Lab

3

**Multiple Subnets – Static Routing**

What you will learn in this lab:

* How to turn a computer with multiple interfaces into a router
* How to set up static routing on Linux PC-routers and Cisco commercial routers
* How ICMP messages update routing table entries
* How Proxy ARP helps to connect different networks without reconfiguring the hosts
* How to work with different network prefixes

Updated: January 2021

**Table of Content**

[Study Material for Lab 3 3](#_Toc61874407)

[Linux Prelab3 4](#_Toc61874408)

[Lab 3 – Multiple Subnets – Static Routing 5](#_Toc61874409)

[Part 1. Exploring Routing Tables 6](#_Toc61874410)

[Exercise 1-a. Network setup 7](#_Toc61874411)

[Part 2. Configuring a Cisco Router 9](#_Toc61874412)

[Exercise 2-a. Switching between Cisco IOS command modes 9](#_Toc61874413)

[Exercise 2-b. Configuring IP interfaces on a Cisco router 12](#_Toc61874414)

[Exercise 2-c. Setting static routing table entries on a Cisco router 13](#_Toc61874415)

[Part 3. Configuring a Linux PC for a multi segment network 15](#_Toc61874416)

[Exercise 3-a. Setting static routing table entries for a Linux PC 15](#_Toc61874417)

[Part 4. Finalizing and Exploring the Router Configuration 17](#_Toc61874418)

[Exercise 4-a. Testing the configuration 18](#_Toc61874419)

[Exercise 4-b. Testing routes with traceroute (on PCs) or trace (on Routers) 18](#_Toc61874420)

[Exercise 4-c. Multiple matches in the routing table 19](#_Toc61874421)

[Part 5. Proxy ARP 20](#_Toc61874422)

[Exercise 5-a. Observing Proxy ARP 21](#_Toc61874423)

[Part 6. ICMP Redirect 23](#_Toc61874424)

[Exercise 6-a. Network setup 24](#_Toc61874425)

[Exercise 6-b. ICMP Redirect 24](#_Toc61874426)

[Part 7. Network Prefixes and Routing 28](#_Toc61874427)

[Exercise 7-a. Network setup 28](#_Toc61874428)

[Exercise 7-b. Exploring the role of prefixes (netmasks) at hosts 29](#_Toc61874429)

[Part 8. Stateless Autoconfiguration and Static Routing in IPv6 30](#_Toc61874430)

[Exercise 8-a. Network setup 31](#_Toc61874431)

[Exercise 8-b. Configuring an IPv6 router on Cisco IOS and stateless autoconfiguration 31](#_Toc61874432)

[Appendix A: Cisco IOS Command Line Interface 35](#_Toc61874433)

[User EXEC Mode 36](#_Toc61874434)

[Privileged EXEC Mode 37](#_Toc61874435)

[Global Configuration Mode 37](#_Toc61874436)

[Interface Configuration Mode 38](#_Toc61874437)

[Router Configuration Mode 38](#_Toc61874438)

[IOS Commands for Interface Configuration 39](#_Toc61874439)

[IOS Commands to Display the Configuration and Other Information 40](#_Toc61874440)

[Navigating the IOS Command Line Interface 42](#_Toc61874441)

[APPENDIX B. A TIP TO SPEED UP the SETUP of EXPERIMENTS 44](#_Toc61874442)

[APPENDIX C. Saving a Router Configuration in GNS3 Project 46](#_Toc61874443)

# Study Material for Lab 3

1. **traceroute**: Read about the traceroute command at

<https://en.wikipedia.org/wiki/Traceroute>

1. **ip route:** Read about the ip route command at

<http://linux-ip.net/html/tools-ip-route.html>

1. **Proxy ARP:** Read about Proxy ARP at

<https://www.practicalnetworking.net/series/arp/proxy-arp/>

1. **Navigating Cisco IOS:** Read the appendix of this lab, which describes the command line interface of the Cisco Internet Operating System (IOS).

## Prelab3

1. How does a router determine whether a datagram to a particular host can be directly delivered through one of its interfaces?
2. List four different methods for configuring (adding/removing/changing) routing table entries. **Note**: Whenever an IP address is configured on a system, a routing table entry is created for the subnet to which the IP address belongs.
3. Write the Linux command that sets the next hop of the default route to 10.0.1.3.
4. In which IOS command mode are static routing table entries added and removed.
5. Provide the IOS commands to enable and disable IPv4 and IPv6 forwarding on a Cisco router.
6. Provide the commands that enable IPv6 forwarding on a Linux system.
7. Write the Cisco IOS command that sets the next hop of the default route to 10.0.1.3.
8. Write the Cisco IOS command that removes the routing table entry for destination 10.2.0.0/16 with next hop IP address 10.2.3.4.
9. True or False? A router with Proxy ARP enabled, forwards ARP Request messages received on one subnet to a subnet where the host is located.
10. Which systems generate ICMP Redirect messages? Routers, hosts, or both?

# Lab 3 – Multiple Subnets – Static Routing

In this lab you work with three different network topologies. The topology and network configuration for Parts 1-5. is shown in Figure 3.1. *PC1* and *PC3* are used as hosts, and *Router1* and *Router2* are set up as IP routers. The PCs and the Cisco routers are connected by Ethernet switches. In Lab 3, all routing table entries are manually configured, a procedure known as *static routing*.

Timeline

Description automatically generated

Figure 3.1. Network topology for Parts 1-5.

|  |  |  |
| --- | --- | --- |
| PC | IPv4 address of *eth0* | IPv4 address of *eth1* |
| PC1 | 10.0.1.11/24 | – |
| PC3 | 10.0.3.33/24 | – |
| **Cisco Router** | **IPv4 address of** ***FastEthernet0/0*** | **IPv4 address of** ***FastEthernet1/0*** |
| Router1 | 10.0.2.1/24 | 10.0.3.1/24 |
| Router2 | 10.0.1.2/24 | 10.0.2.2/24 |

Table 3.1. IPv4 configuration for Parts 1-5.

## Part 1. Exploring Routing Tables

Whenever a system that runs IP protocols wants to transmit a packet, it consults its routing tables. Routing table entries are configured in a number of ways:

1. Whenever an IP address is configured on a system, a routing table entry is created for the subnet to which the IP address belongs.
2. Routing table entries can be manually added. This is referred to as *static routing*.
3. Some ICMP messages can modify or create routing table entries.
4. Routing protocols configure routing tables without manual intervention. This is referred to as *dynamic routing*.

Dynamic routing is covered in Lab 4. In this part of the lab, you observe the routing tables that are added through the configuration of an IPv4 address.

In addition to the routing tables, Linux systems and Cisco routers provide routing caches that offer a faster lookup of a destination IP address. The routing cache is consulted before the routing table.

On Linux, there are several commands to view the content of the routing table. Here are the most useful commands.

|  |
| --- |
| **Display Routing Table on Linux**  ip route  Displays the IPv4 routing table.    netstat -rn  Displays the current IPv4 routing table in a table format. The output of this command is easier to read than that of the ip route command.  ip -6 route  Displays the IPv6 routing table.  netstat -rn -6  Displays the current IPv6 routing table in a table format.  **Delete Routing Table on Linux**  sudo ip route del *10.0.2.0/24*  Deletes the routing table entry for destination *10.0.2.0/24*.  sudo ip route flush table main  Clears the IPv4 routing table. Be careful with this command since it also deletes the routing entries for directly connected networks. |

On Cisco routers, the commands are as given below

|  |
| --- |
| **IOS mode: privileged EXEC**  show ip route  show ipv6 route  Displays the contents of the IPv4/IPv6 routing table.  clear ip route \* clear ipv6 route \*  Deletes all IPv4/IPv6 routing table entries.  show ip route *10.0.3.33*  Lists the routing table entry that matches *10.0.3.33*  show ip cache  Displays the routing cache. The routing cache is a data structure for faster lookup of destination addresses. Entries in the routing cache are added when a routing table entry is looked up. |

### Exercise 1-a. Network setup

1. Connect the Ethernet interfaces of the Linux PCs and the Cisco routers as shown in Figure 3.1. To make configuration setup easier, you can display the device interfaces in GNS3 by clicking on the “Show/Hide interface labels” button at the top of the GNS3 screen. You can move the labels around to make reading them easier. You can also “Annotate” the configuration by adding a note for each device such as an IP Address for an interface or identifying a subnet, as shown here.

Graphical user interface, text, application

Description automatically generated Diagram

Description automatically generated with low confidence Diagram

Description automatically generated

1. On *PC1*, display the IPv4 and IPv6 routing tables before any IP address is configured using the netstat -rn or ip route command. Take snapshots of the output.
2. Configure the IP addresses of the interfaces of the PCs as given in Table 3.1. Here is the command to configure *PC1*:

PC1$ **sudo ip addr add 10.0.1.11/24 dev eth0**

1. Display the IPv4 routing tables on *PC1* and *PC3* using the netstat -rn command. Take screenshots of the output.

Observe the routing table entries that are created for each configured IP address.

* 1. Relate the routing table entries to the configured IP addresses.
  2. Explain the entries in the column with header *Gateway*.

1. Start a Wireshark traffic capture between *PC1* (eth0) and the Ethernet switch. 
2. In a console terminal of *PC1*, issue ping commands to *Router2*, *Router1* and *PC3*. Take a screenshot of the output of the ping commands.

PC1$ **ping -c3 10.0.1.2**

PC1$ **ping -c3 10.0.2.2**

PC1$ **ping -c3 10.0.2.1**

PC1$ **ping -c3 10.0.3.1**

PC1$ **ping -c3 10.0.3.33**

Observe the output and the captured traffic for each ping command.

1. Use the routing table of *PC1* to explain the outcome of the ping commands.
2. Which ping commands generated network traffic, e.g., ARP and ICMP packets?
3. Stop the Wireshark traffic capture.



Lab Questions/Report

1. Include the routing tables saved in Step 2. Explain the entries in the IPv6 routing table.
2. Include the screen captures from Step 4 and answer the questions in Step 4.
3. Include the screen capture from Step 6 and answer the questions in Step 6.

## Part 2. Configuring a Cisco Router

After configuring multiple IPv4 addresses on a Cisco router it is still not acting as an IPv4 router. To act as an IPv4 router, we explicitly tell it to forward IP packets. This is referred to as *enabling IP forwarding*. In addition, the routing table must be populated so that the router knows how to forward a packet.

Before discussing these steps, there are a few exercises to become familiar with navigating the IOS command line interface. The appendix has additional information.

### Exercise 2-a. Switching between Cisco IOS command modes

This exercise walks you through the different Cisco IOS command modes. It is important to understand the different modes, so you know where you are and what commands are accepted at any time.

1. Open a console window on *Router1*. Dependent on the configuration of a Cisco router, when you open a console on a router you may see the prompt:

Router1>

or the prompt:

Router1#

If you see the first prompt continue with Step 2. If you see the second prompt, continue with Step 3.

1. ***User EXEC* mode.**  The prompt

Router1>

indicates that you are in the u*ser EXEC* mode. In this mode, there is only a limited number of commands available, e.g., ping, telnet, trace. It is not possible to change the router configuration. To see which commands are available in this mode, type a question mark (?):

Router1> **?**

When the display pauses, press Enter to continue the list or hit Ctrl-C to stop the command.

Next, switch from the *user EXEC* mode to the *privileged EXEC* mode by typing

Router1> **enable**

Password: <*enable secret*>

Router1#

The enable password may have been disabled. In a real setting, for security reasons, it is good practice to have an enable password.

1. ***Privileged EXEC* mode.** When you see the prompt

Router1#

You are in the *privileged EXEC mode*, which corresponds to an administrator account on other operating systems. You cannot change the router configuration in this state. This requires to switch to a configuration mode. From the *privileged EXEC mode*, you can switch to any configuration mode without an additional password.

Again, type a question mark to display the available commands in this mode.

Router1# **?**

1. ***Global configuration mode.*** This mode is used to modify system wide configuration parameters. You enter this mode from the *privileged EXEC* mode by typing

Router1# **configure terminal**

Router1(config)#

In this lab you will use this mode to enable IP forwarding on the router and to create static routing table entries. You may want to type a question mark to display the available commands in this mode.

1. ***Interface configuration mode.*** This mode is used to configure a network interface. When you enter this mode, you must provide the name of the interface that you want to configure. The mode can be entered from the *privileged EXEC* mode, the *global configuration* modes, or the *interface configuration* mode for another network interface. If you continue from Step 4, and want to configure the network interface *FastEthernet/00*, you type

Router1(config)# **interface FastEthernet0/0**

Router1(config-if)#

If you want to configure another network interface, say FastEthernet1/0, you can jump directly to the interface configuration mode of that interface by typing:

Router1(config-if)# **interface FastEthernet1/0**

Router1(config-if)#

|  |  |
| --- | --- |
|  | **Note:**  When you enter the *interface configuration* mode, you must enter the name of the interface that you configure. Here, the network interface that is configured is **FastEthernet0/0**.  The names of the interface of your router are almost certainly different. If you are unsure about the interfaces and their name use the *privileged EXEC* command  Router1# **show protocols**  **Note** that for the 3640 Cisco router that we use in our labs the interfaces are **FastEthernetx/y**, where usually x=0,1 and y= 0 for the lab setups. E.g.,  Router1(config)# **interface FastEthernet0/0**  Or  Router1(config)# **interface FastEthernet1/0** |

1. **Leaving states.** To go back from the *interface configuration* to the *global configuration* mode, or from the *global configuration* mode to the *privileged EXEC* mode, use the exit command:

Router1(config-if)# **exit**

Router1(config)# **exit**

Router1#

The exit command takes you one step up in the command hierarchy. To directly return to the *privileged EXEC* mode from any *configuration mode*, use the end command:

Router1(config-if)# **end**

Router1#

Skip the next step if you also skipped Step 2.

1. To return from the *privileged EXEC* mode to the *user EXEC* mode, type:

Router1# **disable**

Router1>

Finally, to terminate the console session from the *user EXEC* mode, type:

Router1> **logout**

or

Router1> **exit**

|  |  |
| --- | --- |
|  | **Command Shortcuts/Abbreviations:**  Cisco IOS has shortcuts for commands and command parameters. You only need to type enough characters so that the command/parameter is unique. For example, instead of typing  Router1# **configure terminal** Router1(config)#  you can use the shortcuts  Router1# **conf t** *<Enter>* Router1(config)#  Experienced users know how many characters must be typed to make a given command/ parameter unique. Less experienced users can take advantage of the autocompletion feature with the *Tab* key. Typing the *Tab* key auto-completes a command or parameter as long as the typed characters make the command/parameter unique. For example,  Router1# **conf** *<Tab>* ***t*** *<Tab>*  autocompletes to  Router1# **configure terminal**  Taking advantage of the *Tab* key can be a big time saver! |

### Exercise 2-b. Configuring IP interfaces on a Cisco router

The following exercises use Cisco IOS basic commands needed to configure IP addresses on a Cisco router. You have seen some of the commands in Lab 1. Now direct your attention at the different prompts and IOS command modes.

1. Open a console window on *Router1* and *Router2*.
2. Configure *Router1* and *Router2* with the IP addresses given in Table 3.1. Shown below are the steps for *Router1*, repeat for *Router2*[[1]](#footnote-1).

Router1# **configure terminal**

Router1(config)# **no ip routing**

Router1(config)# **ip routing**

Router1(config)# **interface FastEthernet0/0**

Router1(config-if)# **ip address 10.0.2.1 255.255.255.0**

Router1(config-if)# **no shutdown**

Router1(config-if)# **interface FastEthernet1/0**

Router1(config-if)# **ip address 10.0.3.1 255.255.255.0**

Router1(config-if)# **no shutdown**

Router1(config-if)# **end**

The command *ip routing* enables IP forwarding on *Router1*. This is the command that turns Router1 in an IP router. The command *no ip routing* disables IP forwarding on Router1 and clears its routing table. The purpose of the latter command is to reset and clean up the routing table.

1. When you are done, use the following commands to check the changes you made to the router configuration:

Router1# **show protocols**

Router1# **show ip interface brief**

Router1# **show ip route**

Router1# **show running-config**

1. Analyze the outputs to ensure that you have configured the router correctly.

### Exercise 2-c. Setting static routing table entries on a Cisco router

Next you must add static routes to the routing table of Router1. The routing table must be configured so that it conforms to the network topology shown in Figure 3.1 and Table 3.1.

The IOS command to configure static routing is ip route*.* The command can be used to display, clear, add, and delete entries in the routing table. Below is a summary of the commands.

|  |
| --- |
| **IOS mode: privileged EXEC**  show ip route  Displays the contents of the routing table.  clear ip route \*  Deletes all routing table entries.  show ip cache  Displays the routing cache.  **IOS mode: global Configuration**  ip routing  Enables IP routing.  no ip routing  Clears the routing table and disables IP routing.  **ip route-cache**  Enables route caching. By default, route caching is enabled on a router.  no ip route-cache  Disables route caching.  ip route *10.0.1.0* *255.255.255.0* *10.0.2.2*  Adds a static routing table entry to subnet *10.0.1.0/24* with next hop router *10.0.2.2*. Note that the prefix length of the subnet is provided in terms of a netmask.  ip route *10.0.1.0* *255.255.255.0* *FastEthernet0/0*  Adds a static routing table entry to destination subnet 10.0.1.0/24. Here, the next hop information is given as a network interface (e.g., FastEthernet0/0).  **ip route 0.0.0.0 0.0.0.0 *10.0.2.2***  Adds router 10.0.2.2 as the default gateway. In Cisco, the default gateway is called the “gateway of last resort.”  **no ip route *10.0.1.0* *255.255.255.0* *10.0.2.2***  **no ip route 0.0.0.0 0.0.0.0 *10.0.2.2***  Adding a “no” before an ‘ip route’ command deletes a route table entry. |

Now you are ready to configure the routing tables at Router1.

1. On *Router1*, display the content of the routing table with show ip route*.* Note the routing entries that are already present.

As in Linux, whenever an IP address is configured for a network interface, a routing table entry for the directly connected network is added automatically.

1. There are 3 subnets in Figure 3.1: 10.0.1.0/24, 10.0.2.0/24, and 10.0.3.0/24, and *Router1* needs to have routing table entries for each subnet. Since *Router1* already has table entries for 10.0.2.0/24 and 10.0.3.0/24, only the entry for 10.0.1.0/24 must be added. Likewise, *Router2* has table entries for 10.0.2.0/24 and 10.0.1.0/24, only the entry for 10.0.3.0/24 must be added.

On *Router1*, add a routing table entry for network 10.0.1.0/24 with 10.0.2.2 as next hop router. Note that you must be in the global configuration mode. Repeat for *Router2* adding network 10.0.3.0/24 with 10.0.2.1 as next hop router.

1. Return to the privileged EXEC mode. Display the routing table again with show ip routeand take a screenshot of the output for each router.

After the configuration of *Router1* and *Router2* we will have *PC1* ping *PC3* and see what happens.

1. Start Wireshark on eth0 interface on *PC1*.

**Note**: the following *filter* in Wireshark will make reading the output much easier for any labs with the Cisco routers in the network configuration.

Open the add filter toolbar to add the expression “!(loop || cdp)” and give it a label such as “No cisco data” or something like that. Save the expression. You can now invoke it every time you turn on Wireshark with routers in the configuration.

A screenshot of a computer

Description automatically generated

This will filter out the “loop” and “cdp” packets that the Cisco routers send periodically and that can fill up your Wireshark capture data quickly. Anytime we use the Cisco routers in any of the labs be sure to use this filter for your Wireshark captures.

1. Ping *PC3*:

**PC1$ ping -c3 10.0.3.33**

From the Wireshark output you will observe that *PC1* and *PC3* can still not talk to each other. There is one piece missing: (1) the routing tables of all PCs must be configured to include remote subnets. This will be done in the next exercise.

Lab Questions/Report:

1. Include the routing table saved in Step 3. Explain the fields of the routing table entries of the Cisco router. Explain how the routing table has changed from Step 1 to Step 3.
2. What reason is given for the failed ping on P*C1’s* console? Can you explain?

## Part 3. Configuring a Linux PC for a multi segment network

Enabling PCs and Cisco routers to perform IPv4 forwarding is not sufficient to solve the problem of failing pings. The reason is that the PCs need to know how to reach other subnets.

### Exercise 3-a. Setting static routing table entries for a Linux PC

The next step is to set up the routing tables of the Linux PCs, *PC1* and *PC3*. The routing tables need to be set so that they conform to the network topology shown in Figure 3.1 and Table 3.1. The routes will be configured manually, which is also referred to as *static routing*.

Configuring static routes in Linux is done with the command ip route, which has numerous options for viewing, adding, deleting, or modifying routing entries. The various uses of the ip route command are summarized below. The ip route command can be used for both IPv4 and IPv6 routes.

|  |
| --- |
| **Defining Static Routes in Linux**  **sudo ip route add *10.0.2.0/24* via *10.0.1.2***  Adds a routing table entry for the subnet 10.0.2.0/24 with next hop 10.0.1.2. IP packets with a destination address in this subnet will be forwarded to IP address 10.0.1.2. The next hop is an IP router that must be on the same subnet as the system where the routing table entry is added.  **sudo ip route add default via *10.0.1.2***  Adds the router with IP address 10.0.1.2 as default gateway. This means that, if there is no other match in the routing table for an IP destination address, the IP packet will be forwarded to the default gateway.  **sudo ip route del *10.0.2.0/24***  **sudo ip route del default**  The commands delete an entry from the routing table. |

|  |  |
| --- | --- |
| A picture containing light  Description automatically generated | **Deleting all static routing table entries**  Another way to delete all static routing entries in the routing table associated with an interface is to bring down that interface, and then bring it up.  **sudo ip link set dev eth0 down** **sudo ip link set dev eth0 up**  With these commands, the routing table entries for the configured IP addresses will remain. |

The following steps ask you to configure the routing tables of *PC1* and *PC3*.

1. Icon

   Description automatically generatedConfigure a default route on *PC1* via *Router2* (interface FastEthernet0/0 on subnet 10.0.1.0) with the configuration command

PC1$ **sudo ip route add default via 10.0.1.2**

1. Icon

   Description automatically generatedDisplay the routing table of *PC1* with netstat -rn and take a screenshot of the routing table.
2. Retry the ping commands from Step 6 in Exercise 1. Observe the output of the ping command and the traffic that is captured by Wireshark. Take a snapshot of the output at *PC1*.

* Some of the ping commands that previously failed, now work. Which ones are they, and why do they now work?
* Some of the ping commands that previously failed, still fail. Which ones are they it, and why do they fail?

1. Start traffic capture with Wireshark for
   * *PC1* (for eth0),
   * *Router2* (for FastEthernet1/0), and
   * *PC3* (for eth0*)*
2. On *PC1*, run a ping command that continuously probes the IP address of *PC3*, with the command

PC1$ **ping 10.0.3.33**

Describe the traffic that is captured by the Wireshark applications.

1. The ping from *PC1* to *PC3* continues to fail! The reason is that, while *PC3* receives *ICMP Echo Requests* from *PC1*, it does not have a route back to *PC1*. We can provide such a route by defining a default gateway for *PC3* (we could also define a static route for this purpose). The command is

PC3$ **sudo ip route add default via 10.0.3.1**

Icon

Description automatically generatedOnce the route is added, observe how the captured traffic by Wireshark at *PC1* and *Router2*, changes. Also the output at *PC1* changes. Is the ping now successful? .Describe your observation.

1. Display the routing table of *PC3* with ‘’netstat -rn” and take a screenshot of the result.
2. Stop the instances of Wireshark.

Lab Questions/Report:

1. Include the screenshot of the routing table of PC1 from Step 1. Explain how you identify the default route.
2. Include the screenshot in Step 3 and provide the answer to the question in Step 3.
3. Describe your observations in Step 5.
4. Describe your observations in Step 6.
5. Include the routing table of *PC3* from Step 7.

## Part 4. Finalizing and Exploring the Router Configuration

If the configurations in Parts 1-3 were done correctly, it is now possible to send IP datagrams between any two machines in the network shown in Figure 3.1. If the network is not configured properly, you need to debug and test your setup. The table below illustrates common problems that may arise.

You can use Wireshark to check if traffic is transmitted or received at an interface.

|  |  |  |
| --- | --- | --- |
| Problem | Possible Causes | Debugging |
| ARP Requests are sent but there is no ARP Reply | IP address does not exist on the network | Check the network topology  Verify that the IP addresses are correct |
| PC displays “Network unreachable” when trying to send to a destination | There is no routing table entry for the destination | Verify the content of the routing table. |
| Traffic does not reach destinations on local network | Network interface not configured correctly.  Incorrectly connected, faulty, or loose cables. | Verify the interface configuration with show protocols (in IOS) or ip addr show (in Linux)  Most interface cards and Ethernet hubs/switches have green LED status lights. Check if the status lights are on.  Verify the connection of the cables.  Verify that no cross-over cables are used. |
| Traffic reaches router, but is not forwarded to remote networks | IP forwarding is not enabled.  Routing tables are not configured correctly. | Use show protocols (in IOS)   Display routing tables with show ip route(in IOS)ornetstat -nr(in Linux). Run traceroute between hosts and routers. |
| ICMP Echo Request reaches destination, but ICMP Echo Reply does not reach source | Routing tables are not correctly configured for the reverse path. | Display routing tables with show ip route(in IOS) or netstat -nr(in Linux). Run ping and traceroute in both directions. |
| A change in the routing table has no effect on the flow of traffic. | The neighbor cache has old entries. | Delete the neighbor cache. |

### Exercise 4-a. Testing the configuration

1. Continue with the network configuration of Part 3.
2. Test the network configuration by issuing ping commands between the PCs and the routers. On *PC1*, run the same commands from Part 1.

PC1$ **ping –c3 10.0.1.2**

PC1$ **ping –c3 10.0.2.2**

PC1$ **ping –c3 10.0.2.1**

PC1$ **ping –c3 10.0.3.1**

PC1$ **ping –c3 10.0.3.33**

If all ping commands are successful, the configuration is correct. Otherwise, you need to debug where the commands fail and take corrective actions. You may also use the *traceroute* command described in the next exercise to find where the forwarding fails.

### Exercise 4-b. Testing routes with traceroute (on PCs) or trace (on Routers)

*Traceroute* is a useful command for determining the routers that are traversed on a path from the source to the destination.

1. Icon

   Description automatically generatedStart a Wireshark session on *PC1* (eth0).
2. Execute a traceroute command from *PC1* to *PC3*, and from *Router1* to *PC1*. Take a snapshot of both consoles. Save the output.

PC1$ **traceroute 10.0.3.33**

**Route1# trace 10.0.1.11**

1. Observe how Wireshark gathers information on the route. Pay attention to the TTL fields in the UDP packets that are sent by *PC1*.
2. Stop the traffic capture of Wireshark and save the captured traffic. (Save the summary lines of the packets as well as the details of the packets.)

Lab Questions/Report

1. Using the saved Wireshark data, provide a list of the summary lines of the captured UDP and ICMP packets.
2. For each captured UDP packet, provide the TTL value set by *PC1*.
3. Provide a brief description of the operation of *tracerout*e and *trace* from the output on the consoles and the Wireshark data.

### Exercise 4-c. Multiple matches in the routing table

When there are multiple matches for a destination address in the routing table, the system selects the most specific route. This is called *longest prefix match*. In this exercise, you observe the longest prefix match in action.

For querying the routing table, you can use the commands explained below.

|  |
| --- |
| **Testing routing table lookups in Linux**  ip route show to match *10.0.1.11*  Displays all matches in the IP routing table for destination *10.0.1.11*.  ip route get *10.0.1.11*  Displays the longest prefix match for destination *10.0.1.11*. |

1. Add the following routes to the routing table of *PC1*:

PC1$ **sudo ip route add 10.0.0.0/16 via 10.0.1.71**

PC1$ **sudo ip route add 10.0.3.9 via 10.0.1.81**

PC1$ **sudo ip route add default via 10.0.1.91**

In the second command, no prefix length is given. This is interpreted as a **32-bit long prefix**, that is, *10.0.3.9/32.*This type of routing table entry is called a ***host*** route. If a PC has a default route already assigned, you cannot add a second.

1. Take a snapshot of the IPv4 routing table at *PC1* (using netstat -rn)
2. Use the command ip route show to match **IP address/prefix** to determine how many matches exist in the routing table of *PC1* for the following IP addresses:

*10.0.3.9*

*10.0.3.14*

*10.0.4.1*

*10.1.4.18*

Use the ip route get command to determine which match is selected.

**Note:** It is not relevant to the exercise that devices with IP addresses *10.0.1.71, 10.0.1.81,* and *10.0.1.91* do not exist in the network.

1. Take screenshots of the output of the commands in Step 3.

Lab Question/Report

1. Include the saved screenshots and explain how *PC1* resolves multiple matches in the routing table.
2. Express the ranges of destination addresses that **do not** use the default route. (You can express the ranges as one or more network prefixes.)
3. Express the ranges of addresses that are forwarded to 10.0.1.71 (Here, it is best to not provide the network prefixes, since it would be a long list.)

## Part 5. Proxy ARP

The ARP protocol resolves IPv4 addresses to MAC addresses. Proxy ARP is a configuration option, where an IP router responds to ARP Requests that arrive from one of its connected networks for a host that is on another one of its connected networks. Proxy ARP can help with establishing a data exchange between hosts that have a different network prefix length.

In this part, you will explore how Proxy ARP enables routers to forward an IP datagram even though the sender of the datagram is not aware of this router. We continue with the network configuration from Figure 3.1 and with IP addresses as shown in Table 3.1.

Here are commands to view and refresh the neighbor caches.

|  |
| --- |
| **IOS mode: privileged EXEC**  show arp  Displays the neighbor cache.  clear arp-cache  Refreshes the ARP cache by issuing ARP Requests for each entry in the cache.  **Note:** The command does not delete entries in the neighbor cache. |

The commands to enable and disable Proxy ARP in IOS for a network interface are given below.

|  |
| --- |
| **IOS mode: interface configuration**  ip proxy-arp  no ip proxy-arp  Proxy ARP is enabled/disabled separately on each interface. In IOS, proxy ARP is enabled by default. |

|  |  |
| --- | --- |
|  | **Deleting neighbor cache on Cisco routers**  There is no reliable command that delete the neighbor cache. If this is desired, we recommend to disable all network interfaces. For example, for interface *FastEthernet0/0,* the commands are  Router1# **configure terminal**  Router1(config)# **interface FastEthernet0/0**  Router1(config-if)# **shutdown**  Router1(config)# **end**  If this is done for all interfaces, `show arp’ does not display any entries. However, when re-enabling the interfaces neighbor cache entries sometimes appear again. The commands to enable interface *FastEthernet1/0* are  Router1# **configure terminal**  Router1(config)# **interface FastEthernet1/0**  Router1(config-if)# **no shutdown**  Router1(config)# **end** |

### Exercise 5-a. Observing Proxy ARP

1. The network configuration is as given in Figure 3.1.
2. Erase the neighbor caches at *PC1* and *PC3* (sudo ip neigh flush all).
3. On *PC3*, delete the default route (see Part 3 of this lab).
4. On *Router1*, enable Proxy ARP on the FastEthernet interfaces of *Router1*. To enable Proxy ARP use the following commands (Note that the names of the interfaces may be different).

Router1# **configure terminal**

Router1(config)# **interface FastEthernet0/0**

Router1(config-if)# **ip proxy-arp**

Router1(config-if)# **exit**

Router1(config)# **interface FastEthernet1/0**

Router1(config-if)# **ip proxy-arp**

Router1(config-if)# **end**

1. Change the IPv4 address of *PC3* (eth0) to 10.0.3.33/8. To change the IPv4 address, first delete the old address (ip addr del) and then add the new address (ip addr add)

By changing the prefix length, *PC3* now believes that it is part of subnet *10.0.0.0/8*.

1. For the observations of this experiment, we need to run multiple instances of Wireshark. Start Wireshark traffic captures for

* *PC3* (eth0)*,*
* *Router2* (FastEthernet1/0), and
* *PC1* (eth0).

1. Set a display filter to only display ICMP and ARP packets.

Record the MAC addresses of all interfaces in the network shown in Figure 3.1. You can do this with the commands show interfaces (on *Router1, Router2*) and ip addr (on *PC1*, *PC3*).

An alternative is to send ping messages to the neighbors in the topology, e.g, *PC1⭤ Router2, Router2⭤ Router1, Router1⭤ PC3*) and check the MAC addresses in the packets captured by Wireshark.

1. Issue a ping from *PC3* to *PC1*:

PC3**$ ping -c2 10.0.1.11**

Even though *PC3* has no default routing entry in its table for *Router1*, it is still able to exchange traffic with *PC1*. Explore the captured data to explain this outcome. In particular:

* 1. Note that *PC3* sends an ARP Request for 10.0.1.11, which is replied to by *Router1*.
  2. What is the MAC address that *Router1* sends to *PC3*? Compare this MAC address to the addresses recorded in Step 7.
  3. Go to the Wireshark capture of *PC3* (eth0). Look for an *ICMP Echo Request* sent by *PC3* to *Router1* and check the destination IP and MAC addresses.

1. Display the neighbor cache of *PC3* and take a screenshot.
2. Stop Wireshark on *PC1*, *Router2*, and *PC3*. Save the traffic that is captured by Wireshark.
3. Now, disable Proxy ARP on both interfaces of *Router1* and delete the neighbor cache of *PC3*. Then repeat the ping from Step 7.  
   Is it still feasible to issue a ping from *PC3* to *PC1*?
4. Reset the IP address of *PC3* (eth0) to its original value of10.0.3.33/24and re-enable Proxy ARP on *Router1*.

Lab Question/Report

1. Using the records from Step 7, provide a table that lists, for each interface used in the network from Figure 3.1, the MAC and IPv4 addresses.
2. Include the screenshot of the neighbor cache at *PC3* (Step 9). Compare the cache entry for IP address 10.0.1.11 with the table from the previous question.
3. Provide a time-sequence diagram that shows the order in which the packet transmissions, which were observed by the three Wireshark instances, occur. Below is an example of such a diagram. Refer to <https://www.cisco.com/c/en/us/td/docs/ios/sw_upgrades/interlink/r2_0/api_con/actime.html>

for more examples of time-sequence diagrams. Include only relevant data from the packets:

* For ARP Request: Write “ARP Request” and the requested IP address;
* For ARP Reply: Write “ARP Reply” and the MAC address for the requested IP address;
* For ICMP Echo Request and Reply: Only write “Echo Request” or “Echo Reply”.

Diagram

Description automatically generated

## Part 6. ICMP Redirect

ICMP Redirect messages are sent from a router to a host, when a datagram should have been forwarded to a different router or interface. In Linux, an ICMP Redirect message updates the *routing cache*, but not the *routing table*.

|  |  |
| --- | --- |
|  | **Routing caches in Linux and Cisco IOS** Both the routing cache and the routing table contain information for forwarding traffic. In the past, Linux used the routing cache for a faster lookup. When a Linux system performed a routing table lookup, it first inspected the routing cache. If no matching entry was found in the cache, the routing table is inspected. Starting with Linux kernel 3.6, the routing cache is no longer used in this fashion.  However, for some functions, including route redirection, the routing cache still plays a role. A match for a destination address in the routing cache always takes precedence over a match in the routing table.  In Cisco IOS, the routing cache plays a bigger role in accelerating the routing table lookup.  When changing a routing table entry, it is always a good idea to flush the routing cache. Otherwise, it may happen that the changed routing table entry does not have an immediate effect. |

Relevant commands that deal with the routing cache are given below.

|  |
| --- |
| **Linux**  **ip route show cache**  Displays the routing cache.  **ip route flush cache**  Deletes the routing cache.  **IOS Mode: interface configuration**  **ip route-cache**  Enables the routing cache.  **no ip route-cache**  Disables and deletes the routing cache. |

The network topology is shown in Figure 3.2. Table 3.2 describes the IPv4 configuration of the network.

Diagram

Description automatically generated

Figure 3.2. Network topology for Part 6.

Table 3.2. IP Addresses for Part 6.

|  |  |  |  |
| --- | --- | --- | --- |
| Cisco Router | Interface FastEthernet0/0 | Interface FastEthernet1/0 | Default Gateway |
| Router1 | 10.0.2.1/24 | 10.0.3.1/24 | 10.0.2.2 |
| Router2 | 10.0.1.2/24 | 10.0.2.2/24 | 10.0.2.1 |
| **Linux PC** | **Interface *eth0*** | **Interface *eth1*** | **Default Gateway** |
| PC1 | 10.0.1.11/24 | Disabled | 10.0.1.2 |
| PC2 | 10.0.2.22/24 | Disabled | 10.0.2.2 |
| PC3 | 10.0.3.33/24 | Disabled | 10.0.3.1 |

### Exercise 6-a. Network setup

The network topology can be constructed from the topology in Figure 3.1, by connecting *PC2* (eth0) interface to the switch between *Router1* and *Router2*.

1. Connect the topology as shown in Figure 3.3. Starting from the topology in Figure 3.1, connect the link of *PC2* (eth0) to the switch as shown in Figure 3.3. Configure *PC2’s* IP address as given in Table 3.2
2. Check the IPv4 addresses configured on *PC1, PC2, PC3*, and *Router1* and *Router2* to ensure that they are as given in Table 3.2.
3. Set the default gateways of *PC1* and *PC3* as shown in Table 3.2.

### Exercise 6-b. ICMP Redirect

In the network shown in Figure 3.3, when *PC2* sends datagrams with destination 10.0.3.33 (*PC3*) to 10.0.2.2 (*Router2*), as opposed to 10.0.2.1 (*Router1*), then *Router2* sends an ICMP Redirect to *PC2*. The ICMP Redirect informs *PC2* that it should send datagrams with destination 10.0.3.33 to *Router1* instead.

In this exercise, you create the above scenario. First, you will trigger the transmission of an ICMP Redirect message and subsequently observe a change to the routing cache.

1. First, configure *PC2*.
2. Delete the entries in the neighbor cache entries by issuing

PC2$ **sudo ip neigh flush all**

1. Set the default gateway of PC2 to 10.0.2.2 with

PC2$ **sudo ip route add default via 10.0.2.2**

1. Delete the route cache with the command

PC2$ **sudo ip route flush cache**

1. Run “netstat -rn” to confirm that the default route is correctly configured. Take a screenshot of the output.
2. Continue on *PC2*. These days, Linux systems ignore ICMP Redirect message by default, since they present a vulnerability (which allows an attacker to modify the routing behavior). This command checks on setting of the accept\_redirects parameter. If =0, -> does not accept redirects.

PC2$ **sudo sysctl -a |grep accept\_redirects**

We want *PC2* to accept ICMP Redirects. Change the setting to 1 with this command

PC2$ **sudo sysctl** **-w** **net.ipv4.conf.all.accept\_redirects=1**

1. Next, proceed to *Router2*.
2. Set the default route at *Router2* to *Router1* (interface on common subnet)

Router2# **conf term**

Router2(config)# **ip route 0.0.0.0 0.0.0.0 10.0.2.1**

Router2(config-if)# **end**

1. To ensure that cached (old) information does not interfere with this experiment, delete the the route cache of each interface. This is done with the following commands:

Router2# **conf term**

Router2(config)# **interface FastEthernet0/0**

Router2(config-if)# **no ip route-cache**

Router2(config-if)# **ip route-cache**

Router2(config)# **interface FastEthernet1/0**

Router2(config-if)# **no ip route-cache**

Router2(config-if)# **ip route-cache**

Router2(config-if)# **end**

1. Write down the MAC addresses of the FastEthernet interfaces of *Router2*. The MAC addresses of all interfaces are displayed with the following command:

Router2# **show interfaces**

|  |  |
| --- | --- |
|  | **Re-doing the experiment** If you redo all or parts of this experiment, make sure to delete the routing caches at *PC2*, *Router1*, and *Router2*, as well as the neighbor cache at *PC2*. |

1. Now we configure *Router1*. You will repeat the above *Router2* setup steps 1., 2., and 3. for *Router1*. Note that *Router1* has 10.0.2.2 as the default router.
   1. Set the default route at *Router1* to *Router2* (interface on common subnet) with the commands

Router1# **conf term**

Router1(config)# **ip route 0.0.0.0 0.0.0.0 10.0.2.2**

Router1(config-if)# **end**

* 1. To ensure that cached (old) information does not interfere with this experiment, delete the the route cache of each interface. This is done with the following commands:

Router1# **conf term**

Router1(config)# **interface FastEthernet0/0**

Router1(config-if)# **no ip route-cache**

Router1(config-if)# **ip route-cache**

Router1(config)# **interface FastEthernet1/0**

Router1(config-if)# **no ip route-cache**

Router1(config-if)# **ip route-cache**

Router1(config-if)# **end**

* 1. Write down the MAC addresses of the FastEthernet interfaces of *Router1*. The MAC addresses of all interfaces are displayed with the command

Router1# **show interfaces**

1. Start Wireshark to capture traffic between *PC2* (eth0) and the switch.
2. Issue a ping from *PC2* to *PC3*:

PC2$ **ping -c3 10.0.3.33**

1. Take a screenshot of the output of the ping command. Note that the output mentions that there was a route redirect.
2. Display the routing table (netstat -rn) and the routing cache (ip route show cache) and note your observations. Take a screenshot of the output.
3. Use the traffic captured by Wireshark to trace how the ICMP Request messages from *PC2* are forwarded. Relate the packets to the ICMP Redirect message that you observe.   
   **Note:** To trace the ICMP packets you need the MAC addresses of *Router1* and *Router2*.
4. Stop Wireshark and save the (details of the) ICMP and ARP packets.

Lab Questions/Report

1. Include the screenshot of
   1. The routing table before the ping command.
   2. The output of the ping command.
   3. The routing table and routing cache after the ping command.
2. Describe how the routing table and the routing cache at *PC2* changed, when you issued the ping command. In your answer, refer to the screenshots.
3. Using the records from Step 4, provide a table that lists, the MAC and IPv4 addresses of the FastEthernet interfaces of *Router2*.
4. Provide a timeline that shows the order of ICMP messages that are sent between *PC2, Router2,* and *Router1* in Step 7. Include only relevant data from your saved traffic capture to support your explanations (see instructions for timeline in Part 5).
5. The Wireshark traffic capture does not show the traffic between *Router1* and *Router2.*  (If the Ethernet switch is replaced by an Ethernet **hub**, this traffic can be observed by *PC2*).
   1. Use the captured traffic and the timeline to infer packet transmissions between *Router1* and *Router2*.
   2. Insert the packets between *Router1* and *Router2* at plausible locations in the timeline as drawn in 4. above. Clearly indicate these packets in the timeline, e.g., using a different color.

## Part 7. Network Prefixes and Routing

In this part of the lab you study how the network prefixes (netmasks) play a role when hosts determine if a datagram can be directly delivered or if it must be sent to a router.

We use the network topology as shown in Figure 3.3. Table 3.3 describes the IPv4 configuration of the network. In the network topology *PC1*, *PC2*, *PC3* and *PC4* are hosts and *Router2* is an IP router.

Diagram

Description automatically generated

Figure 3.3. Network topology for Part 7.

|  |  |  |  |
| --- | --- | --- | --- |
| Cisco Router | Interface FastEthernet0/0 | Interface FastEthernet1/0 | Default gateway |
| Router2 | 10.0.1.2/24 | 10.0.2.138/24 | none |
| **Linux PC** | **Interface *eth0*** | **Interface *eth1*** | **Default gateway** |
| PC1 | 10.0.1.11/24 | Disabled | 10.0.1.2 |
| PC2 | 10.0.2.22/24 | Disabled | 10.0.2.138 |
| PC3 | 10.0.2.137/29 | Disabled | 10.0.2.138 |
| PC4 | 10.0.2.139/24 | Disabled | 10.0.2.138 |

Table 3.3. IPv4 addresses for Part 7.

### Exercise 7-a. Network setup

The network topology in Figure 3.3 can be constructed from the topology in Figure 3.1, by deleting *Router1* (and as such subnet 10.0.3.0/24), moving *PC3* to subnet 10.0.2.0/24 switch and adding *PC4* to subnet 10.0.2.0/24 switch. The connections of *PC1, PC2, and Router2* are the same as in Part 6. The IP addresses of *PC1, PC2*, *PC3* and *PC4* and the default gateways are as shown in Table 3.3. Note that PC3 has a different “netmask”. Configure *Router2* interface IP addresses as shown in Table 3.3.

1. Connect the topology as shown in Figure 3.3.
2. Verify that the IPv4 addresses of *PC1, PC2, PC3*, *PC4* and *Router2* are as given in Table 3.3.
3. Configure the default gateways of *PC1*, *PC2*, *PC3,* and *PC4* as given in Table 3.3 and delete the default gateway for *Router2*.

**Note:** To change the default route on a device, you first **remove** the existing default route and then add the new entry.

### Exercise 7-b. Exploring the role of prefixes (netmasks) at hosts

Before we run the experiment, delete all neighbor and route caches in the PCs. Commands shown for *PC1*, repeat for the remaining PCs.

1. Delete all entries in the neighbor cache by issuing

PC1$ **sudo ip neigh flush all**

1. Icon

   Description automatically generatedDelete the route cache with the command

PC1$ **sudo ip route flush cache**

1. Run “netstat -rn” to confirm that the default route is correctly configured. Take a screenshot of the output.
2. Start a Wireshark capture on the link from *PC1* (eth0) to the switch, and a Wireshark capture on link for *PC3* (eth0) to the switch. Set a filter on each capture for **ICMP** and **ARP** traffic.
3. Now issue a ping from *PC3* to *PC1.* ***All Success, via router.***

Icon

Description automatically generatedPC3$ ping -c5 **10.0.1.11**

1. ![A picture containing text

   Description automatically generated]()Screenshot and save the output of *PC1’s* console showing the output of the ping command.
2. Icon

   Description automatically generatedStop the Wireshark capture on both links and save the output.
3. Show the content of the neighbor cache and the route cache on both PCs. Screenshot and save.
4. Delete the route cache on *PC3*.
5. Repeat Steps 4-9 for a ping from *PC3* to *PC4*. **All Success, as PC4 in same subnet as PC3’s IP.**
6. Repeat Steps 4-9 for a ping from *PC3* to *PC2*. **1 fail first PING ICMP TO ROUTER, rest all Success, ICMP Redirects and adds route cache entry.**

Lab Questions/Report

1. Explain what you observed for each of set of “pings” between *PC3* and the other 3 PCs. Use the saved data to support your answers. Provide an explanation for the observations you made. Try to illuminate the ping behavior as witnessed by the ICMP and ARP packet exchanges, e.g., if you observe more ICMP Echo Requests than Echo Replies, try to explain the reason.
   1. MORE ICMP Replies because of ICMP redirect replies.
2. If PC3 had no default entry in its routing table, would you have seen the same results? Explain for each of the pings above what would have been different.
   1. Both PC2 & PC1 fails as only PC4’s IP falls in PC3’s eth0 range 10.0.2.137/29.

## Part 8. Stateless Autoconfiguration and Static Routing in IPv6

This part of the lab exposes you to the Linux and Cisco IOS static route configuration for IPv6. In this part, we use the network topology from Figure 3.4. This is the same topology as used in Parts 1-5. We display the IPv6 addresses.

An interaction between IP routers and hosts in IPv6, which does not exist in IPv4, is *stateless address autoconfiguration*. Stateless address autoconfiguration can set IP parameters of a host without a server or manual configuration. It consists of the following steps:

1. Upon startup, hosts create link-local addresses for each enabled IPv6 interface (Lab 2, Part 5)
2. For each link-local address created, the host sends an ICMPv6 Neighbor Solicitation to it:
   1. If someone replies with an ICMPv6 Neighbor Advertisement, the link-local address is in use.
   2. If there is no reply, the address can be used.
3. Next, the host sends an ICMPv6 Router Solicitation to the “all IPv6 routers” multicast group.
4. A router replies to such a message with an ICMPv6 Router Advertisement, which contains the network prefix, the MTU, and other information.
5. The host creates a globally routable IP address, using the network prefix sent by the router, and the Interface ID from the link-local address.

Diagram

Description automatically generated

Figure 3.4. Network topology for Part 8.

Table 3.4. IPv6 Addresses for topology in Figure 3.4.

|  |  |  |
| --- | --- | --- |
| PC | IPv6 address of *eth0* | IPv6address of *eth1* |
| PC1 | autoconfigured | – |
| PC3 | autoconfigured | – |
| **Cisco Router** | **IPv6 address of** ***FastEthernet0/0*** | **IPv6 address of** ***FastEthernet1/0*** |
| Router1 | fd01:2345:6789:2::1/64 | fd01:2345:6789:3::1/64 |
| Router2 | fd01:2345:6789:1::2/64 | fd01:2345:6789:2::2/64 |

### Exercise 8-a. Network setup

Configure the network as shown in Figure 3.4. The IPv6 addresses of the PCs and *Router1* and *Router2* are given in Table 3.4.

1. Connect the Ethernet interfaces of the Linux PCs and the Cisco routers as shown in Figure 3.4.

### Exercise 8-b. Configuring an IPv6 router on Cisco IOS and stateless autoconfiguration

Next you configure *Router1* and *Router2* as IPv6 routers. Once the configuration is complete, you will instantly observe that the two routers start stateless autoconfiguration of *PC1 and PC3*. This will provide both *PC1* and *PC3* with a global IPv6 address. *PC1* will get a default route to *Router2* and *PC3* to *Router1*.

Below is a list of relevant Cisco IOS commands for an IPv6 configuration.

|  |
| --- |
| **IOS mode: privileged EXEC**  **show ipv6 route**  Displays the contents of theIPv6 routing table.  **clear ipv6 route \***  Deletes all IPv6 routing table entries.  **show ipv6 interfaces brief**  Lists the available network interfaces and their IPv6 configuration  **IOS mode: global configuration**  **ipv6 unicast-routing**  Enables IPv6 unicast forwarding.  **no ipv6 unicast-routing**  Disables IPv6 unicast forwarding.  **ipv6 route *fd01:2345:6789:2::/64* *FastEthernet0/0* *fd01:2345:6789:2::2***  Adds a static route to interface FastEthernet0/0 for destination network fd01:2345:6789:2::/64 and next hop router fd01:2345:6789:2::2.  (Putting a “no” in front of the command deletes the routing table entry.)  **IOS mode: interface configuration**  **ipv6 address** ***fd01:2345:6789:2::1/64***  Configures a global IPv6 address fd01:2345:6789:2::1 with a 64-bit long network prefix.  **ipv6 address** ***fd01:2345:6789:2::/64*****eui-64**  Configures a global IPv6 address, with 64-bit long network prefix fd01:2345:6789:2. The interface identifier is created automatically from the MAC address using the EUI-64 convention.  **ipv6 enable**  Enables IPv6 processing on an interface where no IPv6 address has been configured.  **no ipv6 address** ***fd01:2345:6789:2::1/64***  Deletes an IPv6 address.  **no ipv6 address** ***fd01:2345:6789:2::/64*** **eui-64**  Deletes an IPv6 address with EUI-64 convention  **no ipv6 enable**  Removes the routing table entry or disables IPv6 processing. |

1. Start a Wireshark session to capture traffic between *PC3* (eth0) and the switch.
2. *On* *PC1* and *PC3*, take a note of the link-local IPv6 address of *PC1* (eth0) and *PC3* (eth0), using the command (shown here for *PC3*)

PC3$ **ip addr show dev eth0**

Refresh your knowledge how the link-local IPv6 address is created from the MAC address (Lab 2, Part 5). Take note of the interface identifier (the last 64 bits) of the link-local IPv6 address.

Take a screenshot of the output of the command on each PC.

1. Next you configure *Router1*. First check the IPv6 routing table and interface configuration with the commands

Router1# **show ipv6 route**

Router1# **show ipv6 interface brief**

1. Now configure the IPv6 addresses according to Table 3.4, and add a static routing table entry for subnet *fd01:2345:6789:1::/64*. While typing the commands keep an eye on the packets that are captured by Wireshark.

Router1# **configure terminal**

Router1(config)# **interface FastEthernet0/0**

Router1(config-if)# **ipv6 address fd01:2345:6789:2::1/64**

Router1(config-if)# **no shutdown**

Router1(config-if)# **interface FastEthernet1/0**

Router1(config-if)# **ipv6 address fd01:2345:6789:3::1/64**

Router1(config-if)# **no shutdown**

Router1(config-if)# **exit**

Router1(config)# **ipv6 unicast-routing //**after this we get the advertisement with prefix fd01:2345:6789:3::, and immediately PC3’s global ipv6 is setup with ip fd01:2345:6789:3:2485:55ff:fe72:b3d4/64

Router1(config)# **ipv6 route fd01:2345:6789:1::/64 f0/0 fd01:2345:6789:2::2**

Router1(config)# **exit**

While you type the above commands, you will observe ICMPv6 message flows captured by Wireshark. Pay particular attention to the *ICMPv6 Router Advertisement* sent by *Router1*. Identify the information on network prefix and prefix length that is provided in this message.

If you do not see the *ICMPv6 Router Advertisement*, wait a few minutes until *Router1* sends thismessage.

1. On *Router1*, repeat Step 3 and take a screenshot of the output.
2. Repeat Steps 3-5 on *Router2*. Note that on *Router2* the interface for ipv6 route is FastEthernet1/0, you are adding an entry for subnet *fd01:2345:6789:3::/64* and you are going via IP address *fd01:2345:6789:2::1/64* on *Router1*.
3. On *PC3*, display the IPv6 configuration and the IPv6 routing table with the commands

PC3$ **ip addr show dev eth0**

PC3$ **netstat -rn -6 route**

Take a screenshot of the output.

* You see that *PC3* now has an IPv6 address on the network *fd01:2345:6789:3::/64*. This is the autoconfigured IPv6 address. How did *PC3* construct this address?
  + As described above appending ntw prefix from adv and link-layer host addr.
* The IPv6 routing table has an entry for the default route. Determine the configured IPv6 address of the default gateway. Explain why this IPv6 address is chosen?
  + ::/0 fe80::ce01:fff:fe30:10 UGDAe 1024 1 0 eth0 // this is the eth0(f0/0) of the router 1 which is directly connected to PC3 hence it’s the default, as no other router is present.

1. Icon

   Description automatically generatedRepeat Steps 7 for *PC1*.
2. Now verify that the established routes work by running a ping and a traceroute from PC3 to PC1. Take a screenshot of the output.

PC3$ **ping6 -c2** *IPv6\_address\_of\_PC1*

Icon

Description automatically generatedPC3$ **traceroute6** *IPv6\_address\_of\_PC1*

In the Wireshark output display the details of the *ICMPv6 Router Advertisement* messages. Take a screenshot of the output. **IT SHOWS THE NTW Prefix: fd01:2345:6789:3::**

1. Now on *PC1*, configure on interface eth0, a **unicast local address** on subnet fd01:2345:6789:1::/64 with *interface ID* 11

PC1$ **sudo ip addr add fd01:2345:6789:1::11/64 dev eth0**

1. On *PC1*, display the IPv6 configuration and the IPv6 routing table with the commands

PC1$ **ip addr show dev eth0**

PC1$ **netstat -rn -6 route**

Icon

Description automatically generatedPC1$ **ip neigh show dev eth0**

Take a screenshot of the output.

You see that *PC1* now has an IPv6 address on the network *fd01:2345:6789:1::11/64*. This is the configured IPv6 address.

1. Icon

   Description automatically generatedNow ping from *PC1* to *PC3* and check the neigh cache of *PC1*. Take a screenshot of the output.

PC1$ **ping6 -c2** *IPv6\_address\_of\_PC3*

PC1$ **ip neigh show dev eth0**Icon

Description automatically generated

1. Now flush the neighbor cache for *PC1* and repeat step 12. Take a screenshot of the output.

PC1$ **ping6 -c2** *IPv6\_address\_of\_PC3*

PC1$ **ip neigh show dev eth0**![A picture containing text

Description automatically generated]()

1. Stop Wireshark and save the output

|  |  |
| --- | --- |
|  | **Where is the Router Solicitation?**  In the above experiment, the *Router Advertisement* was sent when the Cisco Router was configured as an IPv6 router. The *Router Advertisement* was not sent in response to a Router Solicitation.  When *PC3* was booted up and the network interface was configured, it sent a *Router Solicitation*. At that time, no router was available.  If you want to see a *Router Solicitation/Router Advertisement* sequence, disable and then enable the eth0 interface of *PC3* with the commands.  PC3$ **sudo ip link set eth0 down**  PC3$ **sudo ip link set eth0 up**  As soon as the interface is enabled, PC3 sends a *Router Solicitation*, which is replied to by *Router1* with a *Router Advertisement*. |

Lab Questions/Report

1. Provide the screen capture from Step 3 (before the *Router Advertisement* was received) and Step 5 (after the *Router Advertisement* was received).
2. Include the screenshot of the routing table of *Router1* from Step 5.
3. From the saved Wireshark data, include one *Router Advertisement* message, which shows the IPv6 source and destination addresses, as well as all fields of the *ICMP6* message.
4. Include the screenshots of the outputs of ping*6* and *traceroute6* from Step 9.
5. Explain how PC3 constructed its global unique unicast address.
   1. As written above the host creates a globally routable IP address, using the network prefix sent by the router, and the Interface ID from the link-local address.
6. Explain how PC3 determines the IPv6 address of the default gateway.
   1. In netstat the entry**: ::/0 fe80::ce01:fff:fe30:10 UGDAe 1024 2 0 eth0**, Tells that for destination all other destination(ie. Default gateway) the next is the MAC address of the Router2(fe80::ce01:fff:fe30:10).
7. On the ping from PC1 to PC3 after configuring PC1’s address what did you observe as the source address of PC1’s request packets? What is the destination of PC1s packets? Does PC1 send a neighbor solicitation? Does the router send one? Explain the behavior that you observe.
   1. Source: fd01:2345:6789:1::11 PC1’s unique local address of
   2. Destination: fd01:2345:6789:3:b093:2cff:fef6:f3d3 PC3’s global ipv6 address
   3. No PC1 does not sends a neigh solicitation.
   4. Yes router sends neigh solicitation, for PC2’s unique local address fd01:2345:6789:1:11, which the PC2 replies with neigh advertisement.

THIS HAPPENS BECAUSE ROUTERS RECEIVES REQUEST FROM UNIQUE LOCAL ADDRESS WHICH ROUTER VERIFIES.

1. After flushing the neighbor cache of PC1, and you repeated the ping, did you see a difference in the Wireshark output? Explain what you see.
   1. YES PC1 sends neigh solicitation for fe80::ce02:21ff:fe93:0 ie. For Router 2
      1. One with src fe80:58de:cbff:fe8c:3c4b which is link local address.
      2. Second with src fd01:2345:6789:1::11 which is unique local address.

# Appendix A: Cisco IOS Command Line Interface

The command line interface of IOS has a rich syntax. There are hundreds of configuration commands, and some commands have numerous options. Different from a Linux Shell, the command line interface of IOS runs in different modes, and each command requires a certain mode. The Internet Lab features only the most common command modes and, for each command mode, uses only a small subset of available commands. The command modes used in the Internet Lab are the *user EXEC mode*, the *privileged EXEC mode*, the *global configuration mode*, the *interface configuration mode*, and the *router configuration mode.*

Each command mode has a different prompt, and a user can derive the current command mode from the command prompt. The user EXEC Mode is indicated by an angle bracket (*>*), the privileged EXEC mode by the pound sign (#), and the configuration modes are indicated by an abbreviation of the configuration mode, followed by the pound sign, for example, *(config)#, (config-if)#,* and *(config-router)#.* Typing a question mark (*?*) in any command mode generates a list of all available commands in the current mode.

Table 3.5 presents a summary of the command modes. Figure 3.6 illustrates the available transitions between different command modes, and which commands need to be issued. For example, changing from the privileged mode to the global configuration mode is done with the command *configure terminal*. Typing *exit*in this mode returns to the privileged mode. As shown in Figure 3.6, it is not feasible to switch arbitrarily from one command mode to another. For example, the global configuration mode cannot be entered from the user EXEC mode.

|  |  |  |
| --- | --- | --- |
| IOS command mode | Role of command mode | Command prompt |
| User EXEC mode | * Limited command set, e.g., ping, telnet, traceroute * No change of system parameters | Router1 > |
| Privileged EXEC mode | * Manage configuration files examine state of router * Access control with password (enable secret) | Router1# |
| Global configuration mode | * Change system wide configuration parameters | Router1(config)# |
| Interface configuration mode | * Modify configuration of a specific interface | Router1(config-if)# |
| Router configuration mode | * Modify the configuration of a specific routing protocol | Router1(config-router)# |

Table 3.5. Cisco IOS Command Modes.



Figure 3.6. Cisco IOS Command Modes.

### User EXEC Mode

The user EXEC mode is entered when the router is accessed via a serial connection or when accessing the router via *telnet*. [[2]](#footnote-2) The command prompt of the user EXEC mode is

Router1>

where *Router1* is the name that is assigned to the router. The user EXEC mode only offers a small set of commands, such as ping*, telnet*, and *traceroute*. Configuration parameters cannot be read or modified in this mode. Typing

Router1>**exit**

logs the user off.

### Privileged EXEC Mode

To change or view configuration information of a Cisco router, a user must enter a system administrator mode. In IOS, the system administrator mode is called the *privileged EXEC mode*. In the privileged EXEC mode, a user has rights similar to the root account on a Linux system. The privileged EXEC mode is used to read configuration files, reboot the router, and set operating parameters. To modify the configuration of a router, a user must proceed from the privileged EXEC mode to the global configuration mode, and, from there, to other configuration modes.

Entering the privileged EXEC mode requires to type a password, called the *enable secret*. The privileged EXEC mode is entered from the user EXEC mode by typing the command

Router1>**enable**  
Password : *<enable secret>*

Typing the correct password displays the following command prompt:

Router1#

To change the command mode back to the user EXEC mode, the user types

Router1#**disable**

Typing ‘exit’ logs the user off.

### Global Configuration Mode

The global configuration mode is used to modify system wide configuration parameters, such as routing algorithms and routing tables. The global configuration mode can only be entered from the privileged EXEC mode. This is done by typing

Router1#**configure terminal**

No additional password is required to enter this mode. The argument *terminal* tells the router that the configuration commands will be entered from a terminal. The alternatives are to issue configuration commands from a configuration file or from a remote machine via a file transfer. The command prompt in the global configuration mode is

Router1(config)#

Global configuration commands include commands that enable or disable IP forwarding and that set static routing table entries. For example, the command

Router1(config)#**ip routing**

enables IP forwarding on the router, and the command

Router1(config)#**ip route** *20.0.1.0/24 10.1.1.1*

adds a network route for destination address 20.0.1.0/24 via gateway 10.1.1.1 to the routing table. Typing *CTRL-z* as in

Router1(config)#**CTRL-z**

changes from the global configuration to the privileged EXEC mode.

### Interface Configuration Mode

To modify the configuration parameters of a specific interface, for example, the IP address, a user must enter the interface configuration mode. The interface configuration mode, which can only be entered from the global configuration mode, for a network interface is entered by typing the keyword *interface* followed by the interface name.

In IOS, each network interface is associated with a name, which specifies an interface type, a slot number, and a port number. Examples of interface types that are used in the Internet Lab are serial WAN interface (*Serial*), 10 Mbps Ethernet (*Ethernet*), and 100 Mbps Ethernet (*FastEthernet*), or 1 Gbps Ethernet (*GigabitEthernet*). The slot number indicates the slot into which the interface card is inserted. The port number identifies a port on the interface card. On some routers the interface name FastEthernet0/0 identifies a 100 Mbps Ethernet adapter port 0 on a card in slot 0 of the router. FastEthernet0/1 identifies port 1 on the same card and FastEthernet1/0 is port 0 of card in slot 1 of the router. On routers which have a fixed number of interfaces and which do not have a slotted chassis, the slot number may be omitted, e.g., *Ethernet0*, *Ethernet1*. IOS assigns interface names automatically without intervention by a user. The privileged EXEC commands *show protocols* or *show interfaces* lists the names of all interfaces on a router.

|  |  |
| --- | --- |
|  | **Naming conventions of network interfaces**  The naming convention for network interfaces of Cisco routers in the lab manual uses *Ethernet0, Ethernet1, …* for the Ethernet interfaces, and *Serial0, Serial1, …* for the serial WAN interfaces. The names of the interfaces of your Cisco routers is almost certainly different, e.g., FastEthernet0/0 or GigabitEthernet0/0/0. You need to replace the actual names of the interfaces with the one given in the lab manual. Refer to Lab 1, where you have determined the names of the interfaces. |

The interface configuration mode for the network interface on port 0 of a 100 Mbps FastEthernet in slot 0 of the router is entered with the command

Router1(config)#**interface FastEthernet0/0**

The command prompt of the interface configuration mode is

Router1(config-if)#

To return to the global configuration mode one types

Router1(config-if)#**exit**

When a global configuration command is typed in the interface configuration mode, then IOS changes to the global configuration command.

### Router Configuration Mode

The router configuration mode is used to configure the parameters for a specific routing protocol. When entering the router configuration mode, the name of the routing protocol must be specified as an argument. IOS supports numerous routing protocols, includingthe *Routing Information Protocol* (*RIP*), *Open Shortest Path First* *(OSPF),* and *Border Gateway Protocol (BGP)*, and many more. The command to enter the routing router configuration mode for the routing protocol RIP from the global configuration mode is

Router1(config)#**router** *rip*

The command prompt for the router configuration protocol is

Router1(config-router)#

Typing

Router1(config-if)#**exit**

changes to the global configuration mode.

### IOS Commands for Interface Configuration

We next discuss the IP configuration of a network interface in IOS. Consider a router with 100 Mbps Ethernet (*FastEthernet*) interface adapter cards with names *FastEthernet0/0* and *FastEtherne1/0*. The following sequence of IOS commands configures port 0 with IP address 10.0.2.1/24 and port 1 with IP address 10.0.3.1/24. In addition, the commands enable IP forwarding on the router.

Router1> enable  
Password: *<enable secret>*   
Router1# **configure terminal**   
Router1(config)# **no ip routing**  
Router1(config)# **ip routing**  
Router1(config)# **interface** **FastEthernet0/0**  
Router1(config-if)# **no shutdown**  
Router1(config-if)# **ip address** **10.0.2.1 255.255.255.0**  
Router1(config-if)# **interface** **FastEthernet1/0**  
Router1(config-if)# **no shutdown**  
Router1(config-if)# **ip address 10.0.3.1 255.255.255.0**  
Router1(config-if)# **end**

The first two commands change to the privileged EXEC mode and, from there, to the global configuration mode. The command ‘no ip routing’, which is the command to disable IP forwarding, is used to reset the contents of the routing table. The next command***,*** ip routing, enables IP forwarding on the router. Then, the interface configuration mode is entered for interface *FastEthernet0/0*. The command *no shutdown*enables the interface, and the command ‘ip address 10.0.2.1 255.255.255.0’sets the IP address to 10.0.2.1/24. The commands to configure the second interface are similar. Note that the interface configuration mode for interface Fast*Ethernet1/0* is entered without returning to the global configuration mode. The last command (*end*) returns to the privileged EXEC mode.

The following list summarizes the IOS commands for enabling IP forwarding and for configuring IP addresses.

|  |
| --- |
| **IOS mode: global configuration**  **ip routing**  Enables IP forwarding.  **no ip routing**  Disables IP forwarding. This command also deletes the content of the routing table.  **IOS: interface configuration**  **shutdown**  Disables network interface.  **no shutdown**  Enables a network interface.  **ip address** 10.0.1.10 255.255.255.0  Sets the IP address to 10.0.1.10/24. |

The names of the interface depend on the types of routers used and installed network cards. On routers with a slotted chassis, the names of the interfaces additionally depend on the slot location of the interface card. The interface names of a router are displayed with the privileged EXEC commands show interfaces or show protocols:

|  |  |
| --- | --- |
| Model | Interface Names |
| Cisco 2514 | Ethernet0, Ethernet1,  Serial0, Serial1 |
| Cisco 2811 | FastEthernet0/0, FastEthernet0/1,  Serial1/0, Serial1/1 |
| Cisco 3640 | FastEthernet0/0, FastEthernet1/0, Serial2/0, Serial2/1, Serial2/2, Serial2/3 |
| Cisco 4321 | GigabitEthernet0/0/0, GigabitEthernet0/0/1,… , |

### IOS Commands to Display the Configuration and Other Information

IOS maintains two configuration files, which are called *startup configuration* and *running configuration*. The configuration files consists of a sequence of IOS commands. The startup configuration is kept in a file on Nonvolatile RAM (NVRAM), and contains the IOS commands that are executed when IOS is booted. To reboot IOS, one can turn the power switch off and then on again. Alternatively, a reboot of IOS is enforced when typing the privileged EXEC command *reload*. When IOS is booted up, the running configuration is set to the startup configuration. The running configuration stores the currently active configuration of the router, and issuing IOS configuration commands modifies the running configuration. The running configuration is kept in RAM and is lost when the router is powered off or when IOS is rebooted. To make changes to the running configuration permanent, the command *copy running-config starting-config* can be used to save the running configuration as the startup configuration.

The commands that display the configuration files are entered from the privileged EXEC mode, and are as given below.

|  |
| --- |
| **IOS mode: privileged EXEC**  **write term**  **show running-config**  Displays the current configuration of the router. Both commands are identical.  **show config**  **show startup-config**  Displays the startup configuration of the router. Both commands are identical.  **reload**  Forces a reboot of IOS. This command discards the running configuration and reloads the startup configuration.  **copy running-config starting-config**  Saves the current configuration as the startup configuration. The new startup configuration will be used the next time IOS is rebooted. |

In addition to configuration files, various commands are available to display information about the router. Below we list some frequently used commands.

|  |
| --- |
| **IOS mode: privileged EXEC**  **show version**  Displays the version of IOS.  **show protocols**  Displays the IP configuration of the interfaces of the router. Also, indicates if IP forwarding is enabled or disabled.  **show ip route**  Displays the routing table.  **show ip cache**  Displays the routing cache.  **show interfaces**  **show interfaces *FastEthernet0/0***  Displays information about all network interfaces. When an interface name is given as argument, here, *FastEthernet0/0*, information is displayed only for the specified interface.  **show ip arp**  Displays the contents of the ARP cache. |

The show protocols command gives a concise overview of the IP configuration of the interfaces of the router.

router1#**show protocols**

Global values:

Internet Protocol routing is enabled

FastEthernet0/0 is up, line protocol is up

Internet address is 10.0.2.1/24

FastEthernet1/0 is up, line protocol is up

Internet address is 10.0.3.1/24

Serial0 is administratively down, line protocol is down

Serial1 is administratively down, line protocol is down

From this output, we can tell that IP forwarding is enabled on the router, that the Ethernet interfaces *FastEthernet0/0* and *FastEthernet1/0* are configured with IP addresses, and that the serial interfaces are currently not used. More extensive information about the interfaces can be displayed with the  
show interfaces command.

### Navigating the IOS Command Line Interface

IOS provides a few features that make typing commands more convenient. We already mentioned that typing a question mark (*?*) in a given command mode generates a list of all available commands in the current command mode. For example,

Router1(config-if)# **?**

lists the available commands in the interface configuration mode. Since IOS commands can only be executed in a certain command mode, this command helps to determine if a command can be executed in the current mode. The question mark can also be used to determine the list of available options of a command. For example,

Router1# **configure ?**

lists all options that are available for the command *configure*.

**Abbreviations:** When typing commands or the names of network interfaces, it is sufficient to type just enough characters so that IOS can interpret the input without ambiguity. The following shows how some abbreviations are interpreted.

| Abbreviation | Command |
| --- | --- |
| conf | configure |
| w t | write terminal |
| int f0/0 | interface FastEthernet0/0 |

When the Tab key (*<Tab>*) is typed in the command line interface, IOS attempts to complete the command. Command completion is successful only if enough characters are typed so that the prefix can be completed without ambiguity. Here are some examples of command line completions.

| Abbreviation | Command Completion |
| --- | --- |
| conf <Tab> | configure |
| conf <Tab> t <Tab> | configure terminal |

An interesting feature of IOS, is that putting a “*no*” in front of same command often creates a valid command. For example, if a certain command enables a feature of a router than adding a “*no*” in front of that command disables the same feature. Sometimes it is the other way around, that is, the command to enable a feature uses the command to disable the feature preceded by a “*no*”. The following are a set of examples.

|  |  |
| --- | --- |
| Feature | Command |
| Enable IP forwarding | ip routing |
| Disable IP forwarding | no ip routing |
| Add a routing table entry | ip route 10.0.2.0 255.255.255.0 10.0.3.1 |
| Delete a routing table entry | no ip route 10.0.2.0 255.255.255.0 10.0.3.1 |
| Disable a network interface | shutdown |
| Enable a network interface | no shutdown |

|  |  |
| --- | --- |
|  | **Getting rid of the pesky CDP and LOOP packets** When capturing traffic of a Cisco router, there can be a large number of packets of type CDP and LOOP. These are Cisco proprietary protocols: Cisco Discovery Protocol (CDP), which is used for routers to discover each other, and the LOOP protocol, which detects link layer loops. You can disable these protocols. To stop CDP messages, enter the global configuration mode and type  Router1(config)# **no cdp run**  To stop LOOP messages, you have to go to the interface configuration mode for each interface and type  Router1(if-config)# **no keepalive** |

## APPENDIX B. A TIP TO SPEED UP the SETUP of EXPERIMENTS

To configure the cisco routers and PCs, you are required to enter the commands manually in the console window. By now you will probably have realized that many of the commands are repeated. Also, you probably noticed that GNS3 resets PC and Cisco router configurations when you stop a GNS3 project. Saving the commands in a file and then using copy and paste of the command set will save you a lot of time.

Below we show you how you can save configuration commands in a Microsoft Excel Spread Sheet, for use at a later time.

1. Open an Excel spread sheet. Then, add a command in each row as shown in Figure A.1. The columns represent the different devices you are saving commands for.

Graphical user interface, application, table, Excel

Description automatically generated

Figure B.1

1. Select a command set from a column and copy the block of commands by pressing the shortcut key “Ctrl + C” for Windows and “Command + C” for Mac. The command set will be highlighted as shown in Figure B.2.

Graphical user interface, application, table, Excel

Description automatically generated

Figure B.2

1. Start a Cisco router R1 and a PC1 in GNS3.
2. Open console windows for each of R1 and PC1.
3. Now you paste the commands by right click on the console windows for Windows users. For Mac users, paste the commands with the shortcut key “Command + V”. You will see that the commands are executing as shown in Figure B.3 for R1.

Text

Description automatically generated

Figure B.3

1. Repeat the above step to configure PC1.
2. Please make sure that the commands do not contain any typos.
3. Save the excel spread sheet. You can use for the next experiment if the configuration is similar. If not you can edit the saved file, e.g., IP address may have changed. And repeat the copy-paste routing to configure your devices.

## APPENDIX C. Saving a Router Configuration in GNS3 Project

For routers, you can use the command ***# copy running-config startup-config***to save the configurations, and then save your GNS3 project. This will ensure your router configurations don't get erased when you restart GNS3 and open that project.

1. Note: you can save the list of router configuration commands in an excel spreadsheet for faster router setup. See Appendix B and C for instructions.. [↑](#footnote-ref-1)
2. Entering the user EXEC mode over a serial connection may require a login password and entering this mode with *telnet* always requires a login password. [↑](#footnote-ref-2)