Lab

1

Introduction to the Internet Lab

PC (“bare metal”) Version

What you will learn in this lab:

- Overview of the equipment
- Console access to a Cisco router
- Saving your data
- Navigating your way around Linux
- Working with the Wireshark protocol analyzer

Updated: September 2021

Table of Content

Study Material for Lab 1

1. Read Appendix B to learn about the Linux file system and Linux commands.

2. **Man Pages:** The PCs run the Linux operating system. This assignment asks you to review some Unix commands. Man pages exist on every lab machine. You can also find the manual pages ("man pages") online at

   http://man7.org/linux/man-pages/

- For each of the following commands, type the name of the command as a search term.
- Read the man pages of the following commands:
  `man`, `pwd`, `ls`, `more`, `mv`, `cp`, `rm`, `mkdir`, `rmdir`, `chmod`, `kill`, `ping`

3. **Wireshark:** Read about the *Wireshark* network analyzer at the website

   https://www.howtogeek.com/104278/how-to-use-wireshark-to-capture-filter-and-inspect-packets/

- There are numerous websites and videos that explain the operation of *Wireshark* (e.g., search for *"Wireshark 101"* on YouTube.com).
  Find a few of these sources and learn about the syntax of display filter expressions in *Wireshark.*

Prelab 1

1. What will happen if you type "`man man`" in Linux?

2. How can you use the command "`ls`" to find out about the size of file `/etc/lilo.conf`?

3. What happens if you have two files with names *file1* and *file2* and you type "`mv file1 file2`"? Which option of "`mv`" issues a warning in this situation?

4. What is the command that you issue if you are in directory "`/`" and want to copy the file `/mydata` to directory `/labdata`?

5. What is the command that you issue if you are in directory "`/`" and want to copy all files and subdirectories under directory `/mydirectory` to directory `/newdirectory`?

6. What happens if you type the command "`rm *`" in a directory?

7. What is the command that you issue if you want to delete all files and directories under the directory `/mydirectory`?

8. Write the syntax of Wireshark display filters to show packets with the following properties:

   a. IPv4 datagrams with source IPv4 address equal to 10.0.1.12.

   b. ICMP messages with source or destination IPv4 address equal to 10.0.1.12.

   c. IPv4 datagrams containing TCP segments with source or destination IP address equal to 10.0.1.12.

   d. IPv4 datagrams containing TCP segments with source or destination port number 23.

9. Write the syntax of Wireshark display filters to show packets with the following properties:

   a. IP datagrams with a destination IP address equal to 10.0.1.50 and frame sizes greater than 400 bytes

   b. ICMP messages with source or destination IP address equal to 10.0.1.12 and frame numbers between 15 and 30.

   c. TCP segments with source or destination IP address equal to 10.0.1.12 and using port number 23.

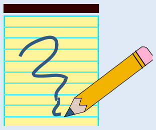
Lab 1: Introduction to the Internet Lab

In the this and all other labs, icons in the page margin ask you do perform certain tasks. There are three different icons.

Icons in Lab instructions

- If you find the icon, you are asked to take a screen capture of a window on your desktop.

- The floppy disk icon tells you that you need to save data from the console of a PC or a router.



- The notepad symbol indicates an assignment or question to be included in the lab report.

In Part 6, the lab provides options for saving data. The following exercise shows how to take a screen capture.

Part 1. Becoming Familiar with the Equipment

The equipment that you are working with in the lab has a setup similar to Figure 1.1 shown below.

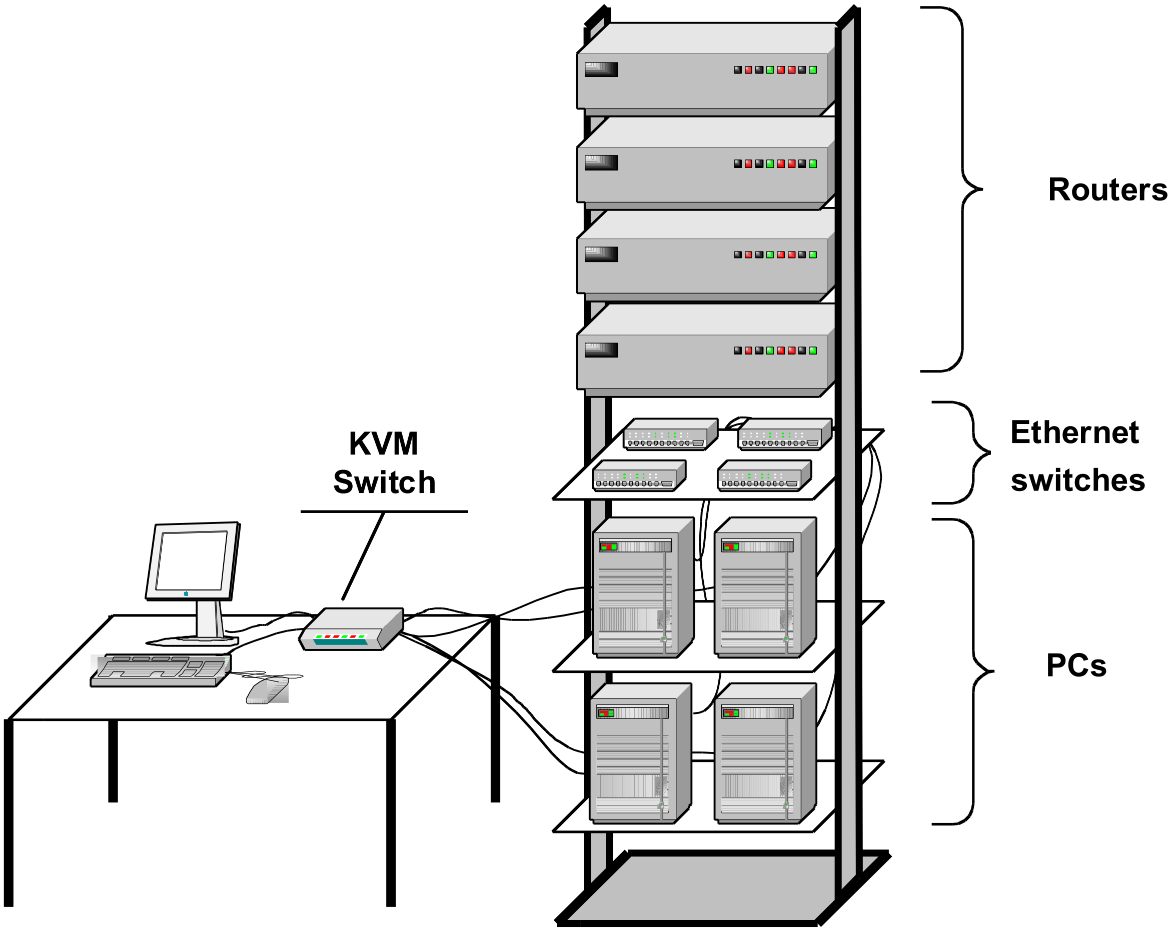


Figure 1.1. Internet Lab equipment rack.

Equipment rack:

1. 4 Cisco routers
2. 4 Ethernet switches
3. 4 hosts
4. 1 Monitor, 1 keyboard, 1 mouse, 1 KVM switch
5. Cables and connectors

Please take a few minutes to compare the following description with the actual equipment:

1. Four Linux PCs, which are labeled as *PC1, PC2, PC3,* and *PC4*. The PCs have the Linux (Ubuntu 20.04) operating system installed. All Linux PCs have two Ethernet network interface controllers (NICs) installed, which are labeled *eth0* and *eth1* on the back of the computer. The PCs have several USB ports and may have a serial port with a DB9 connector with label *ttyS0*.
2. Four Cisco routers, which are labeled as *Router1, Router2, Router3*, and *Router4*. Each router has at least two Ethernet interfaces and two WAN serial interfaces. The Cisco routers also have a console port that is used to access the router for configuration. Depending on the type of router, the console port uses a USB connector, or an RJ-45 connector, or both.
3. Four Ethernet switches, with at least four ports each. These are simply referred to as “switches” in the lab manual.
4. A monitor, a keyboard, a mouse, and a KVM (Keyboard-Video-Mouse) switch. The KVM switch connects the keyboard, monitor, and mouse to the four Linux PCs. The KVM switch gives you control over all four Linux PCs from a single workstation, but you can only access one computer at a time.
5. Different types of cables:
    a. “Ethernet cables”: These cables are used to build Ethernet networks. They are Cat 6 UTP (Category 6 unshielded twisted pair) cables.
    b. “Serial cables”: We use these cables to connect a PC to the console port of a Cisco router. They can be either a USB cable or a so-called console cable. At each PC, a serial port (USB or DB9) is connected to the console port of a Cisco router.
    c. “WAN serial cables”: These cables are used to connect WAN serial ports of the Cisco routers.

Exercise 1. Explore equipment rack

Explore the equipment rack and identify the components listed above.

Part 2. and Using the KVM switch and login to PCs

Exercise 2-a. Using the KVM switch

Step 1: Set the KVM switch to *PC1*.

Step 2: If *PC1* is already up and running, go to Step3. Otherwise, start the computer. Do not change the KVM switch while the computer is booting up.

Step 3: Repeat Steps 1 and 2 for the other PCs.

Reboot PCs

- Reboot all PCs at the start of each lab.
- The PCs have a "Live" version of Ubuntu installed. With this, rebooting a PC reset the PC to the original configuration. Note that all changes and all added files are lost after a reboot.

Don't use KVM switch when PC is rebooting

- Do not switch the KVM switch while a computer is rebooting, otherwise the keyboard and mouse may not work properly.

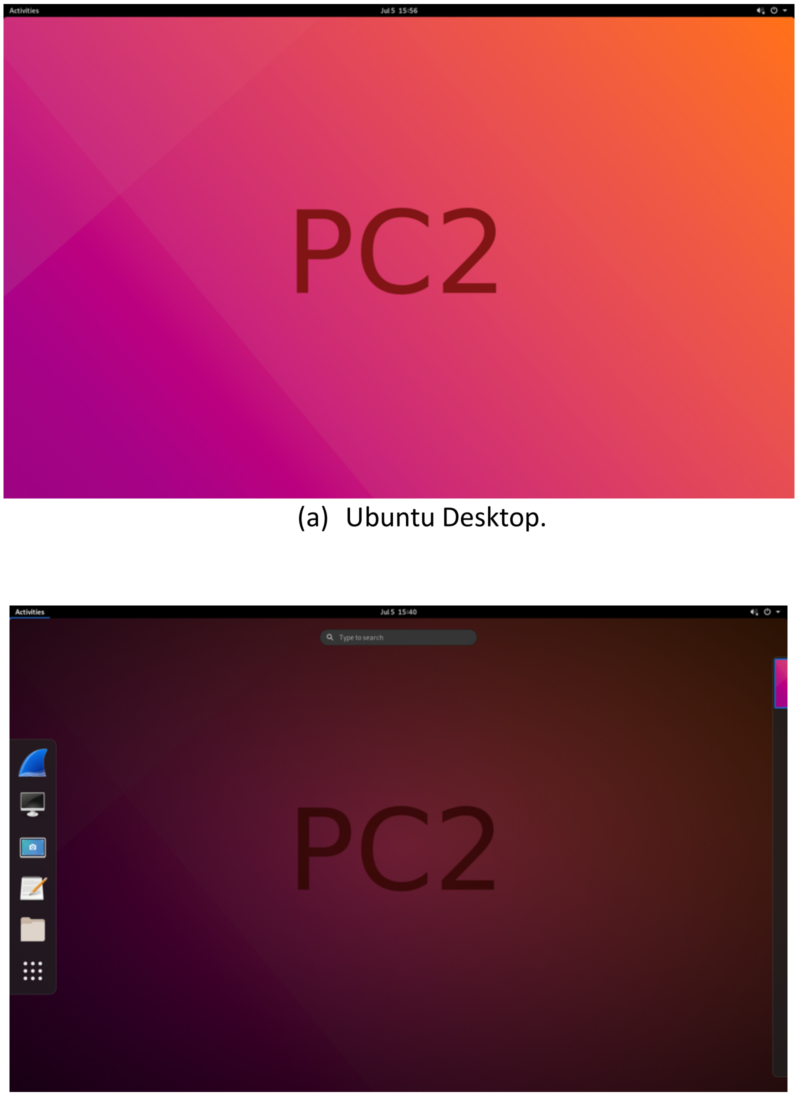
Exercise 2-b. Login to PC

Login to PCs and Cisco routers

- The username and password on the PCs are both "*labuser*". The account *labuser* is a member of the sudo group. This gives *labuser* administrator ("root") privileges.

Step 1: On *PC2*, complete a login for username *labuser*. This displays the desktop of *PC2* as shown in Figure 1.3(a). Moving the mouse to the upper left corner of the desktop ("Activities") displays the sidebar shown in Figure 1.3(b). The sidebar displays shortcuts to frequently used applications. From the top, these are:

- Wireshark;
- LX Terminal (terminal window);
- Screenshot;
- Mousepad (text editor);
- File Manager;
- List all available applications.



(a) Ubuntu Desktop.

(b) Desktop with sidebar.

Figure 1.3. Ubuntu 20.04 Desktop.

Step 2: On *PC2*, open a terminal window by clicking on the *LXTerminal* icon in the sidebar. The terminal window shows the prompt

```
labuser@PC2$
```

For brevity, we will use the following to show the prompt

```
PC2$
```

> ***Terminal window* and *console***
> In the lab manual, a "terminal window" is commonly referred to as a "console window" or simply "console". These terms are used interchangeably.

Step 3: Type the command

```
PC2$ ifconfig
```

The command displays information on the network interfaces of the PC. Note that no IP addresses are assigned to the Ethernet interfaces.

Step 4: Take a snapshot of the output from Step 3. Click on the “Screenshot” icon in the sidebar of the desktop. The tool is self-explanatory. By default, screen snapshots are saved to the *Pictures* folder in the home directory.

Lab Questions/Report

- Provide the screenshot from this exercise.
- Briefly explain the information that is displayed in the screenshot.

Part 3. Setting up a Network Configuration with PCs

In this part of the lab, you set up a network configuration as shown in Figure 1.4.

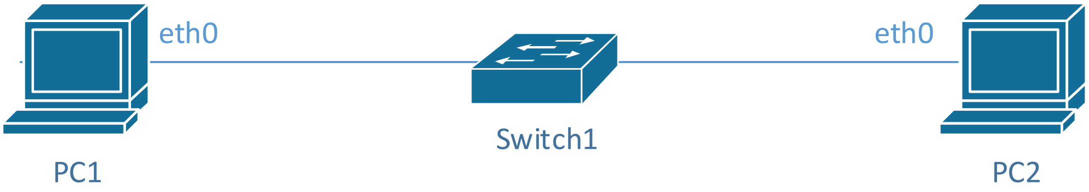


Figure 1.4. Network configuration with PCs.

Exercise 3-a. IPv4 configuration

In this exercise, *PC1* and *PC2* should be connected to a switch with Ethernet cables as shown in Figure 1.4.

The PCs do not have pre-configured IP addresses. You configure an IP address on a PC by typing commands in the console window. When the PC is rebooted, all configured IP addresses will be lost.

The IPv4 addresses of the PCs are shown in Table 1.1. The notation 10.0.1.11/24 means that the IPv4 address is 10.0.1.11 and the network prefix is 24 bits long. A network prefix of 24 bits corresponds to a netmask set to 255.255.255.0. With the above configuration, the *eth0* interfaces of *PC1* and *PC2* are added to the 10.0.1.0/24 subnet.

Table 1.1. IP Addresses of PCs.

PCs	IP Address of *eth0*
PC1	10.0.1.11 / 24
PC2	10.0.1.12 / 24

Step 1: Attach *PC1* and *PC2* to the same switch with Ethernet cables.

- Connect port *eth0* of *PC1* to a port on the switch.
- Connect port *eth0* of *PC2* to another port on the switch.

> **Note:** If your switch has a port that is labeled “Uplink”, do not use this port to connect the PCs. Uplink ports are used to interconnect a switch to another switch.

Step 2: Set the KVM switch to *PC1* and open a terminal window on *PC1*. Configure an IP address on *PC1* with the command

```
PC1$ sudo ip addr flush dev eth0
PC1$ sudo ip addr add 10.0.1.11/24 dev eth0
```

The first command removes any previous IP configuration, and the second command sets the new IP address.

Step 3: Set the KVM switch to *PC2* and open a terminal window in *PC2*. Configure an IP address on *PC2* with

```
PC2$ sudo ip addr flush dev eth0
PC2$ sudo ip addr add 10.0.1.12/24 dev eth0
```

Exercise 3-b. Issuing ping commands

Next you use the ping command to verify that the IP configuration was successful.

Ping on Ubuntu

- On Ubuntu, ping continues to send packets until you interrupt the command with the `Ctrl-c` key.
- You can limit the number of messages sent with the -c option. For example, `ping -c5 10.0.1.12` issues five messages to 10.0.1.12.
- When using ping on the PCs, we recommend to always send at least two ICMP Echo Request packets, since on some occasions, the first ICMP Echo Request does not trigger a response by the receiver.

Step 1: Set the KVM switch to *PC1* and open a terminal window. Issue a ping command to *PC2* by typing

```
PC1$ ping -c2 10.0.1.12
```

Take a snapshot that shows the output of the command.

Lab Questions/Report

- Provide the screenshot from this exercise.

KVM switch:

- From now on, the instructions do not mention the KVM switch further. Whenever you see the command for a particular PC, for example,
  `PC1$ ping 10.0.1.12`
  it is implied that you need to set the KVM switch to the correct PC (*PC1* in the example).

Exercise 3-c. Ethernet connectivity without a switch

If you only connect two PCs you do not need an Ethernet switch. You can simply connect two ports with an Ethernet cable.

Step 1: Use an Ethernet cable to connect the *eth0* port of *PC1* to the *eth0* port of *PC2*.

IP address disappears after disconnecting cable
On Ubuntu, when configuring an IP address with the `ip addr` commands, the IP address is deleted whenever the Ethernet cable is disconnected at the PC or at a switch.

Step 2: Repeat Step 2-3 in Exercise 3-a and Step 1 in Exercise 3-b. You should get the same result.

Trivia about Ethernet cables
Exercise 3-c may appear trivial, but there is technology involved that makes it work. An Ethernet cable is an unshielded twisted pair (UTP) cable with eight conductors. Some conductors are used for transmitting signals and others for receiving them. If the device at one end uses a particular conductor for transmission, the other side of the connection uses the same conductor for receiving signals. Because of this, two devices that are connected by an Ethernet cable must use the connectors differently. In fact, the wiring of an Ethernet port (technically, an “RJ45 port”) is different at a switch and a non-switch, such as a PC.

In the past, different types of cables were used when connecting a PC to a switch, or a PC to another PC. The former was called a “straight-through cable” and the latter was called a “crossover cable”. A crossover cable changes the assignments of pins of the RJ45 connectors at the two ends of the cables, such that the assignment for transmitting and receiving are switched. The default wiring of a UTP cable with RJ45 connectors is straight-through, and all Ethernet cables in the lab are of this type.

Since the last 1990s, Ethernet ports on computers and hubs can detect the required wiring and configures the connectors appropriately. This makes the distinction between straight-through and crossover cables a moot point. The feature of autodetection and configuration of Ethernet ports is associated with the acronym “Auto MDI-X”. In Gigabit Ethernet (1000BaseT), Auto MDI-X is a required feature, thus making crossover cables obsolete. In conclusion, Exercise 3-c is rather obvious because of Auto MDI-X.

Part 4. Connecting to a Cisco Router

Next you learn how to access the Cisco routers. Each PC is connected to a Cisco router via a serial cable, which is either a USB cable or a serial cable that attaches to a DB9 serial port on the PC to the console port of a Cisco router. By running a terminal emulation on a PC, you can set up a console window for configuring the connected Cisco router.

Exercise 4-a. Connect PCs to the console port of the Cisco routers

Each PC connects to the console port of a Cisco router with a serial cable. With this, we can access the Cisco routers from the PCs. We use the convention that *PC1* is connected to *Router1*, *PC2* to *Router2*, and so forth.

Step 1: Check if the PCs and the Cisco routers are connected by a serial cable. If so, continue with Step 2. If not, you need to use the provided serial cables for connecting PCs and Cisco routers.

- If the serial cable has a DB9 connector, attach it to the DB9 serial port of the PC, and the other end of the cable to the console port of the Cisco router. If the PCs have a DB9 serial port, it is labeled as *ttyS0.*
- If the serial cable has a USB connector, attach the cable to any free USB port on the PC. The other end of the cable connects to the console port of the Cisco router.

Once you identified the cable, connect *PC1* to *Router1*, *PC2* to *Router2*, and *PC3* to *Router3*, and *PC4* to *Router4*.

Step 2: In Linux, the serial ports of a PC are accessed via a *device files* located in directory */dev*. You need to identify the device file for the serial port that is connected a Cisco router. The name of the device file depends on whether the connection uses a DB9 port or a USB port as serial interface.

- **DB9 port:** The name of the device file is *ttyS0*, which matches the label next to the DB9 port on the PC. The absolute pathname of the device file is /dev/ttyS0.
- **USB port:** Here, the device file is created dynamically during boot up or at the time when the USB cable is connected to the Cisco router.

  To determine the name of the device file, open a terminal window and type the command

  ```
  PC1$ dmesg | grep tty
  ```

  The *dmesg* command displays messages from the device drivers, including the names of connected USB devices. The sequence `*| grep tty*` filters from the output all lines that contain device file names for a serial connection. Look for a line that looks like this

```
[986.983671] cdc_acm 3-11:1.0: ttyACM0: USB ACM device
```

  The last line shows that the name of the device file is *ttyACM0*, and the absolute pathname of the device file is /dev/*ttyACM0*.

Exercise 4-b. Accessing a Cisco router via the console port with Screen

The next step is to run a terminal emulation program on the PC for sending commands via the serial port to the console port of the Cisco Routers. In the Internet Lab, you use the *screen* communication software to access the router.

The following steps access the console port of *Router1* from *PC1*. The steps are analogous for other PCs.

> **Learn more about the screen command**
> You can check the *man* pages of *screen*
> `$ man screen`
> or consult references on the Internet, such as
> https://www.geeksforgeeks.org/screen-command-in-linux-with-examples/

Step 1: Access the console port of the Cisco router with the *screen* command from a terminal window. Depending on the type of serial port that is used, issue the command

```
PC1$ sudo screen /dev/ttyS0
```

or

```
PC1$ sudo screen /dev/ttyACM0
```

Then, you should see the command prompt of the router. On *PC1*, which is connected to *Router1* by a serial cable, the command prompt is

```
Router1>
```

If the prompt does not appear, then hit the <*Enter*> key several times.

Step 2: Now you can issue/run Cisco IOS commands. To see a list of available commands, type

```
Router1>?
```

Step 3: To view and change system parameters of a Cisco router, you must enter a different command mode, called *privileged EXEC* mode. This is done by typing

```
Router1> enable
Password: <enable secret>
Router1#
```

You need a password, called the *enable secret*, to enter the privileged *EXEC mode*. The password on all routers is *labuser*.

Step 4: Display the current configuration of the router by typing the command

```
Router1#show running-config
```

Take a snapshot that shows the output of the command.

Step 5: End the terminal session by typing "`Ctrl+a`" and then "`k`". When prompted, type "`y`".

Accessing Cisco routers in future labs

- In the following labs, the instructions for accessing any router, say *Router3*, only says *"On Router3, …."*
  Then, follow the instructions as given above to access the router.

Lab Questions/Report

Provide the screenshot from this exercise

Part 5. Setting up a Network Configuration with a Cisco Router

In this part of the lab, you will set up a network configuration between routers as shown in Figure 1.5.

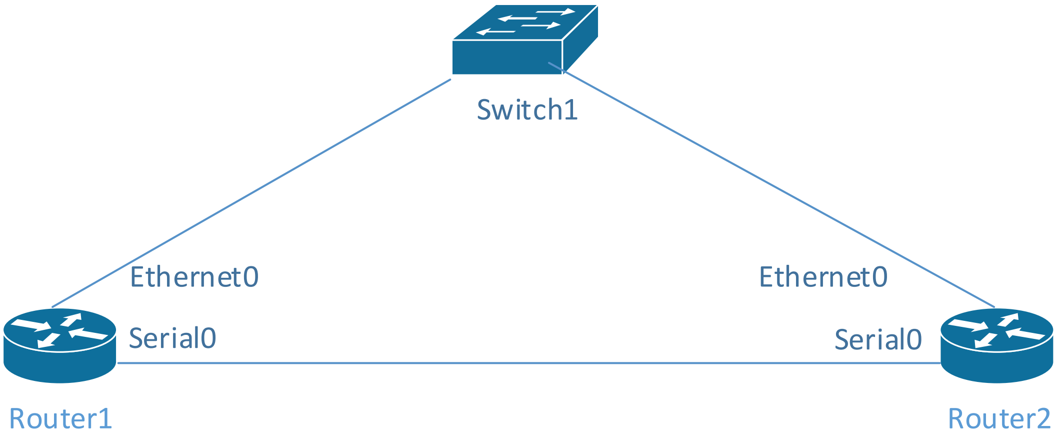


Figure 1.5. Network configuration for Exercise 3-a.

Exercise 5-a. Configuring network interfaces on Cisco routers

In this exercise, you connect two Cisco routers by an Ethernet link.

Naming conventions of network interfaces

The naming convention for network interfaces of Cisco routers in the lab manual is different from the names used by the Cisco router in the equipment rack. The lab instructions *Ethernet0, Ethernet1, …* for the Ethernet interfaces, and *Serial0, Serial1, …* for the serial WAN interfaces. The actual names of the interfaces of the Cisco routers in the equipment rack depends on the type of router as shown in Table 1.4.
For example, if the Cisco routers are 2800 class routers, replace *Ethernet0* in the lab instructions with *FastEthernet0/0*.

Table 1.4. Naming conventions of router interfaces.

Lab manual		Cisco 2800 /2900 class		Cisco 4000 class
Ethernet0	→	*FastEthernet0/0 or F0/0*	→	*GigabitEthernet0/0/0*
Ethernet1	→	*FastEthernet0/1 or F0/1*	→	*GigabitEthernet0/0/1*
Serial0	→	*Serial0/0/0 or S0/0/0*	→	
Serial1	→	*Serial0/0/1 or S0/0/1*	→	

Table 1.5. IP Addresses of routers.

Routers	IP Address of *Ethernet0*	IP Address of *Serial0*
Router1	10.0.1.1 / 24	10.0.2.1 / 24
Router2	10.0.1.2 / 24	10.0.2.2 / 24

Step 1: Set up the network topology shown in Figure 1.5.
Use Ethernet cables to connect the Ethernet0 interfaces of *Router1* and *Router2* to an Ethernet switch (Switch1 in Figure 1.5). Use a WAN serial cable (a blue cable) to connect interfaces *Serial0* of *Router1* and *Router2*.

Step 2: Check that *PC1* and *PC2* are connected to *Router1* and *Router2*, respectively, by a serial cable. If not, follow the instructions from Exercise 4-a. Power on *Router1* and *Router2* (if they are not already on).

Step 3: On *Router1*, open a console using the instructions from Step 1 in Exercise 4-b. Enter the *privileged EXEC* mode with the command

```
Router1> enable
Password: <enable secret>
```

Always enter privileged EXEC mode on the Cisco routers

For the remainder of this lab and in all following labs, always enter the privileged EXEC mode after connecting to a Cisco router. The instructions will assume that you entered the enable command before typing configuration commands.

Step 4: On *Router1*, enter the following commands:

```
Router1# config terminal
Router1 (config)# interface Ethernet0
Router1 (config-if)# ip address 10.0.1.1 255.255.255.0
Router1 (config-if)# no shutdown
Router1 (config-if)# interface Serial0
Router1 (config-if)# ip address 10.0.2.1 255.255.255.0
Router1 (config-if)# no shutdown
Router1 (config-if)# exit
Router1 (config)# exit
Router1>
```

Run the same commands on *Router2*, where you use the IPv4 addresses 10.0.1.2 (to replace 10.0.1.1) and 10.0.2.2 (to replace 10.0.2.1)

The netmask 255.255.255.0 adds the Ethernet0 interfaces of *Router1* and *Router2* to subnet 10.0.1.0/24 and the *Serial0* interfaces to subnet 10.0.2.0/24.

Commands for IPv4 address configuration

The commands for the IP configuration of routers will be covered in Lab 3, and the IP configuration of PCs is covered in Lab 2. For Lab 1, there is no expectation that you have a grasp of the available commands. Just type the commands as they are given here.

Exercise 5-b. Issuing ping commands

After connecting the Cisco routers and configuring the IP addresses, *Router1* and *Router2* can communicate with each other. The following steps verify that the two Cisco routers are properly connected and configured. The test consists of running the *ping* command between *Router1* and *Router2*. A successful *ping* command confirms that a host or router is reachable via the Internet Protocol. The *ping* command sends an ICMP Echo Request datagram to an interface, and expects an ICMP Echo Reply datagram in return.

Ping on Cisco routers

On a Cisco router, a ping command issues five messages. After the messages are sent, the console displays how many replies where received. Receiving at least one reply makes the ping successful. Usually, the success rate is either 0% or 100%.

Step 1: In the console window of *Router1*, issue ping commands to *Router2*. The command

```
Router1# ping 10.0.1.2
```

pings the Ethernet0 interface of *Router2*, and the command

```
Router1# ping 10.0.2.2
```

pings the WAN serial port interface on *Router2*.

Step 2: Take a screen capture of the console output of the ping commands.

Lab Questions/Report

1. Include the screen capture from Exercise 5-b.

Part 6. Saving and transferring data

Most lab exercises ask you to save data that is displayed on the monitor to a file. The purpose of the following exercises is to become familiar with various methods to save data to a file.

> **Saved data on PCs are not persistent**
>
> The data that you save on the PCs is lost when the PCs are rebooted.

All exercises of Part 6 are done on *PC1*.

Exercise 6-a. Saving data to a file with redirection operators

Linux provides an easy way for redirecting the output of a command to a file via the redirection operators > and >>. For example, the command

```
PC1$ ls > fname
```

writes the output of the ls command to file *fname*. All previous data in the file is deleted. The command

```
PC1$ ls >> fname
```

appends the output of the ls command to file *fname*, without deleting any data. Cisco routers do not have such a command.

Exercise 6-b. Saving data with a text editor (along with copy/paste)

In the lab, we use the *Mousepad* editor. To edit a file with name *fname* using *Mousepad*, simply type

```
PC1$ mousepad fname
```

Alternatively, you can start Mousepad by selecting the icon of the editor on the Activities side bar (see Figure 1.3(b)). Once the editor is started you can open a file ("File→Open"), edit the content, and then save the changes ("File→Save"). You terminate Mousepad by selecting "File→Quit".

Part 7 of this lab has an exercise about editing a file with Mousepad.

In *Mousepad*, you can copy text by highlighting the text, and pressing `Ctrl-c`. The text is then pasted by pressing `Ctrl-v`.

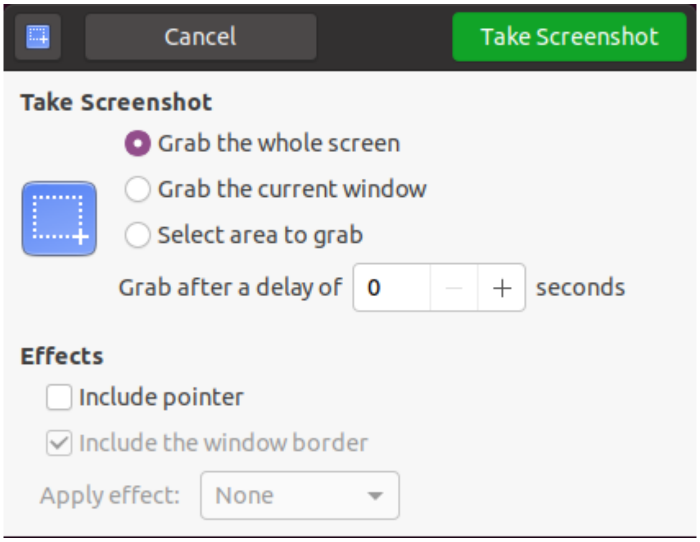
> **Copy/Paste in a terminal window or in a file**
>
> You can also you can copying data to and pasting data from the clipboard in a terminal window. To copy to the clipboard, highlight text and then type *Shift-Ctrl-c* a (or, right-click

on the mouse and select “*Copy*”). To paste from the clipboard, left click the mouse at the intended position and type *Shift-Ctrl-v* a (or, right click on the mouse and select “*Paste*”).

On Linux systems, typing `Ctrl-c` in a terminal window *terminates* a command.

Exercise 6-c. Screen captures

This exercise covers details of the *Screenshot* utility. Recall that you can start *the* utility by selecting the *Screenshot* icon in the sidebar of the desktop (Figure 1.3(b)). You can take a screenshot of the entire desktop, a part of the desktop, or a selected window on the desktop. *Screenshot* displays a window as shown in Figure 1.6. By default, screen captures are save in directory `~/Pictures`.



Take Screenshot

● Grab the whole screen

○ Grab the current window

○ Select area to grab

Grab after a delay of 0 seconds

Effects

□ Include pointer

☑ Include the window border

Apply effect: None

Figure 1.6. Screenshot utility.

Exercise 6-d. Saving data to a flash drive

In all labs, you need the data saved in the lab sessions to complete the lab report. Since the equipment of the Internet Lab is not connected to the Internet, the most convenient way to transfer your saved data is with USB flash drive. Linux systems recognize most USB flash drives used on PCs and Macs. The following description assume that you have a USB flash drive with you.

Step 5: Connect a USB flash drive. When you connect a flash drive with label `FLASHDRIVE` to any of the USB ports, the flash drive is mounted in the directory `/media/labuser/FLASHDRIVE`.

Step 6: View flash drive in the file manager. When you open the File Manager (Figure 1.3(b)), a mounted flash drive will be shown under *Devices*. Figure 1.7 shows how a flash drive with label FLASHDRIVE appears in the File Manager. From there, you can drag and drop files from the *labuser* account to the flash drive.

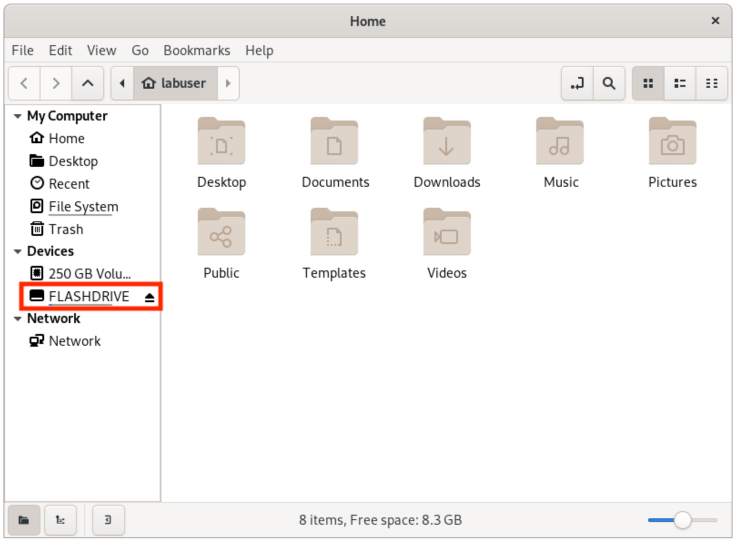


Figure 2.7. File Manager with mounted flash drive

Step 7: Moving files with drag and drop to a flash drive. To quickly move files from a directory on a PC to the flash drive, open a second File Manager window (“*File → New Window*” or *Ctrl-n*). Then select one or more files and drag them to the flash drive.

Alternatively, you can use the command line in a terminal window to move files to the flash drive. Suppose the label of your flash drive on *PC1* is FLASHDRIVE. Then you can move a file *tmp.txt* in your home directory with the command

```
PC1$ mv ~/tmp.txt /media/labuser/FLASHDRIVE
```

Step 8: Disconnecting a flash drive. Before disconnecting the flash drive, push the eject icon in the File Manager. The icon is shown on the right inside the red rectangle in Figure 1.7. After unmounting, you can safely remove the flash drive.

You can also use commands to unmount the flash drive. For a flash drive with label FLASHDRIVE the command is

```
PC1$ umount /media/labuser/FLASHDRIVE
```

Always unmount before removing the flash drive

On Linux systems, always unmount (`eject’) a flash drive before disconnecting it from the computer. Otherwise, the file system on the flash drive may get corrupted.

Part 7. Using the Linux Operating System

Here you explore the Linux system by trying out commands that are typed in a terminal window.

Exercise 7-a. Linux commands

Step 1: Review the Linux commands discussed in Appendix B. Use the commands to perform the following sequence of tasks on PC:

1. Change to the home directory.
2. Create a directory *test* in the home directory.
3. Copy the file */etc/hosts* to directory *test*.
4. Change to directory *test*.
5. Change the name of file hosts to *hostfile*.
6. List the content of directory *test*.
7. List the content of *hostfile*.
8. Remove all files in directory *test*.
9. Remove directory *test*.

Step 2: List the recent commands that you issued on your computer by typing

```
PC1$ history 10
```

If you issued more than 10 commands to complete the list of tasks, replace 10 by a larger number. If you used fewer commands, type a smaller number. Take a snapshot of the output.

Exercise 7-b. The Mousepad text editor

Mousepad can open multiple files at the same time, and has many useful features, such as “find and replace”.

If you are familiar with the command line text editors such as *vi*, *vim*, or *nano*, you may use these instead for this exercise.

Step 1: From the home directory, copy the file */etc/hosts* to *myfile*.

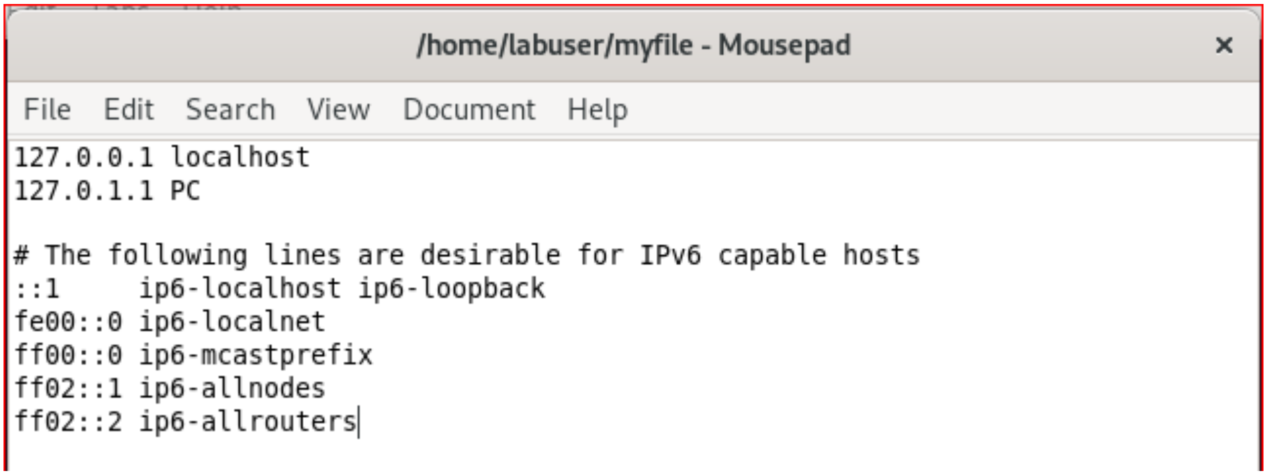
Step 2: Open *myfile* with the *nano* text editor by issuing the command

```
PC1$ mousepad myfile
```

Using Mousepad for system files
To use Mousepad for editing a system file, that is, a file owned by the *root* account, type

```
PC1$ sudo mousepad myfile
```

This will display a window similar to Figure 1.8. The window shows the content of the file. You can make modifications by moving the cursor with the arrow keys and typing or deleting content. Mousepad is a very simple GUI text editor and you can use all the common commands which you usually find in other GUI text editors.


/home/labuser/myfile - Mousepad
×
File Edit Search View Document Help

```
127.0.0.1 localhost
127.0.1.1 PC

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Figure 1.8. The Mousepad text editor

Step 3: Edit the content of the file as follows:

- Revert the order of lines (first line becomes last line, 2nd line becomes 2nd line from the bottom, etc.)
- Change all occurrences of “*ff*” to “*gg*”.
- Add a new line between lines 5 and 6, where you insert your student ID.

Step 4: Take a snapshot of the Mousepad window when you are done.

Step 5: Exit the Mousepad editor by clicking “*File → Close Windows*” or by typing *Ctrl-q*.

Lab Questions/Report

1. Provide the snapshot from Step 2 in Exercise 7-a.

2. Provide the snapshot from Step 4 in Exercise 7-b.

Part 8. Locating Information on the Network Configuration

Linux has a large number of parameters for the network configuration. These parameters include IP addresses, the host name, and whether the system operates as an IP router. Configuration parameters are stored in the operating system kernel and can be modified by a user with administrator (sudo) privileges. Kernel parameters can be accessed in different ways. Here, you learn about two methods to read and modify kernel parameters. One of them is the command *sysctl*, and the other one accesses files in the */proc* directory.

Kernel parameters are initialized through configuration files, which are consulted when the system is booted up. Configuration files in Linux have evolved significantly. What complicates matters is that old versions of configuration files may co-exist with newer versions. This lab exposes you to structure of configuration files that, while not the most recent, does not require a learning curve.

> **Configuration files on a Live Ubuntu system**
> On a "Live" version of Ubuntu, all configuration changes are reset whenever a PC is rebooted.

Exercise 8-a: The *sysctl* command

Step 1: On *PC1*, list the complete set of kernel parameters by typing the command

```
PC1$ sudo sysctl -a
```

Record the output to a file. Browse the file for parameters whose meaning you can guess.

Step 2: Kernel parameters are organized in a hierarchical fashion with dots separating the hierarchy levels. We are mostly interested in the kernel variables for the network operation, which start with "*net.*". On *PC1*, list these kernel variables by typing

```
PC1$ sudo sysctl -a | grep net | less
```

The above line has three commands which are linked by pipes ('|'), such that the output of one command becomes the input of the next command. The first command lists the kernel parameters, the second (*grep net*) filters the network relevant parameters, and the third (*less*) lets you navigate the output with arrow and space keys.

Step 3: You can change the value of a kernel parameter using the -*w* option of the *sysctl* command. Next you use this command to change the host name of *PC1* by overwriting the kernel *parameter kernel.hostname*. On *PC1*, run the commands

```
PC1$ hostname
PC1$ sudo sysctl -w kernel.hostname=X1
PC1$ hostname
```

The first and third command display the hostname.

Since the host name is displayed as part of the command prompt, when you open a new console on *PC1* you will see the newly assigned name. Give it a try!

Step 4: Change the hostname back to `*PC1’*.

Exercise 8-b: The */proc* filesystem

Another method to access the kernel parameters is through the */proc* filesystem, which contains information on every detail of the currently running system. Kernel parameters are located in the directory */proc/sys*.

Step 1: Set up a network as shown in Figure 1.4, with IPv4 addresses configured as in Table 1.1 (see Part 3).

Step 2: Explore the directory */proc/sys/net/ipv4* which contains one file for each kernel parameter for the IPv4 configuration. The name of a file corresponds to a kernel parameter and the contents of the file is the assigned value.

Step 3: The kernel parameter *icmp_echo_ignore_all* in directory */proc/sys/net/ipv4* specifies whether a host replies to a ping, or not. A value *icmp_echo_ignore_all=0* means that the host will reply with an an ICMP Echo Reply message, and *icmp_echo_ignore_all=1* means that it will not reply.

On *PC1*, access the value of the kernel parameter with the command

```
PC1$ cat /proc/sys/net/ipv4/icmp_echo_ignore_all
```

The value will be 0.

Step 4: Verify that *PC1* replies to a *ping*. On *PC2*, issue a ping to *PC1* with

```
PC2$ ping -c 3 10.0.1.11
```

Step 5: Now, change the value of *icmp_echo_ignore* on *PC1* from 0 to 1. This can be done by editing the file with Mousepad with the command

```
PC1$ sudo mousepad /proc/sys/net/ipv4/icmp_echo_ignore_all
```

A more direct method to write the value 1 in the file is with the command

```
PC1$ echo '1' | sudo tee -a /proc/sys/net/ipv4/icmp_echo_ignore_all
```

Step 6: Now go back to *PC2* and issue another ping to *PC1* with the command

```
PC2$ ping -c 10.0.1.11
```

Verify that the ping is not successful.

Step 7: On *PC1*, reset the value of *icmp_echo_ignore* to the original value with

```
PC1$ echo '0' | sudo tee -a /proc/sys/net/ipv4/icmp_echo_ignore_all
```

Exercise 8-c. Linux configuration files

The above methods for changing the network configuration are only temporary. The changes disappear when the system is rebooted. Making changes to the network configuration permanent requires modifications of configuration files. When a Linux system is started it initializes the values of the kernel parameters from such configuration files.

The structure of configuration files in Linux has been undergoing many changes. In particular, the most recent method for the configuration of network parameters (using the netplan tool) involves an additional layer of abstraction that is not always intuitive. The configuration files below will generally work or can be enabled on recent versions of Linux.

> **Different Linux distributions, different configuration files**
> Configuration files can be fundamentally different across different Linux distributions, e.g., Fedora Linux and Debian Linux. Sometimes, the organization of configuration files changes between different releases of the same Linux distribution.
>
> **Again: Live Ubuntu**
> Since the PCs in the Internet Lab are running a “Live” version of Ubuntu, changes to configuration files disappear when the system is rebooted.

`/etc/sysctl.conf`

This file specifies kernel options, including those related to the network configuration, e.g., whether IPv6 is enabled or not. Have a look at its content (using the *less* command).

`/etc/sysctl.d/`

This is a directory, which contains additional configuration files.

`/etc/hosts`

This file specifies the mapping between symbolic names and IP addresses.

`/etc/hostname`

This file specifies the host name of the local system.

Lab Questions/Report

1. Provide the total number of number of kernel parameters saved in Exercise 8-a, as well as the number of kernel parameters related to network configuration.
2. Which files must be edited to change the name of a Linux PC, e.g., from `*PC1’* to `*machine1’*?
3. Which files store information that determines whether a Linux PC performs IP forwarding?

Part 9. Wireshark

Wireshark is a network protocol analyzer with a graphical user interface. Using Wireshark, you can interactively capture and examine network traffic, view summaries and get detailed information for each packet.

Exercise 9-a. Configure and run Wireshark

This exercise walks you through the steps of capturing and saving network traffic with *Wireshark*. The exercise is conducted on *PC1*. The network setup and configuration should be as at the end of Part 8.

Step 1: Set up a network as shown in Figure 1.4, with IPv4 addresses configured as in Table 1.1.

Step 2: **Start Wireshark:** You can start *Wireshark* by selecting the icon of the editor on the Activities side bar of the desktop (see Figure 1.3(b)).

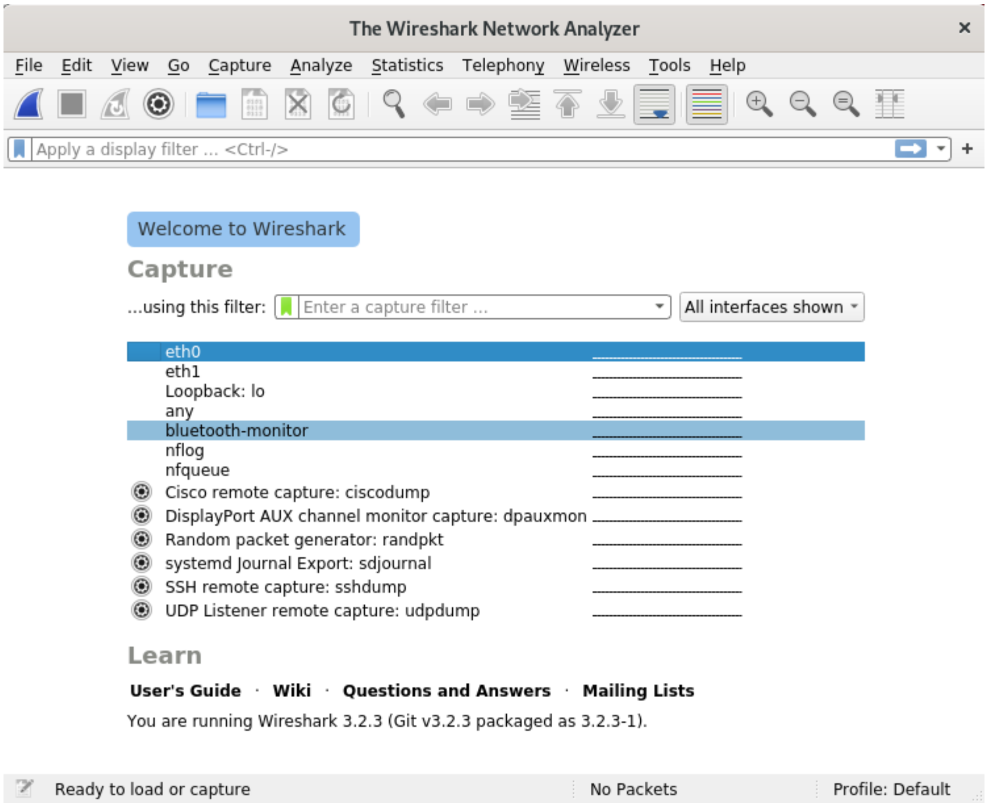
Alternatively, you can start Wireshark from the command line by typing

```
PC1$ sudo wireshark &
```

The *sudo* prefix is needed so that Wireshark can access low level system resources. The ampersand (`&') runs the command as a background process.

When *Wireshark* is started, a window as in Figure 1.9 lists the available network interfaces. Double-clicking on any one interface starts capturing outgoing and incoming packets on that interface.

Step 1: **Start packet capture:** Start capturing packets on interface *eth0* by double-clicking on *eth0* in the displayed window. Then, *Wireshark* opens a new window as shown in Figure 1.10 and starts to capture packets.
The *Wireshark* window in Figure 1.10 has three panes. The first pane, the packet list, displays a summary of the captured packets with one line for each captured packet. One of these packets can be highlighted, by clicking on the corresponding line. Then, the details of the packet headers of the highlighted packet are displayed in the second pane. Packet headers can be expanded and hidden. The third pane displays the hexadecimal and ASCII representation of the highlighted packet.

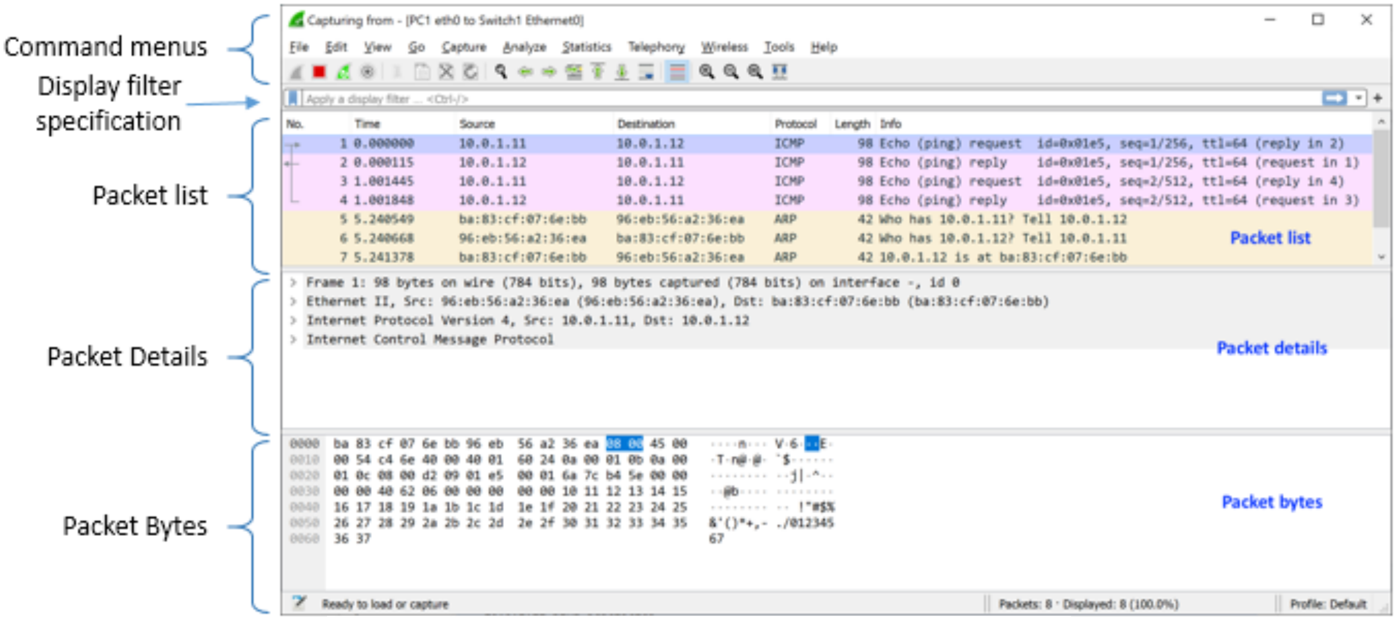

The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Welcome to Wireshark
Capture
…using this filter:
Enter a capture filter …
All interfaces shown
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists

```
eth0
eth1
Loopback: lo
any
bluetooth-monitor
nflog
nfqueue
Cisco remote capture: ciscodump
DisplayPort AUX channel monitor capture: dpauxmon
Random packet generator: randpkt
systemd Journal Export: sdjournal
SSH remote capture: sshdump
UDP Listener remote capture: udpdump
```

You are running Wireshark 3.2.3 (Git v3.2.3 packaged as 3.2.3-1).

Ready to load or capture | No Packets | Profile: Default

Figure 1.9. Welcome window of *Wireshark*.


Command menus
Display filter specification
Packet list
Packet Details
Packet Bytes
Capturing from - [PC1 eth0 to Switch1 Ethernet0]
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Packet list
Packet details
Packet bytes

```
No.   Time       Source             Destination        Protocol Length Info
    1 0.000000   10.0.1.11          10.0.1.12          ICMP       98 Echo (ping) request  id=0x01e5, seq=1/256, ttl=64 (reply in 2)
    2 0.000115   10.0.1.12          10.0.1.11          ICMP       98 Echo (ping) reply    id=0x01e5, seq=1/256, ttl=64 (request in 1)
    3 1.001445   10.0.1.11          10.0.1.12          ICMP       98 Echo (ping) request  id=0x01e5, seq=2/512, ttl=64 (reply in 4)
    4 1.001848   10.0.1.12          10.0.1.11          ICMP       98 Echo (ping) reply    id=0x01e5, seq=2/512, ttl=64 (request in 3)
    5 5.240549   ba:83:cf:07:6e:bb  96:eb:56:a2:36:ea  ARP        42 Who has 10.0.1.11? Tell 10.0.1.12
    6 5.240668   96:eb:56:a2:36:ea  ba:83:cf:07:6e:bb  ARP        42 Who has 10.0.1.12? Tell 10.0.1.11
    7 5.241378   ba:83:cf:07:6e:bb  96:eb:56:a2:36:ea  ARP        42 10.0.1.12 is at ba:83:cf:07:6e:bb
```

```
> Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface -, id 0
> Ethernet II, Src: 96:eb:56:a2:36:ea (96:eb:56:a2:36:ea), Dst: ba:83:cf:07:6e:bb (ba:83:cf:07:6e:bb)
> Internet Protocol Version 4, Src: 10.0.1.11, Dst: 10.0.1.12
> Internet Control Message Protocol
```

```
0000  ba 83 cf 07 6e bb 96 eb  56 a2 36 ea 08 00 45 00   ····n··· V·6···E·
0010  00 54 c4 6e 40 00 40 01  60 24 0a 00 01 0b 0a 00   ·T·n@·@· `$······
0020  01 0c 08 00 d2 09 01 e5  00 01 6a 7c b4 5e 00 00   ········ ··j|·^··
0030  00 00 40 62 06 00 00 00  00 00 10 11 12 13 14 15   ··@b···· ········
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ········ ·· !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,- ./012345
0060  36 37                                              67
```

Ready to load or capture | Packets: 8 · Displayed: 8 (100.0%) | Profile: Default

Figure 1.10. Wireshark window.

Step 3: **Generate traffic:** In a terminal window on *PC1*, run a ping command to *PC2*.

```
PC1$ ping -c 2 10.0.1.12
```

Observe the output in the Wireshark window. Take a snapshot of the Wireshark window.

Click and highlight a captured ICMP ping packet in the Wireshark window in the "Packet list" pane, and view the headers of the captured packet in the "Packet Details" pane.

Step 4: **Stop the traffic capture:** Click the *Stop* icon (🟥) in the window to stop the packet capture. (Selecting the *Start* icon (▲) resumes the capture.).
Do not close Wireshark, you will need the captured traffic in the next exercise.

> **Note to ECE461 students (Sep 2021)**
>
> The PCs seem to have many background processes running that generate extraneous traffic. We plan to have a revised install Ubuntu later on that fixes this issue.
> For now, you can disable the traffic with the following commands.
>
> ```
> $ sudo service isisd stop
> $ sudo service ospfd stop
> $ sudo service pimd stop
> $ sudo service zebrad stop
> ```

Lab Questions/Report

1. Include the snapshot from Step 3.
2. Describe the packets that are shown in the snapshot.

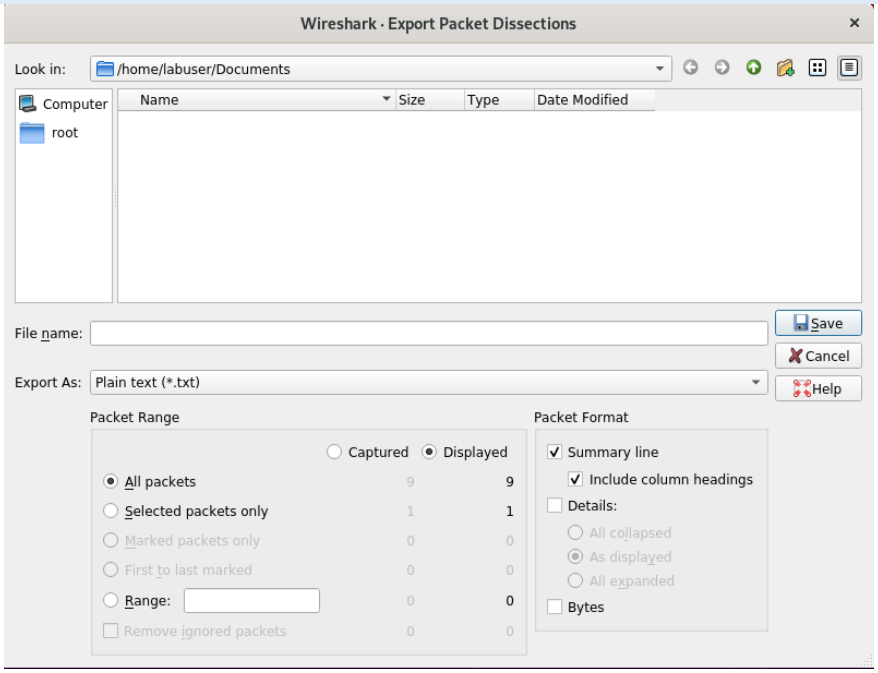
Exercise 9-b. Save captured traffic

Here you save the traffic that was captured in the previous exercise to a file. When saving the captured traffic, consider two options:

1. **Plain text file** (*.txt*): Saving captured traffic as a plain text file is useful when you want to include captured traffic in your lab report. Select "File→ Export Packet Dissections" and then select "As Plain Text…". This displays a window as shown in Figure 1.11. By default, all displayed packets will be saved with detailed information on each packet. By unselecting "Packet details", only a summary line is saved for each packet.
2. **Packet capture format (***.pcapng***)**: This is a binary file format that saves all details of the captured traffic. When saving data in this format to a file, you can later open the file with *Wireshark* for further analysis of the captured traffic. To save packets in this format select "File→ Save As…" and select the *pcapng* format. If you later open a file in packet capture format, select "File→Open" in *Wireshark*.

Saving Wireshark data

Unless specifically requested to do so, do not include “Packet details” in lab reports. This helps with keeping lab reports reasonably short. If detailed information is required, you will be asked to save details of the captured traffic.



Wireshark · Export Packet Dissections

Look in: /home/labuser/Documents

Name	Size	Type	Date Modified

File name:

Export As: Plain text (*.txt)

Packet Range

	○ Captured	● Displayed
● All packets	9	9
○ Selected packets only	1	1
○ Marked packets only	0	0
○ First to last marked	0	0
○ Range:	0	0
□ Remove ignored packets	0	0

Packet Format

☑ Summary line
☑ Include column headings
□ Details:
○ All collapsed
● As displayed
○ All expanded
□ Bytes

Figure 1.11. Saving captured traffic as plain text file in Wireshark.

Step 1: **Save captured traffic**: Save the captured traffic from Exercise 9-a as a plain text file (.*txt*) twice, once with and once without “Packet details” selected. Inspect the two files to get a sense of the information saved. Then, save the captured traffic in packet capture format **(**.*pcapng***)**.

Step 2: **Open a .pcapng file**: Quit and re-start *Wireshark*. Open the .*pcapng* file from Step 1 and verify that *Wireshark* displays the complete set of captured packets.

Lab Questions/Report

1. Provide the saved plain text file from Step 1, where “Packet details” was not selected.

Using Wireshark in future labs

In the following labs, the instructions for capturing traffic will have the form *“Capture the traffic between PC1 and PC2 with Wireshark.”* With these instructions, you need to run Wireshark on either *PC1* or *PC2* using the instructions above.

Exercise 9-c. Display filters in Wireshark

Display filters in *Wireshark* are used to display selected subsets of captured traffic. Refer to Appendix A for the syntax of display filters. Detailed information on filter capabilities and options of *Wireshark* available in the help tab in Wireshark (Select "Help→Manual Pages→Wireshark Filter).

This exercise contains applications of the display filters.

Step 1: Continue with the topology and IP addresses from the previous exercise.

Step 2: Start *Wireshark* on *PC1*, and select a traffic capture on interface *eth0*. Then click on the icon (▲) to start capturing packets.

Step 3: In a console on *PC1*, run another ping command to *PC2*.

```
PC1$ ping -c 2 10.0.1.12
```

Step 4: On *PC1*, use the *secure shell (ssh)* utility to connect to *PC2* by using the following command, and login as *labuser:labuser*.

```
PC1$ ssh labuser@10.0.1.12
```

The secure shell protocol is a cryptographic protocol for remote login over an unsecure network.

- Once you have successfully logged into *PC2*, enter the command `ls`, which displays the current directory of *PC2*.
- Terminate the remote login on *PC2* by typing the command `exit.`

Step 5: Stop the traffic capture (🟥) in the *Wireshark* window. Save the captured traffic as a *.pcapng* file. (Do not close *Wireshark* yet.)

Step 6: **Set a display filter**: You see that Wireshark has captured many packets during Step 4. When you are only interested in certain types of packets or protocols, you can reduce the number of displayed packets by setting a display filter.

Display filters are typed in the text field below the row of icons (with initial display "Apply a display filter"). The syntax for display filters and examples are given in Appendix A. Clicking on the ribbon symbol (🔖) which is located next to the text field shows examples of display filters. Use display filters to selectively display the following packet types:

- Only ARP packets,
- only ICMP packets,
- only IPv4 packets that are sent from IP address 10.0.1.11,
- only TCP packets,
- all TCP packets that arrive from TCP port 22,
- only packets of the secure shell protocol.

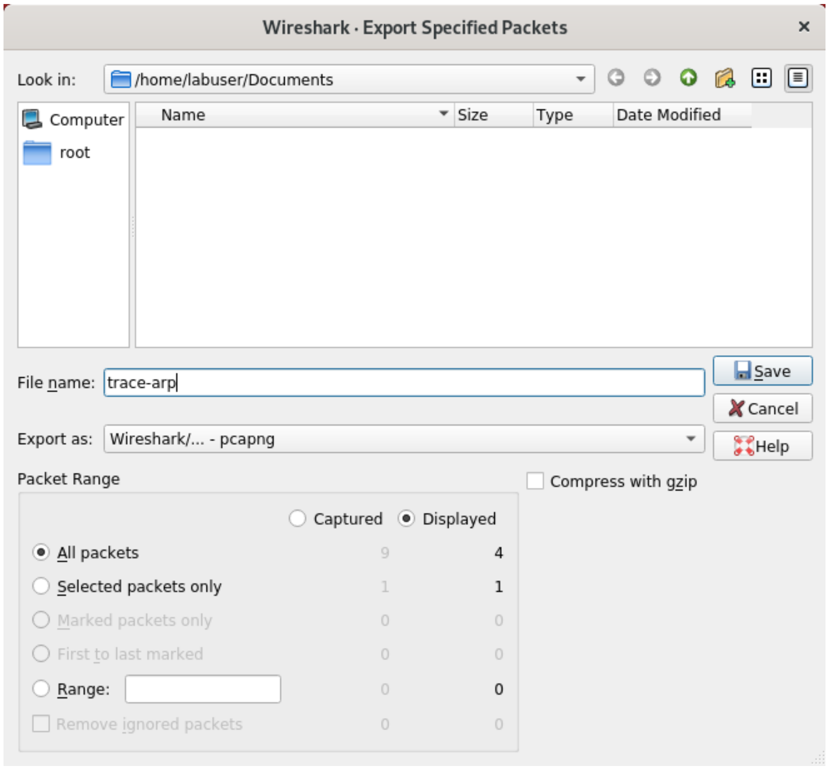
For each display filter, take a screen snapshot that shows typed display filter and the packet list. If the packet list does not fit into a single screen snapshot, only show the first screen.

Step 7: **Save filtered traffic**: Stop the traffic capture.

- **Save plain text file:** Select "File→ Export Packet Dissections" and then select "As Plain Text…". This displays a window as shown in Figure 1.11. Unselect "Packet details" and save the data.

- **Save *pcapng* file:** Select "File → Export Specified Packets". This displays a window as shown in Figure 1.12. Select "All packets" and "Displayed" and select ".pcapng" as file type and save to a file.

Step 8: Verify that the correct data has been saved by opening the .txt file with your favorite application, and opening the .pcapng file in Wireshark.



Wireshark · Export Specified Packets

Look in: /home/labuser/Documents

File name: trace-arp

Export as: Wireshark/... - pcapng

Packet Range

☐ Compress with gzip

	○ Captured	● Displayed
● All packets	9	4
○ Selected packets only	1	1
○ Marked packets only	0	0
○ First to last marked	0	0
○ Range:	0	0
☐ Remove ignored packets	0	0

Figure 1.12. Saving filtered traffic as .pcapng file in Wireshark.

Lab Questions/Report

1. Provide the screen snapshots and the display filters from Step 6.

Part 10. A Network Configuration with PCs and Cisco routers

This is the grand finale of Lab 1, where you put all pieces of this lab together. You set up a topology with routers and PCs (Figure 1.13), configure IP addresses, and capture traffic in the topology with Wireshark.

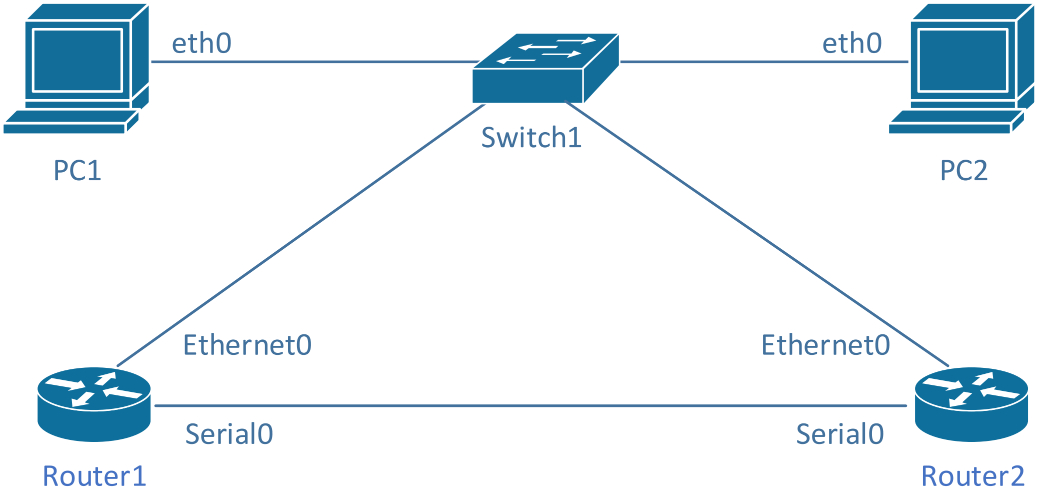


Figure 1.13. Network configuration for the grand finale of Lab 1.

This part of the lab serves as a checkpoint if you are ready to move on to Lab 2. The instructions in the following exercises use the same level of detail (or lack thereof) that you will find in the upcoming labs. If you are stuck with some instructions, e.g., how to configure an IP address, refer to the detailed instructions in earlier parts of this lab.

Exercise 10-a. Creating and testing a network configuration

Step 1: Setup of topology. Create a network topology as shown in Figure 1.13, where two PCs (*PC1*, *PC2*) and two routers (*Router1*, Rotuer2) are connected to an Ethernet switch. In addition, the routers are connected by a serial WAN link.

Step 2: Configure IPv4 addresses. Configure the IP addresses of the Ethernet0 and *Serial0* interfaces of *Router1* and *Router2* as shown in Table 1.1 (Exercise 3-a) and configure the *eth0* interfaces of the PCs as shown in Table 1.4 (Exercise 5-a).

Step 3: Start traffic capture: Start a traffic capture for the traffic between *PC1* (*eth0*) and the Ethernet switch.

Step 4: Generate traffic: Open a console window on *PC1* and issue ping commands to the other devices in the topology:

```
PC1$ ping -c2 10.0.1.12
PC1$ ping -c2 10.0.1.1
PC1$ ping -c2 10.0.1.2
PC1$ ping -c2 10.0.2.1
```

All ping commands, except the last one, should be successful and you should observe the traffic between *PC1* and the other devices. Why does the last ping fail?

Step 5: On *Router1*, additionally perform the command

```
Router1# ping 10.0.2.2
```

Step 6: In *Wireshark*, set a display filter to only show ICMP packets. Take a snapshot of the pane in *Wireshark* that shows the ICMP packets from the packet list pane (see Figure 1.10).

Lab Questions/Report

1. Include the snapshot from Step 6.
2. Explain why the last ping in Step 4 fails?

Appendix A: Display Filter Expressions in Wireshark

The following tables describe frequently used display filter expressions in *Wireshark*. If you need additional expressions, go to “Analyze Display → Filter Expressions”.

ip.dst==10.0.1.2	IPv4 destination address field is 10.0.1.2
ip.src==10.0.1.2	IPv4 source address field is 10.0.1.2
ip.addr==10.0.1.2	IPv4 source or destination address field is 10.0.1.2
ip.src==10.0.1.0/24	IPv4 source address matches the network address 10.0.1.0/24
ip.dst==10.0.1.0/24	IPv4 destination address matches the network address 10.0.1.0/24
ip.addr== 10.0.1.0/24	IPv4 source or destination address matches the network address 10.0.1.0/24
ipv6.dst== fd01:2345:6789:1::1	IPv6 destination address field is *fd01:2345:6789:1::1*
ipv6.src== fd01:2345:6789:1::1	IPv6 source address field is *fd01:2345:6789:1::1*
ipv6.addr== fd01:2345:6789:1::1	IPv6 source or destination address field is *fd01:2345:6789:1::1*
ipv6.src== fd01:2345:6789:1::1/64	IPv6 source address matches the network address *fd01:2345:6789:1::1/64*
ipv6.dst== fd01:2345:6789:1::1/64	IPv6 destination address matches the network address *fd01:2345:6789:1::1/64*
ipv6.addr== fd01:2345:6789:1::1/64	IPv6 source or destination address matches the network address *fd01:2345:6789:1::1/64*

Table 1.4. Display filter expressions in *Wireshark*.

tcp.dstport == 80 or udp.dstport == 80	Destination port is 80 in TCP segment or UDP datagram
tcp.srcport==80 or udp.srcport==80	Source port is 80 in TCP segment or UDP datagram
tcp.port==80 or udp.port==80	Destination or source port is 80 in TCP segment or UDP datagram
(tcp.srcport==80 and tcp.dstport==80) or (udp.srcport==80 and udp.dstport==80)	Destination and source port is 80 in TCP segment or UDP datagram
tcp.port==80 udp.port==80	Destination or source port is 80 in TCP segment Destination or source port is 80 in UDP datagram
eth.len <= 200	Packet size is not longer than 200 bytes
icmp tcp udp ospf	IP payload is ICMP, TCP, UDP, or OSPF

`ip.proto==17`	IP protocol number is set to 17
`eth.dst==ff:ff:ff:ff:ff:ff`	Ethernet broadcast packet
`eth.dst[0]==1`	Ethernet multicast packet
`ip.dst==224.0.0.0/4`	IP multicast packet.
`ip` `arp`	Ethernet payload is IP or ARP

Table 1.5. More display filter expressions in *Wireshark*.

`tcp[0] > 4`	The first byte of the TCP header is greater than 4
`ip[2] <= f`	The third byte of the IPv4 header does not exceed 15
`udp[0:2] == 3:ff`	The first two bytes of the UDP header are equal to 1023 Note: When selecting bytes from a header, each byte is written in hexadecimal notation, and bytes are separated by a colon
`ip.hdr_len > 20`	IPv4 headers that are longer than 20 bytes, i.e., IP headers with options
`ip.frag_offset == 0`	IPv4 packets where the fragment offset field is zero, i.e., unfragmented IP packets or the first fragment of a fragmented IP packet
`tcp.flags.syn==1 or tcp.flags.fin==1`	TCP headers with the SYN flag or the FIN flag set

Table 1.6. Even more display filter expressions in *Wireshark*.

Appendix B: Linux File System and Commands

The Linux File System

Like most operating systems, Linux organizes files as a hierarchical tree of directories. Figure 1.14. shows parts of the directory hierarchy of Linux. The directory at the top of the hierarchy, denoted by `/', is called the *root directory*.

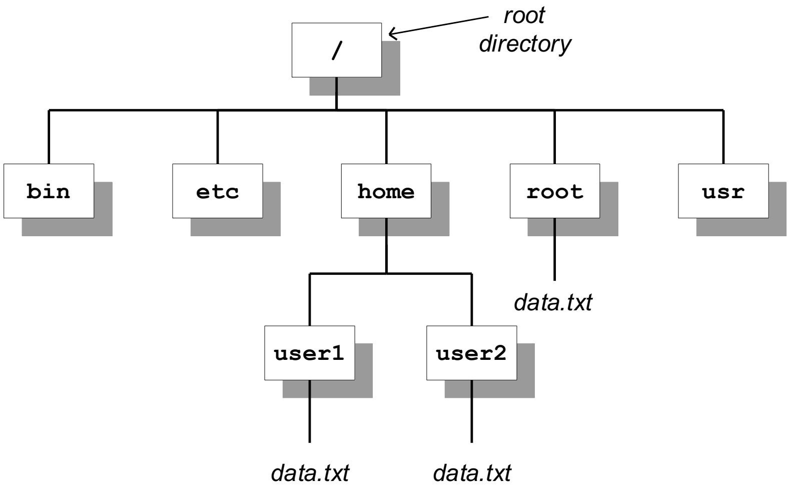


Figure 1.14. Linux directory hierarchy.

Pathnames: Each file and directory in a Linux file system is uniquely identified by a *pathname*. Pathnames can be absolute or relative. Absolute pathnames start at the root directory. The absolute pathname of the root directory is a slash (*/*). In the file hierarchy in Figure 1.14, the absolute pathname of directory *home* in the root directory is */home*, that of directory *user1* in */home* is */home/user1*, and the absolute pathname of file *data.txt* in */home/user1* is */home/user1/data.txt*.

Pathnames that do not begin with a slash are relative pathnames and are interpreted relative to the *current (working) directory*. For example, if the current directory is */home*, then the pathname *user1/data.txt* refers to the absolute pathname */home/user1/data.txt*.

When using relative pathnames, a single dot (.) denotes the *current directory* and two dots (..) denote the *parent directory*, which is the directory immediately above the current directory. With the parent directory, it is feasible to identify each file with relative pathnames. In Figure 1.23, if the current directory is */home/user1*, the relative pathname .. refers to directory */home,* the pathname *../..* refers to the root directory, and the pathname *../user2/data.txt* refers to the file */home/user2/data.txt.*

User accounts: In addition to regular user accounts, Linux systems have an administrator account, called *root*. Changes to the network configuration and other system wide settings require root privileges. In recent years, the use of the root account has been discouraged. Instead, regular user accounts can be assigned administrator privileges by adding the user to the *sudo* group. In the lab, the PCs have the account with name *labuser*, which has been added to the *sudo* group.

Home directories: Each Linux account has a *home directory*, which is located in */home*. So, */home/user1* is the home directory for an account with login *user1*. When a new terminal window is created, the current directory in the terminal window is the home directory.

File permissions: Each file and each directory has an owner. A regular user only owns the home directory and all files created by the user. The root is the owner of all other files on the system. In Linux, each file has a set of access permissions. The permissions are *read ("r"), write ("w"),* and *execute ("x"),* and give, respectively, permission to read the contents of a file, modify the file, or execute the file as a program. Permissions are set by the owner of a file. Linux specifies access permissions separately for the owner of the file, a user group which is associated the file, and the set of all users. So, the owner of a file can set the permissions so that all users can read the files, but only the owner can modify the file. The root and *sudo* users can ignore access permissions and can even change the ownership of files. Since the exercises in the Internet Lab are done from the *labuser* account, which has sudo privileges, access permissions are not important for the Internet Lab. The downside of not having to worry about access permissions is that there is no protection against accidentally deleting or corrupting files.

Linux Devices and Network Interfaces

The software abstraction through which the Linux kernel accesses networking hardware is that of a *network interface.* For example, when assigning an IP address to an Ethernet interface cards, one manipulates the configuration parameters of the network interface which represents the Ethernet card. In the Internet Lab, the names of network interfaces are *eth0* for the first Ethernet interface card and *eth1* for the second Ethernet interface card. There is a special network interface, called *loopback interface*, with name *lo*. The loopback interface is not connected to a real device, but is a virtual interface, which allows a Linux system to send messages to itself.

Linux shell and commands

The command line interface of the Linux operating system is called a *shell.* A shell is a program that interprets and executes Linux commands which are typed in a window terminal. Whenever you create a new terminal window, a shell is started. The shell displays a prompt at which the user can type commands. The prompt can be as simple as

```
%
```

or the prompt can be set to provide additional information. For example, the prompt

```
39: PC1@/root =>
```

displays the name of the computer and the current directory. Throughout this book, we use the prompt *PC1%* if we want to indicate that this is a shell prompt at *PC1*. When you type a command at the prompt, and press the enter key, the shell interprets the command, and, if it is a valid Linux command, executes the command. A shell is terminated by typing *exit* at the command prompt. If the shell is running in a terminal window, the terminal window disappears.

Next, we review some basic Linux commands that are typed in at a shell prompt. Commands in Linux have a common format: a command name, which may be followed by a set of options and arguments. For example, in the command `*ls –l data.txt*', `*ls*' is the command, `*-l*' is an option which further specifies the command, and `*data.txt'* is an argument. Options are generally preceded by a **–** (dash), and multiple options can be specified in a single command.

Since all files in Linux are organized as a tree of directories, you need to become familiar with navigating and manipulating the directory tree. The following are commands that relate to Linux directories.

Directory commands:

`pwd`

Print the absolute path of the current directory.

`cd` ***`dirpath`***
`cd`

Change the current directory to the relative or absolute pathname of the directory *`dirpath`*. If no directory is given, the command changes the current directory to the home directory. The command `‘cd /usr/bin’` changes the working directory to `/usr/bin`, the command ``` `cd ..’ ``` changes it to the parent directory, and the command `cd' without a parameter switches to the home directory.

`mkdir` ***`dirname`***

Create a new directory with name *dirname* in the current directory. For example, the command ``` `mkdir xyz’ ``` creates a subdirectory in the current directory with name *xyz*.

`rmdir` ***`dirname`***

Delete the directory *`dirname`* from the current directory. A directory cannot be deleted when it still contains files or subdirectories. Thus, before deleting a directory, all files and subdirectories must be deleted first.

Before discussing the commands to list and manipulate files, we introduce the *wildcard characters* `*' (star) and `*?'* (question mark). The wildcard character `*' matches any sequence of zero or more characters, and `*?'* matches any single character. Wildcard characters are useful to describe multiple files in a concise manner. For example, the text string *A*.txt* matches all file names that start with an *`A'* and end with *`.txt',* e.g., *ABC.txt, A.txt,* and *Ab.txt* The text string *A?.txt* matches all filenames that are two characters long and start with *`A',* e.g., *Ab.txt* and *A1.txt.*

File commands:

`ls`
`ls` ***`dirname`***

List information about files and directories in the current directory. If the command has a directory name as argument, then the command lists the files in that directory. The `ls` command has several options. The most important is ``` `ls -l’ ```, which displays extensive information about each file, including the access permissions, owner, file size, and the time when the file was last modified.
For example, `

- `‘ls /’` lists all files and directories in the root directory;
- ``` `ls AB*’ ``` lists all files and directories in the current directory that start with AB;

- `ls -l ..’ prints detailed information on each file and directory in the parent directory of the current directory.

mv *fname newfile*
mv *fname dirname*

The first command renames a file or directory with name *fname* as *newfile*. If the destination file (*newfile*) exists, then the content of the file is overwritten, and the old content of *newfile* is lost. The second command moves a file or directory with name *fname* to directory *dirname*.
For example,

- `mv data.txt text.txt’ simply renames file data.txt,
- `mv * /home/labuser’ moves all files from the current directory to directory /root (and gives an error message if the current directory is /home/labuser).

cp *fname newfile*
cp *fname* dirname

Copy the content of file *fname* to *newfile*. If a file with name *newfile* exists, the content of that file is overwritten. If the second argument is a directory, then a copy of *fname* is created in directory *dirname*.
For example,

- `cp *.txt /tmp’ creates a copy of all files that end with `.txt’ in directory /tmp.

rm *fname*

Removes a file. Once removed, the file cannot be recovered.
For example,

- `rm *’ removes all files in the current directory.

Prevent overwriting files

By default, Linux does not issue a warning when a file is overwritten or when a file is removed. With the added option *`-i’*, Linux asks for confirmation before overwriting or deleting files.

Examples:

```
$ cp -i file1 file2
```

gives a warning if file2 already exists and request confirmation to overwrite its content.

```
$ rm -i file1
```

requests to confirm deletion of file *file1*.

An important thing to have in mind is that Linux does not have an undo command that reverses the effects of a previously issued command.

Many lab experiments ask you to save the output of a command to a file. The following commands show how to redirect the output of a terminal window to a file.

Redirecting the output of programs:

cmd > fname

The output of the command *cmd* is written to file *fname*. The file is created if it doesn't already exist, and its contents is overwritten if the file exists.

- The command `ls > mlist.txt’ writes a listing of the current directory into file *mylist.txt*.

cmd >> fname

The >> operator appends the output of command *cmd* to the end of file *fname*.

- The command, `ls >> mylist.txt’ appends a listing of the current directory to file mylist.txt.

In Linux, each console window can run multiple commands at the same time. Also, it is possible to stop a command temporarily and resume it at a later time. In each terminal window, one command can be run as a *foreground process* and multiple command can be run as *background processes*. When a command is issued from the prompt, say

```
PC1$ mousepad
```

the command nano is started in the foreground. When a command is running in the foreground, no shell prompt is displayed until the command is finished. The same command can be run in the *background* by adding an & (ampersand) at the end of the command, by typing

```
PC1$ mousepad &
```

If a command is executed in the background, the shell prints a prompt for the next command without any delay. Using background commands, you can run multiple commands from a single terminal window.

You can switch a command that is running in the foreground to the background and vice-versa. Switching a command from the foreground to the background is done as follows:

```
PC1$ mousepad
```

Then type Ctrl-z to stop the editor from running. You will see the command prompt of the shell. If you then type

```
PC1$ bg
```

the *mousepad* command resumes in the background. To switch a command from the background to the foreground, type

```
PC1$ jobs
```

The command *jobs* lists all commands that are currently running in the background or are stopped, e.g., with `Ctrl-z`*.* The command

```
PC1$ %1
```

resumes the first command from the list in the foreground. The following are a set of Linux commands that control the execution of commands.

Control of commands:

`Ctrl-c`

Terminate the command running in the foreground.

`Ctrl-z`

Stop the command running in the foreground.

***`cmd`* `&`**

Execute the command *cmd* in the background.

`jobs`

List all background and stopped commands of the current user, and assign a number to each command.

`fg` *`%n`*
`fg`

Resume the *n*-th command of the user (as listed by the command jobs). If no number is given, the command refers to the command that was last running, started, or stopped.

`bg` *`%n`*
`bg`

Resume the *n*-th command of the user that is stopped or running in the background. If no number is given the command refers to the command that was last running, started, or stopped.

`kill` *`%n`*

Terminatesthe *n*-th command (listed by `jobs').