**Lab SDN: Legacy Networks: BGP Example as a Distributed System and Autonomous Forwarding Decisions**

### Contents

[1 Introduction 2](#_Toc176872684)

[1.1 Traditional switch architecture 2](#_Toc176872685)

[1.2 Legacy and SDN networks 2](#_Toc176872686)

[1.3 Introduction to FRR 2](#_Toc176872687)

[1.4 FRR architecture 3](#_Toc176872688)

[1.5 FRR and Mininet integration 4](#_Toc176872689)

[1.6 Introduction to BGP 4](#_Toc176872690)

[2 Lab topology 5](#_Toc176872691)

[2.1 Lab settings 6](#_Toc176872692)

[2.2 Loading the topology 7](#_Toc176872693)

[2.3 Loading the configuration file 8](#_Toc176872694)

[2.4 Running the emulation 9](#_Toc176872695)

[2.5 Verify the configuration 10](#_Toc176872696)

[2.6 Test connectivity between end-hosts 14](#_Toc176872697)

[3 Configure BGP routing protocol 14](#_Toc176872698)

[3.1 BGP neighbors on the routers 15](#_Toc176872699)

[3.2 Advertise local networks on the routers 19](#_Toc176872700)

[4 Verify connections 22](#_Toc176872701)

[References 24](#_Toc176872702)

**Overview**

This lab is an introduction to networks using Free Range Routing (FRR), which is a routing software suite that provides Transmission Control Protocol (TCP)/Internet Protocol (IP) based routing services with routing protocols support. In this lab, you will understand the main difference between legacy and Software Defined Networking (SDN) networks. Furthermore, you will explore FRR architecture, and load its basic configuration. Furthermore, this lab emulates a simple legacy network that runs Border Gateway Protocol (BGP) between two Autonomous Systems (ASes).

**Objectives**

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

1. Understand the difference between legacy and SDN networks.
2. Understand the architecture of FRR.
3. Navigate through FRR terminal.
4. Explain the concept of BGP.
5. Configure and verify BGP between two ASes.
6. Perform a connectivity test between end hosts.

**Lab settings**

The information in Table 1 provides the credentials of the machine containing Mininet emulator.

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.
  2. Section 2: Lab topology.
  3. Section 3: Configure BGP routing protocol.
  4. Section 4: Verify connections.

# 1 Introduction

## 1.1 Traditional switch architecture

## 1.2 Legacy and SDN networks

This lab solely focuses on understanding how legacy networks work. You will configure

BGP on legacy routers and inspect the inserted rules on each router’s forwarding table. In order to do the configuration in an emulated environment, FRR will be used, which is an open-source software that allows to configure the routers with a list of supported routing protocols.

## 1.3 Introduction to FRR

FRR suite1 is a package of Unix/Linux software that implements common network routing protocols, such as RIP2, OSPF3, BGP4 and IS-IS5. The package also includes a routing information management process, to act as an intermediary between the various routing protocols and the active routes installed with the kernel. A library provides support for configuration and an interactive command-line interface. The routing protocols supported by FRR, can be extended to enable experimentation, logging, or custom processing. In addition, libraries and kernel daemon provide a framework to facilitate the development of new routing protocol daemons. A wide range of functionalities can be attained by combining other software packages to allow the integration into a single device as well as enabling innovative solutions to networking problems.

## 1.4 FRR architecture

FRR takes a different approach compared to traditional routing software which, consists of a single process program that provides all the routing protocol functionalities. FRR is composed of a suite of daemons that work together to build a routing table. Each routing protocol is implemented in its own daemon. These daemons exchange information through another daemon called zebra, which is responsible for encompassing routing decisions and managing the data plane.

Figure 4 illustrates the FRR architecture. It consists of a set of processes communicating via Inter-process Communication (IPC) protocol. This protocol refers to the mechanism provided by an operating system (OS) to manage shared data between different processes. Network routing protocols such as BGP, OSPF and IS-IS are implemented in processes such as *bgpd*, *ripd*, *ospfd*, *ldpd, etc*. These processes are daemons that implement routing protocols e.g., the BGP daemon is implemented by the *bgpd* process*,* the RIP daemon is implemented by the *ripd* process and so on. Another daemon, called *zebra*, acts as an intermediary between the kernel’s forwarding plane and the routing protocol processes. Additionally, an interactive command-line tool called *vtysh* allows these processes to be monitored and configured. The *vtysh* command-line tool communicates with other processes via a simple string passing protocol, where the strings are essentially identical to the commands entered.

The *zebra* process is a fundamental part of FRR architecture. Its purpose is to maintain a backup of packet forwarding states, such as the network interfaces and the table of currently active routes. The currently active routes are also referred to as the Forwarding Information Base (FIB) 6. Usually, the kernel manages packet forwarding therefore, the kernel maintains these. The *zebra* process also collects routing information from the routing protocol processes and stores these, together with its shadow copy of the FIB, in its own Routing Information Base (RIB)6 whereas, static routes are also configured. The *zebra* process then is responsible for selecting the best route from all those available for a destination and updating the FIB7. Additionally, the information about the current best routes may be distributed to the protocol daemons. The *zebra* process maintains the routing daemons updated if any change occurs in the network interface state.

Vtysh

Zebra (RIB)

Kernel (FIB)

bgpd

ripd

ospfd

ldpd

...

Interactive

Command

-

line

Protocol

Daemons

Service

Daemons

Data

Plane

Packet

Forwarding

User Space

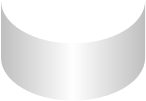
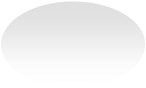
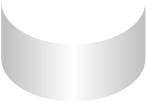
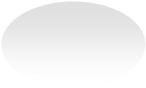
Figure 4. FRR architecture.

## 1.5 FRR and Mininet integration

## 1.6 Introduction to BGP

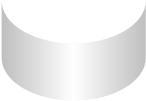
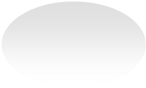
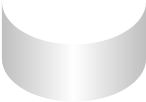
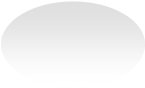
BGP is an exterior gateway protocol designed to exchange routing and reachability information among ASes on the Internet. BGP is relevant to network administrators of large organizations which connect to one or more Internet Service Providers (ISPs), as well as to ISPs who connect to other network providers. 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 administration8.

Two routers that establish a BGP connection are referred to as BGP peers or neighbors. BGP sessions run over 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 (IBGP)1. Figure 5 shows a network running the BGP protocol. Routers that exchange information within the same AS use Internal BGP (IBGP), while routers that exchange information between different ASes use EBGP.



**AS 200**

IBGP



**AS 100**

IBGP

EBGP

Figure 5. Routers that exchange information within the same AS use IBGP, while routers that exchange information between different ASes use EBGP.

# 2. Lab topology

Consider Figure 6. The topology consists of two networks, Network 1, and Network 2, each in an AS. Both networks have the following elements: a router to connect the networks together, a switch that defines a Local Area Network (LAN) and lastly, a host aimed to test end-to-end connectivity. The Autonomous System Numbers (ASNs) assigned to routers r1 and r2 are 100 and 200, respectively. Routers r1 and r2 exchange routing information via EBGP.

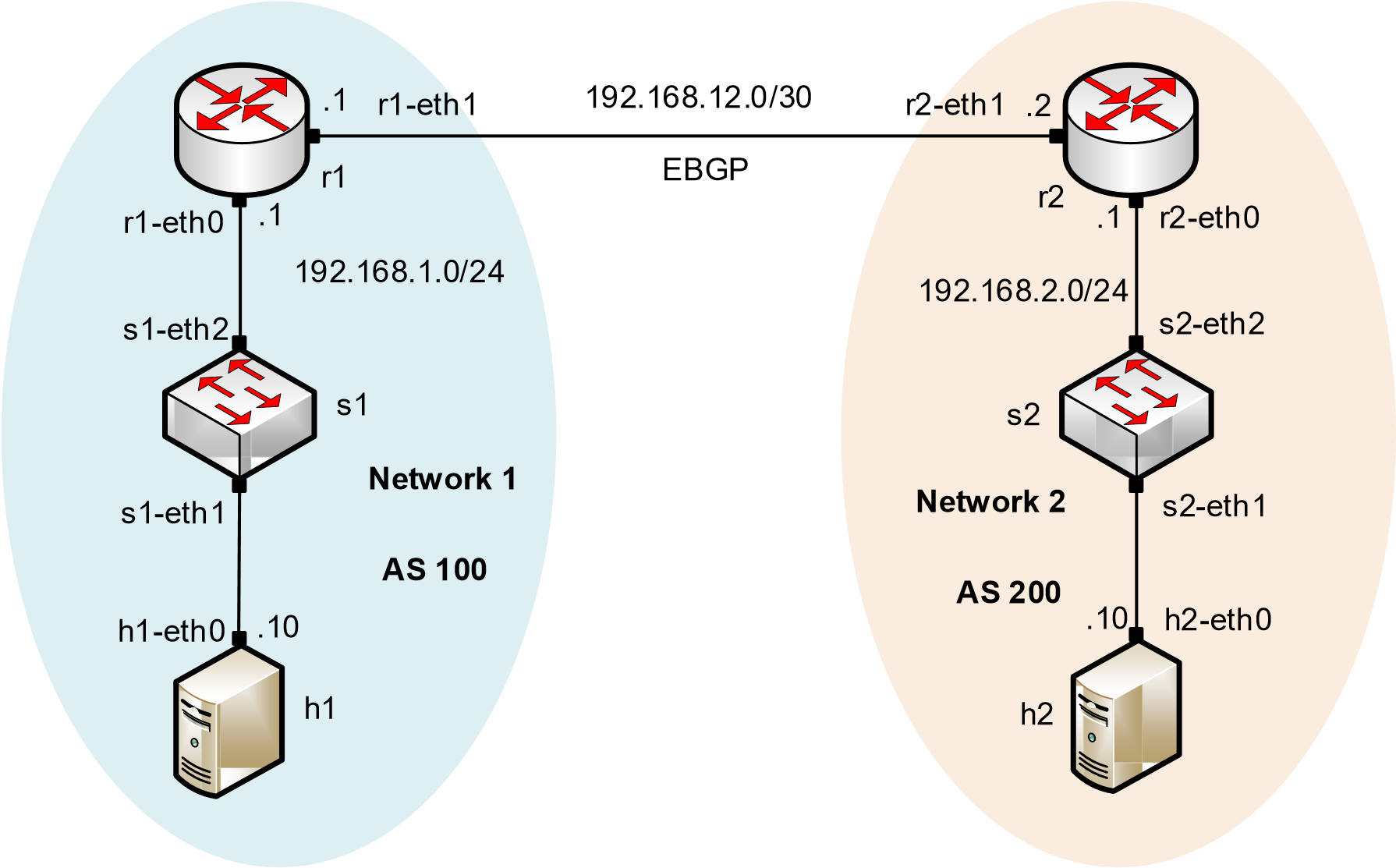


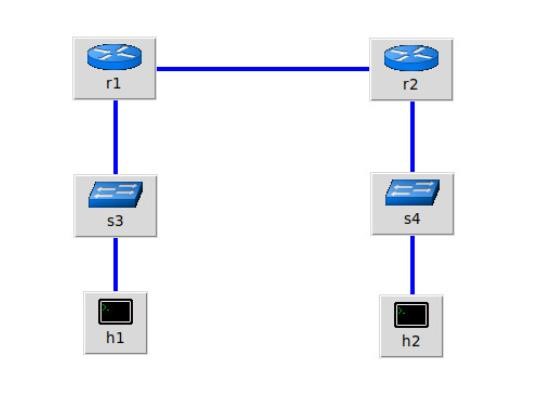
Figure 6. Lab topology.

## 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** | **IP Address** | **Subnet** | **Default gateway** |
| Router r1 | r1-eth0 | 192.168.1.1 | /24 | N/A |
| r1-eth1 | 192.168.12.1 | /30 | N/A |
| Router r2 | r2-eth0 | 192.168.2.1 | /24 | N/A |
| r2-eth1 | 192.168.12.2 | /30 | N/A |
| Host h1 | h1-eth0 | 192.168.1.10 | /24 | 192.168.1.1 |
| Host h2 | h2-eth0 | 192.168.2.10 | /24 | 192.168.2.1 |



## 2.2 Loading the topology

In this section, you will open MiniEdit10 and load the lab topology. MiniEdit provides a Graphical User Interface (GUI) that facilitates the creation and emulation of network topologies in Mininet. This tool has additional capabilities such as: configuring network elements (IP addresses, default gateway), save the topology and export a layer 2 model.

|  |
| --- |
|  |

**Step 1.** Start MiniEdit

**Step 2.** On MiniEdit’s menu bar, click on *File* then *open* to load the lab’s topology. Open the *Lab2.mn* topology file stored in the default directory, */home/sdn/SDN\_Labs /lab2* and click on *Open*.

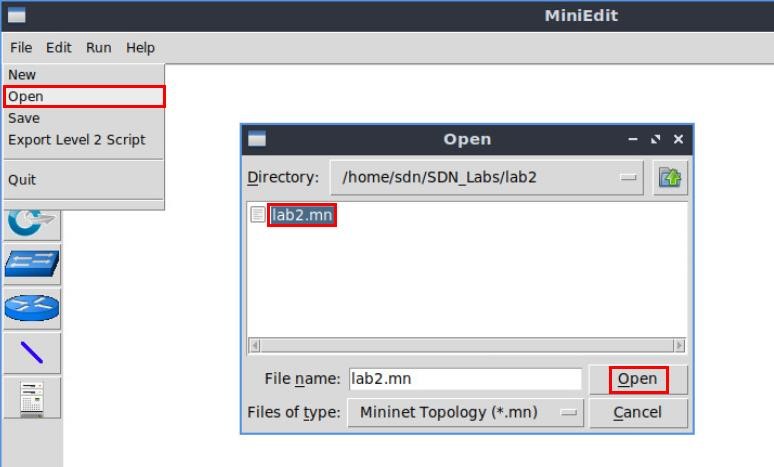


Figure 8. MiniEdit’s open dialog.

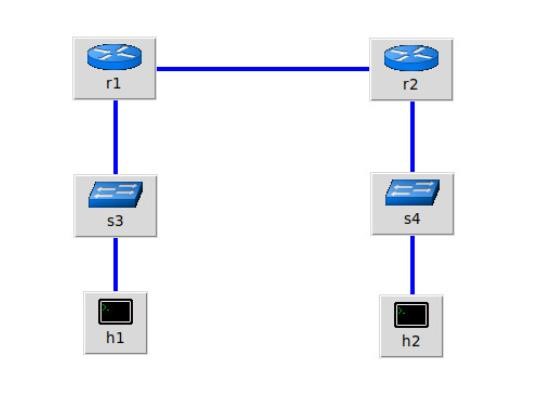


Figure 9. Mininet’s topology.

## 2.3 Loading the configuration file

At this point the topology is loaded. However, the interfaces are not configured. In order to assign IP addresses to the interfaces of the devices, you will execute a script that loads the configuration to the routers.

**Step 1.** Click on the icon below to open the Linux terminal.

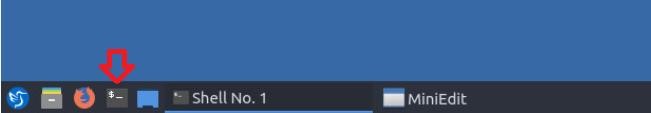


Figure 10. Opening Linux terminal.

|  |
| --- |
| cd |

**Step 2.** Navigate into *SDN\_Labs/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 interfaces of the router. The command is short for change directory followed by an argument that specifies the destination directory.

cd SDN\_Labs/lab2

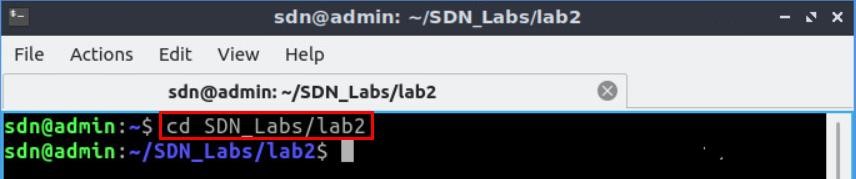


Figure 11. Entering the *SDN\_Labs/lab2* directory.

**Step 3.** 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

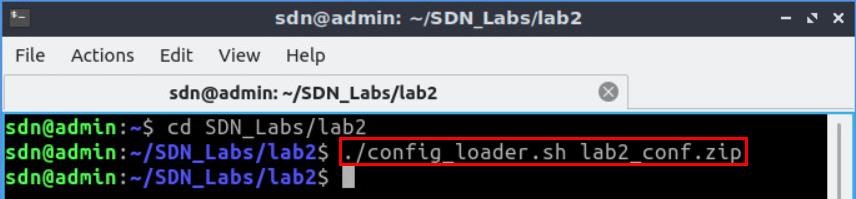


Figure 12. Executing the shell script to load the configuration.

**Step 4.** Type the following command to exit the Linux terminal.

exit

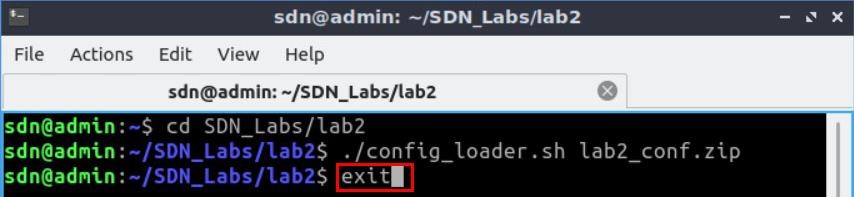


Figure 13. Exiting from the terminal.

## 2.4 Running the emulation

In this section, you will run the emulation and check the links and interfaces that connect the devices in the given topology.

**Step 1.** At this point host h1 and host h2 interfaces are configured. To proceed with the emulation, click on the *Run* button located in lower left-hand side.



Figure 14. Starting the emulation.

**Step 2.** Issue the following command to display the interface names and connections.

links

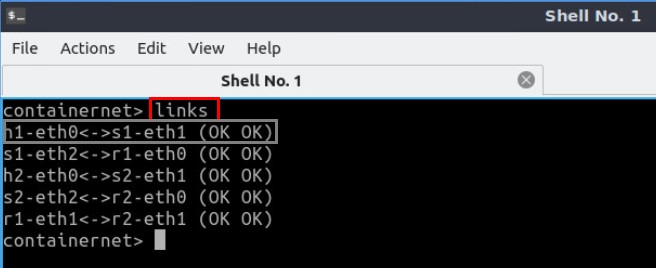


Figure 15. Displaying network interfaces.

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*).

## 2.5 Verify the configuration

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

**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.

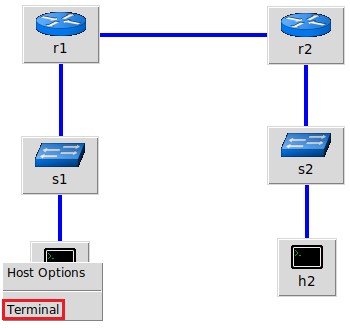


Figure 16. Opening a terminal on host h1.

**Step 2**. On host h1 terminal, type the command shown below to verify that the IP address was assigned successfully. You will corroborate that host h1 has two interfaces. Interface *h1-eth0* is configured with the IP address of 192.168.1.10 and the subnet mask 255.255.255.0. Interface *lo* is configured with the IP address of 127.0.0.1 with the subnet mask of 255.0.0.0.

ifconfig

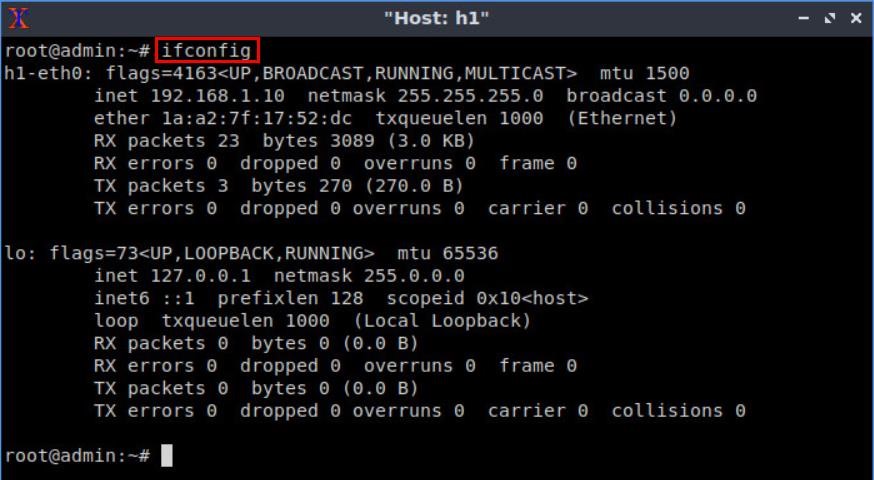


Figure 17. Output of command.

**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

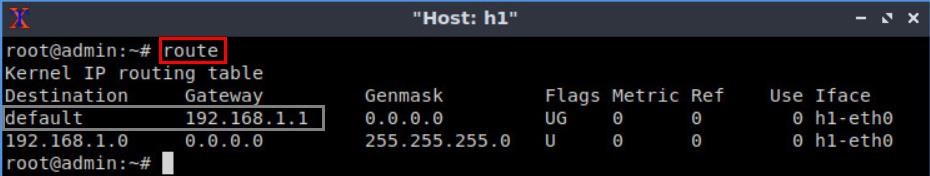


Figure 18. Output of command.

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

**Step 5**. In order to verify router r1, hold right-click on router r1 and select *Terminal*.

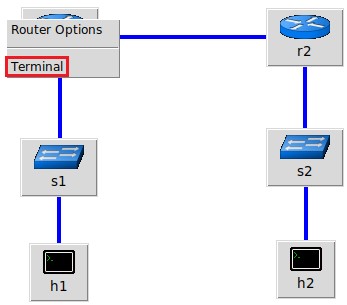


Figure 19. Opening a terminal on router r1.

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

zebra

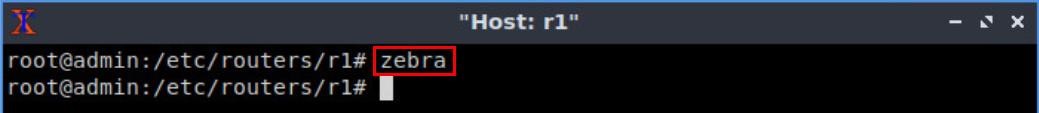


Figure 20. Starting zebra daemon.

**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

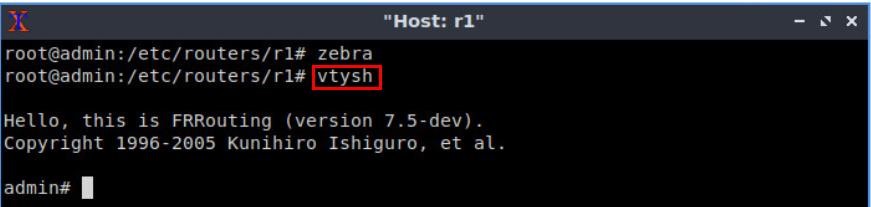


Figure 21. Starting vtysh on router r1.

**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 of router r2 (192.168.2.0/24) as there is no routing protocol configured yet.

show ip route

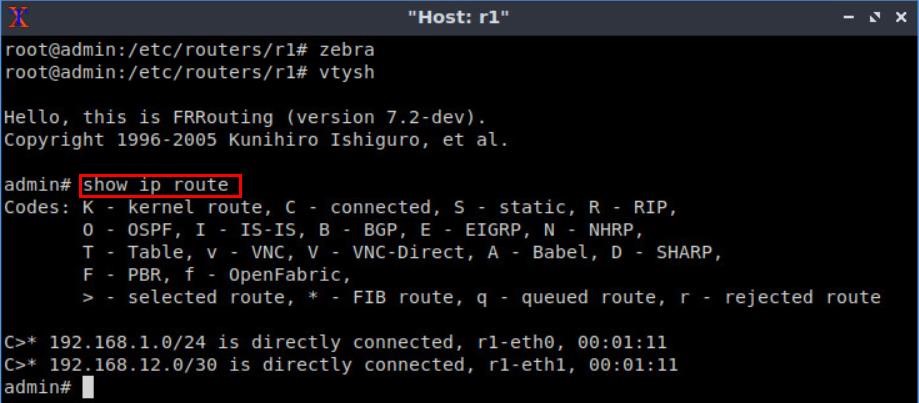


Figure 22. Displaying the routing table of router r1.

The output in the figure above shows that the network 192.168.1.0/24 is directly connected through the interface *r1-eth0*. The network 192.168.12.0/30 is connected via the interface *r1-eth1*.

**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.

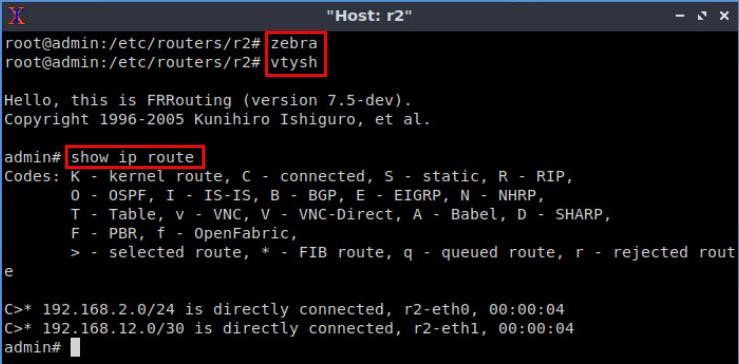


Figure 23. Displaying the routing table of router r2.

## 2.6 Test connectivity between end-hosts

In this section you will run a connectivity test between host 1 and host 2. You will notice that there is no connectivity because there is no routing protocol configured in the routers.

**Step 1.** On host h1 terminal, type the command shown below. Notice that according to Table 1, the IP address 192.168.2.10 is assigned to host h2.

ping 192.168.2.10

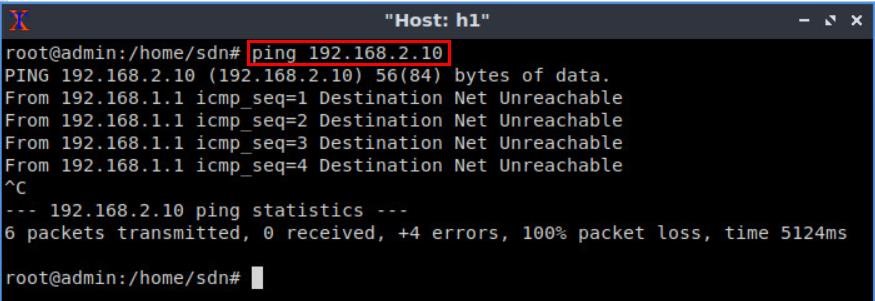


Figure 24. Connectivity test between host h1 and host h2.

|  |
| --- |
| Ctrl+c |

To stop the test press . The result in the figure above shows an unsuccessful connectivity test.

# 3 Configure BGP routing protocol

In this section you will configure a routing protocol in order to establish a connection between the two networks. You will configure BGP in order to establish a connection between AS 100 and AS 200. First, you will initialize the daemon that enables BGP configuration then. Then, you need to assign BGP neighbors to allow BGP peering to the remote neighbor. Additionally, you will advertise the local networks of each router.

## 3.1 BGP neighbors on the routers

In this section, you will add the neighbor IP address to allow BGP peering to the remote neighbor.

**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

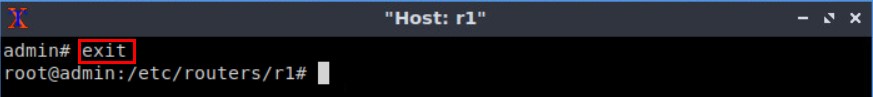


Figure 25. Exiting the vtysh session.

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

bgpd



Figure 26. Starting BGP daemon.

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

vtysh

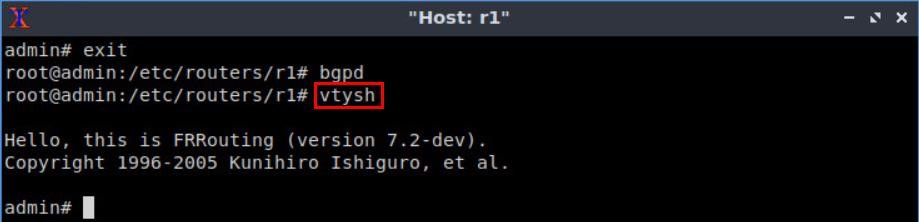


Figure 27. Starting vtysh on router r1.

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

configure terminal

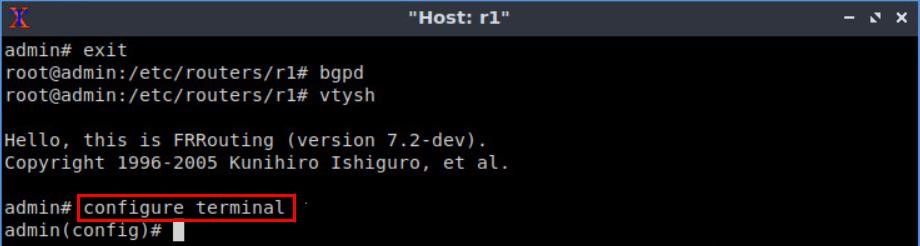


Figure 28. Enabling configuration mode on router r1.

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

router bgp 100

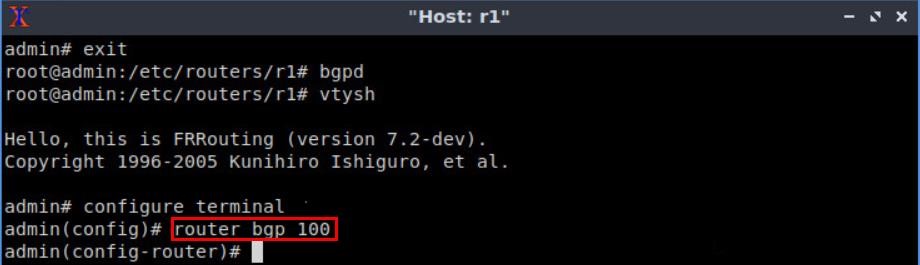


Figure 29. Configuring BGP on router r1.

**Step 6.** To configure a BGP neighbor to router r1 (AS 100), type the command shown below. This command specifies the neighbor IP address (192.168.12.2) and ASN of the remote BGP peer (AS 200).

neighbor 192.168.12.2 remote-as 200

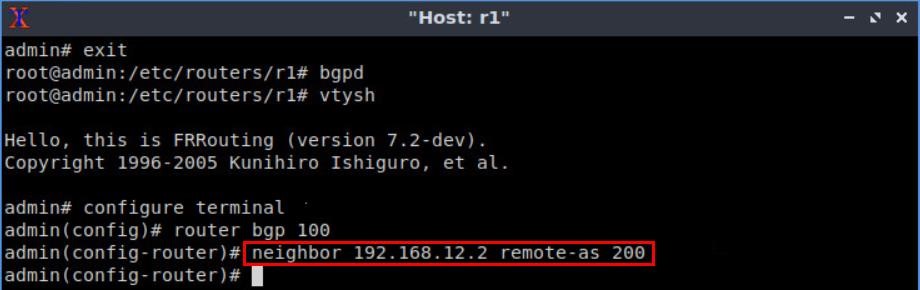


Figure 30. Assigning BGP neighbor to router r1.

**Step 7.** Type the following command to exit from the configuration mode.

end

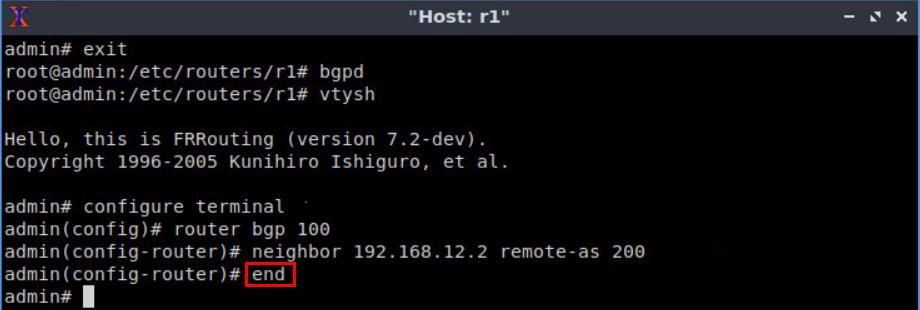


Figure 31. Exiting from configuration mode.

**Step 8.** Type the following command to verify BGP neighbors. You will verify that the neighbor IP address is 192.168.12.2. The corresponding ASN is 200.

show ip bgp neighbors

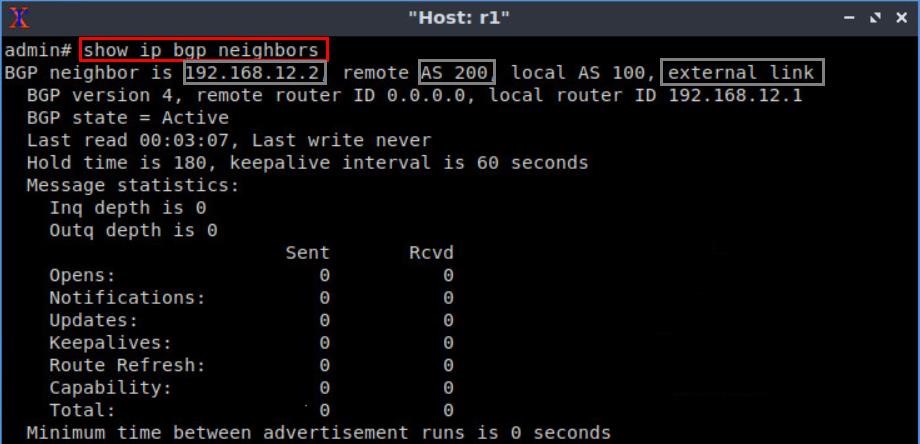


Figure 32. Verifying BGP neighbors on router r1.

**Step 9.** Router r2is 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.

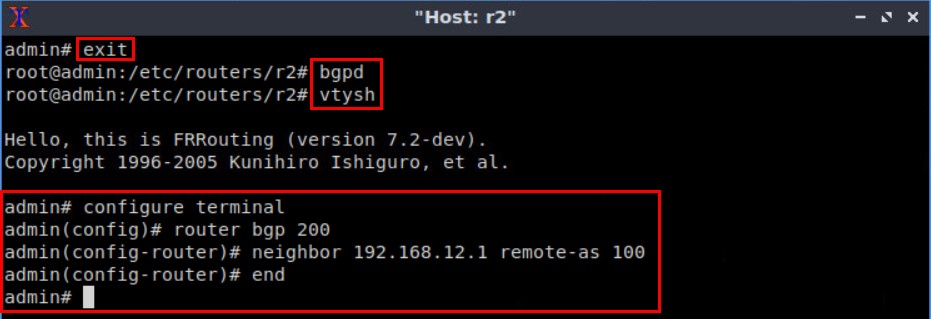


Figure 33. Assigning BGP neighbor to router r2.

**Step 10.** Type the following command to verify BGP neighbors. You will verify that the neighbor IP address is 192.168.12.1. The corresponding ASN is 100.

show ip bgp neighbors

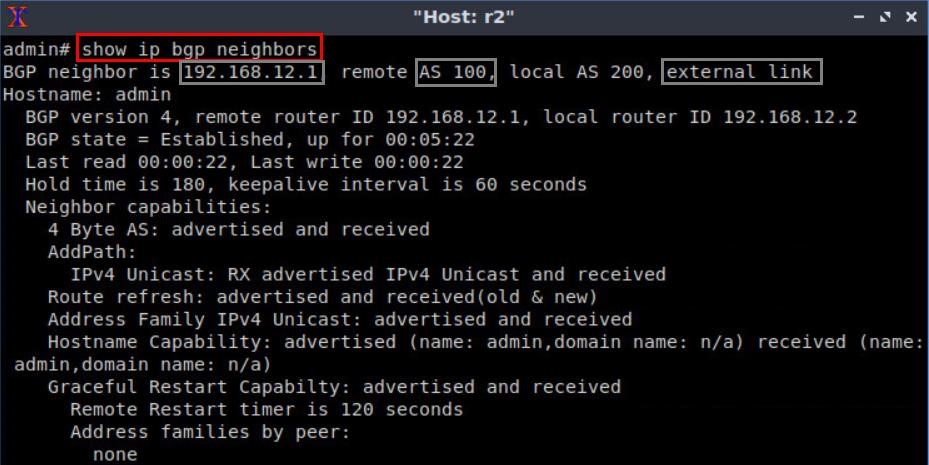


Figure 34. Verifying BGP neighbors on router r2.

**Step 11.** In router r2 terminal, perform a connectivity test by running the command shown below.

ping 192.168.12.1

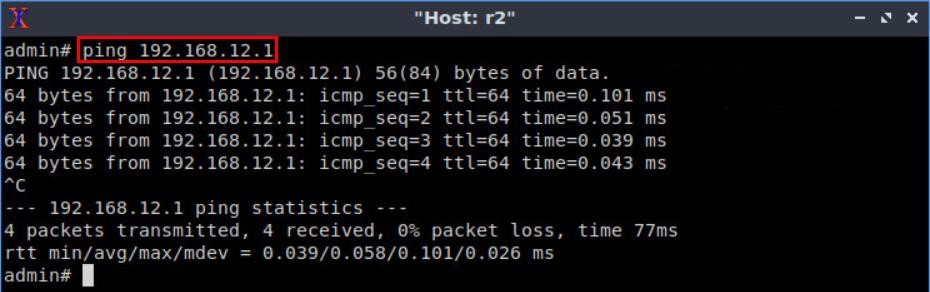


Figure 35. Connectivity test using command.

|  |
| --- |
| Ctrl+c |

To stop the test, press . The result in the figure above shows a successful connectivity test between router r1 and router r2.

**Step 12.** In router r2 terminal, perform a connectivity test between router r2 and host h1 by issuing the command shown below.

ping 192.168.1.10

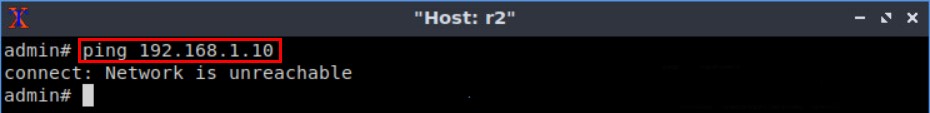


Figure 36. Connectivity test using command.

To stop the test, press Ctrl + c. As shown in the figure above, router r2 cannot reach host h1 at this point as the routing table of router r2 does not contain the network address of host h1.

## 3.2 Advertise local networks on the routers

In this section, you will advertise the LANs so that the neighbor can receive the network address through EBGP.

**Step 1.** In router r1 terminal, issue the following command.

configure terminal

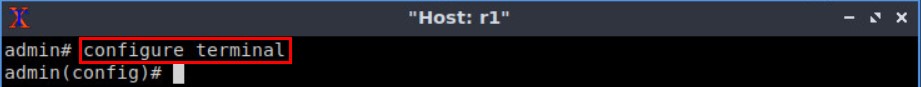


Figure 37. Enabling configuration mode on router r1.

**Step 2.** You will advertise the LAN connected to router r1 via BGP.Type the following command to enable BGP configuration mode.

router bgp 100

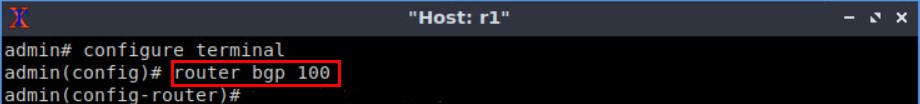


Figure 38. Entering to BGP configuration mode.

**Step 3.** Issue the following command so that router r1 advertises the network 192.168.1.0/24.

network 192.168.1.0/24

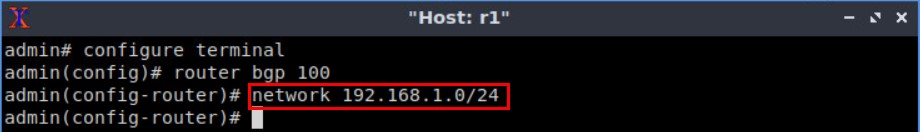
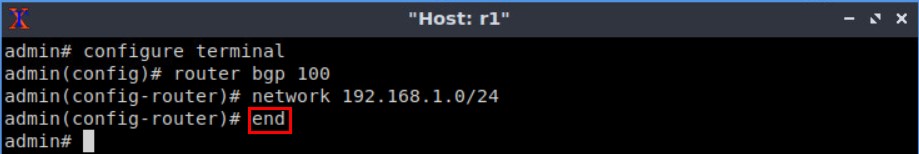


Figure 39. Advertising the network connected to router r1.

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

end

 Figure 40. Exiting from configuration mode.

**Step 5.** Type the following command to verify BGP networks.

show ip bgp

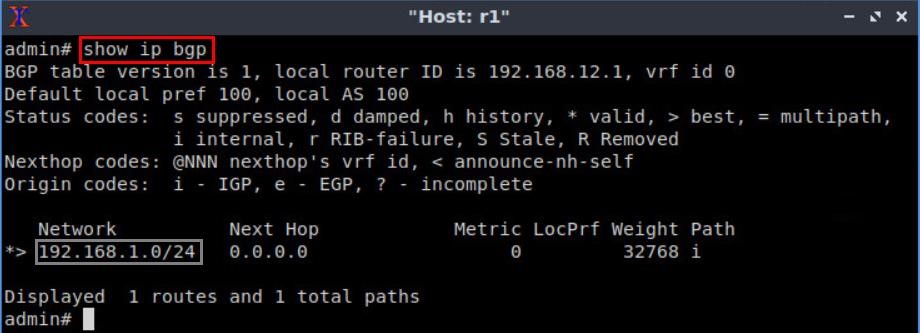


Figure 41. Verifying BGP networks on router r1.

**Step 6.** Type the following command to verify the routing table of router r2. You will observe the route to network 192.168.1.0/24, which is advertised by router r1. It also shows that router r2 will use the neighbor IP 192.168.12.1 to reach the network 192.168.1.0/24.

show ip route

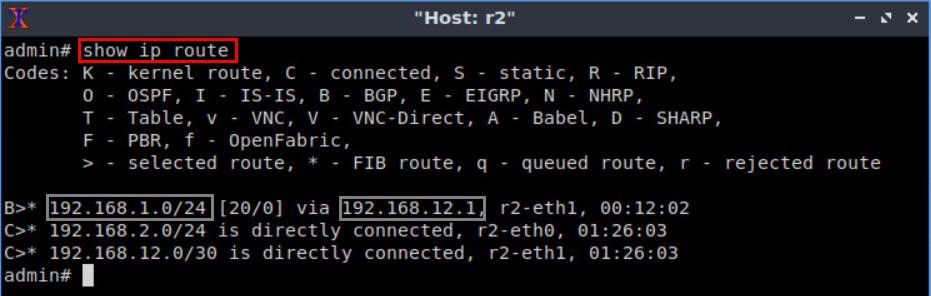


Figure 42. Verifying routing table of router r2.

**Step 7.** In order to verify the BGP table of router r2, issue the command shown below. The output indicates that the network connected to router r1 is listed in the BGP table of router r2. Additionally, it displays the next hop IP address (192.168.12.1) which corresponds to router r2’s neighbor IP address (router r1).

show ip bgp

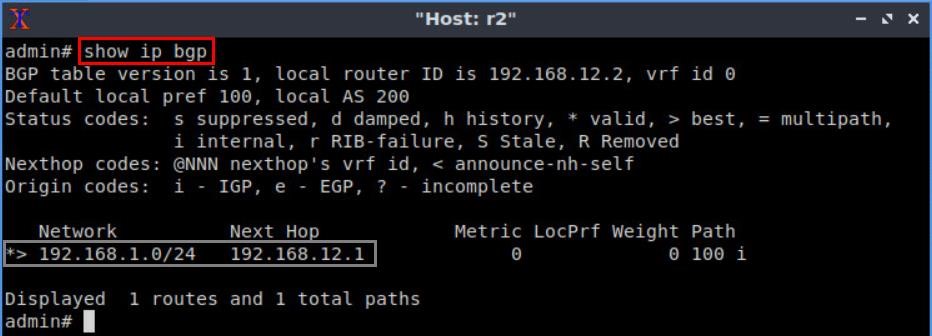


Figure 43. Verifying BGP table of router r2.

**Step 8.** Follow from step 1 to step 4 but with different metrics in order to advertise the LAN connected to router r2. All these steps are summarized in the following figure.

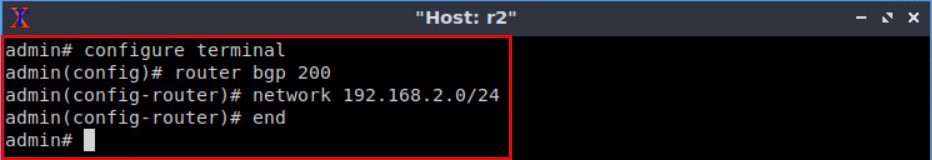


Figure 44. Advertising the network connected to router r2.

**Step 9.** In router r2 terminal, issue the following command to verify the BGP table of router r2. The output will list all the available BGP networks. In particular, the routing table contains its own network (192.168.2.0/24) and the remote network (192.168.1.0/24) which was advertised via EBGP.

show ip bgp

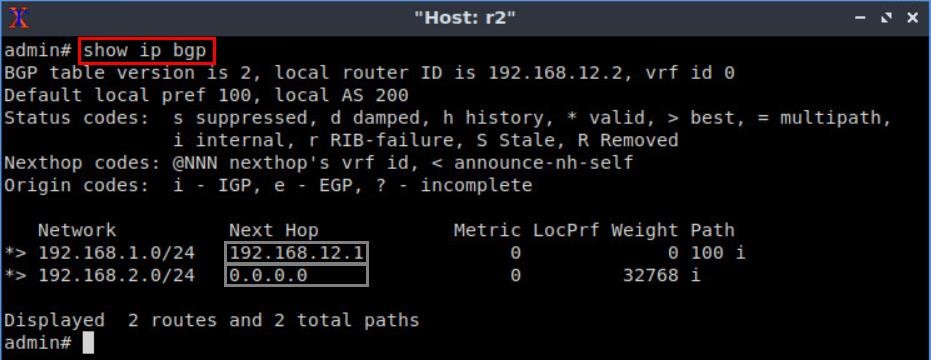


Figure 45. Verifying BGP table of router r2.

**Step 10.** In router r1 terminal, verify the routing table by typing the following command. The output lists that router r1 contains a route to the network 192.168.2.0/24. Notice that, this route was advertised by router r2.

show ip route

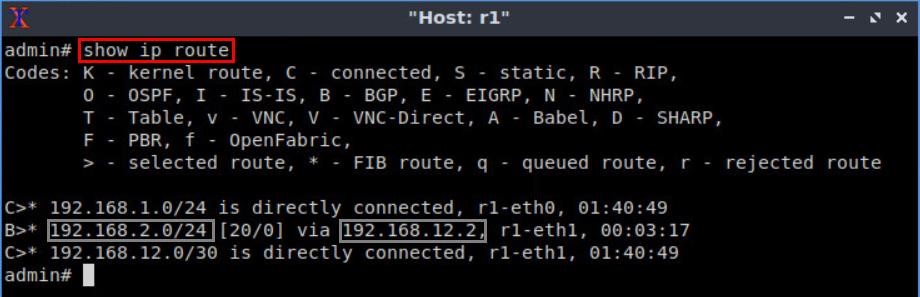


Figure 46. Verifying routing table of router r1.

# 4 Verify connections

In this section, you will verify that the applied configuration is working correctly by running a connectivity test between host h1 and host h2.

**Step 1.** On host h1 terminal, perform a connectivity test between host h1 and host h2 by issuing the command shown below.

ping 192.168.2.10

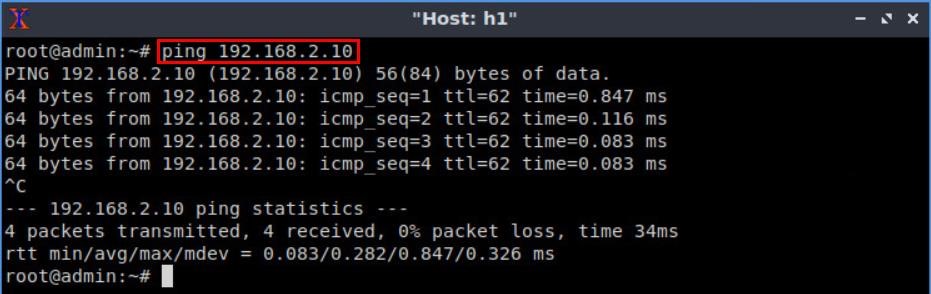


Figure 47. Connectivity test using command.

To stop the test, press Ctrl + c. The result in the figure above shows a successful connectivity test.

**Step 2.** Hold right-click on host h2 and select *Terminal*. This opens the terminal of host h2.

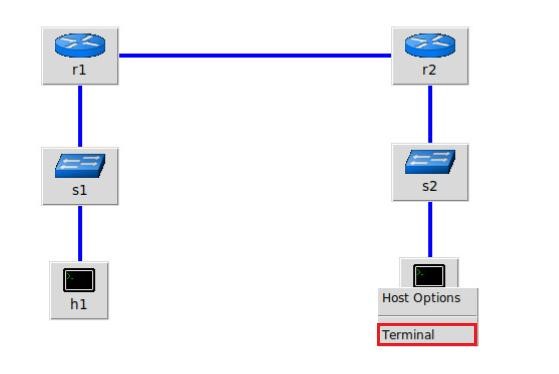


Figure 48. Opening host h2 terminal.

**Step 3.** Similarly, on host h2 terminal, perform a connectivity test between host h2 and host h1 by issuing the command shown below.

*ping 192.168.1.10*

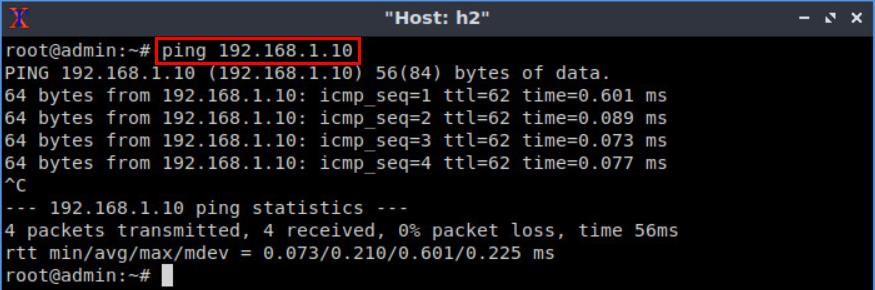


Figure 49. Connectivity test using command.

To stop the test, press Ctrl + c. The result in the figure above shows a successful connectivity test.

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

# References

1. Linux foundation collaborative projects, “*FRR routing documentation*”, 2017.

[Online]. Available: http://docs.frrouting.org/en/latest/

1. G. Malkin, “*RIP Version 2*,” RFC 2453 updated by RFC 4822, 1998. [Online].

Available: http://www.ietf.org/rfc/rfc2453.txt.

1. J. Moy, “*OSPF version 2*”, 1998. [Online]. Available:

https://www.hjp.at/doc/rfc/rfc2178.html

1. Y. Rekhter, T. Li, S. Hares, “*A border gateway protocol 4 (BGP-4),*” RFC 4271 updated by RFCs 6286, 6608, 6793, 2006. [Online]. Available:

http://www.ietf.org/rfc/rfc4271.txt.

1. D. Oran, “*OSI IS-IS intra-domain routing protocol,*” RFC 1142, 1990. [Online]. Available: [http://www.ietf.org/rfc/rfc1142.txt.](http://www.ietf.org/rfc/rfc1142.txt)
2. P. Jakma, D. Lamparter. “*Introduction to the quagga routing suite*,” 2014, *IEEE Network* 28.
3. K. Ishiguro, “*Gnu zebra,*”. [Online]. Available: *http://www. zebra. org* (2002).
4. A. Tanenbaum, D. Wetherall, “*Computer networks*”, 5th Edition, Pearson, 2012.
5. Mininet walkthrough. [Online]. Available: http://Mininet.org.
6. B. Lantz, G. Gee, “*MiniEdit: a simple network editor for Mininet,*” 2013. [Online]. Available: https://github.com/Mininet/Mininet/blob/master/examples.
7. P. Goransson, C. Black, T. Culver. “*Software defined networks: a comprehensive approach”*. Morgan Kaufmann, 2016