

Based on
Mastering Networks - An Internet Lab Manual
by Jörg Liebeherr and Magda Al Zarki

Adapted for
'Labo Computernetwerken'
by Johan Bergs, Nicolas Letor, Michael Voorhaen and Kurt Smolderen

Completed by
Josse Coen Armin Halilovic Jonas Vanden Branden Group 2

March 10, 2016

Lab 1

Introduction to the Internet Lab

What you will learn in this lab:

- Overview of the equipment
- Saving your data
- Navigating your way around Linux
- Working with protocol analysers: `tcpdump`, Wireshark

1.1 Prelab 1

Basic Linux commands

Before entering the lab, you should familiarise yourself with some basic Linux commands and the Wireshark network analyser tool.

Man Pages

The PCs run the Linux operating system, a Unix-like 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://manpages.ubuntu.com/>

For each of the following commands, type the name of the command as a search term. The search will return the appropriate man page.

Read the man pages of the following commands for the operating system version "trusty 14.04 LTS":

`man, pwd, ls, more, mv, cp, rm, mkdir, rmdir, chmod, kill, ping, tcpdump`

Wireshark

The man page for Wireshark, a network analyser tool, can be found on every lab machine. You can also read about the Wireshark network analyzer at the website <https://www.wireshark.org/docs/>. Read the introduction and the manual pages of Wireshark.

Prelab Questions

Question 1)

What will happen if you type `man man` in Linux?

The manual page for the command `man` will be displayed.

Question 2)

How can you use the command `ls` to find out about the size of file `/etc/fstab`?

`ls /etc/fstab -l` displays the size of the file in bytes. The flag `-h` will put this in human readable format.

Question 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?

File 2 will be overwritten by file 1. Option `-i` will prompt before overwriting the file.

Question 4)

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

`cp mydata labdata/`

Question 5)

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

`cp -r mydirectory/* newdirectory/`

Question 6)

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

`rm *` removes all files (no directories) in the working directory.

Question 7)

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

`rm -r /mydirectory/*`

1.2 Lab 1

In Lab 1, you will acquaint yourself with the equipment of the Internet Lab, the Linux operating system, and some traffic measurement tools.

Part 1. Getting Started

Lab reservations

You need to book labs in advance at: <https://student.mosaic.uantwerpen.be/telecomlabo/>. To be able to go online you need to register and accept the terms of service published here: <https://euterpe.mosaic.uantwerpen.be/laboaccess/>.

Cables

When you have booked a lab, you can ask for a box of cables in room M.G.236 (go through the server room M.G.235). The box contains a set of UTP cables in different colors and 1 cross-cable. Please return it when you are done.

Part 2. Overview of the Internet Lab Hardware

Each lab is equipped with:

- 4 PC's, each equipped with their own monitor, keyboard, mouse and external USB hub.
- 4 10BASE-T Hubs
- 1 10BASE-T/100BASE-TX Hub
- Patch panel: The interfaces of each of the PC's are connected with a patch panel, which means that it should not be necessary to change any of the network cabling directly on the PC.
- 4 Cisco 1760 routers.

Using the lab PC's for Computer Networks Lab

It is important to know that on booting the PC's all previous changes are automatically flushed from the system, returning them to their original state.

Note: Please contact one of the course instructors to recover any lost files if the PC was accidentally shut down.

You need to log in using your UA credentials. Make sure you select the "telecomlabo" option in the login screen! This will disable all network related services on the device and configure the network stack for the Lab. Not doing this might affect your lab results.

When working in the terminal either use `sudo` before each command to get root privileges.

```
|> sudo ip addr add 192.168.1.2/24 dev eth0
```

Or type in `sudo su -` at the beginning of each new terminal session to keep working with root privileges.

```
|> sudo su -
```

You will be asked for a password when using the `sudo` command. For the remainder of this course, whenever a password is requested that is not your own student password, you should use `mvkbj1n`, which is derived from the phrase "Met veel kabels bouw je ÃÃn netwerk".

Please turn off the lab PC's and screens when you are done with them.

Setting up an uplink to the internet

Each lab PC should have 3 interfaces (`eth0`, `eth1` and `internet`). `eth0` and `eth1` are meant for your lab setups, while the latter interface can be used to set up an uplink to the Internet. There is an icon "Enable Internet Access" on the desktop, double-click it and follow the instructions. You will be asked to enter your UA credentials.

Note: to go online you will have to first register at: <http://euterpe.mosaic.uantwerpen.be/laboaccess>

Removing the uplink to the internet

When performing any of the lab tests it is recommended that they are not connected to the internet. This makes sure that the configurations made for the Internet uplink do not interfere with the lab setup. To remove the uplink double-click on the icon 'Disable Internet Access' on the desktop.

Part 3. Saving files

Each PC is configured with a USB hub that is placed next to the monitor. You can connect a USB stick to the computer and copy your results and traces to it. You can either do this using the file manager or from the command line. The USB sticks are automatically mounted in the `/media/` directory.

Make sure to unmount your stick before removing to avoid any data loss, either via the icon on the desktop or on the command line using the `umount` command.

```
| umount /media/usbstick/
```

We would like to stress again that, on booting the PC's, all previous changes are automatically flushed from the system, returning them to their original state.

You can use the simple single segment network that we will be discussing in Part 4 to copy files from one computer to another over the network. You can use tools like `scp` or `rsync` to do this. Please check the man pages of the programs on how to use them.

If Internet access is working you can also copy your files to an external server:

```
|> scp -r /root/lab1/ user@student.mosaic.uantwerpen.be:labocomputernetwerken/
```

The above example will copy all files and directories from the `/root/lab1/` directory to the `labocomputernetwerken` directory in the user's home folder on `student.mosaic.uantwerpen.be`.

Part 4. Setting up your first network

We will now construct the simple single segment network shown in Figure 1.1 that we will be using throughout the remainder of this lab.

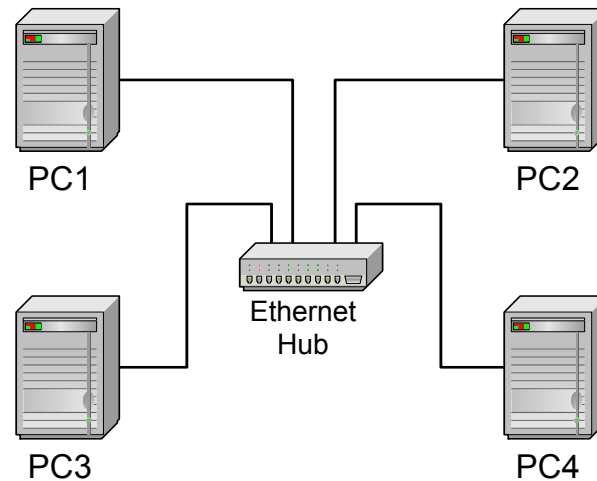


Figure 1.1: Network configuration for Lab 1.

- All four Linux PCs are connected to a single Ethernet segment via a single hub as shown in Figure 1.1.
- IP addresses for the Linux PCs are configured as in Table 1.1

Linux PC	IP Addresses of Ethernet Interface <i>eth0</i>
PC1	10.0.1.11/24
PC2	10.0.1.12/24
PC3	10.0.1.13/24
PC3	10.0.1.14/24

Table 1.1: IPv4 addresses for Lab 1

- The notation 10.0.1.11/24 means that the IP 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 this netmask, all hosts are on the 10.0.1.0/24 network.

Mapping of the patch panel ports to the PC interfaces

1. Use the mapping in Table 1.2 to figure out which interface of which PC is which port on the patch panel. Connect the PC's to the hub using Ethernet cables.
2. Configure *eth0* for each of the PCs. e.g. for PC1 use the following command.

```
| PC1% ifconfig eth0 10.0.1.11 netmask 255.255.255.0 broadcast 10.0.1.255 up
```

As an alternative, you can use the *ip* command:

Linux PC	Interface	Patch panel port
PC1	<i>eth0</i>	1
PC1	<i>eth1</i>	2
PC2	<i>eth0</i>	3
PC2	<i>eth1</i>	4
PC3	<i>eth0</i>	5
PC3	<i>eth1</i>	6
PC4	<i>eth0</i>	7
PC4	<i>eth1</i>	8

Table 1.2: IP addresses for Lab 1

```
PC1% ip addr add 10.0.1.11/24 dev eth0
PC1% ip link set dev eth0 up
```

3. Test connectivity by using ping, e.g.

```
PC1% ping -c 5 10.0.1.11
```

Part 5. Locating Configuration Files in Linux

Linux has numerous configuration files which set the environment variables of the operating system. For example, if you want to set up your Linux PC as an IP router, you merely need to change a single line in one of the configuration files. Studying configuration files also provides a way of learning what network configuration options are available to you.

! If you are unable to revert some of the changes you made, try rebooting the PC's as they will be reverted to their original settings.

! Configuration files are fundamentally different across different versions of Unix-like operating system (e.g., AIX, Solaris, Linux, FreeBSD). Sometimes the structure of configuration files changes between releases of the same Unix version. For example, the configuration files of different Linux distributions, such as, Ubuntu, Fedora, OpenSuSE and Slackware, are quite different. Furthermore, the configuration files between different versions of the same Linux distribution can have significant differences.

The file `/etc/network/interfaces` contains settings on how the interfaces should be configured when starting the network service on Debian-based Linux distributions. You will notice that there is not much in this file as our lab PCs are preconfigured NOT to bring up their interfaces. You can read the man page to find out more on this file. It will not be used in the remainder of this course.

```
| > man interfaces
```

You will not find much in `sysctl.conf` either: check out the `sysctl` tool and briefly browse `/proc/sys/net`. You will see that there are several parameters that allow you to tweak the behavior of the linux network stack.

```
| > ls /proc/sys/net
| > man sysctl
```

The `/etc/hosts` file contains a mapping of hostnames to IP addresses.

```
| > less /etc/hosts
```

Exercise 5. Network configuration files

Question 5.1)

Which file(s) must be edited to change the name of a Linux PC (e.g. from PC1 to machine1)?

[/proc/sys/kernel/hostname](#)

Question 5.2)

Which file(s) include information that determines whether a Linux PC performs IP forwarding?

[/proc/sys/net/ipv4/ip_forward](#)

Part 6. Using Ping

One of the most basic, but also most effective tools to debug IP networks is the `ping` command. The `ping` command tests whether another host or router on the Internet is reachable. The `ping` command sends an ICMP Echo Request datagram to an interface, and expects an ICMP Echo Reply datagram in return.

Note:

- On Linux systems, `ping` continues to send packets until you interrupt the command with the `Ctrl-c` keys.
- When using `ping` on the Linux PCs, we recommend to always send at least two ICMP Echo Request packets. We have observed that in some occasions, the first ICMP Echo Request may be dropped at the receiver.

Exercise 6. Issuing ping commands

1. From PC1, send 5 ping messages (using the `-c` option) to PC2. Save the output.

```
| PC1% ping -c 5 10.0.1.12
```

2. On PC2, issue a ping to the IP address of PC1. Also, issue a `ping` command to the *loopback* interface, 127.0.0.1. Limit the number of pings to 5. Save the output.

Question 6.1)

Include the output you saved in the exercise.

From PC1, send 5 ping messages (using the `-c` option) to PC2. Save the output:

```
1 student@lab2pc1:~$ ping -c 5 10.0.1.12
   PING 10.0.1.12 (10.0.1.12) 56(84) bytes of data:
3  64 bytes from 10.0.1.12: icmp_seq=1 ttl=64 time=0.623 ms
   64 bytes from 10.0.1.12: icmp_seq=2 ttl=64 time=0.610 ms
5  64 bytes from 10.0.1.12: icmp_seq=3 ttl=64 time=0.554 ms
   64 bytes from 10.0.1.12: icmp_seq=4 ttl=64 time=0.618 ms
7  64 bytes from 10.0.1.12: icmp_seq=5 ttl=64 time=0.585 ms

9  --- 10.0.1.12 ping statistics ---
   5 packets transmitted, 5 received, 0% packet loss, time 3997ms
11  rtt min/avg/max/mdev = 0.554/0.598/0.623/0.025 ms
```

On PC2, issue a ping to the IP address of PC1:

```
1 student@lab2pc1:~$ ping -c 5 10.0.1.11
   PING 10.0.1.11 (10.0.1.11) 56(84) bytes of data:
3  64 bytes from 10.0.1.11: icmp_seq=1 ttl=64 time=0.614 ms
   64 bytes from 10.0.1.11: icmp_seq=2 ttl=64 time=0.594 ms
5  64 bytes from 10.0.1.11: icmp_seq=3 ttl=64 time=0.603 ms
   64 bytes from 10.0.1.11: icmp_seq=4 ttl=64 time=0.596 ms
7  64 bytes from 10.0.1.11: icmp_seq=5 ttl=64 time=0.602 ms

9  --- 10.0.1.11 ping statistics ---
   5 packets transmitted, 5 received, 0% packet loss, time 3999ms
11  rtt min/avg/max/mdev = 0.594/0.601/0.614/0.031 ms
```

Also, issue a ping command to the loopback interface, 127.0.0.1:

```

1 student@lab2pc1:~$ ping -c 5 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data:
3 64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.032 ms
  64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.031 ms
5 64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.024 ms
  64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.032 ms
7 64 bytes from 127.0.0.1: icmp_seq=5 ttl=64 time=0.030 ms

9  — 127.0.0.1 ping statistics —
  5 packets transmitted, 5 received, 0% packet loss, time 3996ms
11 rtt min/avg/max/mdev = 0.024/0.029/0.032/0.007 ms

```

Question 6.2)

Explain the difference between pinging the local Ethernet interface and the *loopback* interface. Specifically, on PC1, what is the difference between typing `ping 10.0.1.11` and `ping 127.0.0.1`? (This is a conceptual question on the role of the *loopback* interface. The response to the ping command does not provide you with the answer to this question.) Hint: Try using `tcpdump` or Wireshark, which are explained in the next section to verify your answer. Hint: Is this handled by software (the linux routing stack), hardware (the network card, hubs, switches, routers) or both?

```

1 student@lab2pc1:~$ ping -c 5 10.0.1.11
PING 10.0.1.11 (10.0.1.11) 56(84) bytes of data:
3 64 bytes from 10.0.1.11: icmp_seq=1 ttl=64 time=0.032 ms
  64 bytes from 10.0.1.11: icmp_seq=2 ttl=64 time=0.017 ms
5 64 bytes from 10.0.1.11: icmp_seq=3 ttl=64 time=0.025 ms
  64 bytes from 10.0.1.11: icmp_seq=4 ttl=64 time=0.030 ms
7 64 bytes from 10.0.1.11: icmp_seq=5 ttl=64 time=0.036 ms

9  — 10.0.1.11 ping statistics —
  5 packets transmitted, 5 received, 0% packet loss, time 3996ms
11 rtt min/avg/max/mdev = 0.017/0.028/0.036/0.006 ms

```

We expected the pings to 10.0.1.11 to go through the network (eth0), unlike the pings to 127.0.0.1, but we noticed they both went through the loopback interface.

We found this on the web:

The loopback interface does not represent any actual hardware, but exists so applications running on your computer can always connect to servers on the same machine.

The local Ethernet interface is used for packets that are meant for other hosts.

The loopback interface is used by a computer to communicate to itself.

Question 6.3)

Find a host connected to the Internet. Send ping messages to a number of web servers on the Internet and collect statistics on the maximum round-trip delay of the ICMP Echo Request/Echo Reply. Try to find a host with a very long round-trip time. To avoid overloading the destination, do not send more than 10 ping packets to any destination machine. Save the output data and include it in your lab report.

```

1 Results ordered by rtt avg. The maximum we found was 312.6 by baidu.com
3 — wikipedia.org ping statistics —

```

```
5 10 packets transmitted, 10 received, 0% packet loss, time 1807ms
   rtt min/avg/max/mdev = 6.404/6.556/6.639/0.115 ms
7
   — reddit.com ping statistics —
9 10 packets transmitted, 10 received, 0% packet loss, time 1805ms
   rtt min/avg/max/mdev = 7.258/7.338/7.387/0.083 ms
11
   — google.com ping statistics —
13 10 packets transmitted, 10 received, 0% packet loss, time 1805ms
    rtt min/avg/max/mdev = 11.047/11.223/11.435/0.148 ms
15
   — gigaspeedsurfer.de ping statistics —
17 10 packets transmitted, 10 received, 0% packet loss, time 1805ms
    rtt min/avg/max/mdev = 16.853/17.242/18.096/0.374 ms
19
   — scssoftware.com ping statistics —
21 10 packets transmitted, 10 received, 0% packet loss, time 1805ms
    rtt min/avg/max/mdev = 26.872/27.194/27.891/0.399 ms
23
   — facebook.com ping statistics —
25 10 packets transmitted, 10 received, 0% packet loss, time 1805ms
    rtt min/avg/max/mdev = 97.362/97.464/97.591/0.150 ms
27
   — www.a.shifen.com ping statistics —
29 10 packets transmitted, 10 received, 0% packet loss, time 9001ms
    rtt min/avg/max/mdev = 311.690/312.642/314.052/0.866 ms
```


Part 7. Basics of `tcpdump`

`tcpdump` allows you to capture traffic on a network and display the packet headers of the captured traffic. `tcpdump` can be used to identify network problems or to monitor network activities.

Exercise 7-A. Simple `tcpdump` exercise

Use `tcpdump` to observe the network traffic that is generated by issuing `ping` commands.

If you use the `tee` or `tail` commands to simultaneously view and save the output from `tcpdump`, you need to use the `-l` option of `tcpdump`. For example:

```
> tcpdump -n -l > filename & tail -f filename  
> tcpdump -n -l | tee filename
```

Note: It may be necessary to hit `Ctrl-C` to terminate the `tcpdump` session. In some situations, it may be best to simply redirect the output of `tcpdump` straight to a file (e.g., `tcpdump > filename`) and view it afterwards with the `less` command or a text editor.

1. Switch to PC1. Start `tcpdump` so that it monitors all packets that contain the IP address of PC2, by typing

```
PC1% tcpdump -n host 10.0.1.12
```

2. Open a new window and execute

```
PC1% ping -c 1 10.0.1.12
```

3. Observe the output of `tcpdump`. Save the output to a file.

Question 7.A)

Include the saved output in your lab report. Explain the meaning of each field in the captured data.

```
1 13:55:42.315433 IP 10.0.1.11 > 10.0.1.12: ICMP echo request, id 9383, seq 1, length  
64  
13:55:42.316043 IP 10.0.1.12 > 10.0.1.11: ICMP echo reply, id 9383, seq 1, length  
64
```

Fields:

- *time*: states the time the message was sent
- *network_layer_protocol*: the protocol over which the message was sent
- *source_address*: the address of the sender

- *destination_address*: the address of the receiver
- *ICMP control_message*: the type of message: ICMP echo request/ ICMP echo reply for ping
- *id*: identifier used to match pings in a session
- *seq*: sequence number, used to match the requests with their reply
- *length*: length of the message in bytes

Exercise 7-B. Another `tcpdump` traffic capture

1. On PC1, start capturing packets using the `tcpdump -n` command.
2. Issue a ping to the non-existing IP address 111.111.111.111:

```
| PC1% ping -c 1 111.111.111.111
```

3. Issue a ping to the broadcast address 10.0.1.255 using the command:

```
| PC1% ping -c 2 -b 10.0.1.255
```

When sending pings to the broadcast address is not working, check out the `/proc/sys/net/ipv4/` directory for possible broadcast-related settings. Try enabling/disabling them if necessary (see the previous paragraph).



Explain what you did, what happened and why in the lab report. Simply stating that broadcast pings didn't work is not a correct answer.

4. Save the outputs of `ping` and `tcpdump` to a file.

Question 7.B)

Include the saved output in your lab report and interpret the results. How many of the Linux PCs responded to the broadcast ping? Why is this?

Issue a ping to the non-existing IP address 111.111.111.111:

```
1 connect: Network is unreachable\|
```

Issue a ping to the broadcast address 10.0.1.255 using the command:

```
1 14:58:25.587269 IP 10.0.1.11 > 10.0.1.255: ICMP echo request, id 9803, seq 1,
   length 64
   14:58:26.588257 IP 10.0.1.11 > 10.0.1.255: ICMP echo request, id 9803, seq 2,
   length 64
```

There was no reply to the requests because every host on the network had the `net.ipv4.icmp_echo_ignore_broadcasts` option enabled.

After disabling this with `sysctl net.ipv4.icmp_echo_ignore_broadcasts=0`, we get the following output:

```
14:59:27.536259 IP 10.0.1.11 > 10.0.1.255: ICMP echo request, id 9805, seq 1,
length 64
2 14:59:27.536820 IP 10.0.1.14 > 10.0.1.11: ICMP echo reply, id 9805, seq 1, length
64
14:59:27.536850 IP 10.0.1.13 > 10.0.1.11: ICMP echo reply, id 9805, seq 1, length
64
4 14:59:27.536860 IP 10.0.1.12 > 10.0.1.11: ICMP echo reply, id 9805, seq 1, length
64
14:59:28.535238 IP 10.0.1.11 > 10.0.1.255: ICMP echo request, id 9805, seq 2,
length 64
6 14:59:28.535809 IP 10.0.1.14 > 10.0.1.11: ICMP echo reply, id 9805, seq 2, length
64
14:59:28.535822 IP 10.0.1.13 > 10.0.1.11: ICMP echo reply, id 9805, seq 2, length
64
8 14:59:28.535825 IP 10.0.1.12 > 10.0.1.11: ICMP echo reply, id 9805, seq 2, length
64
```

The three other hosts have now responded.

Part 8. Basics of 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 8. Running wireshark

This exercise walks you through the steps of capturing and saving network traffic with Wireshark. The exercise is conducted on PC1.



Figure 1.2: Wireshark Main Window.

1. Starting Wireshark: On PC1, start Wireshark by typing

```
| PC1% wireshark
```

This displays the Wireshark main window on your desktop as shown in Figure 1.2.



Do not forget to start Wireshark with root permissions using `sudo` if you are not root already.

2. Selecting the capture options: Use the instructions in Figure 1.3 to set the options of Wireshark in preparation for capturing traffic. Use the same options in other labs, whenever Wireshark is started.

Selecting capture preferences in Wireshark

- a) From the main window, select “Capture:Start”.
- b) This displays the following “Capture Preferences” window:

The screenshot shows the "Capture Preferences" window in Wireshark. It is divided into several sections:

- Capture:**
 - Interface: `eth0`
 - IP address: `192.168.1.1, fe80::20e:cff:fe6e:17ad`
 - Link-layer header type: `Ethernet`
 - ☒ Capture packets in promiscuous mode
 - ☐ Capture packets in pcap-ng format (experimental)
 - ☐ Limit each packet to `1` bytes
 - Capture Filter: (empty)
- Capture File(s):**
 - File: (empty) [Browse...]
 - ☐ Use multiple files
 - ☒ Next file every `1` megabyte(s)
 - ☐ Next file every `1` minute(s)
 - ☒ Ring buffer with `2` files
 - ☐ Stop capture after `1` file(s)
- Stop Capture ...:**
 - ☐ ... after `1` packet(s)
 - ☐ ... after `1` megabyte(s)
 - ☐ ... after `1` minute(s)
- Display Options:**
 - ☒ Update list of packets in real time
 - ☒ Automatic scrolling in live capture
 - ☒ Hide capture info dialog
- Name Resolution:**
 - ☒ Enable MAC name resolution
 - ☐ Enable network name resolution
 - ☒ Enable transport name resolution

Buttons at the bottom: `Help`, `Cancel`, `Start`.

- Select `eth0` in “Interface”.
- Select “Capture packets in promiscuous mode”.
- Select “Update list of packets in real time”.
- Select “Automatic scrolling in live capture”.
- Unselect “Enable MAC name resolution”.
- Unselect “Enable network name resolution”.
- Unselect “Enable transport name resolution”.

Figure 1.3: General capture settings for Wireshark

- Starting the traffic capture: Start the packet capture by clicking “OK” in the “Capture Preferences” window.
- Generating traffic: In a separate window on PC1, execute a `ping` command to PC3.

```
PC1% ping -c 2 10.0.1.13
```

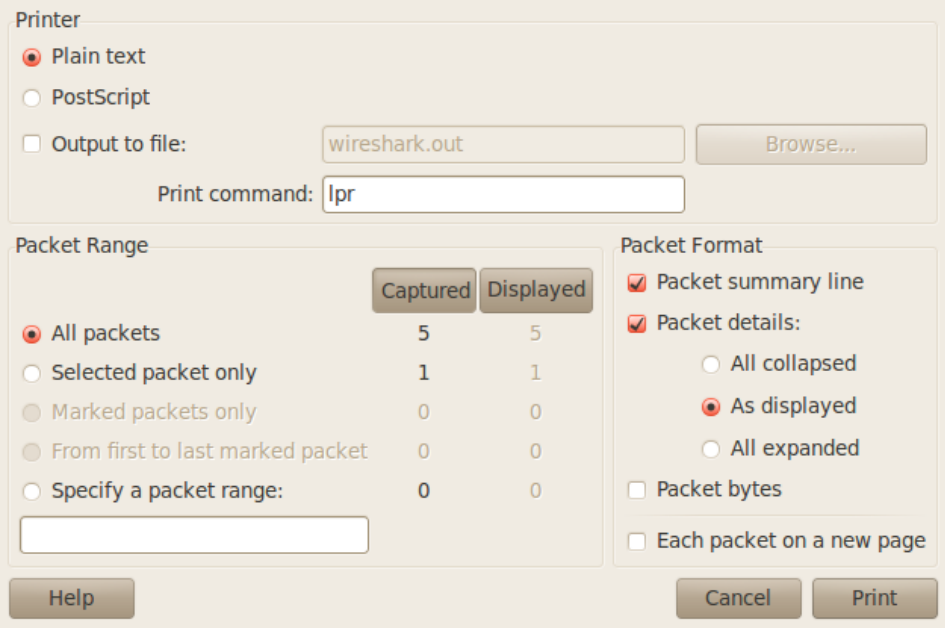
Observe the output in Wireshark’s main window. Click and highlight a captured packet in the Wireshark window, and view the headers of the captured traffic.

5. Stopping the traffic capture: Click “Stop” in the window “Ethernet Capture”.
6. Saving captured traffic: Save the results of the captured traffic as a plain text file. Use the instructions in Figure 1.4 in order to configure the print options..

Selecting print options for saving captured traffic to plain text files

a) From the main window, select “File:Print”.

b) This displays the following “Printer options” window:



	Captured	Displayed
<input checked="" type="radio"/> All packets	5	5
<input type="radio"/> Selected packet only	1	1
<input type="radio"/> Marked packets only	0	0
<input type="radio"/> From first to last marked packet	0	0
<input type="radio"/> Specify a packet range:	0	0

- Select the format “PlainText”.
- Select the “File” checkbox and type the filename in the field next to the “File” button.
- Select “Print summary” if you want to save only some high level information on each packet. Print summary is usually sufficient.
- Select “Print detail” and “Expand all levels” if you want to save all details of all packets at all levels.
- Click the “OK” button to complete the save operation.

Figure 1.4: Selecting print options.



If you select “Save” in the “File” menu, the captured data is saved in the format of a libpcap file. This format can be interpreted by both *tcpdump* and Wireshark. Measurements saved in libpcap format can be analyzed at a later time. However, libpcap files are not plain text files and are not useful for preparing your report.



In general, unless asked to do otherwise, always select the “Print summary” option when you include saved data in the lab report. This will help keep the length of the lab report reasonably small. If detailed information is required you will be asked to save “details” of the captured traffic. In this case, select the “Print detail” option.



Always include binary libpcap traces with your report, in addition to printed summaries or details that you use in the report to prove your answer is correct.

Question 8)

Include the file with the captured data in your lab report. Save the details of the captured traffic using the “Print Detail” option in the “Print” Window. Describe the differences between the files saved by `tcpdump` (in Part 7) and by Wireshark (in this part).

The files saved by `tcpdump` contain one line of information per packet, while wireshark's output contains many.

