```



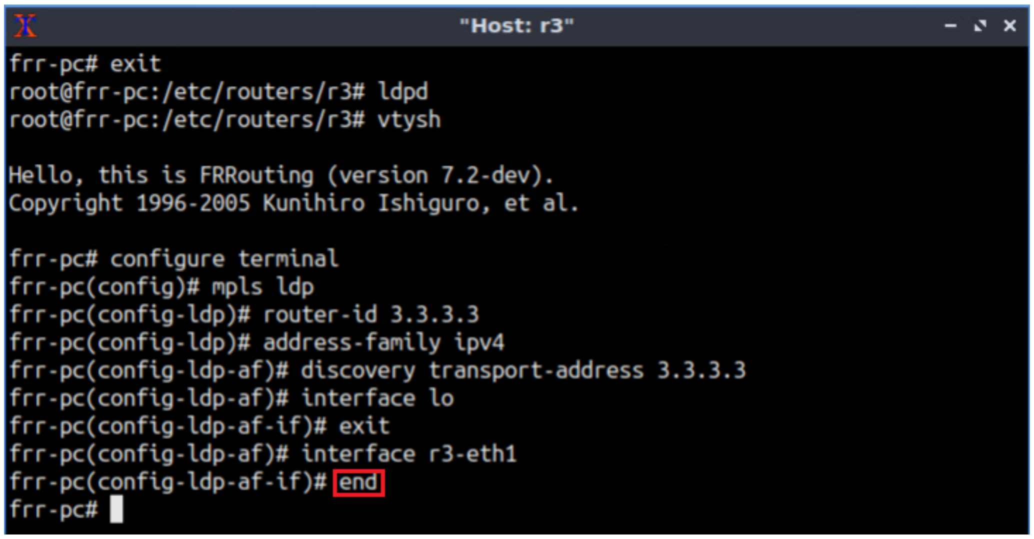
"Host: r3"

```
frr-pc# exit
root@frr-pc:/etc/routers/r3# ldpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# mpls ldp
frr-pc(config-ldp)# router-id 3.3.3.3
frr-pc(config-ldp)# address-family ipv4
frr-pc(config-ldp-af)# discovery transport-address 3.3.3.3
frr-pc(config-ldp-af)# interface lo
frr-pc(config-ldp-af-if)# exit
frr-pc(config-ldp-af)# interface r3-eth1
frr-pc(config-ldp-af-if)#
```

Figure 48. Assigning interface *r3-eth1* to participate in the LDP session.

Step 12. Type the following command to exit from the configuration mode.

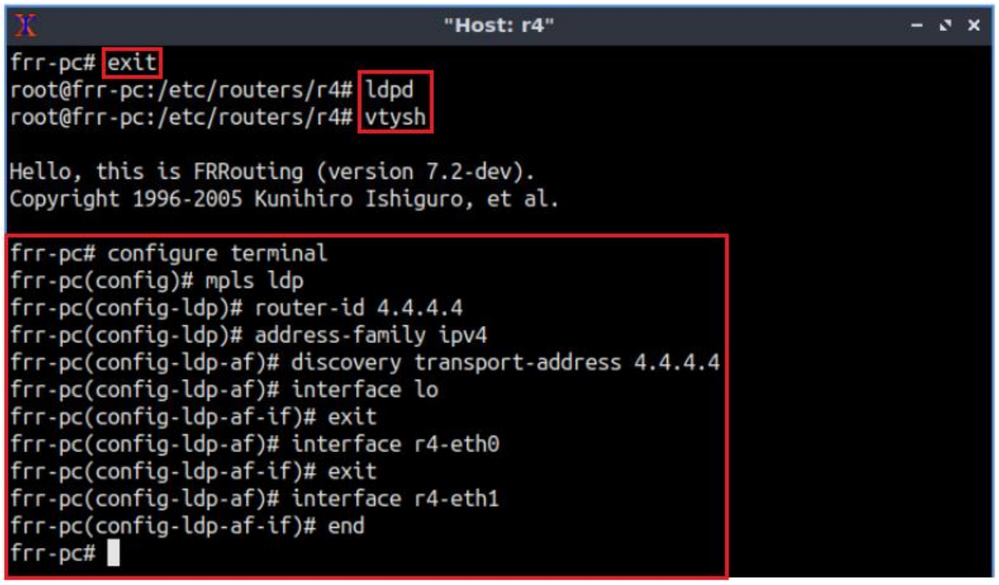
```
end
```



```
frr-pc# exit
root@frr-pc:/etc/routers/r3# ldpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# mpls ldp
frr-pc(config-ldp)# router-id 3.3.3.3
frr-pc(config-ldp)# address-family ipv4
frr-pc(config-ldp-af)# discovery transport-address 3.3.3.3
frr-pc(config-ldp-af)# interface lo
frr-pc(config-ldp-af-if)# exit
frr-pc(config-ldp-af)# interface r3-eth1
frr-pc(config-ldp-af-if)# end
frr-pc#
```

Figure 49. Exiting from configuration mode.

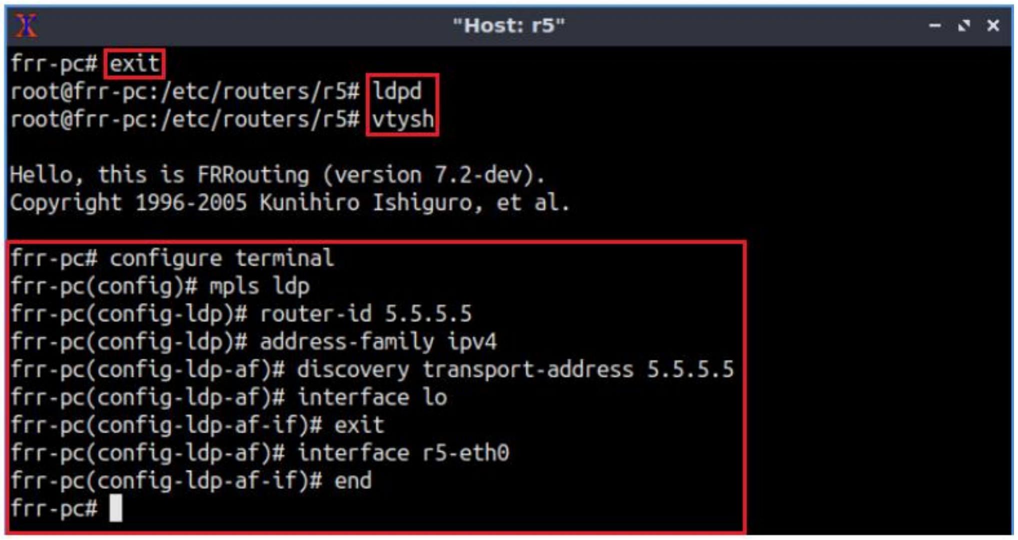
Step 13. Router r4 is configured similarly to router r3 but, with different metrics. Those steps are summarized in the following figure. To proceed, in router r4 terminal, issue the commands depicted below.



```
frr-pc# exit
root@frr-pc:/etc/routers/r4# ldpd
root@frr-pc:/etc/routers/r4# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# mpls ldp
frr-pc(config-ldp)# router-id 4.4.4.4
frr-pc(config-ldp)# address-family ipv4
frr-pc(config-ldp-af)# discovery transport-address 4.4.4.4
frr-pc(config-ldp-af)# interface lo
frr-pc(config-ldp-af-if)# exit
frr-pc(config-ldp-af)# interface r4-eth0
frr-pc(config-ldp-af-if)# exit
frr-pc(config-ldp-af)# interface r4-eth1
frr-pc(config-ldp-af-if)# end
frr-pc#
```

Figure 50. Configuring LDP in router r4.

Step 14. Router r5 is configured similarly to router r3 but, with different metrics. Those steps are summarized in the following figure. To proceed, in router r5 terminal, issue the commands depicted below.



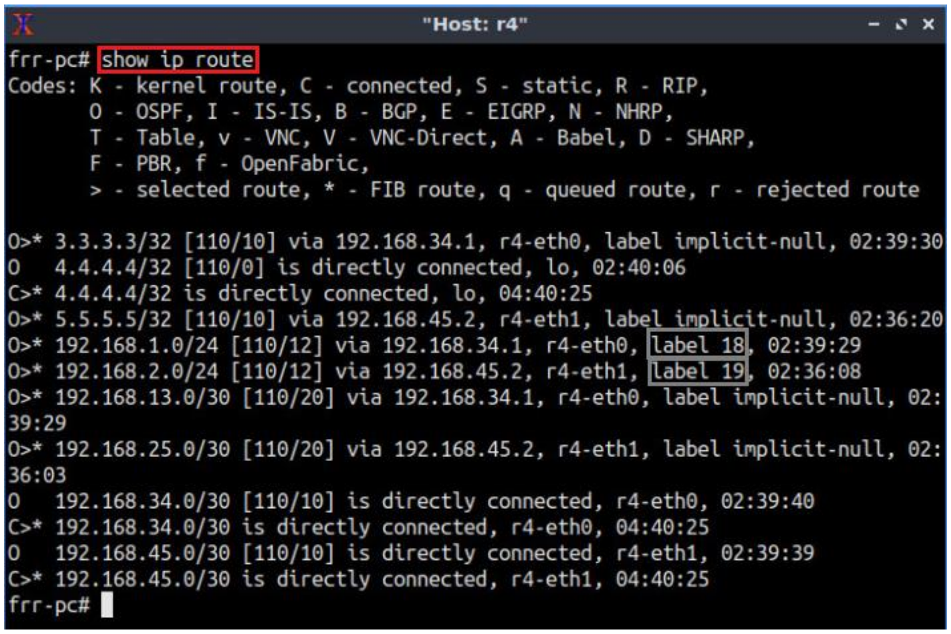
```
"Host: r5"
frr-pc# exit
root@frr-pc:/etc/routers/r5# ldpd
root@frr-pc:/etc/routers/r5# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# mpls ldp
frr-pc(config-ldp)# router-id 5.5.5.5
frr-pc(config-ldp)# address-family ipv4
frr-pc(config-ldp-af)# discovery transport-address 5.5.5.5
frr-pc(config-ldp-af)# interface lo
frr-pc(config-ldp-af-if)# exit
frr-pc(config-ldp-af)# interface r5-eth0
frr-pc(config-ldp-af-if)# end
frr-pc#
```

Figure 51. Configuring LDP in router r5.

You can get different labels for each network as LDP labels are assigned randomly.

Step 15. Type the following command to verify the routing table of router r4.

```
show ip route
```



```
"Host: r4"
frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

O>* 3.3.3.3/32 [110/10] via 192.168.34.1, r4-eth0, label implicit-null, 02:39:30
O   4.4.4.4/32 [110/0] is directly connected, lo, 02:40:06
C>* 4.4.4.4/32 is directly connected, lo, 04:40:25
O>* 5.5.5.5/32 [110/10] via 192.168.45.2, r4-eth1, label implicit-null, 02:36:20
O>* 192.168.1.0/24 [110/12] via 192.168.34.1, r4-eth0, label 18, 02:39:29
O>* 192.168.2.0/24 [110/12] via 192.168.45.2, r4-eth1, label 19, 02:36:08
O>* 192.168.13.0/30 [110/20] via 192.168.34.1, r4-eth0, label implicit-null, 02:
39:29
O>* 192.168.25.0/30 [110/20] via 192.168.45.2, r4-eth1, label implicit-null, 02:
36:03
O   192.168.34.0/30 [110/10] is directly connected, r4-eth0, 02:39:40
C>* 192.168.34.0/30 is directly connected, r4-eth0, 04:40:25
O   192.168.45.0/30 [110/10] is directly connected, r4-eth1, 02:39:39
C>* 192.168.45.0/30 is directly connected, r4-eth1, 04:40:25
frr-pc#
```

Figure 52. Verifying the routing table of router r4.

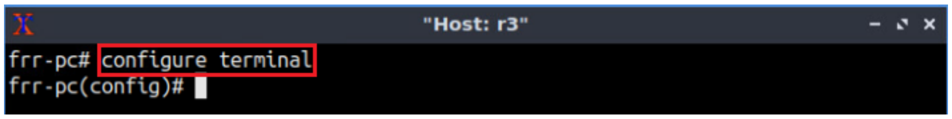
Consider the figure above. The figure shows thar router r4 will assign label 18 for the destination network 192.168.1.0/24. Router r1 will receive the packet and the label (18). The label will be removed, and the packet will be delivered to the destination router r1.

Similarly, for network 192.168.2.0/24, router r5 will receive label 19 which will be removed in order to send the IP packet to the destination router, r2.

You can get different labels for each network as LDP labels are assigned randomly.

Step 16. You will configure static MPLS in router r3 so that the router pops the incoming label (label 18) and forward the IP packet to the destination router, r1. Type the following command to enable router r3 configuration mode.

```
configure terminal
```

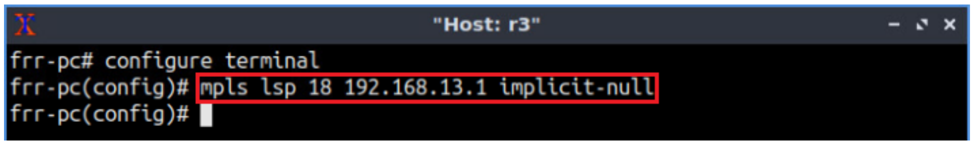

"Host: r3"

```
frr-pc# configure terminal
frr-pc(config)#
```

Figure 53. Enabling configuration mode on router r3.

Step 17. Type the following command to configure static MPLS in router r3. Router r3 will receive label 18 assigned by router r4. It will pop the label and the IP packet will be forwarded to the destination router r1 via 192.168.13.1.

```
mpls lsp 18 192.168.13.1 implicit-null
```


"Host: r3"

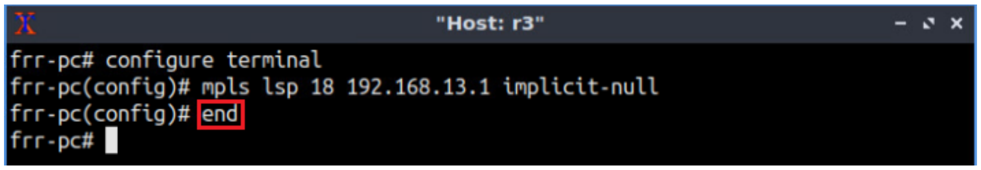
```
frr-pc# configure terminal
frr-pc(config)# mpls lsp 18 192.168.13.1 implicit-null
frr-pc(config)#
```

Figure 54. configuring static MPLS in router r3.

The keyword *implicit-null* indicates that the router will perform a pop and deliver the IP packet to the destination.

Step 18. Type the following command to exit from the configuration mode.

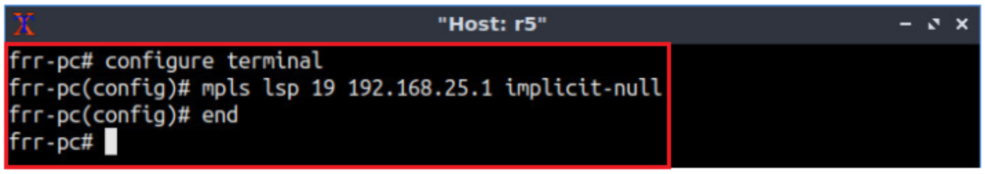
```
end
```


"Host: r3"

```
frr-pc# configure terminal
frr-pc(config)# mpls lsp 18 192.168.13.1 implicit-null
frr-pc(config)# end
frr-pc#
```

Figure 55. Exiting from configuration mode.

Step 19. You will configure router r3 similar to router r5 so that the router pops the incoming label 19 which was assigned by router r4 and forward the IP packet to the destination router, r2 via 192.168.25.1. All the steps are summarized in the following figure.


"Host: r5"

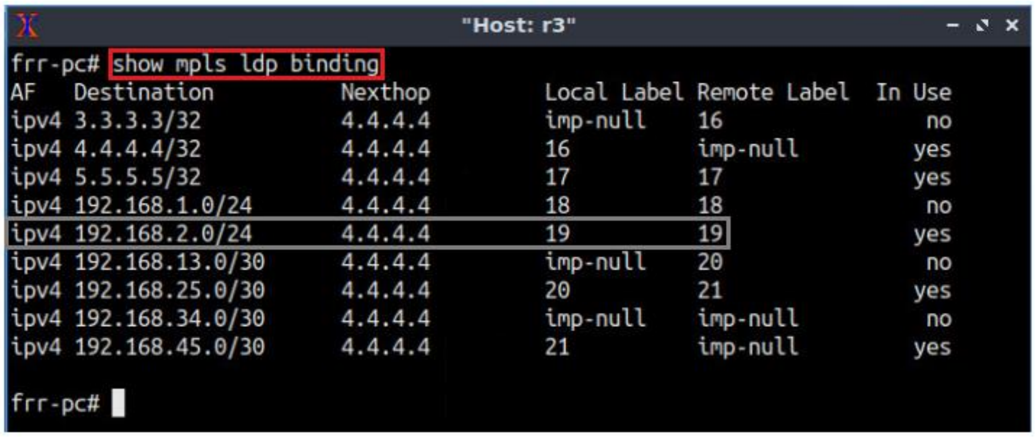
```
frr-pc# configure terminal
frr-pc(config)# mpls lsp 19 192.168.25.1 implicit-null
frr-pc(config)# end
frr-pc#
```

Figure 56. configuring static MPLS in router r5.

6 Verifying configuration

Step 1. Type the following command to verify the MPLS binding in router r3. You can verify that router r3 is using label 19 to reach network 192.168.2.0/24.

```
show mpls ldp binding
```

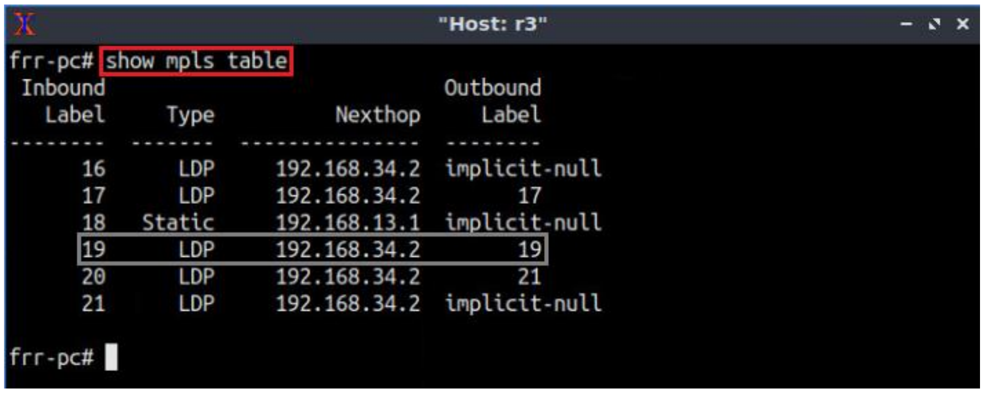


```
"Host: r3"
frr-pc# show mpls ldp binding
AF   Destination         Nexthop         Local Label Remote Label  In Use
ipv4 3.3.3.3/32          4.4.4.4         imp-null    16                no
ipv4 4.4.4.4/32          4.4.4.4         16          imp-null         yes
ipv4 5.5.5.5/32          4.4.4.4         17          17               yes
ipv4 192.168.1.0/24      4.4.4.4         18          18                no
ipv4 192.168.2.0/24      4.4.4.4         19          19               yes
ipv4 192.168.13.0/30     4.4.4.4         imp-null    20                no
ipv4 192.168.25.0/30     4.4.4.4         20          21               yes
ipv4 192.168.34.0/30     4.4.4.4         imp-null    imp-null          no
ipv4 192.168.45.0/30     4.4.4.4         21          imp-null         yes

frr-pc#
```

Figure 57. Verifying MPLS binding in router r3.

Step 2. Type the following command to verify MPLS table in router r3.

```
show mpls table
```



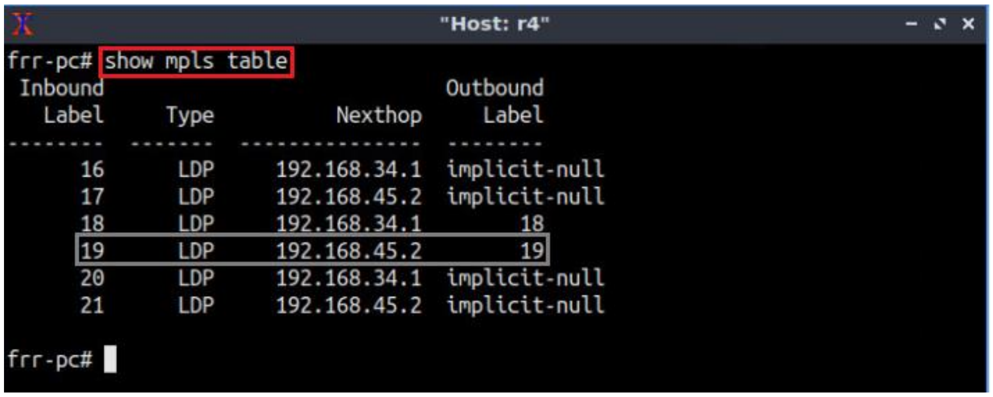
```
"Host: r3"
frr-pc# show mpls table
 Inbound                            Outbound
   Label     Type          Nexthop     Label
--------  -------  ---------------  --------
      16      LDP     192.168.34.2  implicit-null
      17      LDP     192.168.34.2        17
      18   Static     192.168.13.1  implicit-null
      19      LDP     192.168.34.2        19
      20      LDP     192.168.34.2        21
      21      LDP     192.168.34.2  implicit-null

frr-pc#
```

Figure 58. Verifying MPLS table in router r3.

You will notice all the Inbound and outbound labels. To reach network 192.168.2.0/24, router r3 will assign label 19 (Inbound) and the IP packet will be forwarded to router r4 (outbound) through 192.168.34.2.

Step 3. Type the following command to verify the MPLS table in router r4.

```
show mpls table
```



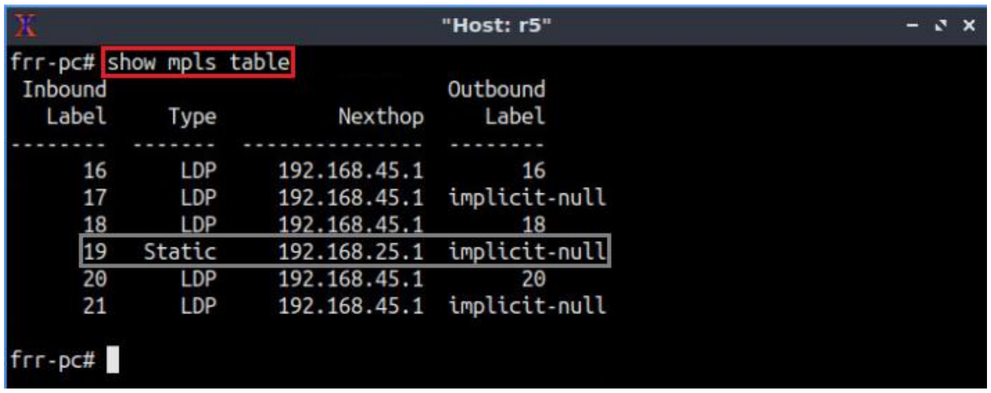
```
"Host: r4"
frr-pc# show mpls table
 Inbound                                  Outbound
   Label     Type            Nexthop        Label
 --------  -------  ---------------  --------
      16       LDP     192.168.34.1  implicit-null
      17       LDP     192.168.45.2  implicit-null
      18       LDP     192.168.34.1           18
      19       LDP     192.168.45.2           19
      20       LDP     192.168.34.1  implicit-null
      21       LDP     192.168.45.2  implicit-null

frr-pc#
```

Figure 59. Verifying MPLS table in router r4.

Consider the figure above. Router r4 will receive label 19 (inbound) and the packet will be forwarded to router r5 using label 19 (outbound) through 192.168.45.2.

Step 4. Type the following command to verify the MPLS table in router r5.

```
show mpls table
```



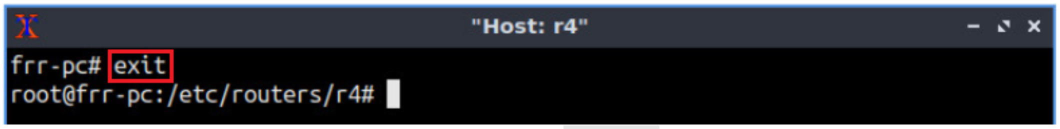
```
"Host: r5"
frr-pc# show mpls table
 Inbound                                  Outbound
   Label     Type            Nexthop        Label
 --------  -------  ---------------  --------
      16       LDP     192.168.45.1           16
      17       LDP     192.168.45.1  implicit-null
      18       LDP     192.168.45.1           18
      19    Static     192.168.25.1  implicit-null
      20       LDP     192.168.45.1           20
      21       LDP     192.168.45.1  implicit-null

frr-pc#
```

Figure 60. Verifying MPLS table in router r5.

Consider the figure above. Router r5 will receive label 19 (inbound), pop the label (outbound) and forward the IP packet to the destination router, r2 through 192.168.25.1.

Step 5. In router r4's terminal, type the following command to exit the vtysh session:

```
exit
```

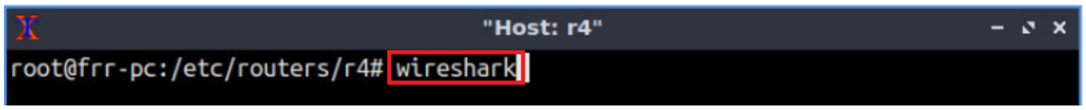


```
"Host: r4"
frr-pc# exit
root@frr-pc:/etc/routers/r4#
```

Figure 61. Exiting the `vtysh` session.

Step 6. Type the following command to start Wireshark packet analyzer. A new window will emerge.

```
wireshark
```



```
root@frr-pc:/etc/routers/r4# wireshark
```

Figure 62. Starting Wireshark packet analyzer.

Step 7. Select interface *r4-eth0* and click on the icon located on the upper left-hand side to start capturing packets.

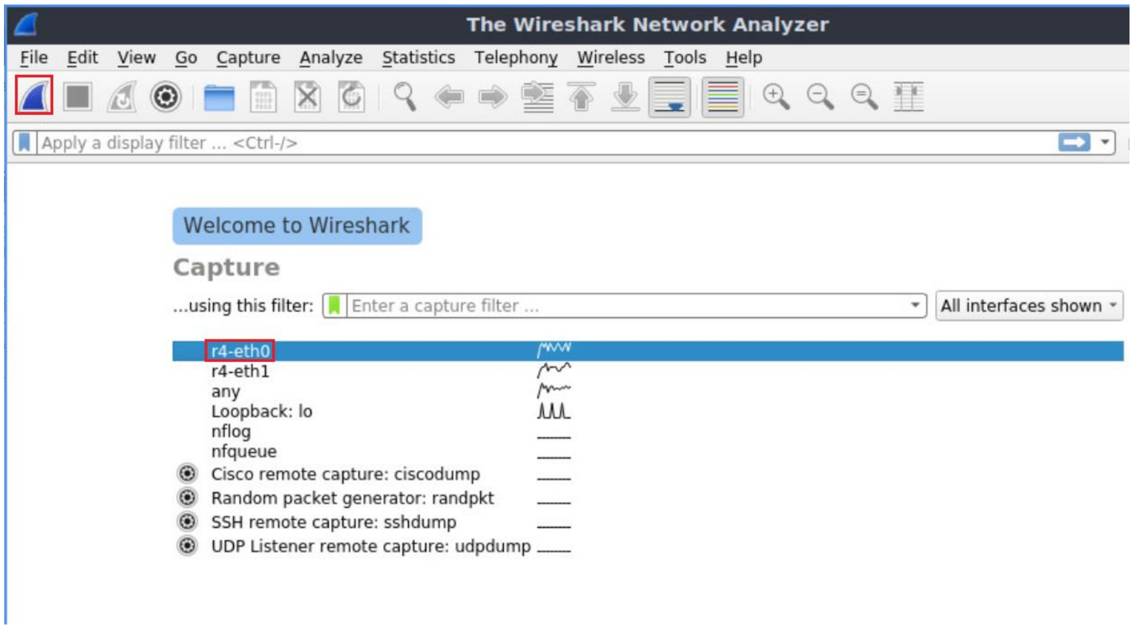
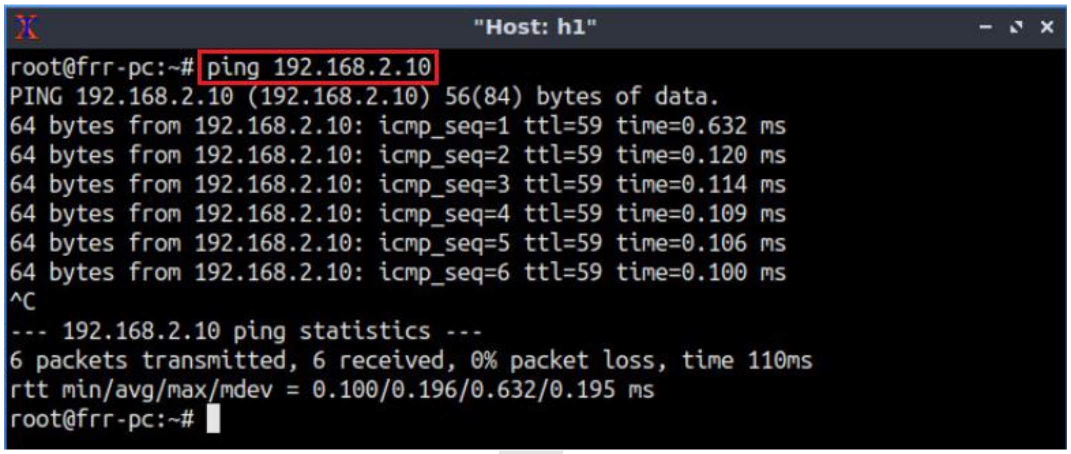

The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Capture
...using this filter:
Enter a capture filter ...
All interfaces shown
r4-eth0
r4-eth1
any
Loopback: lo
nflog
nfqueue
Cisco remote capture: ciscodump
Random packet generator: randpkt
SSH remote capture: sshdump
UDP Listener remote capture: udpdump

Figure 63. Starting packet capture.

Step 8. Test the connectivity between host h1 and host h2 using the `ping` command. On host h1, type the command specified below. To stop the test, press `Ctrl+c`. The figure below shows a successful connectivity test.

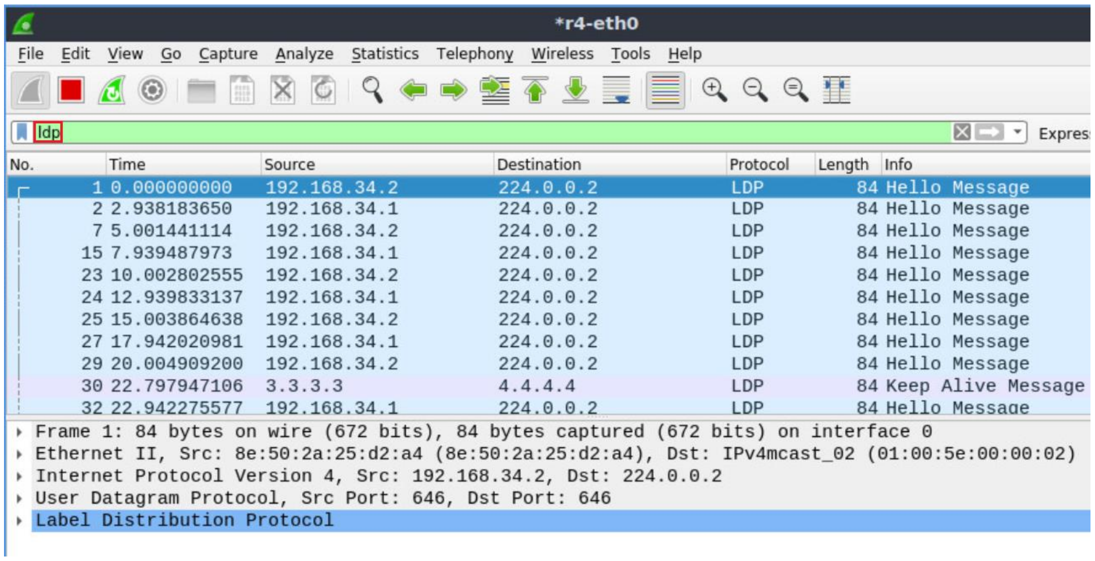
```
ping 192.168.2.10
```



```
root@frr-pc:~# ping 192.168.2.10
PING 192.168.2.10 (192.168.2.10) 56(84) bytes of data.
64 bytes from 192.168.2.10: icmp_seq=1 ttl=59 time=0.632 ms
64 bytes from 192.168.2.10: icmp_seq=2 ttl=59 time=0.120 ms
64 bytes from 192.168.2.10: icmp_seq=3 ttl=59 time=0.114 ms
64 bytes from 192.168.2.10: icmp_seq=4 ttl=59 time=0.109 ms
64 bytes from 192.168.2.10: icmp_seq=5 ttl=59 time=0.106 ms
64 bytes from 192.168.2.10: icmp_seq=6 ttl=59 time=0.100 ms
^C
--- 192.168.2.10 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 110ms
rtt min/avg/max/mdev = 0.100/0.196/0.632/0.195 ms
root@frr-pc:~#
```

Figure 64. Output of `ping` command in host h1.

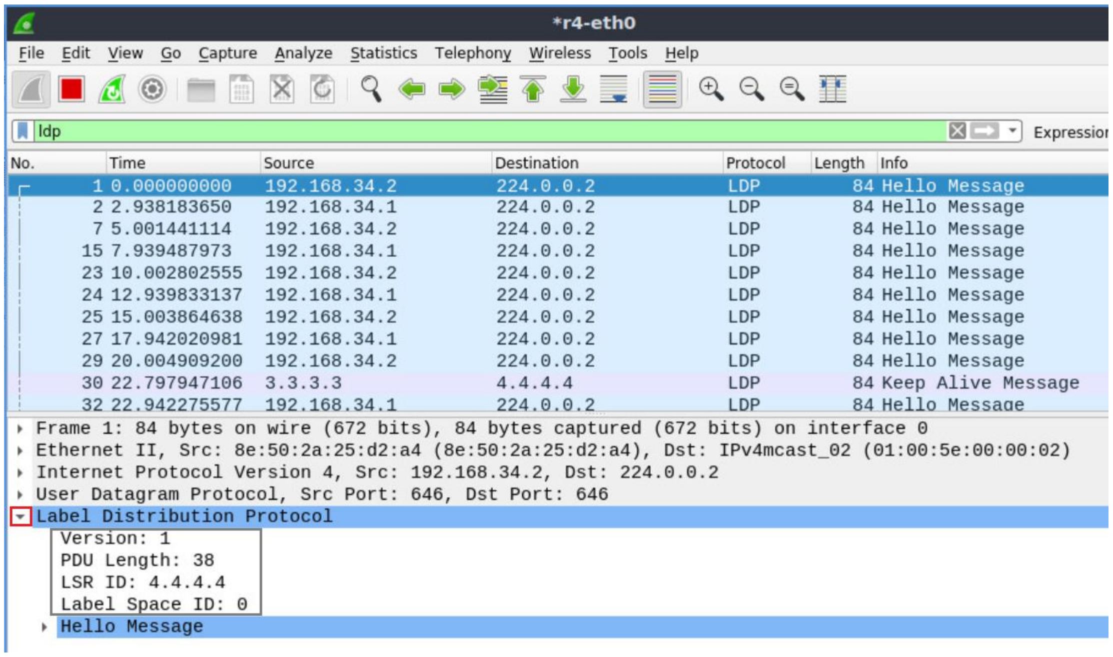
Step 9. In the filter box located on the upper left-hand side, type *ldp* in order to filter packets.


*r4-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ldp
Expres

```
No.   Time           Source          Destination   Protocol  Length Info
    1 0.000000000    192.168.34.2    224.0.0.2     LDP       84 Hello Message
    2 2.938183650    192.168.34.1    224.0.0.2     LDP       84 Hello Message
    7 5.001441114    192.168.34.2    224.0.0.2     LDP       84 Hello Message
   15 7.939487973    192.168.34.1    224.0.0.2     LDP       84 Hello Message
   23 10.002802555   192.168.34.2    224.0.0.2     LDP       84 Hello Message
   24 12.939833137   192.168.34.1    224.0.0.2     LDP       84 Hello Message
   25 15.003864638   192.168.34.2    224.0.0.2     LDP       84 Hello Message
   27 17.942020981   192.168.34.1    224.0.0.2     LDP       84 Hello Message
   29 20.004909200   192.168.34.2    224.0.0.2     LDP       84 Hello Message
   30 22.797947106   3.3.3.3         4.4.4.4       LDP       84 Keep Alive Message
   32 22.942275577   192.168.34.1    224.0.0.2     LDP       84 Hello Message
▸ Frame 1: 84 bytes on wire (672 bits), 84 bytes captured (672 bits) on interface 0
▸ Ethernet II, Src: 8e:50:2a:25:d2:a4 (8e:50:2a:25:d2:a4), Dst: IPv4mcast_02 (01:00:5e:00:00:02)
▸ Internet Protocol Version 4, Src: 192.168.34.2, Dst: 224.0.0.2
▸ User Datagram Protocol, Src Port: 646, Dst Port: 646
▸ Label Distribution Protocol
```

Figure 65. Filtering network traffic.

Step 10. Click on the arrow located on the leftmost side of the field called *Label Distribution Protocol.* A list will be displayed.

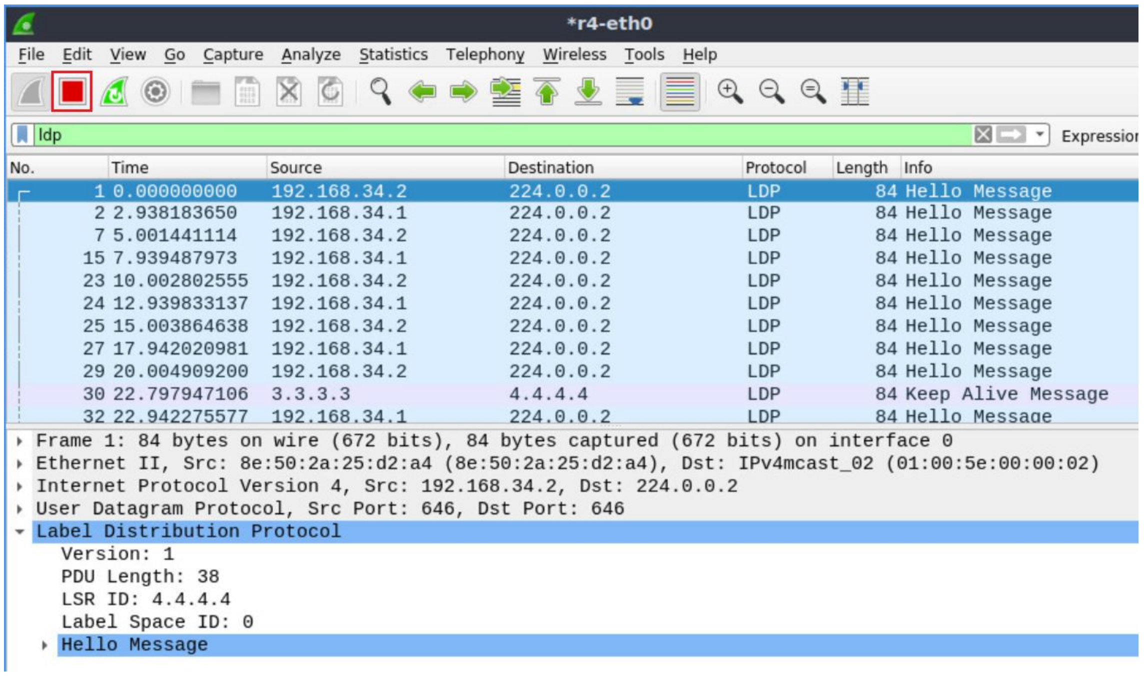

*r4-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ldp
Expression

```
No.   Time           Source          Destination   Protocol  Length Info
    1 0.000000000    192.168.34.2    224.0.0.2     LDP       84 Hello Message
    2 2.938183650    192.168.34.1    224.0.0.2     LDP       84 Hello Message
    7 5.001441114    192.168.34.2    224.0.0.2     LDP       84 Hello Message
   15 7.939487973    192.168.34.1    224.0.0.2     LDP       84 Hello Message
   23 10.002802555   192.168.34.2    224.0.0.2     LDP       84 Hello Message
   24 12.939833137   192.168.34.1    224.0.0.2     LDP       84 Hello Message
   25 15.003864638   192.168.34.2    224.0.0.2     LDP       84 Hello Message
   27 17.942020981   192.168.34.1    224.0.0.2     LDP       84 Hello Message
   29 20.004909200   192.168.34.2    224.0.0.2     LDP       84 Hello Message
   30 22.797947106   3.3.3.3         4.4.4.4       LDP       84 Keep Alive Message
   32 22.942275577   192.168.34.1    224.0.0.2     LDP       84 Hello Message
▸ Frame 1: 84 bytes on wire (672 bits), 84 bytes captured (672 bits) on interface 0
▸ Ethernet II, Src: 8e:50:2a:25:d2:a4 (8e:50:2a:25:d2:a4), Dst: IPv4mcast_02 (01:00:5e:00:00:02)
▸ Internet Protocol Version 4, Src: 192.168.34.2, Dst: 224.0.0.2
▸ User Datagram Protocol, Src Port: 646, Dst Port: 646
▾ Label Distribution Protocol
    Version: 1
    PDU Length: 38
    LSR ID: 4.4.4.4
    Label Space ID: 0
  ▸ Hello Message
```

Figure 66. Verifying LDP packets.

Consider the figure above. The particular field contains all the information about LDP. You will notice the LSR ID is 4.4.4.4 (router 4).

Step 11. To stop packet capturing, click on the red button located on the upper left-hand side and close Wireshark.


*r4-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ldp
Expression

```
No.  Time          Source          Destination  Protocol  Length Info
   1 0.000000000   192.168.34.2    224.0.0.2    LDP       84 Hello Message
   2 2.938183650   192.168.34.1    224.0.0.2    LDP       84 Hello Message
   7 5.001441114   192.168.34.2    224.0.0.2    LDP       84 Hello Message
  15 7.939487973   192.168.34.1    224.0.0.2    LDP       84 Hello Message
  23 10.002802555  192.168.34.2    224.0.0.2    LDP       84 Hello Message
  24 12.939833137  192.168.34.1    224.0.0.2    LDP       84 Hello Message
  25 15.003864638  192.168.34.2    224.0.0.2    LDP       84 Hello Message
  27 17.942020981  192.168.34.1    224.0.0.2    LDP       84 Hello Message
  29 20.004909200  192.168.34.2    224.0.0.2    LDP       84 Hello Message
  30 22.797947106  3.3.3.3         4.4.4.4      LDP       84 Keep Alive Message
  32 22.942275577  192.168.34.1    224.0.0.2    LDP       84 Hello Message
```

```
▸ Frame 1: 84 bytes on wire (672 bits), 84 bytes captured (672 bits) on interface 0
▸ Ethernet II, Src: 8e:50:2a:25:d2:a4 (8e:50:2a:25:d2:a4), Dst: IPv4mcast_02 (01:00:5e:00:00:02)
▸ Internet Protocol Version 4, Src: 192.168.34.2, Dst: 224.0.0.2
▸ User Datagram Protocol, Src Port: 646, Dst Port: 646
▾ Label Distribution Protocol
    Version: 1
    PDU Length: 38
    LSR ID: 4.4.4.4
    Label Space ID: 0
  ▸ Hello Message
```

Figure 67. Stopping packet capture.

This concludes Lab 5. Stop the emulation and then exit out of MiniEdit.

References

1. Lydia Parziale, David T. Britt, Chuck Davis, Jason Forrester, Wei Liu, Carolyn Matthews, Nicolas Rosselot, "*TCP/IP tutorial and technical overview*", 8th Edition, Pearson, 2006.
2. Luc De Ghein, "*MPLS fundamentals*", 1st Edition, Pearson. 2006.
3. Juniper networks, "*MPLS label distribution protocols overview*", 2018. [Online]. Available: https://www.juniper.net/documentation/en_US/junose15.1/topics/concept/mpls-label-distribution-protocols.html
4. Linux foundation collaborative projects, "*FRR routing documentation*", 2017. [Online]. Available: http://docs.frrouting.org/en/latest/zebra.html#mpls-commands
5. Juniper, "*Understanding MPLS label operations*", 2020. [Online]. Available: https://www.juniper.net/documentation/en_US/junos/topics/concept/mpls-label-operations-qfx-series.html

MPLS AND ADVANCED BGP TOPICS

Lab 6: Virtual Routing and Forwarding (VRF)

Document Version: 03-04-2021

Award 1829698
"CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers"

Contents

Overview

The lab discusses the concept of Virtual Routing and Forwarding (VRF) that allows to create multiple routing tables in a single router. This increases functionality by allowing network paths to be segmented without using multiple devices. This lab aims to configure and verify VRF in a single router to generate segregated traffic.

Objectives

By the end of this lab, students should be able to:

1. Explain the concept of VRF.
2. Configure static route on routers.
3. Configure VRF in routers.
4. Verify VRF configuration to ensure the existence of multiple routing instances in a single router.

Lab settings

The information in Table 1 provides the credentials to access Client machine.

Table 1. Credentials to access Client machine.

Device	Account	Password
Client	admin	password

Lab roadmap

This lab is organized as follows:

1. Section 1: Introduction.
2. Section 2: Lab topology.
3. Section 3: Verify connectivity among all routers.
4. Section 4: Configure VRF on ISP router.
5. Section 5: Configure static routes on VRFs.
6. Section 6: Verify VRF configuration.

1 Introduction

1.1 Static routing

Static routing is a routing mechanism that is configured manually, rather than information from a dynamic routing protocol. Static routing allows a router to learn about a route to a remote network[1]. The administrator is responsible for discovering and propagating routes through the network. Unlike dynamic routing, it requires manual configuration if there is any changes in the network. It uses less bandwidth compared to dynamic routing protocols and provides ease of routing table maintenance in smaller networks. It can be used to define an exit point from a router when no other routes are available or necessary which is called a default route.

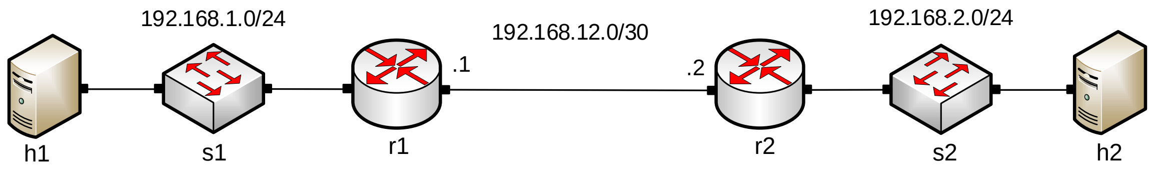


Figure 1. Static routing.

Consider Figure 1. Router r1 uses static route to reach host h2 (192.168.2.0/24). To configure a static route, Router r1 will specify the next-hop address (192.168.12.2) to reach the destination network, 192.168.2.0/24. If host h1 wants to send any data packet to host h2, the packet will be received by router r2 through the next-hop, 192.168.12.2. Finally, router r2 will deliver the packet to the destination host, h2.

1.2 Introduction to VRF

VRF is a technology that allows multiple routing tables to exist in a router and work simultaneously[2]. The concept of VRF is similar to Virtual Local Area Network (VLAN). Since the routing instances are separated for each VRF, overlapping IP addresses can be used. An interface cannot belong to more than one VRF at any time. VRFs act like a logical router and requires a forwarding table that decides how to forward the traffic. It prevents traffic from being forwarded from the outside of a particular VRF.

VRF is widely used in Layer 3 MPLS VPN. The Provider Edge (PE) router will maintain separate and distinct routing tables for each customer. CE refers to Customer Edge router. Each PE builds these unique routing tables with their own routing table mechanisms for each customer that is connected to the PE. This unique separation of routing tables allows routers to store routes and forward packets even if the customers are using identical addressing.

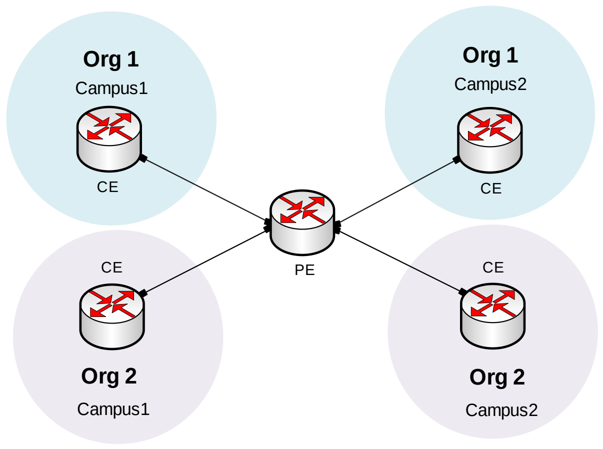


Figure 2. Virtual Routing and Forwarding (VRF).

Consider figure 2. Four CE routers are connected to PE router. Each organization (*org1* and *org2*) has two sites (campus1 and campus2). Two different VRFs (*org1* and *org2*) are running on PE router in order to create different routing instance for each organization.

2 Lab topology

Consider Figure 3. Two organizations (*org1* and *org2*) have two sites (campus 1 and campus 2) and those are connected to ISP. All the campus routers can communicate with each other. Two different VRFs will be created in ISP router so that both the organization have different routing table and cannot communicate with each other.

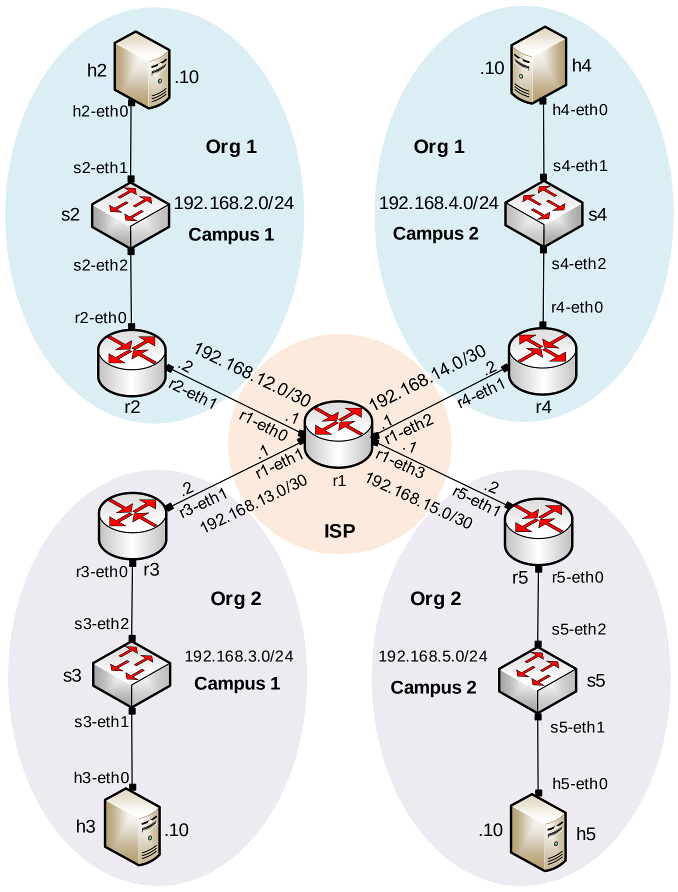


Figure 3. Lab topology.

2.1 Lab settings

Routers and hosts are already configured according to the IP addresses shown in Table 2.

Table 2. Topology information.

Device	Interface	IPV4 Address	Subnet	Default gateway
r1	r1-eth0	192.168.12.1	/30	N/A
	r1-eth1	192.168.13.1	/30	N/A
	r1-eth1	192.168.14.1	/30	N/A
	r1-eth1	192.168.15.1	/30	N/A
r2	r2-eth0	192.168.2.1	/24	N/A
	r2-eth1	192.168.12.2	/30	N/A
r3	r3-eth0	192.168.3.1	/24	N/A
	r3-eth1	192.168.13.2	/30	N/A
	r4-eth0	192.168.4.1	/24	N/A

r4	r4-eth1	192.168.14.2	/30	N/A
r5	r5-eth0	192.168.5.1	/24	N/A
	r5-eth1	192.168.15.2	/30	N/A
h2	h2-eth0	192.168.2.10	/24	192.168.2.1
h3	h3-eth0	192.168.3.10	/24	192.168.3.1
h4	h4-eth0	192.168.4.10	/24	192.168.4.1
h5	h5-eth0	192.168.5.10	/24	192.168.5.1

2.2 Open topology and load the configuration

Step 1. Start by launching MiniEdit by clicking on desktop's shortcut. When prompted for a password, type `password`.

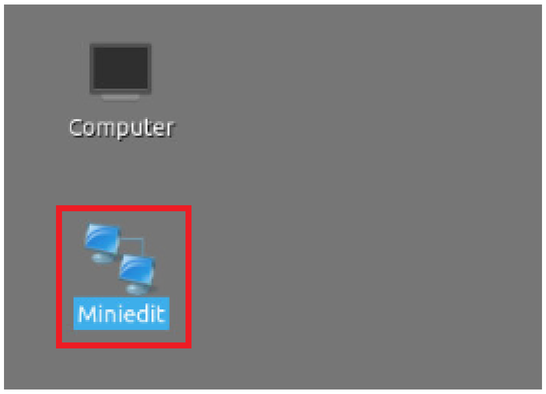


Figure 4. MiniEdit shortcut.

Step 2. On MiniEdit's menu bar, click on *File* then *open* to load the lab's topology. Locate the *lab6.mn* topology file in the default directory, */home/frr/MPLS_advanced_BGP/lab6* and click on *Open*.

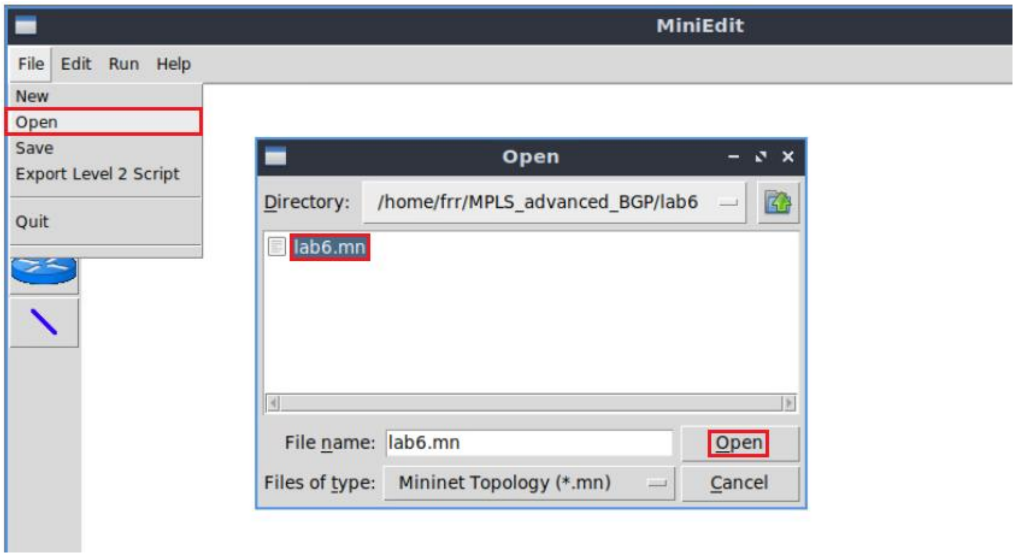


Figure 5. MiniEdit's Open dialog.

At this point the topology is loaded with all the required network components. You will execute a script that will load the configuration of the routers.

Step 3. Open the Linux terminal.

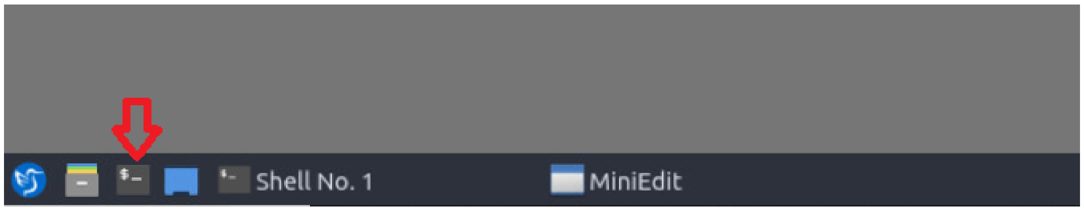
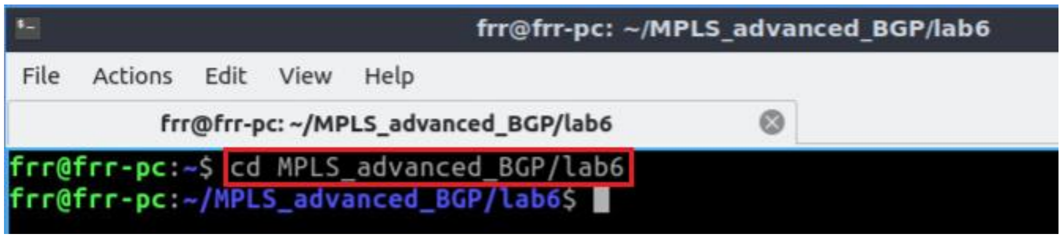


Figure 6. Opening Linux terminal.

Step 4. Click on the Linux's terminal and navigate into *MPLS_advanced_BGP/lab6* directory by issuing the following command. This folder contains a configuration file and the script responsible for loading the configuration. The configuration file will assign IP addresses to the routers' interfaces. The `cd` command is short for change directory followed by an argument that specifies the destination directory.

```
cd MPLS_advanced_BGP/lab6
```

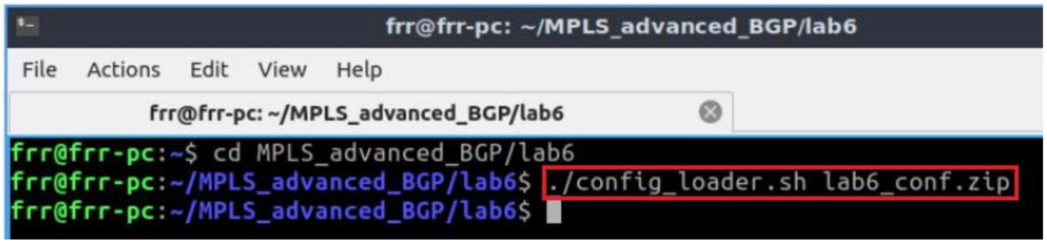

frr@frr-pc: ~/MPLS_advanced_BGP/lab6
File
Actions
Edit
View
Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab6

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab6
frr@frr-pc:~/MPLS_advanced_BGP/lab6$
```

Figure 7. Entering to the *MPLS_advanced_BGP/lab6* directory.

Step 5. To execute the shell script, type the following command. The argument of the program corresponds to the configuration zip file that will be loaded in all the routers in the topology.

```
./config_loader.sh lab6_conf.zip
```

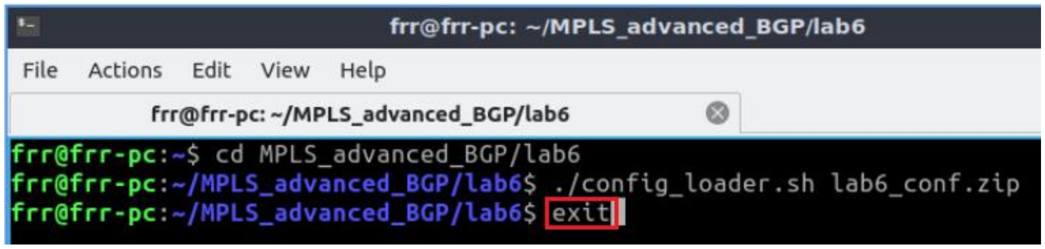

frr@frr-pc: ~/MPLS_advanced_BGP/lab6
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab6

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab6
frr@frr-pc:~/MPLS_advanced_BGP/lab6$ ./config_loader.sh lab6_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab6$
```

Figure 8. Executing the shell script to load the configuration.

Step 6. Type the following command to exit the Linux terminal.

```
exit
```


frr@frr-pc: ~/MPLS_advanced_BGP/lab6
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab6

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab6
frr@frr-pc:~/MPLS_advanced_BGP/lab6$ ./config_loader.sh lab6_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab6$ exit
```

Figure 9. Exiting from the terminal.

Step 7. At this point hosts h2, h3, h4 and h5 interfaces are configured. To proceed with the emulation, click on the *Run* button located in lower left-hand side.

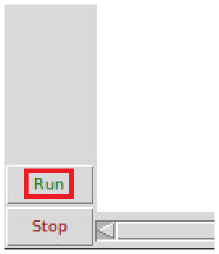


Figure 10. Starting the emulation.

Step 8. Click on Mininet's terminal, i.e., the one launched when MiniEdit was started.

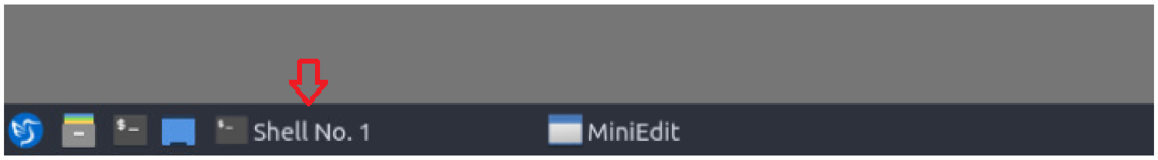
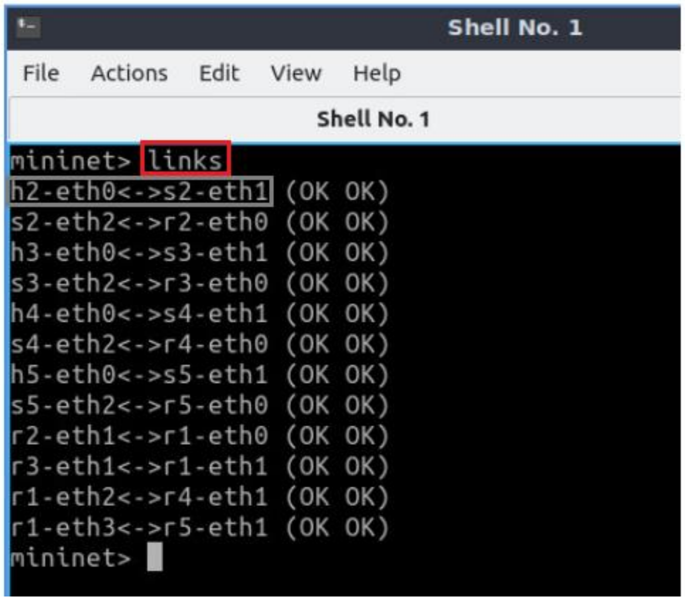


Figure 11. Opening Mininet's terminal.

Step 9. Issue the following command to display the interface names and connections.

```
links
```



```
mininet> links
h2-eth0<->s2-eth1 (OK OK)
s2-eth2<->r2-eth0 (OK OK)
h3-eth0<->s3-eth1 (OK OK)
s3-eth2<->r3-eth0 (OK OK)
h4-eth0<->s4-eth1 (OK OK)
s4-eth2<->r4-eth0 (OK OK)
h5-eth0<->s5-eth1 (OK OK)
s5-eth2<->r5-eth0 (OK OK)
r2-eth1<->r1-eth0 (OK OK)
r3-eth1<->r1-eth1 (OK OK)
r1-eth2<->r4-eth1 (OK OK)
r1-eth3<->r5-eth1 (OK OK)
mininet> 
```

Figure 12. Displaying network interfaces.

In Figure 12, the link displayed within the gray box indicates that interface eth1 of router r1 connects to interface eth1 of switch s1 (i.e., *h2-eth0<->s2-eth1*).

3 Load required daemons and verify connectivity among routers

Static routes are configured in ISP router to get connectivity to all campus routers. Default route is running in campus routers (r2, r3, r4 and r5) so that all the campus routers can communicate through router r1. In this section, you will load required daemons and verify the connectivity between two organizations.

Step 1. Hold right-click on host h2 and select *Terminal*. This opens the terminal of host h2 and allows the execution of commands on that host.

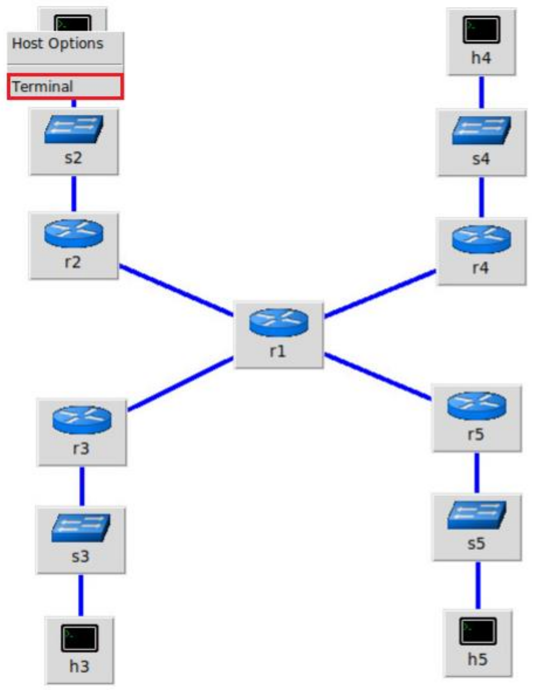
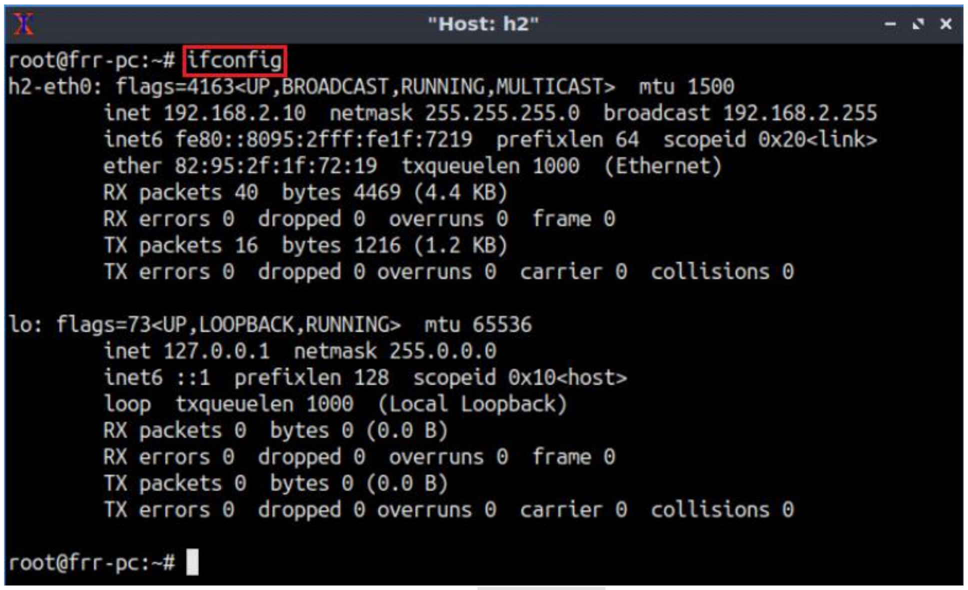


Figure 13. Opening a terminal on host h2.

Step 2. In host h2 terminal, type the command shown below to verify that the IP address was assigned successfully. You will verify that host h2 has interface *h2-eth0* configured with IP address 192.168.2.10 and subnet mask 255.255.255.0.

```
ifconfig
```

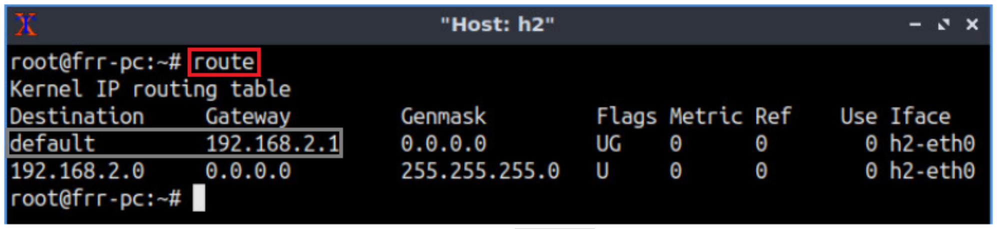


```
root@frr-pc:~# ifconfig
h2-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.10  netmask 255.255.255.0  broadcast 192.168.2.255
        inet6 fe80::8095:2fff:fe1f:7219  prefixlen 64  scopeid 0x20<link>
        ether 82:95:2f:1f:72:19  txqueuelen 1000  (Ethernet)
        RX packets 40  bytes 4469 (4.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 16  bytes 1216 (1.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@frr-pc:~# 
```

Figure 14. Output of `ifconfig` command.

Step 3. In host h2 terminal, type the command shown below to verify that the default gateway IP address is 192.168.2.1.

```
route
```



```
root@frr-pc:~# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         192.168.2.1     0.0.0.0         UG    0      0        0 h2-eth0
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 h2-eth0
root@frr-pc:~#
```

Figure 15. Output of `route` command.

Step 4. In order to verify host h3, h4 and h5, proceed similarly by repeating from step 1 to step 3 on host terminals. Similar results should be observed.

Step 5. In order to open router r1 terminal, hold right-click on router r1 and select Terminal.

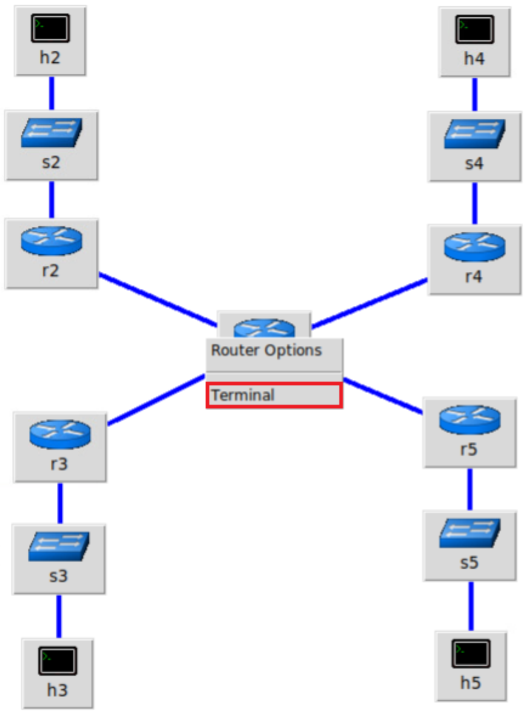
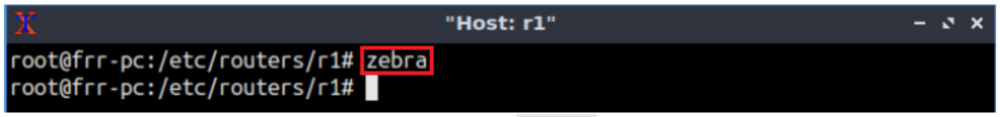


Figure 16. Opening a terminal on router r1.

Step 6. On router r1 terminal, you will start zebra daemon, which is a multi-server routing software that provides TCP/IP based routing protocols. The configuration will not be working if you do not enable zebra daemon initially. In order to start the zebra, type the following command:

```
zebra
```

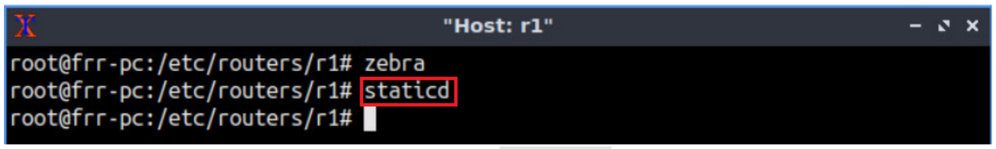


```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1#
```

Figure 17. Starting `zebra` daemon.

Step 7. Type the following command in router r1 terminal to enable *staticd* daemon.

```
staticd
```

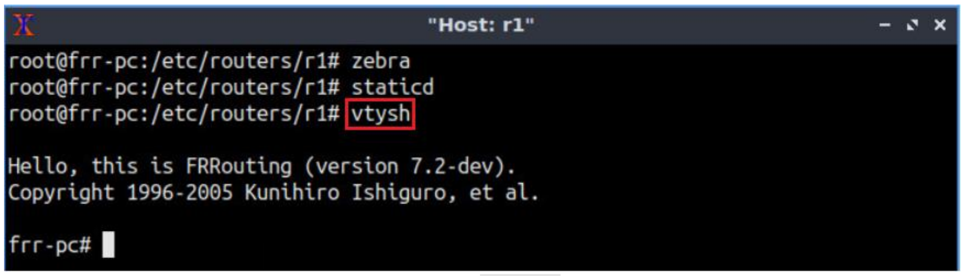


```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# staticd
root@frr-pc:/etc/routers/r1#
```

Figure 18. Starting `staticd` daemon.

Step 8. After initializing zebra, vtysh should be started in order to provide all the CLI commands defined by the daemons. To proceed, issue the following command:

```
vtysh
```

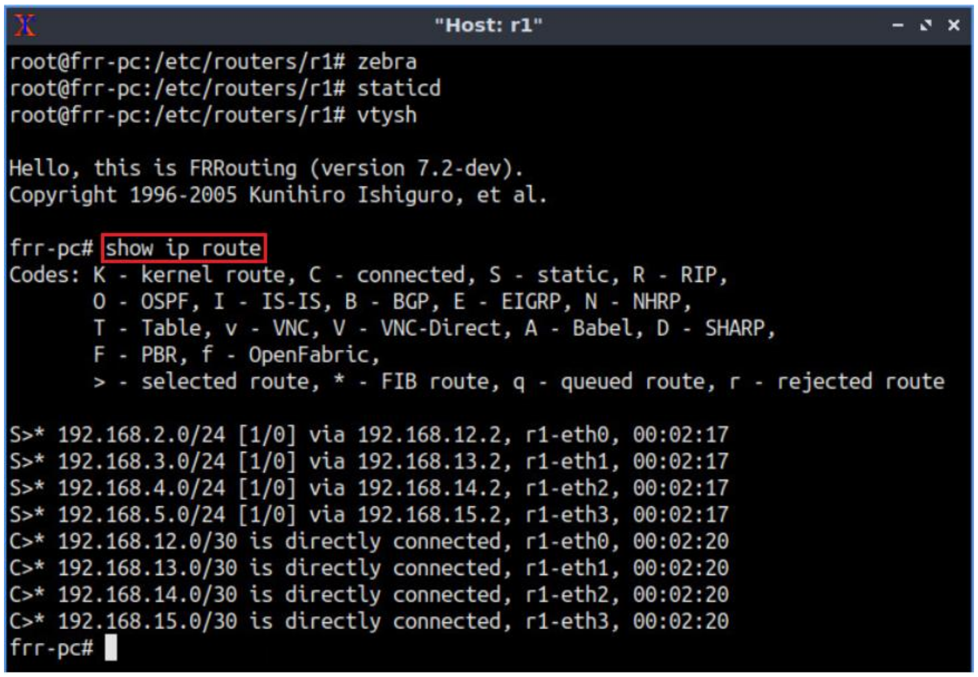


```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# staticd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 19. Starting `vtysh` in router r1.

Step 9. Type the following command in router r1 terminal to verify the routing table of router r1.

```
show ip route
```

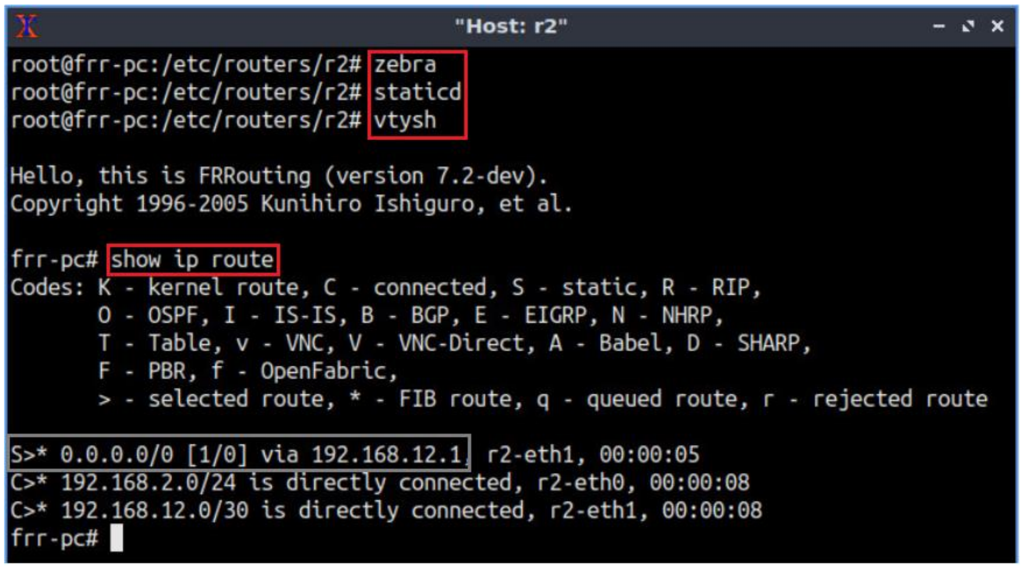


```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# staticd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

S>* 192.168.2.0/24 [1/0] via 192.168.12.2, r1-eth0, 00:02:17
S>* 192.168.3.0/24 [1/0] via 192.168.13.2, r1-eth1, 00:02:17
S>* 192.168.4.0/24 [1/0] via 192.168.14.2, r1-eth2, 00:02:17
S>* 192.168.5.0/24 [1/0] via 192.168.15.2, r1-eth3, 00:02:17
C>* 192.168.12.0/30 is directly connected, r1-eth0, 00:02:20
C>* 192.168.13.0/30 is directly connected, r1-eth1, 00:02:20
C>* 192.168.14.0/30 is directly connected, r1-eth2, 00:02:20
C>* 192.168.15.0/30 is directly connected, r1-eth3, 00:02:20
frr-pc#
```

Figure 20. Displaying routing table of router r1.

Consider the figure above. It shows all the directly connected networks and the networks learned through static routing. Router r1 has connectivity to all other routers at this time.

Step 10. Follow step 5 to step 9 to verify the routing table in router r2. All the steps are summarized in the following figure. To proceed, in router r2 terminal, issue the commands depicted below.

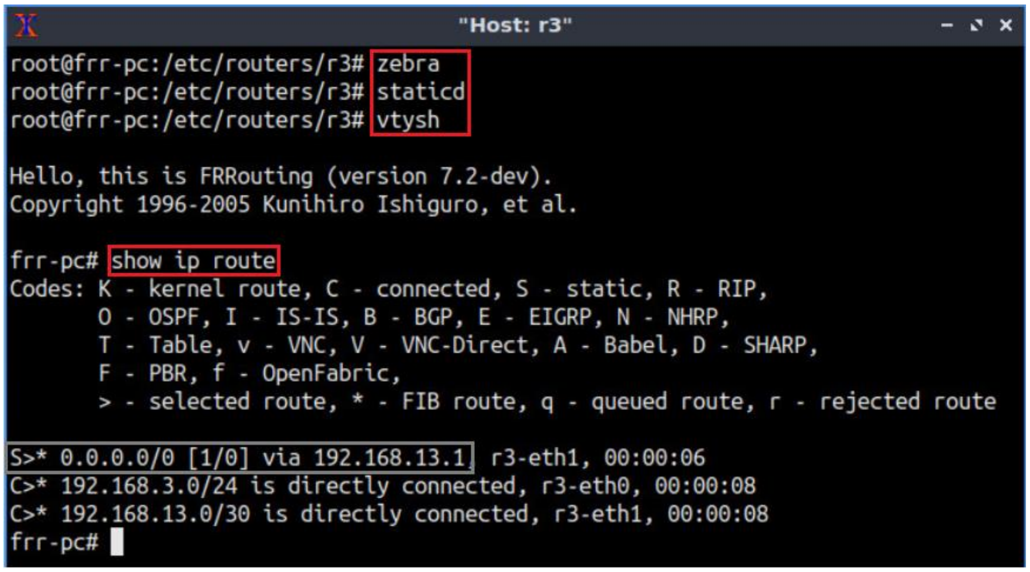


```
root@frr-pc:/etc/routers/r2# zebra
root@frr-pc:/etc/routers/r2# staticd
root@frr-pc:/etc/routers/r2# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

S>* 0.0.0.0/0 [1/0] via 192.168.12.1, r2-eth1, 00:00:05
C>* 192.168.2.0/24 is directly connected, r2-eth0, 00:00:08
C>* 192.168.12.0/30 is directly connected, r2-eth1, 00:00:08
frr-pc#
```

Figure 21. Displaying routing table of router r2.

Consider the figure above. By configuring default route, the router can communicate with other routers via 192.168.12.1.

Step 11. Follow step 5 to step 9 to verify the routing table in router r3. All the steps are summarized in the following figure. To proceed, in router r3 terminal, issue the commands depicted below.

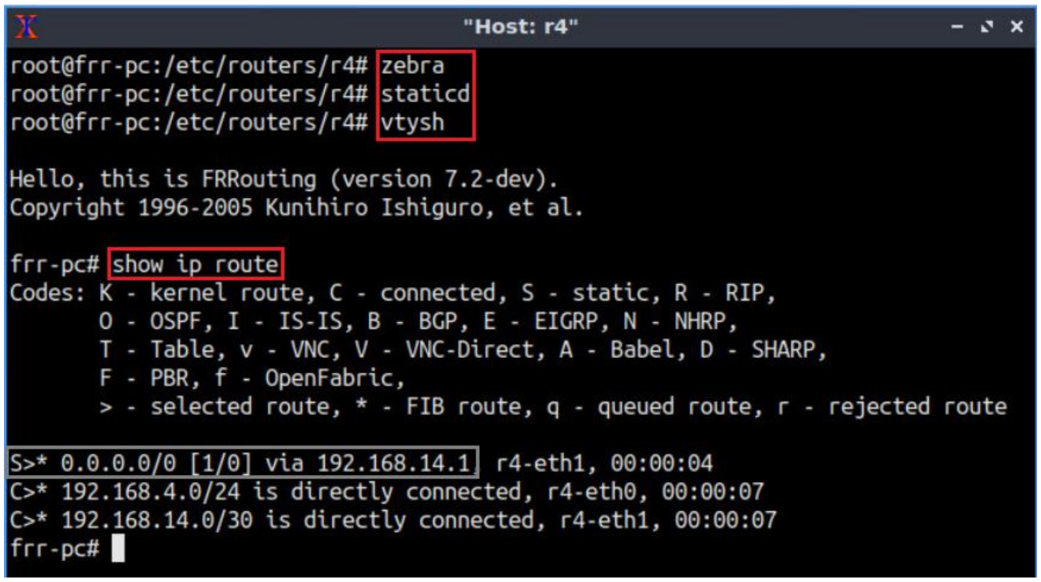


```
root@frr-pc:/etc/routers/r3# zebra
root@frr-pc:/etc/routers/r3# staticd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

S>* 0.0.0.0/0 [1/0] via 192.168.13.1, r3-eth1, 00:00:06
C>* 192.168.3.0/24 is directly connected, r3-eth0, 00:00:08
C>* 192.168.13.0/30 is directly connected, r3-eth1, 00:00:08
frr-pc#
```

Figure 22. Displaying routing table of router r3.

Consider the figure above. By configuring default route, the router can communicate with other routers via 192.168.13.1.

Step 12. Follow step 5 to step 9 to verify the routing table in router r4. All the steps are summarized in the following figure. To proceed, in router r4 terminal, issue the commands depicted below.

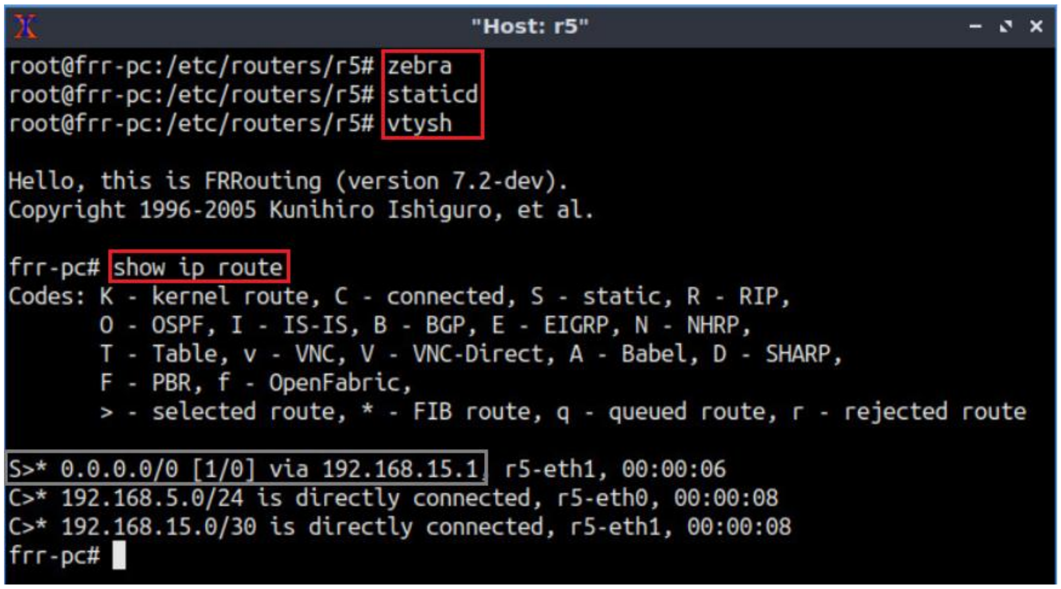


```
"Host: r4"
root@frr-pc:/etc/routers/r4# zebra
root@frr-pc:/etc/routers/r4# staticd
root@frr-pc:/etc/routers/r4# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

S>* 0.0.0.0/0 [1/0] via 192.168.14.1, r4-eth1, 00:00:04
C>* 192.168.4.0/24 is directly connected, r4-eth0, 00:00:07
C>* 192.168.14.0/30 is directly connected, r4-eth1, 00:00:07
frr-pc#
```

Figure 23. Displaying routing table of router r4.

Consider the figure above. By configuring default route, the router can communicate with other routers via 192.168.12.1.

Step 13. Follow step 5 to step 9 to verify the routing table in router r5. All the steps are summarized in the following figure. To proceed, in router r5 terminal, issue the commands depicted below.



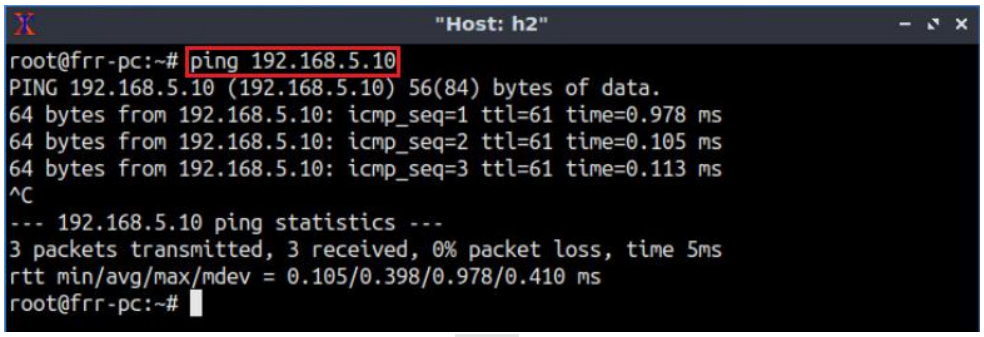
```
"Host: r5"
root@frr-pc:/etc/routers/r5# zebra
root@frr-pc:/etc/routers/r5# staticd
root@frr-pc:/etc/routers/r5# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

S>* 0.0.0.0/0 [1/0] via 192.168.15.1, r5-eth1, 00:00:06
C>* 192.168.5.0/24 is directly connected, r5-eth0, 00:00:08
C>* 192.168.15.0/30 is directly connected, r5-eth1, 00:00:08
frr-pc#
```

Figure 24. Displaying routing table of router r5.

Consider the figure above. By configuring default route, the router can communicate with other routers via 192.168.12.1.

Step 14. All the hosts will be reachable at this point. Test the connectivity between host h2 and host h5 using the `ping` command. In host h2's terminal, type the command specified below. To stop the test, press `Ctrl+c`. The figure below shows a successful connectivity test.

```
ping 192.168.5.10
```



```
root@frr-pc:~# ping 192.168.5.10
PING 192.168.5.10 (192.168.5.10) 56(84) bytes of data.
64 bytes from 192.168.5.10: icmp_seq=1 ttl=61 time=0.978 ms
64 bytes from 192.168.5.10: icmp_seq=2 ttl=61 time=0.105 ms
64 bytes from 192.168.5.10: icmp_seq=3 ttl=61 time=0.113 ms
^C
--- 192.168.5.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 5ms
rtt min/avg/max/mdev = 0.105/0.398/0.978/0.410 ms
root@frr-pc:~#
```

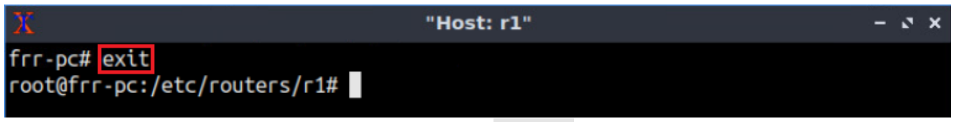
Figure 25. Output of `ping` command in host h2.

4 Configure VRF in ISP router

In this section, you will configure VRF in ISP router to create different routing tables for each campus. To do so, you will create two different VRFs assigned to different routing tables. Additionally, you will add the interfaces of the ISP router to the adequate VRF.

Step 1. In router r1, type the following command to exit the vtysh session:

```
exit
```

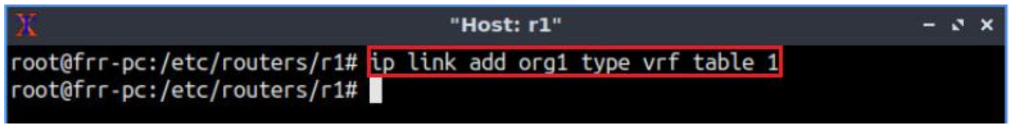


```
frr-pc# exit
root@frr-pc:/etc/routers/r1#
```

Figure 26. Exiting `vtysh` session.

Step 2. Type the following command to create a VRF and assign it to a routing table. VRF named *org1* will be created which is assigned to table 1.

```
ip link add org1 type vrf table 1
```

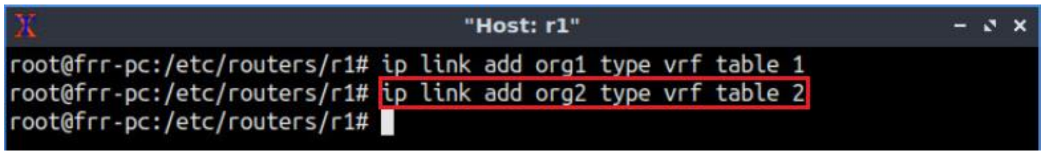


```
root@frr-pc:/etc/routers/r1# ip link add org1 type vrf table 1
root@frr-pc:/etc/routers/r1#
```

Figure 27. Creating VRF in router r1.

Step 3. Type the following command to create another VRF *org2* and assign it to table 2.

```
ip link add org2 type vrf table 2
```

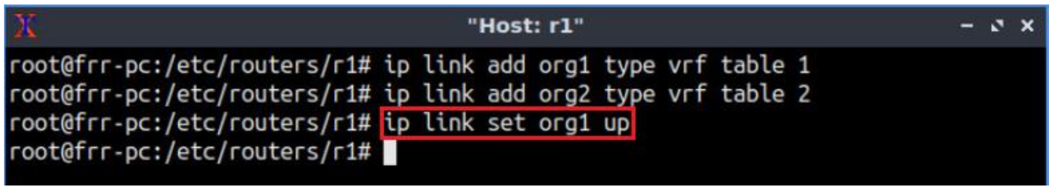


```
"Host: r1"
root@frr-pc:/etc/routers/r1# ip link add org1 type vrf table 1
root@frr-pc:/etc/routers/r1# ip link add org2 type vrf table 2
root@frr-pc:/etc/routers/r1# 
```

Figure 28. Creating VRF in router r1.

Step 4. Type the following command to enable VRF *org1*.

```
ip link set org1 up
```

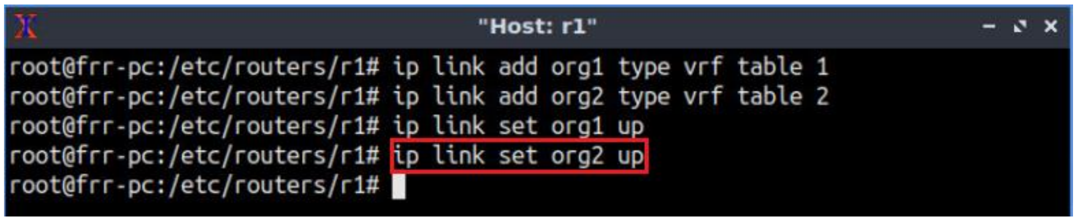


```
"Host: r1"
root@frr-pc:/etc/routers/r1# ip link add org1 type vrf table 1
root@frr-pc:/etc/routers/r1# ip link add org2 type vrf table 2
root@frr-pc:/etc/routers/r1# ip link set org1 up
root@frr-pc:/etc/routers/r1# 
```

Figure 29. Enabling VRF in router r1.

Step 5. Type the following command to enable VRF *org2*.

```
ip link set org2 up
```

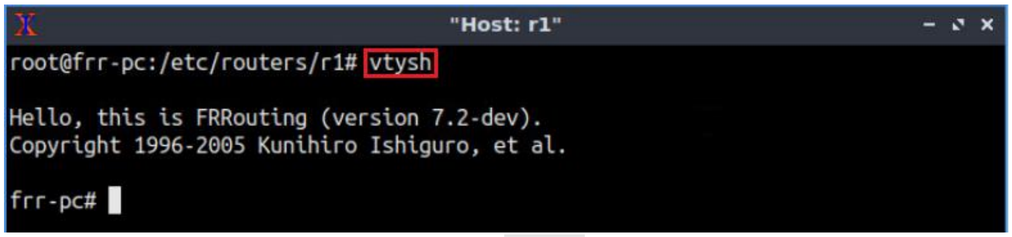


```
"Host: r1"
root@frr-pc:/etc/routers/r1# ip link add org1 type vrf table 1
root@frr-pc:/etc/routers/r1# ip link add org2 type vrf table 2
root@frr-pc:/etc/routers/r1# ip link set org1 up
root@frr-pc:/etc/routers/r1# ip link set org2 up
root@frr-pc:/etc/routers/r1# 
```

Figure 30. Enabling VRF in router r1.

Step 6. In order to enter to router r1 terminal, type the following command:

```
vtysh
```

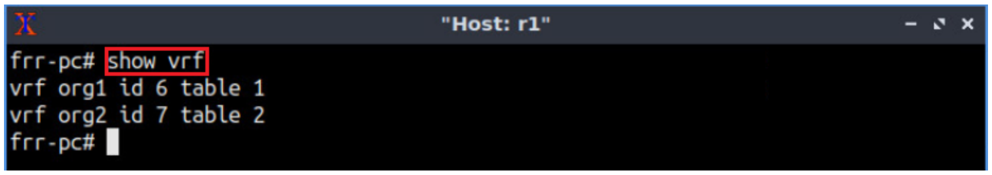


```
"Host: r1"
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# 
```

Figure 31. Starting `vtysh` in router r1.

Step 7. Type the following command to verify all the VRFs running in router r1. You will notice VRFs *org1* and *org2* in the VRF table.

```
show vrf
```

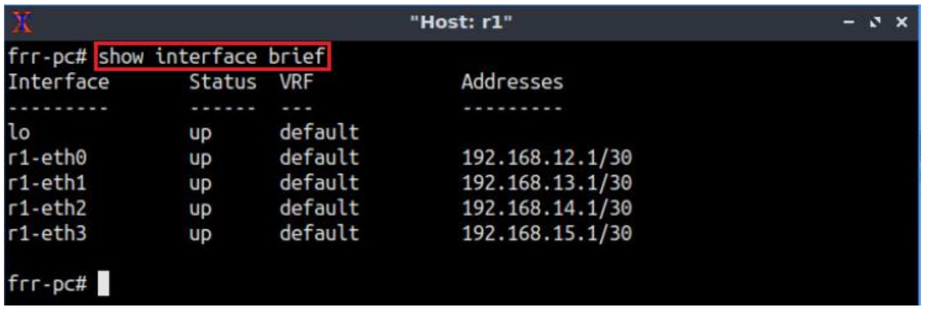


```
frr-pc# show vrf
vrf org1 id 6 table 1
vrf org2 id 7 table 2
frr-pc#
```

Figure 32. Verifying VRF list in router r1.

Step 8. Type the following command to verify interfaces in router r1. You will notice all the interfaces are still available in the global routing table.

```
show interface brief
```

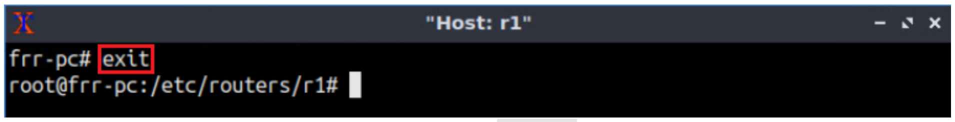


```
frr-pc# show interface brief
Interface       Status  VRF             Addresses
---------       ------  ---             ---------
lo              up      default
r1-eth0         up      default         192.168.12.1/30
r1-eth1         up      default         192.168.13.1/30
r1-eth2         up      default         192.168.14.1/30
r1-eth3         up      default         192.168.15.1/30

frr-pc#
```

Figure 33. Verifying interfaces in router r1.

Step 9. Type the following command to exit the vtysh session in router r1:

```
exit
```

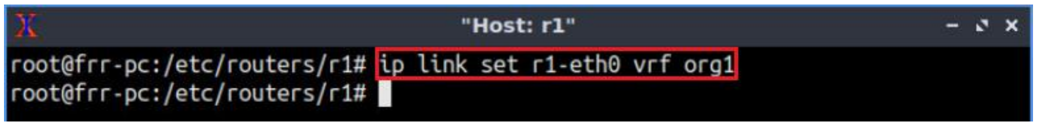


```
frr-pc# exit
root@frr-pc:/etc/routers/r1#
```

Figure 34. Exiting `vtysh` session.

Step 10. Type the following command to assign interface *r1-eth0* to VRF *org1*.

```
ip link set r1-eth0 vrf org1
```

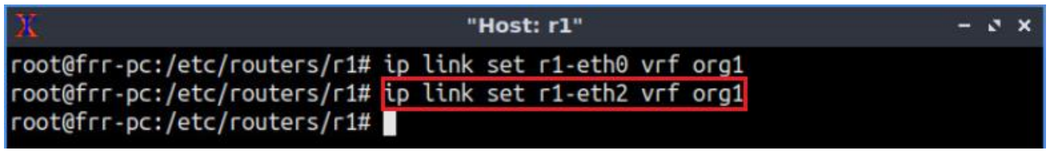


```
root@frr-pc:/etc/routers/r1# ip link set r1-eth0 vrf org1
root@frr-pc:/etc/routers/r1#
```

Figure 35. Assigning interface to VRF.

Step 11. Type the following command to assign interface *r1-eth2* to VRF *org1*.

```
ip link set r1-eth2 vrf org1
```

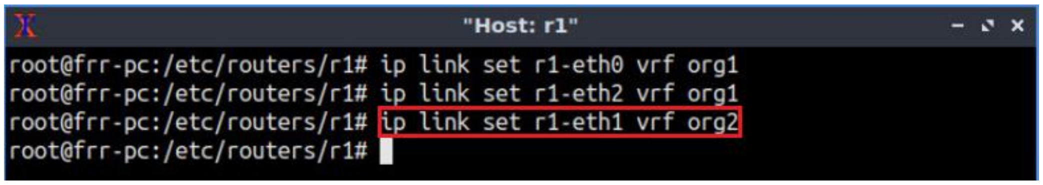


```
root@frr-pc:/etc/routers/r1# ip link set r1-eth0 vrf org1
root@frr-pc:/etc/routers/r1# ip link set r1-eth2 vrf org1
root@frr-pc:/etc/routers/r1#
```

Figure 36. Assigning interface to VRF.

Step 12. Type the following command to assign interface *r1-eth1* to VRF *org2*.

```
ip link set r1-eth1 vrf org2
```

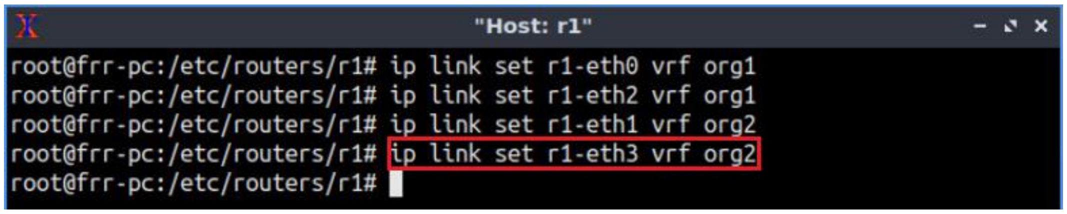


```
"Host: r1"
root@frr-pc:/etc/routers/r1# ip link set r1-eth0 vrf org1
root@frr-pc:/etc/routers/r1# ip link set r1-eth2 vrf org1
root@frr-pc:/etc/routers/r1# ip link set r1-eth1 vrf org2
root@frr-pc:/etc/routers/r1#
```

Figure 37. Assigning interface to VRF.

Step 13. Type the following command to assign interface *r1-eth3* to VRF *org2*.

```
ip link set r1-eth3 vrf org2
```

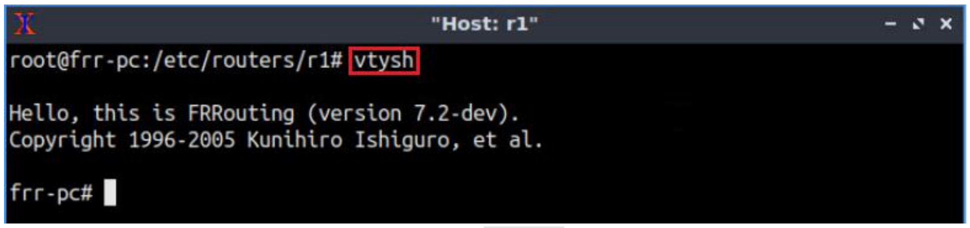


```
"Host: r1"
root@frr-pc:/etc/routers/r1# ip link set r1-eth0 vrf org1
root@frr-pc:/etc/routers/r1# ip link set r1-eth2 vrf org1
root@frr-pc:/etc/routers/r1# ip link set r1-eth1 vrf org2
root@frr-pc:/etc/routers/r1# ip link set r1-eth3 vrf org2
root@frr-pc:/etc/routers/r1#
```

Figure 38. Assigning interface to VRF.

Step 14. In order to enter to router r1 terminal, issue the following command:

```
vtysh
```

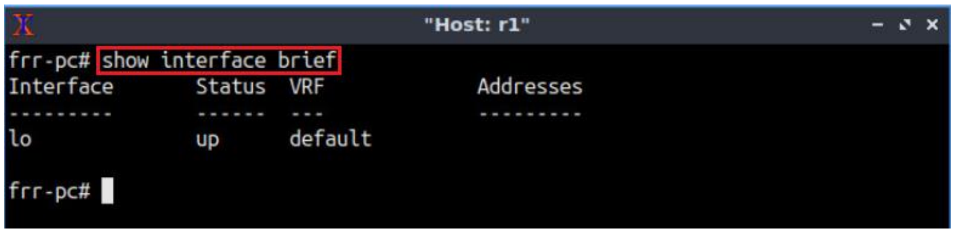


```
"Host: r1"
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 39. Starting `vtysh` in router r1.

Step 15. Type the following command to verify all the active interfaces in the global routing table. You will notice that the global routing table is empty at this point as all the interfaces are assigned to VRFs.

```
show interface brief
```

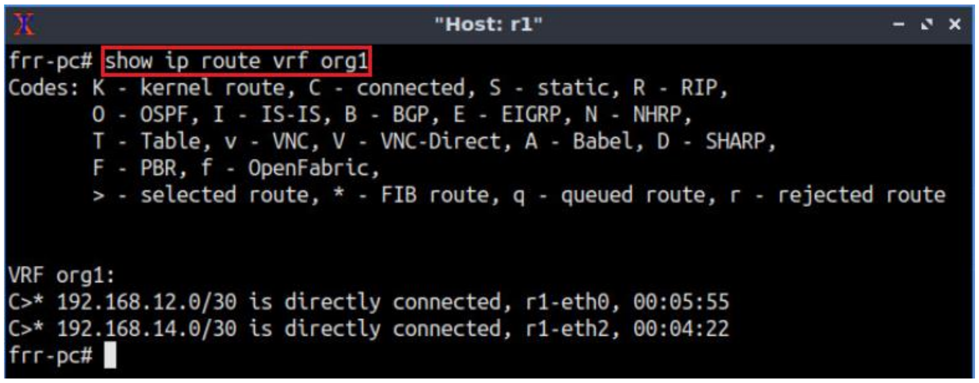


```
"Host: r1"
frr-pc# show interface brief
Interface       Status  VRF             Addresses
---------       ------  ---             ---------
lo              up      default

frr-pc#
```

Figure 40. Verifying interfaces in router r1.

Step 16. Type the following command to verify VRF table of *org1*. You will notice two interfaces, connected to router r2 (r1-eth0), and r4 (r1-eth2) are assigned to VRF *org1*.

```
show ip route vrf org1
```

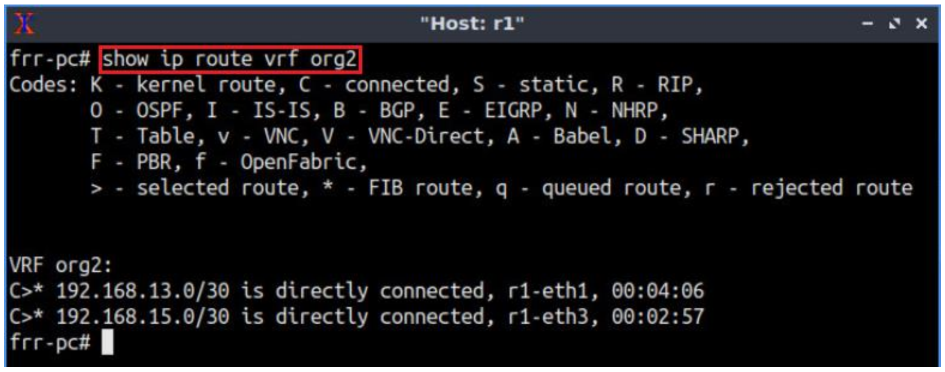


```
frr-pc# show ip route vrf org1
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route


VRF org1:
C>* 192.168.12.0/30 is directly connected, r1-eth0, 00:05:55
C>* 192.168.14.0/30 is directly connected, r1-eth2, 00:04:22
frr-pc#
```

Figure 41. Verifying VRF table in router r1.

Step 17. Type the following command to verify VRF table of *org2*. You will notice two interfaces, connected to router r3 (r1-eth1) and r5 (r1-eth3) are assigned to VRF *org2*.

```
show ip route vrf org2
```



```
frr-pc# show ip route vrf org2
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route


VRF org2:
C>* 192.168.13.0/30 is directly connected, r1-eth1, 00:04:06
C>* 192.168.15.0/30 is directly connected, r1-eth3, 00:02:57
frr-pc#
```

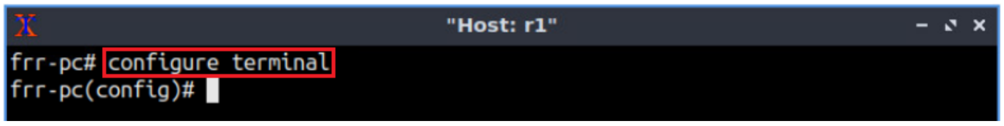
Figure 42. Verifying VRF table in router r1.

5 Configure static route in ISP router

Routers r2 and r4 belong to VRF *org1* whereas routers r3 and r5 belong to VRF *org2*. At this point, ISP router (r1) does not have any route to the destination routers (r2, r3, r4, and r5). In this section, you will configure static routes in ISP router for each VRF in order to get connectivity to all campus routers.

Step 1. To enable router r1 configuration mode, issue the following command:

```
configure terminal
```

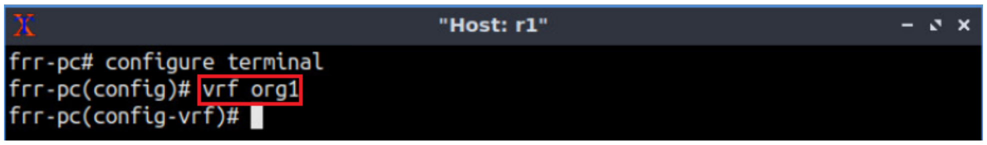


```
frr-pc# configure terminal
frr-pc(config)#
```

Figure 43. Enabling configuration mode in router r1.

Step 2. Type the following command to enable VRF *org1* configuration mode.

```
vrf org1
```

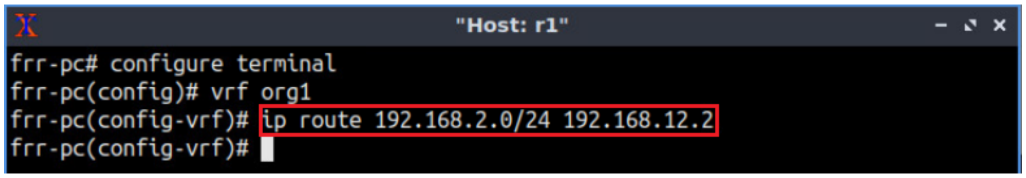


```
frr-pc# configure terminal
frr-pc(config)# vrf org1
frr-pc(config-vrf)#
```

Figure 44. Enabling VRF configuration mode in router r1.

Step 3. Type the following command to configure static route for VRF *org1*. Router r1 will reach network 192.168.2.0/24 via 192.168.12.2.

```
ip route 192.168.2.0/24 192.168.12.2
```

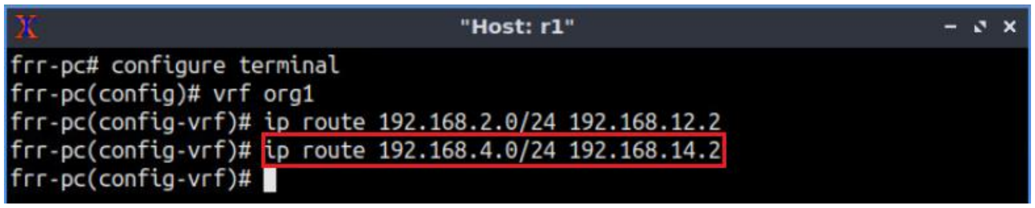


```
frr-pc# configure terminal
frr-pc(config)# vrf org1
frr-pc(config-vrf)# ip route 192.168.2.0/24 192.168.12.2
frr-pc(config-vrf)#
```

Figure 45. Configuring static route in router r1.

Step 4. Type the following command to configure static route for VRF *org1*. Router r1 will reach network 192.168.4.0/24 via 192.168.14.2.

```
ip route 192.168.4.0/24 192.168.14.2
```

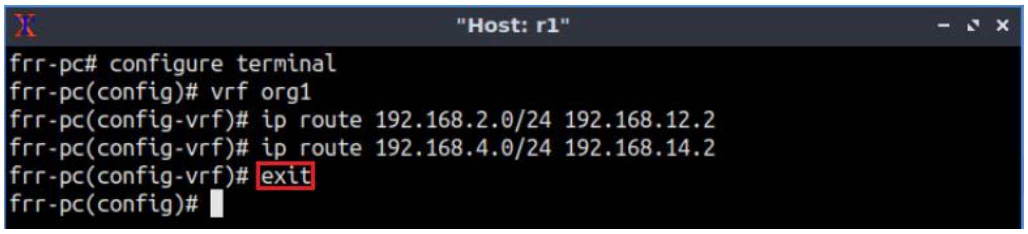


```
frr-pc# configure terminal
frr-pc(config)# vrf org1
frr-pc(config-vrf)# ip route 192.168.2.0/24 192.168.12.2
frr-pc(config-vrf)# ip route 192.168.4.0/24 192.168.14.2
frr-pc(config-vrf)#
```

Figure 46. Configuring static route on router r1.

Step 5. Type the following command to exit from the VRF configuration mode.

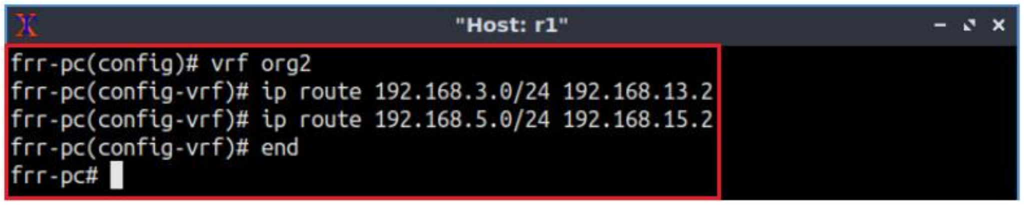
```
exit
```



```
frr-pc# configure terminal
frr-pc(config)# vrf org1
frr-pc(config-vrf)# ip route 192.168.2.0/24 192.168.12.2
frr-pc(config-vrf)# ip route 192.168.4.0/24 192.168.14.2
frr-pc(config-vrf)# exit
frr-pc(config)#
```

Figure 47. Exiting from VRF configuration mode.

Step 6. Configuring static route for VRF *org2* is similar to *org1*. Router r1 will reach networks 192.168.3.0/24 and 192.168.5.0/24 using the next-hop 192.168.13.2 and 192.168.15.2, respectively.



```
frr-pc(config)# vrf org2
frr-pc(config-vrf)# ip route 192.168.3.0/24 192.168.13.2
frr-pc(config-vrf)# ip route 192.168.5.0/24 192.168.15.2
frr-pc(config-vrf)# end
frr-pc#
```

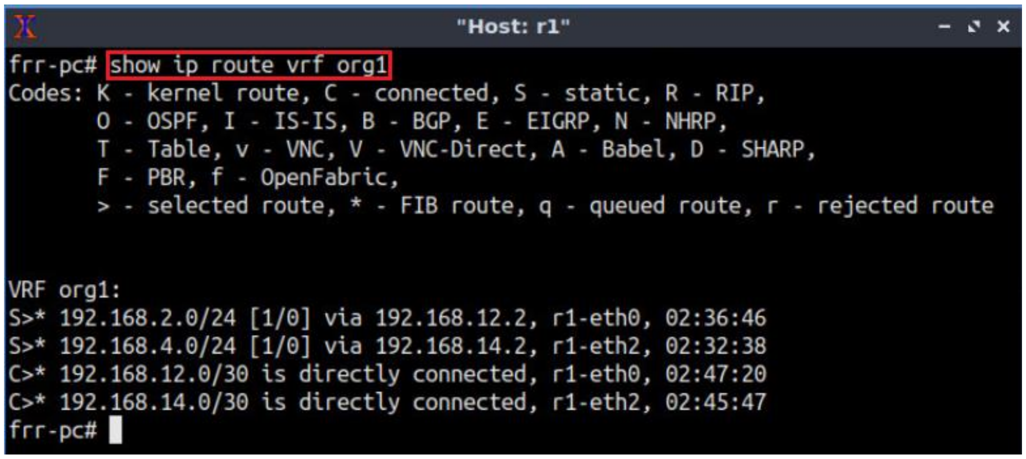
Figure 48. Configuring static route in router r1.

6 Verify VRF configuration

In this section, you will verify the VRF configuration. According to the lab requirements, *org1* and *org2* have separate routing tables and campus hosts belong to a particular organization can communicate with each other.

Step 1. Type the following command to verify VRF table of *org1*. You will notice that the VRF table contains static routes for the networks 192.168.2.0/24 and 192.168.4.0/24. These networks are unable to communicate with any other networks connected to the ISP router.

```
show ip route vrf org1
```

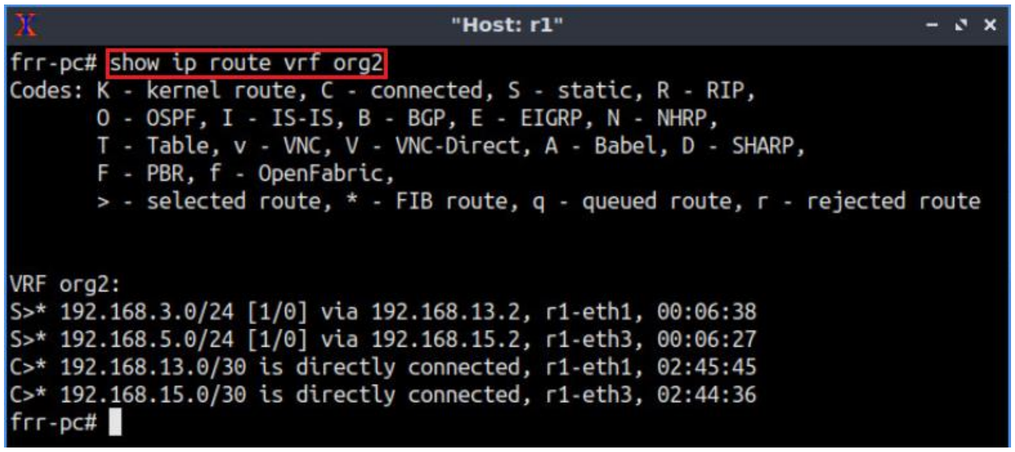


```
frr-pc# show ip route vrf org1
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route


VRF org1:
S>* 192.168.2.0/24 [1/0] via 192.168.12.2, r1-eth0, 02:36:46
S>* 192.168.4.0/24 [1/0] via 192.168.14.2, r1-eth2, 02:32:38
C>* 192.168.12.0/30 is directly connected, r1-eth0, 02:47:20
C>* 192.168.14.0/30 is directly connected, r1-eth2, 02:45:47
frr-pc#
```

Figure 49. Verifying VRF table in router r1.

Step 2. Type the following command to verify VRF table of *org2*. You will notice that the VRF table contains static routes for the networks 192.168.3.0/24 and 192.168.5.0/24. These networks are unable to communicate with any other networks connected to the ISP router.

```
show ip route vrf org2
```

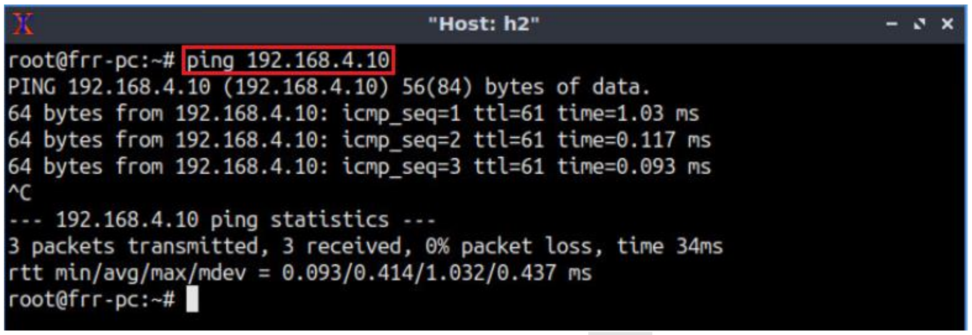


```
frr-pc# show ip route vrf org2
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route


VRF org2:
S>* 192.168.3.0/24 [1/0] via 192.168.13.2, r1-eth1, 00:06:38
S>* 192.168.5.0/24 [1/0] via 192.168.15.2, r1-eth3, 00:06:27
C>* 192.168.13.0/30 is directly connected, r1-eth1, 02:45:45
C>* 192.168.15.0/30 is directly connected, r1-eth3, 02:44:36
frr-pc#
```

Figure 50. Verifying VRF table in router r1.

Step 3. Test the connectivity between host h2 and host h4 using the `ping` command. In host h2, type the command specified below. To stop the test, press `Ctrl+c`. The figure below shows a successful connectivity test as both the hosts belong to the same VRF table (*org1*).

```
ping 192.168.4.10
```

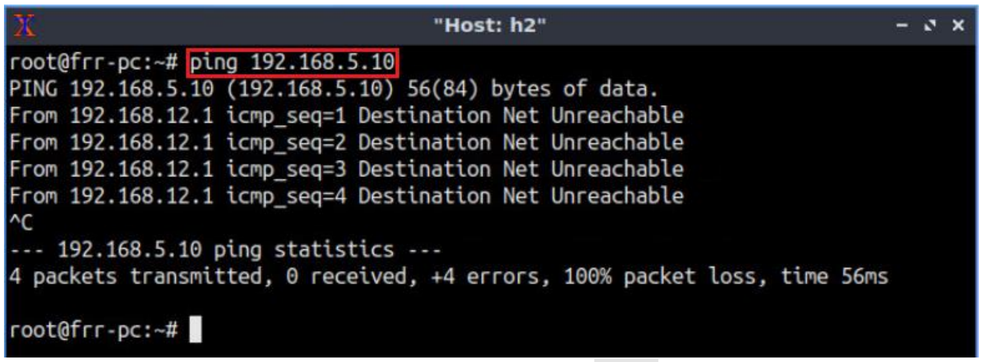


```
root@frr-pc:~# ping 192.168.4.10
PING 192.168.4.10 (192.168.4.10) 56(84) bytes of data.
64 bytes from 192.168.4.10: icmp_seq=1 ttl=61 time=1.03 ms
64 bytes from 192.168.4.10: icmp_seq=2 ttl=61 time=0.117 ms
64 bytes from 192.168.4.10: icmp_seq=3 ttl=61 time=0.093 ms
^C
--- 192.168.4.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 34ms
rtt min/avg/max/mdev = 0.093/0.414/1.032/0.437 ms
root@frr-pc:~#
```

Figure 51. Connectivity test using `ping` command.

Step 4. Test the connectivity between host h2 and host h5 using the `ping` command. In host h2, type the command specified below. To stop the test, press `Ctrl+c`. Host h2 cannot reach host h5 as the destination host (h5) belongs to a different VRF table (*org2*).

```
ping 192.168.5.10
```



```
root@frr-pc:~# ping 192.168.5.10
PING 192.168.5.10 (192.168.5.10) 56(84) bytes of data.
From 192.168.12.1 icmp_seq=1 Destination Net Unreachable
From 192.168.12.1 icmp_seq=2 Destination Net Unreachable
From 192.168.12.1 icmp_seq=3 Destination Net Unreachable
From 192.168.12.1 icmp_seq=4 Destination Net Unreachable
^C
--- 192.168.5.10 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 56ms

root@frr-pc:~#
```

Figure 52. Connectivity test using `ping` command.

This concludes Lab 6. Stop the emulation and then exit out of MiniEdit.

References

1. Greek University, “*Static Route*”, [Online]. Available: https://geek-university.com/ccna/static-routes/#:~:text=Static%20routes%20are%20manually%20added,to%20one%20of%20its%20interfaces.
2. Cisco, “*Virtual Route Forwarding Design Guide*”, 2008, [Online]. Available: https://www.cisco.com/c/en/us/td/docs/voice_ip_comm/cucme/vrf/design/guide/vrfDesignGuide.html
3. Rissal Efendi, “*A Simulation Analysis of Latency and Packet Loss on Virtual Private Network through Multi Virtual Routing and Forwarding*”, 2012.
4. Juniper, “Creating Unique VPN Routes Using VRF Tables”, [Online]. Available: https://www.juniper.net/documentation/en_US/junos/topics/topic-map/l3-vpns-routes-vrf-tables.html
5. Plixer, “*What is VRF: Virtual Routing and Forwarding*”, [Online]. Available: https://www.plixer.com/blog/what-is-vrf-virtual-routing-and-forwarding/
6. Mininet walkthrough, [Online]. Available: http://mininet.org.
7. Linux foundation collaborative projects, “*FRRouting: what’s in your router*”, [Online]. Available: https://frrouting.org/

MPLS AND ADVANCED BGP TOPICS

Lab 7: MPLS Layer 3 VPN using MP-BGP

Document Version: 03-04-2021

Award 1829698
"CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers"

Contents

Overview

The lab discusses the concept of Multiprotocol Label Switching Layer 3 Virtual Private Network (MPLS Layer 3 VPN) which provides network virtualization solutions for each customer connected to service provider. The lab also includes the concept of Route Distinguisher (RD) and Route Target (RT) that are used in MPLS VPN to distinguish and exchange routes. In this lab, MPLS Layer3 VPN will be created and verified while hosts are using overlapping IPv4 prefixes.

Objectives

By the end of this lab, you should be able to:

1. Explain the concept of Virtual Routing and Forwarding (VRF).
2. Introduces the concept of Layer3 VPN.
3. Demonstrate the concept of RD and RT.
4. Enable and verify MPLS implementation.
5. Configure VRF instances.
6. Use Multiprotocol BGP (MP-BGP) to exchange VPN routing updates.

Lab settings

The information in Table 1 provides the credentials to access Client machine.

Table 1**.** Credentials to access Client machine.

Device	**Account**	**Password**
Client	admin	password

Lab roadmap

This lab is organized as follows:

1. Section 1: Introduction.
2. Section 2: Lab topology.
3. Section 3: Configure MP-BGP on PE routers.
4. Section 4: Advertise VPN routes.
5. Section 5: Verify configuration.

1 Introduction

1.1 MPLS Layer 3 VPN

An MPLS VPN is a Layer 3 VPN that allows the routing of packets through a MPLS core. This type of VPN provides a customer with connections to multiple sites through a service provider's network. The service provider not only provides the physical connection, but the ability to dynamically route between the VPN endpoints. This is impressive when one considers that the customers may not be using globally unique Layer 3 addresses. For instance, different customers can use private addresses (sometimes overlapping addresses) and use the same transit provider. The Provider Edge (PE) routers control the entire MPLS VPN from end to end. The entire communication is forwarded using layer 3 Virtual Routing and Forwarding (VRF) techniques. Layer 3 VPN requires border gateway protocol (BGP) to send and receive VPN-related data and utilizes VRF techniques to create and manage user data[1].

1.2 Route distinguisher (RD) and Route Target (RT)

Figure 1 shows that two organizations (Org 1 and Org 2) connected to service provider network. The VPN prefixes are propagated across the MPLS VPN network by Multiprotocol BGP (MP-BGP). When BGP carries these IPv4 prefixes across the service provider network, they must be unique. If the customers had overlapping IP addressing, the routing would be wrong[3].

A Route Distinguisher (RD) is a 64-bit field used to make the VRF prefixes unique when MP-BGP carries them. When VPN routes are advertised among PE routers via MP-BGP, the RD is included as part of the route along with the IP prefix.

A Route Target (RT) is a BGP extended community that indicates which routes should be imported from MP-BGP into the VRF[3]. The RT is used to identify a subset of routes within the BGP unicast table that should be used in a VRF for a particular customer.

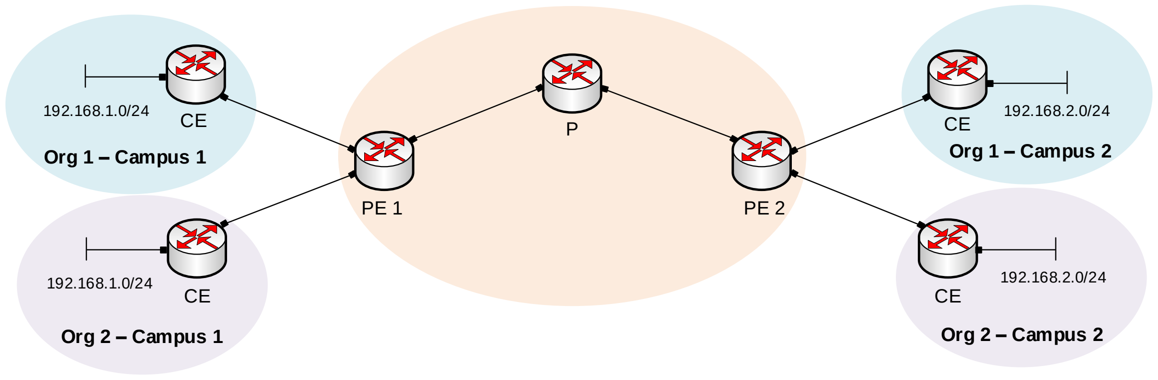


Figure 1. VRF with overlapping IPv4 prefixes.

Consider Figure 1. Both the organizations (*org1* and *org2*) are using the same prefixes (192.168.1.0/24 & 192.168.2.0/24). PE routers will assign RD value to each prefix to distinguish the same prefix from different customer. On PE routers, if we assign RD value

1:1 for *org1*, and RD value 2:2 for *org2*, then the prefixes will become unique as the following table:

VRF	Prefix
org 1	1:1:192.168.1.0/24
org 2	2:2:192.168.1.0/24
org 1	1:1:192.168.2.0/24
org2	2:2:192.168.2.0/24

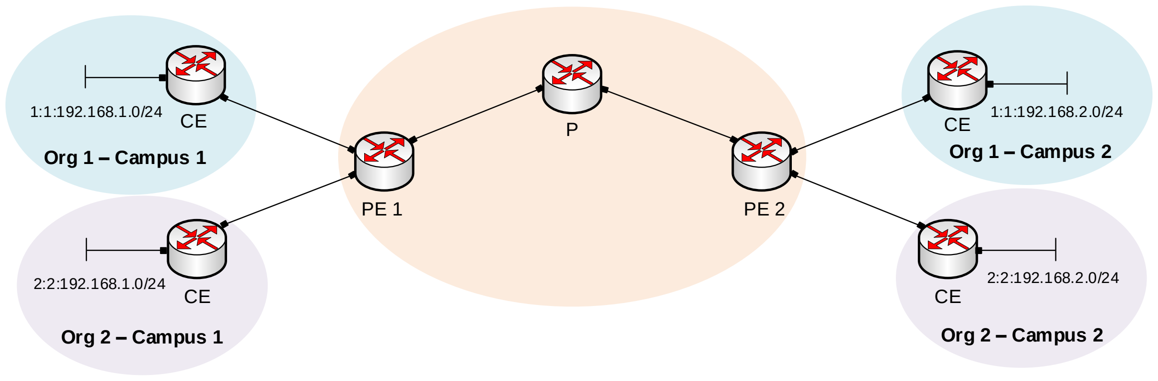


Figure 2. VRF with unique IPv4 prefixes using RD value.

Consider Figure 2. By assigning an RD in front of the route, we have created a globally unique set of BGP prefixes in the BGP table that can be shared between peers[4]. At this point, PE1 does not know which of these routes belong to *org1*, and which of these routes belong to *org2*. If we assign RT value 100:100 on both sides of *org1*, routes that have this 100:100 value will be inserted into the routing table of *org1*.

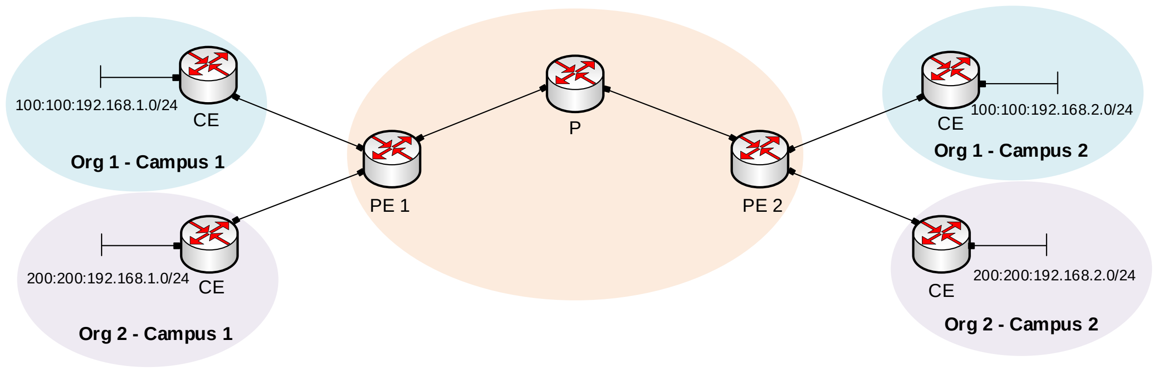


Figure 3. Creating separate VRF instances according to RT values.

Consider Figure 3. PE routers are using RT values (both import and export) 100:100 for *org1*, RT value 200:200 for *org2*. A VRF instance will be created for the routes using the same RT value.

2 Lab topology

Consider Figure 4. Two organizations (*Org1* and *Org2*) have two sites (campus 1 and campus 2) and those are connected to ISP through static routes. Campuses from different organizations are using overlapping IP addresses. PE routers (r5 and r7) contains separate routing table for each organization. MP-BGP is running on PE routers to exchange VPN routes. BGP advertises VPN routes with MPLS label and RTs.

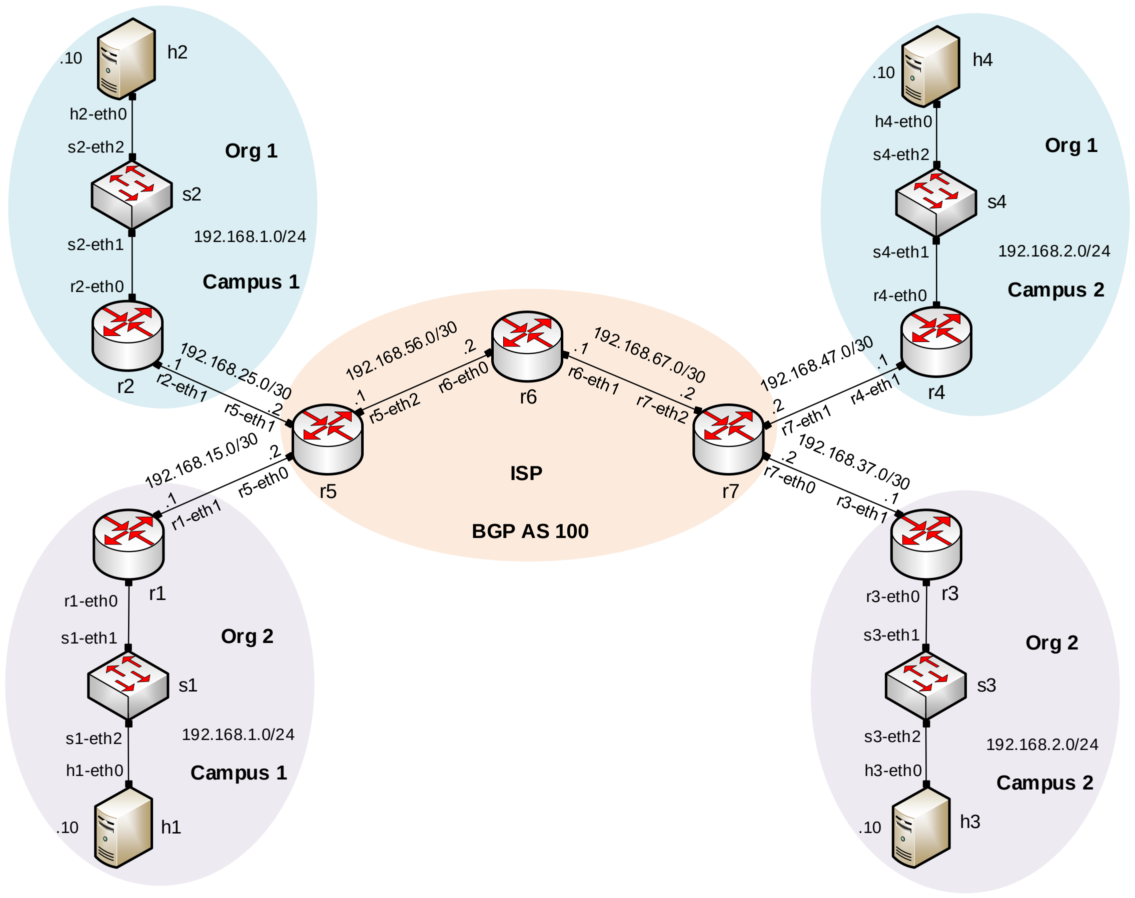


Figure 4. Lab topology.

2.1 Lab settings

Routers and hosts are already configured according to the IP addresses shown in Table 2.

Table 2. Topology information.

Device	Interface	IPV4 Address	Subnet	Default gateway
r1	r1-eth0	192.168.1.1	/24	N/A
	r1-eth1	192.168.15.1	/30	N/A
r2	r2-eth0	192.168.1.1	/24	N/A
	r2-eth1	192.168.25.1	/30	N/A
r3	r3-eth0	192.168.2.1	/24	N/A
	r3-eth1	192.168.37.1	/30	N/A

r4	r4-eth0	192.168.2.1	/24	N/A
	r4-eth1	192.168.47.1	/30	N/A
r5	r5-eth0	192.168.15.2	/30	N/A
	r5-eth1	192.168.25.2	/30	N/A
	r5-eth2	192.168.56.1	/30	N/A
	lo	5.5.5.5	/32	N/A
r6	r6-eth0	192.168.56.2	/30	N/A
	r6-eth1	192.168.67.1	/30	N/A
	lo	6.6.6.6	/32	N/A
r7	r7-eth0	192.168.37.2	/30	N/A
	r7-eth1	192.168.47.2	/30	N/A
	r7-eth2	192.168.67.2	/30	N/A
	lo	7.7.7.7	/32	N/A
h1	h1-eth0	192.168.1.10	/24	192.168.1.1
h2	h2-eth0	192.168.1.10	/24	192.168.1.1
h3	h3-eth0	192.168.2.10	/24	192.168.2.1
h4	h4-eth0	192.168.2.10	/24	192.168.2.1

2.2 Open topology and load the configuration

Step 1. Start by launching Miniedit by clicking on Desktop’s shortcut. When prompted for a password, type `password`.

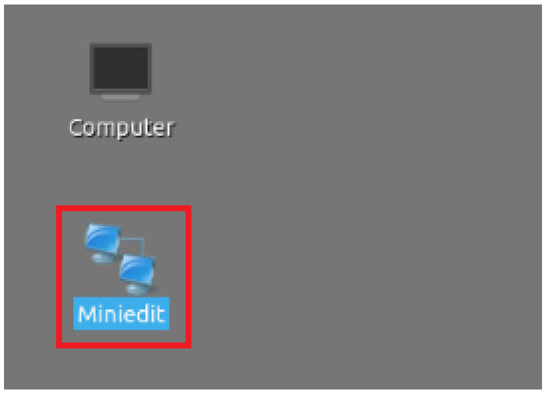


Figure 5. MiniEdit shortcut.

Step 2. On Miniedit's menu bar, click on *File* then *open* to load the lab's topology. Locate the *lab7.mn* topology file in the default directory, */home/frr/MPLS_advanced_BGP/lab7* and click on *Open*.

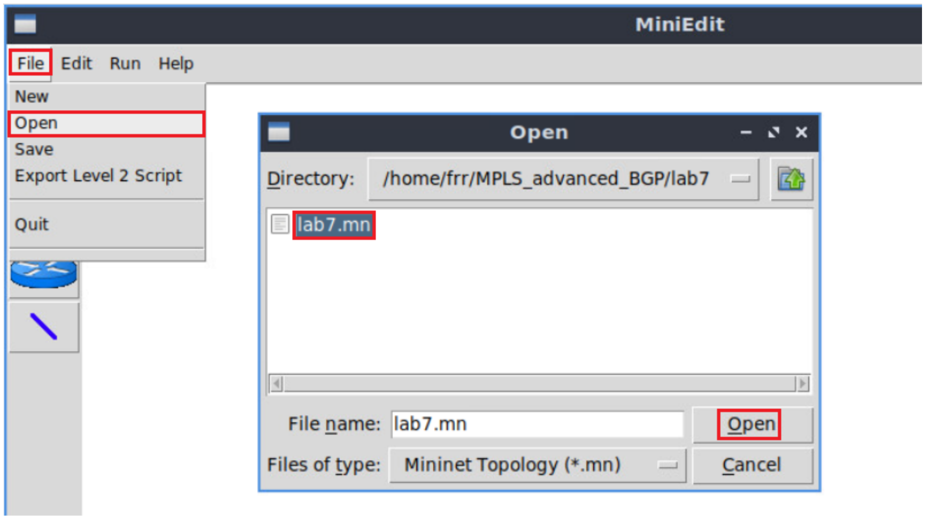


Figure 6. MiniEdit's Open dialog.

At this point the topology is loaded with all the required network components. You will execute a script that will load the configuration of the routers.

Step 3. Open the Linux terminal.

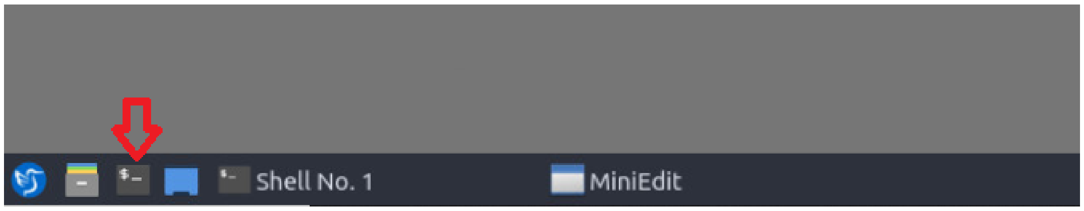
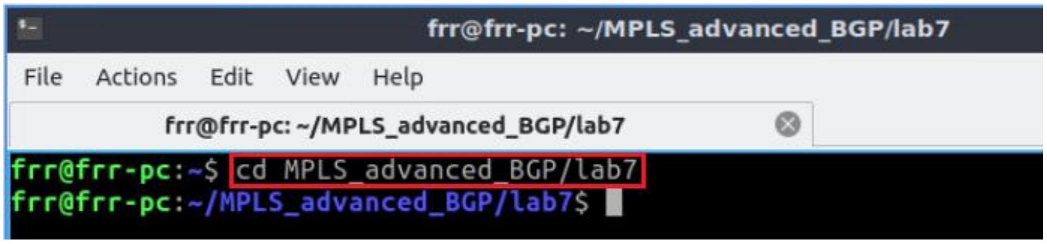


Figure 7. Opening Linux terminal.

Step 4. Click on the Linux's terminal and navigate into *MPLS_advanced_BGP/lab7* directory by issuing the following command. This folder contains a configuration file and the script responsible for loading the configuration. The configuration file will assign the IP addresses to the routers' interfaces. The `cd` command is short for change directory followed by an argument that specifies the destination directory.

```
cd MPLS_advanced_BGP/lab7
```

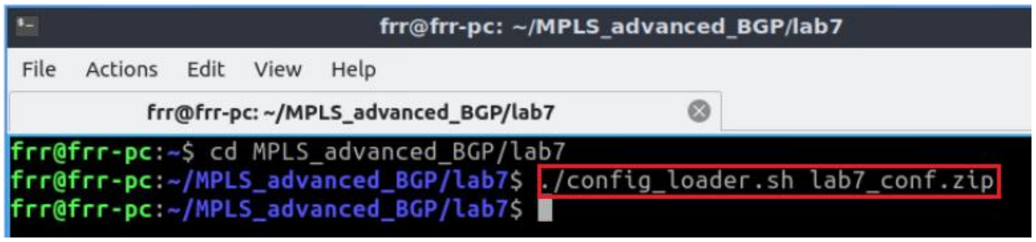


```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab7
frr@frr-pc:~/MPLS_advanced_BGP/lab7$
```

Figure 8. Entering to the *MPLS_advanced_BGP/lab7* directory.

Step 5. To execute the shell script, type the following command. The argument of the program corresponds to the configuration zip file that will be loaded in all the routers in the topology.

```
./config_loader.sh lab7_conf.zip
```


frr@frr-pc: ~/MPLS_advanced_BGP/lab7
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab7

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab7
frr@frr-pc:~/MPLS_advanced_BGP/lab7$ ./config_loader.sh lab7_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab7$
```

Figure 9. Executing the shell script to load the configuration.

Step 6. At this point hosts h1, h2, h3 and h4 interfaces are configured. To proceed with the emulation, click on the *Run* button located in lower left-hand side.

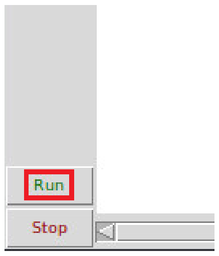


Figure 10. Starting the emulation.

Step 7. Click on Mininet's terminal, i.e., the one launched when MiniEdit was started.

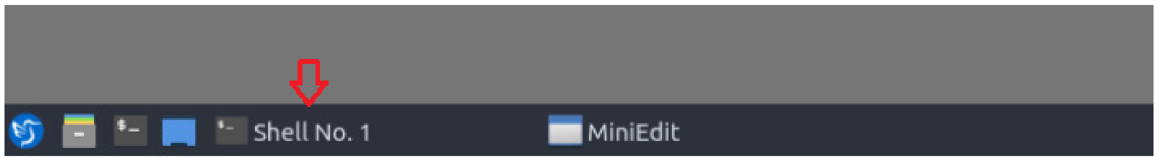
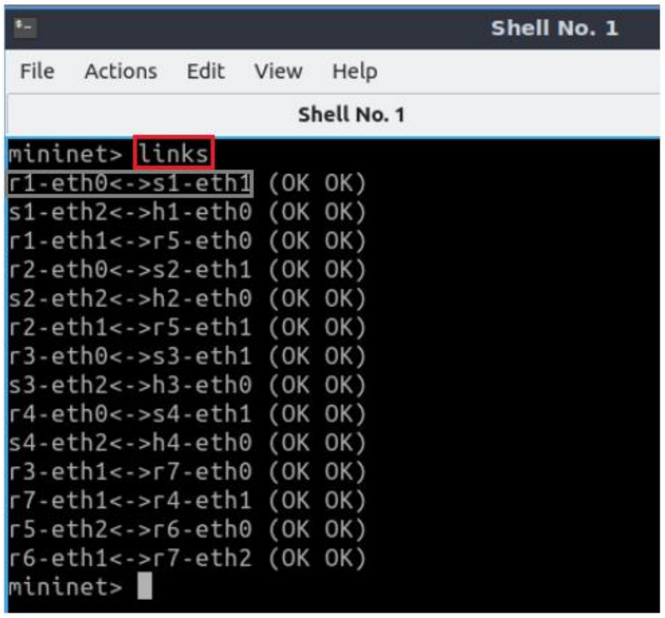


Figure 11. Opening Mininet's terminal.

Step 8. Issue the following command to display the interface names and connections.

```
links
```



```
mininet> links
r1-eth0<->s1-eth1 (OK OK)
s1-eth2<->h1-eth0 (OK OK)
r1-eth1<->r5-eth0 (OK OK)
r2-eth0<->s2-eth1 (OK OK)
s2-eth2<->h2-eth0 (OK OK)
r2-eth1<->r5-eth1 (OK OK)
r3-eth0<->s3-eth1 (OK OK)
s3-eth2<->h3-eth0 (OK OK)
r4-eth0<->s4-eth1 (OK OK)
s4-eth2<->h4-eth0 (OK OK)
r3-eth1<->r7-eth0 (OK OK)
r7-eth1<->r4-eth1 (OK OK)
r5-eth2<->r6-eth0 (OK OK)
r6-eth1<->r7-eth2 (OK OK)
mininet>
```

Figure 12. Displaying network interfaces.

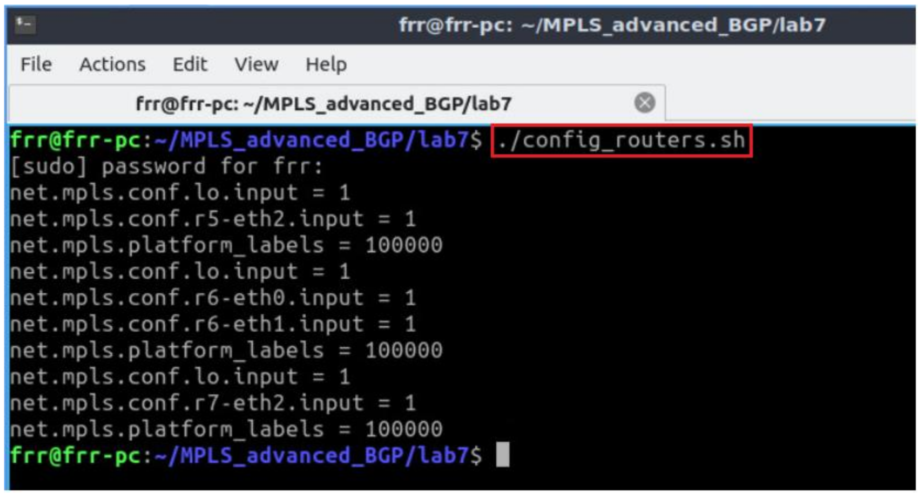
In the figure above, the link displayed within the gray box indicates that interface eth0 of router r1 connects to interface eth1 of switch s1 (i.e., *r1-eth0<->s1-eth1*).

2.3 Enable MPLS forwarding in routers r5, r6 and r7

In this section, you will enable MPLS forwarding in the kernel. All the router interfaces assign labels which is used to forward packets. You will assign value 1 to all the router interfaces so that they participate in label processing. *Platform_labels* is the table which recognizes all the assigned labels and participates in label forwarding. You will assign value 100000 (maximum value for label forwarding) to *platform_labels* in order to enable label forwarding.

Step 1. In the Linux terminal type the following command to enable MPLS forwarding in routers r5, r6 and r7. When prompted for password, type `password`.

```
./config_routers.sh
```


frr@frr-pc: ~/MPLS_advanced_BGP/lab7
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab7

```
frr@frr-pc:~/MPLS_advanced_BGP/lab7$ ./config_routers.sh
[sudo] password for frr:
net.mpls.conf.lo.input = 1
net.mpls.conf.r5-eth2.input = 1
net.mpls.platform_labels = 100000
net.mpls.conf.lo.input = 1
net.mpls.conf.r6-eth0.input = 1
net.mpls.conf.r6-eth1.input = 1
net.mpls.platform_labels = 100000
net.mpls.conf.lo.input = 1
net.mpls.conf.r7-eth2.input = 1
net.mpls.platform_labels = 100000
frr@frr-pc:~/MPLS_advanced_BGP/lab7$
```

Figure 13. Configuring MPLS forwarding in routers r5, r6, and r7.

The output will show a successful configuration.

Routers within ISP network will participate in label forwarding. Router interfaces connected to customer routers do not perform label forwarding.

2.4 Load zebra daemon and routing daemons in all routers

In this section, you will load the required routing daemons to enable connectivity between the routers.

Step 1. Hold right-click on router r1 and select *Terminal.*

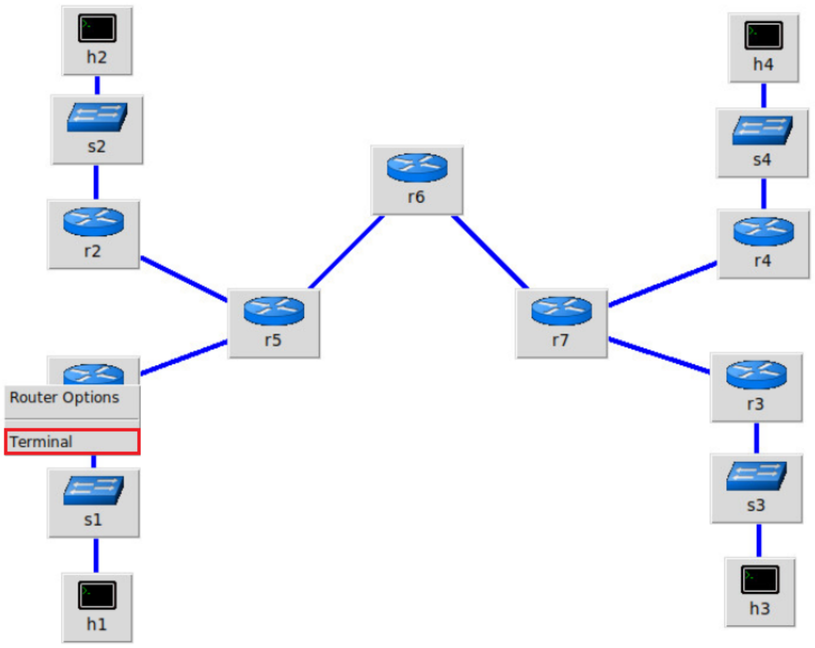
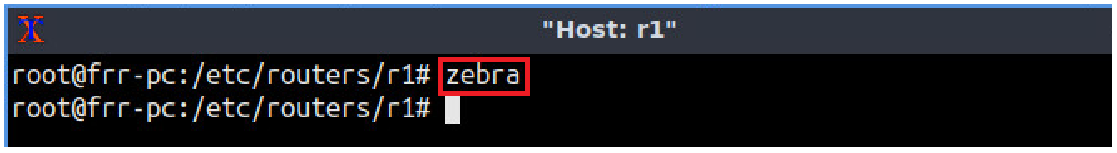


Figure 14. Opening a terminal on router r1.

Step 2. On router r1 terminal, you will start *zebra* daemon, which is a multi-server routing software that provides TCP/IP based routing protocols. The configuration will not be working if you do not enable zebra daemon initially. In order to start the zebra, type the following command:

```
zebra
```

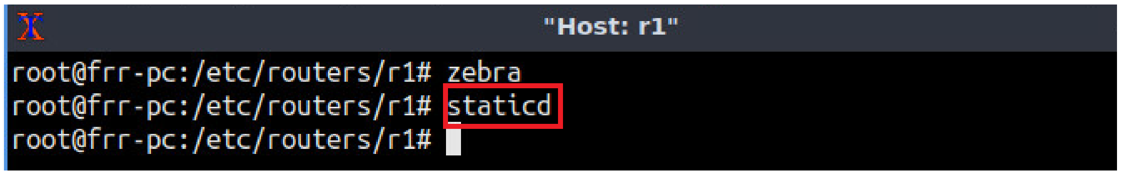

"Host: r1"

```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# 
```

Figure 15. Starting zebra daemon.

Step 3. To initialize the static routing daemon, type the following command:

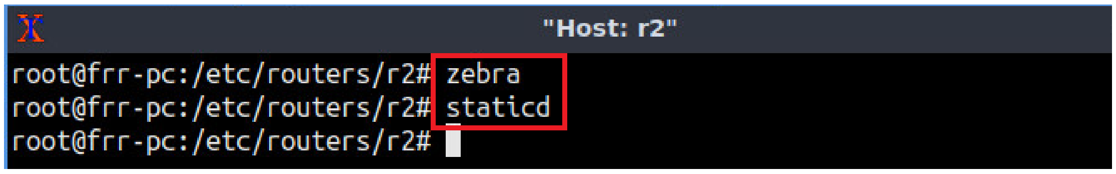
```
staticd
```


"Host: r1"

```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# staticd
root@frr-pc:/etc/routers/r1# 
```

Figure 16. Starting static daemon on router r1.

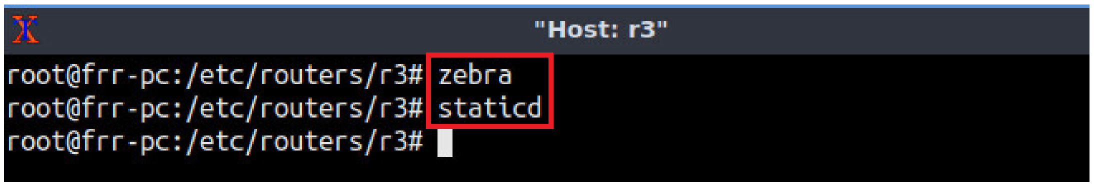
Step 4. Proceed similarly in router r2. Those steps are summarized below.


"Host: r2"

```
root@frr-pc:/etc/routers/r2# zebra
root@frr-pc:/etc/routers/r2# staticd
root@frr-pc:/etc/routers/r2# 
```

Figure 17. Starting daemons on router r2.

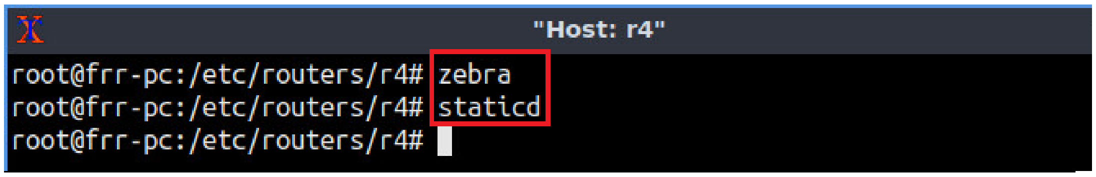
Step 5. Proceed similarly in router r3. Those steps are summarized below.


"Host: r3"

```
root@frr-pc:/etc/routers/r3# zebra
root@frr-pc:/etc/routers/r3# staticd
root@frr-pc:/etc/routers/r3# 
```

Figure 18. Starting daemons on router r3.

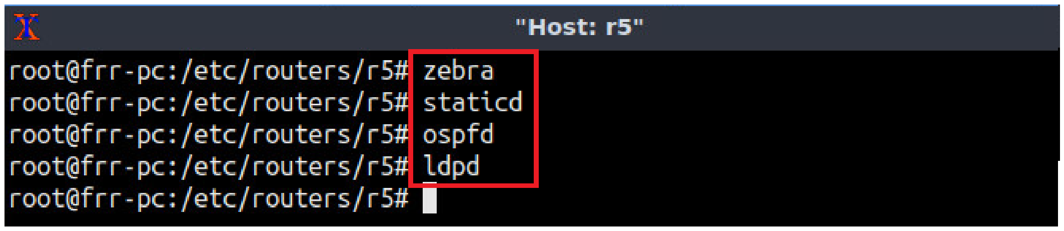
Step 6. Proceed similarly in router r4. Those steps are summarized below.


"Host: r4"

```
root@frr-pc:/etc/routers/r4# zebra
root@frr-pc:/etc/routers/r4# staticd
root@frr-pc:/etc/routers/r4# 
```

Figure 19. Starting daemons on router r4.

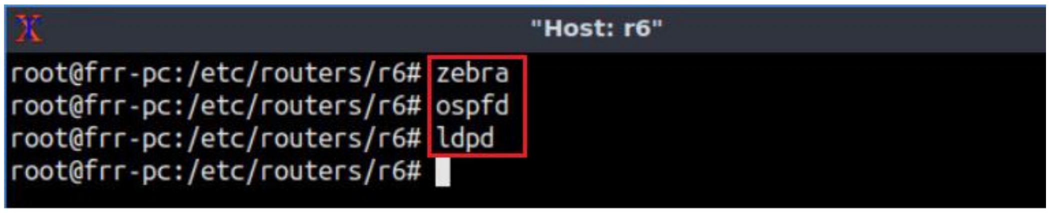
Step 7. In router r5 enable the following daemons.



```
"Host: r5"
root@frr-pc:/etc/routers/r5# zebra
root@frr-pc:/etc/routers/r5# staticd
root@frr-pc:/etc/routers/r5# ospfd
root@frr-pc:/etc/routers/r5# ldpd
root@frr-pc:/etc/routers/r5#
```

Figure 20. Starting daemons on router r5.

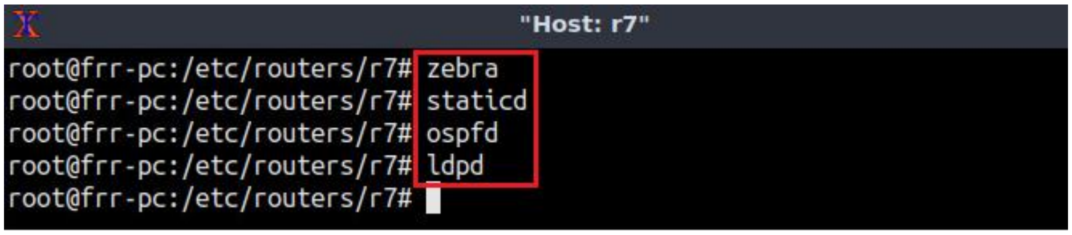
Step 8. Similarly, in router r6 enable the following daemons. Those steps are summarized below.



```
"Host: r6"
root@frr-pc:/etc/routers/r6# zebra
root@frr-pc:/etc/routers/r6# ospfd
root@frr-pc:/etc/routers/r6# ldpd
root@frr-pc:/etc/routers/r6#
```

Figure 21. Starting daemons on router r6.

Step 9. Similarly, in router r7 enable the following daemons. Those steps are summarized below.

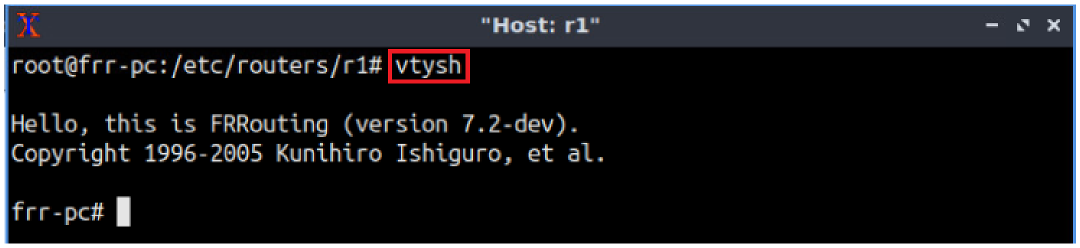


```
"Host: r7"
root@frr-pc:/etc/routers/r7# zebra
root@frr-pc:/etc/routers/r7# staticd
root@frr-pc:/etc/routers/r7# ospfd
root@frr-pc:/etc/routers/r7# ldpd
root@frr-pc:/etc/routers/r7#
```

Figure 22. Starting daemons on router r7.

Step 10. In order to enter to router r1 terminal, issue the following command:

```
vtysh
```

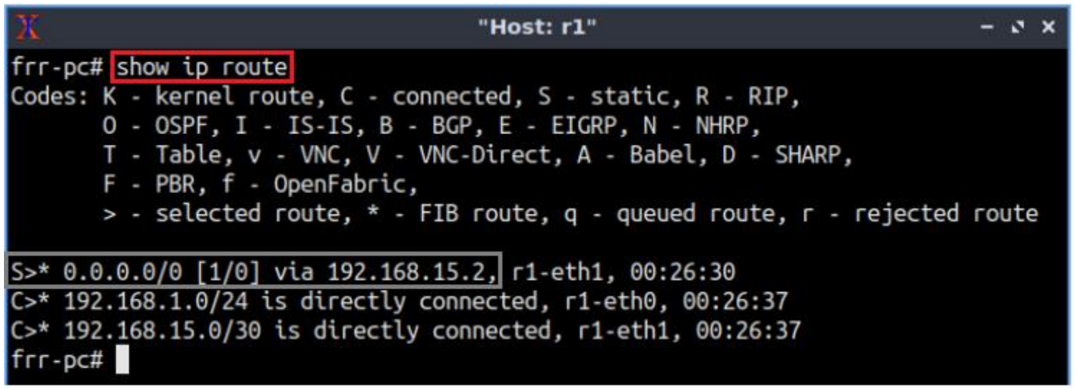


```
"Host: r1"
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 23. Starting vtysh on router r1.

Step 11. In order to verify static routes on router r1 , issue the following command:

```
show ip route
```



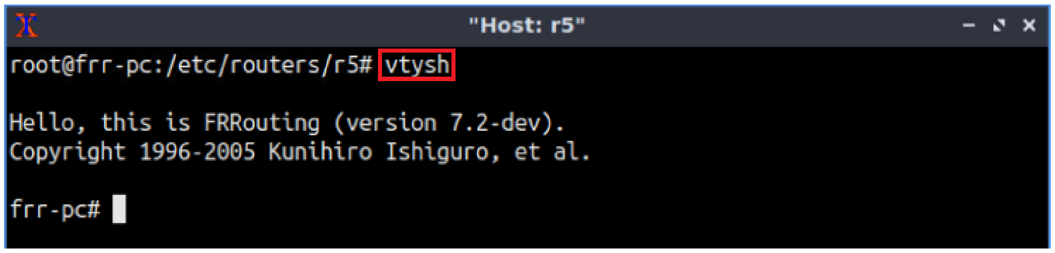
```
"Host: r1"
frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

S>* 0.0.0.0/0 [1/0] via 192.168.15.2, r1-eth1, 00:26:30
C>* 192.168.1.0/24 is directly connected, r1-eth0, 00:26:37
C>* 192.168.15.0/30 is directly connected, r1-eth1, 00:26:37
frr-pc#
```

Figure 24. Verifying routing table on router r1.

Consider the figure above. You will notice a static route, router r1 can communicate with other routers via 192.168.15.2.

Step 12. In order to enter to router r5 terminal, issue the following command:

```
vtysh
```

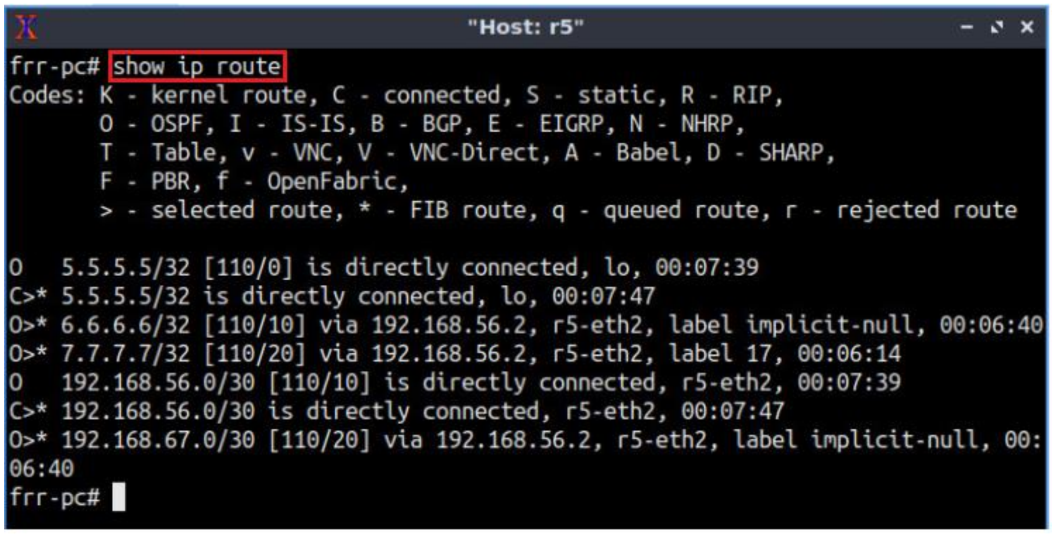


```
"Host: r5"
root@frr-pc:/etc/routers/r5# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 25. Starting vtysh on router r5.

Step 13. In order to verify routing table on router r5 terminal, issue the following command:

```
show ip route
```



```
"Host: r5"
frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

O   5.5.5.5/32 [110/0] is directly connected, lo, 00:07:39
C>* 5.5.5.5/32 is directly connected, lo, 00:07:47
O>* 6.6.6.6/32 [110/10] via 192.168.56.2, r5-eth2, label implicit-null, 00:06:40
O>* 7.7.7.7/32 [110/20] via 192.168.56.2, r5-eth2, label 17, 00:06:14
O   192.168.56.0/30 [110/10] is directly connected, r5-eth2, 00:07:39
C>* 192.168.56.0/30 is directly connected, r5-eth2, 00:07:47
O>* 192.168.67.0/30 [110/20] via 192.168.56.2, r5-eth2, label implicit-null, 00:
06:40
frr-pc#
```

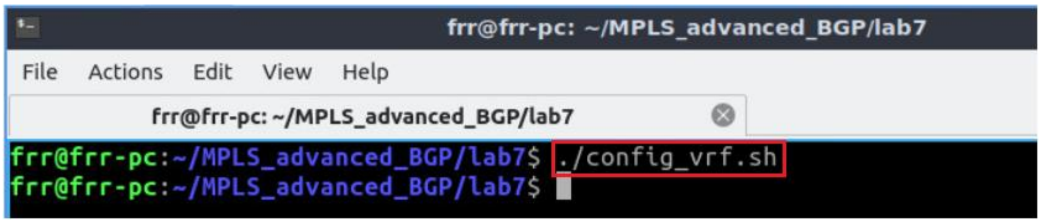
Figure 26. Verifying routing table on router r5.

Consider the figure above. ISP routers are reachable through OSPF routing protocol. On top of that, LDP is running. You will notice all the labels assigned in the routing table.

2.5 Enable VRF operation in ISP routers

Step 1. In order to enable VRF operations in the ISP routers, execute the following script in the Linux terminal.

```
./config_vrf.sh
```

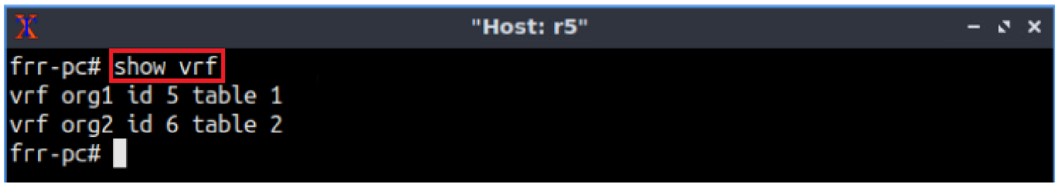

frr@frr-pc: ~/MPLS_advanced_BGP/lab7
File
Actions
Edit
View
Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab7

```
frr@frr-pc:~/MPLS_advanced_BGP/lab7$ ./config_vrf.sh
frr@frr-pc:~/MPLS_advanced_BGP/lab7$
```

Figure 27. Enabling VRF in the ISP routers.

Step 2. In order to verify vrf on router r5 terminal, issue the following command:

```
show vrf
```



```
frr-pc# show vrf
vrf org1 id 5 table 1
vrf org2 id 6 table 2
frr-pc#
```

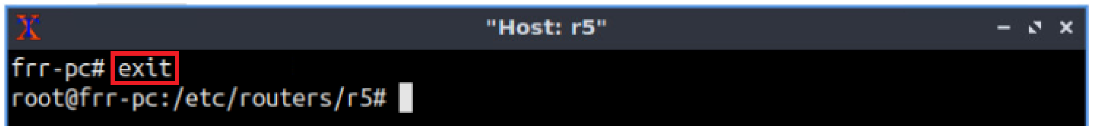
Figure 28. Verifying vrf information on router r5.

3 Configure BGP on PE routers

In this section, you will configure MP-BGP on PE routers to exchange VPNv4 routes.

Step 1. On router r5, type the following command to make exit from vtysh:

```
exit
```

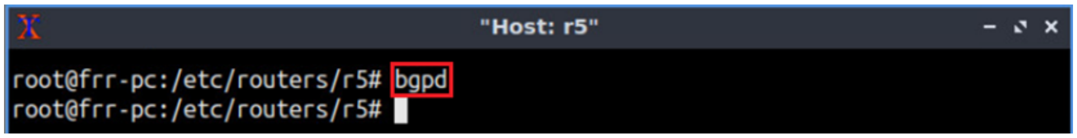


```
frr-pc# exit
root@frr-pc:/etc/routers/r5#
```

Figure 29. Exiting from vtysh.

Step 2. Type the following command in router r5 terminal to enable BGP daemon.

```
bgpd
```

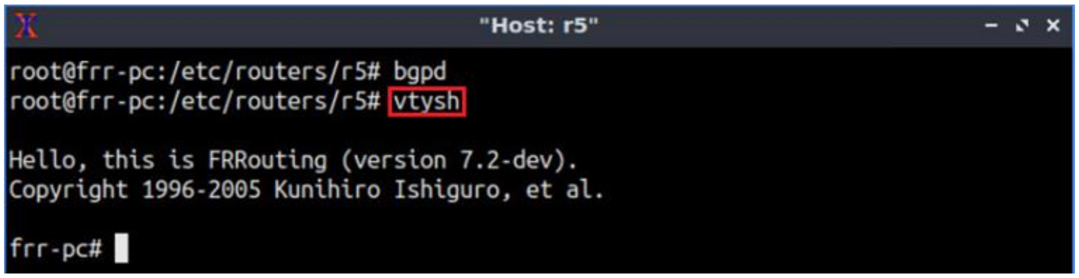


```
root@frr-pc:/etc/routers/r5# bgpd
root@frr-pc:/etc/routers/r5#
```

Figure 30. Starting BGP daemon.

Step 3. In order to enter to router r5 terminal, issue the following command:

```
vtysh
```

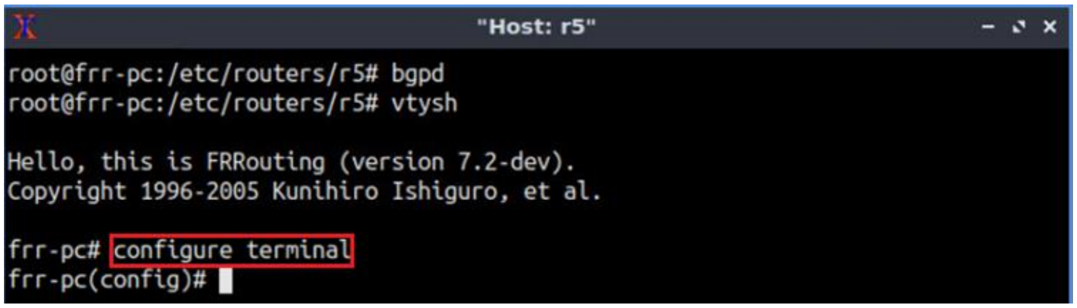


```
root@frr-pc:/etc/routers/r5# bgpd
root@frr-pc:/etc/routers/r5# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 31. Starting vtysh on router r5.

Step 4. To enable router r5 configuration mode, issue the following command:

```
configure terminal
```

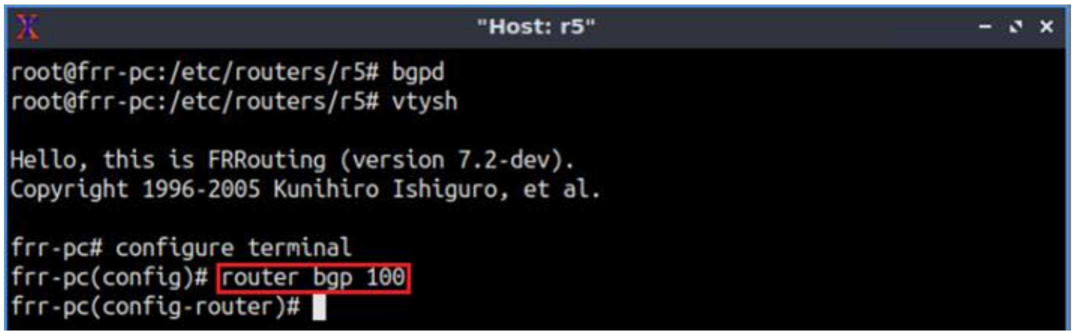


```
root@frr-pc:/etc/routers/r5# bgpd
root@frr-pc:/etc/routers/r5# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)#
```

Figure 32. Enabling configuration mode in router r5.

Step 5. The BGP Autonomous System Number (ASN) assigned for router r5 is 100. In order to configure BGP, type the following command:

```
router bgp 100
```

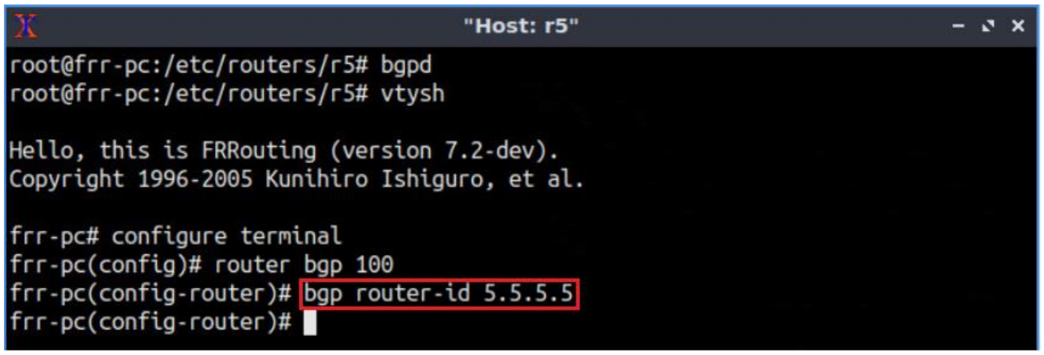


```
root@frr-pc:/etc/routers/r5# bgpd
root@frr-pc:/etc/routers/r5# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)#
```

Figure 33. Configuring BGP in router r5.

Step 6. Assign a router ID to router r5 by issuing the following command.

```
bgp router-id 5.5.5.5
```

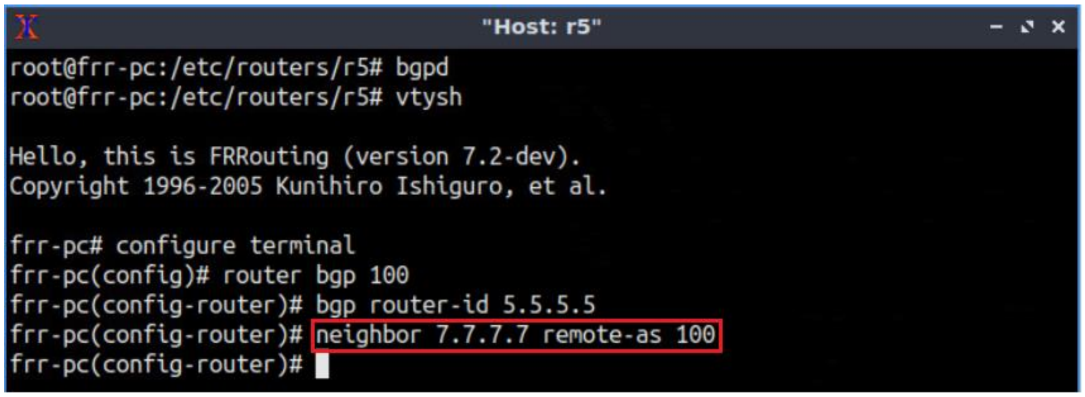


```
"Host: r5"
root@frr-pc:/etc/routers/r5# bgpd
root@frr-pc:/etc/routers/r5# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 5.5.5.5
frr-pc(config-router)#
```

Figure 34. Assigning a router ID in router r5.

Step 7. To configure a BGP neighbor to router r5 (AS 100), type the command shown below. This command specifies the neighbor IP address (7.7.7.7) and ASN of the remote BGP peer (AS 100).

```
neighbor 7.7.7.7 remote-as 100
```

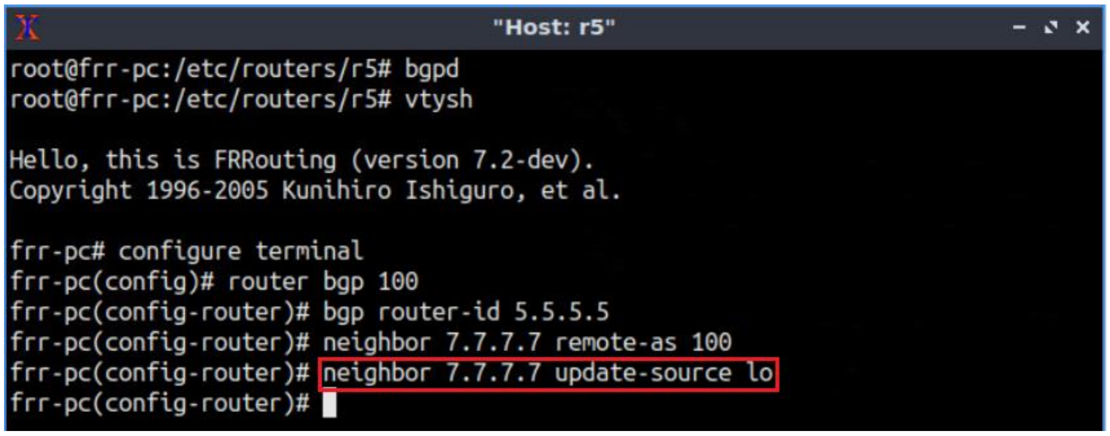


```
"Host: r5"
root@frr-pc:/etc/routers/r5# bgpd
root@frr-pc:/etc/routers/r5# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 5.5.5.5
frr-pc(config-router)# neighbor 7.7.7.7 remote-as 100
frr-pc(config-router)#
```

Figure 35. Assigning BGP neighbor to router r5.

Step 8. Type the following command to assign *lo* as the source IP in router r5.

```
neighbor 7.7.7.7 update source-lo
```

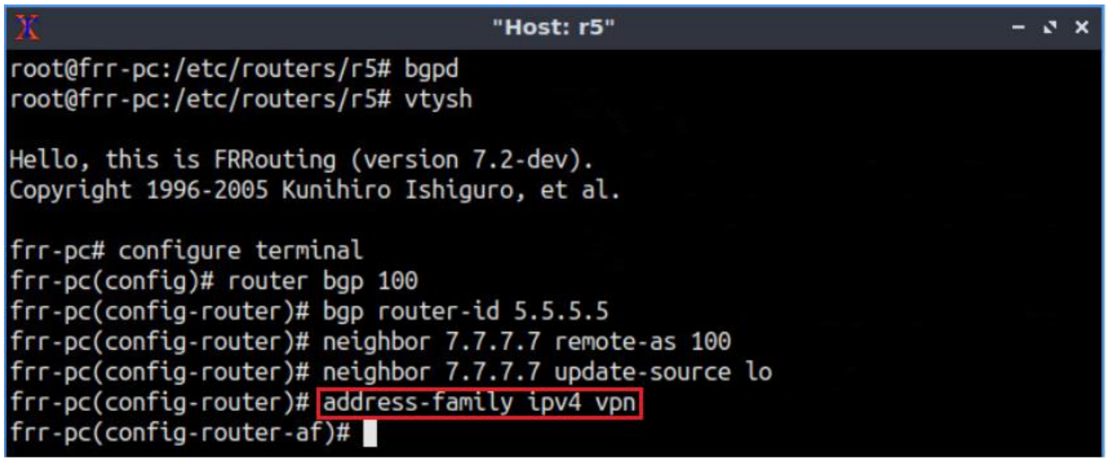


```
"Host: r5"
root@frr-pc:/etc/routers/r5# bgpd
root@frr-pc:/etc/routers/r5# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 5.5.5.5
frr-pc(config-router)# neighbor 7.7.7.7 remote-as 100
frr-pc(config-router)# neighbor 7.7.7.7 update-source lo
frr-pc(config-router)#
```

Figure 36. Assigning loopback as source IP for the neighbor 7.7.7.7.

Step 9. Type the following command to enter address-family mode where you can configure VPN routing sessions that use standard IPv4 address prefixes.

```
address-family ipv4 vpn
```

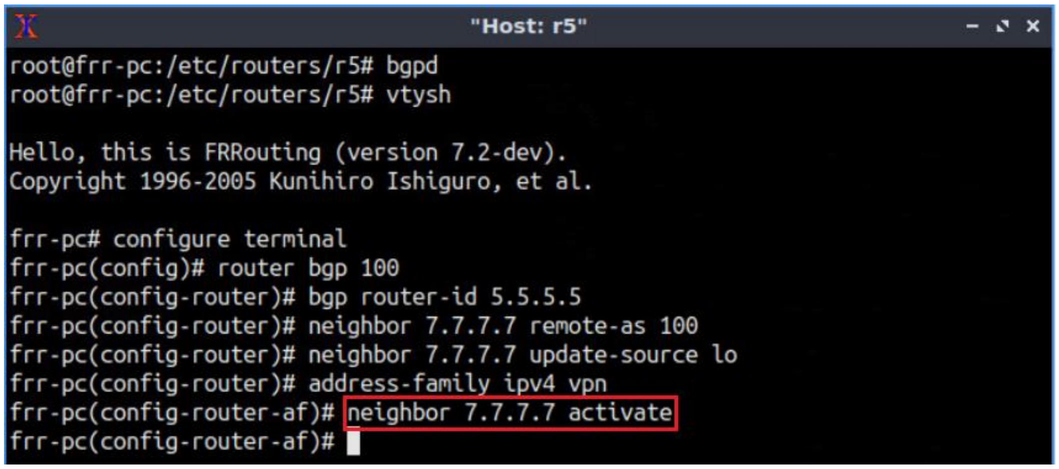


```
root@frr-pc:/etc/routers/r5# bgpd
root@frr-pc:/etc/routers/r5# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 5.5.5.5
frr-pc(config-router)# neighbor 7.7.7.7 remote-as 100
frr-pc(config-router)# neighbor 7.7.7.7 update-source lo
frr-pc(config-router)# address-family ipv4 vpn
frr-pc(config-router-af)#
```

Figure 37. Enabling address-family IPv4 configuration mode in router r5.

Step 10. Type the following command to activate the neighbor 7.7.7.7 so that this neighbor is used to exchange VPNv4 routes with router r5.

```
neighbor 7.7.7.7 activate
```

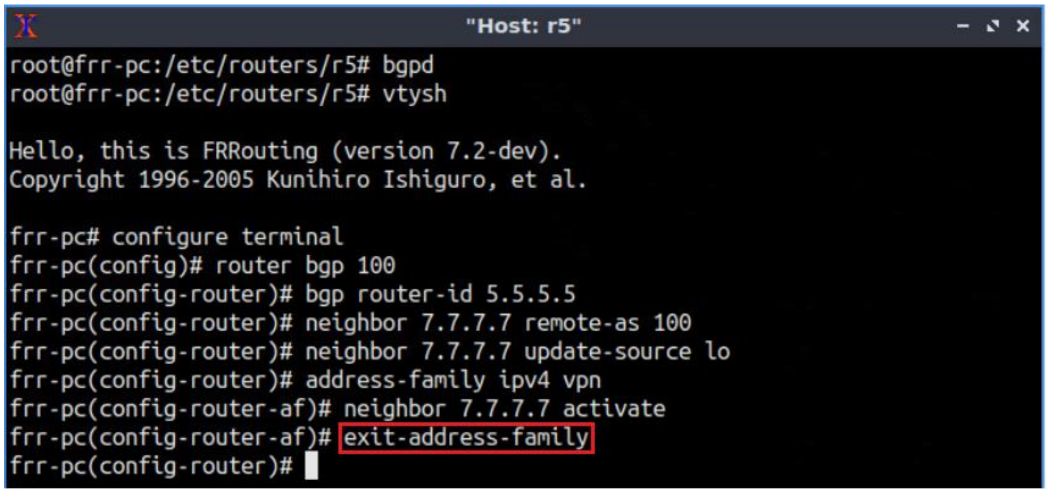


```
root@frr-pc:/etc/routers/r5# bgpd
root@frr-pc:/etc/routers/r5# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 5.5.5.5
frr-pc(config-router)# neighbor 7.7.7.7 remote-as 100
frr-pc(config-router)# neighbor 7.7.7.7 update-source lo
frr-pc(config-router)# address-family ipv4 vpn
frr-pc(config-router-af)# neighbor 7.7.7.7 activate
frr-pc(config-router-af)#
```

Figure 38. Activating IBGP neighbor to advertise IPv4 network.

Step 11. Type the following command to exit from the address-family mode.

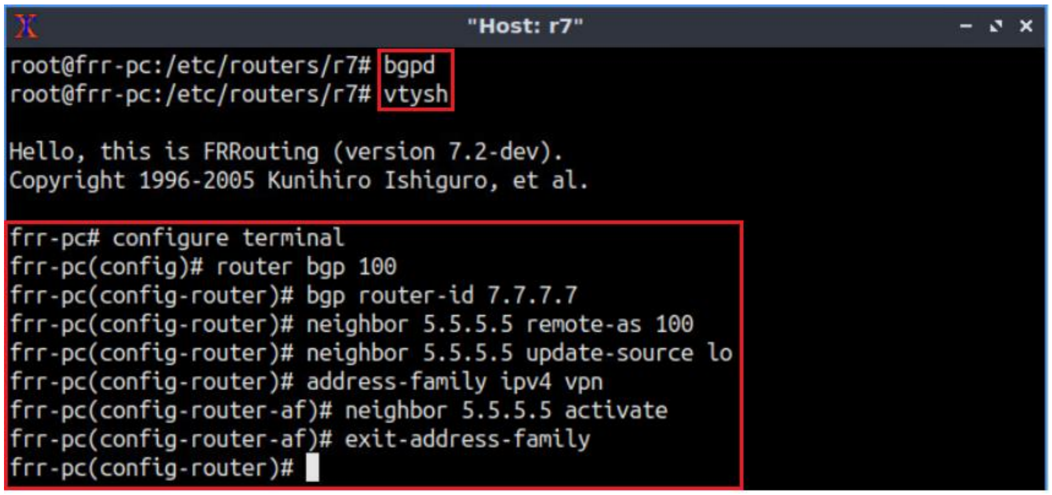
```
exit-address-family
```



```
root@frr-pc:/etc/routers/r5# bgpd
root@frr-pc:/etc/routers/r5# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 5.5.5.5
frr-pc(config-router)# neighbor 7.7.7.7 remote-as 100
frr-pc(config-router)# neighbor 7.7.7.7 update-source lo
frr-pc(config-router)# address-family ipv4 vpn
frr-pc(config-router-af)# neighbor 7.7.7.7 activate
frr-pc(config-router-af)# exit-address-family
frr-pc(config-router)#
```

Figure 39. Exiting from address-family mode.

Step 12. Router r7 is configured similarly to router r5 but, with different metrics in order to establish IBGP peering with router r5. All the steps are summarized in the following figure.



```
root@frr-pc:/etc/routers/r7# bgpd
root@frr-pc:/etc/routers/r7# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 7.7.7.7
frr-pc(config-router)# neighbor 5.5.5.5 remote-as 100
frr-pc(config-router)# neighbor 5.5.5.5 update-source lo
frr-pc(config-router)# address-family ipv4 vpn
frr-pc(config-router-af)# neighbor 5.5.5.5 activate
frr-pc(config-router-af)# exit-address-family
frr-pc(config-router)#
```

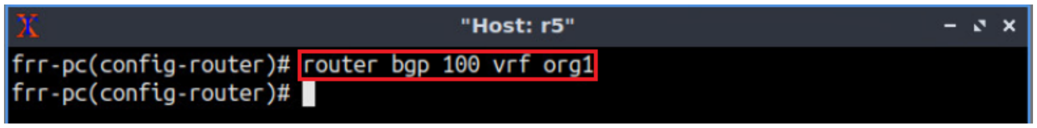
Figure 40. Configuring BGP in router r7.

4 Advertise VPN routes

At this point, both VRF and BGP are running on PE routers. In this section, you will advertise VPN information through BGP.

Step 1. Type the following command to enter into BGP mode for VRF org1.

```
router bgp 100 vrf org1
```

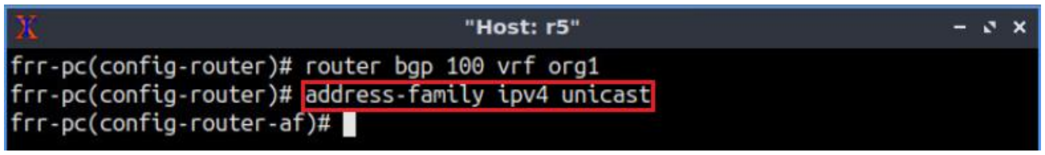


```
frr-pc(config-router)# router bgp 100 vrf org1
frr-pc(config-router)#
```

Figure 41. Configuring BGP in router r5.

Step 2. Type the following command to enter address-family mode where you can configure routing sessions that use standard IPv4 address prefixes.

```
address-family ipv4 unicast
```

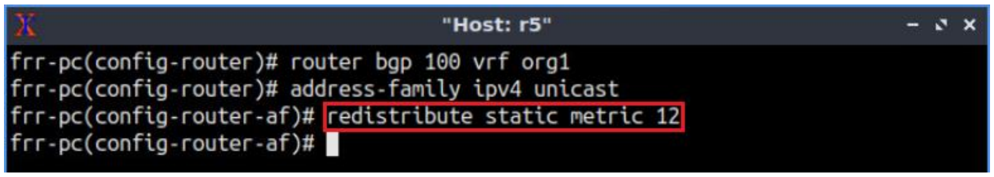


```
"Host: r5"
frr-pc(config-router)# router bgp 100 vrf org1
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)#
```

Figure 42. Enabling address-family IPv4 configuration mode in router r5.

Step 3. Type the following command to redistribute all the static routes on router r5.

```
redistribute static metric 12
```



```
"Host: r5"
frr-pc(config-router)# router bgp 100 vrf org1
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# redistribute static metric 12
frr-pc(config-router-af)#
```

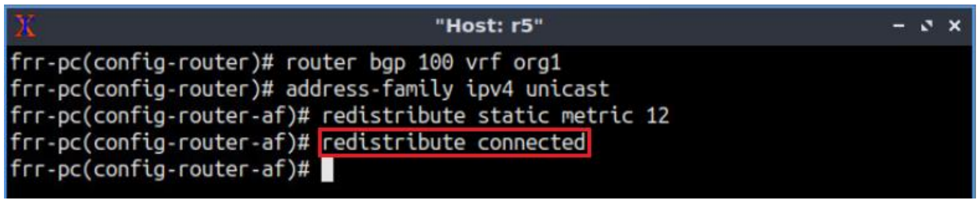
Figure 43. Redistribute static routes in router r5.

You can choose any number within the range (0-16777214) in order to configure the default metric. For the purpose of this lab, you will specify the metric `12`.

Consider the figure above. BGP will advertise all the static routes to neighbor routers so that campus routers are reachable through BGP.

Step 4. Type the following command to redistribute all the connected networks on router r5.

```
redistribute connected
```

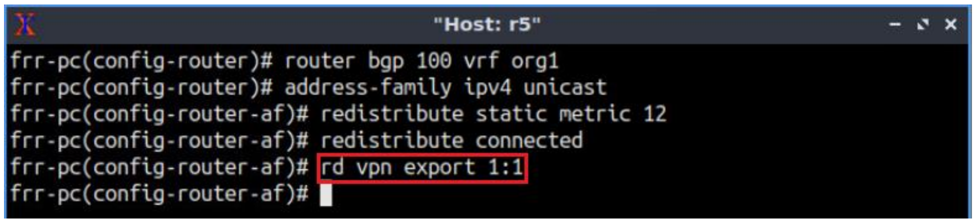


```
"Host: r5"
frr-pc(config-router)# router bgp 100 vrf org1
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# redistribute static metric 12
frr-pc(config-router-af)# redistribute connected
frr-pc(config-router-af)#
```

Figure 44. Redistribute connected routes in router r5.

Step 5. Type the following command to assign RD value for VRF org1 in order to make the IP address (192.168.1.0/24) unique.

```
rd vpn export 1:1
```

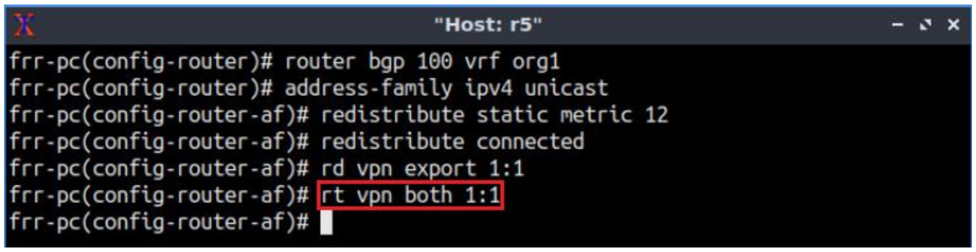


```
"Host: r5"
frr-pc(config-router)# router bgp 100 vrf org1
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# redistribute static metric 12
frr-pc(config-router-af)# redistribute connected
frr-pc(config-router-af)# rd vpn export 1:1
frr-pc(config-router-af)#
```

Figure 45. Assigning RD value for VRF org1.

Step 6. Type the following command to assign RT value for VRF org1 in order to transfer routes within the VPN.

```
rt vpn both 1:1
```



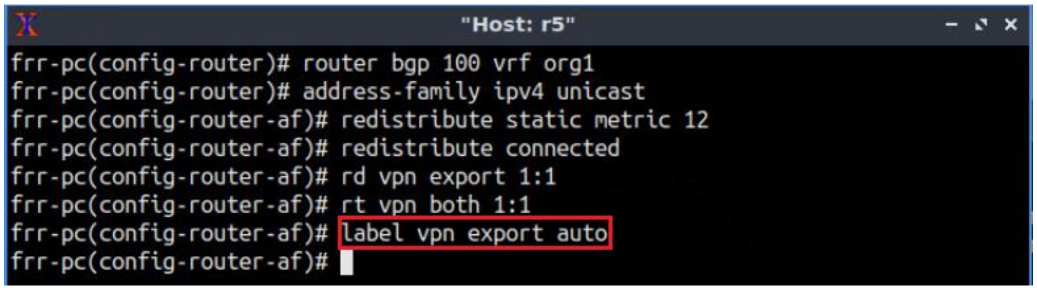
```
"Host: r5"
frr-pc(config-router)# router bgp 100 vrf org1
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# redistribute static metric 12
frr-pc(config-router-af)# redistribute connected
frr-pc(config-router-af)# rd vpn export 1:1
frr-pc(config-router-af)# rt vpn both 1:1
frr-pc(config-router-af)#
```

Figure 46. Assigning RT value for VRF org1.

The keyword *both* is used to assign same RT value for export and import. While exporting, router r5 will assign RT value 1:1 and the VRF having the same RT value on router r7 will accept the route (import).

Step 7. Type the following command to enable VPN processing on router r5.

```
label vpn export auto
```

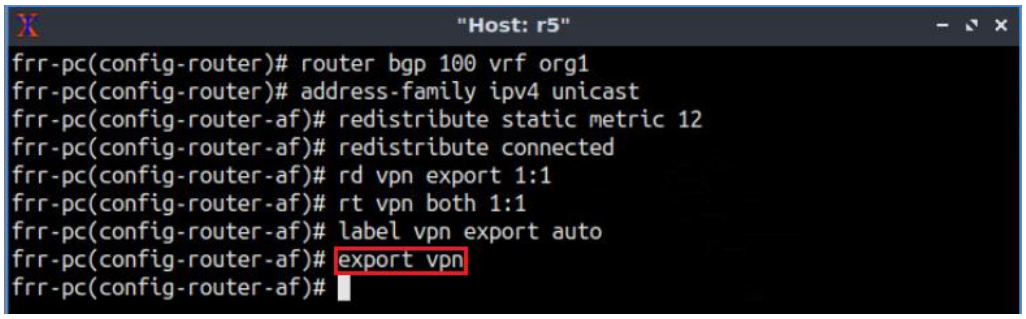


```
"Host: r5"
frr-pc(config-router)# router bgp 100 vrf org1
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# redistribute static metric 12
frr-pc(config-router-af)# redistribute connected
frr-pc(config-router-af)# rd vpn export 1:1
frr-pc(config-router-af)# rt vpn both 1:1
frr-pc(config-router-af)# label vpn export auto
frr-pc(config-router-af)#
```

Figure 47. Enabling VPN in router r5.

Step 8. Type the following command in order to enable VPN export on router r5.

```
export vpn
```

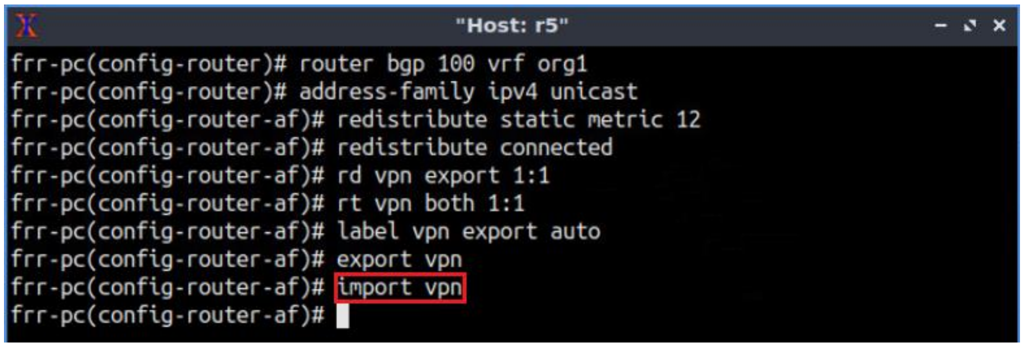


```
"Host: r5"
frr-pc(config-router)# router bgp 100 vrf org1
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# redistribute static metric 12
frr-pc(config-router-af)# redistribute connected
frr-pc(config-router-af)# rd vpn export 1:1
frr-pc(config-router-af)# rt vpn both 1:1
frr-pc(config-router-af)# label vpn export auto
frr-pc(config-router-af)# export vpn
frr-pc(config-router-af)#
```

Figure 48. Enabling VPN export in router r5.

Step 9. Type the following command in order to enable VPN import on router r5.

```
import vpn
```

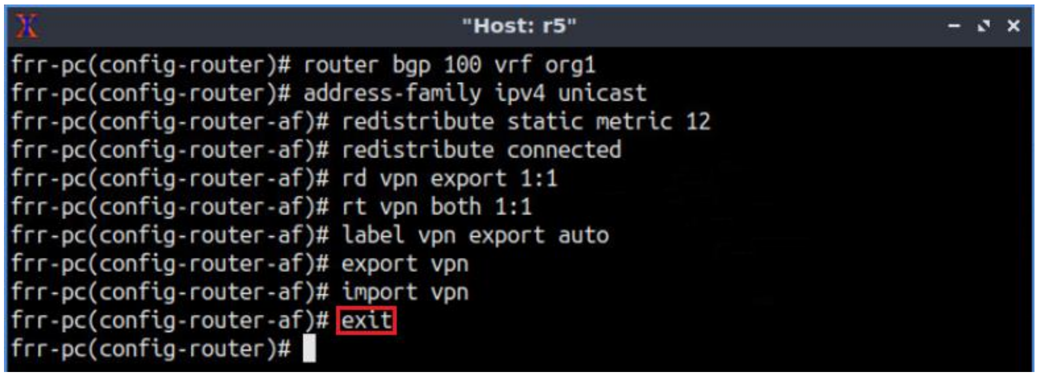


```
"Host: r5"
frr-pc(config-router)# router bgp 100 vrf org1
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# redistribute static metric 12
frr-pc(config-router-af)# redistribute connected
frr-pc(config-router-af)# rd vpn export 1:1
frr-pc(config-router-af)# rt vpn both 1:1
frr-pc(config-router-af)# label vpn export auto
frr-pc(config-router-af)# export vpn
frr-pc(config-router-af)# import vpn
frr-pc(config-router-af)#
```

Figure 49. Enabling VPN import in router r5.

Step 10. Type the following command to exit from BGP configuration mode.

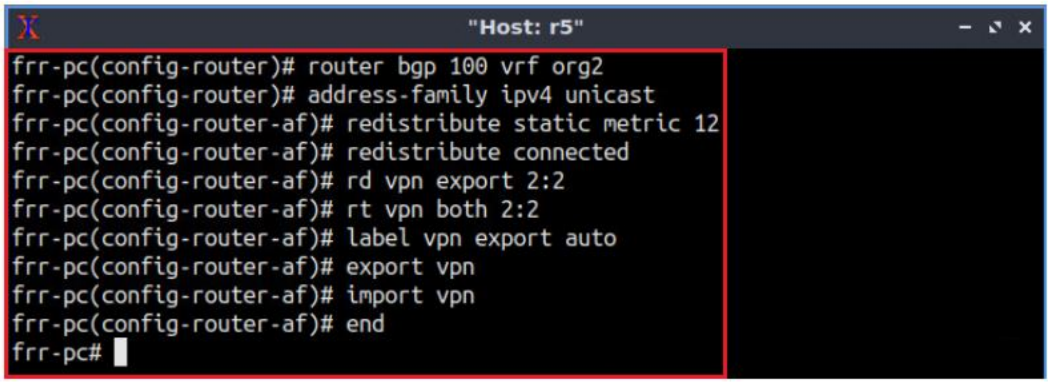
```
exit
```



```
"Host: r5"
frr-pc(config-router)# router bgp 100 vrf org1
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# redistribute static metric 12
frr-pc(config-router-af)# redistribute connected
frr-pc(config-router-af)# rd vpn export 1:1
frr-pc(config-router-af)# rt vpn both 1:1
frr-pc(config-router-af)# label vpn export auto
frr-pc(config-router-af)# export vpn
frr-pc(config-router-af)# import vpn
frr-pc(config-router-af)# exit
frr-pc(config-router)#
```

Figure 50. Exiting from BGP configuration mode.

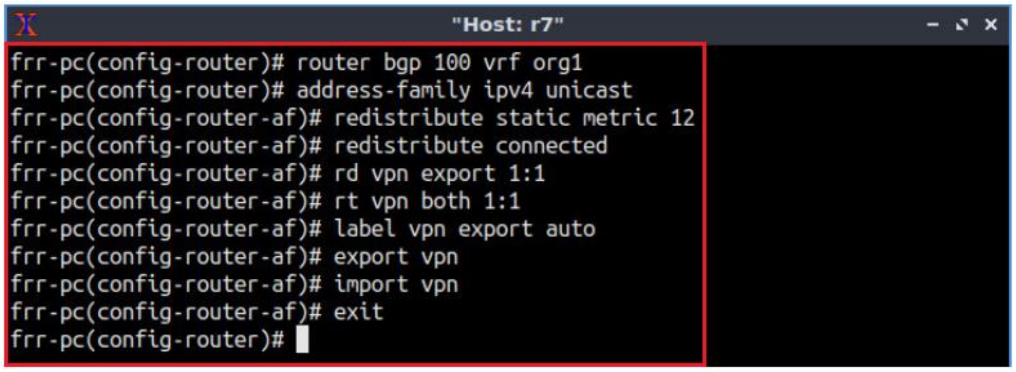
Step 11. Follow steps 1 to 10 in order to inject VRF org2 and static routes in router r5. The value of RD and RT will be different from VRF org1. All the steps are summarized in the following figure.



```
"Host: r5"
frr-pc(config-router)# router bgp 100 vrf org2
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# redistribute static metric 12
frr-pc(config-router-af)# redistribute connected
frr-pc(config-router-af)# rd vpn export 2:2
frr-pc(config-router-af)# rt vpn both 2:2
frr-pc(config-router-af)# label vpn export auto
frr-pc(config-router-af)# export vpn
frr-pc(config-router-af)# import vpn
frr-pc(config-router-af)# end
frr-pc#
```

Figure 51. Inject VRF and static routes in router r5.

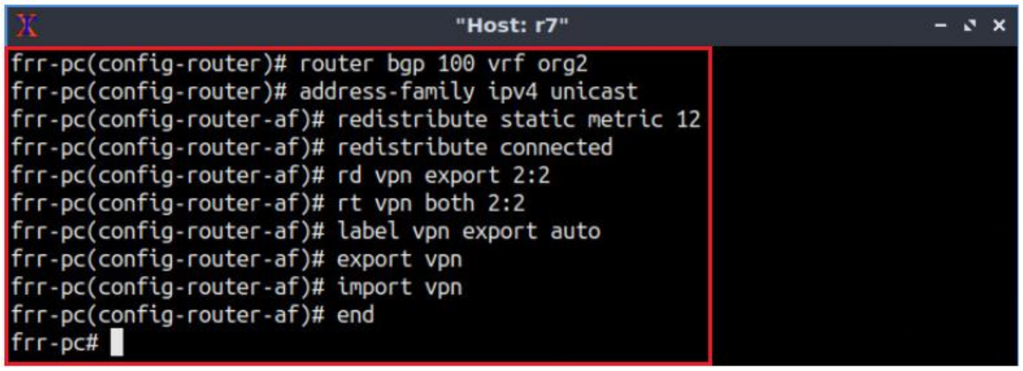
Step 12. Follow steps 1 to 10 in order to inject VRF org1 and static routes in router r7. All the steps are summarized in the following figure.



```
"Host: r7"
frr-pc(config-router)# router bgp 100 vrf org1
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# redistribute static metric 12
frr-pc(config-router-af)# redistribute connected
frr-pc(config-router-af)# rd vpn export 1:1
frr-pc(config-router-af)# rt vpn both 1:1
frr-pc(config-router-af)# label vpn export auto
frr-pc(config-router-af)# export vpn
frr-pc(config-router-af)# import vpn
frr-pc(config-router-af)# exit
frr-pc(config-router)#
```

Figure 52. Inject VRF and static routes in router r7.

Step 13. Follow steps 1 to 10 in order to inject VRF org2 and static routes in router r7. The value of RD and RT will be different from VRF org1. All the steps are summarized in the following figure.



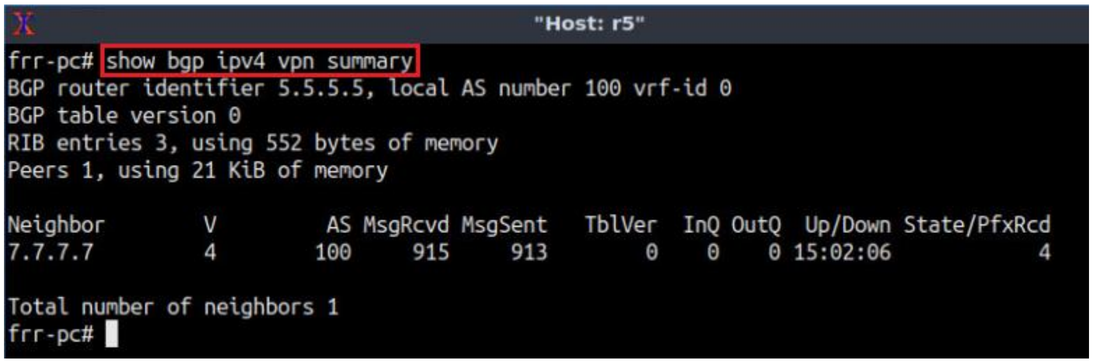
```
"Host: r7"
frr-pc(config-router)# router bgp 100 vrf org2
frr-pc(config-router)# address-family ipv4 unicast
frr-pc(config-router-af)# redistribute static metric 12
frr-pc(config-router-af)# redistribute connected
frr-pc(config-router-af)# rd vpn export 2:2
frr-pc(config-router-af)# rt vpn both 2:2
frr-pc(config-router-af)# label vpn export auto
frr-pc(config-router-af)# export vpn
frr-pc(config-router-af)# import vpn
frr-pc(config-router-af)# end
frr-pc#
```

Figure 53. Inject VRF and static routes in router r7.

5 Verify configuration

Step 1. Type the following command to verify BGP neighbors in router r5.

```
show bgp ipv4 vpn summary
```



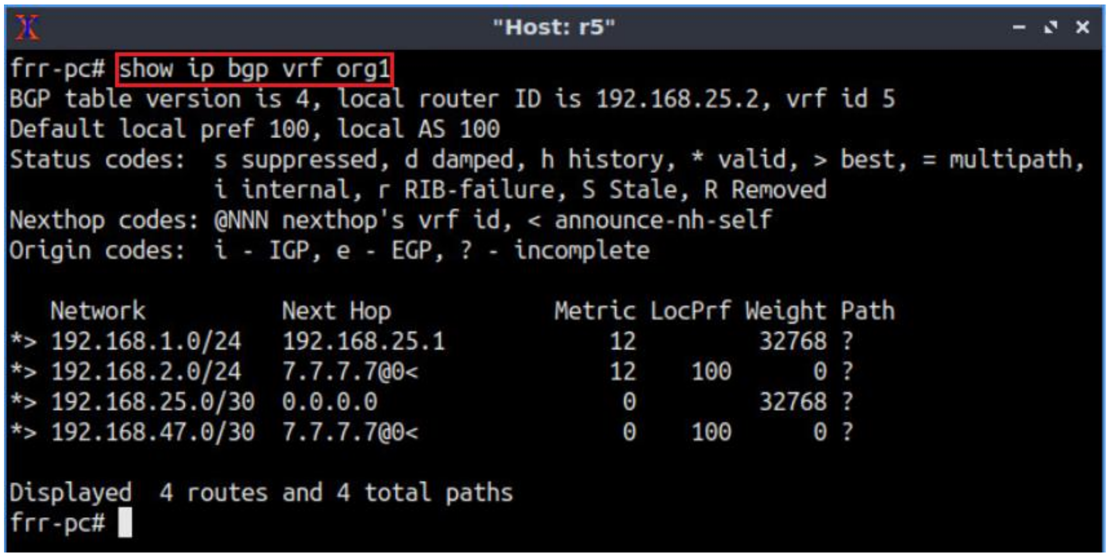
```
frr-pc# show bgp ipv4 vpn summary
BGP router identifier 5.5.5.5, local AS number 100 vrf-id 0
BGP table version 0
RIB entries 3, using 552 bytes of memory
Peers 1, using 21 KiB of memory

Neighbor        V         AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd
7.7.7.7         4        100     915     913        0    0    0 15:02:06            4

Total number of neighbors 1
frr-pc#
```

Figure 54. Verifying BGP neighbors in router r5.

Consider the figure above. The figure shows VPN summary and contains information such as router identifier, AS number and VPN neighbor.

Step 2. In router r5, type the following command to verify BGP table of VRF org1.

```
show ip bgp vrf org1
```



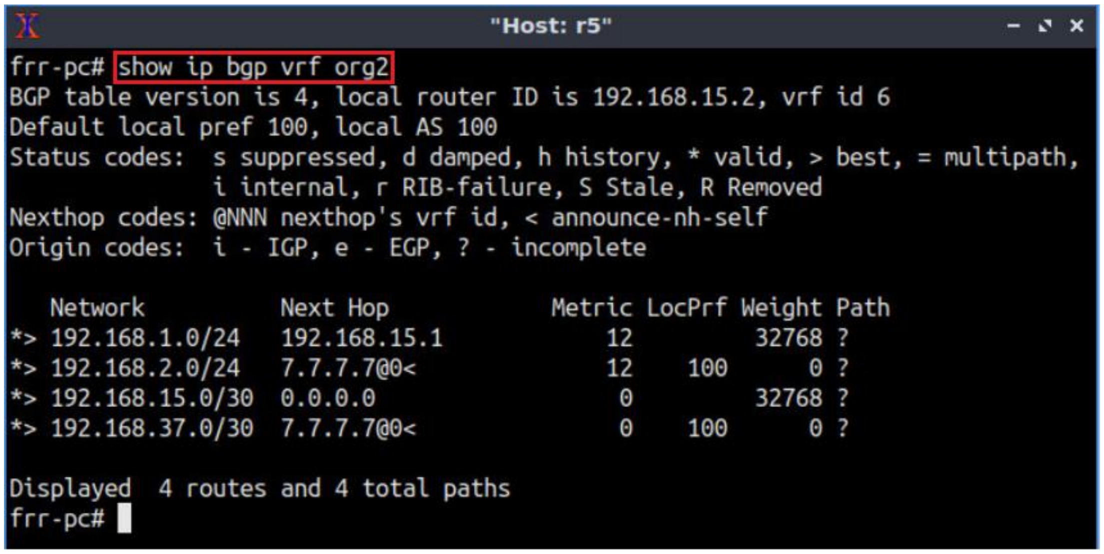
```
frr-pc# show ip bgp vrf org1
BGP table version is 4, local router ID is 192.168.25.2, vrf id 5
Default local pref 100, local AS 100
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 192.168.1.0/24   192.168.25.1            12         32768 ?
*> 192.168.2.0/24   7.7.7.7@0<              12    100      0 ?
*> 192.168.25.0/30  0.0.0.0                  0         32768 ?
*> 192.168.47.0/30  7.7.7.7@0<               0    100      0 ?

Displayed  4 routes and 4 total paths
frr-pc#
```

Figure 55. Verifying BGP table in router r5.

Consider the figure above. The figure shows the networks assigned for org 1.

Step 3. In router r5, type the following command to verify BGP table of VRF org2.

```
show ip bgp vrf org2
```



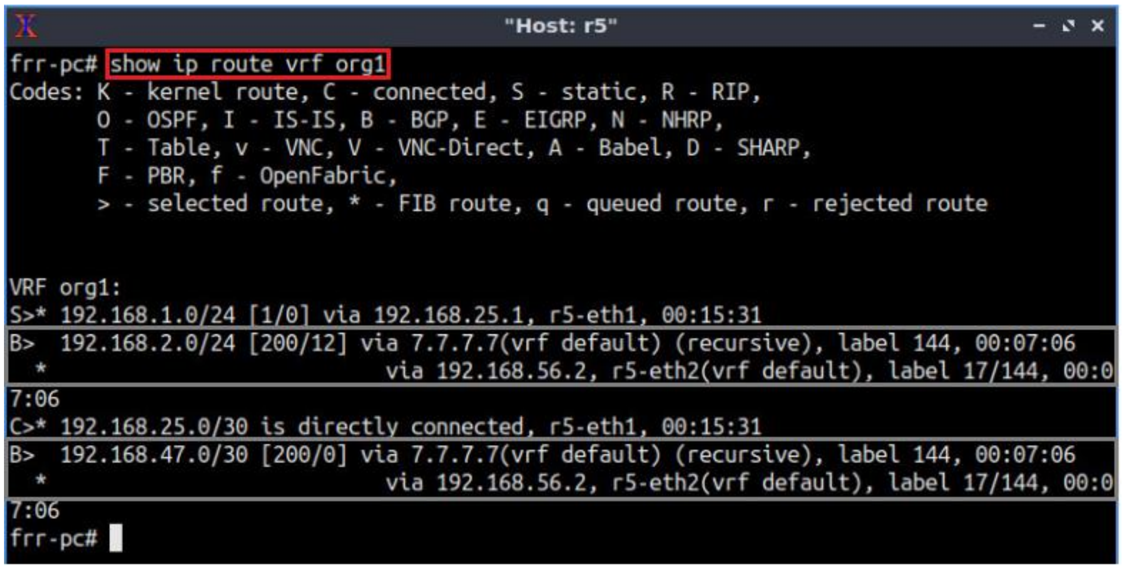
```
"Host: r5"
frr-pc# show ip bgp vrf org2
BGP table version is 4, local router ID is 192.168.15.2, vrf id 6
Default local pref 100, local AS 100
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 192.168.1.0/24   192.168.15.1            12         32768 ?
*> 192.168.2.0/24   7.7.7.7@0<              12    100      0 ?
*> 192.168.15.0/30  0.0.0.0                  0         32768 ?
*> 192.168.37.0/30  7.7.7.7@0<               0    100      0 ?

Displayed  4 routes and 4 total paths
frr-pc#
```

Figure 56. Verifying BGP table in router r5.

Consider the figure above. The figure shows the networks assigned for org 2.

Step 4. In router r5, type the following command to verify routing table of VRF org1.

```
show ip route vrf org1
```



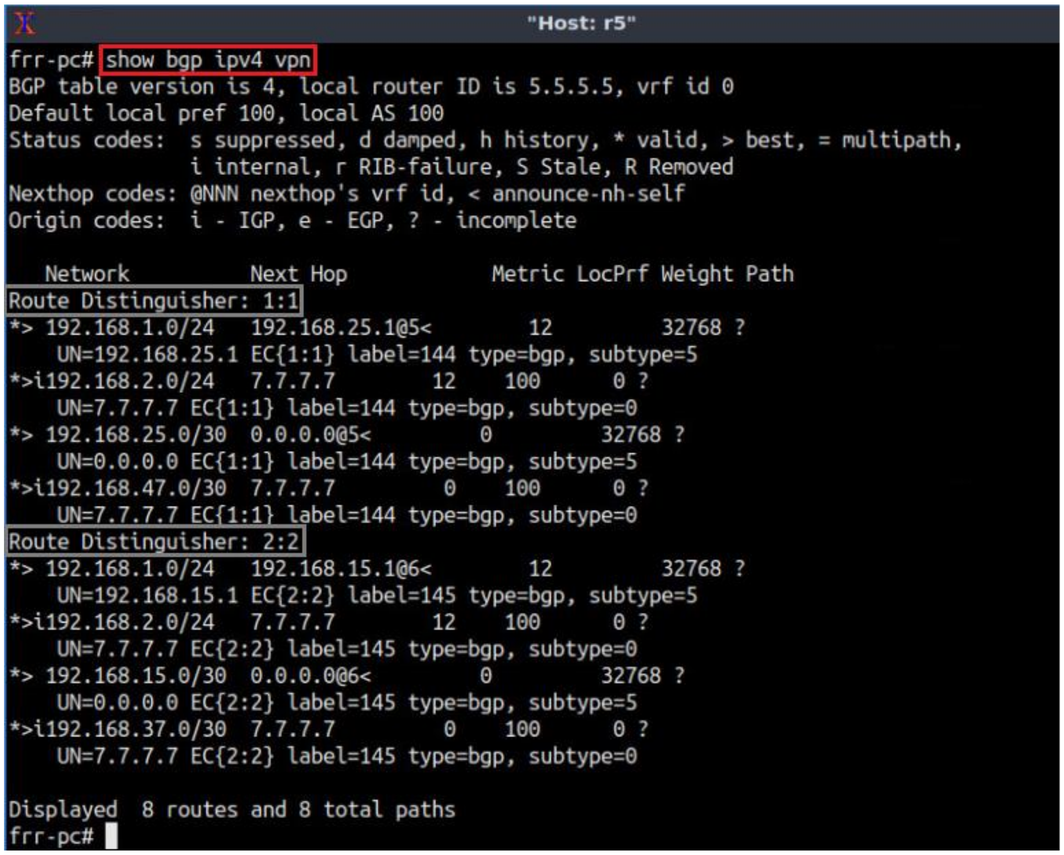
```
"Host: r5"
frr-pc# show ip route vrf org1
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route


VRF org1:
S>* 192.168.1.0/24 [1/0] via 192.168.25.1, r5-eth1, 00:15:31
B>  192.168.2.0/24 [200/12] via 7.7.7.7(vrf default) (recursive), label 144, 00:07:06
  *                          via 192.168.56.2, r5-eth2(vrf default), label 17/144, 00:0
7:06
C>* 192.168.25.0/30 is directly connected, r5-eth1, 00:15:31
B>  192.168.47.0/30 [200/0] via 7.7.7.7(vrf default) (recursive), label 144, 00:07:06
  *                          via 192.168.56.2, r5-eth2(vrf default), label 17/144, 00:0
7:06
frr-pc#
```

Figure 57. Verifying routing table in router r5.

Consider the figure above. The figure shows two entries learned through BGP. The routes were learned from router r7 (7.7.7.7). In order to communicate with the network 192.168.2.0/24, the packet will be forwarded to router r6 (192.168.56.2).

Step 5. Type the following command to verify VPN information in router r5.

```
show bgp ipv4 vpn
```



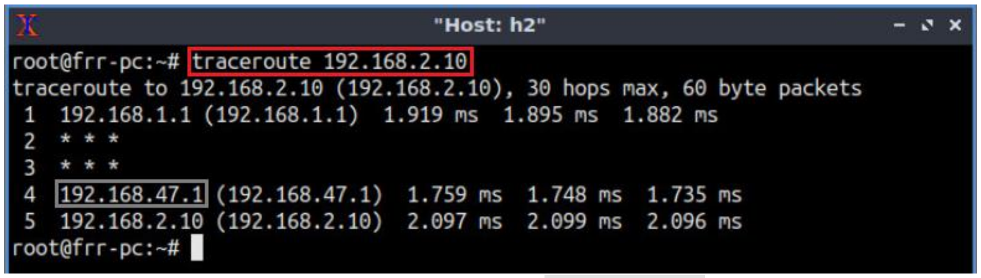
```
"Host: r5"
frr-pc# show bgp ipv4 vpn
BGP table version is 4, local router ID is 5.5.5.5, vrf id 0
Default local pref 100, local AS 100
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 1:1
*> 192.168.1.0/24   192.168.25.1@5<         12         32768 ?
    UN=192.168.25.1 EC{1:1} label=144 type=bgp, subtype=5
*>i192.168.2.0/24   7.7.7.7          12    100      0 ?
    UN=7.7.7.7 EC{1:1} label=144 type=bgp, subtype=0
*> 192.168.25.0/30  0.0.0.0@5<          0         32768 ?
    UN=0.0.0.0 EC{1:1} label=144 type=bgp, subtype=5
*>i192.168.47.0/30  7.7.7.7           0    100      0 ?
    UN=7.7.7.7 EC{1:1} label=144 type=bgp, subtype=0
Route Distinguisher: 2:2
*> 192.168.1.0/24   192.168.15.1@6<         12         32768 ?
    UN=192.168.15.1 EC{2:2} label=145 type=bgp, subtype=5
*>i192.168.2.0/24   7.7.7.7          12    100      0 ?
    UN=7.7.7.7 EC{2:2} label=145 type=bgp, subtype=0
*> 192.168.15.0/30  0.0.0.0@6<          0         32768 ?
    UN=0.0.0.0 EC{2:2} label=145 type=bgp, subtype=5
*>i192.168.37.0/30  7.7.7.7           0    100      0 ?
    UN=7.7.7.7 EC{2:2} label=145 type=bgp, subtype=0

Displayed  8 routes and 8 total paths
frr-pc#
```

Figure 58. Verifying VPN information in router r5.

Consider the figure above. The figure shows VPN information such as RD value, RT value, and MPLS label for org 1 and org 2. You will notice two IBGP entries for the network 192.168.2.0/24 and the nexthop is router r7 (7.7.7.7). For org 1, RT value is 1:1 and MPLS label value is 144 whereas RT and MPLS label values are 2:2 and 145, respectively for org 2.

Step 6. On host h2 terminal, perform a connectivity test between host h2 and host h4 by issuing the command shown below.

```
traceroute 192.168.2.10
```



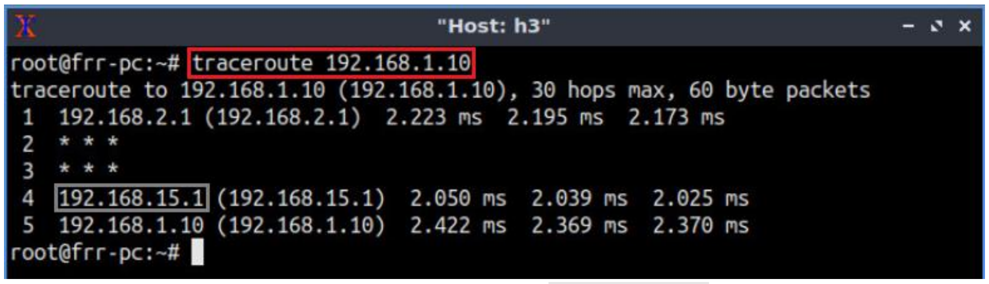
```
"Host: h2"
root@frr-pc:~# traceroute 192.168.2.10
traceroute to 192.168.2.10 (192.168.2.10), 30 hops max, 60 byte packets
 1  192.168.1.1 (192.168.1.1)  1.919 ms  1.895 ms  1.882 ms
 2  * * *
 3  * * *
 4  192.168.47.1 (192.168.47.1)  1.759 ms  1.748 ms  1.735 ms
 5  192.168.2.10 (192.168.2.10)  2.097 ms  2.099 ms  2.096 ms
root@frr-pc:~#
```

Figure 59. Connectivity test using `traceroute` command.

Consider the figure above. You can verify that host h2 is communicating with the host connected to router r4 (192.168.47.1) as the host h4 belongs to the same VRF instance (org1).

Step 7. On host h3 terminal, perform a connectivity between host h3 and host h1 by issuing the command shown below.

```
traceroute 192.168.1.10
```



```
"Host: h3"
root@frr-pc:~# traceroute 192.168.1.10
traceroute to 192.168.1.10 (192.168.1.10), 30 hops max, 60 byte packets
 1  192.168.2.1 (192.168.2.1)  2.223 ms  2.195 ms  2.173 ms
 2  * * *
 3  * * *
 4  192.168.15.1 (192.168.15.1)  2.050 ms  2.039 ms  2.025 ms
 5  192.168.1.10 (192.168.1.10)  2.422 ms  2.369 ms  2.370 ms
root@frr-pc:~#
```

Figure 60. Connectivity test using `traceroute` command.

Consider the figure above. You can verify that host h3 is communicating with the host connected to router r1 (192.168.15.1) as the host h1 belongs to the same VRF instance (org2).

This concludes Lab 7. Stop the emulation and then exit out of MiniEdit.

References

1. Techopedia, "*Layer 3 VPN (L3VPN)*", [Online]. Available https://www.techopedia.com/definition/30757/layer-3-vpn-l3vpn
2. Cisco, "*Virtual route forwarding design guide*", 2008, [Online]. Available: https://www.cisco.com/c/en/us/td/docs/voice_ip_comm/cucme/vrf/design/guide/vrfDesignGuide.html
3. Luc De Ghein, "*MPLS fundamentals*", 1st Edition, Pearson. 2006.
4. CCIE Blog, "*Difference between the RD and RT*", [Online]. Available: https://ccieblog.co.uk/mpls/difference-between-the-rd-and-rt
5. Juniper, "*MPLS VPN overview*", 2019, [Online]. Available: https://www.juniper.net/documentation/en_US/junos/topics/concept/mpls-security-vpn-overview.html

UNIVERSITY OF
SOUTH CAROLINA

MPLS AND ADVANCED BGP TOPICS

Lab 8: Ethernet VPN (EVPN) using MP-BGP

Document Version: 03-04-2021

Award 1829698
"CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers"

Contents

Overview

This lab presents Ethernet Virtual Private Network (EVPN), a technology that provides virtual multipoint bridged connectivity between different Layer 2 domains over an IP network. This lab aims to configure EVPN where two sites of campus network attempt to be in the same network and EVPN is responsible for transporting Layer 2 MAC and Layer 3 IP information through Multiprotocol BGP (MP-BGP).

Objectives

By the end of this lab, students should be able to:

1. Understand the concept of EVPN.
2. Apply VXLAN configuration to isolate traffic.
3. Configure EVPN in routers.
4. Verify the configuration by inspecting packets using *Wireshark*.

Lab settings

The information in Table 1 provides the credentials to access Client machine.

Table 1**.** Credentials to access Client machine.

Device	**Account**	**Password**
Client	admin	password

Lab roadmap

This lab is organized as follows:

1. Section 1: Introduction.
2. Section 2: Lab topology.
3. Section 3: Verifying OPSF configuration.
4. Section 4: Configuring VXLAN.
5. Section 5: Configuring IBGP in routers.
6. Section 6: Configuring EVPN.
7. Section 7: Verifying configuration.

1 Introduction

1.1 VXLAN architecture

Overlay networks in general, and Internet protocol IP overlay networks in particular, are gaining popularity for providing virtual machine (VM) mobility over Layer 3 (L3) networks. VXLAN is a technique for providing a Layer 2 (L2) overlay on an L3 network[1]. In particular, VXLAN is used to address the need for overlay networks within virtualized data centers accommodating multiple tenants. Each overlay is unique within the tenant domain and is known as a VXLAN segment. The communication is restricted just between VMs within the same VXLAN segment. Each VXLAN segment is identified by a 24-bit segment ID, called the VXLAN Network Identifier (VNI). This allows up to 16 million (2^{24}) VXLAN segments to coexist within the same administrative domain.

Figure 1(a) presents a topology that illustrates how VXLAN segments are transported over a layer 3 network (e.g., IP network). The endpoints of the tunnels are known as the VXLAN Tunnel End Point (VTEP). The VTEP is responsible for encapsulating the layer 2 frames in a VXLAN header and forward that on the IP Network. It also handles the reversal process that consists of de-encapsulating an incoming VXLAN segment and forward the original frame to its corresponding Local Area Network (LAN). Figure 1(b) shows that the VXLAN framework is represented as a tunneling scheme that transports layer 2 frames on top of a layer 3 network. The tunnels are stateless, meaning that each frame is encapsulated according to a set of predefined rules. The end-hosts do not store session information.

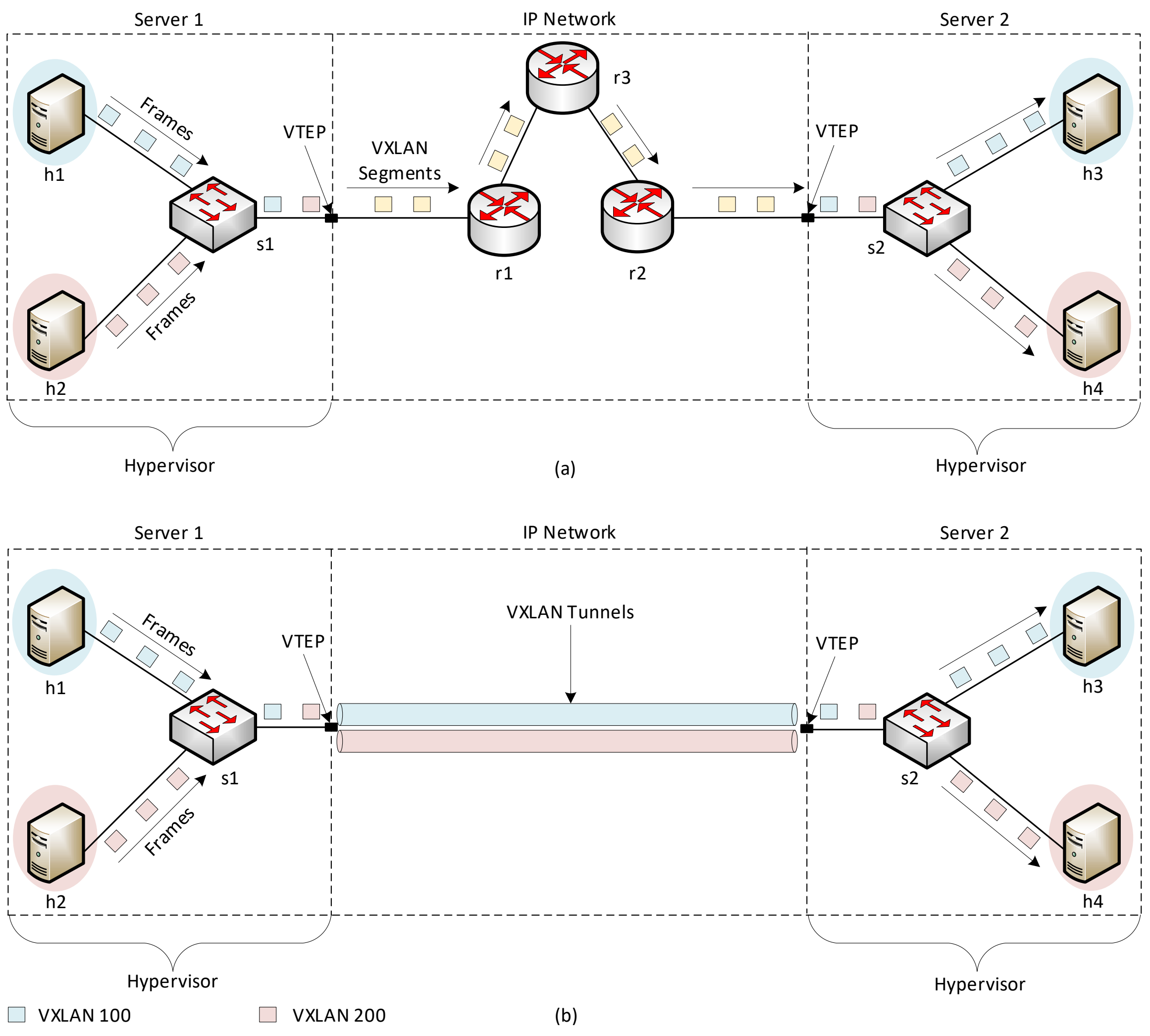


Figure 1. VXLAN overview. (a) VXLAN segments are transported over a layer 3 network (e.g., IP network). (b) The VXLAN framework is represented as a tunneling scheme that transports layer 2 frames on top of layer 3 networks.

1.2 EVPN

EVPN is a technology that provides virtual bridged connectivity between layer 2 domains over an IP network. It is considered as a BGP control plane to advertise both Layer 2 MAC and Layer 3 IP information at the same time. By having the set of MAC and IP information available for forwarding decisions, optimized routing and switching within a network becomes feasible[5]. To provide flexibility, EVPN-VXLAN decouples the underlay network (physical topology) from the overlay network (virtual topology). By using overlays, organizations gain the flexibility of providing Layer 2/Layer 3 connectivity between endpoints across campus and data centers, while maintaining a consistent underlay architecture.

VXLAN does not provide the control plane, and MAC address learning is implemented by traffic flooding on the data plane, resulting in high traffic volumes on networks. When the switch receives a frame dedicated for a particular destination, but that destination does not have an entry in the MAC Address table, the switch has no choice but to flood the frame. The goal of this flood is that the device using the MAC address in the destination of the frame will receive the flood and respond to the message. If that device responds, then the switch can learn their MAC address and map it to the port into which the message arrives. To address this problem, VXLAN uses EVPN as the control plane. EVPN replaces flood-and-learn behavior of traditional VXLAN with the BGP control plane – MAC addresses are propagated as BGP prefixes within the EVPN address family[8].

2 Lab topology

Consider Figure 2. The figure refers to a Wide Area Network (WAN) where two Campus sites are connected to Internet Service Provider (ISP). Routers are communicating with each other over layer 3 using the OSPF routing protocol. Routers r1 and r2 are acting as VXLAN Tunnel End Point (VTEP). Hosts h1 and h2 can interact with each other using VNI 10 directly over layer 2 via the VXLAN tunnel. EVPN is running in routers in order to transport Layer 2 MAC and Layer 3 IP information through Multiprotocol BGP (MP-BGP). Router r3 is acting as a route-reflector which is responsible for advertising all the BGP information to the neighbor routers.

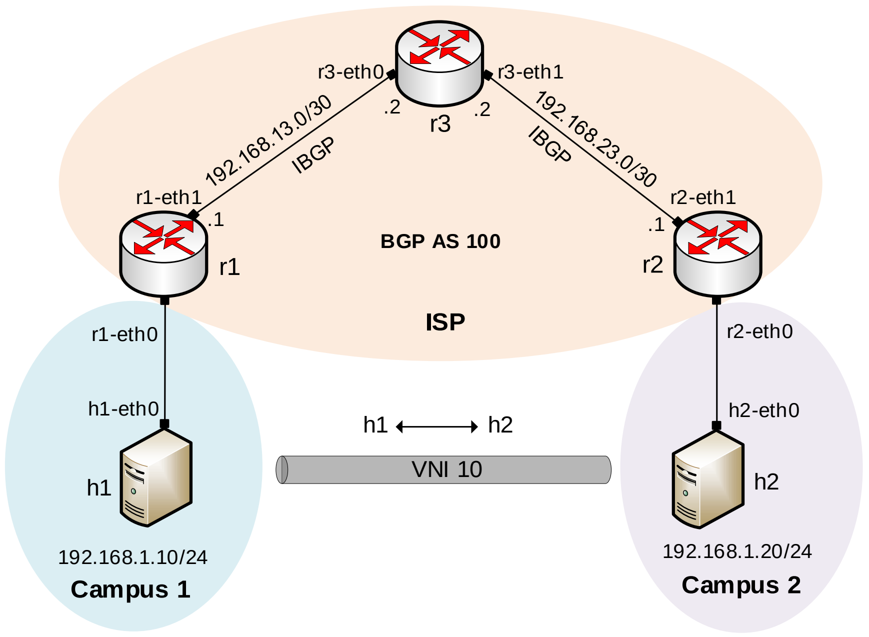


Figure 2. Lab topology.

2.1 Lab settings

Routers and hosts are already configured according to the IP addresses shown in Table 2.

Table 2. Topology information.

Device	Interface	IPV4 Address	Subnet	Default gateway

r1	r1-eth1	192.168.13.1	/30	N/A
	lo	1.1.1.1	/32	N/A
r2	r2-eth1	192.168.23.1	/30	N/A
	lo	1.1.1.2	/32	N/A
r3	r3-eth0	192.168.13.2	/30	N/A
	r3-eth1	192.168.23.2	/30	N/A
	lo	1.1.1.3	/32	N/A
h1	h1-eth0	192.168.1.10	/24	N/A
h2	h2-eth0	192.168.1.20	/24	N/A

2.2 Open topology and load the configuration

Step 1. Start by launching MiniEdit by clicking on desktop's shortcut. When prompted for a password, type `password`.

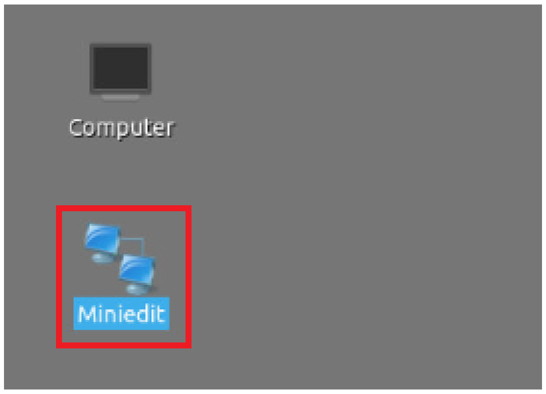


Figure 3. MiniEdit shortcut.

Step 2. On MiniEdit's menu bar, click on *File* then *open* to load the lab's topology. Locate *lab8.mn* topology file in the default directory, */home/frr/MPLS_advanced_BGP/lab8* and click on *Open*.

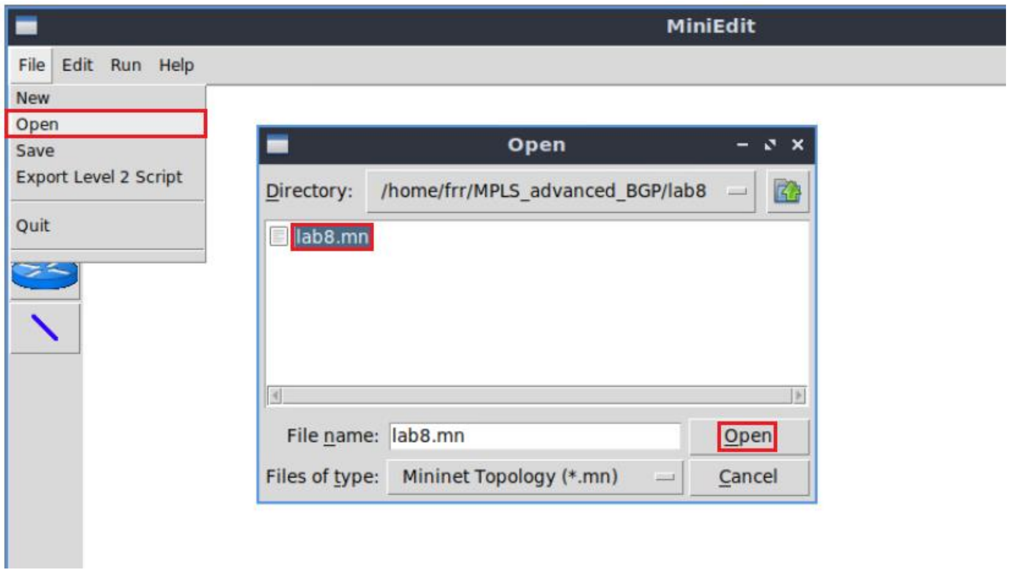


Figure 4. MiniEdit's Open dialog.

At this point, the topology is loaded with all the required network components. You will execute a script that will load the configuration of the routers.

Step 3. Open the Linux terminal.

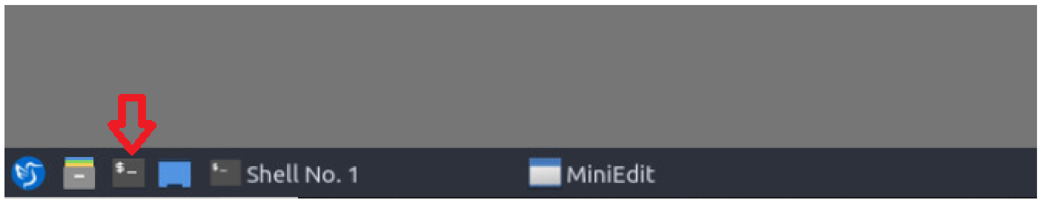
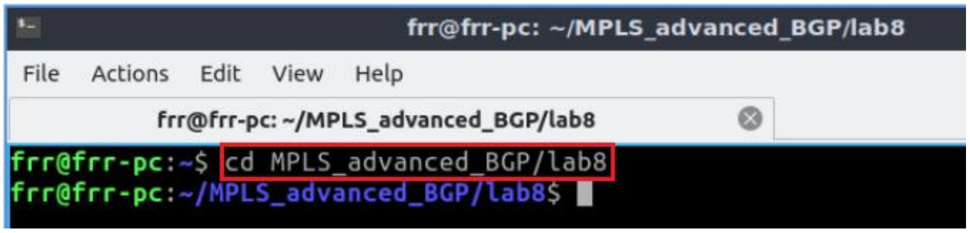


Figure 5. Opening Linux terminal.

Step 4. Click on the Linux terminal and navigate into *MPLS_advanced_BGP/lab8* directory by issuing the following command. This folder contains a configuration file and the script responsible for loading the configuration. The configuration file will assign the IP addresses to the routers' interfaces. The file also contains the OSPF configuration for the routers. The `cd` command is short for change directory followed by an argument that specifies the destination directory.

```
cd MPLS_advanced_BGP/lab8
```

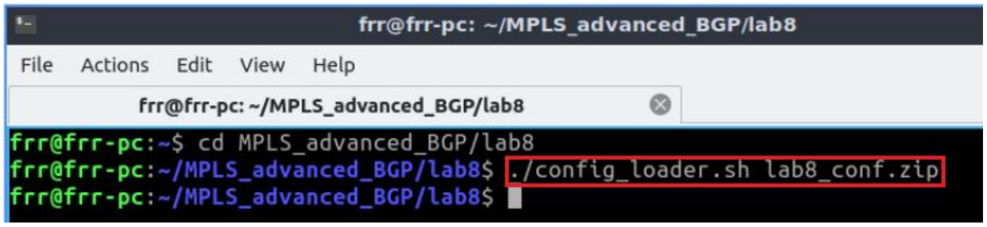


```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab8
frr@frr-pc:~/MPLS_advanced_BGP/lab8$
```

Figure 6. Entering to the *MPLS_advanced_BGP/lab8* directory.

Step 5. To execute the shell script, type the following command. The argument of the program corresponds to the configuration zip file that will be loaded in all the routers.

```
./config_loader.sh lab8_conf.zip
```

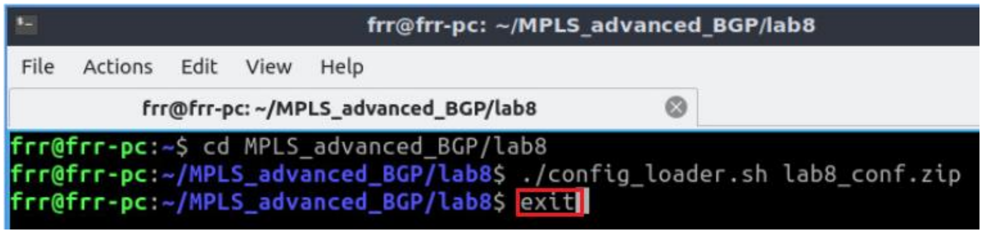

frr@frr-pc: ~/MPLS_advanced_BGP/lab8
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab8

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab8
frr@frr-pc:~/MPLS_advanced_BGP/lab8$ ./config_loader.sh lab8_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab8$
```

Figure 7. Executing the shell script to load the configuration.

Step 6. Type the following command to exit the Linux terminal.

```
exit
```


frr@frr-pc: ~/MPLS_advanced_BGP/lab8
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab8

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab8
frr@frr-pc:~/MPLS_advanced_BGP/lab8$ ./config_loader.sh lab8_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab8$ exit
```

Figure 8. Exiting from the terminal.

Step 7. At this point, the interfaces of hosts h1 and h2 are configured. To proceed with the emulation, click on the *Run* button located on the lower left-hand side.

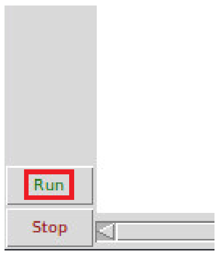


Figure 9. Starting the emulation.

Step 8. Click on Mininet's terminal, i.e., the one launched when MiniEdit was started.

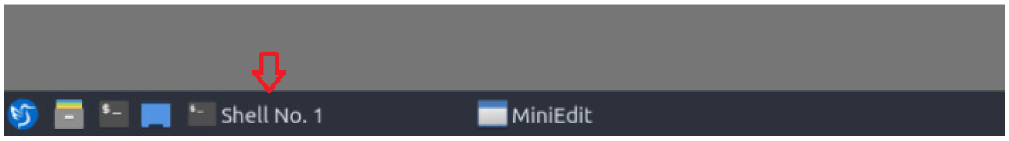
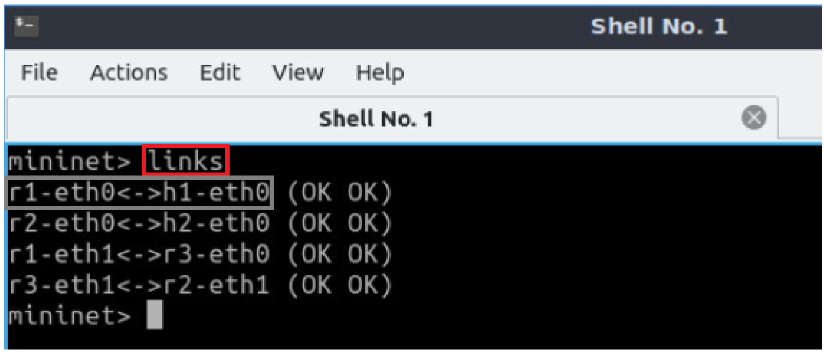


Figure 10. Opening Mininet's terminal.

Step 9. Issue the following command to display the interface names and connections.

```
links
```



```
mininet> links
r1-eth0<->h1-eth0 (OK OK)
r2-eth0<->h2-eth0 (OK OK)
r1-eth1<->r3-eth0 (OK OK)
r3-eth1<->r2-eth1 (OK OK)
mininet>
```

Figure 11. Displaying network interfaces.

In Figure 11, the link displayed within the gray box indicates that interface eth0 of router r1 connects to interface eth0 of host h1 (i.e., *r1-eth0<->h1-eth0*).

2.3 Load zebra daemon and verify configuration

You will verify the IP addresses listed in Table 2 and inspect the routing table of the routers.

Step 1. Hold right-click on host h1 and select *Terminal*. This opens the terminal of host h1 and allows the execution of commands on that host.

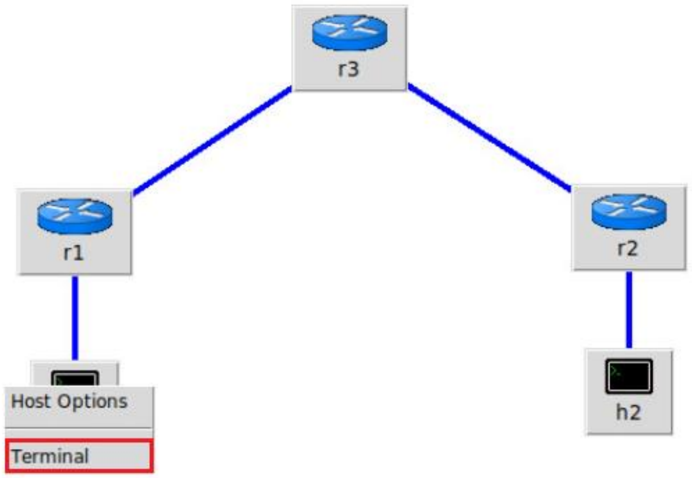
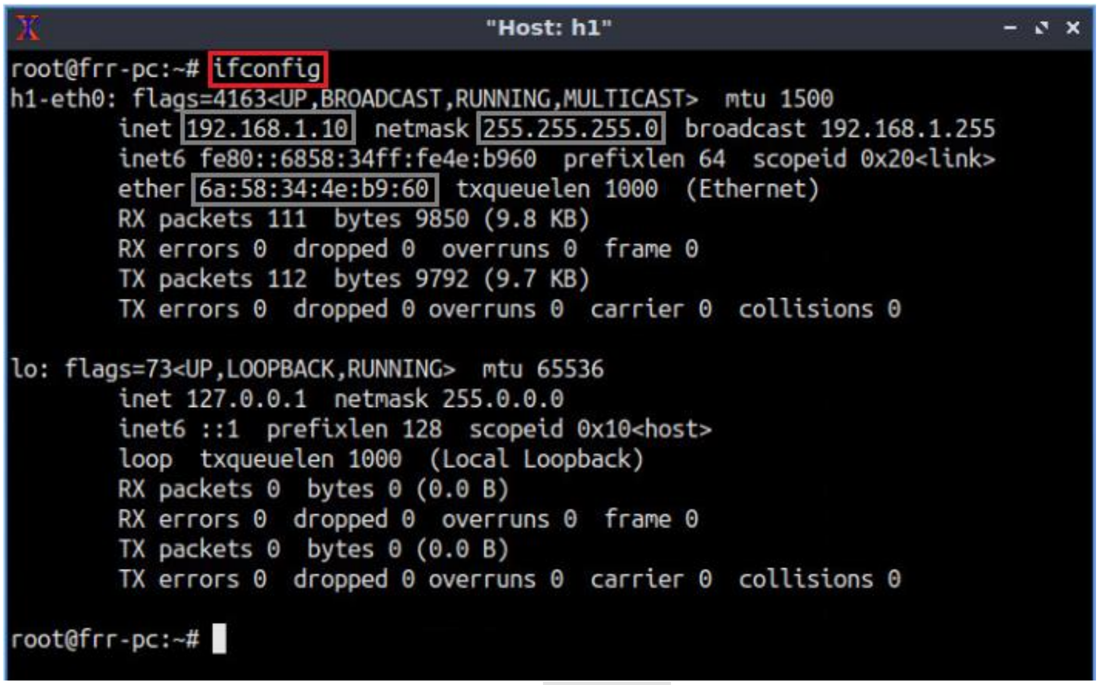


Figure 12. Opening a terminal in host h1.

Step 2. On host h1 terminal, type the command shown below to verify that the IP address was assigned successfully. You will verify host h1 interface, *h1-eth0* configured with the IP address 192.168.1.10, subnet mask 255.255.255.0 and MAC address 6a:58:34:4e:b9:60.

```
ifconfig
```



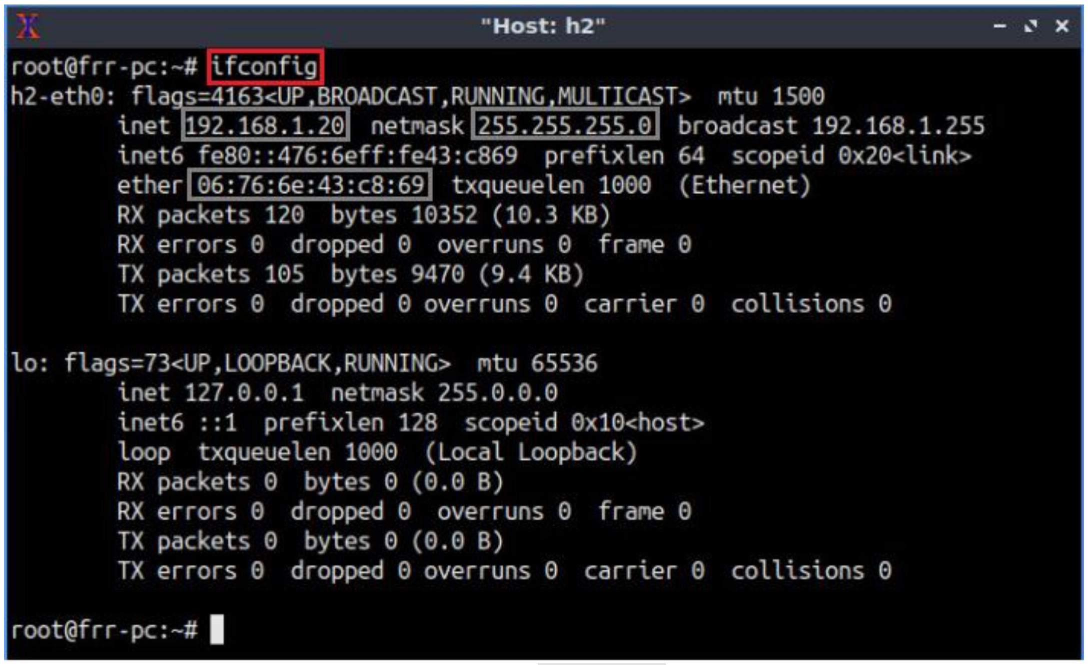
```
root@frr-pc:~# ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.10  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::6858:34ff:fe4e:b960  prefixlen 64  scopeid 0x20<link>
        ether 6a:58:34:4e:b9:60  txqueuelen 1000  (Ethernet)
        RX packets 111  bytes 9850 (9.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 112  bytes 9792 (9.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@frr-pc:~# 
```

Figure 13. Output of `ifconfig` command.

You may notice a different MAC address each time you run the lab since MAC addresses are assigned randomly.

Step 3. In order to verify host h2, proceed similarly by repeating step 1 and step 2 on host terminal. You will verify host h2 interface, *h2-eth0* configured with the IP address 192.168.1.20, subnet mask 255.255.255.0 and MAC address 06:76:6e:43:c8:69.

```
ifconfig
```



```
root@frr-pc:~# ifconfig
h2-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.20  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::476:6eff:fe43:c869  prefixlen 64  scopeid 0x20<link>
        ether 06:76:6e:43:c8:69  txqueuelen 1000  (Ethernet)
        RX packets 120  bytes 10352 (10.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 105  bytes 9470 (9.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@frr-pc:~# 
```

Figure 14. Output of `ifconfig` command.

Step 4. In order to open router r1 terminal, hold right-click on router r1 and select Terminal.

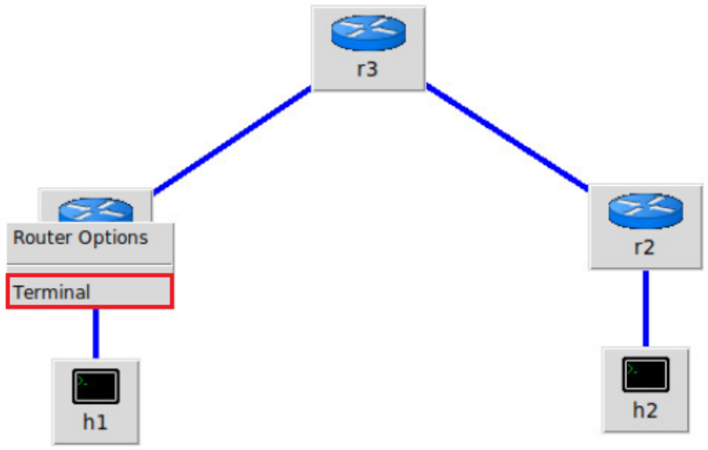
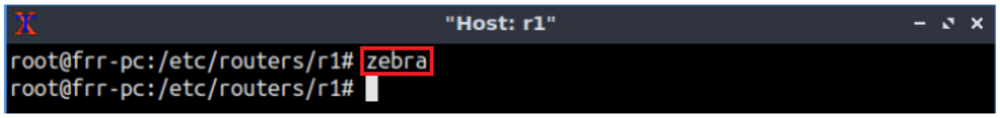


Figure 15. Opening a terminal on router r1.

Step 5. In router r1 terminal, you will start zebra daemon, which is a multi-server routing software that provides TCP/IP based routing protocols. The configuration will not be working if you do not enable zebra daemon initially. In order to start the zebra, type the following command:

```
zebra
```

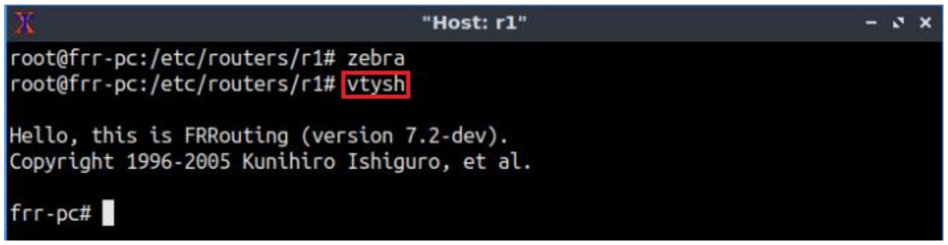


```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# 
```

Figure 16. Starting zebra daemon.

Step 6. After initializing zebra, vtysh should be started in order to provide all the CLI commands defined by the daemons. To proceed, issue the following command:

```
vtysh
```

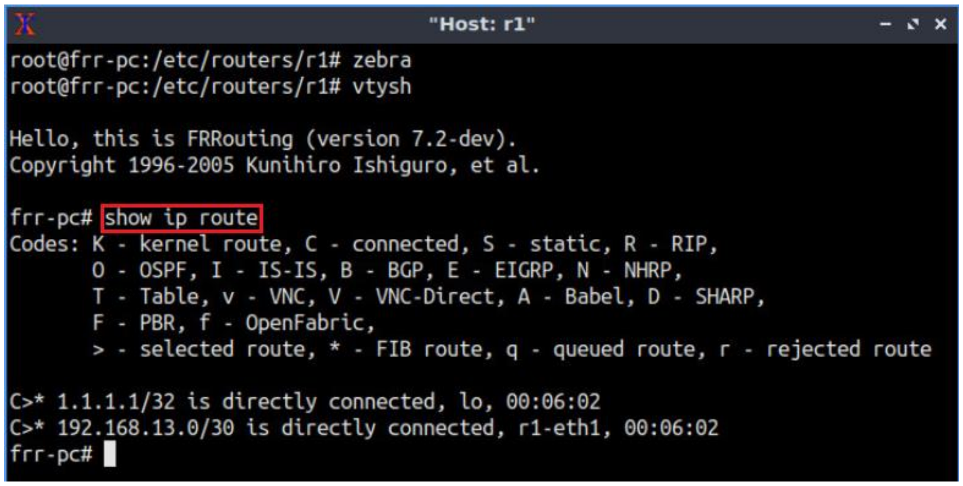


```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# 
```

Figure 17. Starting vtysh on router r1.

Step 7. Type the following command in router r1 terminal to verify the routing table of router r1. It will list all the directly connected networks. The routing table of router r1 does not contain any route to the network attached to routers r2 and r3 (1.1.1.2/32 and 1.1.1.3/32) as there is no routing protocol configured yet.

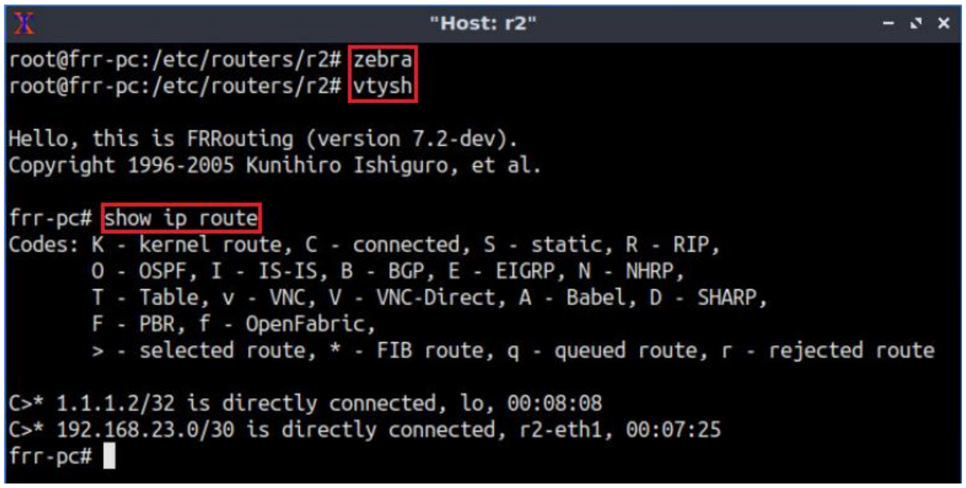
```
show ip route
```



```
"Host: r1"
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 1.1.1.1/32 is directly connected, lo, 00:06:02
C>* 192.168.13.0/30 is directly connected, r1-eth1, 00:06:02
frr-pc#
```

Figure 18. Displaying routing table of router r1.

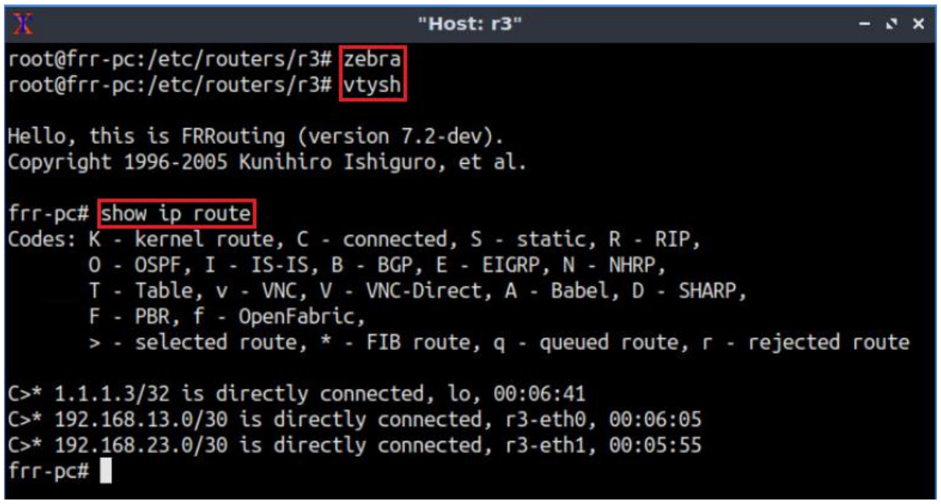
Step 8. Router r2 is configured similarly to router r1 but, with different IP addresses (see Table 2). Those steps are summarized in the following figure. To proceed, in router r2 terminal, issue the commands depicted below. In the end, you will verify all the directly connected networks of router r2.



```
"Host: r2"
root@frr-pc:/etc/routers/r2# zebra
root@frr-pc:/etc/routers/r2# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 1.1.1.2/32 is directly connected, lo, 00:08:08
C>* 192.168.23.0/30 is directly connected, r2-eth1, 00:07:25
frr-pc#
```

Figure 19. Displaying routing table of router r2.

Step 9. Router r3 is configured similarly to router r1 but, with different IP addresses (see Table 2). Those steps are summarized in the following figure. To proceed, in router r3 terminal, issue the commands depicted below. In the end, you will verify all the directly connected networks of router r3.



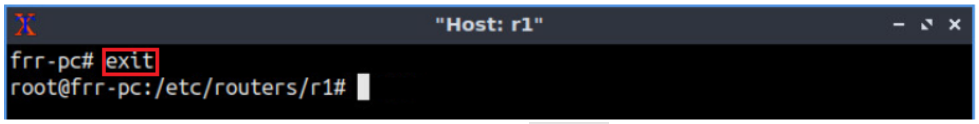
```
"Host: r3"
root@frr-pc:/etc/routers/r3# zebra
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 1.1.1.3/32 is directly connected, lo, 00:06:41
C>* 192.168.13.0/30 is directly connected, r3-eth0, 00:06:05
C>* 192.168.23.0/30 is directly connected, r3-eth1, 00:05:55
frr-pc#
```

Figure 20. Displaying routing table of router r3.

3 Verifying OSPF configuration

Step 1. In router r1, type the following command to exit the vtysh session:

```
exit
```

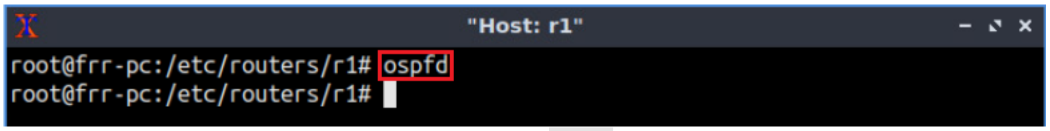


```
"Host: r1"
frr-pc# exit
root@frr-pc:/etc/routers/r1#
```

Figure 21. Exiting the `vtysh` session.

Step 2. Type the following command in router r1 terminal to enable OSPF daemon.

```
ospfd
```

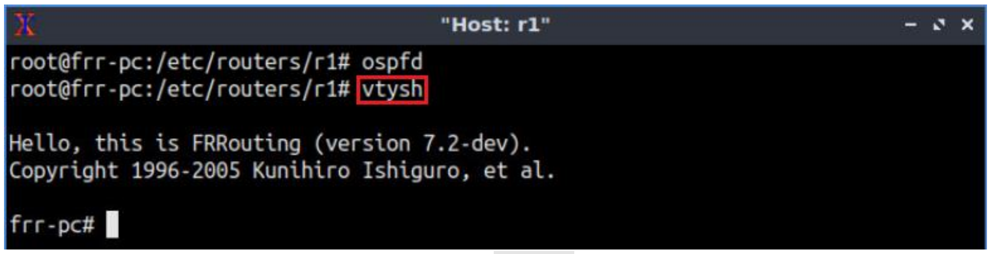


```
"Host: r1"
root@frr-pc:/etc/routers/r1# ospfd
root@frr-pc:/etc/routers/r1#
```

Figure 22. Starting `OSPF` daemon.

Step 3. In order to enter to router r1 terminal, issue the following command:

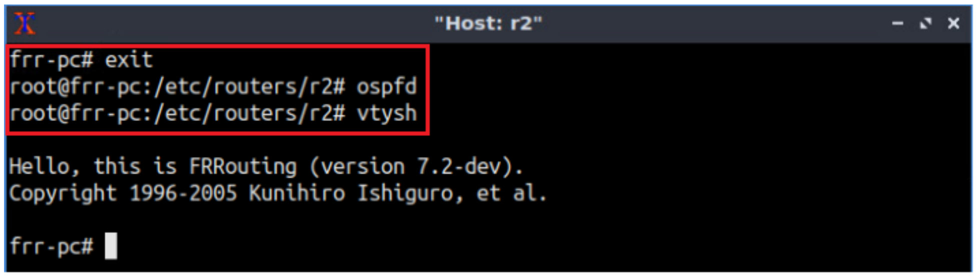
```
vtysh
```



```
"Host: r1"
root@frr-pc:/etc/routers/r1# ospfd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 23. Starting `vtysh` in router r1.

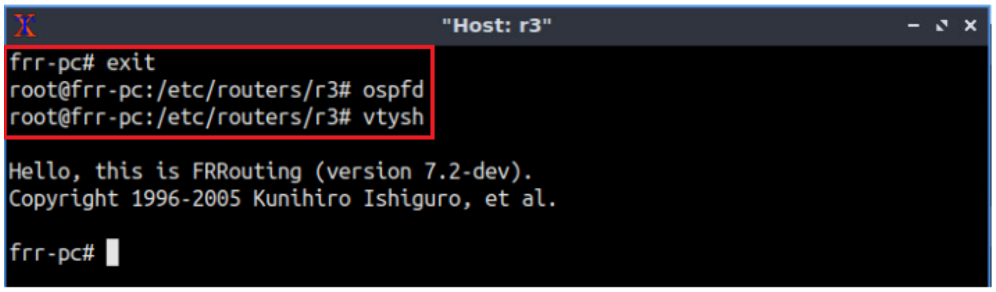
Step 4. Follow from step 1 to step 3 to enable OSPF daemon in router r2. All the steps are summarized in the following figure.



```
"Host: r2"
frr-pc# exit
root@frr-pc:/etc/routers/r2# ospfd
root@frr-pc:/etc/routers/r2# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 24. Verifying the routing table of router r2.

Step 5. Follow from step 1 to step 3 to enable OSPF daemon in router r3. All the steps are summarized in the following figure.

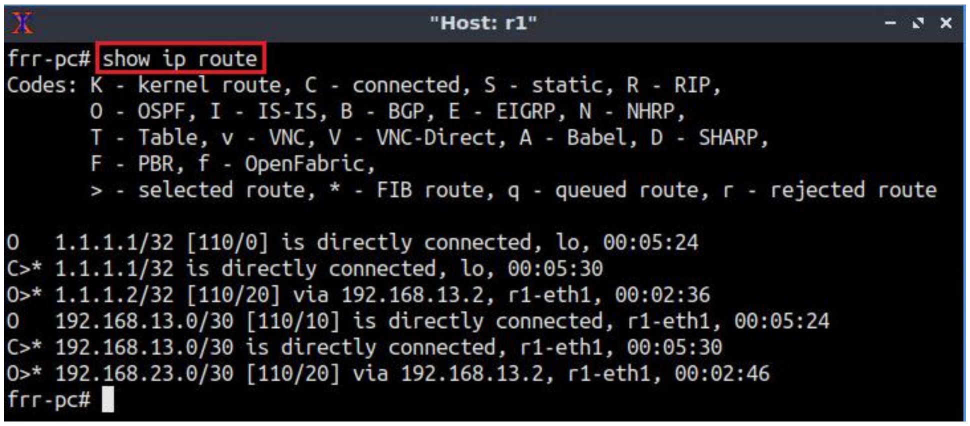


```
"Host: r3"
frr-pc# exit
root@frr-pc:/etc/routers/r3# ospfd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 25. Verifying the routing table of router r2.

Step 6. Type the following command to display the routing table of router r1.

```
show ip route
```



```
"Host: r1"
frr-pc# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

O   1.1.1.1/32 [110/0] is directly connected, lo, 00:05:24
C>* 1.1.1.1/32 is directly connected, lo, 00:05:30
O>* 1.1.1.2/32 [110/20] via 192.168.13.2, r1-eth1, 00:02:36
O   192.168.13.0/30 [110/10] is directly connected, r1-eth1, 00:05:24
C>* 192.168.13.0/30 is directly connected, r1-eth1, 00:05:30
O>* 192.168.23.0/30 [110/20] via 192.168.13.2, r1-eth1, 00:02:46
frr-pc#
```

Figure 26. Verifying the routing table of router r1.

Consider the figure above. Router r1 has routes to the networks 1.1.1.2/32 and 192.168.23.0/30 via IP address 192.168.13.2. Other networks have two available paths from router r1. The Administrative Distance (AD) of the paths advertised through OSPF is 110. The AD is a value used by the routers to select the best path when there are multiple available routes to the same destination. A smaller AD is always preferable to the routers. The characters `>*` indicate that the following path is used to reach a specific network.

Router r1 prefers directly connected networks over OSPF since the former has a lower AD than the latter.

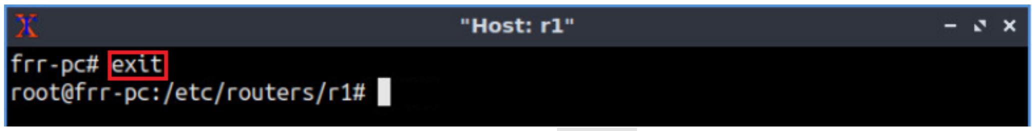
It may take some time to show all the routes.

4 Configuring VXLAN

In this section, you will configure VXLAN associated with a virtual tunnel (bridge).

Step 1. In router r1, type the following command to exit the vtysh session:

```
exit
```

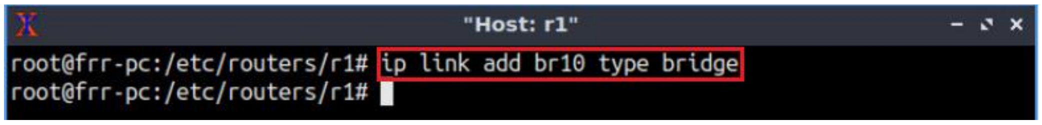


```
frr-pc# exit
root@frr-pc:/etc/routers/r1#
```

Figure 27. Exiting the `vtysh` session.

Step 2. Type the following command to create a tunnel named *br10* in router r1:

```
ip link add br10 type bridge
```

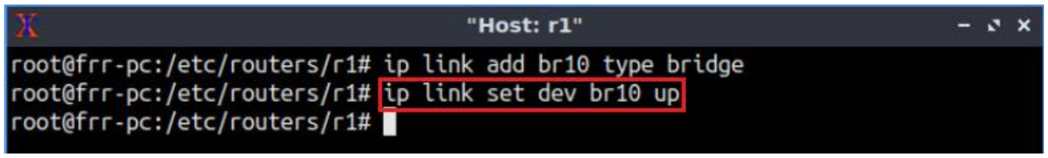


```
root@frr-pc:/etc/routers/r1# ip link add br10 type bridge
root@frr-pc:/etc/routers/r1#
```

Figure 28. Creating a bridge in router r1.

Step 3. Type the following command to enable bridge *br10*:

```
ip link set dev br10 up
```

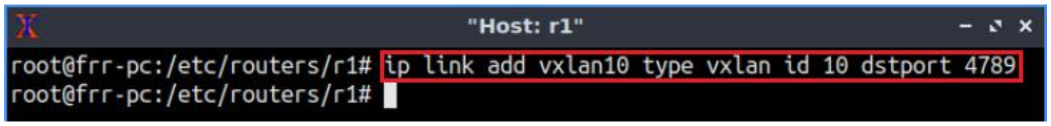


```
root@frr-pc:/etc/routers/r1# ip link add br10 type bridge
root@frr-pc:/etc/routers/r1# ip link set dev br10 up
root@frr-pc:/etc/routers/r1#
```

Figure 29. Enabling a bridge in router r1.

Step 4. Type the following command to create a VXLAN named *vxlan10* in router r1.

```
ip link add vxlan10 type vxlan id 10 dstport 4789
```



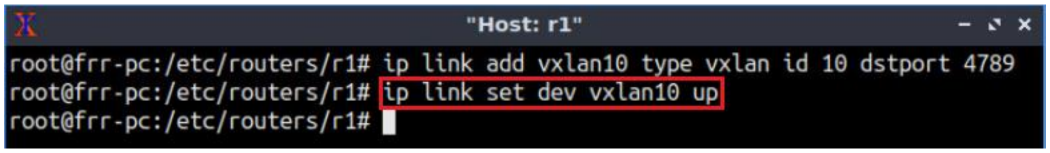
```
root@frr-pc:/etc/routers/r1# ip link add vxlan10 type vxlan id 10 dstport 4789
root@frr-pc:/etc/routers/r1#
```

Figure 30. Creating VXLAN in router r1.

The command creates *vxlan10* where the VNI is 10 which is a tag used to uniquely identify the VXLAN. The destination port is also referred to as *4789* which is the UDP port recognized as the VXLAN socket.

Step 5. Type the following command to enable the VXLAN, *vxlan10*.

```
ip link set dev vxlan10 up
```

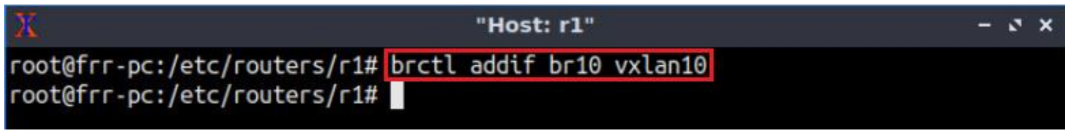


```
root@frr-pc:/etc/routers/r1# ip link add vxlan10 type vxlan id 10 dstport 4789
root@frr-pc:/etc/routers/r1# ip link set dev vxlan10 up
root@frr-pc:/etc/routers/r1#
```

Figure 31. Enabling VXLAN in router r1.

Step 6. Type the following command in router r1 in order to link *vxlan10* to the bridge *br10*.

```
brctl addif br10 vxlan10
```

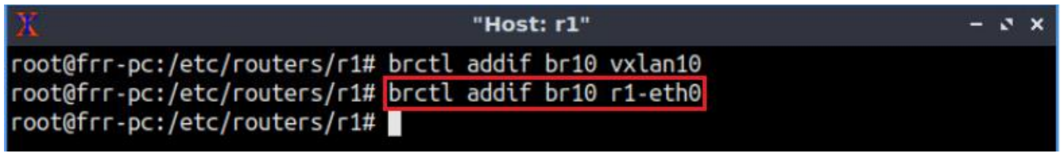


```
root@frr-pc:/etc/routers/r1# brctl addif br10 vxlan10
root@frr-pc:/etc/routers/r1#
```

Figure 32. Linking the VXLAN to the bridge.

Step 7. Type the following command in router r1 in order to link router interface *r1-eth0* to the bridge *br10*. The interface will be used to create the tunnel with router r2.

```
brctl addif br10 r1-eth0
```

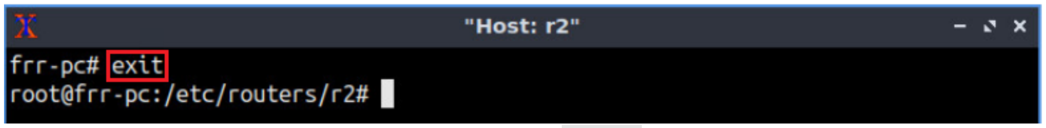


```
root@frr-pc:/etc/routers/r1# brctl addif br10 vxlan10
root@frr-pc:/etc/routers/r1# brctl addif br10 r1-eth0
root@frr-pc:/etc/routers/r1#
```

Figure 33. Linking the router interface to the bridge.

Step 8. In router r2, type the following command to exit the vtysh session:

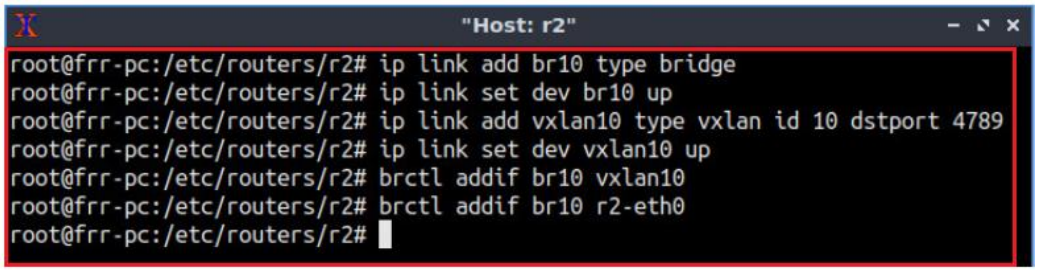
```
exit
```



```
frr-pc# exit
root@frr-pc:/etc/routers/r2#
```

Figure 34. Exiting the `vtysh` session.

Step 9. Similar to router r1, create *vxlan10* associated with bridge *br10* in router r2. All the steps are summarized in the following figure.



```
root@frr-pc:/etc/routers/r2# ip link add br10 type bridge
root@frr-pc:/etc/routers/r2# ip link set dev br10 up
root@frr-pc:/etc/routers/r2# ip link add vxlan10 type vxlan id 10 dstport 4789
root@frr-pc:/etc/routers/r2# ip link set dev vxlan10 up
root@frr-pc:/etc/routers/r2# brctl addif br10 vxlan10
root@frr-pc:/etc/routers/r2# brctl addif br10 r2-eth0
root@frr-pc:/etc/routers/r2#
```

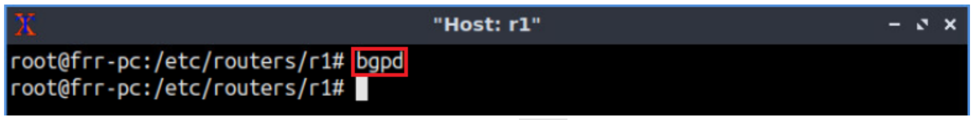
Figure 35. Creating bridge and VXLAN in router r2.

5 Configuring IBGP in routers

In this section, you will configure IBGP in routers. Through BGP configuration, routers r1 and r2 will establish a neighborship with router r3. Router r3 will get routing and MAC address information through BGP and the information will be advertised to other routers since router r3 will act as a route-reflector.

Step 1. Type the following command in router r1 terminal to enable BGP daemon.

```
bgpd
```

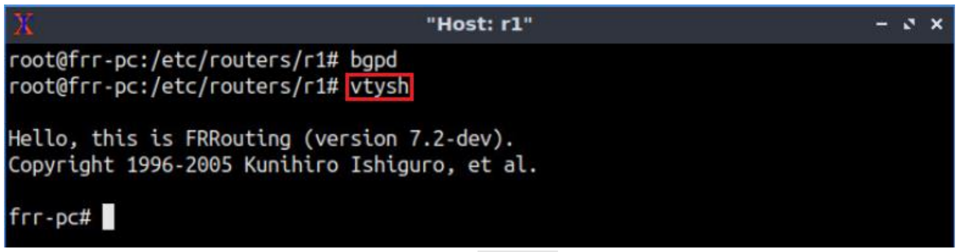


```
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1#
```

Figure 36. Starting `BGP` daemon.

Step 2. In order to enter router r1 terminal, issue the following command:

```
vtysh
```

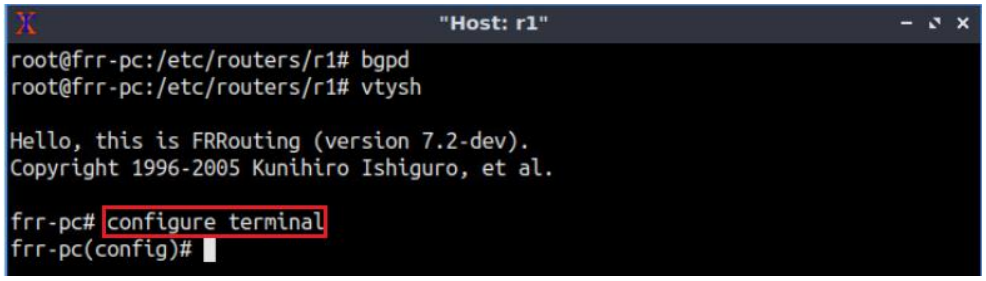


```
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 37. Starting `vtysh` in router r1.

Step 3. To enable configuration mode in router r1, issue the following command:

```
configure terminal
```

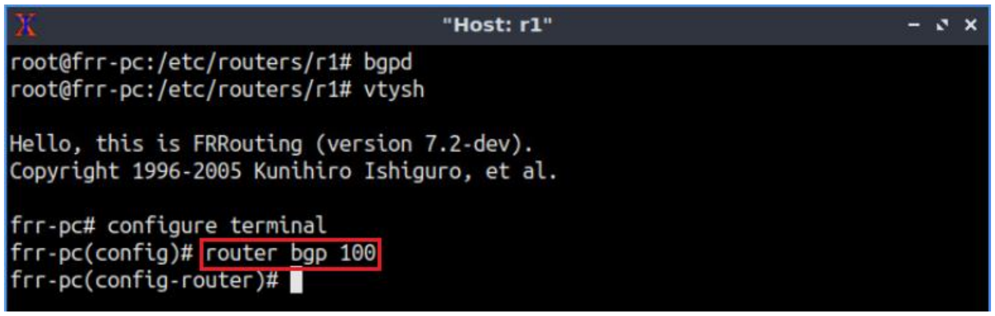


```
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)#
```

Figure 38. Enabling configuration mode in router r1.

Step 4. The BGP Autonomous System Number (ASN) assigned for router r1 is 100. In order to configure BGP, type the following command:

```
router bgp 100
```

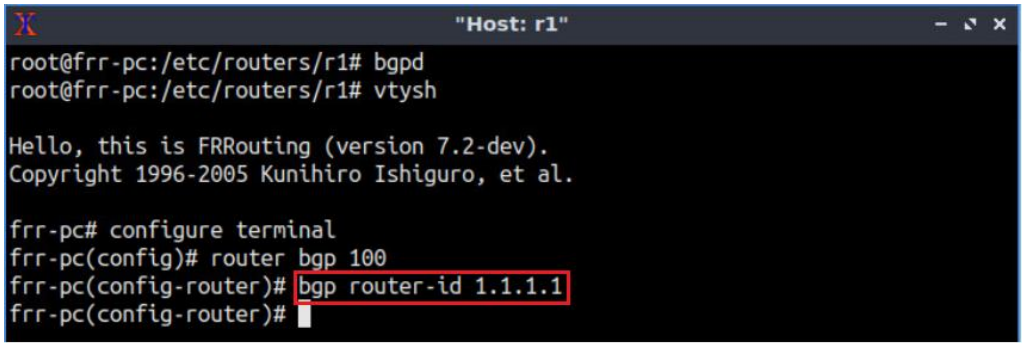


```
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)#
```

Figure 39. Configuring BGP in router r1.

Step 5. Assign a router ID to router r1 by issuing the following command.

```
bgp router-id 1.1.1.1
```

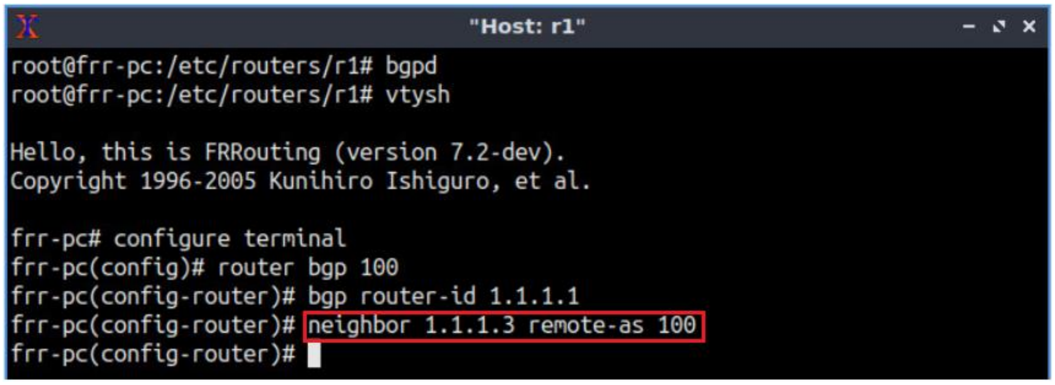


```
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 1.1.1.1
frr-pc(config-router)#
```

Figure 40. Assigning a router ID in router r1.

Step 6. To configure a BGP neighbor to router r1 (AS 100), type the command shown below. This command specifies the neighbor IP address (1.1.1.3) and ASN of the remote BGP peer (AS 100).

```
neighbor 1.1.1.3 remote-as 100
```

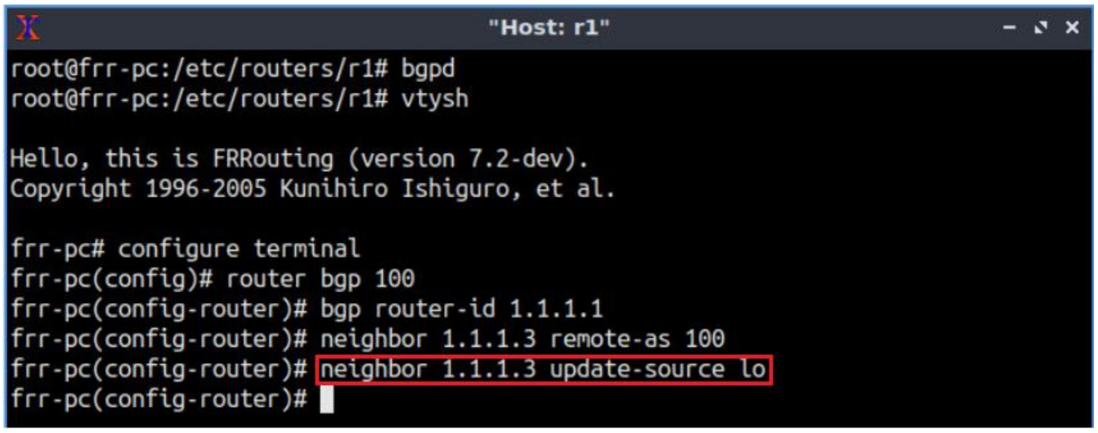


```
"Host: r1"
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 1.1.1.1
frr-pc(config-router)# neighbor 1.1.1.3 remote-as 100
frr-pc(config-router)#
```

Figure 41. Assigning BGP neighbor to router r1.

Step 7. Type the following command to assign *lo* as the source IP in router r1.

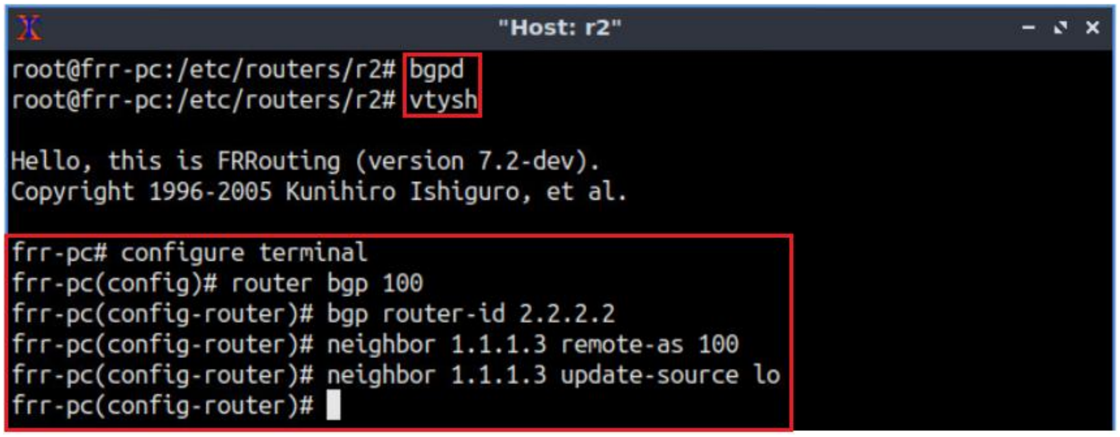
```
neighbor 1.1.1.3 update-source lo
```



```
"Host: r1"
root@frr-pc:/etc/routers/r1# bgpd
root@frr-pc:/etc/routers/r1# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 1.1.1.1
frr-pc(config-router)# neighbor 1.1.1.3 remote-as 100
frr-pc(config-router)# neighbor 1.1.1.3 update-source lo
frr-pc(config-router)#
```

Figure 42. Assigning loopback as source IP for the neighbor 1.1.1.3.

Step 8. Follow from step 1 to step 7 to configure BGP on router r2. All the steps are summarized in the following figure.

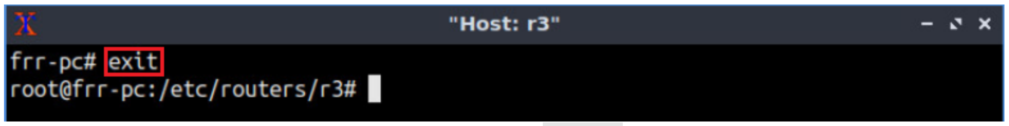


```
"Host: r2"
root@frr-pc:/etc/routers/r2# bgpd
root@frr-pc:/etc/routers/r2# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 2.2.2.2
frr-pc(config-router)# neighbor 1.1.1.3 remote-as 100
frr-pc(config-router)# neighbor 1.1.1.3 update-source lo
frr-pc(config-router)#
```

Figure 43. Configuring IBGP in router r2.

Step 9. In router r3, type the following command to exit the vtysh session:

```
exit
```

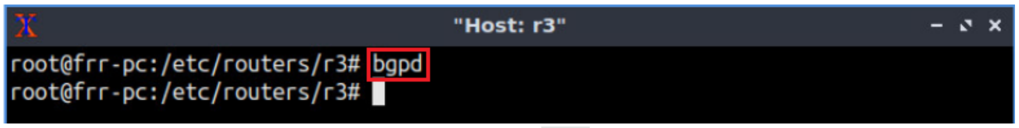


```
frr-pc# exit
root@frr-pc:/etc/routers/r3#
```

Figure 44. Exiting the `vtysh` session.

Step 10. Type the following command on router r3 terminal to enable BGP daemon.

```
bgpd
```

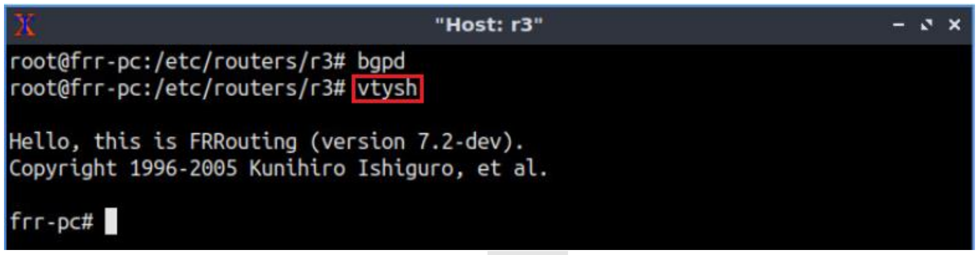


```
root@frr-pc:/etc/routers/r3# bgpd
root@frr-pc:/etc/routers/r3#
```

Figure 45. Starting `BGP` daemon.

Step 11. In order to enter to router r3 terminal, issue the following command:

```
vtysh
```

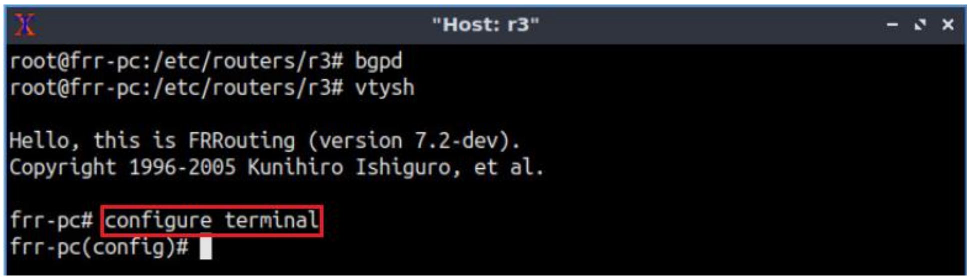


```
root@frr-pc:/etc/routers/r3# bgpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc#
```

Figure 46. Starting `vtysh` in router r1.

Step 12. To enable configuration mode in router r3, issue the following command:

```
configure terminal
```

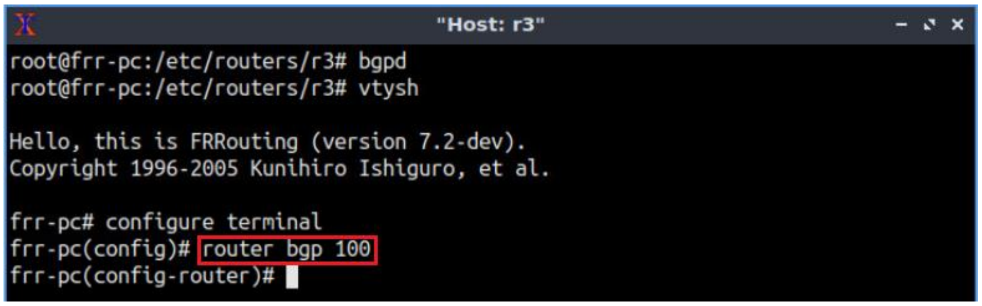


```
root@frr-pc:/etc/routers/r3# bgpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)#
```

Figure 47. Enabling configuration mode in router r3.

Step 13. The BGP Autonomous System Number (ASN) assigned for router r3 is 100. In order to configure BGP, type the following command:

```
router bgp 100
```

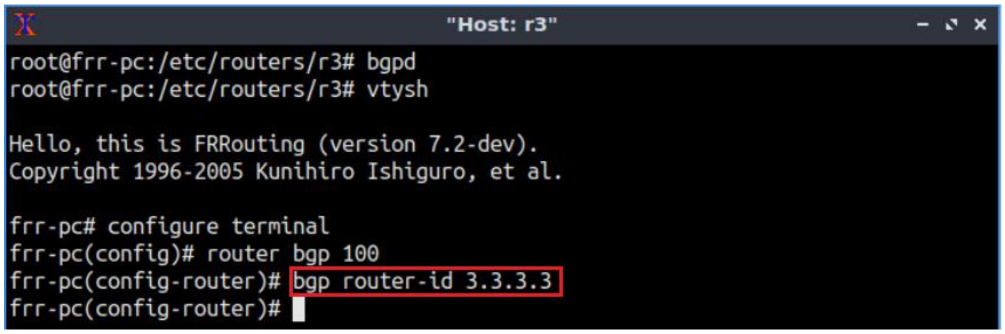


```
root@frr-pc:/etc/routers/r3# bgpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)#
```

Figure 48. Configuring BGP in router r3.

Step 14. Assign a router ID to router r3 by issuing the following command.

```
bgp router-id 3.3.3.3
```

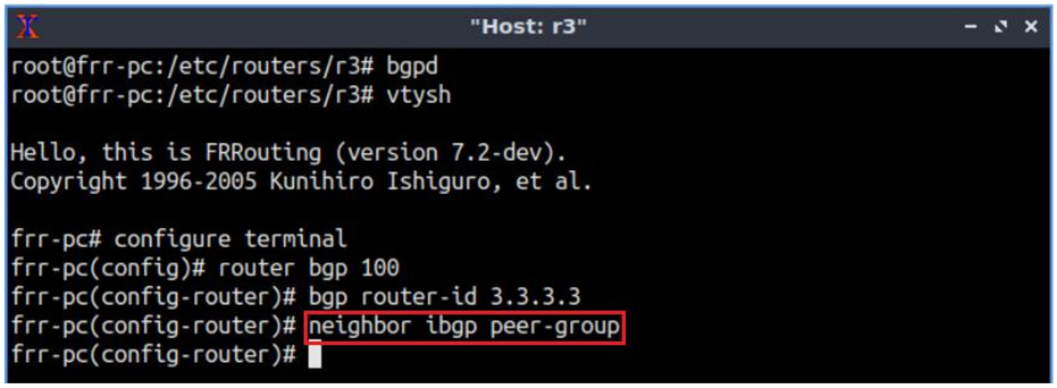


```
root@frr-pc:/etc/routers/r3# bgpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 3.3.3.3
frr-pc(config-router)#
```

Figure 49. Assigning a router ID in router r3.

Step 15. Type the following command to create a peer-group in router r3.

```
neighbor ibgp peer-group
```



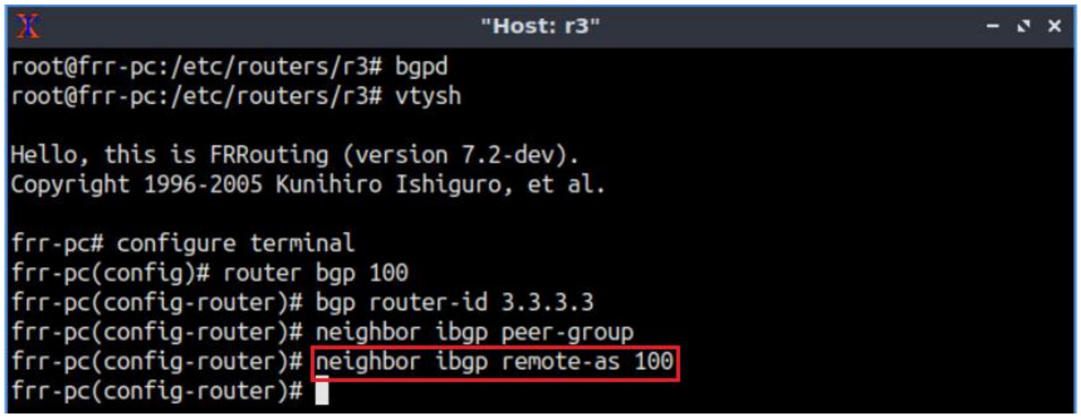
```
root@frr-pc:/etc/routers/r3# bgpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 3.3.3.3
frr-pc(config-router)# neighbor ibgp peer-group
frr-pc(config-router)#
```

Figure 50. Creating BGP peer-group in router r3.

This command is used when a router has a group of neighbors with the same update policies. The update is generated once per group rather than for each neighbor. Router r3 will create neighborships with routers r1 and r2 using the same policies.

Step 16. In router r3 terminal, type the following command to configure BGP neighbors. This command will allow router r3 to create neighborships with routers r1 and r2.

```
neighbor ibgp remote-as 100
```

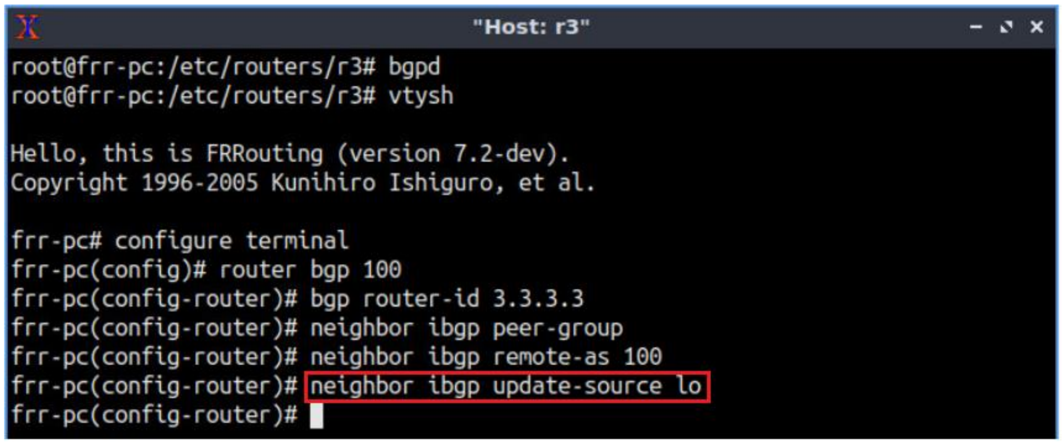


```
"Host: r3"
root@frr-pc:/etc/routers/r3# bgpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 3.3.3.3
frr-pc(config-router)# neighbor ibgp peer-group
frr-pc(config-router)# neighbor ibgp remote-as 100
frr-pc(config-router)# 
```

Figure 51. Assigning BGP neighbor to router r3.

Step 17. Type the following command to assign *lo* as the source IP in router r3.

```
neighbor ibgp update-source lo
```

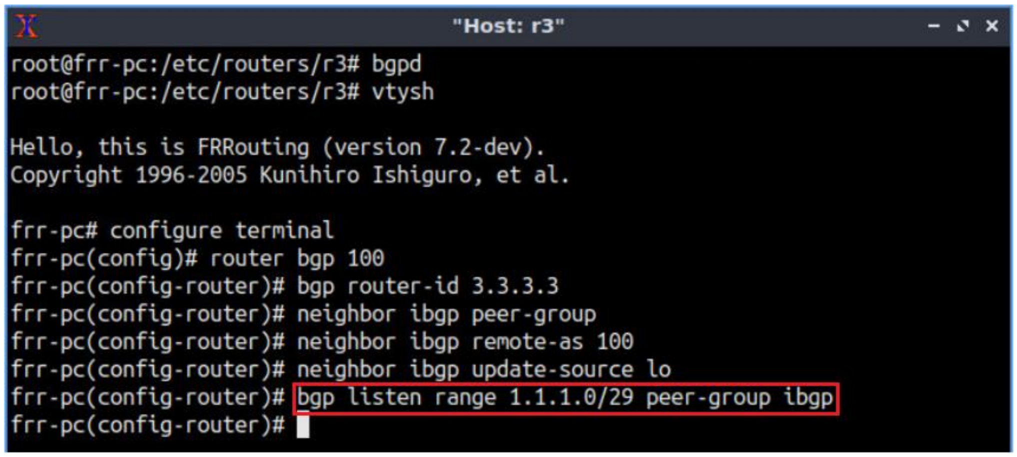


```
"Host: r3"
root@frr-pc:/etc/routers/r3# bgpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 3.3.3.3
frr-pc(config-router)# neighbor ibgp peer-group
frr-pc(config-router)# neighbor ibgp remote-as 100
frr-pc(config-router)# neighbor ibgp update-source lo
frr-pc(config-router)# 
```

Figure 52. Assigning the loopback address as the source IP in router r3.

Step 18. Type the following command to define the IP subnet range and associate it with the peer group.

```
bgp listen range 1.1.1.0/29 peer-group ibgp
```



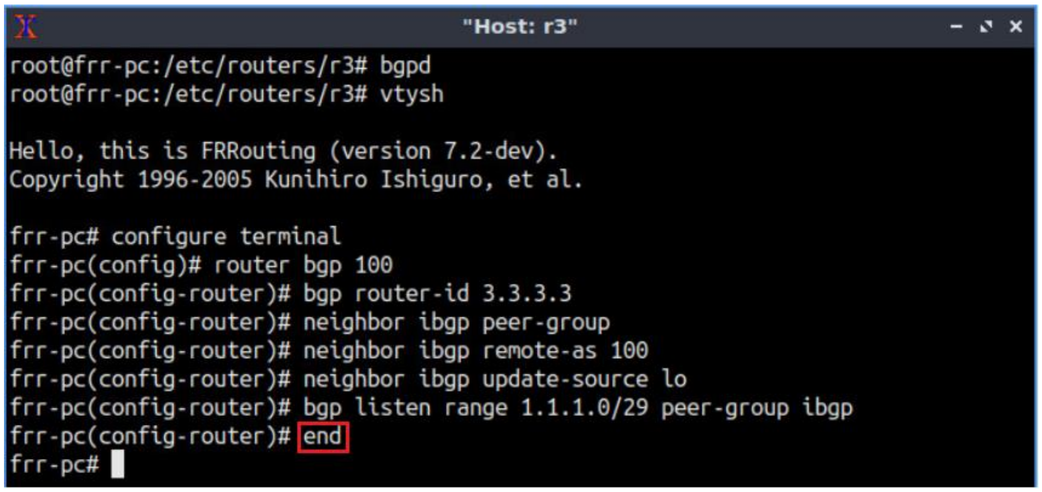
```
"Host: r3"
root@frr-pc:/etc/routers/r3# bgpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 3.3.3.3
frr-pc(config-router)# neighbor ibgp peer-group
frr-pc(config-router)# neighbor ibgp remote-as 100
frr-pc(config-router)# neighbor ibgp update-source lo
frr-pc(config-router)# bgp listen range 1.1.1.0/29 peer-group ibgp
frr-pc(config-router)# 
```

Figure 53. Defining IP subnet range in router r3.

Consider the command above. A range of IP addresses is specified which is trusted to become BGP peers with the router r3. Router r3 will create BGP neighborships with the loopback address of routers r1 and r2 (1.1.1.1, 1.1.1.2).

Step 19. Type the following command to exit configuration mode.

```
end
```

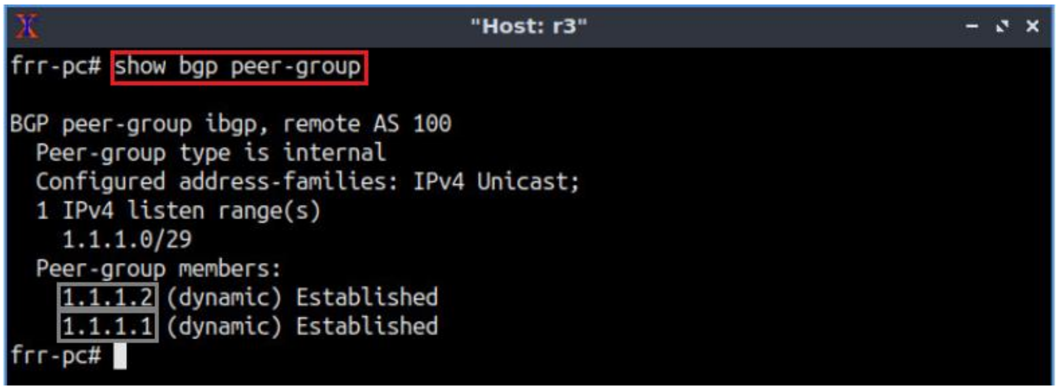


```
root@frr-pc:/etc/routers/r3# bgpd
root@frr-pc:/etc/routers/r3# vtysh

Hello, this is FRRouting (version 7.2-dev).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# bgp router-id 3.3.3.3
frr-pc(config-router)# neighbor ibgp peer-group
frr-pc(config-router)# neighbor ibgp remote-as 100
frr-pc(config-router)# neighbor ibgp update-source lo
frr-pc(config-router)# bgp listen range 1.1.1.0/29 peer-group ibgp
frr-pc(config-router)# end
frr-pc#
```

Figure 54. Exiting from the configuration mode.

Step 20. Type the following command to display the BGP peer-group in router r3.

```
show bgp peer-group
```



```
frr-pc# show bgp peer-group

BGP peer-group ibgp, remote AS 100
  Peer-group type is internal
  Configured address-families: IPv4 Unicast;
  1 IPv4 listen range(s)
    1.1.1.0/29
  Peer-group members:
    1.1.1.2 (dynamic) Established
    1.1.1.1 (dynamic) Established
frr-pc#
```

Figure 55. Verifying BGP peer-group in router r3.

Consider the figure above. The loopback addresses of routers r1 and r2 are assigned as peer-group members of router r3.

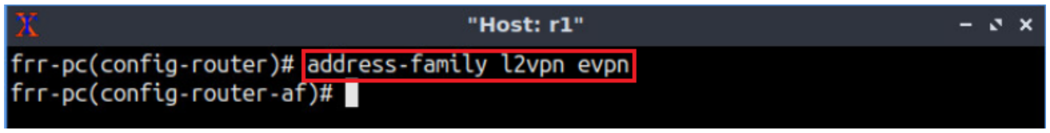
It takes 90 seconds to establish the connections.

6 Configuring EVPN

In this section, you will configure the EVPN address family. You will create an address family for Layer 2 VPN and advertise the VNI through BGP so that routers can advertise all the VXLAN information to other routers.

Step 1. In router r1, type the following command to create an address-family for EVPN.

```
address-family l2vpn evpn
```

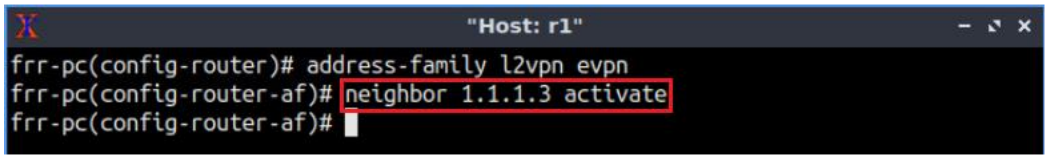


```
frr-pc(config-router)# address-family l2vpn evpn
frr-pc(config-router-af)#
```

Figure 56. Creating address-family in router r1.

Step 2. Type the following command to activate the neighborship with router r3.

```
neighbor 1.1.1.3 activate
```

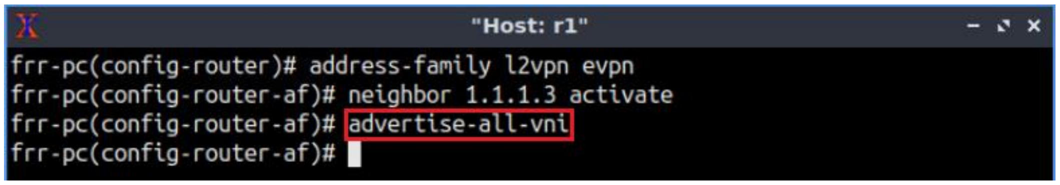


```
frr-pc(config-router)# address-family l2vpn evpn
frr-pc(config-router-af)# neighbor 1.1.1.3 activate
frr-pc(config-router-af)#
```

Figure 57. Activating neighborship with router r1.

Step 3. Type the following command in order to advertise the VNI.

```
advertise-all-vni
```



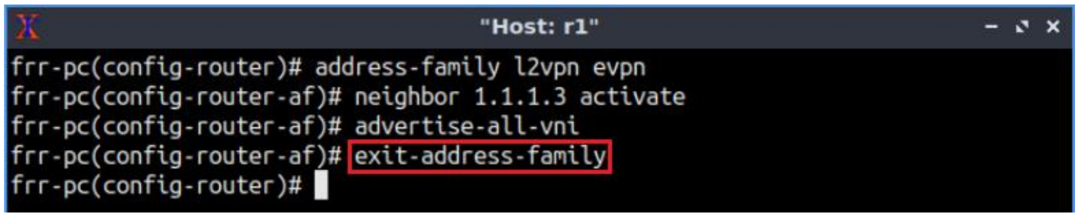
```
frr-pc(config-router)# address-family l2vpn evpn
frr-pc(config-router-af)# neighbor 1.1.1.3 activate
frr-pc(config-router-af)# advertise-all-vni
frr-pc(config-router-af)#
```

Figure 58. Advertising VNI in router r1.

The above command will allow router r1 to exchange VNI information with other routers through BGP.

Step 4. Type the following command to exit address-family mode.

```
exit-address-family
```

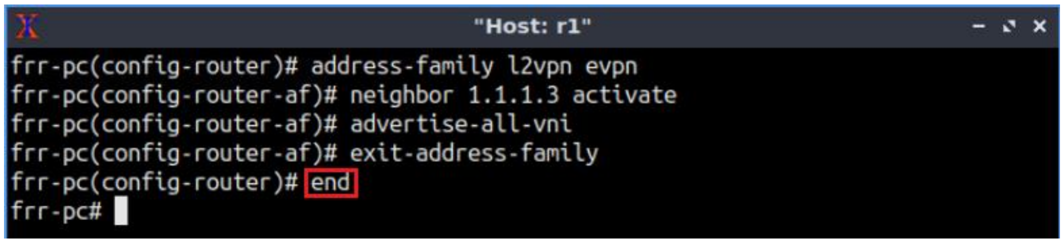


```
frr-pc(config-router)# address-family l2vpn evpn
frr-pc(config-router-af)# neighbor 1.1.1.3 activate
frr-pc(config-router-af)# advertise-all-vni
frr-pc(config-router-af)# exit-address-family
frr-pc(config-router)#
```

Figure 59. Exiting address-family mode.

Step 5. Type the following command to exit from configuration mode.

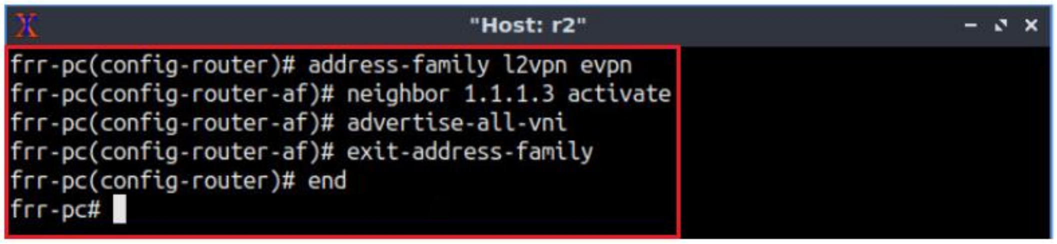
```
end
```



```
"Host: r1"
frr-pc(config-router)# address-family l2vpn evpn
frr-pc(config-router-af)# neighbor 1.1.1.3 activate
frr-pc(config-router-af)# advertise-all-vni
frr-pc(config-router-af)# exit-address-family
frr-pc(config-router)# end
frr-pc#
```

Figure 60. Exiting configuration mode.

Step 6. Router r2 is configured similarly to router r1. Those steps are summarized in the following figure.

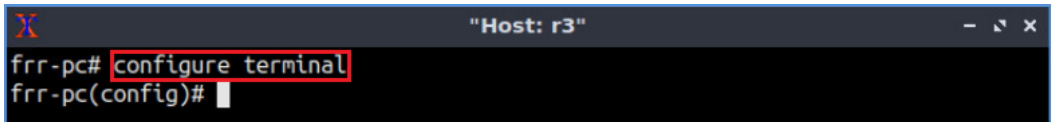


```
"Host: r2"
frr-pc(config-router)# address-family l2vpn evpn
frr-pc(config-router-af)# neighbor 1.1.1.3 activate
frr-pc(config-router-af)# advertise-all-vni
frr-pc(config-router-af)# exit-address-family
frr-pc(config-router)# end
frr-pc#
```

Figure 61. Configuring EVPN in router r2.

Step 7. To enable router r3 configuration mode, issue the following command:

```
configure terminal
```

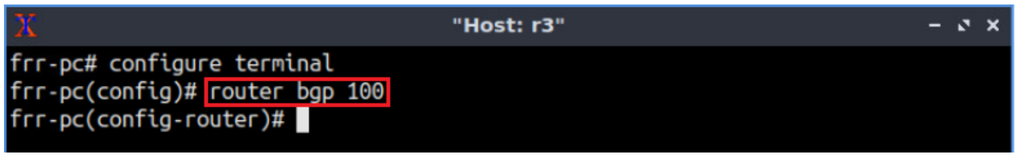


```
"Host: r3"
frr-pc# configure terminal
frr-pc(config)#
```

Figure 62. Enabling configuration mode in router r3.

Step 8. In order to configure BGP, type the following command:

```
router bgp 100
```

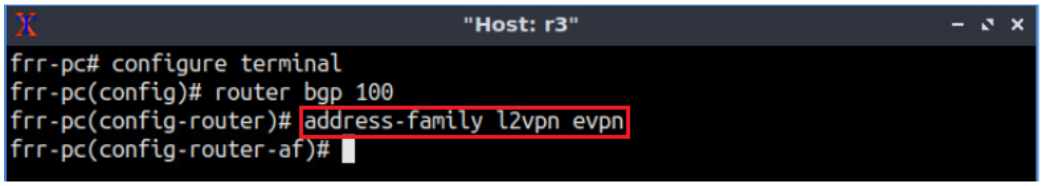


```
"Host: r3"
frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)#
```

Figure 63. Configuring BGP in router r3.

Step 9. Type the following command to create an address-family for EVPN in router r3.

```
address-family l2vpn evpn
```

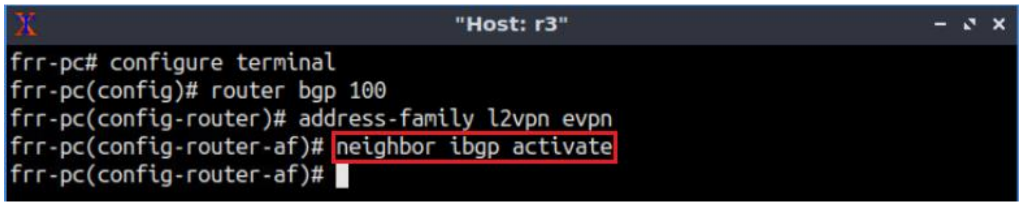


```
"Host: r3"
frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# address-family l2vpn evpn
frr-pc(config-router-af)#
```

Figure 64. Creating address-family in router r3.

Step 10. Type the following command to activate neighborship with routers r1 and r2.

```
neighbor ibgp activate
```



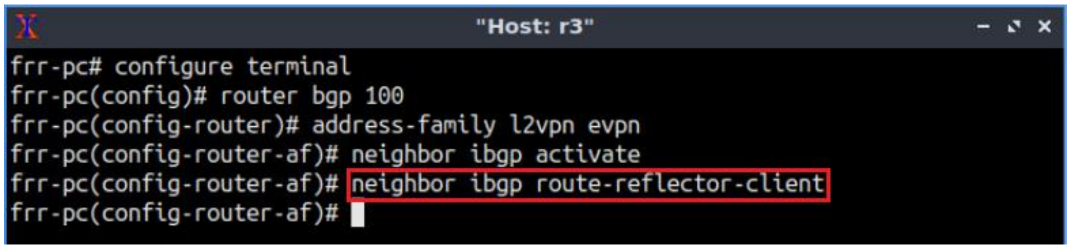
```
"Host: r3"
frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# address-family l2vpn evpn
frr-pc(config-router-af)# neighbor ibgp activate
frr-pc(config-router-af)#
```

Figure 65. Activating neighborship in router r3.

The keyword `ibgp` refers to the peer-group which was created in the previous section.

Step 11. Type the following command to configure router r3 so that it advertises routes to its IBGP neighbor routers, r1 and r2.

```
neighbor ibgp route-reflector-client
```

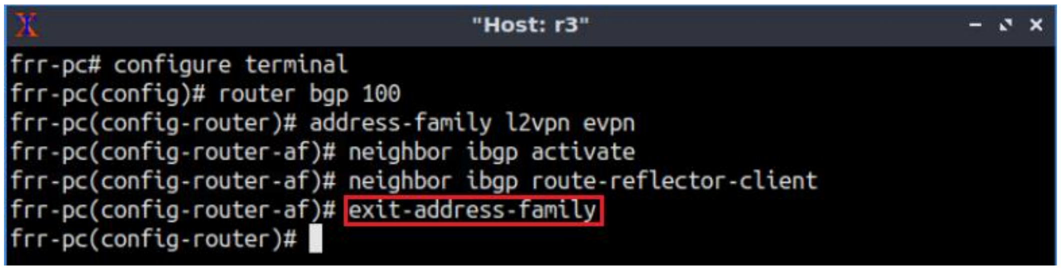


```
"Host: r3"
frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# address-family l2vpn evpn
frr-pc(config-router-af)# neighbor ibgp activate
frr-pc(config-router-af)# neighbor ibgp route-reflector-client
frr-pc(config-router-af)#
```

Figure 66. Configuring client peer to the route reflector in router r3.

Step 12. Type the following command to exit address-family mode.

```
exit-address-family
```

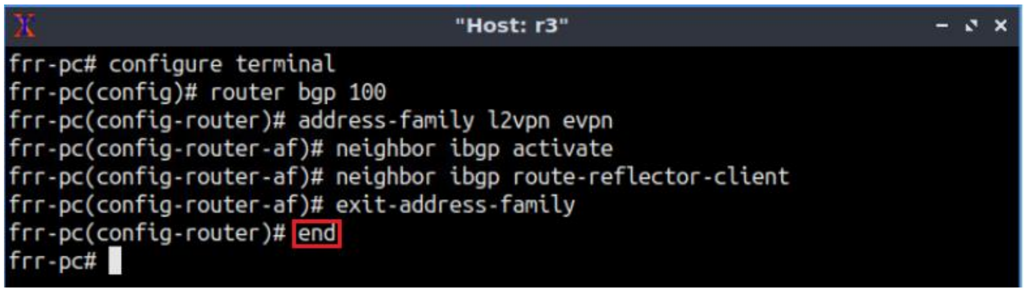


```
"Host: r3"
frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# address-family l2vpn evpn
frr-pc(config-router-af)# neighbor ibgp activate
frr-pc(config-router-af)# neighbor ibgp route-reflector-client
frr-pc(config-router-af)# exit-address-family
frr-pc(config-router)#
```

Figure 67. Exiting from address-family mode.

Step 13. Type the following command to exit from configuration mode.

```
end
```



```
frr-pc# configure terminal
frr-pc(config)# router bgp 100
frr-pc(config-router)# address-family l2vpn evpn
frr-pc(config-router-af)# neighbor ibgp activate
frr-pc(config-router-af)# neighbor ibgp route-reflector-client
frr-pc(config-router-af)# exit-address-family
frr-pc(config-router)# end
frr-pc#
```

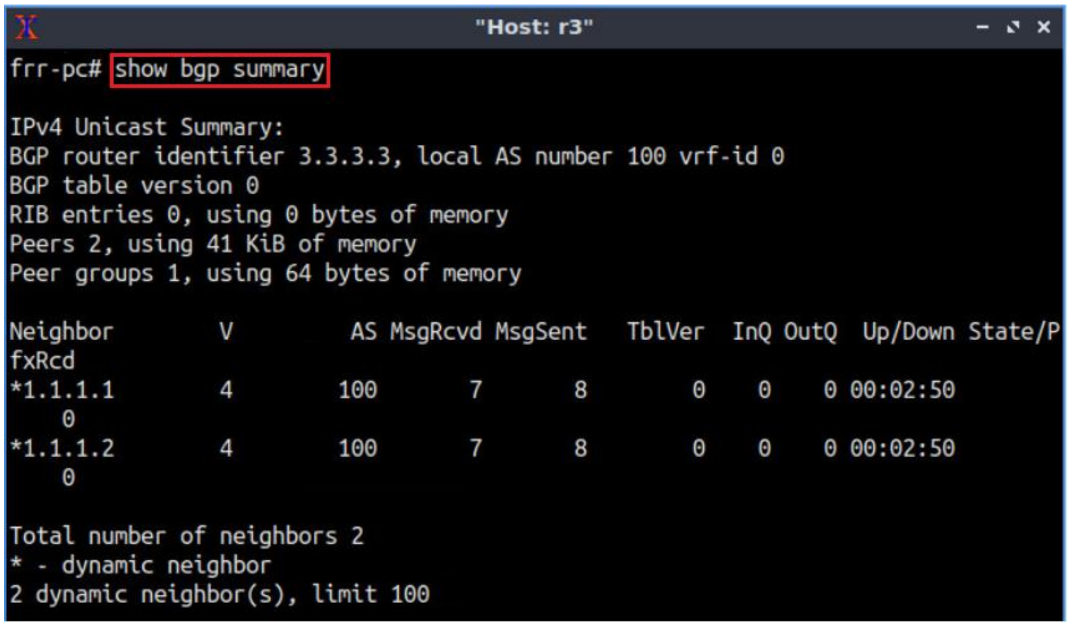
Figure 68. Exiting from configuration mode.

7 Verifying configuration

In this section, you will verify EVPN configuration.

Step 1. Type the following command to verify BGP configuration in router r3.

```
show bgp neighbors
```



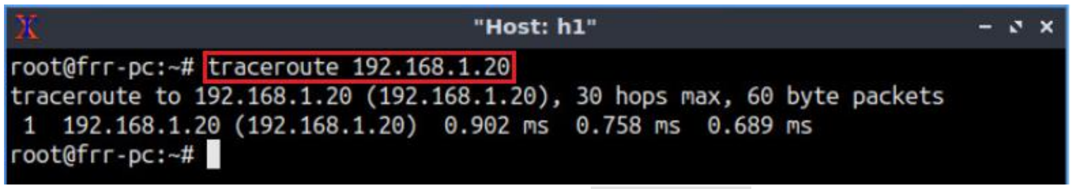
```
frr-pc# show bgp summary

IPv4 Unicast Summary:
BGP router identifier 3.3.3.3, local AS number 100 vrf-id 0
BGP table version 0
RIB entries 0, using 0 bytes of memory
Peers 2, using 41 KiB of memory
Peer groups 1, using 64 bytes of memory

Neighbor        V         AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down State/P
fxRcd
*1.1.1.1        4        100       7       8        0    0    0 00:02:50
    0
*1.1.1.2        4        100       7       8        0    0    0 00:02:50
    0

Total number of neighbors 2
* - dynamic neighbor
2 dynamic neighbor(s), limit 100
```

Figure 69. Verifying BGP neighbors in router r3.

The above figure shows all the BGP information including neighbors, peer-groups, router identifiers, ASNs, and other information.

Step 2. In host h1 terminal, perform a connectivity test between host h1 and host h2 by issuing the `traceroute` command.

```
traceroute 192.168.1.20
```



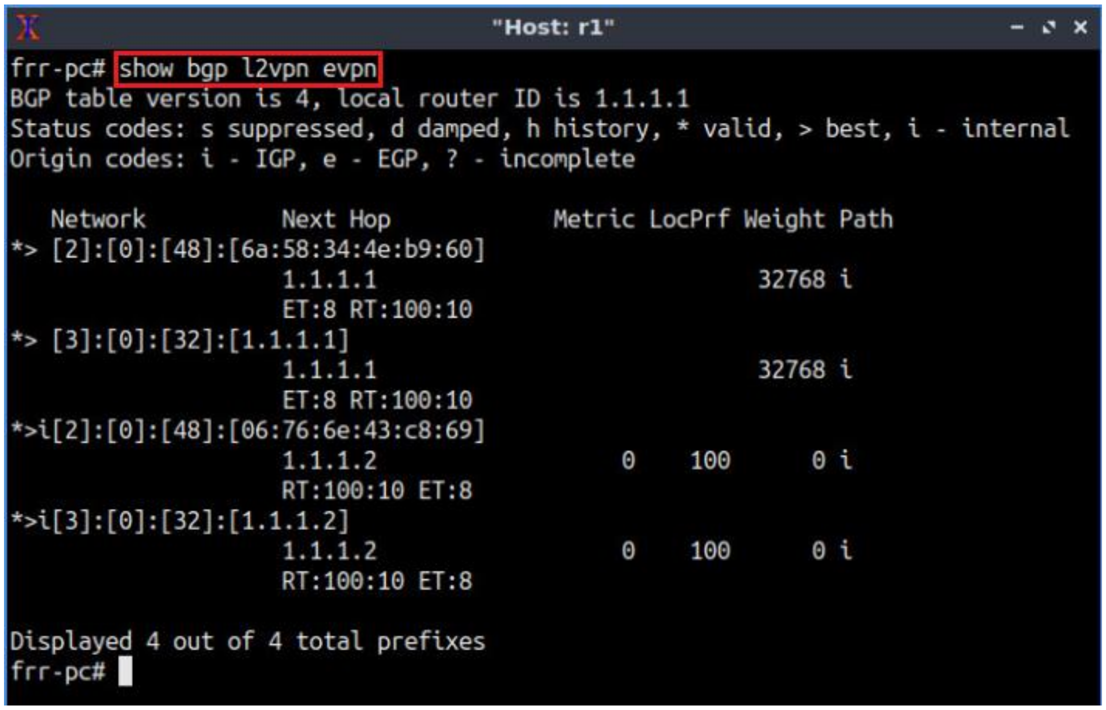
```
root@frr-pc:~# traceroute 192.168.1.20
traceroute to 192.168.1.20 (192.168.1.20), 30 hops max, 60 byte packets
 1  192.168.1.20 (192.168.1.20)  0.902 ms  0.758 ms  0.689 ms
root@frr-pc:~#
```

Figure 70. Connectivity test using `traceroute` command.

As the hosts are communicating with each other via layer 2 tunnel. The above figure shows only one hop (192.168.1.20) to reach the destination as it assumes the host to be directly connected via a VXLAN tunnel.

Step 3. Type the following command to verify EVPN configuration in router r1.

```
show bgp l2vpn evpn
```



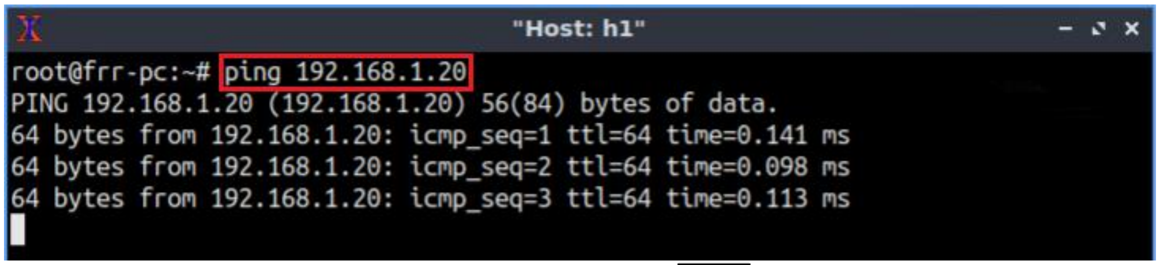
```
frr-pc# show bgp l2vpn evpn
BGP table version is 4, local router ID is 1.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> [2]:[0]:[48]:[6a:58:34:4e:b9:60]
                    1.1.1.1                            32768 i
                    ET:8 RT:100:10
*> [3]:[0]:[32]:[1.1.1.1]
                    1.1.1.1                            32768 i
                    ET:8 RT:100:10
*>i[2]:[0]:[48]:[06:76:6e:43:c8:69]
                    1.1.1.2                  0    100      0 i
                    RT:100:10 ET:8
*>i[3]:[0]:[32]:[1.1.1.2]
                    1.1.1.2                  0    100      0 i
                    RT:100:10 ET:8

Displayed 4 out of 4 total prefixes
frr-pc#
```

Figure 71. Verifying EVPN configuration in router r1.

Consider the figure above. The figure shows EVPN control plane information captured from BGP updates. Host MAC addresses are shown as routing entries. The network [2]:[0]:[48]:[6a:58:34:4e:b9:60] refers to route 2 (MAC), ethernet tag set to 0 and 48 bit MAC address. Similarly, the network [3]:[0]:[32]:[1.1.1.2] refers to route 3 (IP address), ethernet tag set to 0 and 32 bit IP address.

Step 4. In host h1 terminal, perform a connectivity test between host h1 and host h2 by issuing the command shown below. The result will show a successful connectivity test.

```
ping 192.168.1.20
```



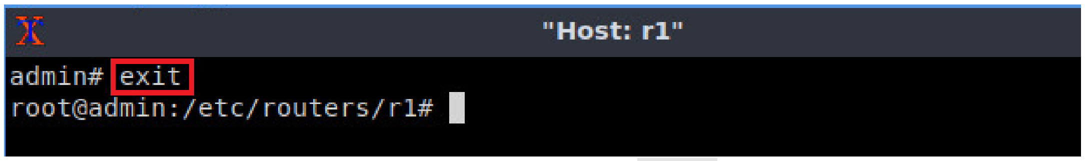
```
root@frr-pc:~# ping 192.168.1.20
PING 192.168.1.20 (192.168.1.20) 56(84) bytes of data.
64 bytes from 192.168.1.20: icmp_seq=1 ttl=64 time=0.141 ms
64 bytes from 192.168.1.20: icmp_seq=2 ttl=64 time=0.098 ms
64 bytes from 192.168.1.20: icmp_seq=3 ttl=64 time=0.113 ms
```

Figure 72. Connectivity test using `ping` command.

Do not stop the connectivity test.

Step 5. Click on router r1 terminal and issue the following command to exit the vtysh session.

```
exit
```

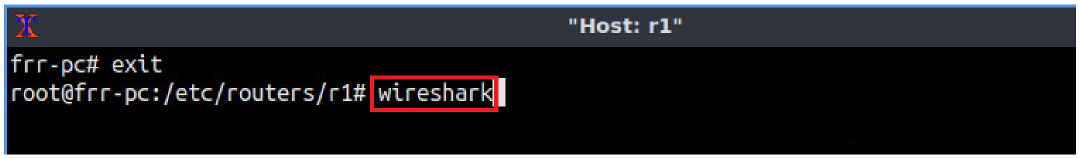


```
admin# exit
root@admin:/etc/routers/r1#
```

Figure 73. Exiting from `vtysh`.

Step 6. In router r1 terminal, start Wireshark dissector by issuing the following command. A new window will emerge.

```
wireshark
```



```
frr-pc# exit
root@frr-pc:/etc/routers/r1# wireshark
```

Figure 74. Starting Wireshark network analyzer.

Step 7. Click on interface *r1-eth1* then on the icon located on the upper left-hand side to start capturing packets on this interface.

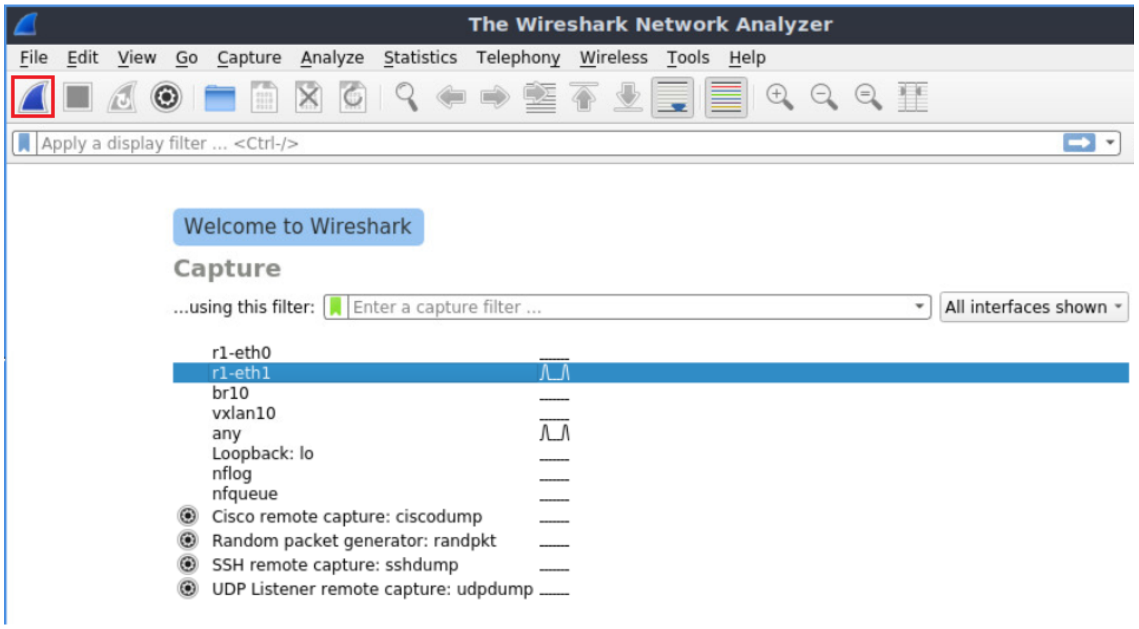
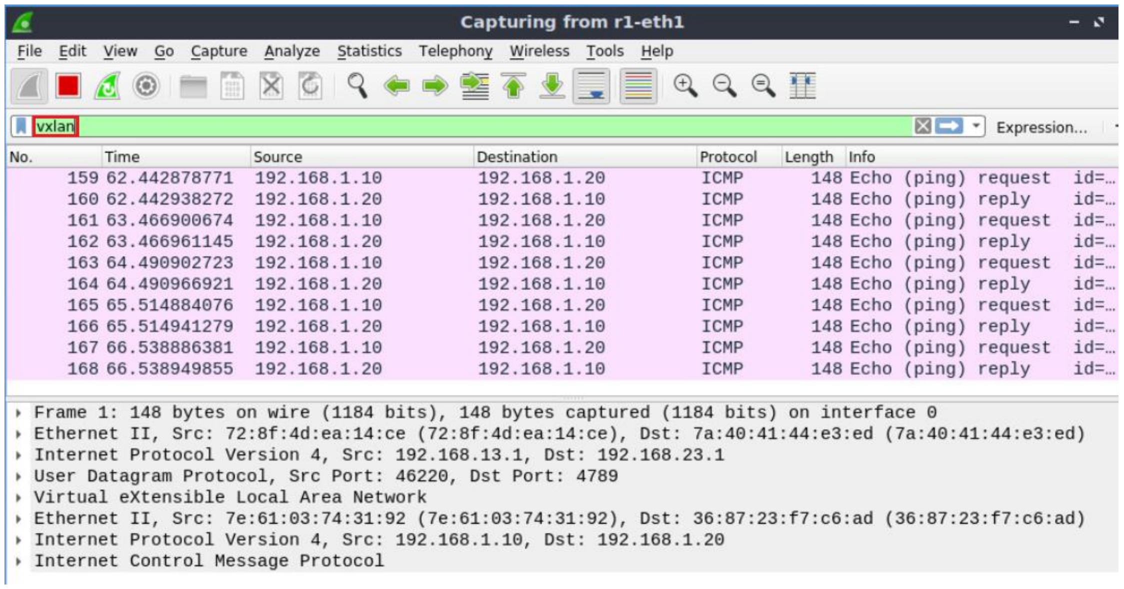

The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Capture
...using this filter: Enter a capture filter ...
All interfaces shown
r1-eth0
r1-eth1
br10
vxlan10
any
Loopback: lo
nflog
nfqueue
Cisco remote capture: ciscodump
Random packet generator: randpkt
SSH remote capture: sshdump
UDP Listener remote capture: udpdump

Figure 75. Starting packet capturing on interface r1-eth1.

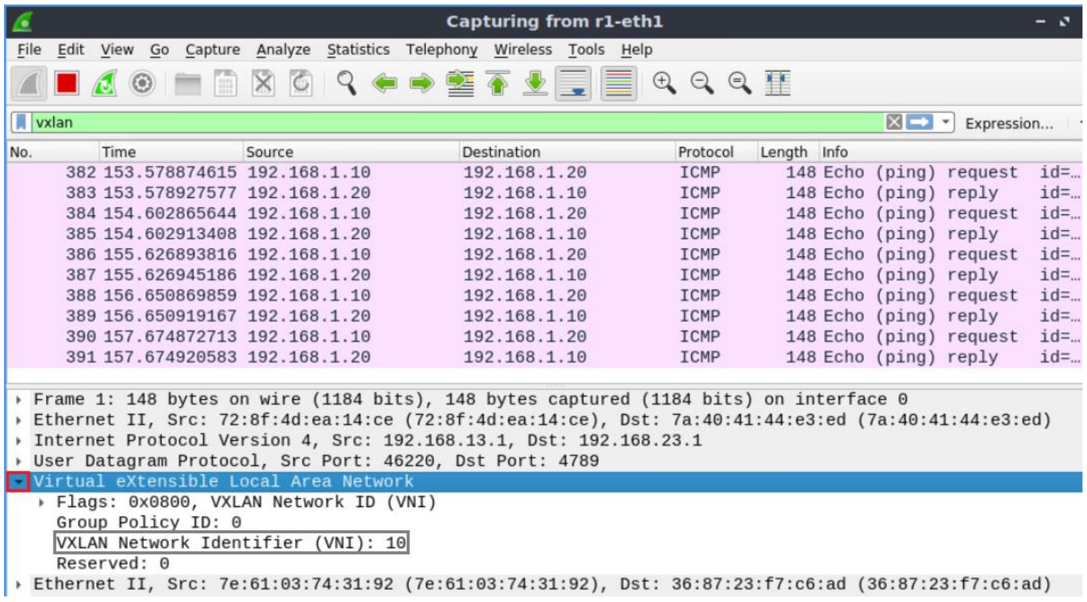
Step 8. In the filter box located on the upper left-hand side, type *vxlan* in order to filter the packets that contain VXLAN tags.


Capturing from r1-eth1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
vxlan
Expression...

```
No.   Time          Source        Destination   Protocol Length Info
  159 62.442878771  192.168.1.10  192.168.1.20  ICMP     148 Echo (ping) request  id=…
  160 62.442938272  192.168.1.20  192.168.1.10  ICMP     148 Echo (ping) reply    id=…
  161 63.466900674  192.168.1.10  192.168.1.20  ICMP     148 Echo (ping) request  id=…
  162 63.466961145  192.168.1.20  192.168.1.10  ICMP     148 Echo (ping) reply    id=…
  163 64.490902723  192.168.1.10  192.168.1.20  ICMP     148 Echo (ping) request  id=…
  164 64.490966921  192.168.1.20  192.168.1.10  ICMP     148 Echo (ping) reply    id=…
  165 65.514884076  192.168.1.10  192.168.1.20  ICMP     148 Echo (ping) request  id=…
  166 65.514941279  192.168.1.20  192.168.1.10  ICMP     148 Echo (ping) reply    id=…
  167 66.538886381  192.168.1.10  192.168.1.20  ICMP     148 Echo (ping) request  id=…
  168 66.538949855  192.168.1.20  192.168.1.10  ICMP     148 Echo (ping) reply    id=…

▸ Frame 1: 148 bytes on wire (1184 bits), 148 bytes captured (1184 bits) on interface 0
▸ Ethernet II, Src: 72:8f:4d:ea:14:ce (72:8f:4d:ea:14:ce), Dst: 7a:40:41:44:e3:ed (7a:40:41:44:e3:ed)
▸ Internet Protocol Version 4, Src: 192.168.13.1, Dst: 192.168.23.1
▸ User Datagram Protocol, Src Port: 46220, Dst Port: 4789
▸ Virtual eXtensible Local Area Network
▸ Ethernet II, Src: 7e:61:03:74:31:92 (7e:61:03:74:31:92), Dst: 36:87:23:f7:c6:ad (36:87:23:f7:c6:ad)
▸ Internet Protocol Version 4, Src: 192.168.1.10, Dst: 192.168.1.20
▸ Internet Control Message Protocol
```

Figure 76. Filtering network traffic.

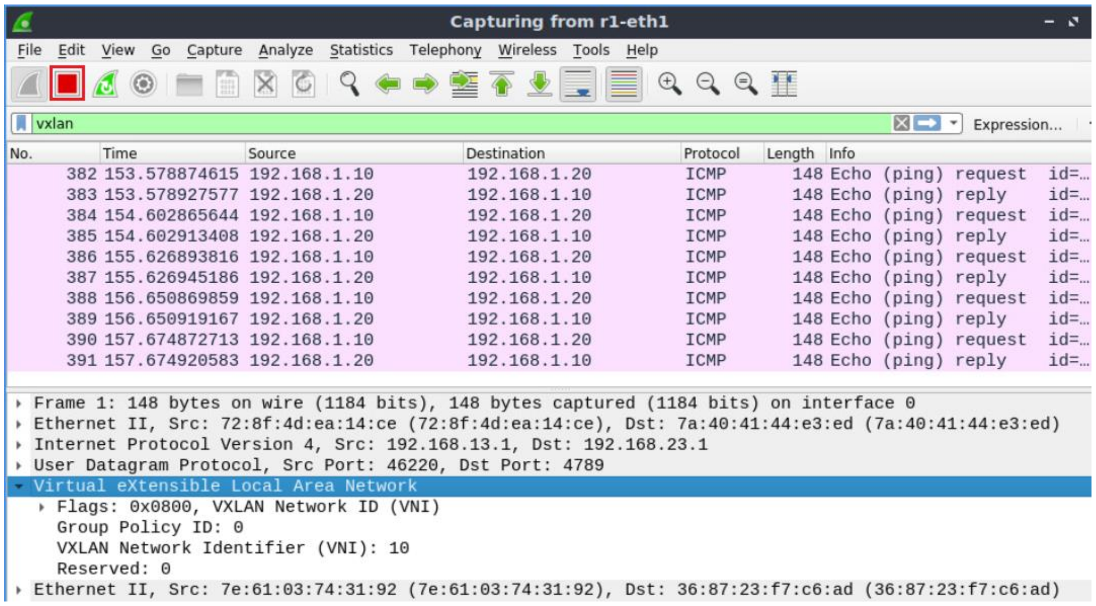
Step 9. Click on the arrow located on the left most of the field called *Virtual eXtensible Local Area Network.* A list will be displayed. Verify that the *VXLAN Network Identifier* is 10. Notice that such tag corresponds to the traffic from h1 to h2.


Capturing from r1-eth1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
vxlan
Expression...

```
No.   Time           Source          Destination     Protocol  Length  Info
  382 153.578874615  192.168.1.10    192.168.1.20    ICMP         148 Echo (ping) request  id=…
  383 153.578927577  192.168.1.20    192.168.1.10    ICMP         148 Echo (ping) reply    id=…
  384 154.602865644  192.168.1.10    192.168.1.20    ICMP         148 Echo (ping) request  id=…
  385 154.602913408  192.168.1.20    192.168.1.10    ICMP         148 Echo (ping) reply    id=…
  386 155.626893816  192.168.1.10    192.168.1.20    ICMP         148 Echo (ping) request  id=…
  387 155.626945186  192.168.1.20    192.168.1.10    ICMP         148 Echo (ping) reply    id=…
  388 156.650869859  192.168.1.10    192.168.1.20    ICMP         148 Echo (ping) request  id=…
  389 156.650919167  192.168.1.20    192.168.1.10    ICMP         148 Echo (ping) reply    id=…
  390 157.674872713  192.168.1.10    192.168.1.20    ICMP         148 Echo (ping) request  id=…
  391 157.674920583  192.168.1.20    192.168.1.10    ICMP         148 Echo (ping) reply    id=…

▸ Frame 1: 148 bytes on wire (1184 bits), 148 bytes captured (1184 bits) on interface 0
▸ Ethernet II, Src: 72:8f:4d:ea:14:ce (72:8f:4d:ea:14:ce), Dst: 7a:40:41:44:e3:ed (7a:40:41:44:e3:ed)
▸ Internet Protocol Version 4, Src: 192.168.13.1, Dst: 192.168.23.1
▸ User Datagram Protocol, Src Port: 46220, Dst Port: 4789
▾ Virtual eXtensible Local Area Network
  ▸ Flags: 0x0800, VXLAN Network ID (VNI)
    Group Policy ID: 0
    VXLAN Network Identifier (VNI): 10
    Reserved: 0
▸ Ethernet II, Src: 7e:61:03:74:31:92 (7e:61:03:74:31:92), Dst: 36:87:23:f7:c6:ad (36:87:23:f7:c6:ad)
```

Figure 77. Verifying VXLAN network identifier.

Step 10. To stop packet capturing, click on the red button located on the upper left-hand side.


Capturing from r1-eth1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
vxlan
Expression...

```
No.   Time           Source          Destination     Protocol  Length  Info
  382 153.578874615  192.168.1.10    192.168.1.20    ICMP         148 Echo (ping) request  id=…
  383 153.578927577  192.168.1.20    192.168.1.10    ICMP         148 Echo (ping) reply    id=…
  384 154.602865644  192.168.1.10    192.168.1.20    ICMP         148 Echo (ping) request  id=…
  385 154.602913408  192.168.1.20    192.168.1.10    ICMP         148 Echo (ping) reply    id=…
  386 155.626893816  192.168.1.10    192.168.1.20    ICMP         148 Echo (ping) request  id=…
  387 155.626945186  192.168.1.20    192.168.1.10    ICMP         148 Echo (ping) reply    id=…
  388 156.650869859  192.168.1.10    192.168.1.20    ICMP         148 Echo (ping) request  id=…
  389 156.650919167  192.168.1.20    192.168.1.10    ICMP         148 Echo (ping) reply    id=…
  390 157.674872713  192.168.1.10    192.168.1.20    ICMP         148 Echo (ping) request  id=…
  391 157.674920583  192.168.1.20    192.168.1.10    ICMP         148 Echo (ping) reply    id=…

▸ Frame 1: 148 bytes on wire (1184 bits), 148 bytes captured (1184 bits) on interface 0
▸ Ethernet II, Src: 72:8f:4d:ea:14:ce (72:8f:4d:ea:14:ce), Dst: 7a:40:41:44:e3:ed (7a:40:41:44:e3:ed)
▸ Internet Protocol Version 4, Src: 192.168.13.1, Dst: 192.168.23.1
▸ User Datagram Protocol, Src Port: 46220, Dst Port: 4789
▾ Virtual eXtensible Local Area Network
  ▸ Flags: 0x0800, VXLAN Network ID (VNI)
    Group Policy ID: 0
    VXLAN Network Identifier (VNI): 10
    Reserved: 0
▸ Ethernet II, Src: 7e:61:03:74:31:92 (7e:61:03:74:31:92), Dst: 36:87:23:f7:c6:ad (36:87:23:f7:c6:ad)
```

Figure 78. Stopping packet capturing.

Step 11. In host h1, press `Ctrl+c` to stop the test and close Wireshark.

This concludes Lab 8. Stop the emulation and then exit out of MiniEdit.

References

1. Liqin Dong, Yibin Yang, "*Scalable handling of BGP route information in VXLAN with EVPN control plane*", 2017.

2. M. Mahalingam, D. Dutt, K. Duda, P. Agarwal, L. Kreeger, T. Sridhar, M. Bursell and C. Wright. "*Virtual eXtensible local area network (VXLAN): a framework for overlaying virtualized layer 2 networks over layer 3 Networks*." *RFC* 7348 (2014): 1-22.
3. Juniper networks, "*EVPN overview*", [Online]. Available: https://www.juniper.net/documentation/en_US/junos/topics/concept/evpns-overview.html
4. Mininet walkthrough, [Online]. Available: http://mininet.org.
5. Cisco, "*A modern, open and scalable fabric VXLAN EVPN*", [Online]. Available: https://www.cisco.com/c/dam/en/us/td/docs/switches/datacenter/nexus9000/sw/vxlan_evpn/VXLAN_EVPN.pdf
6. B. Buresh, D. Eline, D. Jansen, J. Gmitter, J. Ostermiller, J. Moreno, K. Lei, L. Quan, L. Krattiger, M. Ardica, R. Parameswaran, R. Tappenden and S. Kondalam. "*A modern, open and scalable fabric VXLAN EVPN*."
7. Juniper Networks, "*What is EVPN-VXLAN in an Enterprise network*", [Online]. Available: https://www.juniper.net/us/en/products-services/what-is/evpn-vxlan/
8. IPSpace, "*What is EVPN*", [Online]. Available: https://blog.ipspace.net/2018/05/what-is-evpn.html
9. Cisco, "*IP Routing: BGP Configuration Guide, Cisco IOS Release 15M&T*", 2013.

UNIVERSITY OF
SOUTH CAROLINA

MPLS AND ADVANCED BGP TOPICS

Lab 9: Introduction to Segment Routing over IPv6 (SRv6)

Document Version: 12-21-2020

Award 1829698
"CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers"

Contents

Overview

This lab presents Segment Routing over IPv6 (SRv6). Segment routing uses the source packet routing technique. In source packet routing, the source or ingress router specifies the path a packet will take through the network, in contrast with traditional routing, where each router consults its local routing table to forward packets. In this lab, the user will enable and configure the routers to perform segment routing over IPv6 in the Linux kernel.

Objectives

By the end of this lab, students should be able to:

1. Understand the concept of segment routing.
2. Differentiate segment routing from traditional IP routing.
3. Configure segment routing over IPv6 on the Linux kernel.
4. Change the segment routing configuration to redirect packets to another path.
5. Verify the configuration using *Wireshark*.

Lab settings

The information in Table 1 provides the credentials to access the Client machine.

Table 1**.** Credentials to access the Client machine.

Device	**Account**	**Password**
Client	admin	password

Lab roadmap

This lab is organized as follows:

1. Section 1: Introduction to segment routing.
2. Section 2: Lab topology.
3. Section 3: Open topology and load the configuration.
4. Section 4: Enable segment routing in all routers.

1 Introduction to Segment Routing

In traditional IP routing, every router in the network maintains a local table containing routes to particular network destinations. This table is known as the routing table. Upon receiving a new packet, the router performs a routing table lookup using the packet's destination IP address to determine the appropriate output port.

Source routing is a different routing technique in which a network node (typically an ingress router) specifies the path that a packet will take through the network. An example of a technology that uses source routing is the Multiprotocol Label Switching (MPLS). In MPLS, labels are added to packets to determine how the packets will be forwarded in the network. MPLS requires intermediate nodes to maintain state information.

Segment routing (SR) is a recent source routing technique that enables an ingress router to insert a list of instructions (known as "segments") into the packet headers. The segments include the specifications of the required network path. SR and MPLS are similar in the sense that they are both based on source routing. However, there is a critical difference between the two. Unlike MPLS, no path information is maintained in intermediate nodes with SR. Moreover, SR can be used with MPLS and was not designed to replace existing traffic engineering mechanisms. Instead, SR can complement them while offering its advantages concerning path storage requirements[1].

1.1 Segment Routing Path

The SR path describes an ordered list of network segments that connect an SR ingress node to an SR egress node. An SR path can be ruled by policies such as the least-cost path from ingress to egress. For traffic engineering and network service administration purposes, the segments can follow a different path. Consider Figure 1, which shows an SR domain. When a packet arrives at the ingress node (R1), the router checks if it matches the conditions for an SR path. If there is a match, the router encapsulates the packet in a tunnel that traverses the network segment by segment in the SR path. Each segment is terminated by an endpoint that examines the packet, pops the outermost label/header, and forwards the packet to the next endpoint. When the packet reaches the egress node (R6), the router removes the SR header and forwards the packet based on the destination IP.

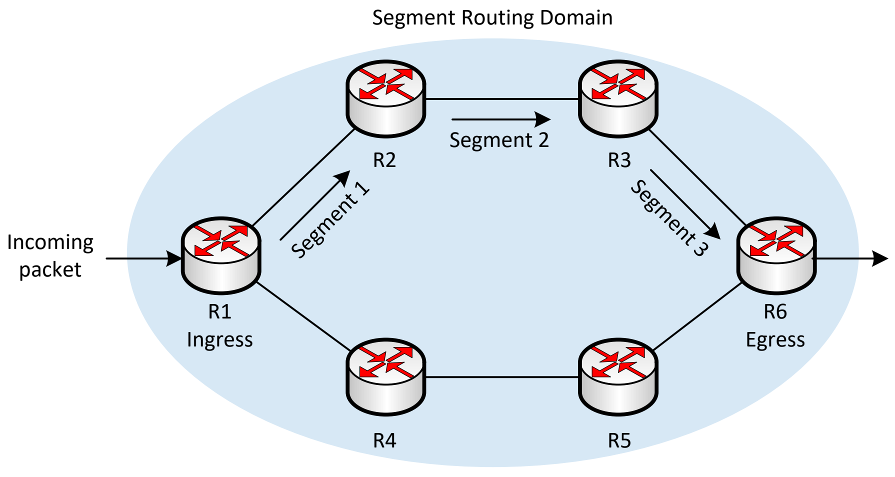


Figure 1. Segment routing domain.

1.3 Segment routing using IPv6 (SRv6)

SR has two data plane instantiations. SR can be used with MPLS (SR-MPLS) without any changes to the forwarding plane. Each segment represents a label, and a segment list represents a label stack. The main advantage of SR-MPLS is its capability of working on existing hardware. Alternatively, SR can be used with IPv6 (known as SRv6). In SRv6, a segment represents an address, and a segment list represents an address list in the SR header[2]. Note that SRv6 can operate with non-source routing nodes and has full interoperability with existing networks. Recently, there has been a significant interest in SRv6 by hardware vendors and network operators[2].

SRv6 inherits all the advantages of SR-MPLS. For instance, since the network state information in transit routers and nodes is removed, the scalability is greatly improved. Furthermore, SRv6 is highly responsive to network changes and hence offers substantial flexibility and agility. Packets outside an SR domain will have zero modifications, enabling end-to-end packet integrity.

SRv6 further simplifies the network since it relies on the native IPv6 header and eliminates MPLS altogether[3]. SRv6 introduced the capability of programming the network by leveraging the IPv6 Extension Headers[4]. With SRv6, it is possible to pack more than IPv6 addresses into a segment ID, and hence, SRv6 enables functionalities beyond routing and traffic steering. Examples of such functionalities include Traffic Engineering, Service Function Chaining, and Virtual Private Networks, etc. This lab focuses only on the basic configuration of SRv6 to perform traffic steering.

1.4 SRv6 on Linux

SRv6 is available in the mainstream Linux kernel since version 4.10[5]. It is activated through the *sysctl* tool. *sysctl* is a tool for dynamically changing parameters in the Linux operating system[6]. It allows users to modify kernel parameters dynamically without rebuilding the Linux kernel.

To enable SRv6 on a Linux node, the following *sysctl* parameters must be activated on all interfaces that will process SR packets:

```
net.ipv6.conf.<iface>.seg6_enabled
```

```
net.ipv6.conf.<iface>.forwarding
```

For example, assuming that a Linux node has two "eth1" and "eth2" interfaces that will process SR packets. The following commands are used to enable SR:

```
sysctl -w net.ipv6.conf.eth1.seg6_enabled=1
```

```
sysctl -w net.ipv6.conf.eth1.forwarding=1
sysctl -w net.ipv6.conf.eth2.seg6_enabled=1
```

```
sysctl -w net.ipv6.conf.eth2.forwarding=1
```

```
sysctl -w net.ipv6.conf.all.seg6_enabled=1
```

```
sysctl -w net.ipv6.conf.all.forwarding=1
```

To add an SR header to packets, the iproute2 tool is used as follows:

```
ip -6 route add <prefix> encap seg6 mode <encapmode> segs <segments> [hmac
<keyid>] dev <device>
```

This command can be summarized as follows:

- `ip -6 route add`: add a new IPv6 route.
- `<prefix>`: IPv6 prefix of the route.
- `encap seg6 mode <encapmode>`: encapsulate matching packets into an IPv6 header if `<encapmode>` is `encap`. If `<encapmode>` is `inline`, the SR header is inserted right after the IPv6 header of the original packet.
- `segs <segments>`: comma-separated list of segments.
- `[hmac <keyid>]`: HMAC key ID, optional.
- `dev <device>`: non-loopback interface name to forward matching packets from.

2 Lab topology

Consider Figure 2. The topology is composed of four routers and two end-hosts. Each interface has been assigned an IPv6 address. This lab uses segment routing to establish the path that packets will take. There are two paths for host h1 to reach host h2. The first path is r1-r2-r4, and the second path consists of the routers r1-r3-r4.

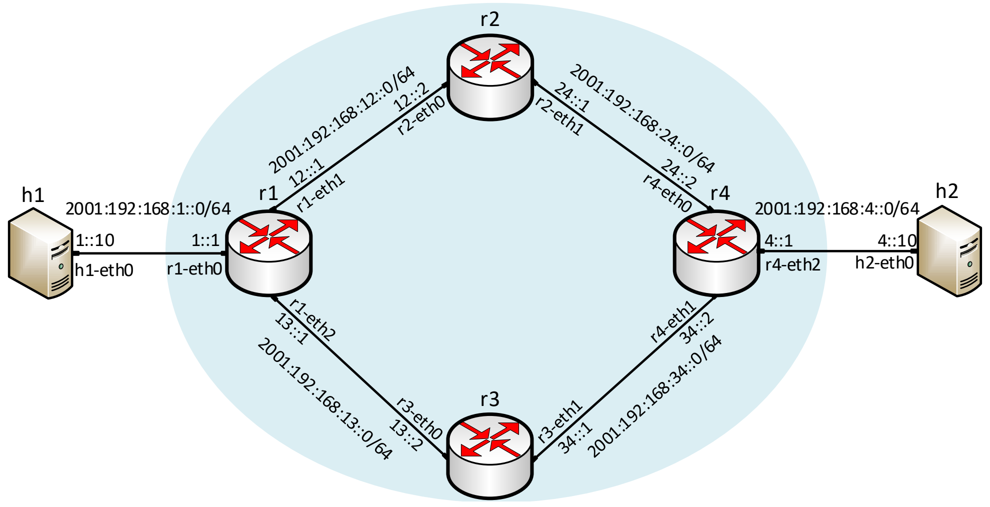


Figure 2. Lab topology.

2.1 Lab settings

Routers and hosts are already configured according to the IP addresses shown in Table 2.

Table 2. Topology information.

Device	Interface	IPv6 Address	Default gateway
r1	r1-eth0	2001:192:168:1::1/64	N/A
	r1-eth1	2001:192:168:12::1/64	N/A
	r1-eth2	2001:192:168:13::1/64	N/A
r2	r2-eth0	2001:192:168:12::2/64	N/A
	r2-eth1	2001:192:168:24::1/64	N/A
r3	r3-eth0	2001:192:168:13::2/64	N/A
	r3-eth1	2001:192:168:34::1/64	N/A
r4	r4-eth0	2001:192:168:24::2/64	N/A
	r4-eth1	2001:192:168:34::2/64	N/A
	r4-eth2	2001:192:168:4::1/64	N/A
h1	h1-eth0	2001:192:168:1::10/64	2001:192:168:1::1
h2	h2-eth0	2001:192:168:4::10/64	2001:192:168:4::1

3 Open topology and load the configuration

Step 1. Start by launching MiniEdit by clicking on desktop's shortcut. When prompted for a password, type `password`.

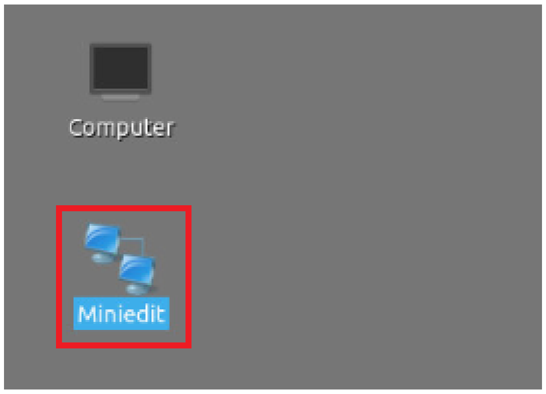


Figure 3. MiniEdit shortcut.

Step 2. On MiniEdit's menu bar, click on *File,* then *open* to load the lab's topology. Locate the *lab9.mn* topology file in the default directory, */home/frr/MPLS_advanced_BGP/lab9* and click on *Open*.

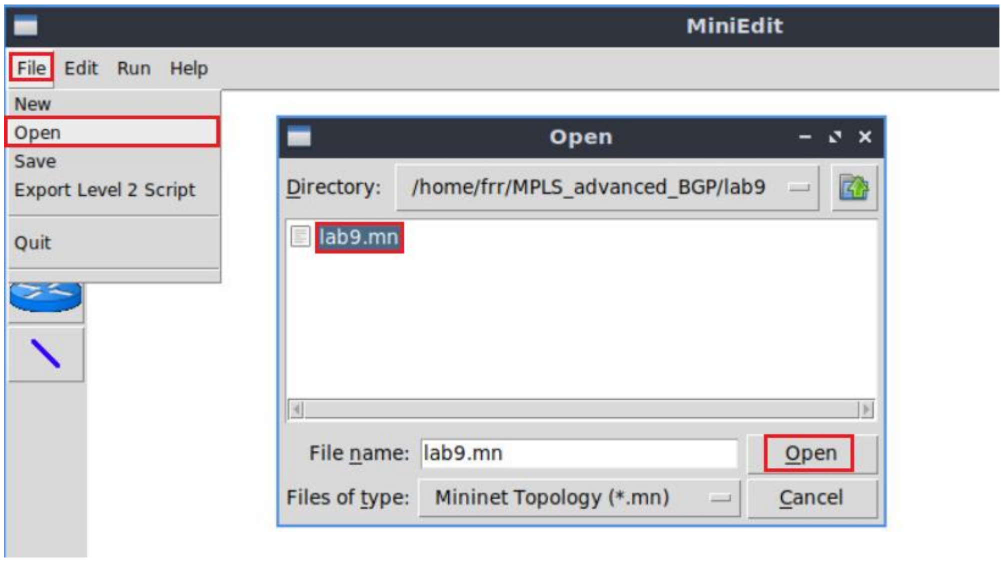


Figure 4. MiniEdit's Open dialog.

At this point, the topology is loaded with all the required network components. You will execute a script that will load the configuration of the routers.

Step 3. Open the Linux terminal.

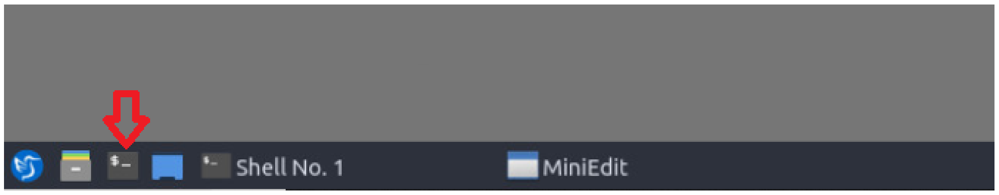
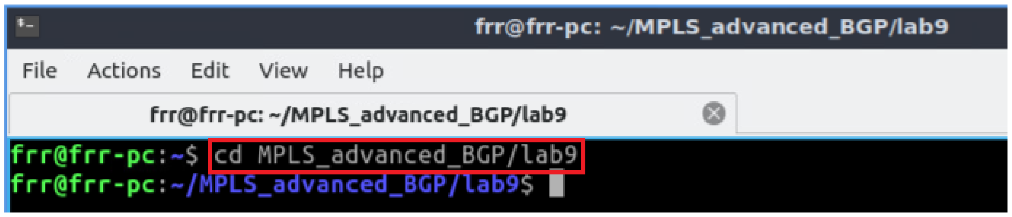


Figure 5. Opening Linux terminal.

Step 4. Click on the Linux's Terminal and navigate into *MPLS_advanced_BGP/lab9* directory by issuing the following command. This folder contains a configuration file and the script responsible for loading the configuration. The configuration file will assign the IP addresses to the routers' interfaces. The `cd` command is short for change directory, followed by an argument that specifies the destination directory.

```
cd MPLS_advanced_BGP/lab9
```

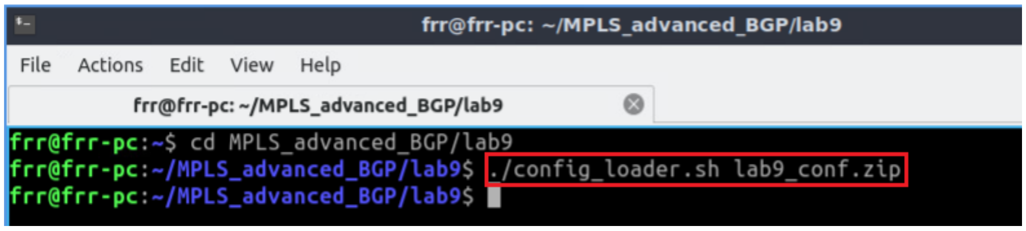

frr@frr-pc: ~/MPLS_advanced_BGP/lab9
File
Actions
Edit
View
Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab9

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab9
frr@frr-pc:~/MPLS_advanced_BGP/lab9$
```

Figure 6. Entering to the *MPLS_advanced_BGP/lab9* directory.

Step 5. To execute the shell script, type the following command. The program's argument corresponds to the configuration zip file that will be loaded in all the routers in the topology.

```
./config_loader.sh lab9_conf.zip
```

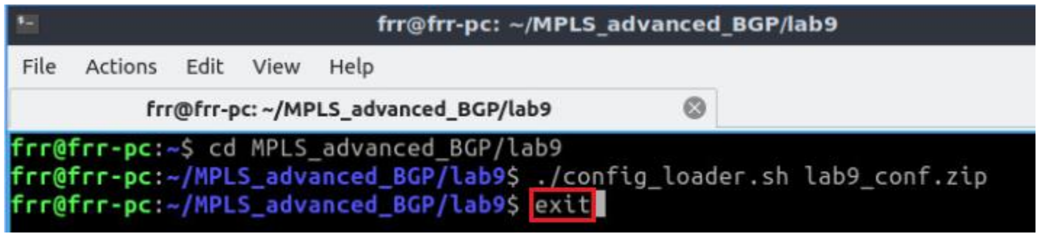

frr@frr-pc: ~/MPLS_advanced_BGP/lab9
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab9

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab9
frr@frr-pc:~/MPLS_advanced_BGP/lab9$ ./config_loader.sh lab9_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab9$
```

Figure 7. Executing the shell script to load the configuration.

Step 6. Type the following command to exit the Linux terminal.

```
exit
```


frr@frr-pc: ~/MPLS_advanced_BGP/lab9
File Actions Edit View Help
frr@frr-pc: ~/MPLS_advanced_BGP/lab9

```
frr@frr-pc:~$ cd MPLS_advanced_BGP/lab9
frr@frr-pc:~/MPLS_advanced_BGP/lab9$ ./config_loader.sh lab9_conf.zip
frr@frr-pc:~/MPLS_advanced_BGP/lab9$ exit
```

Figure 8. Exiting from the Terminal.

Step 7. At this point, the interfaces of hosts h1 and h2 are configured. To proceed with the emulation, click on the *Run* button located on the lower left-hand side.

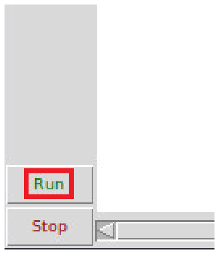


Figure 9. Starting the emulation.

Step 8. Click on Mininet's Terminal, i.e., the one launched when MiniEdit was started.

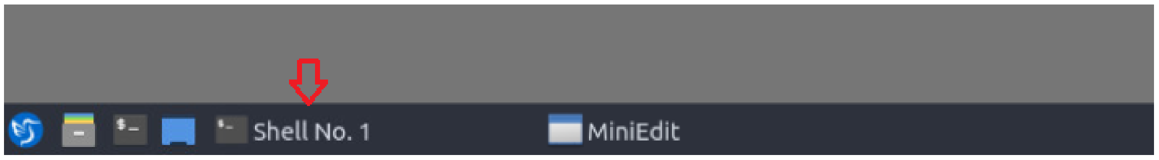
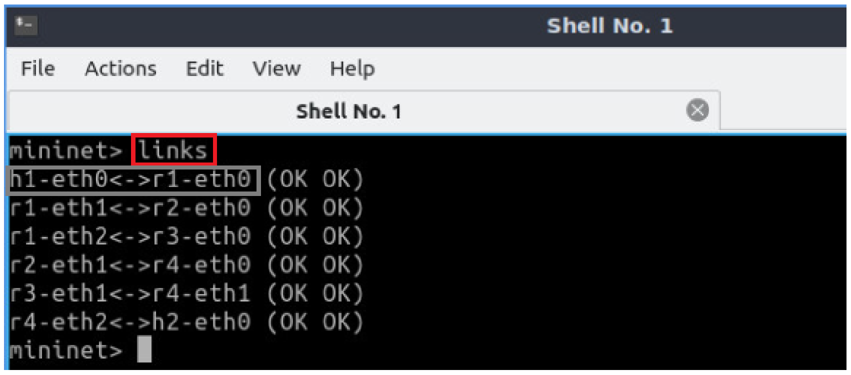


Figure 10. Opening Mininet's Terminal.

Step 9. Issue the following command to display the interface names and connections.

```
links
```



```
mininet> links
h1-eth0<->r1-eth0 (OK OK)
r1-eth1<->r2-eth0 (OK OK)
r1-eth2<->r3-eth0 (OK OK)
r2-eth1<->r4-eth0 (OK OK)
r3-eth1<->r4-eth1 (OK OK)
r4-eth2<->h2-eth0 (OK OK)
mininet>
```

Figure 11. Displaying network interfaces.

In Figure 11, the link displayed within the gray box indicates that interface *eth0* of host h1 connects to interface *eth0* of router r1 (i.e., *h1-eth0<->r1-eth0*).

3.1 Load zebra daemon

Step 1. Hold right-click on router r1 and select *Terminal.*

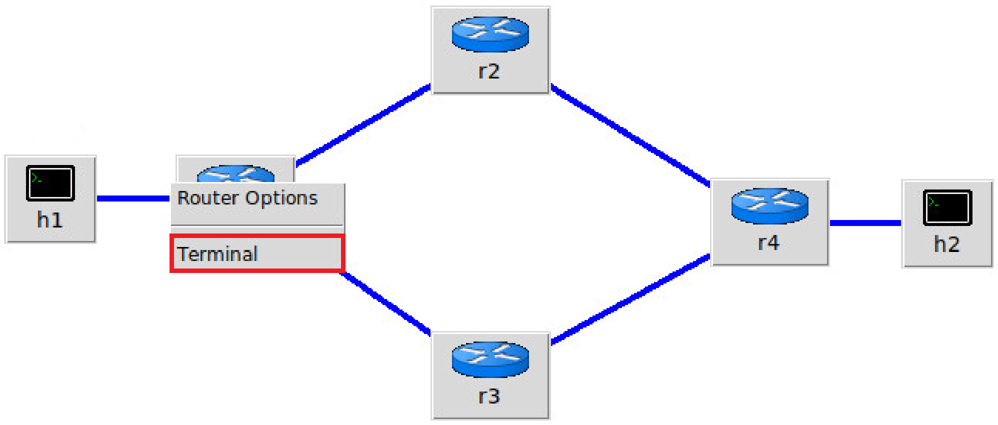
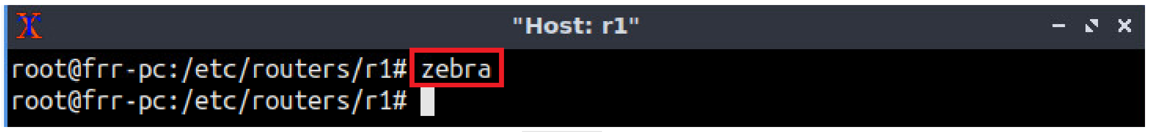


Figure 12. Opening a terminal in router r1.

Step 2. To enable zebra daemon in router r1, type the following command:

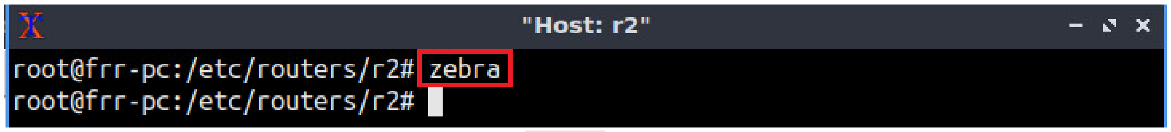
```
zebra
```

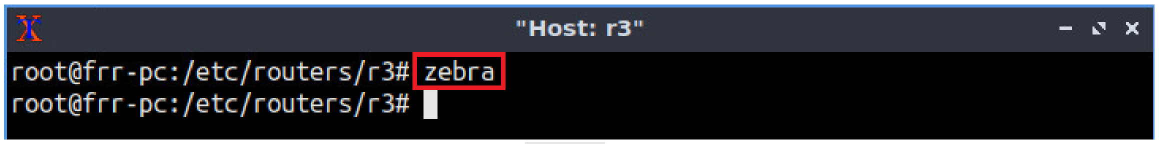


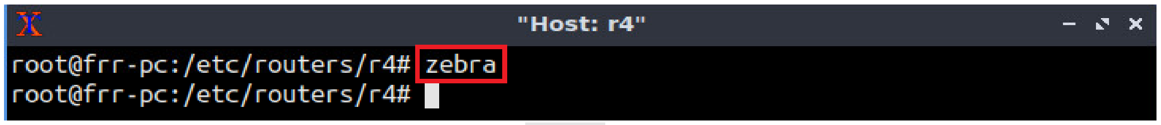
```
root@frr-pc:/etc/routers/r1# zebra
root@frr-pc:/etc/routers/r1#
```

Figure 13. Starting `zebra` daemon in router r1.

Step 3. Proceed similarly in routers r2, r3, and r4. The steps are summarized in the figures below.


```
"Host: r2"
root@frr-pc:/etc/routers/r2# zebra
root@frr-pc:/etc/routers/r2#
```
Figure 14. Starting `zebra` daemon in router r2.


```
"Host: r3"
root@frr-pc:/etc/routers/r3# zebra
root@frr-pc:/etc/routers/r3#
```
Figure 15. Starting `zebra` daemon in router r3.


```
"Host: r4"
root@frr-pc:/etc/routers/r4# zebra
root@frr-pc:/etc/routers/r4#
```
Figure 16. Starting `zebra` daemon in router r4.

3.2 Configure end-hosts with an IPv6 address and the default gateway

Step 1. Hold right-click on host h1 and select *Terminal*. This opens the Terminal of host h1 and allows the execution of commands on that host.

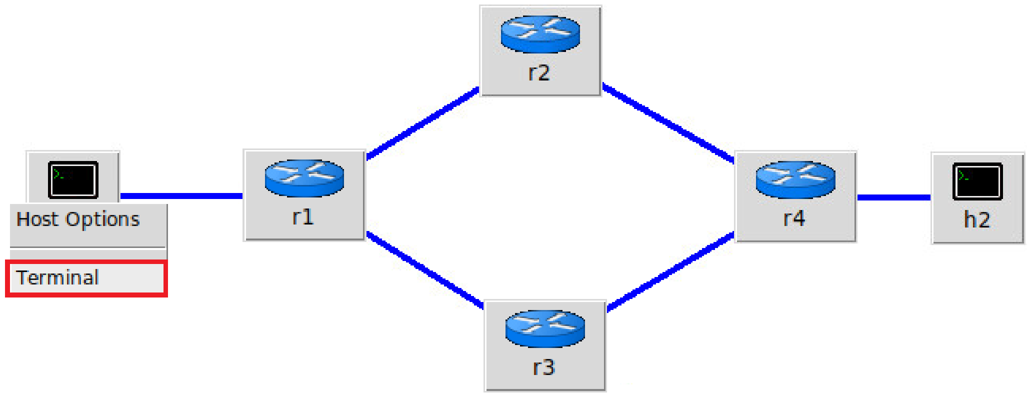
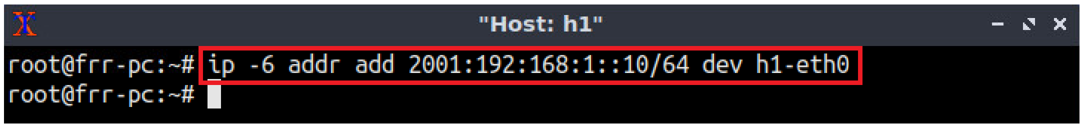


Figure 17. Opening a terminal in host h1.

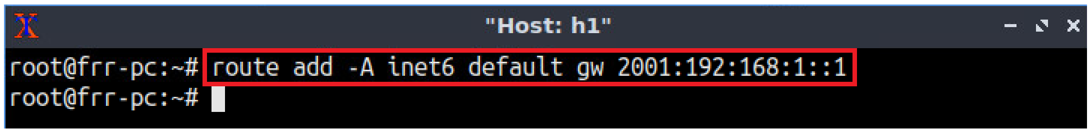
Step 2. In host h1 Terminal, type the following command:

```
ip -6 addr add 2001:192:168:1::10/64 dev h1-eth0
```


```
"Host: h1"
root@frr-pc:~# ip -6 addr add 2001:192:168:1::10/64 dev h1-eth0
root@frr-pc:~#
```
Figure 18. Configure host h1 IPv6 address.

Step 3. Configure the default gateway in host h1 by typing the following command:

```
route add -A inet6 default gw 2001:192:168:1::1
```

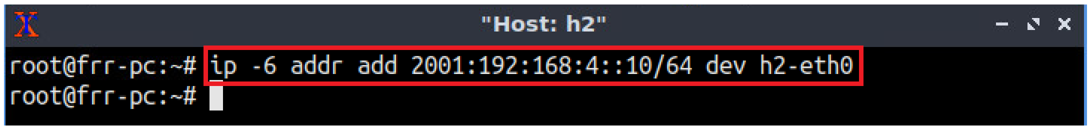


```
root@frr-pc:~# route add -A inet6 default gw 2001:192:168:1::1
root@frr-pc:~#
```

Figure 19. Configuring the default gateway in host h1.

Step 4. Similarly, in host h2 Terminal, type the following command:

```
ip -6 addr add 2001:192:168:4::10/64 dev h2-eth0
```

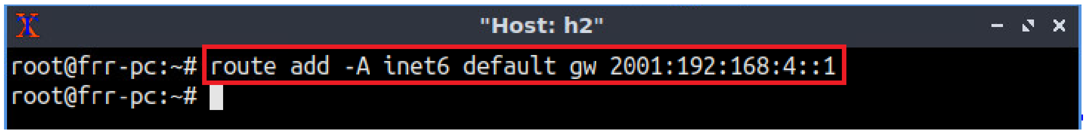


```
root@frr-pc:~# ip -6 addr add 2001:192:168:4::10/64 dev h2-eth0
root@frr-pc:~#
```

Figure 20. Configure host h2 IPv6 address.

Step 5. Similarly, in host h2 Terminal, type the following command:

```
route add -A inet6 default gw 2001:192:168:4::1
```



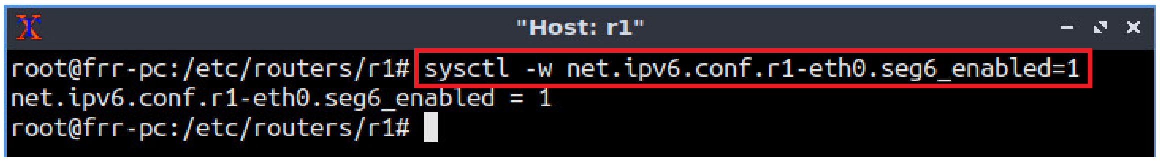
```
root@frr-pc:~# route add -A inet6 default gw 2001:192:168:4::1
root@frr-pc:~#
```

Figure 21. Configuring the default gateway in host h2.

4 Enable SR on all routers

Step 1. In router r1 Terminal, type the following command to enable segment routing in the interface *r1-eth0*.

```
sysctl -w net.ipv6.conf.r1-eth0.seg6_enabled=1
```

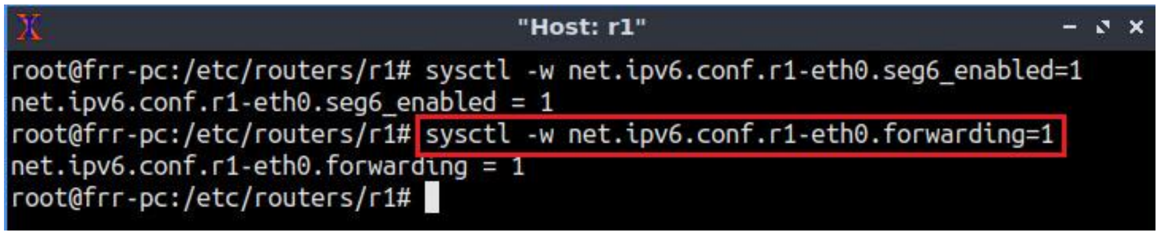


```
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.seg6_enabled=1
net.ipv6.conf.r1-eth0.seg6_enabled = 1
root@frr-pc:/etc/routers/r1#
```

Figure 22. Enabling segment routing in the interface r1-eth0.

Step 2. Enable packet forwarding in the interface *r1-eth0* by typing the following command.

```
sysctl -w net.ipv6.conf.r1-eth0.forwarding=1
```

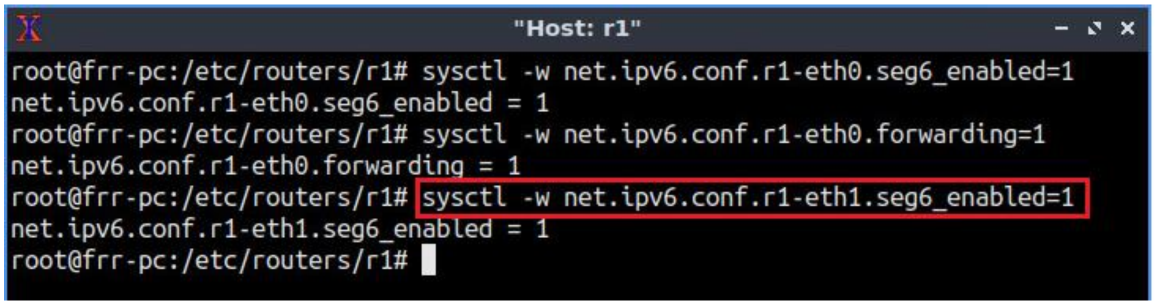


```
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.seg6_enabled=1
net.ipv6.conf.r1-eth0.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.forwarding=1
net.ipv6.conf.r1-eth0.forwarding = 1
root@frr-pc:/etc/routers/r1#
```

Figure 23. Enabling packet forwarding in the interface r1-eth0.

Step 3. Enable segment routing in the interface *r1-eth1* by typing the following command:

```
sysctl -w net.ipv6.conf.r1-eth1.seg6_enabled=1
```

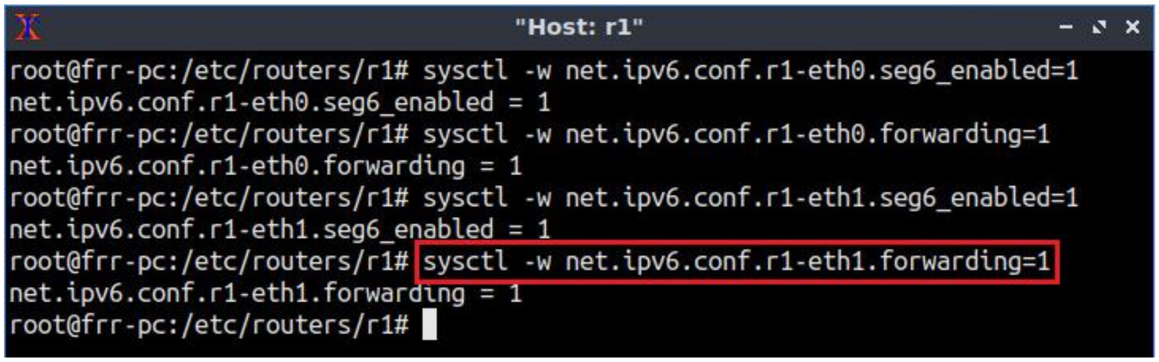


```
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.seg6_enabled=1
net.ipv6.conf.r1-eth0.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.forwarding=1
net.ipv6.conf.r1-eth0.forwarding = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth1.seg6_enabled=1
net.ipv6.conf.r1-eth1.seg6_enabled = 1
root@frr-pc:/etc/routers/r1#
```

Figure 24. Enabling segment routing in the interface r1-eth1.

Step 4. Enable packet forwarding in the interface *r1-eth1* by typing the following command:

```
sysctl -w net.ipv6.conf.r1-eth1.forwarding=1
```

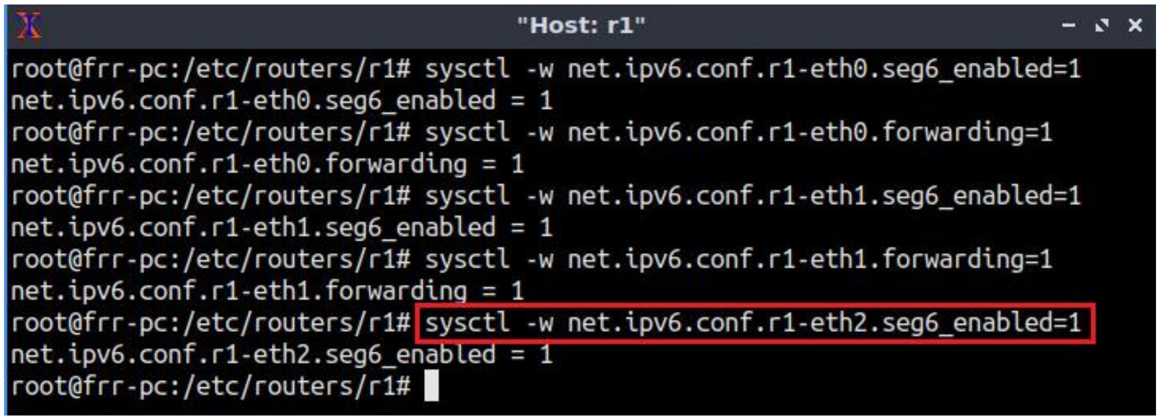


```
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.seg6_enabled=1
net.ipv6.conf.r1-eth0.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.forwarding=1
net.ipv6.conf.r1-eth0.forwarding = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth1.seg6_enabled=1
net.ipv6.conf.r1-eth1.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth1.forwarding=1
net.ipv6.conf.r1-eth1.forwarding = 1
root@frr-pc:/etc/routers/r1#
```

Figure 25. Enabling packet forwarding in the interface r1-eth1.

Step 5. Enable segment routing in the interface *r1-eth2* by typing the following command:

```
sysctl -w net.ipv6.conf.r1-eth2.seg6_enabled=1
```

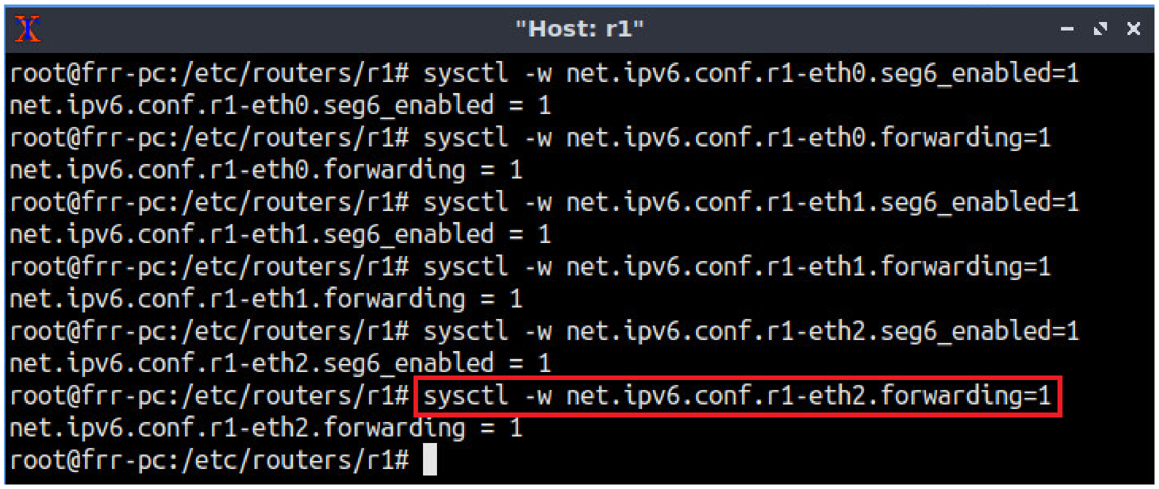


```
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.seg6_enabled=1
net.ipv6.conf.r1-eth0.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.forwarding=1
net.ipv6.conf.r1-eth0.forwarding = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth1.seg6_enabled=1
net.ipv6.conf.r1-eth1.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth1.forwarding=1
net.ipv6.conf.r1-eth1.forwarding = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth2.seg6_enabled=1
net.ipv6.conf.r1-eth2.seg6_enabled = 1
root@frr-pc:/etc/routers/r1#
```

Figure 26. Enabling segment routing in the interface r1-eth2.

Step 6. Enable packet forwarding in the interface *r1-eth2* by typing the following command:

```
sysctl -w net.ipv6.conf.r1-eth2.forwarding=1
```

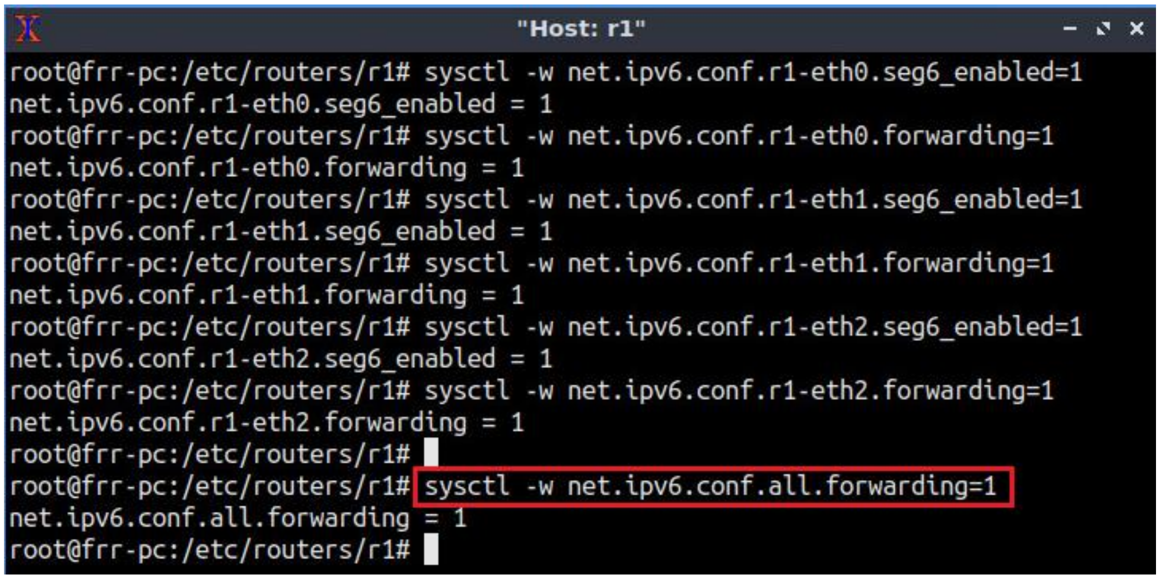


```
"Host: r1"
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.seg6_enabled=1
net.ipv6.conf.r1-eth0.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.forwarding=1
net.ipv6.conf.r1-eth0.forwarding = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth1.seg6_enabled=1
net.ipv6.conf.r1-eth1.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth1.forwarding=1
net.ipv6.conf.r1-eth1.forwarding = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth2.seg6_enabled=1
net.ipv6.conf.r1-eth2.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth2.forwarding=1
net.ipv6.conf.r1-eth2.forwarding = 1
root@frr-pc:/etc/routers/r1# 
```

Figure 27. Enabling packet forwarding in the interface r1-eth2.

Step 7. Enable IPv6 forwarding in router r1 by typing the following command:

```
sysctl -w net.ipv6.conf.all.forwarding=1
```

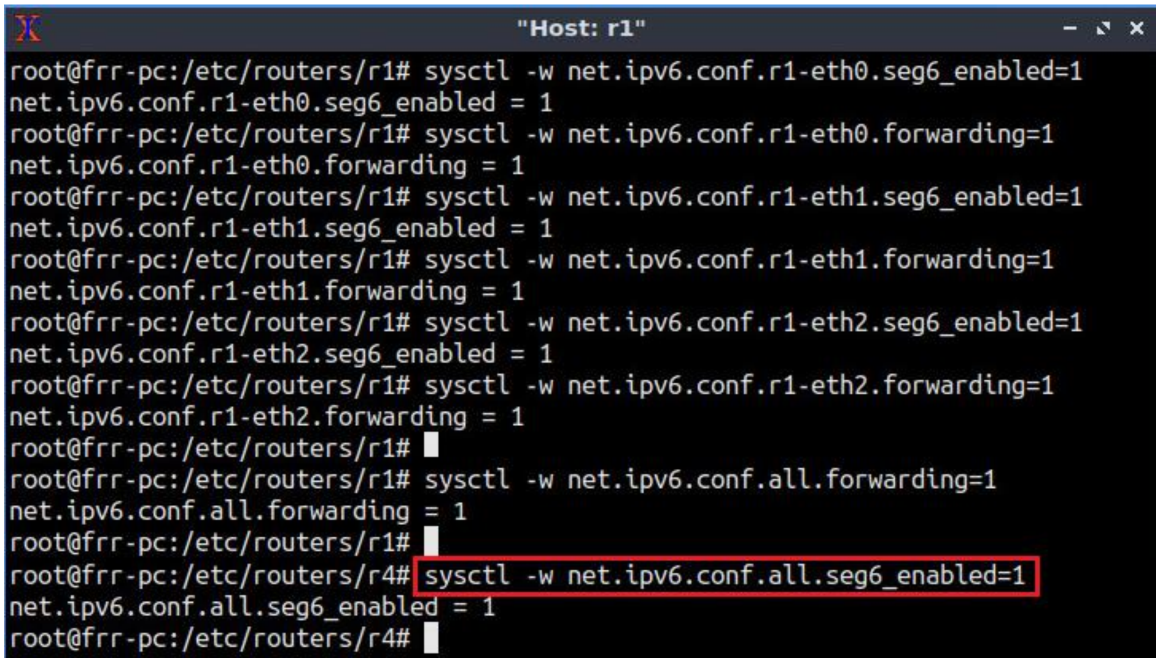


```
"Host: r1"
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.seg6_enabled=1
net.ipv6.conf.r1-eth0.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.forwarding=1
net.ipv6.conf.r1-eth0.forwarding = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth1.seg6_enabled=1
net.ipv6.conf.r1-eth1.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth1.forwarding=1
net.ipv6.conf.r1-eth1.forwarding = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth2.seg6_enabled=1
net.ipv6.conf.r1-eth2.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth2.forwarding=1
net.ipv6.conf.r1-eth2.forwarding = 1
root@frr-pc:/etc/routers/r1# 
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.all.forwarding=1
net.ipv6.conf.all.forwarding = 1
root@frr-pc:/etc/routers/r1# 
```

Figure 28. Enabling packet forwarding in router r1.

Step 8. Enable segment routing in router r1 by typing the following command:

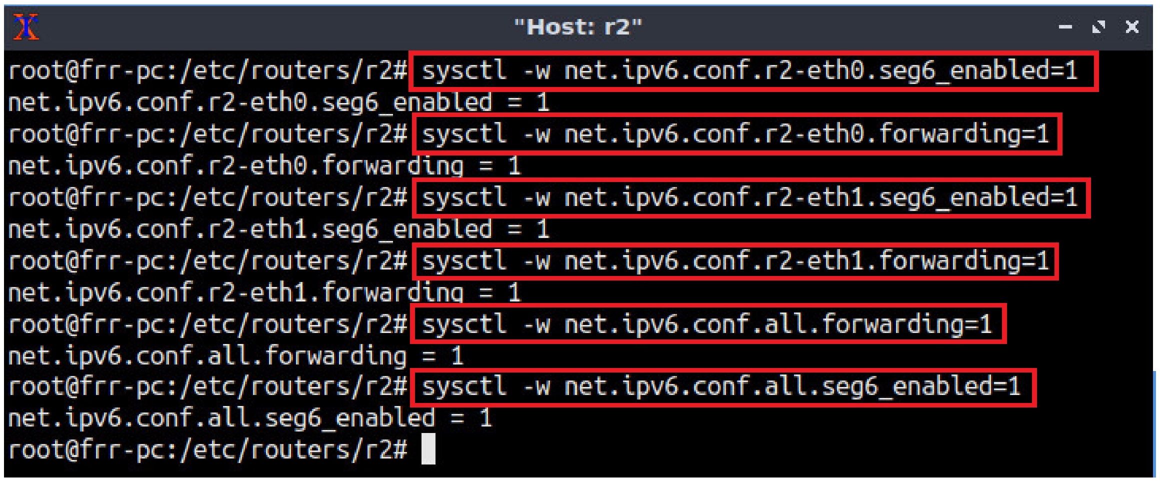
```
sysctl -w net.ipv6.conf.all.seg6_enabled=1
```



```
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.seg6_enabled=1
net.ipv6.conf.r1-eth0.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth0.forwarding=1
net.ipv6.conf.r1-eth0.forwarding = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth1.seg6_enabled=1
net.ipv6.conf.r1-eth1.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth1.forwarding=1
net.ipv6.conf.r1-eth1.forwarding = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth2.seg6_enabled=1
net.ipv6.conf.r1-eth2.seg6_enabled = 1
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.r1-eth2.forwarding=1
net.ipv6.conf.r1-eth2.forwarding = 1
root@frr-pc:/etc/routers/r1# 
root@frr-pc:/etc/routers/r1# sysctl -w net.ipv6.conf.all.forwarding=1
net.ipv6.conf.all.forwarding = 1
root@frr-pc:/etc/routers/r1# 
root@frr-pc:/etc/routers/r4# sysctl -w net.ipv6.conf.all.seg6_enabled=1
net.ipv6.conf.all.seg6_enabled = 1
root@frr-pc:/etc/routers/r4# 
```

Figure 29. Enabling segment routing in router r1.

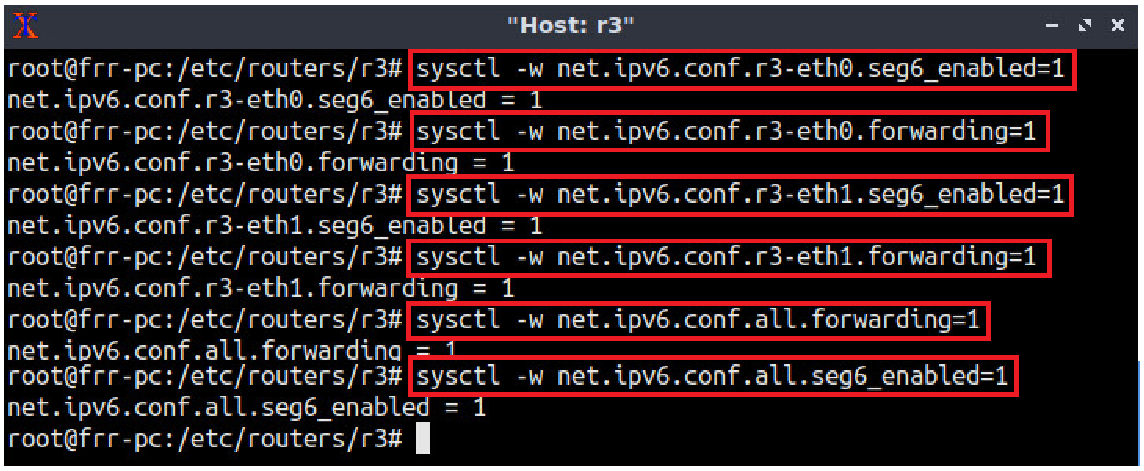
Step 9. Proceed similarly in router r2, the configuration is summarized in the following figure:



```
root@frr-pc:/etc/routers/r2# sysctl -w net.ipv6.conf.r2-eth0.seg6_enabled=1
net.ipv6.conf.r2-eth0.seg6_enabled = 1
root@frr-pc:/etc/routers/r2# sysctl -w net.ipv6.conf.r2-eth0.forwarding=1
net.ipv6.conf.r2-eth0.forwarding = 1
root@frr-pc:/etc/routers/r2# sysctl -w net.ipv6.conf.r2-eth1.seg6_enabled=1
net.ipv6.conf.r2-eth1.seg6_enabled = 1
root@frr-pc:/etc/routers/r2# sysctl -w net.ipv6.conf.r2-eth1.forwarding=1
net.ipv6.conf.r2-eth1.forwarding = 1
root@frr-pc:/etc/routers/r2# sysctl -w net.ipv6.conf.all.forwarding=1
net.ipv6.conf.all.forwarding = 1
root@frr-pc:/etc/routers/r2# sysctl -w net.ipv6.conf.all.seg6_enabled=1
net.ipv6.conf.all.seg6_enabled = 1
root@frr-pc:/etc/routers/r2# 
```

Figure 30. Configuration summary of router r2.

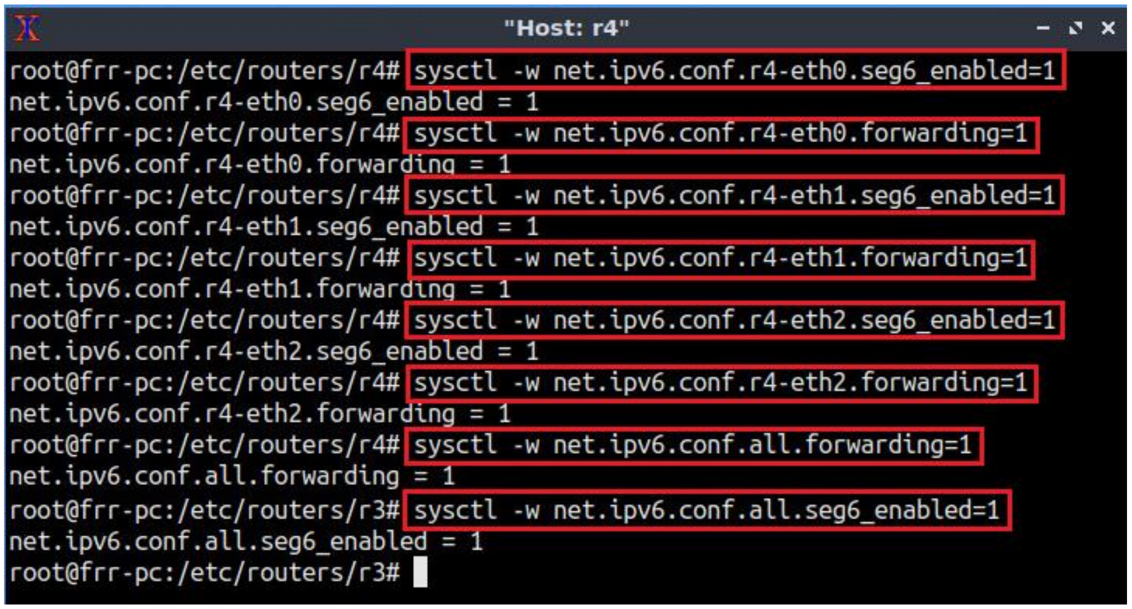
Step 10. Proceed similarly in router r3, the configuration is summarized in the following figure:



```
"Host: r3"
root@frr-pc:/etc/routers/r3# sysctl -w net.ipv6.conf.r3-eth0.seg6_enabled=1
net.ipv6.conf.r3-eth0.seg6_enabled = 1
root@frr-pc:/etc/routers/r3# sysctl -w net.ipv6.conf.r3-eth0.forwarding=1
net.ipv6.conf.r3-eth0.forwarding = 1
root@frr-pc:/etc/routers/r3# sysctl -w net.ipv6.conf.r3-eth1.seg6_enabled=1
net.ipv6.conf.r3-eth1.seg6_enabled = 1
root@frr-pc:/etc/routers/r3# sysctl -w net.ipv6.conf.r3-eth1.forwarding=1
net.ipv6.conf.r3-eth1.forwarding = 1
root@frr-pc:/etc/routers/r3# sysctl -w net.ipv6.conf.all.forwarding=1
net.ipv6.conf.all.forwarding = 1
root@frr-pc:/etc/routers/r3# sysctl -w net.ipv6.conf.all.seg6_enabled=1
net.ipv6.conf.all.seg6_enabled = 1
root@frr-pc:/etc/routers/r3#
```

Figure 31. Configuration summary of router r3.

Step 11. Proceed similarly in router r4, the configuration is summarized in the following figure:



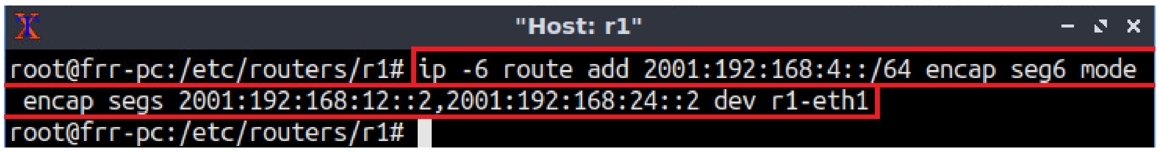
```
"Host: r4"
root@frr-pc:/etc/routers/r4# sysctl -w net.ipv6.conf.r4-eth0.seg6_enabled=1
net.ipv6.conf.r4-eth0.seg6_enabled = 1
root@frr-pc:/etc/routers/r4# sysctl -w net.ipv6.conf.r4-eth0.forwarding=1
net.ipv6.conf.r4-eth0.forwarding = 1
root@frr-pc:/etc/routers/r4# sysctl -w net.ipv6.conf.r4-eth1.seg6_enabled=1
net.ipv6.conf.r4-eth1.seg6_enabled = 1
root@frr-pc:/etc/routers/r4# sysctl -w net.ipv6.conf.r4-eth1.forwarding=1
net.ipv6.conf.r4-eth1.forwarding = 1
root@frr-pc:/etc/routers/r4# sysctl -w net.ipv6.conf.r4-eth2.seg6_enabled=1
net.ipv6.conf.r4-eth2.seg6_enabled = 1
root@frr-pc:/etc/routers/r4# sysctl -w net.ipv6.conf.r4-eth2.forwarding=1
net.ipv6.conf.r4-eth2.forwarding = 1
root@frr-pc:/etc/routers/r4# sysctl -w net.ipv6.conf.all.forwarding=1
net.ipv6.conf.all.forwarding = 1
root@frr-pc:/etc/routers/r3# sysctl -w net.ipv6.conf.all.seg6_enabled=1
net.ipv6.conf.all.seg6_enabled = 1
root@frr-pc:/etc/routers/r3#
```

Figure 32. Configuration summary of router r4.

4.1 Configure SR path

Step 1. In router r1, configure the encapsulation mode and the route where the packets will be forwarded by typing the following command:

```
ip -6 route add 2001:192:168:4::/64 encap seg6 mode encap segs
2001:192:168:12::2,2001:192:168:24::2 dev r1-eth1
```

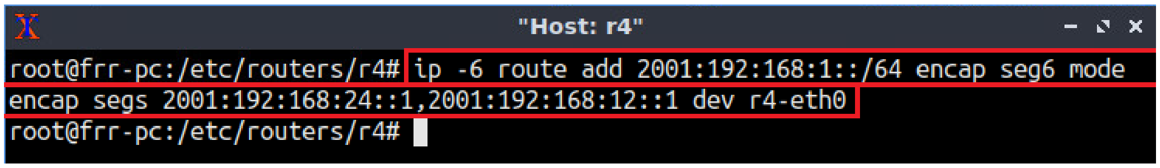


```
"Host: r1"
root@frr-pc:/etc/routers/r1# ip -6 route add 2001:192:168:4::/64 encap seg6 mode
 encap segs 2001:192:168:12::2,2001:192:168:24::2 dev r1-eth1
root@frr-pc:/etc/routers/r1#
```

Figure 33. Specifying route configuration in router r1.

Step 2. Configure the encapsulation mode and the route where the packets will be forwarded by typing the following command:

```
ip -6 route add 2001:192:168:1::/64 encap seg6 mode encap segs
2001:192:168:24::1,2001:192:168:12::1 dev r4-eth0
```



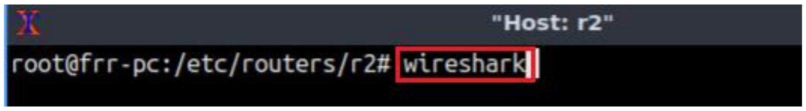
```
"Host: r4"
root@frr-pc:/etc/routers/r4# ip -6 route add 2001:192:168:1::/64 encap seg6 mode
encap segs 2001:192:168:24::1,2001:192:168:12::1 dev r4-eth0
root@frr-pc:/etc/routers/r4#
```

Figure 34. Specifying route configuration in router r4.

4.2 Verify the configuration

Step 1. In router r2's Terminal, start Wireshark dissector by issuing the following command. A new window will emerge.

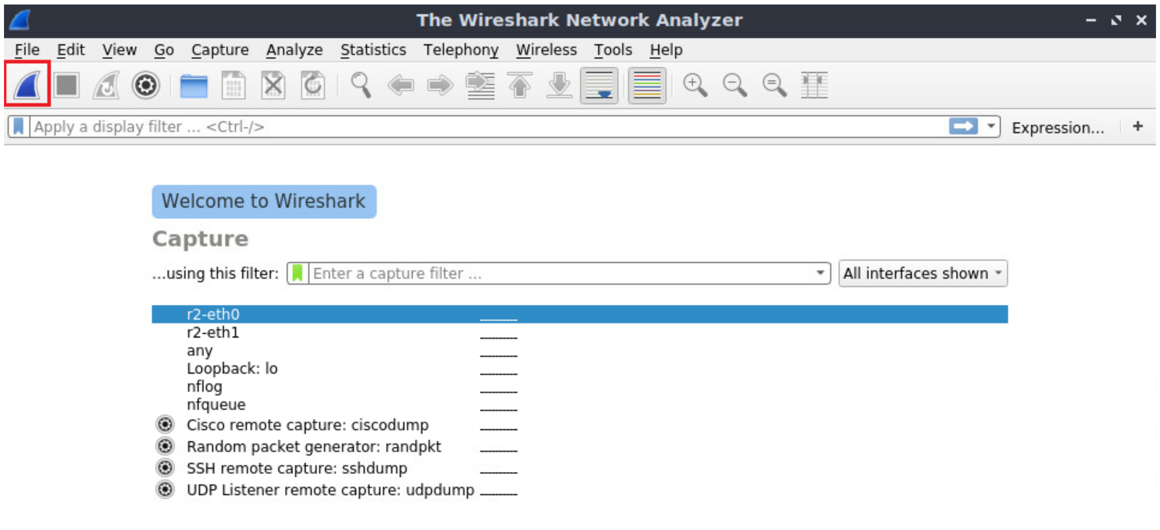
```
wireshark
```



```
"Host: r2"
root@frr-pc:/etc/routers/r2# wireshark
```

Figure 35. Starting packet capturing with Wireshark.

Step 2. Click on the icon located on the upper left-hand side to start capturing packets on the interface *r2-eth0*.



The Wireshark Network Analyzer

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/> Expression... +

Welcome to Wireshark

Capture

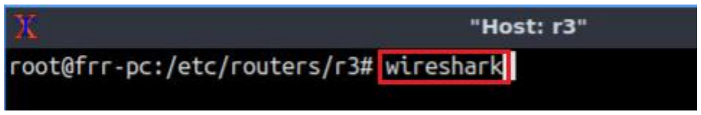
...using this filter: Enter a capture filter ... All interfaces shown

r2-eth0
r2-eth1
any
Loopback: lo
nflog
nfqueue
Cisco remote capture: ciscodump
Random packet generator: randpkt
SSH remote capture: sshdump
UDP Listener remote capture: udpdump

Figure 36. Starting packet capture on r2-eth0.

Step 3. In router r3's Terminal, start Wireshark dissector by issuing the following command.

```
wireshark
```



```
root@frr-pc:/etc/routers/r3# wireshark
```

Figure 37. Starting packet capturing with Wireshark.

Step 4. Click on the icon located on the upper left-hand side to start capturing packets on the interface *r3-eth1*.

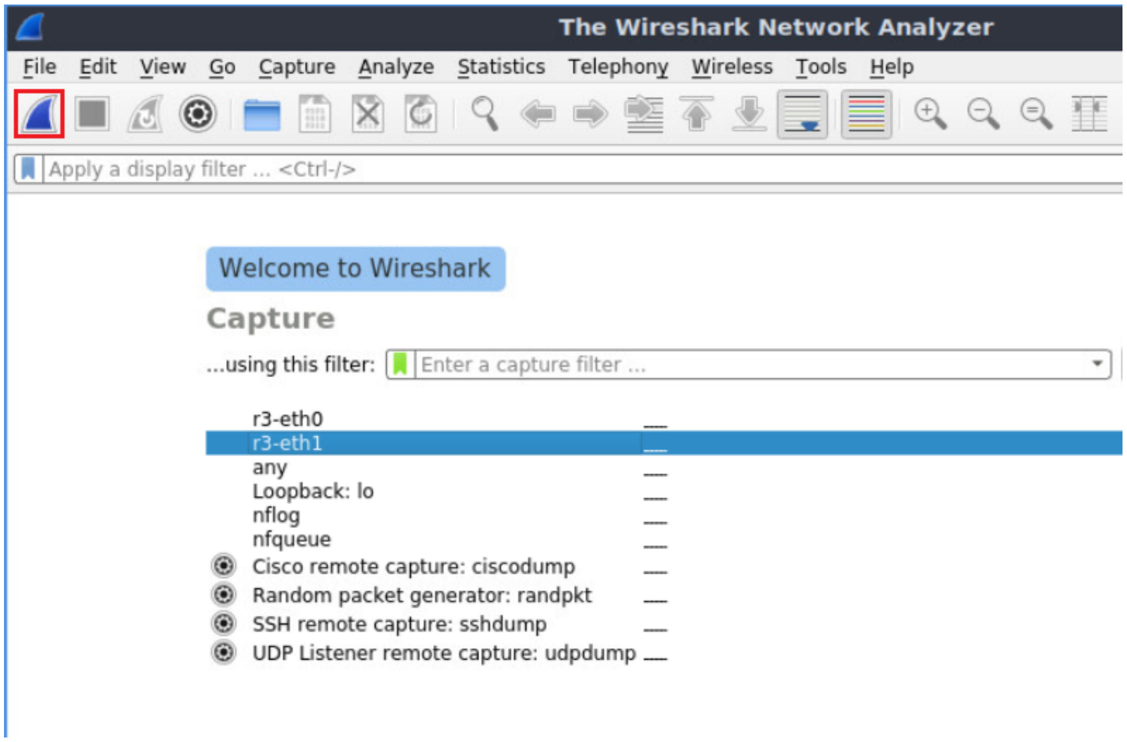
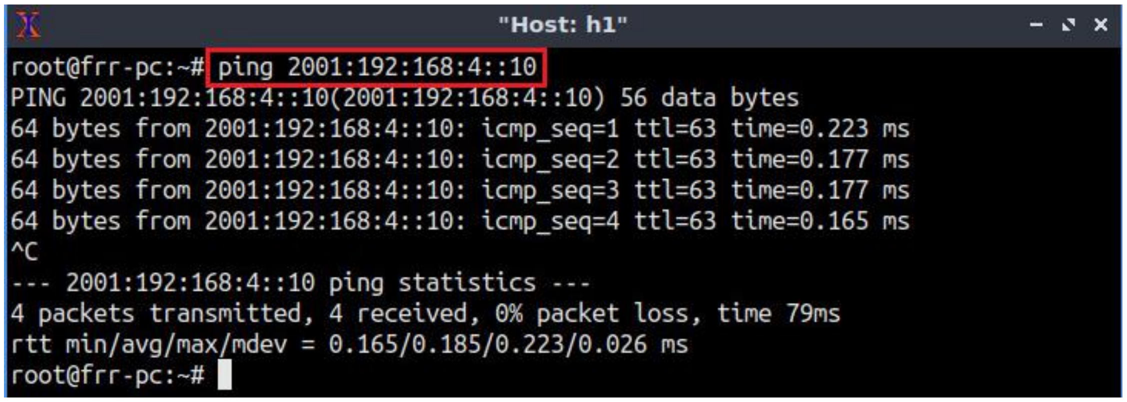

The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Capture
...using this filter:
Enter a capture filter ...
r3-eth0
r3-eth1
any
Loopback: lo
nflog
nfqueue
Cisco remote capture: ciscodump
Random packet generator: randpkt
SSH remote capture: sshdump
UDP Listener remote capture: udpdump

Figure 38. Starting packet capture.

Step 5. Navigate to the host h1 terminal and perform a connectivity test by typing the following command, then press `ctrl+c` to stop the test.

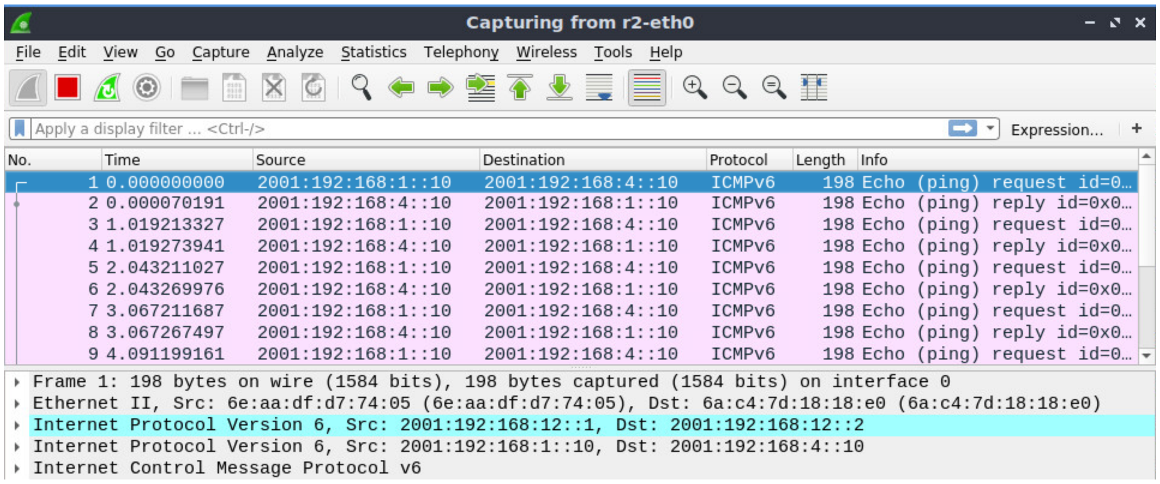
```
ping 2001:192:168:4::10
```



```
root@frr-pc:~# ping 2001:192:168:4::10
PING 2001:192:168:4::10(2001:192:168:4::10) 56 data bytes
64 bytes from 2001:192:168:4::10: icmp_seq=1 ttl=63 time=0.223 ms
64 bytes from 2001:192:168:4::10: icmp_seq=2 ttl=63 time=0.177 ms
64 bytes from 2001:192:168:4::10: icmp_seq=3 ttl=63 time=0.177 ms
64 bytes from 2001:192:168:4::10: icmp_seq=4 ttl=63 time=0.165 ms
^C
--- 2001:192:168:4::10 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 79ms
rtt min/avg/max/mdev = 0.165/0.185/0.223/0.026 ms
root@frr-pc:~#
```

Figure 39. Performing a connectivity test between host h1 and host h2.

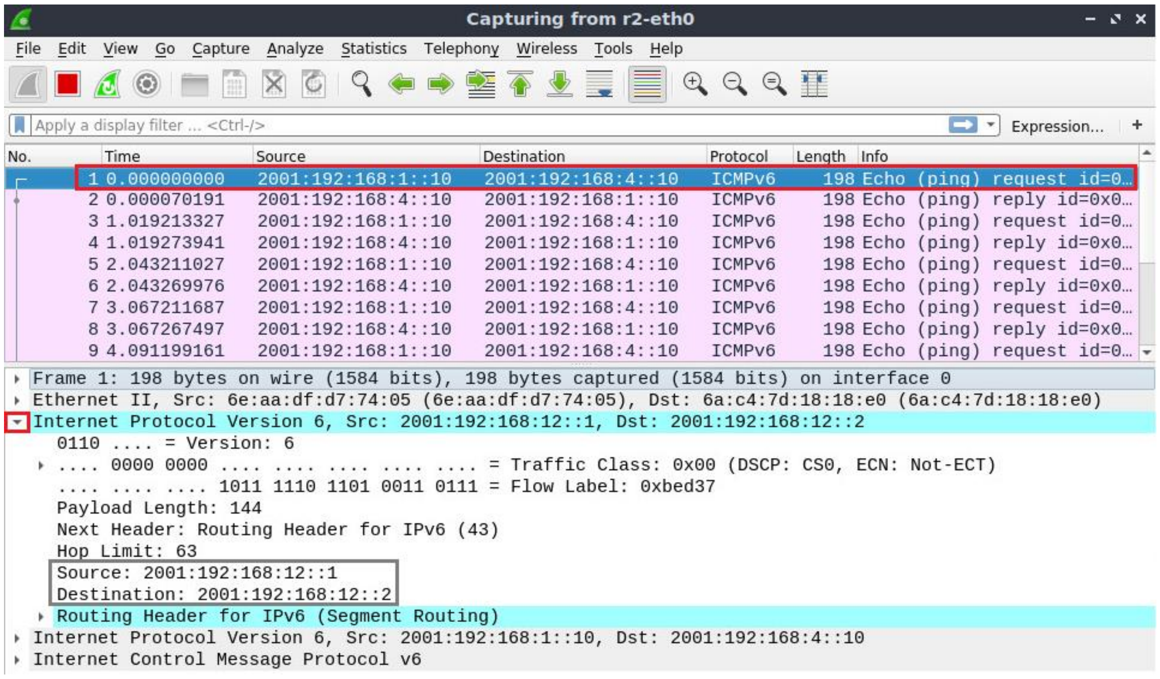
Step 6. Verify the packet capturing for the interface *r2-eth0* on Wireshark. Note that we are capturing on the interface before the router is modifying the packet. You will observe that the packets are passing across router r2 and not by router r3.


Capturing from r2-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.  Time         Source               Destination          Protocol  Length Info
   1 0.000000000  2001:192:168:1::10   2001:192:168:4::10   ICMPv6    198 Echo (ping) request id=0…
   2 0.000070191  2001:192:168:4::10   2001:192:168:1::10   ICMPv6    198 Echo (ping) reply id=0x0…
   3 1.019213327  2001:192:168:1::10   2001:192:168:4::10   ICMPv6    198 Echo (ping) request id=0…
   4 1.019273941  2001:192:168:4::10   2001:192:168:1::10   ICMPv6    198 Echo (ping) reply id=0x0…
   5 2.043211027  2001:192:168:1::10   2001:192:168:4::10   ICMPv6    198 Echo (ping) request id=0…
   6 2.043269976  2001:192:168:4::10   2001:192:168:1::10   ICMPv6    198 Echo (ping) reply id=0x0…
   7 3.067211687  2001:192:168:1::10   2001:192:168:4::10   ICMPv6    198 Echo (ping) request id=0…
   8 3.067267497  2001:192:168:4::10   2001:192:168:1::10   ICMPv6    198 Echo (ping) reply id=0x0…
   9 4.091199161  2001:192:168:1::10   2001:192:168:4::10   ICMPv6    198 Echo (ping) request id=0…

Frame 1: 198 bytes on wire (1584 bits), 198 bytes captured (1584 bits) on interface 0
Ethernet II, Src: 6e:aa:df:d7:74:05 (6e:aa:df:d7:74:05), Dst: 6a:c4:7d:18:18:e0 (6a:c4:7d:18:18:e0)
Internet Protocol Version 6, Src: 2001:192:168:12::1, Dst: 2001:192:168:12::2
Internet Protocol Version 6, Src: 2001:192:168:1::10, Dst: 2001:192:168:4::10
Internet Control Message Protocol v6
```

Figure 40. Packets are passing across router r2.

Step 7. Open the first tab corresponding to *Internet Protocol Version 6* as depicted in the figure below.

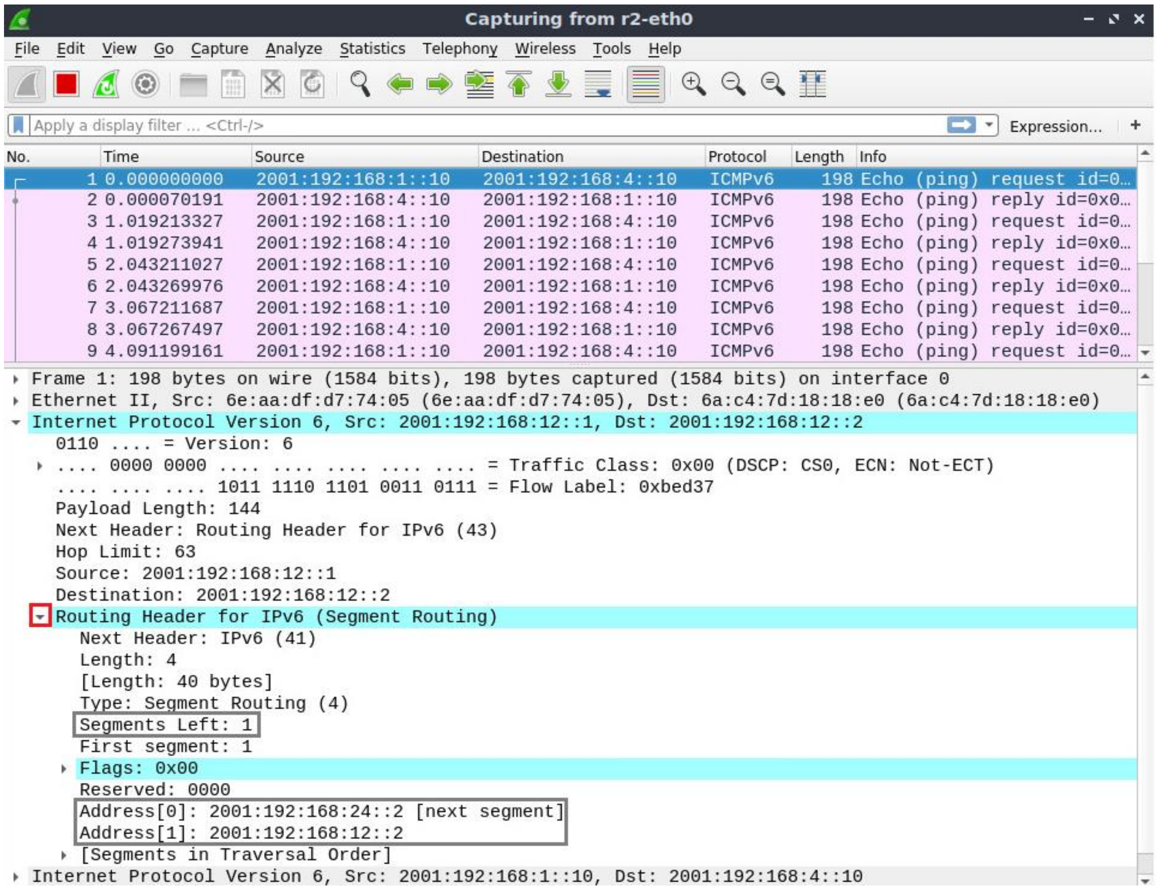

Capturing from r2-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.  Time         Source               Destination          Protocol  Length Info
   1 0.000000000  2001:192:168:1::10   2001:192:168:4::10   ICMPv6    198 Echo (ping) request id=0…
   2 0.000070191  2001:192:168:4::10   2001:192:168:1::10   ICMPv6    198 Echo (ping) reply id=0x0…
   3 1.019213327  2001:192:168:1::10   2001:192:168:4::10   ICMPv6    198 Echo (ping) request id=0…
   4 1.019273941  2001:192:168:4::10   2001:192:168:1::10   ICMPv6    198 Echo (ping) reply id=0x0…
   5 2.043211027  2001:192:168:1::10   2001:192:168:4::10   ICMPv6    198 Echo (ping) request id=0…
   6 2.043269976  2001:192:168:4::10   2001:192:168:1::10   ICMPv6    198 Echo (ping) reply id=0x0…
   7 3.067211687  2001:192:168:1::10   2001:192:168:4::10   ICMPv6    198 Echo (ping) request id=0…
   8 3.067267497  2001:192:168:4::10   2001:192:168:1::10   ICMPv6    198 Echo (ping) reply id=0x0…
   9 4.091199161  2001:192:168:1::10   2001:192:168:4::10   ICMPv6    198 Echo (ping) request id=0…

Frame 1: 198 bytes on wire (1584 bits), 198 bytes captured (1584 bits) on interface 0
Ethernet II, Src: 6e:aa:df:d7:74:05 (6e:aa:df:d7:74:05), Dst: 6a:c4:7d:18:18:e0 (6a:c4:7d:18:18:e0)
Internet Protocol Version 6, Src: 2001:192:168:12::1, Dst: 2001:192:168:12::2
  0110 .... = Version: 6
  .... 0000 0000 .... .... .... .... .... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)
  .... .... .... 1011 1110 1101 0011 0111 = Flow Label: 0xbed37
  Payload Length: 144
  Next Header: Routing Header for IPv6 (43)
  Hop Limit: 63
  Source: 2001:192:168:12::1
  Destination: 2001:192:168:12::2
  Routing Header for IPv6 (Segment Routing)
Internet Protocol Version 6, Src: 2001:192:168:1::10, Dst: 2001:192:168:4::10
Internet Control Message Protocol v6
```

Figure 41. Displaying IPv6 header information.

The user will visualize the source IP address 2001:192:168:12::1 in the segment routing header corresponding to the egress interface *r1-eth1* of router r1. The destination IP address 2001:192:168:12::2 in the segment routing header corresponds to the ingress interface *r2-eth0* of router r2. Note how the IPv6 addresses 2001:192:168:1::10 and 2001:192:168:4::10 that correspond to hosts h1 and h2 in the original packet headers remain unchanged.

Step 8. Click on the tab corresponding to *Routing Header for IPv6 (Segment Routing),* as shown in the figure below.

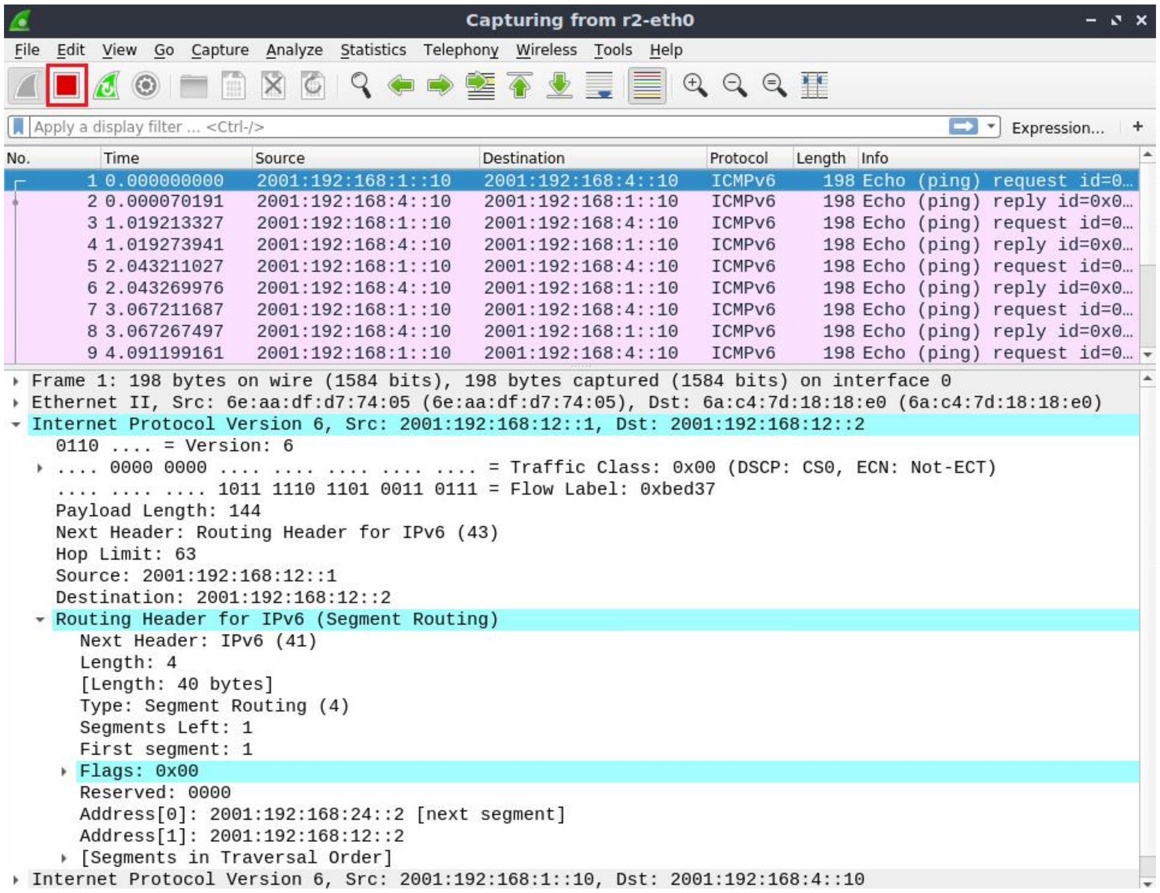

Capturing from r2-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Expression…

```
No. Time         Source              Destination         Protocol Length Info
  1 0.000000000  2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
  2 0.000070191  2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
  3 1.019213327  2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
  4 1.019273941  2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
  5 2.043211027  2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
  6 2.043269976  2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
  7 3.067211687  2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
  8 3.067267497  2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
  9 4.091199161  2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…

▸ Frame 1: 198 bytes on wire (1584 bits), 198 bytes captured (1584 bits) on interface 0
▸ Ethernet II, Src: 6e:aa:df:d7:74:05 (6e:aa:df:d7:74:05), Dst: 6a:c4:7d:18:18:e0 (6a:c4:7d:18:18:e0)
▾ Internet Protocol Version 6, Src: 2001:192:168:12::1, Dst: 2001:192:168:12::2
    0110 .... = Version: 6
  ▸ .... 0000 0000 .... .... .... .... .... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)
    .... .... .... 1011 1110 1101 0011 0111 = Flow Label: 0xbed37
    Payload Length: 144
    Next Header: Routing Header for IPv6 (43)
    Hop Limit: 63
    Source: 2001:192:168:12::1
    Destination: 2001:192:168:12::2
  ▾ Routing Header for IPv6 (Segment Routing)
      Next Header: IPv6 (41)
      Length: 4
      [Length: 40 bytes]
      Type: Segment Routing (4)
      Segments Left: 1
      First segment: 1
    ▸ Flags: 0x00
      Reserved: 0000
      Address[0]: 2001:192:168:24::2 [next segment]
      Address[1]: 2001:192:168:12::2
    ▸ [Segments in Traversal Order]
▸ Internet Protocol Version 6, Src: 2001:192:168:1::10, Dst: 2001:192:168:4::10
```

Figure 42. Displaying segment routing hops IP addresses.

The information displayed in the figure above shows the segment routing path. *Address[0]: 2001:192:168:24::2* is the next segment and corresponds to the next-hop IP address (router r4). *Address[1]: 2001:192:168:12::2* is the IP address of the ingress interface *r2-eth0* in router r2. The segment routing header also contains the "Segments Left" field used by routers to determine how many segments need to be processed before removing the segment routing header from the packet.

Step 9. Click on the red button located on the upper left-hand side to stop packet capturing and close Wireshark.


Capturing from r2-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.  Time         Source              Destination         Protocol Length Info
   1 0.000000000  2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
   2 0.000070191  2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
   3 1.019213327  2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
   4 1.019273941  2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
   5 2.043211027  2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
   6 2.043269976  2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
   7 3.067211687  2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
   8 3.067267497  2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
   9 4.091199161  2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
```

```
Frame 1: 198 bytes on wire (1584 bits), 198 bytes captured (1584 bits) on interface 0
Ethernet II, Src: 6e:aa:df:d7:74:05 (6e:aa:df:d7:74:05), Dst: 6a:c4:7d:18:18:e0 (6a:c4:7d:18:18:e0)
Internet Protocol Version 6, Src: 2001:192:168:12::1, Dst: 2001:192:168:12::2
  0110 .... = Version: 6
  .... 0000 0000 .... .... .... .... .... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)
  .... .... .... 1011 1110 1101 0011 0111 = Flow Label: 0xbed37
  Payload Length: 144
  Next Header: Routing Header for IPv6 (43)
  Hop Limit: 63
  Source: 2001:192:168:12::1
  Destination: 2001:192:168:12::2
  Routing Header for IPv6 (Segment Routing)
    Next Header: IPv6 (41)
    Length: 4
    [Length: 40 bytes]
    Type: Segment Routing (4)
    Segments Left: 1
    First segment: 1
    Flags: 0x00
    Reserved: 0000
    Address[0]: 2001:192:168:24::2 [next segment]
    Address[1]: 2001:192:168:12::2
    [Segments in Traversal Order]
Internet Protocol Version 6, Src: 2001:192:168:1::10, Dst: 2001:192:168:4::10
```

Figure 43. Stopping packet capture.

Step 10. Verify the packet capturing for the interface *r3-eth1* on Wireshark. You will observe that no packets were captured.

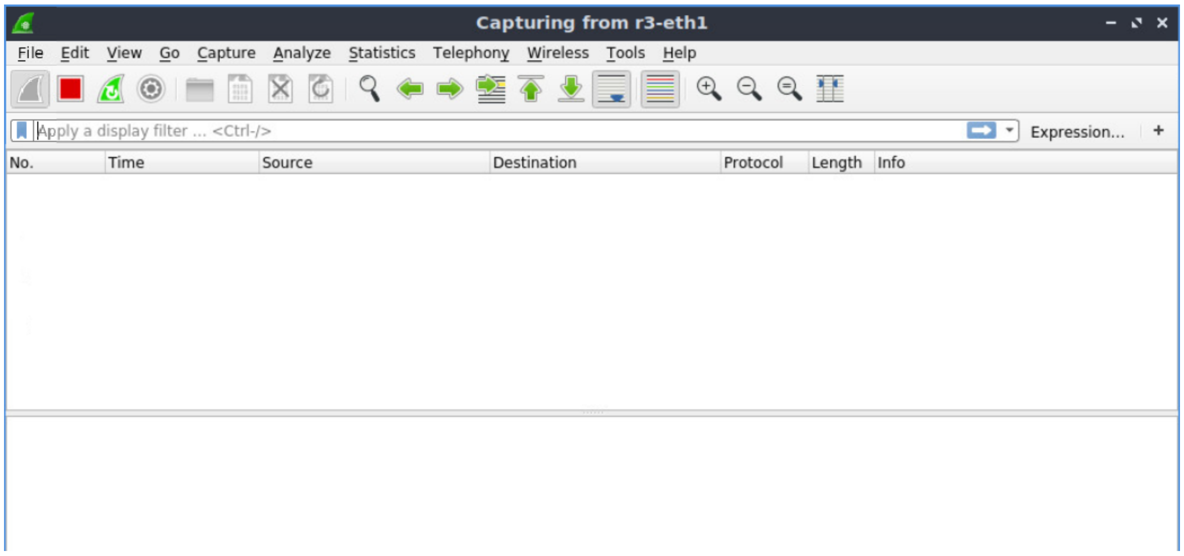

Capturing from r3-eth1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...
No. Time Source Destination Protocol Length Info

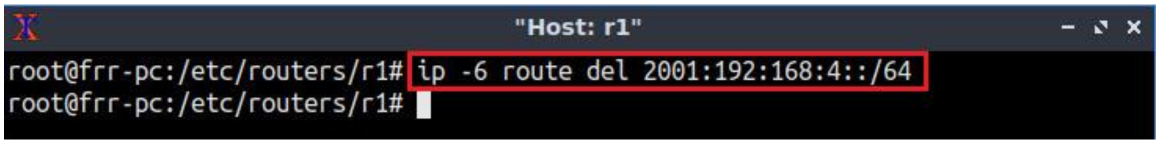
Figure 44. Packets are not passing across router r3.

Step 11. Click on the red button located on the upper left-hand side to stop packet capturing and close Wireshark.

4.3 Change the SR path

Step 1. Navigate into router r1 interface. Delete the previous route configuration by typing the following command:

```
ip -6 route del 2001:192:168:4::/64
```

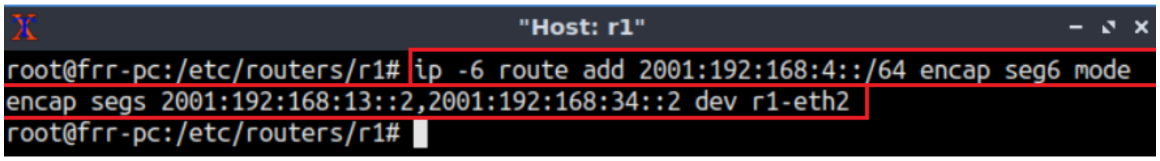


```
"Host: r1"
root@frr-pc:/etc/routers/r1# ip -6 route del 2001:192:168:4::/64
root@frr-pc:/etc/routers/r1#
```

Figure 45. Deleting previous configuration.

Step 2. In router r1 terminal, specify the new route that passes across router r3 by typing the following command:

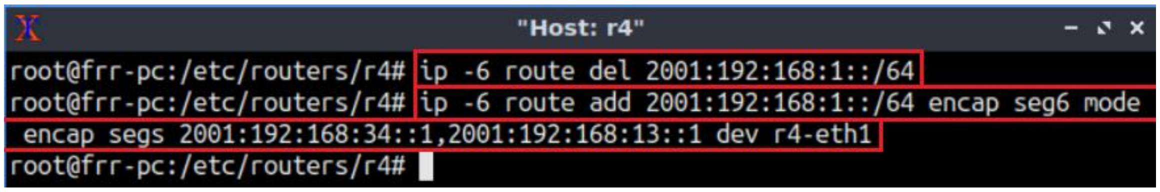
```
ip -6 route add 2001:192:168:4::/64 encap seg6 mode encap segs
2001:192:168:13::2,2001:192:168:34::2 dev r1-eth2
```



```
"Host: r1"
root@frr-pc:/etc/routers/r1# ip -6 route add 2001:192:168:4::/64 encap seg6 mode
encap segs 2001:192:168:13::2,2001:192:168:34::2 dev r1-eth2
root@frr-pc:/etc/routers/r1#
```

Figure 46. Specifying route configuration in router r1.

Step 3. Proceed similarly in router r4 Terminal. All the steps are summarized in the following figure:



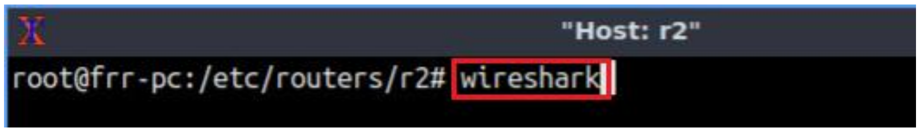
```
"Host: r4"
root@frr-pc:/etc/routers/r4# ip -6 route del 2001:192:168:1::/64
root@frr-pc:/etc/routers/r4# ip -6 route add 2001:192:168:1::/64 encap seg6 mode
 encap segs 2001:192:168:34::1,2001:192:168:13::1 dev r4-eth1
root@frr-pc:/etc/routers/r4#
```

Figure 47. Configuration summary of router r4.

4.4 Verify the configuration after changing the SR path

Step 1. In router r2's Terminal, start Wireshark dissector by issuing the following command. A new window will emerge.

```
wireshark
```



```
"Host: r2"
root@frr-pc:/etc/routers/r2# wireshark
```

Figure 48. Starting packet capturing with Wireshark.

Step 2. Click on the icon located on the upper left-hand side to start capturing packets on the interface *r2-eth1*.

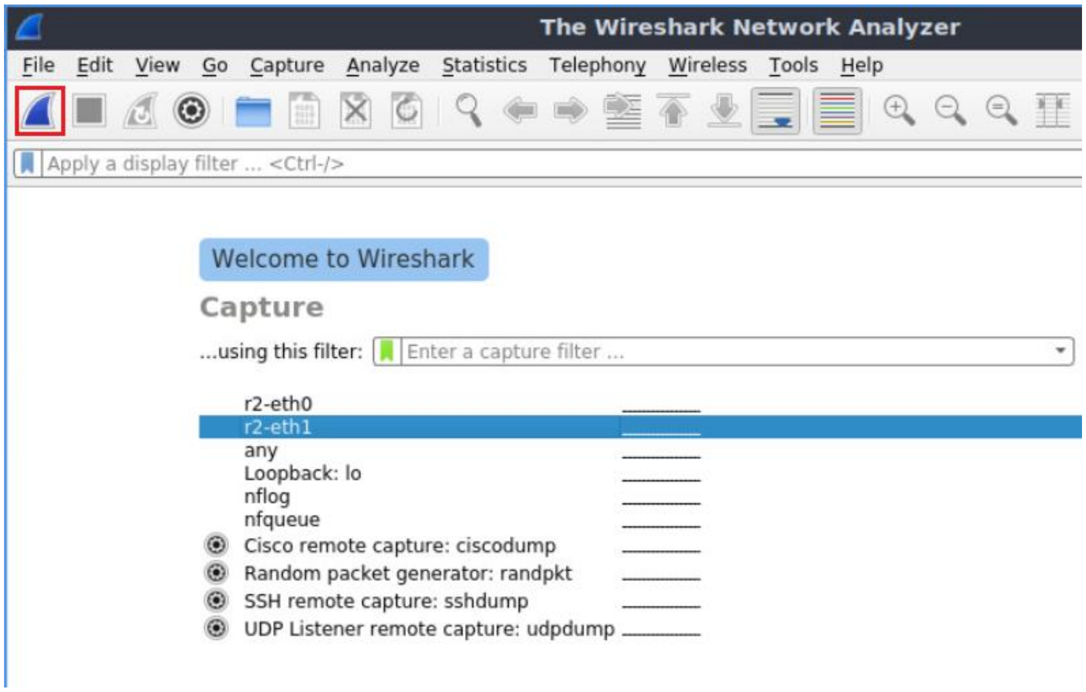
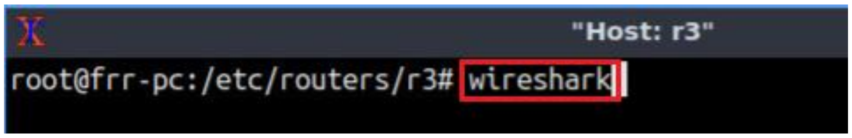

The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Capture
...using this filter:
Enter a capture filter ...
r2-eth0
r2-eth1
any
Loopback: lo
nflog
nfqueue
Cisco remote capture: ciscodump
Random packet generator: randpkt
SSH remote capture: sshdump
UDP Listener remote capture: udpdump

Figure 49. Starting packet capture.

Step 3. In router r3's Terminal, start Wireshark dissector by issuing the following command. A new window will emerge.

```
wireshark
```



```
"Host: r3"
root@frr-pc:/etc/routers/r3# wireshark
```

Figure 50. Starting packet capturing with Wireshark.

Step 4. Click on the icon located on the upper left-hand side to start capturing packets on the interface *r3-eth0*.

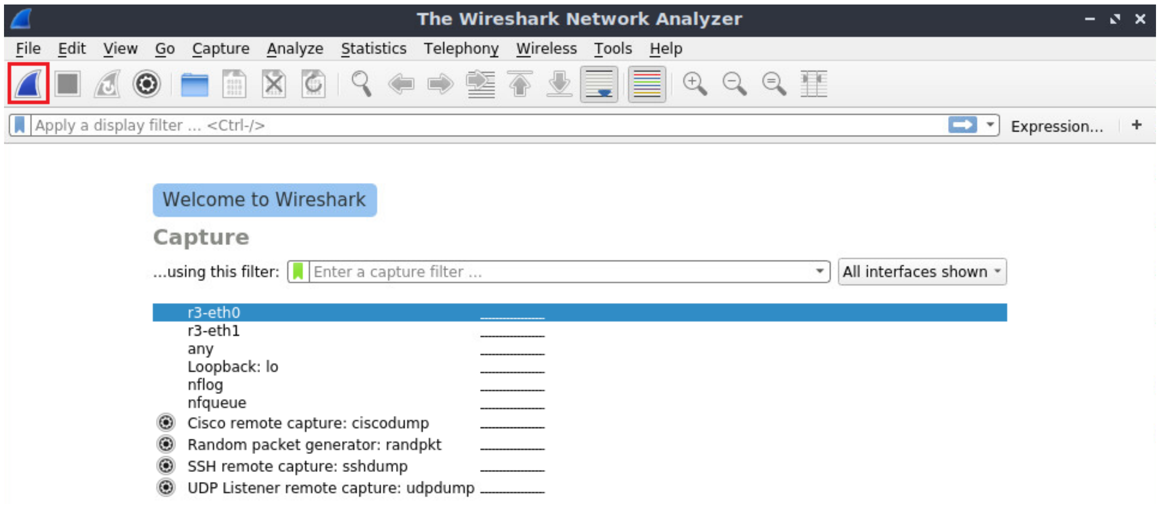
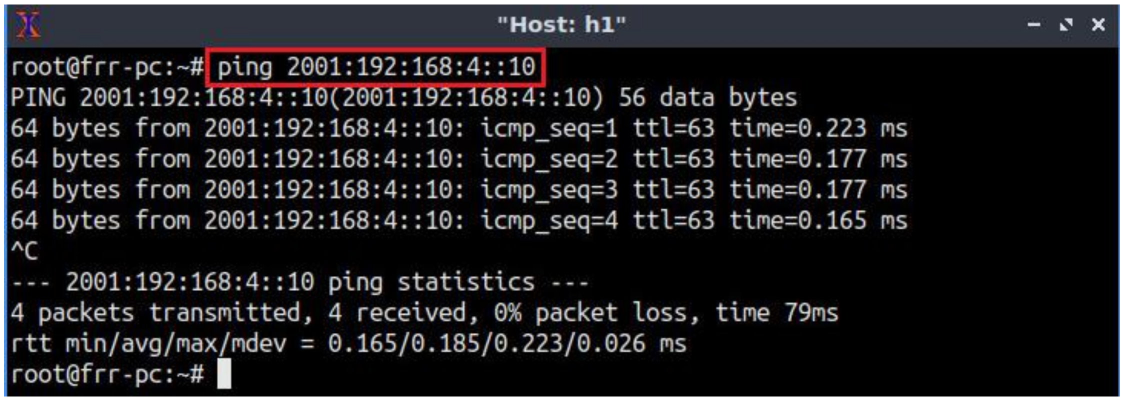

The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Expression…
Welcome to Wireshark
Capture
…using this filter:
Enter a capture filter …
All interfaces shown
r3-eth0
r3-eth1
any
Loopback: lo
nflog
nfqueue
Cisco remote capture: ciscodump
Random packet generator: randpkt
SSH remote capture: sshdump
UDP Listener remote capture: udpdump

Figure 51. Starting packet capture.

Step 5. Navigate to the host h1 terminal and perform a connectivity test by typing the following command, then press `ctrl+c` to stop the test.

```
ping 2001:192:168:4::10
```



"Host: h1"

```
root@frr-pc:~# ping 2001:192:168:4::10
PING 2001:192:168:4::10(2001:192:168:4::10) 56 data bytes
64 bytes from 2001:192:168:4::10: icmp_seq=1 ttl=63 time=0.223 ms
64 bytes from 2001:192:168:4::10: icmp_seq=2 ttl=63 time=0.177 ms
64 bytes from 2001:192:168:4::10: icmp_seq=3 ttl=63 time=0.177 ms
64 bytes from 2001:192:168:4::10: icmp_seq=4 ttl=63 time=0.165 ms
^C
--- 2001:192:168:4::10 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 79ms
rtt min/avg/max/mdev = 0.165/0.185/0.223/0.026 ms
root@frr-pc:~#
```

Figure 52. Performing a connectivity test between host h1 and host h2.

Step 6. Verify the packet capturing for the interface *r2-eth1* on Wireshark. You will observe that no packets were captured.

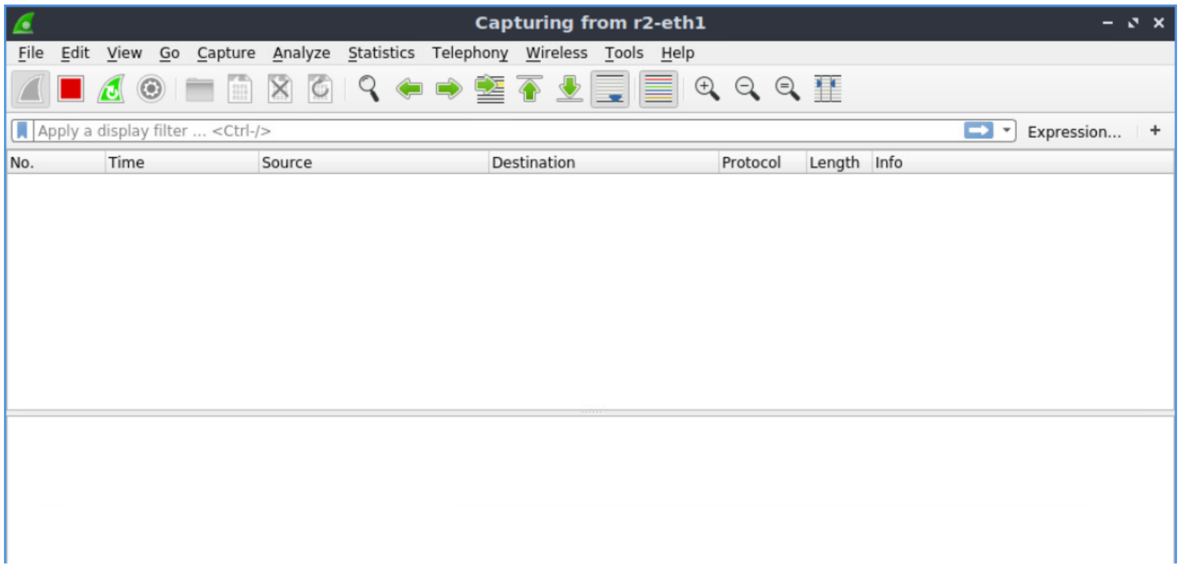

Capturing from r2-eth1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...
No. Time Source Destination Protocol Length Info

Figure 53. Packets are not passing across router r2.

Step 7. Click on the red button located on the upper left-hand side to stop packet capturing and close Wireshark.

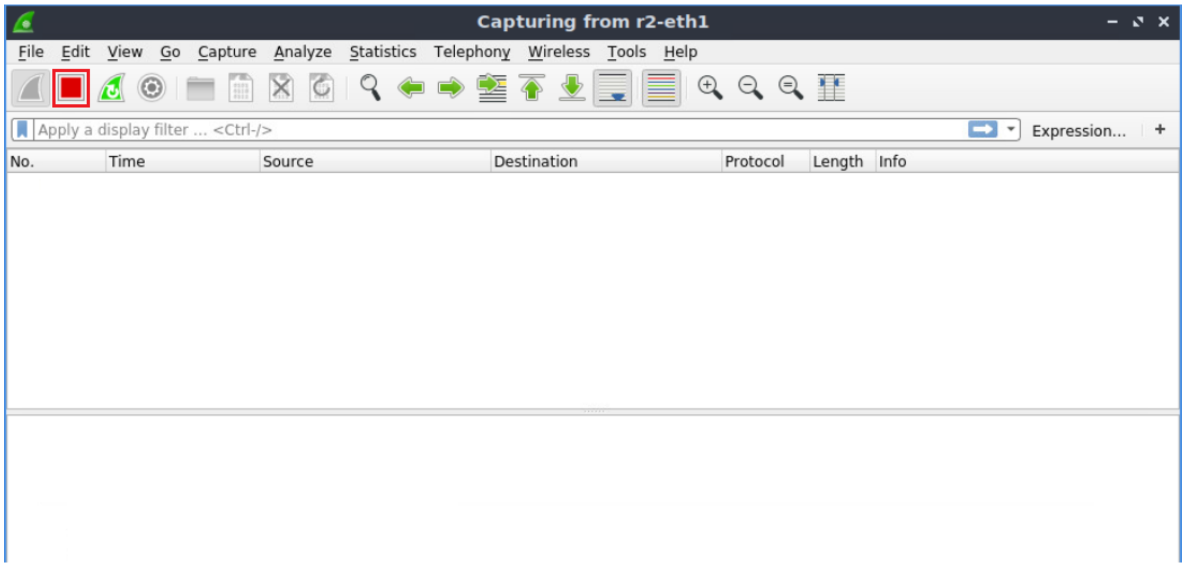
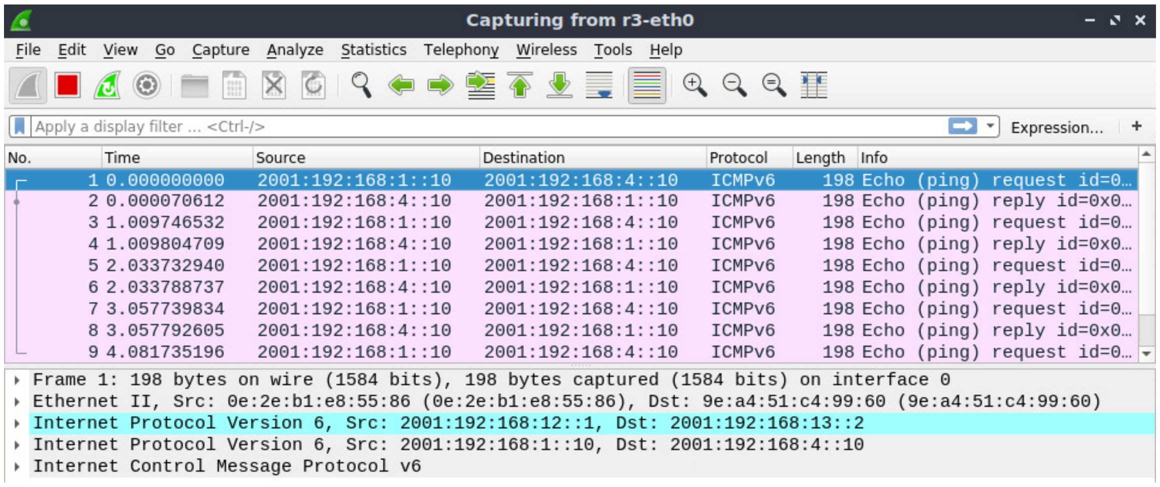

Capturing from r2-eth1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...
No. Time Source Destination Protocol Length Info

Figure 54. Stopping packet capture.

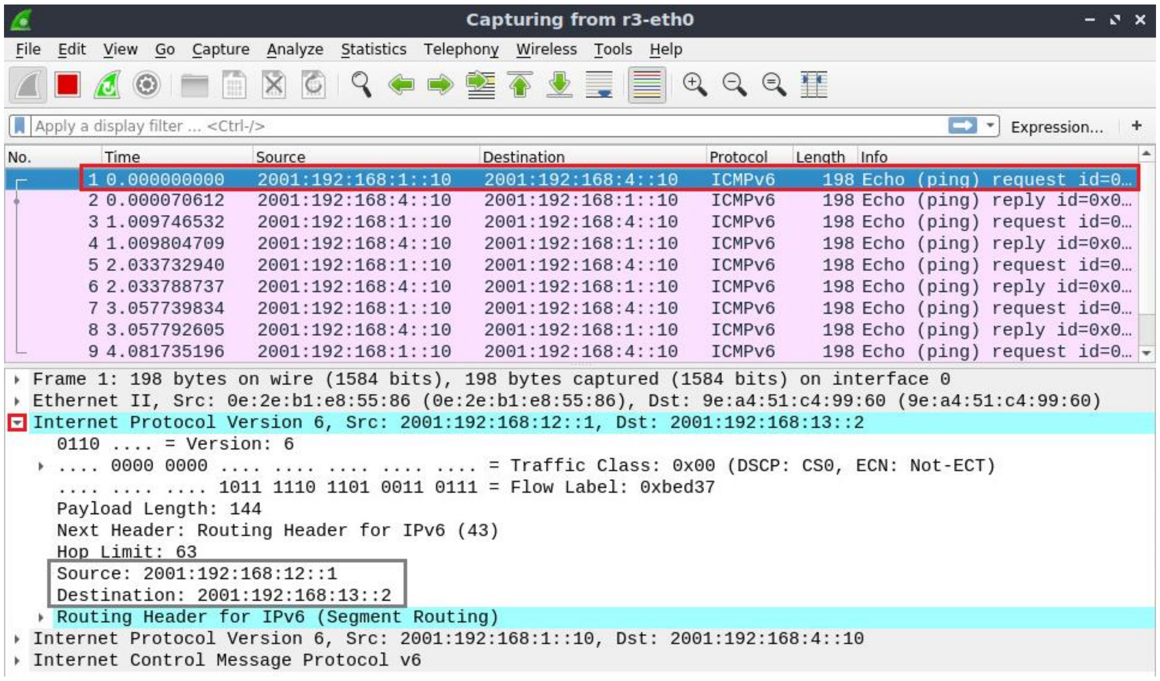
Step 8. Verify the packet capturing for the interface *r3-eth0* on Wireshark. You will observe that the packets are passing across router r3.


Capturing from r3-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No. Time          Source              Destination         Protocol Length Info
  1 0.000000000   2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
  2 0.000070612   2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
  3 1.009746532   2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
  4 1.009804709   2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
  5 2.033732940   2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
  6 2.033788737   2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
  7 3.057739834   2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
  8 3.057792605   2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
  9 4.081735196   2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…

Frame 1: 198 bytes on wire (1584 bits), 198 bytes captured (1584 bits) on interface 0
Ethernet II, Src: 0e:2e:b1:e8:55:86 (0e:2e:b1:e8:55:86), Dst: 9e:a4:51:c4:99:60 (9e:a4:51:c4:99:60)
Internet Protocol Version 6, Src: 2001:192:168:12::1, Dst: 2001:192:168:13::2
Internet Protocol Version 6, Src: 2001:192:168:1::10, Dst: 2001:192:168:4::10
Internet Control Message Protocol v6
```

Figure 55. Packets are passing across router r3.

Step 9. Open the first tab corresponding to *Internet Protocol Version 6* as depicted in the figure below.

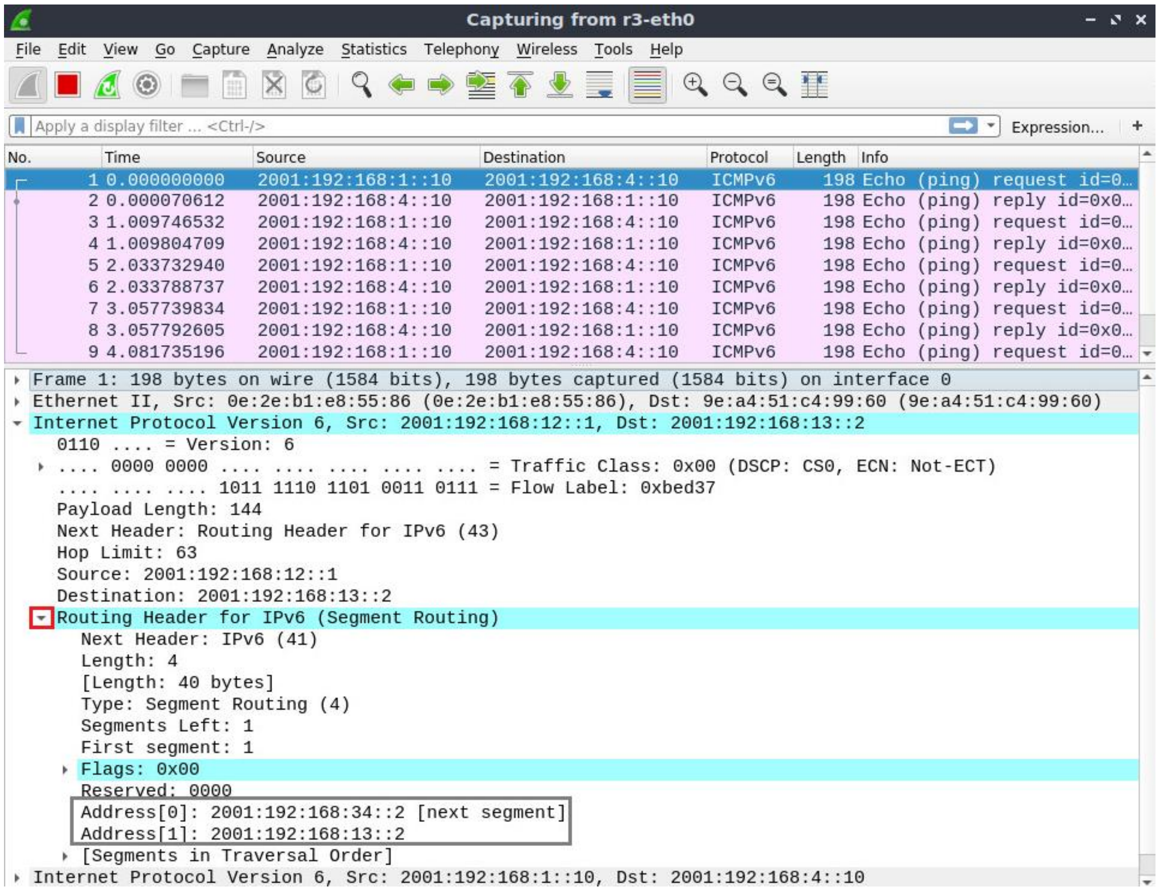

Capturing from r3-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No. Time          Source              Destination         Protocol Length Info
  1 0.000000000   2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
  2 0.000070612   2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
  3 1.009746532   2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
  4 1.009804709   2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
  5 2.033732940   2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
  6 2.033788737   2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
  7 3.057739834   2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…
  8 3.057792605   2001:192:168:4::10  2001:192:168:1::10  ICMPv6   198 Echo (ping) reply id=0x0…
  9 4.081735196   2001:192:168:1::10  2001:192:168:4::10  ICMPv6   198 Echo (ping) request id=0…

Frame 1: 198 bytes on wire (1584 bits), 198 bytes captured (1584 bits) on interface 0
Ethernet II, Src: 0e:2e:b1:e8:55:86 (0e:2e:b1:e8:55:86), Dst: 9e:a4:51:c4:99:60 (9e:a4:51:c4:99:60)
Internet Protocol Version 6, Src: 2001:192:168:12::1, Dst: 2001:192:168:13::2
  0110 .... = Version: 6
  .... 0000 0000 .... .... .... .... .... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)
  .... .... .... 1011 1110 1101 0011 0111 = Flow Label: 0xbed37
  Payload Length: 144
  Next Header: Routing Header for IPv6 (43)
  Hop Limit: 63
  Source: 2001:192:168:12::1
  Destination: 2001:192:168:13::2
  Routing Header for IPv6 (Segment Routing)
Internet Protocol Version 6, Src: 2001:192:168:1::10, Dst: 2001:192:168:4::10
Internet Control Message Protocol v6
```

Figure 56. Displaying IPv6 header information.

The user will visualize the source IP address 2001:192:168:12::1 in the segment routing header corresponding to the egress interface *r1-eth1* of router r1. The destination IP address 2001:192:168:13::2 in the segment routing header corresponds to the ingress interface *r3-eth0* of router r3. Note how the IPv6 addresses 2001:192:168:1::10 and 2001:192:168:4::10 that correspond to hosts h1 and h2 in the original packet headers remain unchanged.

Step 10. Click on the tab corresponding to *Routing Header for IPv6 (Segment Routing)* as shown in the figure below.


Capturing from r3-eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.  Time         Source              Destination         Protocol Length Info
   1 0.000000000  2001:192:168:1::10  2001:192:168:4::10  ICMPv6    198 Echo (ping) request id=0…
   2 0.000070612  2001:192:168:4::10  2001:192:168:1::10  ICMPv6    198 Echo (ping) reply id=0x0…
   3 1.009746532  2001:192:168:1::10  2001:192:168:4::10  ICMPv6    198 Echo (ping) request id=0…
   4 1.009804709  2001:192:168:4::10  2001:192:168:1::10  ICMPv6    198 Echo (ping) reply id=0x0…
   5 2.033732940  2001:192:168:1::10  2001:192:168:4::10  ICMPv6    198 Echo (ping) request id=0…
   6 2.033788737  2001:192:168:4::10  2001:192:168:1::10  ICMPv6    198 Echo (ping) reply id=0x0…
   7 3.057739834  2001:192:168:1::10  2001:192:168:4::10  ICMPv6    198 Echo (ping) request id=0…
   8 3.057792605  2001:192:168:4::10  2001:192:168:1::10  ICMPv6    198 Echo (ping) reply id=0x0…
   9 4.081735196  2001:192:168:1::10  2001:192:168:4::10  ICMPv6    198 Echo (ping) request id=0…

Frame 1: 198 bytes on wire (1584 bits), 198 bytes captured (1584 bits) on interface 0
Ethernet II, Src: 0e:2e:b1:e8:55:86 (0e:2e:b1:e8:55:86), Dst: 9e:a4:51:c4:99:60 (9e:a4:51:c4:99:60)
Internet Protocol Version 6, Src: 2001:192:168:12::1, Dst: 2001:192:168:13::2
  0110 .... = Version: 6
  .... 0000 0000 .... .... .... .... .... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)
  .... .... .... 1011 1110 1101 0011 0111 = Flow Label: 0xbed37
  Payload Length: 144
  Next Header: Routing Header for IPv6 (43)
  Hop Limit: 63
  Source: 2001:192:168:12::1
  Destination: 2001:192:168:13::2
  Routing Header for IPv6 (Segment Routing)
    Next Header: IPv6 (41)
    Length: 4
    [Length: 40 bytes]
    Type: Segment Routing (4)
    Segments Left: 1
    First segment: 1
    Flags: 0x00
    Reserved: 0000
    Address[0]: 2001:192:168:34::2 [next segment]
    Address[1]: 2001:192:168:13::2
    [Segments in Traversal Order]
Internet Protocol Version 6, Src: 2001:192:168:1::10, Dst: 2001:192:168:4::10
```

Figure 57. Displaying segment routing hops IP addresses.

The information displayed in the figure above shows the segment routing path. *Address[0]: 2001:192:168:34::2* is the next segment and corresponds to the next-hop IP address (router r4). *Address[1]: 2001:192:168:13::2* is the IP address of the ingress interface *r3-eth0* in router r3. The segment routing header also contains the "Segments Left" field used by routers to determine how many segments need to be processed before removing the segment routing header from the packet.

Step 11. Click on the red button located on the upper left-hand side to stop packet capturing and close Wireshark.

This concludes Lab 9. Stop the emulation and then exit out of MiniEdit.

References

1. Juniper Networks, "*Segment Routing (SR) – What You Need To Know*," [Online]. Available: https://blogs.juniper.net/en-us/industry-solutions-and-trends/Segment-Routing-SR-What-You-Need-To-Know/
2. C. Filsfils, K. Michielsen, P. Camarillo, F. Clad. "*SRv6*", [Online]. Available: https://www.segment-routing.net/tutorials/2017-12-05-srv6-introduction/
3. F. Khella, Cisco. "*Cisco Introduces Segment Routing v6 on Nexus 9000 GX Series Platforms*", [Online]. Available: https://blogs.cisco.com/datacenter/cisco-introduces-segment-routing-v6-on-nexus-9000-gx-series-platforms

4. Internet Protocol, Version 6 (IPv6) Specification, RFC8200, [Online]. Available: https://datatracker.ietf.org/doc/html/rfc8200
5. SRv6 Linux Kernel Implementation, [Online]. Available: https://segment-routing.org/
6. sysctl(8) — Linux manual page, [Online]. Available: https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt

MPLS AND ADVANCED BGP TOPICS

Exercise 1: Configuring MPLS Layer 3 VPN using MP-BGP

Document Version: 08-31-2021

Award 1829698
“CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers”

Contents

1 Exercise topology

This exercise is about MPLS Layer 3 VPN which provides network virtualization solutions for each customer connected to the service provider. The entire communication is forwarded using layer 3 Virtual Routing and Forwarding (VRF) techniques. Layer 3 VPN requires border gateway protocol (BGP) to send and receive VPN-related data. This exercise aims to test the learner's knowledge, in particular labs 5, 6, and 7.

The topology shown in Figure 1 is composed of eight routers, four switches, and four end-hosts. The goal of this exercise is to configure MPLS Layer3 VPN for two different organizations (Org1 and Org2) while hosts are using overlapping IPv4 prefixes. The topology below is already built.

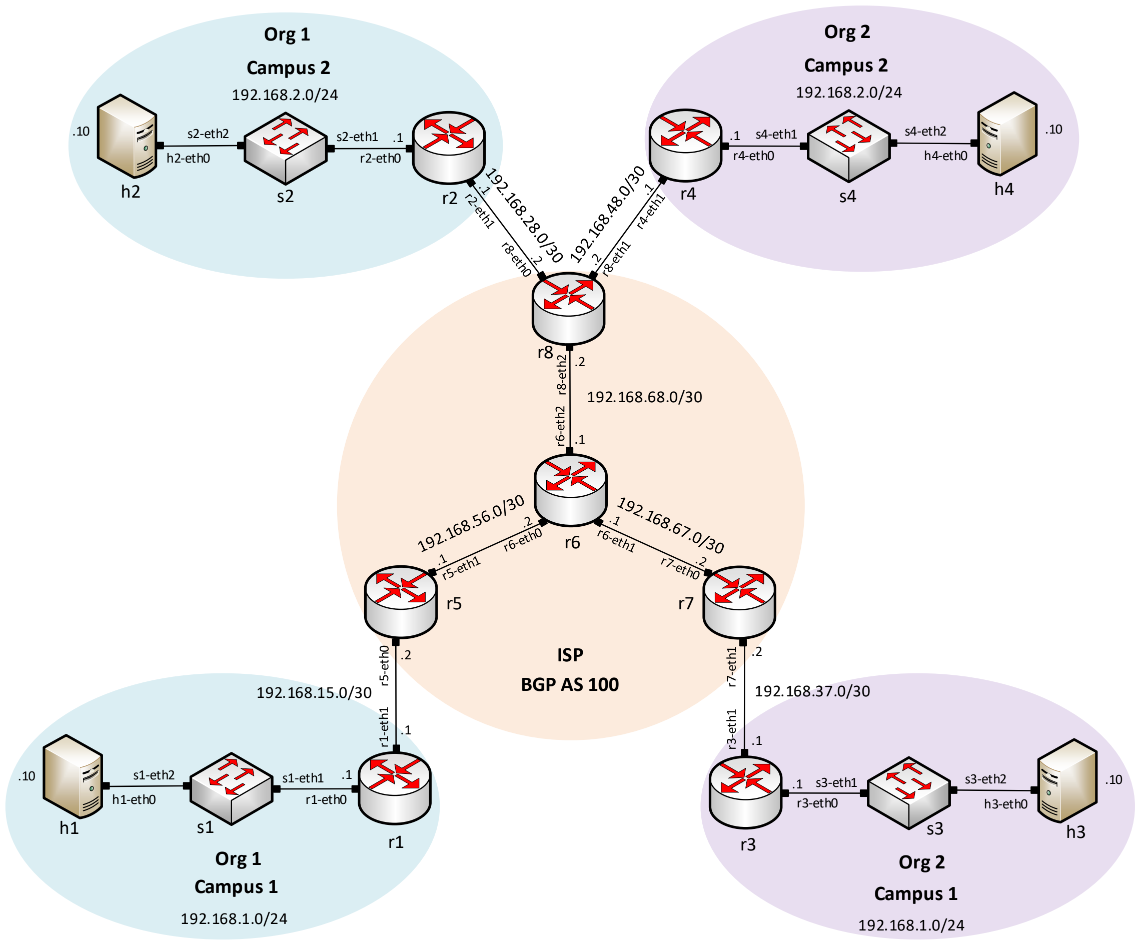


Figure 1. Exercise topology.

1.1 Topology settings

The devices are already configured according to Table 1.

Table 1. Topology information.

Device	Interface	IPV4 Address	Subnet	Default gateway

r1	r1-eth0	192.168.1.1	/24	N/A
	r1-eth1	192.168.15.1	/30	N/A
r2	r2-eth0	192.168.2.1	/24	N/A
	r2-eth1	192.168.28.1	/30	N/A
r3	r3-eth0	192.168.1.1	/24	N/A
	r3-eth1	192.168.37.1	/30	N/A
r4	r4-eth0	192.168.2.1	/24	N/A
	r4-eth1	192.168.48.1	/30	N/A
r5	r5-eth0	192.168.15.2	/30	N/A
	r5-eth1	192.168.56.1	/30	N/A
	lo	5.5.5.5	/32	N/A
r6	r6-eth0	192.168.56.2	/30	N/A
	r6-eth1	192.168.67.1	/30	N/A
	r6-eth2	192.168.68.1	/30	N/A
	lo	6.6.6.6	/32	N/A
r7	r7-eth0	192.168.67.2	/30	N/A
	r7-eth1	192.168.37.2	/30	N/A
	lo	7.7.7.7	/32	N/A
r8	r8-eth0	192.168.28.2	/30	N/A
	r8-eth1	192.168.48.2	/30	N/A
	r8-eth2	192.168.68.2	/30	N/A
	lo	8.8.8.8	/32	N/A
h1	h1-eth0	192.168.1.10	/24	192.168.1.1
h2	h2-eth0	192.168.2.10	/24	192.168.2.1
h3	h3-eth0	192.168.1.10	/24	192.168.1.1
h4	h4-eth0	192.168.2.10	/24	192.168.2.1

1.2 Credentials

The information in Table 2 provides the credentials to access the Client's virtual machine.

Table 2**.** Credentials to access the Client's virtual machine.

Device	Account	Password
Client	admin	password

2 Deliverables

Follow the steps below to complete the exercise.

a) Start MiniEdit by clicking on MiniEdit's shortcut. Load the topology *topology.mn* located at *~/MPLS_advanced_BGP/Exercise1* as shown in the figure below. Verify in Mininet terminal that the links conform to the topology figure and settings table above.

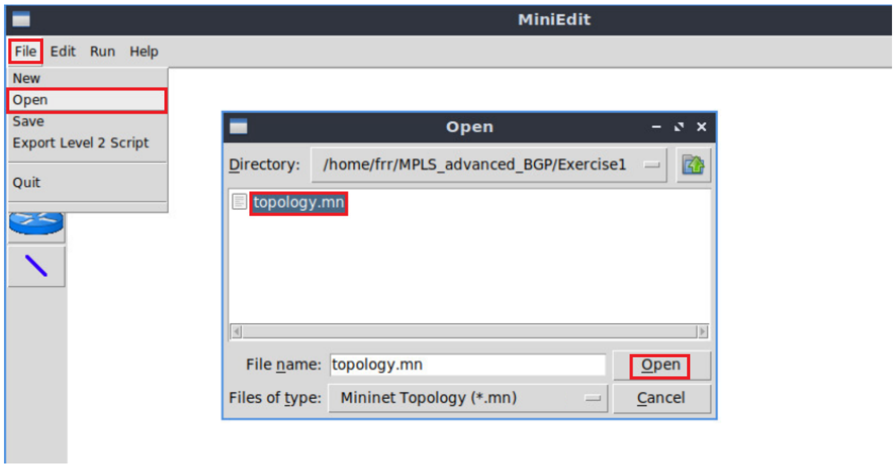


Figure 2. Loading the topology file in Mininet.

b) *~/MPLS_advanced_BGP/Exercise1* folder contains a configuration file *config.zip* and a script *config_loader.sh* responsible for loading the IP addresses, static route, and OSPF configuration. Execute the script using the following command:

```
cd MPLS_advanced_BGP/Exercise1
```

```
./config_loader.sh config.zip
```

c) Run the emulation in Mininet.

d) In the Mininet terminal, launch the command that displays the interface names and connections of the current topology. Verify that links conform to the topology in Figure 1.

e) *~/MPLS_advanced_BGP/Exercise1* folder also contains a configuration file *config_routers.sh* to enable MPLS forwarding on routers r5, r6, r7, and r8. Type the following command to enable MPLS forwarding:

```
./config_routers.sh
```

f) Enable zebra daemon on all routers. Enable static daemon on all routers except router r6. Enable OSPF daemon on ISP routers (r5, r6, r7, and r8).

g) Using the FRR shell (vtysh), inspect the routing tables of the ISP routers and verify that they have routes to each router's loopback address learned via OSPF. Test the connectivity between the ISP routers using their loopback addresses.

h) Enable LDP daemon and configure LDP on ISP routers. Verify LDP configuration on ISP routers.

i) Create VRF org1 in routers r5 and r8. Create VRF org2 in routers r7 and r8. Assign interfaces to the corresponding VRFs. Configure static routes for VRFs org1 and org2 on PE routers. Verify the routing table for each VRF on PE routers.

j) Enable BGP daemon and configure BGP on PE routers. Establish BGP neighborship between routers r5 and r8 for VRF org1. Establish BGP neighborship between routers r7 and r8 for VRF org2. Activate BGP neighbors for IPv4 VPN.

k) Configure BGP on PE routers in order to exchange VPN routes through BGP. Redistribute static routes, assign RD and RT values. Enable VPN processing on the routers.

l) Inspect BGP neighborship, BGP table, and routing table for each VRF to verify MPLS layer 3 VPN configuration. Perform connectivity tests between two campuses of the same organization using the `traceroute` command.

MPLS AND ADVANCED BGP TOPICS

Exercise 2: Configuring Segment Routing over IPv6 (SRv6)

Document Version: 08-31-2021

Award 1829698
"CyberTraining CIP: Cyberinfrastructure Expertise on High-throughput Networks for Big Science Data Transfers"

Contents

1 Exercise topology

Segment routing uses the source packet routing technique. In source packet routing, the source or ingress router specifies the path a packet will take through the network. Instead of storing path information in transit routers, segment routing encodes the path into packet headers. Segment routing can be implemented over MPLS (SR-MPLS) or IPv6 (SRv6). This exercise aims to test the learner's knowledge on SRv6 (lab 9).

The topology shown in Figure 1 is composed of five routers and two end-hosts.

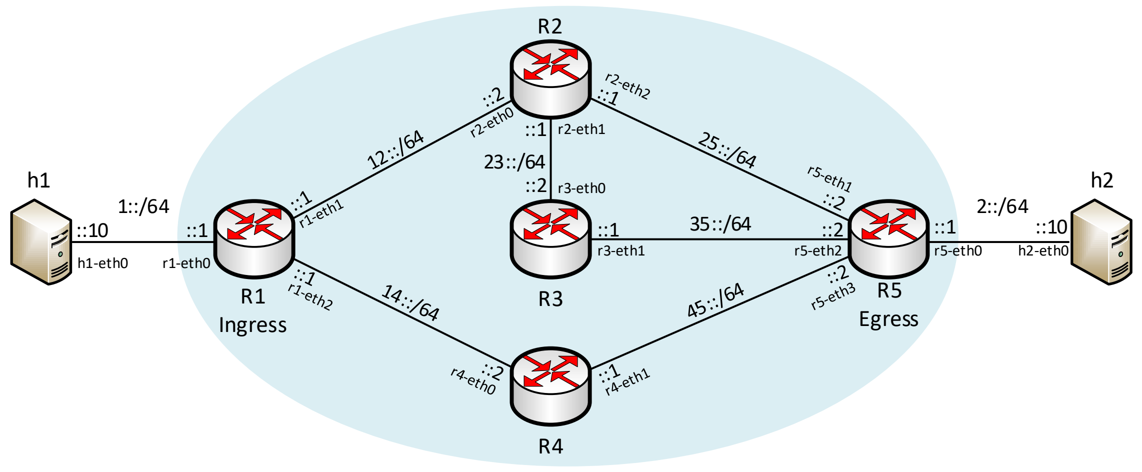


Figure 1. Exercise topology.

1.1 Topology settings

Routers and hosts are configured according to the IP addresses shown in Table 1.

Table 1. Topology information.

Device	Interface	IPv6 Address	Default gateway
r1	r1-eth0	1::1/64	N/A
	r1-eth1	12::1/64	N/A
	r1-eth2	14::1/64	N/A
r2	r2-eth0	12::2/64	N/A
	r2-eth1	23::1/64	N/A
	r2-eth2	25::1/64	N/A
r3	r3-eth0	23::2/64	N/A
	r3-eth1	35::1/64	N/A
r4	r4-eth0	14::2/64	N/A

	r4-eth1	45::1/64	N/A
r5	r5-eth0	2::1/64	N/A
	r5-eth1	25::2/64	N/A
	r5-eth2	35::2/64	N/A
	r5-eth3	45::2/64	N/A
h1	h1-eth0	1::10/64	1::1
h2	h2-eth0	2::10/64	2::1

Consider using the manual of lab 9 of the MPLS and Advance BGP Topics lab series as a reference to complete this exercise.

1.1 Credentials

The information in Table 2 provides the credentials to access the Client's virtual machine.

Table 2. Credentials to access the Client's virtual machine.

Device	Account	Password
Client	admin	password

2 Deliverables

Follow the steps below to complete the exercise.

a) Start MiniEdit by clicking on MiniEdit's shortcut. Load the topology *topology.mn* located at *~/MPLS_advanced_BGP/Exercise2* as shown in the figure below. Verify in Mininet terminal that the links conform to the topology figure and settings table above.

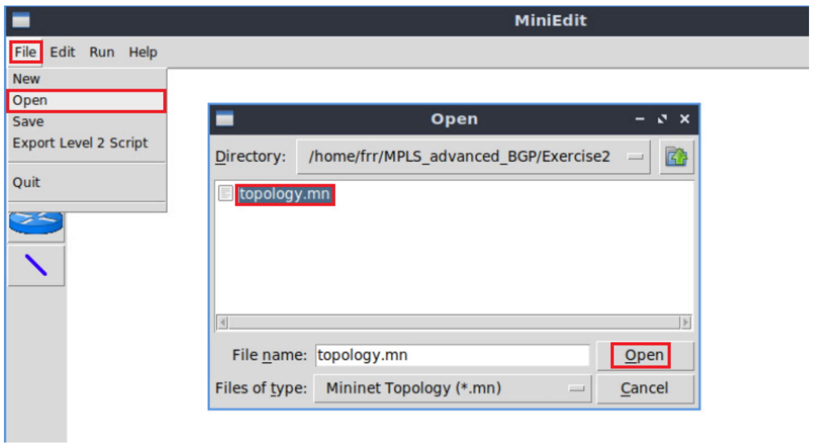


Figure 1. Loading the topology file in Mininet.

b) Issue the following commands to configure the IP addresses on the interfaces of the routers r1, r2, r3, r4, and r5.

```
cd MPLS_advanced_BGP/Exercise2
```

```
./config_loader.sh conf.zip
```

c) Start the emulation and enable the zebra daemon on all routers.

d) Configure the hosts with IPv6 addresses and default gateways.

e) Enable SRv6 on all routers, then configure an SRv6 path on routers r1 and r5 so that traffic between hosts h1 and host h2 traverses the routers r1, r4, and r5.

f) Verify the connectivity between the hosts h1 and h2.

g) Open Wireshark and capture packets leaving router r1. Inspect the segments left field in the SRv6 header.

h) Modify the previous path so that traffic traverses r1-r2-r3-r5.

i) Verify the connectivity between the hosts h1 and h2.

j) Open Wireshark and capture packets leaving router r1. Compare the segments left field in the SRv6 header to the number reported with the previous path.

k) Disable SRv6 on r2. Do not disable IPv6 forwarding.

l) Configure an SRv6 path on routers r1 and r5 so that traffic between hosts h1 and host h2 traverses the routers r1, r2, and r5. Recall that r2 does not process SRv6 headers.

m) At this point, there will be no connectivity between the hosts. Inspect the routing tables of routers r1 and r5 and create the corresponding routes.

The command to configure a static route has the following syntax:
`ip -6 route add <dest_network> via <next_hop>`

n) Verify the connectivity between the hosts h1 and h2.

o) Open Wireshark and capture packets entering and leaving router r2. Verify that the SRv6 header is not modified.