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: October 2021

Table of Content

Study Material for Lab 3

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

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

2. **ip route:** Read about the ip route command at
   http://linux-ip.net/html/tools-ip-route.html

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

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

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

Prelab 3

1. How does a router determine whether a datagram to particular host can be directly delivered through one of its interfaces?
2. List four different methods for configuring (adding/removing/changing) routing table entries. 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. Can a Linux system be configured to run as both a host and a router? Justify your answer.
8. Write the Cisco IOS command that sets the next hop of the default route to *10.0.1.3.*
9. 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*.
10. 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.
11. Which systems generate *ICMP Redirect* messages? Routers, hosts, or both?

Lab 3 – Multiple Subnets – Static Routing

In this lab you work with four different network topologies. The topology and network configuration for Parts 1-4 is shown in Figure 3.1. *PC1* and *PC3* are used as hosts, and *PC2* and *Router1* are set up as IP routers. The PCs and the Cisco router are connected by Ethernet switches. In Lab 3, all routing table entries are manually configured, a procedure known as *static routing*.

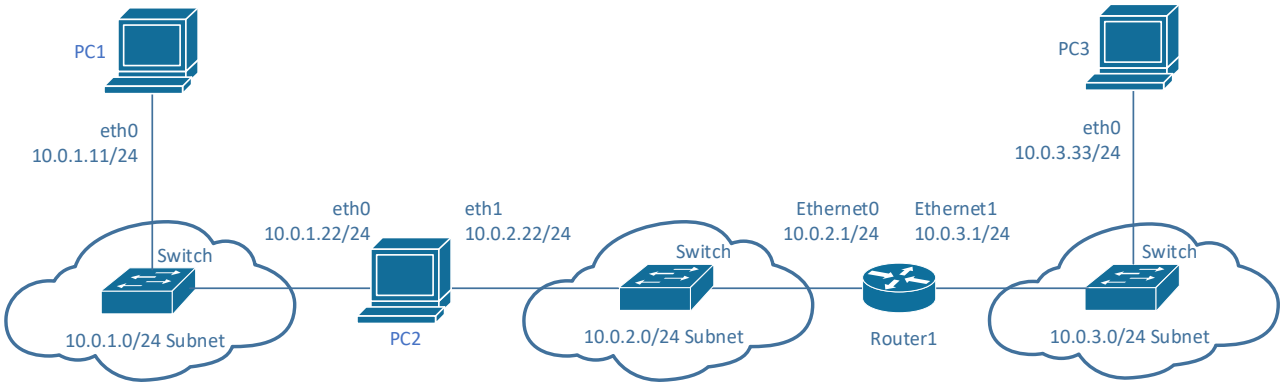


Figure 3.1. Network topology for Parts 1-5.

PC	IPv4 address of *eth0*	IPv4 address of *eth1*
PC1	10.0.1.11/24	–
PC2	10.0.1.22/24	10.0.2.22/24
PC3	10.0.3.33/24	–
Cisco Router	**IPv4 address of *Ethernet0***	**IPv4 address of *Ethernet1***
Router1	10.0.2.1/24	10.0.3.1/24

Table 3.1. IPv4 configuration for Parts 1 – 4.

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`
Displays the contents of theIPv4 routing table.

`show ipv6 route`
Displays the contents of the IPv6 routing table.

`clear ip route *`
`clear ipv6 route *`
Deletes all IPv4/IPv6 routing table entries.

`show ip route`
Lists the routing table entries that ma

`show ip route` ***`10.0.3.33`***
Lists the routing table entry that match ***`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

Step 1: Connect the Ethernet interfaces of the Linux PCs and the Cisco router as shown in Figure 3.1.

Step 2: On PC1, display the IPv4 and IPv6 routing tables before any IP address is configured using the *netstat* command. Take snapshots of the output.

Step 3: 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
```

Step 4: Display the IPv4 routing tables on PC1 and PC2 using the *netstat* command. Take screenshots of the output.

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

a. Relate the routing table entries to the configured IP addresses.

b. Explain the entries in the column with header *Gateway*.

Step 5: Start a traffic capture between PC1 (eth0) and the Ethernet switch.

Step 6: In a console terminal of PC1, issue *ping* commands to PC2, Router1 and PC3. Take a screenshot of the output of the *ping* commands.

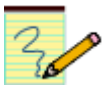
```
PC1$ ping -c3 10.0.1.22
PC1$ ping -c3 10.0.2.22
PC1$ ping -c3 10.0.2.1
PC1$ ping -c3 10.0.3.1
PC1$ ping -c3 10.0.3.33
```

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

a. Use the routing table of PC1 to explain the outcome of the ping commands.

b. Which *ping* commands generated network traffic, e.g., ARP and ICMP packets?

Step 7: 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.

Step 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.

Step 2: ***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`, `traceroute`. 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.

Step 3: ***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# ?
```

Step 4: ***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.

Step 5: ***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 *Ethernet0*, you type

```
Router1(config)# interface Ethernet0
Router1(config-if)#
```

If you want to configure another network interface, say Ethernet1, you can jump directly to the interface configuration mode of that interface by typing.

```
Router1(config-if)# interface Ethernet1
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 Ethernet0. The names of the interface of your router are almost certainly different. If you are unsure about the interfaces and your name use the privileged EXEC command

```
Router1# show protocols
```

Step 6: **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.

Step 7: 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.

Again, the names of the Ethernet interfaces on your routers are (almost certainly) different than given here.

Step 1: Open a console window on Router1.

Step 2: Configure Router1 with the IP addresses given in Table 3.1.

```
Router1# configure terminal
Router1(config)# no ip routing
Router1(config)# ip routing
Router1(config)# interface Ethernet0
Router1(config-if)# ip address 10.0.2.1 255.255.255.0
Router1(config-if)# no shutdown
Router1(config-if)# interface Ethernet1
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.

Step 3: 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
```

Step 4: Analyze the output 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 10.0.2.22`
Adds a static routing table entry to subnet *`10.0.1.0/24`* with next hop router *`10.0.2.22`*. 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 Ethernet0`
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., Ethernet0).

`ip route 0.0.0.0 0.0.0.0 10.0.2.22`
Adds router 10.0.2.22 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 10.0.2.22`
`no ip route 0.0.0.0 0.0.0.0 10.0.2.22`
Adding a "no" before an '`ip route`' command deletes a route table entry.

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

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

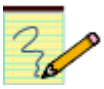
As in Linux, whenever an IP address is configured for a network interface, a routing table entry for the directly connected network is added automatically.

Step 2: 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.

On Router1, add a routing table entry for network 10.0.1.0/24 with 10.0.2.22 as next hop router. Note that you must be in the global configuration mode.

Step 3: Return to the privileged EXEC mode. Display the routing table again with `show ip route` and take a screen snapshot of the output.

After the configuration of Router1, PC1 and PC3 can still not talk to each other. There are two pieces missing: (1) PC2 must be set up as an IPv4 router, and (2) the routing tables of all PCs must be configured. This will be done in the next part of the lab.



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.

Part 3. Configuring a Linux PC as a Router

Any Linux PC with at two or more network interfaces can be set up as an IP router. Configuring a Linux PC as an IP router involves two steps:

1) Modifying the configuration of Linux so that IP forwarding is enabled, and
2) Configuring the routing table.

In this part, you will configure PC2 as an IP router.

To configure any IP enabled device as an IP router, "IP forwarding" must be enabled on that router. Often, IP forwarding is disabled by default. With IP forwarding disabled, when the device receives an IP packet with a destination address that is not an IP address of the local interface, the IP packet is dropped. When IP forwarding is enabled, the device makes a lookup in its routing table and tries to forward the packet to another device.

Exercise 3-a. Configuring a Linux PC as a router

On a Linux PC, the state of IP forwarding is maintained in a kernel variable `net.ipv4.ip_forward` for IPv4 and in `net.ipv6.conf.all.forwarding` for IPv6. As seen in Lab 1 (Part 8), kernel variables can be accessed and modified either with the `sysctl` command or via the /proc file system.

`sysctl net.ipv4.ip_forward`
Displays the state of IP forwarding for IPv4. IP forwarding is enabled when the value is "1" and disabled when the value is "0".

`sudo sysctl -w net.ipv4.ip_forward=1`
`sudo sysctl -w net.ipv4.ip_forward=0`
Enables/disables IPv4 forwarding.

`sysctl net.ipv6.conf.all.forwarding`
Displays the state of IPv6 forwarding. The system is an IPv6 router when the value is "1".

`sudo sysctl -w net.ipv6.conf.all.forwarding=1`
`sudo sysctl -w net.ipv6.conf.all.forwarding=0`
Enables/disables IPv6 forwarding.

Step 1: On PC2, check whether IP forwarding for IPv4 and IPv6 is enabled or disabled. Also, list the routing tables of PC2.

Step 2: Enable IP forwarding for IPv4 on PC2 using the command shown above. This turns PC2 into a router as an IP router.
Did the routing tables of PC2 change?

Step 3: Repeat Step 1 to confirm that IPv4 forwarding is enabled on PC2.

Making changes permanent
The sysctl commands have an immediate effect, however, changes are not permanent and are lost when the system is rebooted. Modifying the IP forwarding state permanently requires changes to the configuration file /etc/sysctl.conf.

To permanently configure a Linux PC as an IPv4 router, the /etc/sysctl.conf file should have a line
`net.ipv4.ip_forward=1`

Changes to /etc/sysctl.conf take effect the next time the system rebooted. To enforce that changes to the file have an immediate impact use the command

```
sudo sysctl -p /etc/sysctl.conf
```

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

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.

The next step is to set up the routing tables of the Linux PCs. *PC1* and *PC3* are hosts, and *PC2* is an IPv4 router. 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 10.0.1.22`

Adds a routing table entry for the subnet 10.0.2.0/24 with next hop 10.0.1.12. IP packets with a destination address in this subnet will be forwarded to IP address 10.0.1.12. 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 10.0.1.22`

Adds the router with IP address 10.0.1.12 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.

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.

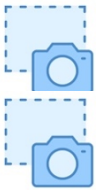
```
sudo ip link set dev eth0 down
sudo ip link set dev eth0 up
```

With these commands, the routing table entries for the configured IP addresses will remain.

The next steps ask you to configure the routing tables of *PC1*, *PC2*, and *PC3*.

Step 1: Configure a default route on *PC1* with the configuration command

```
PC1$ sudo ip route add default via 10.0.1.22
```



Step 2: Display the routing table of PC1 with `netstat -rn' and take a screenshot of the routing table.

Step 3: 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 it, and why do they now work?
- Some of the *ping* commands that previously failed, still work. Which ones are they it, and why do they fail?

Step 4: Start traffic capture with Wireshark for

- *PC1* (for *eth0*),

- *PC2* (for *eth1*), and
- *PC3* (for *eth0)*.

Note to ECE461 students (Oct. 2021)

To avoid seeing packets send by a running DNS process, set a display filter to
`icmp and ip.addr==10.0.0.0/8`

Step 5: 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.

Step 6: Next configure stating routing table entries for PC2. We have a total of three subnets: 10.0.1.0/24, 10.0.2.0/24, 10.0.3.0/24. Since *PC2* already has routing table entries for 10.0.1.0/24 and 10.0.2.0/24, we only need to add a routing table entry for 10.0.3.0/24 with next hop 10.0.2.1. The command is

```
PC2$ sudo ip route add 10.0.3.0/24 via 10.0.2.1
```

Once the route is added, the traffic that is captured by *Wireshark* for *PC1* (*eth0*) and *PC2* (*eth1*) is different. Also, the output at *PC1* changes. Is the *ping* (from *PC1* to *PC3*) successful? Describe your observations.

Step 7: 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 *PC3*. 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

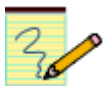
```
PC3$ sudo ip route add default via 10.0.3.1
```

Once the route is added, observe how the captured traffic by the two instances of *Wireshark* changes. Describe your observation. Is the *ping* now successful?

Step 8: Display the routing table of *PC2* with `'netstat -nr'` and take a screenshot of the result.

Step 9: Stop the instances of *Wireshark*.

Lab Questions/Report:



1. Include the screen snapshot of the routing table of PC1 from Step 1. Explain how you identify the default route.
2. Include the screen snapshot 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. Describe your observations in Step 7.
6. Include the routing table of *PC2* from Step 8.

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 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) or `sysctl net.ipv4.ip_forward` (in Linux) to display the forwarding status Display routing tables with `show ip route` (in IOS) or `netstat -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

Step 1: Continue with the network configuration of Part 3.

Step 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 10.0.1.22
PC1$ ping -c3 10.0.2.22
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

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

Step 1: Start a *Wireshark* session on *PC1* (*eth0*).

Note to ECE461 students (Oct. 2021)

You may want to set a display filter to:
"`UDP or ICMP`"

Step 2: Execute a `traceroute` command from *PC1* to *PC3*, and save the output.

```
PC1$ traceroute 10.0.3.33
```

Step 3: Observe how *Wireshark* gathers information on the route. Pay attention to the TTL fields in the UDP packets that are sent by *PC1*.

Step 4: Stop the traffic capture of *Wireshark* and save the captured traffic as a plain text file. (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 *traceroute*.

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*.

Step 1: Add the following routes to the routing table of *PC2*:

```
PC2$ sudo ip route add 10.0.0.0/16 via 10.0.1.71
PC2$ sudo ip route add 10.0.3.9 via 10.0.1.81
PC2$ 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.

Step 2: Take a screen snapshot of the IPv4 routing table at *PC2* (using `netstat -rn`)

Step 3: Use the commands `ip route to match` and to determine how many matches exist in the routing table of PC2 for the following IP addresses:

10.0.3.9
10.0.3.14
10.0.4.1
10.1.4.18

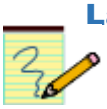
Use the `ip route get` command to determine which match is selected.

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.

Step 4: Take a screen snapshots of the output of the commands in Step 3.

Step 5: Delete the routing table entries that were added in Step 1.

Lab Question/Report



1. Include the saved screenshots and explain how *PC2* 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 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 and 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 *Ethernet0,* the commands are

```
Router1# configure terminal
Router1(config)# interface Ethernet0
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 *Ethernet1* are

```
Router1# configure terminal
Router1(config)# interface Ethernet0
Router1(config-if)# no shutdown
Router1(config)# end
```

Exercise 5-a. Observing Proxy ARP

Step 2: The network configuration is identical to that used in Part 4.

Step 3: Erase the neighbor caches at *PC1*, *PC2*, and *PC3* (`sudo ip neigh flush all`).

Step 4: On *PC3*, delete the default route (see Part 3 of this lab).

Step 5: On *Router1*, enable Proxy ARP on the *Ethernet* 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 Ethernet0
Router1(config-if)# ip proxy-arp
Router1(config-if)# exit
Router1(config)# interface Ethernet1
Router1(config-if)# ip proxy-arp
Router1(config-if)# end
```

Step 6: 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*.

Step 7: For the observations of this experiment, we need to run multiple instances of *Wireshark*. Start *Wireshark* traffic captures for

- *PC3 (eth0),*
- *PC2 (eth1),* and
- *PC1 (eth0).*

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*) and `ip addr` (on *PC1*, *PC2*, *PC3*).

An alternative is to send ping messages to the neighbors in the topology, e.g, *PC1↔PC2, PC2↔Router1, Router1↔PC3*) and check the MAC addresses in the packets captured by *Wireshark*.

Step 8: 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:

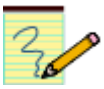
- Note that *PC3* sends an ARP Request for 10.0.1.11, which is replied to by *Router1*.
- What is the MAC address that *Router1* sends to *PC3*? Compare this MAC address to the addresses recorded in Step 7.
- Go to the *Wireshark* capture of *PC3* (*eth0*). Look for an *ICMP Echo Request* sent by *PC3*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

Step 9: Display the neighbor cache of *PC3* and take a screenshot.

Step 10: Stop *Wireshark* on *PC1*, *PC2*, and *PC3*. Save the traffic that is captured by *Wireshark*.

Step 11: 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*?

Step 12: Reset the IP address of *PC3* (*eth0*) to its original value of *10.0.3.33/24* and re-enable Proxy ARP on *Router1*.



Lab Question/Report

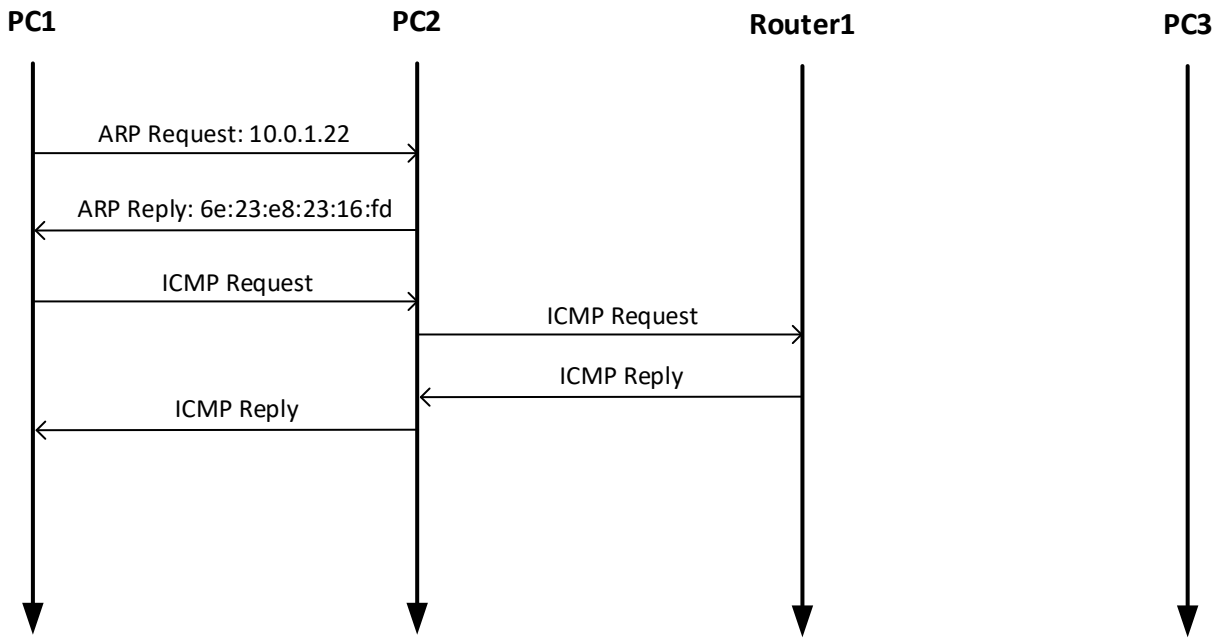
1. Using the records from Step 7, provide a table that lists, for each interface used in the network from Figure 3.1, the MAC and IPv4 addresses.

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”.


PC1
PC2
Router1
PC3
ARP Request: 10.0.1.22
ARP Reply: 6e:23:e8:23:16:fd
ICMP Request
ICMP Request
ICMP Reply
ICMP Reply

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 for this part of the lab requires an additional Cisco router. The network topology is shown in Figure 3.3. Table 3.2 describes the IPv4 configuration of the network.

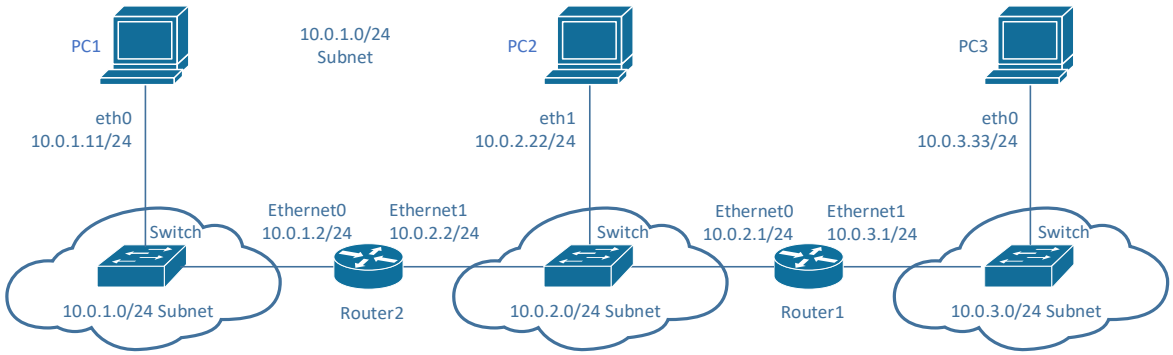


Figure 3.3. Network topology for Part 6.

Cisco Router	Interface Ethernet0	Interface Ethernet1	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	Disabled	10.0.2.22/24	10.0.2.2
PC3	10.0.3.33/24	Disabled	10.0.3.1

Table 3.2. IP Addresses for Part 6.

Exercise 6-a. Network setup

The network topology can be constructed from the topology in Figure 3.1, by disconnecting the *PC2* (*eth0*) interface, and then adding *Router2*.

Step 1: Connect the topology as shown in Figure 3.3. Starting from the topology in Figure 3.1, disconnect the link of *PC2* (*eth0*) from the switch. Then add *Router2* to the configuration as shown in Figure 3.3.

Step 2: Check the IPv4 addresses configured on *PC1, PC2, PC3*, and *Router1* to ensure that they are as given in Table 3.2.

Step 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.

Step 1: First, configure *PC2*.

1. Disable the *eth0* interface with the command

   ```
   PC2$ sudo ip link set dev eth0 down
   ```

2. Delete the entries in the neighbor cache entries by issuing

   ```
   PC2$ sudo ip neigh flush all
   ```

3. Delete the static route to destination 10.0.3.0/24 that was added in Exercise 3-b and set the default gateway of PC2 to 10.0.2.2 with

   ```
   PC2$ sudo ip route del 10.0.3.0/24
   PC2$ sudo ip route add default via 10.0.2.2
   ```

4. Delete the route cache with the command

   ```
   PC2$ sudo ip route flush cache
   ```

5. Run "`netstat -rn`" to confirm that the default route is correctly configured. Take a screenshot of the output.

Step 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). To get a sense of the settings on *PC2*, issue the command

```
PC2$ sudo sysctl -a |grep accept_redirects
```

We want that *PC2* accepts ICMP Redirects. Change the settings of the relevant kernel parameters with the command

```
PC2$ sudo sysctl -w net.ipv4.conf.all.accept_redirects=1
```

Step 3: Next, proceed to *Router2*.

1. Enable IPv4 forwarding and configure the IPv4 addresses of the interfaces as shown in Table 3.2.

2. Set the default route at Router2 to send all traffic to Router1 with the commands

   ```
   Router2# conf term
   ```

```
Router2(config)# ip route 0.0.0.0 0.0.0.0 10.0.2.1
Router2(config-if)# end
```

3. 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 Ethernet0
Router2(config-if)# no ip route-cache
Router2(config-if)# ip route-cache
Router2(config)# interface Ethernet1
Router2(config-if)# no ip route-cache
Router2(config-if)# ip route-cache
Router2(config-if)# end
```

4. Write down the MAC addresses of the Ethernet interfaces of *Router2*. The MAC addresses of all interfaces are displayed with the command

```
Router1# 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*.

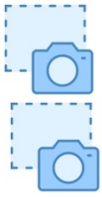
Step 4: Next is the configuration of *Router1*. The IP configuration and the routing tables are exactly as at the end of Part 4.

1. Verify that the IPv4 addresses of the Ethernet interfaces are configured as given in Tables 3.1 and 3.2.
2. Delete the routing table entry for subnet 10.0.1.0/24 with next hop 10.0.2.22.
3. As done on *Router2*, delete the route cache of each interface.
4. Take a note of the MAC address of the *Ethernet0* interface of the router. (This was recorded earlier in this lab.)

Step 5: Start *Wireshark* to capture traffic between *PC2* (*eth1*) and the switch.

Step 6: Issue a *ping* from *PC2* to *PC3*:

```
PC2$ ping -c3 10.0.3.33
```

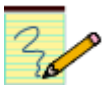


1. Take a screenshot of the output of the ping command. Note that the output mentions that there was a route redirect.

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.

Step 7: Stop *Wireshark* and save the (details of the) ICMP and ARP packets.



Lab Questions/Report

1. Include the screenshot of
   a. The routing table before the *ping* command.
   b. The output of the *ping* command.
   c. 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 Ethernet 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*).
   a. Use the captured traffic and the timeline to infer packet transmissions between *Router1* and *Router2*.
   b. Insert the packets between *Router1* and *Router2* at plausible locations in the timeline drawn in the previous picture. Clearly indicate these packets in the timeline, e.g., using a different color.

Part 7. Forwarding Loops

A potential problem when setting routing tables manually is that forwarding loops may occur. In this part of the lab, you intentionally configure a forwarding loop in the configuration of the routing table and observe what happens to network traffic in such a situation.

The network topology is shown in Figure 3.4. Table 3.3 describes the IPv4 configuration of the network. In the network topology *PC1*, *PC2*, *PC3* and *Router4* are hosts (IPv4 forwarding not enabled) and *Router1*, *Router2*, *Router3*, and *PC4* are IP routers (IPv4 forwarding enabled).

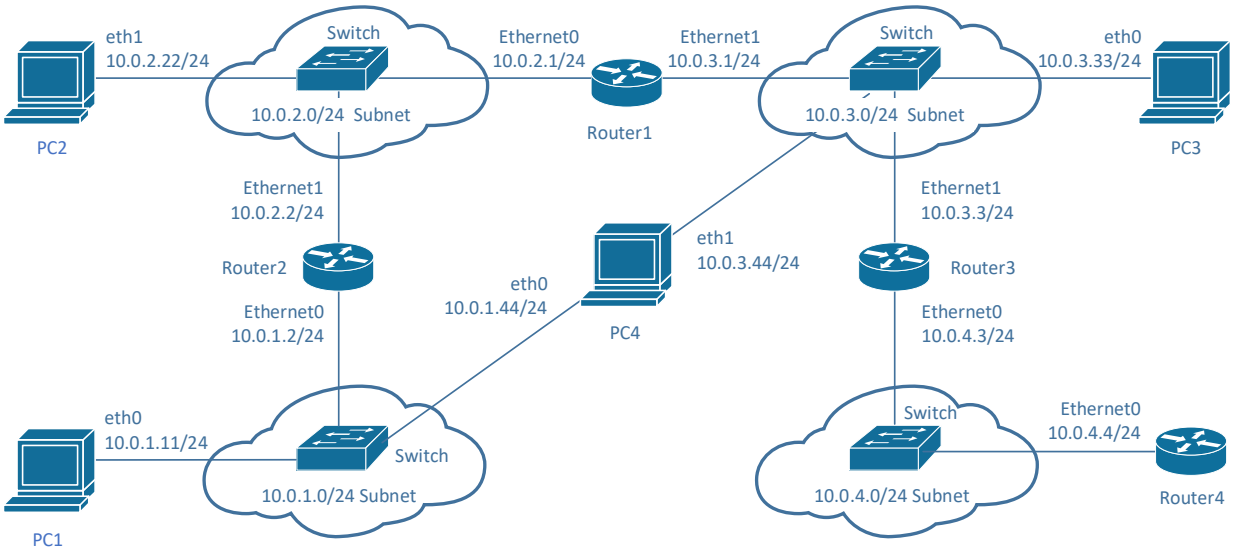


Figure 3.4. Network topology for Part 7.

Cisco Router	Interface Ethernet0	Interface Ethernet1	Default gateway
Router1	10.0.2.1/24	10.0.3.1/24	10.0.3.44
Router2	10.0.1.2/24	10.0.2.2/24	10.0.2.1
Router3	10.0.4.3/24	10.0.3.3/24	10.0.3.44
Router4	10.0.4.4/24	Disabled	10.0.4.3
Linux PC	**Interface *eth0***	**Interface *eth1***	**Default gateway**
PC1	10.0.1.11/24	Disabled	10.0.1.2
PC2	Disabled	10.0.2.22/24	10.0.2.2
PC3	10.0.3.33/24	Disabled	10.0.3.44
PC4	10.0.1.44/24	10.0.3.44/24	10.0.1.2

Table 3.3. IPv4 addresses for Part 7.

Exercise 7-a. Network setup

The network topology in Figure 3.4 can be constructed from the topology in Figure 3.3, by adding *PC4*, *Router3*, *Router4*, and an additional Ethernet switch. The connections of the other devices (*PC1, PC2, P3, Router1, and Router2*) and their IPv4 addresses are the same as in Part 6.

Step 1: Connect the topology as shown in Figure 3.4. Starting with the topology in Figure 3.3, add *PC4*, *Router3*, and *Router4*.

Step 2: Verify that the IPv4 addresses of *PC1, PC2, PC3*, *Router1,* and *Router2* are as given in Table 3.3.

Step 3: Configure the default routes of *PC1*, *PC2*, *PC3*, *Router1*, and *Router2* as given in Table 3.3. If you have completed Part 6 of the lab, most of the IP configurations are already in place. The configurations that must be changed are:

- on *PC1*: Change the default gateway to 10.0.1.2.
- on *PC3*: Change the default gateway to 10.0.3.44.
- on *Router1*: Change the default gateway to 10.0.3.44.

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

Exercise 7-b. Forwarding loop

Step 1: Configure *Router4* as given in Table 3.3. Note that Router4 is running as a host. The commands are

```
Router4# configure terminal
Router4(config)# no ip routing
Router4(config)# ip route 0.0.0.0 0.0.0.0 10.0.4.3
Router4(config-if)# interface Ethernet0
Router4(config-if)# ip address 10.0.4.4 255.255.255.0
Router4(config-if)# no shutdown
Router4(config-if)# end
```

Step 2: Configure Router3 as an IPv4 router with addresses and default gateway as given in Table 3.3, by typing

```
Router3# configure terminal
Router3(config)# no ip routing
Router3(config)# ip routing
Router3(config)# ip route 0.0.0.0 0.0.0.0 10.0.3.44
Router3(config-if)# interface Ethernet0
Router3(config-if)# ip address 10.0.4.3 255.255.255.0
Router3(config-if)# no shutdown
Router3(config-if)# interface Ethernet1
Router3(config-if)# ip address 10.0.3.3 255.255.255.0
Router3(config-if)# no shutdown
Router3(config-if)# end
```

Step 3: Configure *PC4* as an IPv4 router with addresses and default gateway as given in Table 3.3. This is done with

```
PC4$ sudo ip addr add 10.0.1.44/24 dev eth0
PC4$ sudo ip addr add 10.0.3.44/24 dev eth1
PC4$ sudo ip route add default via 10.0.1.2
PC4$ sudo sysctl -w net.ipv4.ip_forward=1
```

~~On PC4, we need an additional configuration. The default configuration in Ubuntu Linux is~~

Step 4: On *PC3*, clear the neighbor cache (`ip neigh flush all`).

Step 5: Issue a *traceroute* from *PC3* to *Router4* to verify that a loop exists:

```
PC3$ traceroute 10.0.4.4
```

You should observe that the traced path contains a loop. The traceroute continues probing until the maximum number of probes (64) have been issued. Take a screenshot of the output of the traceroute command (20 lines of output are sufficient).

Step 6: Start *Wireshark* sessions on one of the interfaces of PC4 (it does not matter which interface is monitored).

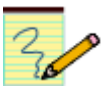
Step 7: Issue a *ping* from *PC3* to *Router4* with the command

```
PC3$ ping -c2 10.0.4.4
```

Observe in *Wireshark* that the same ICMP Echo Request message is looping. Pay particular attention to the Time-To-Live (TTL) field in the IP header.

Step 8: Count the number of times that *Wireshark* sees the same ICMP Echo Request message. Save (the details of) two back-to-back two of these ICMP Echo Request messages for the lab report.

Step 9: Terminate the *Wireshark* traffic capture.



Lab Questions/Report

1. Include the screen capture from Step 7 and explain the outcome.

2. Include the details of the IP headers of two back-to-back ICMP Echo Requests that were saved by *Wireshark*. Describe the difference(s) between the IP headers.

3. Explain why the ICMP Echo Requests do not loop forever in the network.

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. For this part of the lab, we use the network topology from Figure 3.5. This is the same topology as used in Parts 1-5.

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, a host creates link-local addresses for each enabled IPv6 interface (see Lab 2, Part 5)
2. For each link-local address, the host sends an ICMPv6 Neighbor Solicitation to the created address:
   a. If someone replies with an ICMPv6 Neighbor Advertisement, the link-local address is in use.
   b. 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.

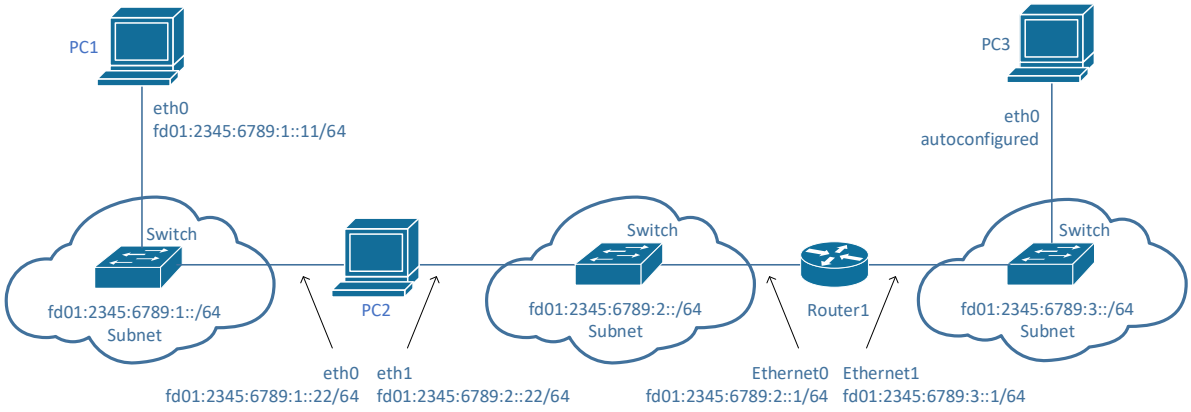


Figure 3.5. Network topology for Part 8.

Table 3.4. IPv6 Addresses for topology in Figure 3.1.

PC	IPv6 address of *eth0*	IPv6address of *eth1*
PC1	`fd01:2345:6789:1::11/64`	–
PC2	`fd01:2345:6789:1::22/64`	`fd01:2345:6789:2::22/64`
PC3	autoconfigured	–
Cisco Router	**IPv6 address of *Ethernet0***	**IPv6 address of *Ethernet1***
Router1	`fd01:2345:6789:2::1/64`	`fd01:2345:6789:3::1/64`

Exercise 8-a. Network setup

Configure the network as shown in Figure 3.5. The IPv6 addresses of the PCs and *Router1* are given in Table 3.4. If you continue from Part 7, follow the instructions in Step 1 (you just need to change a few connections). Otherwise follow the instructions in Step 2.

Step 1: Make sure that PC2 is connected as shown in Figure 3.5, that is, the *eth0* interface of *PC2* connects to the same switch that *PC1 (eth0)* is connected to. If the *eth0* interface of PC2 has been disabled previously, re-enable it with the command

```
PC2$ sudo ip link set dev eth0 up
```

There is no need to change IP addresses or routing tables from the previous parts (They are were all related to IPv4).

Step 2: Connect the Ethernet interfaces of the Linux PCs and the Cisco router as shown in Figure 3.5.

Exercise 8-b. Static IPv6 routes and IPv5 forwarding in Linux

Here you configure *PC2* as an IPv6 router and set up static routes. Conceptually, configuring IPv6 forwarding a Linux PC is analogous to IPv4 forwarding. Below we show relevant commands.

Display IPv6 Routing Table on Linux

`ip -6 route`
Displays the IPv6 routing table.

`netstat -6rn`
Displays the current IPv6 routing table in a table format.

Static IPv6 Routes on Linux

`sudo ip route add` *`fd01:2345:6789:3::/64`* **`via`** *`fd01:2345:6789:2::1`*

Adds a routing table entry for the subnet *fd01:2345:6789:3::/64* with next hop *fd01:2345:6789:2::1/64*.

sudo ip route add default via *fd01:2345:6789:1::22*
Adds the router with IP address *fd01:2345:6789:1::22* 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 *fd01:2345:6789:3::/64*
sudo ip route del default
The commands delete an entry from the routing table.

Enabling/Disabling IPv6 Forwarding

sysctl net.ipv6.conf.all.forwarding
Displays the state of IPv6 forwarding. The system is an IPv6 router when the value is "1".

sudo sysctl -w net.ipv6.conf.all.forwarding=1
Enable IPv6 forwarding

sudo sysctl -w net.ipv6.conf.all.forwarding=0
Disable IPv6 forwarding

Step 1: Start a *Wireshark* session to capture traffic of *PC1* (*eth0*) and *PC2* (*eth0*).

Step 2: On *PC1*, configure the IPv6 address of interface eth0 as given in Table 3.4, and configure PC2 as a default gateway with the commands

```
PC1$ sudo ip addr add fd01:2345:6789:1::11/64 dev eth0
PC1$ sudo ip route add default via fd01:2345:6789:1::22
```

Observe the messages that are sent by *PC1* when the configuration commands are issued.

Check the routing table of *PC1* (with 'netstat -6rn') and take a note of the prefix in the routing table for the default route.

Take a snapshot of the IPv6 routing table at *PC1* displayed with `netstat -6rn' and 'ip –6 route' of *PC2*.

Step 3: On *PC2*, configure the IPv6 addresses as given in Table 3.4. Then, enable IPv6 forwarding on *PC2*. Observe the messages that are sent by *PC1*2, when the configuration commands are issued.

Step 4: Continue on *PC2*. Create a routing table for network *fd01:2345:6789:3::/64* with *Router1* as next hop, with the command

```
PC2$ sudo ip route add fd01:2345:6789:3::/64 via fd01:2345:6789:2::1
```

Take a snapshot of the IPv6 routing table of *PC2* displayed with `netstat -6rn' and 'ip -6 route' of *PC2*.

Step 5: Switch to *PC1*. Verify the configuration of *PC1* and *PC2* by issuing a *ping* from PC1 to *PC2*.

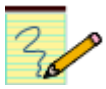
```
PC1$ ping6 -c2 fd01:2345:6789:1::22
```

Now issue a *ping* from *PC1* to the (not yet configured) IPv6 address of Router1 by typing

```
PC1$ ping6 -c2 fd01:2345:6789:2::1
```

With the configured default route on *PC1*, you should observe that *PC1* sends an ICMPv6 Echo Request to *PC2*. Do you observe any other ICMPv6 messages?

Step 6: Stop *Wireshark* and save all ICMP packets.



Lab Questions/Report

1. Include the routing tables of *PC1* and *PC2* from Step 2 and Step 4.
2. Use the screenshots of the routing tables for a comparison of the outputs of the commands `netstat -6pn' and `ip -6 route'. Specifically, describe the types of entries that are provided by one command, by not by the other.
3. Provide a list of the different types of ICMPv6 messages that *Wireshark* captured in Exercise 8-b. For each message type:
   - State the ICMP type (in words)
   - Explain the purpose of this message.
   - For one packet of this type, provide the detailed IPv6 header and the fields of the ICMP6 message.

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

Next you configure *Router1* as an IPv6 router. Once the configuration is completely, you will instantly observe that *Router1* starts a stateless autoconfiguration of *PC3*. This will provide *PC3* with a global IPv6 address and with a default route 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 Ethernet0 fd01:2345:6789:2::22*
Adds a static route to interface Ethernet0 for destination network fd01:2345:6789:2::/64 and next hop router fd01:2345:6789:2::22.
(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*
no ipv6 address *fd01:2345:6789:2::/64* **eui-64**
no ipv6 enable
Removes the routing table entry or disables IPv6 processing.

Step 1: Start a *Wireshark* session to capture traffic between *PC3* (*eth0*) and the switch.

Step 2: *On* PC3, take a note of the link-local IPv6 address of *PC3* (*eth0*), using the command

```
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 screen capture of the output of the command.

Step 3: 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
```

Step 4: 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 Ethernet0
Router1(config-if)# ipv6 address fd01:2345:6789:2::1/64
Router1(config-if)# no shutdown
Router1(config-if)# interface Ethernet1
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
Router1(config)# ipv6 route fd01:2345:6789:1::/64 Ethernet0 fd01:2345:6789:2::22
Router1(config)# exit
```

While you type the above commands, you will observe ICMPv6 messages that are 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 this message.

Step 5: On *Router1*, repeat Step 3 and take a screenshot of the output.

Step 6: On *PC3*, display the IPv6 configuration and the IPv6 routing table with the commands

```
PC3$ ip addr show dev eth0
PC3$ ip -6 route
```

Take a screen snapshot 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?
- 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?

Step 7: Now, verify that the established routes work by running a *ping* and a *traceroute* from PC3 to PC1.

```
PC3$ ping6 -c2 fd01:2345:6789:1::11
PC3$ traceroute6 fd01:2345:6789:1::11
```

Take a screen snapshot of the output.

Step 8: Stop *Wireshark* and save the details of the *ICMPv6 Router Advertisement* messages.

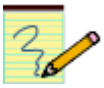
Where is the Router Solicitation?
In the above experiment, the *Router Advertisement* was sent when the Cisco Router was configured as an IPv6 router. The *Router Advertisement* was not sent in response to a Router Solicitation.

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 add a 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 2 (before the *Router Advertisement* was received) and Step 6 (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 *ping6* and *traceroute6* from Step 7.
5. Explain how PC3 constructed its global unique unicast address.
6. Explain how PC3 determines the IPv6 address of the default gateway.

Appendix: 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.

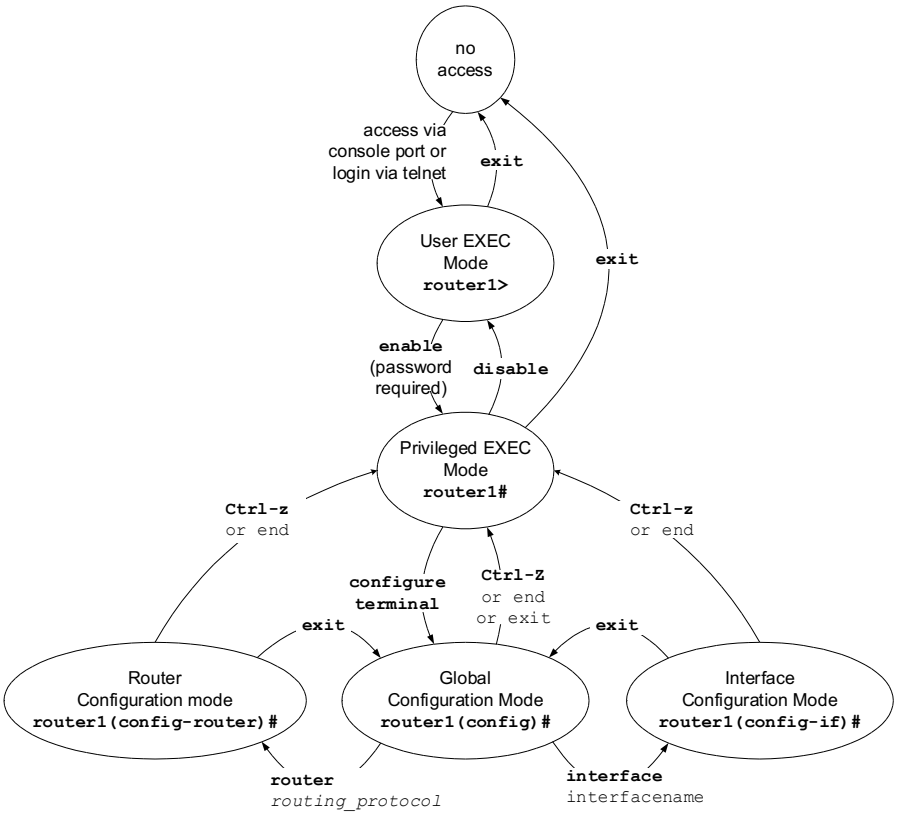


Figure 3.6. Cisco IOS Command Modes.

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*. [1] 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.

[1] 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.

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 in port 0 of slot 0 of the router. FastEthernet0/1 identifies port 1 on the same card. 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., Fast*Ethernet0*, Fast*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., Fast Ethernet 0/0 or Gigabit Ethernet 0/0/0. 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 1 of a 10 Mbps Ethernet card in slot 0 of the router is entered with the command

```
Router1(config)#interface Ethernet0
```

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, including the *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 a 10 Mbps Ethernet (*Ethernet*) interface card with two ports which is located with names *Ethernet0* and *Etherne1*. 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 Ethernet0
Router1(config-if)# no shutdown
Router1(config-if)# ip address 10.0.2.1 255.255.255.0
Router1(config-if)# interface Ethernet1
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 *Ethernet0*. 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 *Ethernet0* 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 *Ethernet0*
Displays information about all network interfaces. When an interface name is given as argument, here, *Ethernet0*, 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

Ethernet0 is up, line protocol is up
  Internet address is 10.0.2.1/24

Ethernet1 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 *Ethernet0* and *Ethernet1* 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 e0/0`	`interface Ethernet0/0`

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
```