

# INFRASTRUCTURE REPORT

30<sup>th</sup> APRIL 2020

**Made by:**

**Name : Baluku Joseph**

**Student No:- 4076788**



**SUPERVISOR**

**Mr. VLADIMIR KABZAR**

DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY OF  
FONTYS UNIVERSITY OF APPLIED SCIENCE , SCHOOL OF ICT

An Infrastructure Report to be Submitted to the School of Information and Communication  
Technology at

Fontys University of Applied sciences

## Table Of Contents

Table Of Contents .....	2
Chapter one .....	3
Introduction about student .....	3
Chapter Two : Network Basics Week 7.....	4
Investigation of hardware .....	4
Chapter Three : Internet Protocol Week 8 .....	7
Linux, Static IP address / Subnet configuration.....	7
Chapter Four :- Routing Week 9.....	18
IP Routing .....	18
Chapter Five : TCP/UDP Week 10 .....	22
TCP/UDP .....	22
Chapter Six : Conclusion .....	28
Chapter Seven : Personal Refraction .....	29

## **CHAPTER ONE**

### **INTRODUCTION ABOUT STUDENT**

I am Baluku Joseph, a first year student ,semester One doing a bachelors degree in Information and communication Technology at Fontys University School of Applied Science. I attend to PC05, and among the course unit we had in the First 11 weeks , we studied ICT and Infrastructure, which is about being about being to manage the existing IT infrastructure in all its facets and being able to design and realise a new infrastructure. I enjoyed being taught by Mr. Metaxas and then later Mr. Vladimir and I am happy I was so far I was introduced the introductory basics of the course.

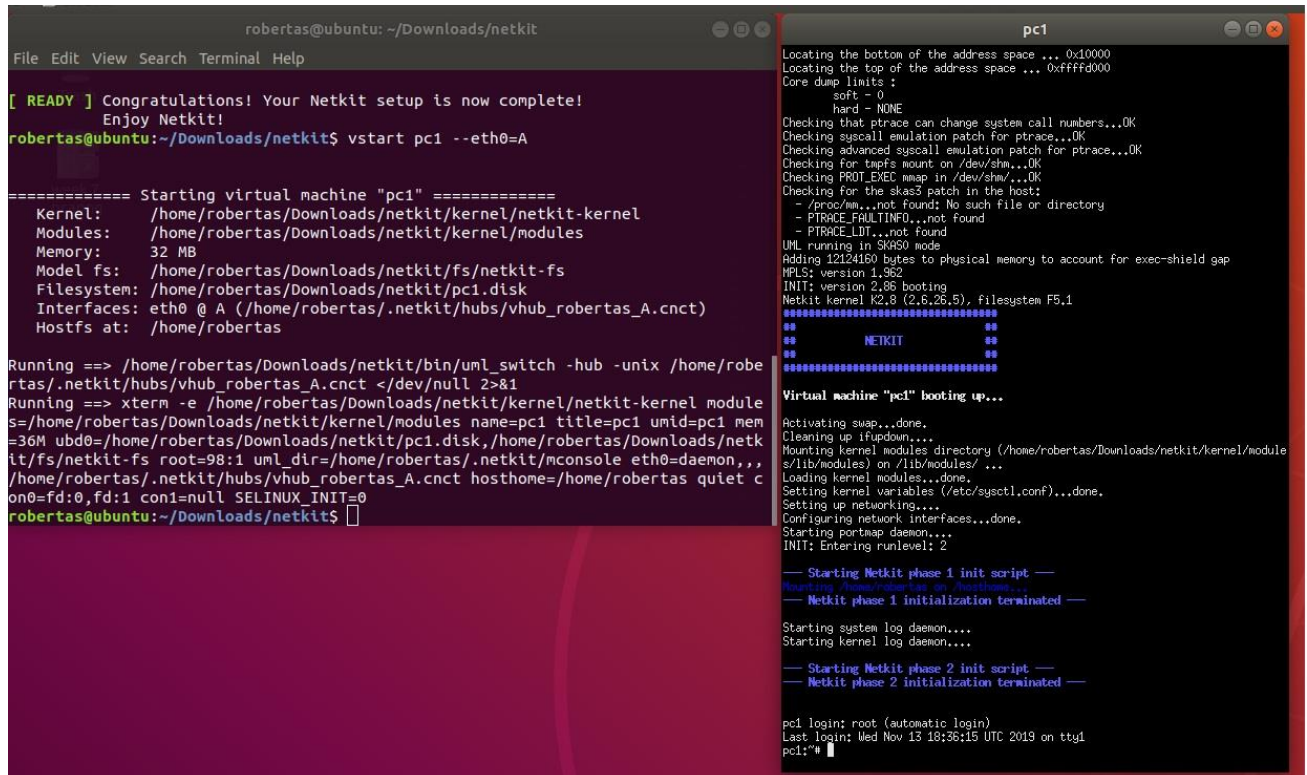
## CHAPTER TWO : NETWORK BASICS WEEK 7

### INVESTIGATION OF HARDWARE

#### *Task 1: install and Test Netkit Tool*

Consult this week's theory presentation and use the Netkit commands to start and halt a network node as described in the presentation. Netkit and Wireshark are already installed in the preconfigured Linux. If you installed the Linux yourself, then you need to install these tools yourself. (there is a guideline in the Canvas)

Provide screenshot of the started node.



```
robertas@ubuntu: ~/Downloads/netkit
File Edit View Search Terminal Help

[ READY ] Congratulations! Your Netkit setup is now complete!
Enjoy Netkit!
robertas@ubuntu:~/Downloads/netkit$ vstart pc1 --eth0=A

===== Starting virtual machine "pc1" =====
Kernel: /home/robertas/Downloads/netkit/kernel/netkit-kernel
Modules: /home/robertas/Downloads/netkit/kernel/modules
Memory: 32 MB
Model fs: /home/robertas/Downloads/netkit/fs/netkit-fs
Filesystem: /home/robertas/Downloads/netkit/pc1.disk
Interfaces: eth0 @ A (/home/robertas/.netkit/hubs/vhub_robertas_A.cnct)
Hostfs at: /home/robertas

Running ==> /home/robertas/Downloads/netkit/bin/uml_switch -hub -unix /home/robertas/.netkit/hubs/vhub_robertas_A.cnct </dev/null 2>&1
Running ==> xterm -e /home/robertas/Downloads/netkit/kernel/netkit-kernel module s=/home/robertas/Downloads/netkit/kernel/modules name=pc1 title=pc1 umid=pc1 mem=36M ubd0=/home/robertas/Downloads/netkit/pc1.disk,/home/robertas/Downloads/netkit/fs/netkit-fs root=98:1 uml_dir=/home/robertas/.netkit/mconsole eth0=daemon,, /home/robertas/.netkit/hubs/vhub_robertas_A.cnct hosthome=/home/robertas quiet c on0=fd:0,fd:1 con1=null SELINUX_INIT=0
robertas@ubuntu:~/Downloads/netkit$

Virtual machine "pc1" booting up...
Locating the bottom of the address space ... 0x10000
Locating the top of the address space ... 0xffffd000
Core dump limits :
  soft - 0
  hard - NONE
Checking that ptrace can change system call numbers...OK
Checking syscall emulation patch for ptrace...OK
Checking advanced syscall emulation patch for ptrace...OK
Checking for tmpfs mount on /dev/shm...OK
Checking PROT_EXEC map in /dev/shm...OK
Checking for the skas3 patch in the host!
  - /proc/mem...not found: No such file or directory
  - PT_TRACE_FAULTINFO...not found
  - PT_TRACE_LDT...not found
UML running in SKAS0 mode
Adding 12124160 bytes to physical memory to account for exec-shield gap
INIT: version 2.86 booting
Netkit kernel K2.8 (2.6.26.5), filesystem F5.1
=====
** NETKIT **
=====

Virtual machine "pc1" booting up...
Activating swap...done.
Cleaning up ifupdown...
Mounting kernel modules directory (/home/robertas/Downloads/netkit/kernel/module s/lib/modules) on /lib/modules/ ...
Loading kernel modules...done.
Setting kernel variables (/etc/sysctl.conf)...done.
Setting up networking....
Configuring network interfaces...done.
Starting portmap daemon...
INIT: Entering runlevel: 2

--- Starting Netkit phase 1 init script ---
Starting /home/robertas/.netkit/bin/...
--- Netkit phase 1 initialization terminated ---

Starting system log daemon....
Starting kernel log daemon....

--- Starting Netkit phase 2 init script ---
--- Netkit phase 2 initialization terminated ---

pc1 login: root (automatic login)
Last login: Wed Nov 13 18:36:15 UTC 2019 on tty1
pc1:~#
```

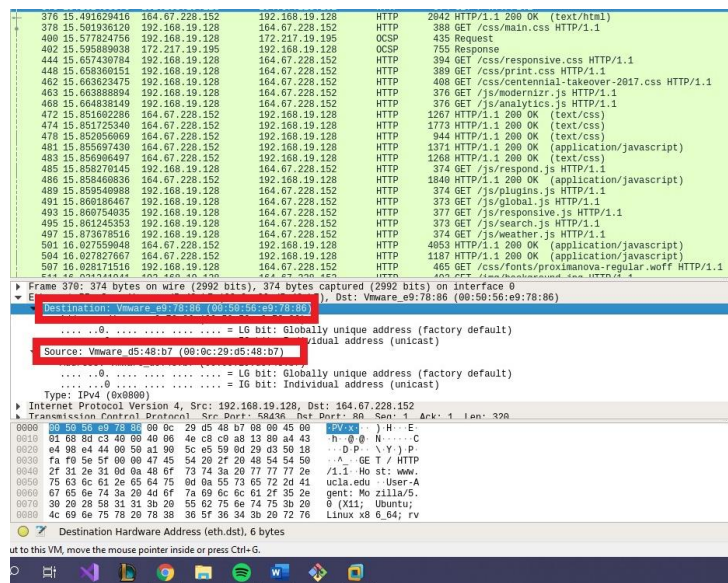
## Task 2: TCP/IP Layers in Wireshark

Find a Wireshark Tutorial on the web. Run Wireshark.

Start capturing the network traffic. To generate HTTP traffic, go to some web page with your web browser (e.g. [www.fontys.nl](http://www.fontys.nl)). Don't forget to stop capturing as you can get a lot of traffic in your capture. Look at your captured packets and find an HTTP GET packet and Answer the following questions and provide the screenshots:

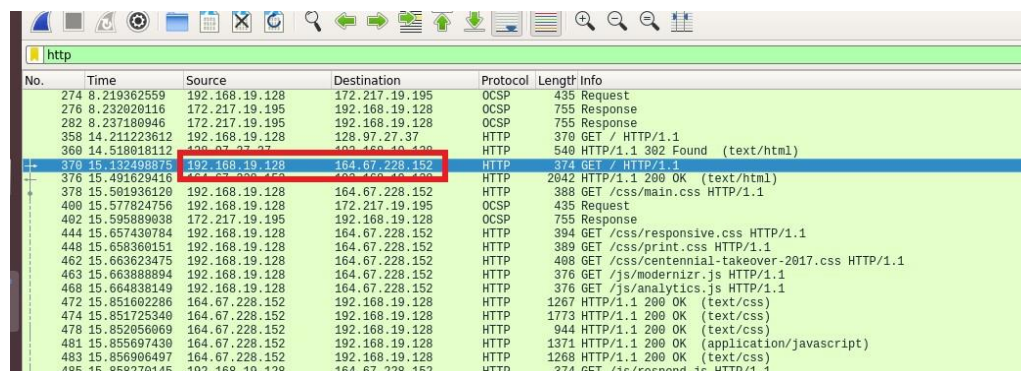
- What is the source and destination MAC address of this HTTP packet? Provide a screenshot to prove it

The source MAC address is 00:0c:29:d5:48:b7 and the destination MAC address is 00:50:56:e9:78:86

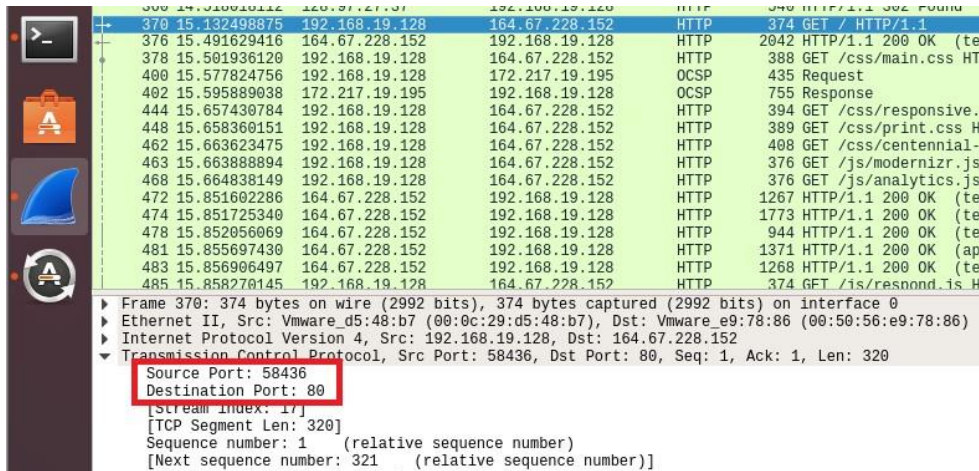


- What is the source and destination IP address of this HTTP packet? Provide a screenshot to prove it

The source IP address is 192.168.19.128 and the destination address is 164.67.228.152



- What is the source and destination port of this HTTP packet? Provide a screenshot to prove it  
Source port is 58436 and destination port is 80



- What is the host name of this HTTP Get packet?  
www.ucla.edu\r\n
- Find the HTTP Response belonging to the HTTP Get packet. How much time elapsed between the HTTP Get and HTTP response?  
Around 0.36 seconds elapsed between the request and response.

### Task 3: Do Linux Tutorial

Go to <http://www.ee.surrey.ac.uk/Teaching/Unix/index.html> and do the tutorial three.

Provide screenshots of all exercises in section 3.4

## CHAPTER THREE : INTERNET PROTOCOL WEEK 8

### LINUX, STATIC IP ADDRESS / SUBNET CONFIGURATION

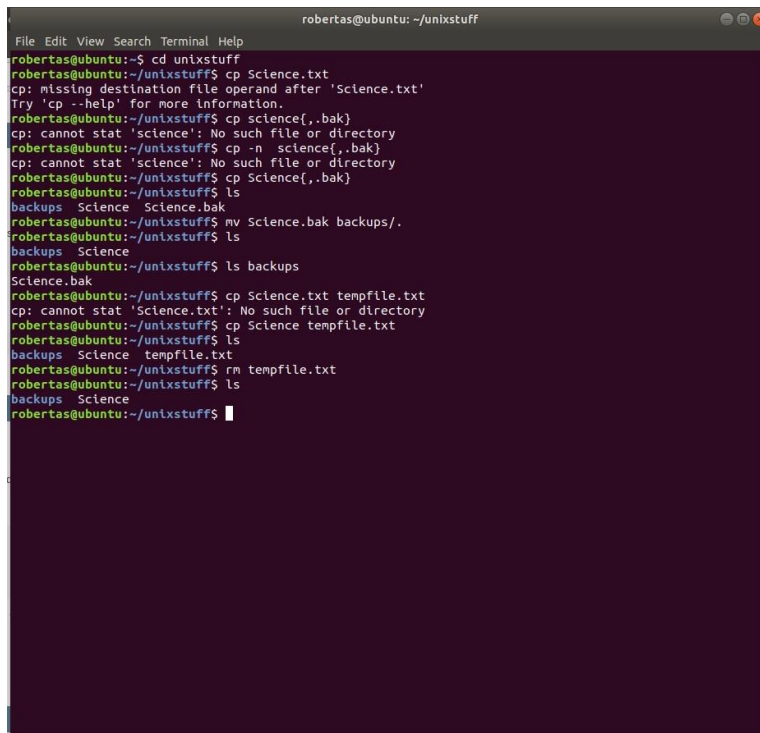
#### *LINUX, STATIC IP ADDRESS/SUBNETS CONFIGURATION*

##### *Task 1a: Do Linux Tutorial*

Go to <http://www.ee.surrey.ac.uk/Teaching/Unix/unix2.html> and do the 2nd basic Unix tutorial.

Provide screenshots of all exercises 2a and 2b. Do all subsections of this tutorial – all of them are really useful! This task should be done individually, so each member of the team should provide his/her evidence(screenshots).

2a



```
robertas@ubuntu: ~/unixstuff
File Edit View Search Terminal Help
robertas@ubuntu:~$ cd unixstuff
robertas@ubuntu:~/unixstuff$ cp Science.txt
cp: missing destination file operand after 'Science.txt'
Try 'cp --help' for more information.
robertas@ubuntu:~/unixstuff$ cp science{,.bak}
cp: cannot stat 'science': No such file or directory
robertas@ubuntu:~/unixstuff$ cp -n science{,.bak}
cp: cannot stat 'science': No such file or directory
robertas@ubuntu:~/unixstuff$ cp Science{,.bak}
robertas@ubuntu:~/unixstuff$ ls
backups  Science  Science.bak
robertas@ubuntu:~/unixstuff$ mv Science.bak backups/.
robertas@ubuntu:~/unixstuff$ ls
backups  Science
robertas@ubuntu:~/unixstuff$ ls backups
Science.bak
robertas@ubuntu:~/unixstuff$ cp Science.txt tempfile.txt
cp: cannot stat 'Science.txt': No such file or directory
robertas@ubuntu:~/unixstuff$ cp Science tempfile.txt
robertas@ubuntu:~/unixstuff$ ls
backups  Science  tempfile.txt
robertas@ubuntu:~/unixstuff$ rm tempfile.txt
robertas@ubuntu:~/unixstuff$ ls
backups  Science
robertas@ubuntu:~/unixstuff$
```

2b



```
robertas@ubuntu: ~/unixstuff
File Edit View Search Terminal Help
robertas@ubuntu:~/unixstuff$ mkdir tempstuff
robertas@ubuntu:~/unixstuff$ ls
backups Science tempstuff
robertas@ubuntu:~/unixstuff$ rmdir tempstuff
robertas@ubuntu:~/unixstuff$ ls
backups Science
robertas@ubuntu:~/unixstuff$
```

2a

```
Joseph@ubuntu: ~/unixstuff
File Edit View Search Terminal Help
Joseph@ubuntu:~$ cd foo
Joseph@ubuntu:~/foo$ ./check_configuration.sh
ash: ./check_configuration.sh: No such file or directory
Joseph@ubuntu:~/foo$ cd Downloads
ash: cd: Downloads: No such file or directory
Joseph@ubuntu:~/foo$ cd
Joseph@ubuntu:~$ ls
desktop Downloads foo Pictures Templates Videos
documents examples.desktop Music Public unixstuff
Joseph@ubuntu:~$ cd unixstuff
Joseph@ubuntu:~/unixstuff$ cp Science.txt
cp: missing destination file operand after 'Science.txt'
Try 'cp --help' for more information.
Joseph@ubuntu:~/unixstuff$ cp Science(.bak)
Joseph@ubuntu:~/unixstuff$ ls
Science Science.bak
Joseph@ubuntu:~/unixstuff$ mv Science.bak backups/.
mv: cannot stat 'Science.bak': No such file or directory
Joseph@ubuntu:~/unixstuff$ ls
Science Science.bak
Joseph@ubuntu:~/unixstuff$ mkdir backup
Joseph@ubuntu:~/unixstuff$ ls
backup Science Science.bak
Joseph@ubuntu:~/unixstuff$ ls backup
Joseph@ubuntu:~/unixstuff$ mv Science.bak backup/.
mv: cannot stat 'Science.bak': No such file or directory
Joseph@ubuntu:~/unixstuff$ mv Science.bak backup/.
Joseph@ubuntu:~/unixstuff$ ls
backup Science
Joseph@ubuntu:~/unixstuff$ ls backup
Science.bak
Joseph@ubuntu:~/unixstuff$
```



2b

```
File Edit View Search Terminal Help
joseph@ubuntu: ~/junkstuff
joseph@ubuntu:~/junkstuff$ mkdir tempstuff
joseph@ubuntu:~/junkstuff$ ls
backup Science tempstuff
joseph@ubuntu:~/junkstuff$ rmdir tempstuff
joseph@ubuntu:~/junkstuff$ ls
backup Science
joseph@ubuntu:~/junkstuff$
```

### Task 1b: Networking exercise

Do the networking online exercises via this link <https://courses.codemax.net/w8.html>.

Provide screenshots of all exercises.

Consider a Network Device with the following address: 147.223.79.225/13

Which of the following IP addresses belongs to the same network?

- ☐ 147.224.11.107
- ☐ 147.215.150.188
- ☐ 147.224.74.213
- ☐ 147.224.95.188
- ☐ 147.218.170.171
- ☐ 147.215.197.137
- ☐ 147.215.185.91

Completed exercises		
Classful IP addresses		
address class	:	5/5
classful subnets	:	5/5
Classless IP addresses		
8-bit aligned subnets	:	5/5
arbitrary subnets	:	4/4
network address ranges	:	2/2

## Task 2: Build A Simple Netkit Network

Read the explanation of the basic Netkit commands and use them to build a simple network of two nodes connected to a LAN interface.

Try the following configurations:

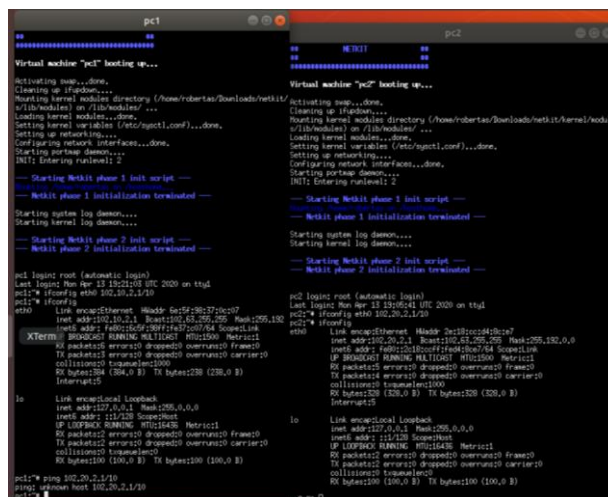
A) Configure the IP addresses of the 2 nodes by using the “ifconfig” command explained in the theory lesson.

1. Node1 has an IP address 102.10.2.1/24
2. Node2 has an IP address 102.20.2.1/24

Check whether your configuration was successful by using ping command between these two nodes.

1. What is the result of the ping? Can you explain it? Provide a screenshot.

“Network is unreachable” The Node1 cannot reach Node2 since their Network id does not match



```
pc1
#
# Virtual machine "pc1" booting up...
#
# Activating net...done.
# Loading up ifconfig...
# Loading kernel modules directory (/home/robertas/Downloads/netkit/kernel/modules) on /lib/modules/...
# Loading kernel modules...done.
# Setting kernel variables (/etc/sysctl.conf)...done.
# Setting up network...done.
# Configuring network interface...done.
# Starting portmap daemon...
# [MTI] Entering rulelevel 2
#
# --- Starting Netkit phase 1 init script ---
# --- Netkit phase 1 initialization terminated ---
# Starting system log daemon...
# Starting kernel log daemon...
# --- Starting Netkit phase 2 init script ---
# --- Netkit phase 2 initialization terminated ---
#
# pc1 login root (automatic login)
# Last login: Mon Apr 13 13:21:18 UTC 2020 on ttyd
# pc1~# ifconfig
# eth0
# Link encap:Ethernet  HWaddr 0a:5d:56:37:0c:07
# inet addr:102.10.2.1 Bcast:102.10.2.255 Mask:255.255.0
# inet6 addr: fe80::102:10:2:1::1/64 ScopeLink
# RX: 0 packets received: 0 dropped: 0 overruns: 0 frame0
# TX: 0 packets transmitted: 0 dropped: 0 overruns: 0 carrier0
# collisions:0 txqueuelen:1000
# RX bytes:136 (136.0 B) TX bytes:288 (288.0 B)
# Interrupts:0
#
# lo
# Link encap:Local Loopback
# inet addr:127.0.0.1 Mask:255.0.0.0
# inet6 addr: ::1/128 ScopeHost
# IP: 0 packets received: 0 dropped: 0 overruns: 0 frame0
# TX: 0 packets transmitted: 0 dropped: 0 overruns: 0 carrier0
# collisions:0 txqueuelen:0
# RX bytes:100 (100.0 B) TX bytes:100 (100.0 B)
#
# pc1~# ping 102.20.2.1/24
# ping: unknown host 102.20.2.1/24
# pc1~#
```

```
pc2
#
# Virtual machine "pc2" booting up...
#
# Activating net...done.
# Loading up ifconfig...
# Loading kernel modules directory (/home/robertas/Downloads/netkit/kernel/modules) on /lib/modules/...
# Loading kernel modules...done.
# Setting kernel variables (/etc/sysctl.conf)...done.
# Setting up network...done.
# Configuring network interface...done.
# Starting portmap daemon...
# [MTI] Entering rulelevel 2
#
# --- Starting Netkit phase 1 init script ---
# --- Netkit phase 1 initialization terminated ---
# Starting system log daemon...
# Starting kernel log daemon...
# --- Starting Netkit phase 2 init script ---
# --- Netkit phase 2 initialization terminated ---
#
# pc2 login root (automatic login)
# Last login: Mon Apr 13 13:21:41 UTC 2020 on ttyd
# pc2~# ifconfig
# eth0
# Link encap:Ethernet  HWaddr 0a:5d:56:37:0c:07
# inet addr:102.20.2.1 Bcast:102.20.2.255 Mask:255.255.0
# inet6 addr: fe80::102:20:2:1::1/64 ScopeLink
# IP: 0 packets received: 0 dropped: 0 overruns: 0 frame0
# TX: 0 packets transmitted: 0 dropped: 0 overruns: 0 carrier0
# collisions:0 txqueuelen:1000
# RX bytes:136 (136.0 B) TX bytes:288 (288.0 B)
# Interrupts:0
#
# lo
# Link encap:Local Loopback
# inet addr:127.0.0.1 Mask:255.0.0.0
# inet6 addr: ::1/128 ScopeHost
# IP: 0 packets received: 0 dropped: 0 overruns: 0 frame0
# TX: 0 packets transmitted: 0 dropped: 0 overruns: 0 carrier0
# collisions:0 txqueuelen:0
# RX bytes:100 (100.0 B) TX bytes:100 (100.0 B)
#
# pc2~#
```

2. Look at the ARP entries of your Node1 and Node2. Which command do you use? Which ARP entries are there?

There are no ARP entries because the ping didn't work.

B) Configure the IP addresses of the 2 nodes by using the “ip” command explained in the theory lesson.

1. Node1 has an IP address 102.10.2.1/10
2. Node2 has an IP address 102.20.2.1/10

Check whether your configuration was successful by using ping command between these two nodes.



3. After successful ping ARP entries of both nodes should be changed. Provide a screenshot of the new ARP situation and explain it. What is the command to clear the ARP cache again?

Pc2 sent a ping to pc1 and pc1 gave information back to the pc2. The command to delete ARP entries is `arp -s`

```
TX packets:18 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueue:0
RX bytes:1268 (1.2 KiB) TX bytes:1268 (1.2 KiB)

pc1:~# arp
Address          HWtype  HWaddress      Flags Mask
Iface
102.10.2.2        ether    2e:18:cc:d4:8c:e7  C
eth0
pc1:~#
```

```
--- 102.10.2.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2031ms
rtt min/avg/max/mdev = 0.695/1.755/3.618/1.321 ms

pc2:~# arp
Address          HWtype  HWaddress      Flags Mask
Iface
102.10.2.1        ether    6e:5f:98:37:0c:07  C
eth0
pc2:~#
```

### Task 3: Configuring Network

For this assignment you can use a preconfigured netkit lab provided in net\_routing.zip file. To do this you need to copy the provided zip file somewhere in your Linux environment, e.g. in ~/netkit\_labs. Unzip the file. You have now a preconfigured lab Deliver the lab network of this task in your *git* project. *Thus, when you are done write below the URL of your git project (I should be able to access your results using “git clone” and the provided git URL).*

Each simulated node has its own directory. Also, each simulated node has a <node>.startup file where any commands can be added that should be executed before startup of the node.

To start the lab issue the following command in the root directory of your lab:

lstart

Note: When you issue this command, you’ll be prompted for a password which in your case is **student**.

Netkit uses the file “labs.conf” in order to initialize the Ethernet devices and their respective collision domains for each node. For example inside the labs.conf there is a line “RouterAC[0]=LANA” and a line “RouterAC[1]=LANC”.

These two lines have same effect when the node “RouterAC” is initialized, as if we would run the command:

*“vstart RouterAC --eth0=LANA --eth1=LANC”.*

Now all the nodes should be started. However, the nodes are not configured yet. You need to configure them as follows:

Configure the Ethernet devices connected via the collision domain LANA using the IP range 10.X.0.0/16, where X is the number of your pair/group.

Configure the Ethernet devices connected via the collision domain LANB using the IP range 172.16.X.0/24, where X is the number of your pair/group.

Configure the Ethernet devices connected via the collision domain LANC using the IP range 192.168.X.0/24, where X is the number of your pair/group.

For example if your group number is 230 you should use IP address from the range 10.230.0.0/16 for LANA, 172.16.230.0/24 for LANB and 192.168.230.0/24. (see also table 1).

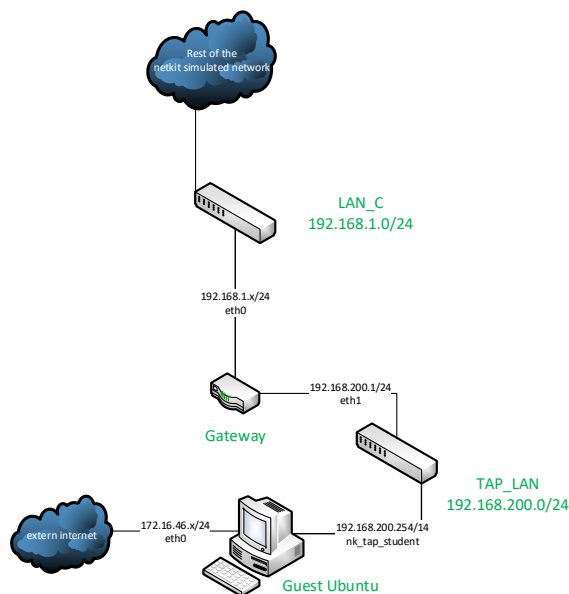
There are 2 ways to configure your interfaces. *We recommend you all use the first option and for your own experiment you can use the second option but make sure all your submissions follow the first option:*

1. Use either ifconfig or ip commands. Once you know how the commands should look like, it is highly recommended to put them in <node>.startup files, so next time you want to restart and present your lab, you don’t have to reconfigure it by hand again. Note : Please don’t remove the commands which are already present in the <node>.startup files. They are necessary for starting up Linux networking service.
2. Use <node>/etc/network/interfaces file of the node you want to configure.

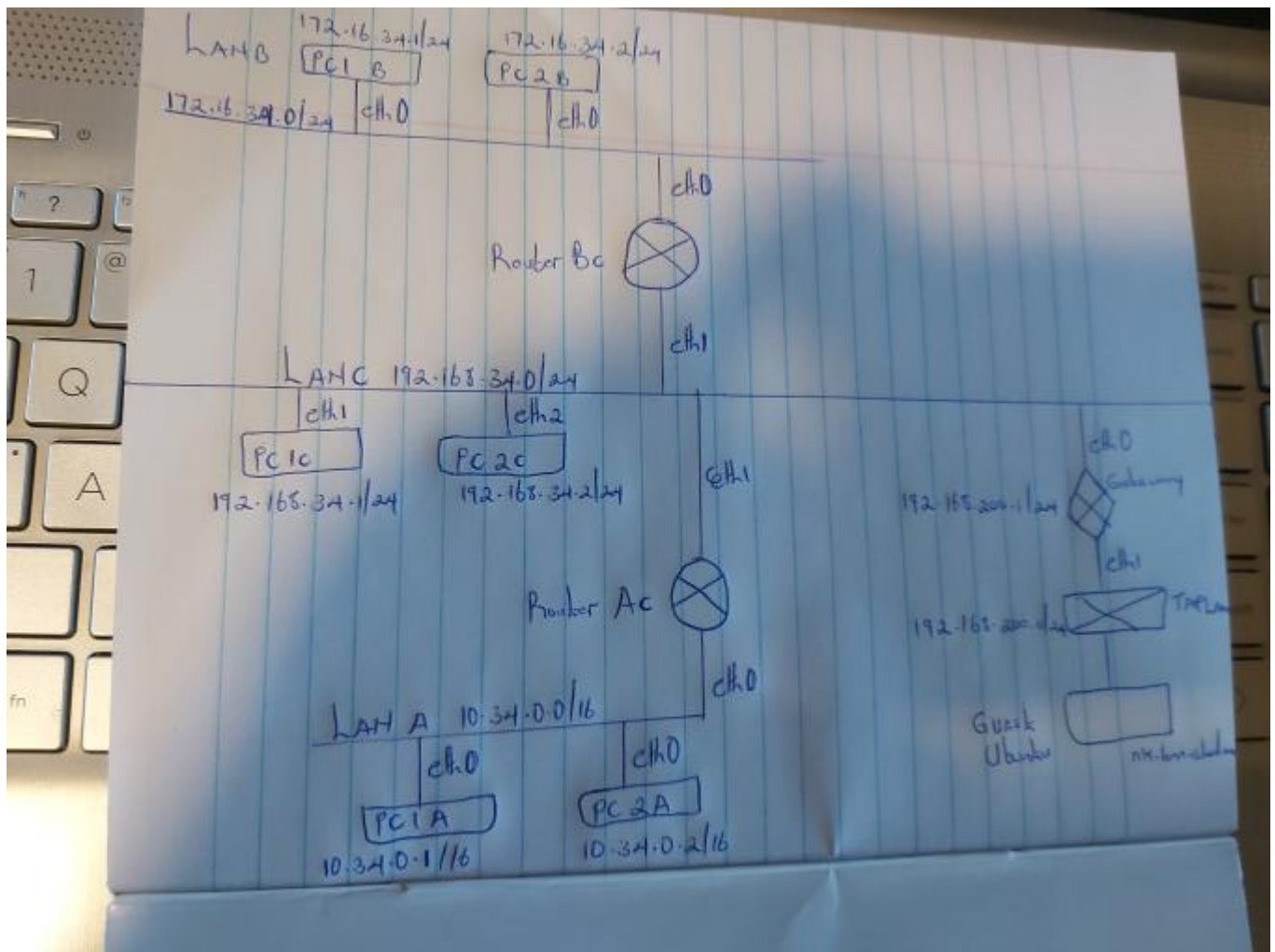
In the netkit lab environment you can put any files the contents of which you want to see in the simulated node in the <node> directory. In this way, you can also put there <node>/etc/network/interfaces file. This file is used by Linux system to configure the network interfaces. An example of such a file is provided in the lab for PC1A node.

The network of the lab is as follows:

1. PC1A, PC2A and RouterAC are connected to LAN\_A
2. PC1B, PC2B and RouterBC are connected to LAN\_B
3. PC1C, PC2C, RouterBC, RouterAC and Gateway are connected to LAN\_C
4. Gateway is connected to LAN\_C through fixed eth0 interface with IP address 192.168.1.x/24 and to TAP\_LAN through eth1 interface with IP address 192.168.200.1. The TAP\_LAN is a Netkit-specific interface used for the connection to your guest Linux system. The Gateway node will be used for the optional part of the Assignment 3.
5. Your guest Linux system is connected to your simulated Netkit node Gateway through Netkit specific tap interface nk\_tap\_student 192.168.200.254, see the detail of the connection between the Netkit simulated environment and your Guest machine in the picture below.



**Provide the network drawing** of your lab network you can use <https://app.diagrams.net/> and screenshots of the pings which are possible WITHIN LAN\_A, LAN\_B and LAN\_C (PC1A to PC2A, PC1B to PC2B and so on). When creating the network drawing, don't forget to mention the IP addresses/subnet masks for all nodes of your network. It is also useful to include the names of the network interfaces (eth0, eth1, ...).



```

PC1A
=====
Lab directory (host): /home/robertas/Desktop/netkit_lab/net_routing
Version: 1.0
Author: Renata Franken
Email: <none>
Web: <none>
Description:
NETKIT Assignment 2/3 : IP addressing and routing
=====
--- Netkit phase 2 initialization terminated ---

PC1a login: root (automatic login)
PC1a:~# ping 10.34.0.2
PING 10.34.0.2 (10.34.0.2) 56(84) bytes of data:
64 bytes from 10.34.0.2: icmp_seq=1 ttl=64 time=6.87 ms
^C
--- 10.34.0.2 ping statistics ---
2 packets transmitted, 1 received, 0% packet loss, time 1ms
rtt min/avg/max/ndev = 6.878/6.878/6.878/0.000 ms
PC1a:~#

PC1B
=====
Lab directory (host): /home/robertas/Desktop/netkit_lab/net_routing
Version: 1.0
Author: Renata Franken
Email: <none>
Web: <none>
Description:
NETKIT Assignment 2/3 : IP addressing and routing
=====
--- Netkit phase 2 initialization terminated ---

PC1b login: root (automatic login)
PC1b:~# ping 172.16.34.2
PING 172.16.34.2 (172.16.34.2) 56(84) bytes of data:
64 bytes from 172.16.34.2: icmp_seq=1 ttl=64 time=4.45 ms
64 bytes from 172.16.34.2: icmp_seq=2 ttl=64 time=0.644 ms
^C
--- 172.16.34.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1011ms
rtt min/avg/max/ndev = 0.644/2.548/4.453/1.905 ms
PC1b:~#

PC1C
=====
Author: Renata Franken
Email: <none>
Web: <none>
Description:
NETKIT Assignment 2/3 : IP addressing and routing
=====
--- Netkit phase 2 initialization terminated ---

PC1c login: root (automatic login)
Last login: Tue Apr 14 17:10:05 UTC 2020 on tttyl
PC1c:~# ping 192.168.34.2
-bash: 192.168.34.2: command not found
PC1c:~# ping 192.168.34.2
PING 192.168.34.2 (192.168.34.2) 56(84) bytes of data:
64 bytes from 192.168.34.2: icmp_seq=1 ttl=64 time=8.36 ms
64 bytes from 192.168.34.2: icmp_seq=2 ttl=64 time=0.566 ms
^C
--- 192.168.34.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1011ms
rtt min/avg/max/ndev = 0.566/4.765/8.364/4.139 ms
PC1c:~#

PC2A
=====
Lab directory (host): /home/robertas/Desktop/netkit_lab/net_routing
Version: 1.0
Author: Renata Franken
Email: <none>
Web: <none>
Description:
NETKIT Assignment 2/3 : IP addressing and routing
=====
--- Netkit phase 2 initialization terminated ---

PC2a login: root (automatic login)
PC2a:~# ping 10.34.0.1
PING 10.34.0.1 (10.34.0.1) 56(84) bytes of data:
64 bytes from 10.34.0.1: icmp_seq=1 ttl=64 time=0.373 ms
64 bytes from 10.34.0.1: icmp_seq=2 ttl=64 time=0.427 ms
^C
--- 10.34.0.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1012ms
rtt min/avg/max/ndev = 0.373/0.400/0.427/0.027 ms
PC2a:~#

PC2B
=====
Lab directory (host): /home/robertas/Desktop/netkit_lab/net_routing
Version: 1.0
Author: Renata Franken
Email: <none>
Web: <none>
Description:
NETKIT Assignment 2/3 : IP addressing and routing
=====
--- Netkit phase 2 initialization terminated ---

PC2b login: root (automatic login)
PC2b:~# ping 172.16.34.1
PING 172.16.34.1 (172.16.34.1) 56(84) bytes of data:
64 bytes from 172.16.34.1: icmp_seq=1 ttl=64 time=0.263 ms
64 bytes from 172.16.34.1: icmp_seq=2 ttl=64 time=0.460 ms
^C
--- 172.16.34.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1017ms
rtt min/avg/max/ndev = 0.263/0.364/0.460/0.097 ms
PC2b:~#

PC2C
=====
Lab directory (host): /home/robertas/Desktop/netkit_lab/net_routing
Version: 1.0
Author: Renata Franken
Email: <none>
Web: <none>
Description:
NETKIT Assignment 2/3 : IP addressing and routing
=====
--- Netkit phase 2 initialization terminated ---

PC2c login: root (automatic login)
PC2c:~# ping 192.168.34.1
PING 192.168.34.1 (192.168.34.1) 56(84) bytes of data:
64 bytes from 192.168.34.1: icmp_seq=1 ttl=64 time=0.294 ms
64 bytes from 192.168.34.1: icmp_seq=2 ttl=64 time=0.441 ms
^C
--- 192.168.34.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1011ms
rtt min/avg/max/ndev = 0.294/0.367/0.441/0.075 ms
PC2c:~#

```

You don't need to be able to route between all nodes of this network; that is the second part of the assignment, which will be done next week. 10



Note 1: In the provided netkit lab there are files HOWTO, interfaces.example and Example.startup which can give you more info on how to use and configure the lab.

Table 1 : IPv4 address ranges per student group

Group	LANA	LANB	LANC
1	10.1.0.0/16	172.16.1.0/24	192.168.1.0/24
2	10.2.0.0/16	172.16.2.0/24	192.168.2.0/24
...			
n	10.n.0.0/16	172.16.n.0/24	192.168.n.0/24

#### ***Task 4: CIDR IP Addressing Exercises***

**1. Suppose we have IP address 122.33.196.145/24**

Fill in the following items for this address:

1. Network Address  
122.33.196.0
2. Broadcast Address  
122.33.196.255
3. Subnet Mask  
255.255.255.0

**2. Suppose we have IP address 163.249.223.229/25**

Fill in the following items for this address:

1. Network Address  
163.249.223.129/25
2. First Host  
163.249.223.129
3. Last Host  
163.249.223.254
4. Broadcast Address  
163.249.223.255

## Chapter Four :- Routing Week 9

### IP Routing

#### Task 1: A bit more complex network: Part 2

Last week you did the configuration of your IP network for the preconfigured lab.

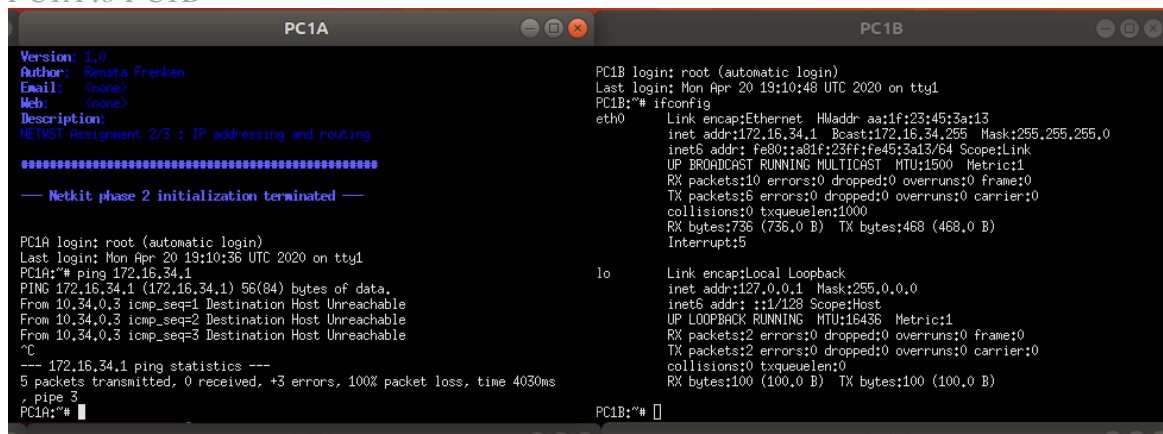
If you have done well and used either scripts or network/interfaces files, you should be able to restart your configured environment again. Also, you should have a drawing of your network.

Your task is adding routing information to your nodes in such a way, that every node of your network should be able to ping any other node of your network. The routes should be optimal, so the shortest path from node to node should be used. To implement routing, you'll have to use different types of routes as learned on the theory lesson.

Tip: Use the network drawing from the last week assignment (week 8 ) and first think about the way you're going to route. Use **tcpdump** and **traceroute** commands to debug your routing.

Provide screenshots of the following pings:

#### 1. PC1A to PC1B



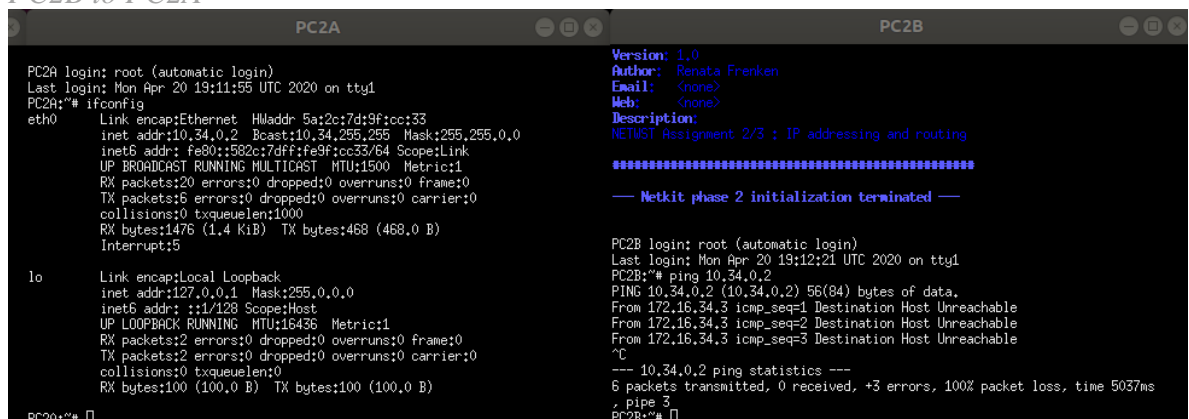
```
PC1A
Version: 1.0
Author: Renata Franken
Email: <none>
Web: <none>
Description:
NETWST Assignment 2/3 : IP addressing and routing
*****
--- Netkit phase 2 initialization terminated ---

PC1A login: root (automatic login)
Last login: Mon Apr 20 19:10:36 UTC 2020 on tty1
PC1A:~# ping 172.16.34.1
PING 172.16.34.1 (172.16.34.1) 56(84) bytes of data,
From 10.34.0.3 icmp_seq=1 Destination Host Unreachable
From 10.34.0.3 icmp_seq=2 Destination Host Unreachable
From 10.34.0.3 icmp_seq=3 Destination Host Unreachable
^C
--- 172.16.34.1 ping statistics ---
5 packets transmitted, 0 received, +3 errors, 100% packet loss, time 4030ms
, pipe 3
PC1A:~#

PC1B
PC1B login: root (automatic login)
Last login: Mon Apr 20 19:10:48 UTC 2020 on tty1
PC1B:~# ifconfig
eth0      Link encap:Ethernet  HWaddr aa:1f:23:45:3a:13
          inet addr:172.16.34.1  Bcast:172.16.34.255  Mask:255.255.255.0
          inet6 addr: fe80::a81f:23ff:fe45:3a13/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:10 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:736 (736.0 B)  TX bytes:468 (468.0 B)
          Interrupt:5

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:100 (100.0 B)  TX bytes:100 (100.0 B)
```

#### 2. PC2B to PC2A



```
PC2A
PC2A login: root (automatic login)
Last login: Mon Apr 20 19:11:55 UTC 2020 on tty1
PC2A:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 5a:2c:7d:9f:cc:33
          inet addr:10.34.0.2  Bcast:10.34.255.255  Mask:255.255.0.0
          inet6 addr: fe80::582c:7dff:fe9f:cc33/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:20 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1476 (1.4 KiB)  TX bytes:468 (468.0 B)
          Interrupt:5

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:100 (100.0 B)  TX bytes:100 (100.0 B)

PC2A:~#

PC2B
Version: 1.0
Author: Renata Franken
Email: <none>
Web: <none>
Description:
NETWST Assignment 2/3 : IP addressing and routing
*****
--- Netkit phase 2 initialization terminated ---

PC2B login: root (automatic login)
Last login: Mon Apr 20 19:12:21 UTC 2020 on tty1
PC2B:~# ping 10.34.0.2
PING 10.34.0.2 (10.34.0.2) 56(84) bytes of data,
From 172.16.34.3 icmp_seq=1 Destination Host Unreachable
From 172.16.34.3 icmp_seq=2 Destination Host Unreachable
From 172.16.34.3 icmp_seq=3 Destination Host Unreachable
^C
--- 10.34.0.2 ping statistics ---
6 packets transmitted, 0 received, +3 errors, 100% packet loss, time 5037ms
, pipe 3
PC2B:~#
```

### 3. PC2A to PC1C

```

PC1C login: root (automatic login)
Last login: Mon Apr 20 19:11:32 UTC 2020 on tty1
PC1C:~# ifconfig
eth0      Link encap:Ethernet  HWaddr ae:9a:8e:56:30:7d
          inet addr:192.168.34.1  Bcast:192.168.34.255  Mask:255.255.255.0
          inet6 addr: fe80::ac9a:8eff:fe56:307d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:28 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1456 (1.4 KiB)  TX bytes:468 (468.0 B)
          Interrupt:5

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:10 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:628 (628.0 B)  TX bytes:628 (628.0 B)

PC1C:~#

PC2A:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 5a:2c:7d:9f:cc:33
          inet addr:10.34.0.2  Bcast:10.34.255.255  Mask:255.255.0.0
          inet6 addr: fe80::582c:7dff:fe9f:cc33/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:21 errors:0 dropped:0 overruns:0 frame:0
          TX packets:15 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1504 (1.4 KiB)  TX bytes:1294 (1.2 KiB)
          Interrupt:5

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:100 (100.0 B)  TX bytes:100 (100.0 B)

PC2A:~# ping 192.168.34.1
PING 192.168.34.1 (192.168.34.1) 56(84) bytes of data.

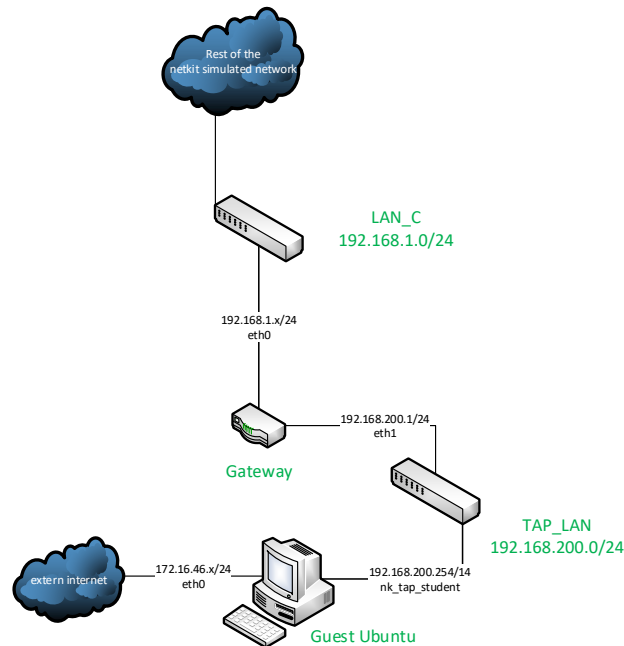
```

Give a list of all nodes where you had to adjust the routing tables and the screenshots of their configured routing tables.

Node	Configuration
PC2C.startup	<pre> ifconfig eth0 up /etc/init.d/networking start ifconfig eth0 192.168.34.2/24 route add -net 172.16.34.0/24 gw 192.168.34.3 route add -net 10.34.0.0/16 gw 192.168.34.4 route add default gw 192.168.34.5 </pre>
RouterAC.startup	<pre> /etc/init.d/networking start ifconfig eth0 10.34.0.3/16 ifconfig eth1 192.168.34.3/24 route add default gw 192.168.34.5 route add -net 172.16.34.0/24 gw 192.168.34.3 </pre>
RouterBC.startup	<pre> /etc/init.d/networking start sysctl -w net.ipv6.conf.all.forwarding=1 ifconfig eth0 172.16.34.3/24 ifconfig eth1 192.168.34.4/24 route add default gw 192.168.34.5 route add -net 10.34.0.0/16 gw 192.168.34.4 </pre>
PC1B.startup	<pre> ifconfig eth0 up /etc/init.d/networking start ifconfig eth0 172.16.34.1/24 route add default gw 172.16.34.3 </pre>
PC1A.startup	<pre> ifconfig eth0 up /etc/init.d/networking start ifconfig eth0 10.34.0.1/16 route add default gw 10.34.0.3 </pre>
PC2B.startup	<pre> ifconfig eth0 up /etc/init.d/networking start ifconfig eth0 172.16.34.2/24 route add default gw 172.16.34.3 </pre>
PC2A.startup	<pre> ifconfig eth0 up /etc/init.d/networking start ifconfig eth0 10.34.0.2/16 route add default gw 10.34.0.3 </pre>

## Task 2 (Optional): Access the outside world

The provided lab has also an interface outside of the Netkit to your host Linux machine, so called Netkit tap interface. To use this interface you need to use node Gateway, which is connected with one interface to LANC and with the other (tap) interface to your guest Linux system which is then connected to the outside world. The schematics of this interface is:



Configure your network in such a way that you can reach a node on Internet.

To prove your correct configuration you should be able to ping a host like 8.8.8.8 (Google DNS server) from any node on your network.

Provide screenshots of the following ping:

PC1A to 8.8.8.8, PC1B to 8.8.8.8

*Table 2 : IPv4 address ranges per pair*

<i>Pair</i>	<i>LANA</i>	<i>LANB</i>	<i>LANC</i>
<i>1</i>	<i>10.1.0.0/16</i>	<i>172.16.1.0/24</i>	<i>192.168.1.0/24</i>
<i>2</i>	<i>10.2.0.0/16</i>	<i>172.16.2.0/24</i>	<i>192.168.2.0/24</i>
<i>3</i>	<i>10.3.0.0/16</i>	<i>172.16.3.0/24</i>	<i>192.168.3.0/24</i>
<i>4</i>	<i>10.4.0.0/16</i>	<i>172.16.4.0/24</i>	<i>192.168.4.0/24</i>
<i>5</i>	<i>10.5.0.0/16</i>	<i>172.16.5.0/24</i>	<i>192.168.5.0/24</i>
<i>6</i>	<i>10.6.0.0/16</i>	<i>172.16.6.0/24</i>	<i>192.168.6.0/24</i>
<i>7</i>	<i>10.7.0.0/16</i>	<i>172.16.7.0/24</i>	<i>192.168.7.0/24</i>
<i>8</i>	<i>10.8.0.0/16</i>	<i>172.16.8.0/24</i>	<i>192.168.8.0/24</i>
<i>9</i>	<i>10.9.0.0/16</i>	<i>172.16.9.0/24</i>	<i>192.168.9.0/24</i>
<i>10</i>	<i>10.10.0.0/16</i>	<i>172.16.10.0/24</i>	<i>192.168.10.0/24</i>
<i>11</i>	<i>10.11.0.0/16</i>	<i>172.16.11.0/24</i>	<i>192.168.11.0/24</i>
<i>12</i>	<i>10.12.0.0/16</i>	<i>172.16.12.0/24</i>	<i>192.168.12.0/24</i>
<i>13</i>	<i>10.13.0.0/16</i>	<i>172.16.13.0/24</i>	<i>192.168.13.0/24</i>
<i>14</i>	<i>10.14.0.0/16</i>	<i>172.16.14.0/24</i>	<i>192.168.14.0/24</i>
<i>15</i>	<i>10.15.0.0/16</i>	<i>172.16.15.0/24</i>	<i>192.168.15.0/24</i>

## CHAPTER FIVE : TCP/UDP WEEK 10

### TCP/UDP

#### *Task 1: TCP in Netcat*

*To do this assignment we will use the Netcat tool which is provided in the Netkit. Netcat makes it possible to create and use TCP/UDP connections. If you want more info about Netcat you can consult Internet. To make this assignment we will reuse the net\_routing lab from the previous assignments. Let's start a chat session by connecting 2 netcat instances via a TCP connection.*

*To listen to the TCP connections, go to one of your simulated nodes (e.g. PC1A) and issue the following command:*

***nc -l -p <port\_nr>***

*This will make netcat listen to port number that you have specified in port\_nr and accept connections.*

*Note: Any port number would be ok, as long as it is not used by another application.*

*To establish a TCP connection you can issue the following command from another simulated node (e.g. PC1C)*

- ***nc <IP address of the "listening" node> <port\_nr of the "listening node">***

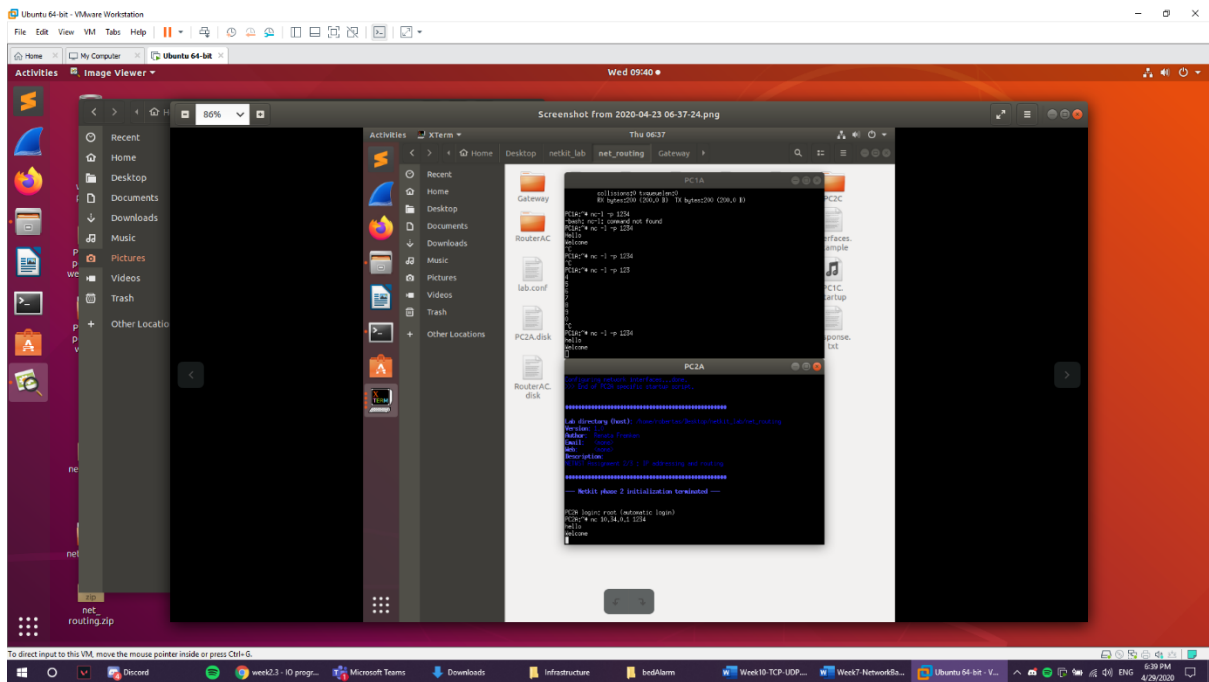
*This will make a TCP connection with the listening netcat instance. Now you can chat from one netcat instance to the another. Try it out!*

*Your task:*

- *Netcat can also be used to copy the contents of a file from one place (file, folder, computer) to another. Find out how and try it out.*

*Provide screenshots of the sending and receiving command.*





*Let's now build a basic one-page webserver using netcat.*

*As a first step create a textfile 'response.txt' with following content:*

```
HTTP/1.1 200 OK
Content-Type: text/html;
Content-Length: 12
Connection: close

Hello World!
```

*This is a proper HTTP response.*

*You are going to simulate HTTP server and HTTP client (browser) using netcat. You are going to use Gateway node for running HTTP client (browser), see section below about how to install [links](#) browser on Gateway before you start the exercise. You are going to use any node of your network (e.g. PC2C) to simulate HTTP server.*

*Your task:*

*Construct an appropriate netcat command to listen to a port on your HTTP server (e.g. PC2C) and send the contents of the file response.txt to the HTTP client (Gateway) when a connection is made to this port. That is roughly what a webserver does too. You can test it by entering the following URL in links browser on Gateway.*

[\*\*http://<IP ADDRESS OF WEBSERVER>:<port\\_nr>\*\*](http://<IP ADDRESS OF WEBSERVER>:<port_nr>)

*If everything works well the links should show the "Hello World" webpage.*

## Links installation:

Before starting this task, you have to install links text-mode web browser. Follow these steps:

- Go to your lab's Gateway directory. Create etc subdirectory and create there a `resolv.conf` text file with the following contents:  
`nameserver 8.8.8.8`  
`search localdomain`
- Start your lab. Now you should be able to connect to the Internet, so do the following installation on the Gateway node:  
`apt-get install links`

Now the `links` web browser should be available on your Gateway node. Watch out, once you stop your Gateway node, you have to do '`apt-get install links`' command again to install links.

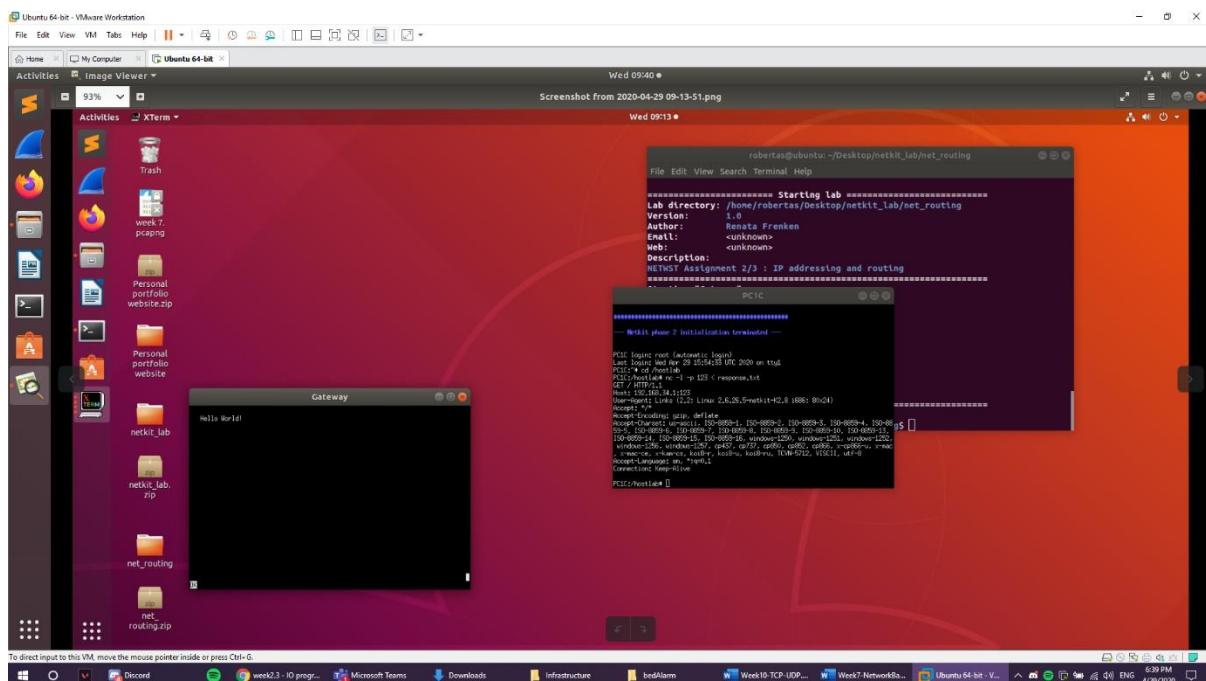
Note1: You can start links browser by issuing this command:

`links`

To be able to enter the URL in links browser press "G".

Note2 : You can put the `response.txt` file in your `net_routing` directory **before** start of the lab (`lstart` command). **After** starting the lab, you can find this file in the `/hostlab` directory of your node.

Provide a screenshot of the netcat command you used and of the links browser output.

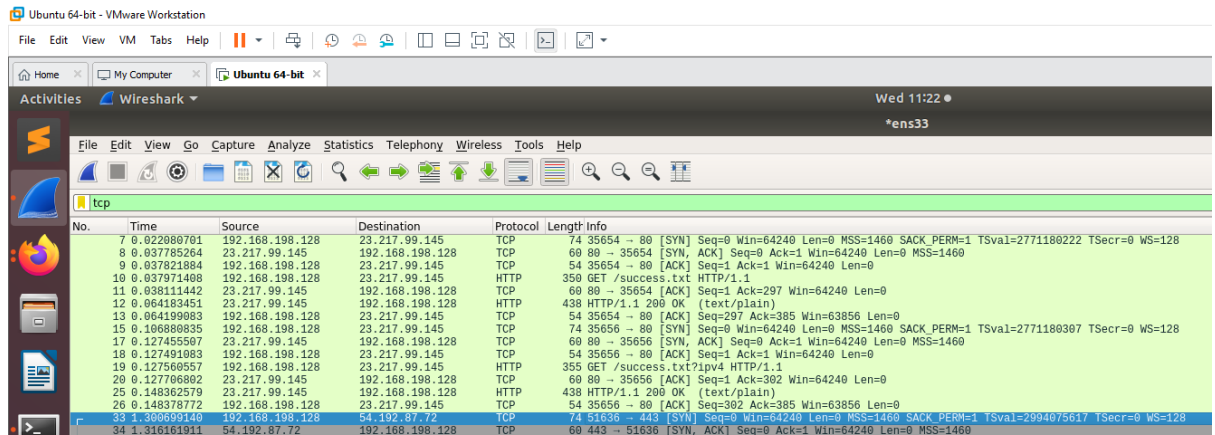


## Task 2: Find 2 TCP uses

Think about two different scenarios for TCP use that you can simulate (you can do this on your own PC, so you don't need Ubuntu for this). Start a Wireshark trace for both scenarios.

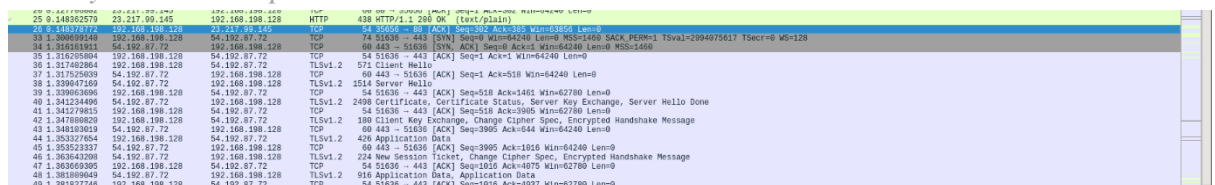
Describe the chosen scenarios and a proof of TCP use in them by attaching a Wireshark trace showing TCP packets.

1. To get access to a website the TCP packets are used



No.	Time	Source	Destination	Protocol	Length	Info
7	0.022080701	192.168.198.128	23.217.99.145	TCP	74	35654 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2771180222 TSecr=0 WS=128
8	0.037785264	23.217.99.145	192.168.198.128	TCP	60	80 → 35654 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
9	0.037821884	192.168.198.128	23.217.99.145	TCP	54	35654 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
10	0.037971408	192.168.198.128	23.217.99.145	HTTP	350	GET /success.txt HTTP/1.1
11	0.038111442	23.217.99.145	192.168.198.128	TCP	60	80 → 35654 [ACK] Seq=1 Ack=297 Win=64240 Len=0
12	0.064183451	23.217.99.145	192.168.198.128	HTTP	438	HTTP/1.1 200 OK (text/plain)
13	0.064199083	192.168.198.128	23.217.99.145	TCP	54	35654 → 80 [ACK] Seq=297 Ack=385 Win=63856 Len=0
15	0.106880835	192.168.198.128	23.217.99.145	TCP	74	35654 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2771180387 TSecr=0 WS=128
17	0.127455507	23.217.99.145	192.168.198.128	TCP	60	80 → 35656 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
18	0.127491083	192.168.198.128	23.217.99.145	TCP	54	35656 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
19	0.127506057	192.168.198.128	23.217.99.145	HTTP	355	GET /success.txt?ip=v4 HTTP/1.1
20	0.127706002	23.217.99.145	192.168.198.128	TCP	60	80 → 35656 [ACK] Seq=1 Ack=302 Win=64240 Len=0
25	0.148362579	23.217.99.145	192.168.198.128	HTTP	438	HTTP/1.1 200 OK (text/plain)
26	0.148378772	192.168.198.128	23.217.99.145	TCP	54	35656 → 80 [ACK] Seq=302 Ack=385 Win=63856 Len=0
33	1.30699140	192.168.198.128	54.192.87.72	TCP	74	51636 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2994075617 TSecr=0 WS=128
34	1.316161911	54.192.87.72	192.168.198.128	TCP	60	443 → 51636 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460

2. To communicate with the server of a website or online application and receive access or deny access TCP packets are used.



No.	Time	Source	Destination	Protocol	Length	Info
438	1.316161911	192.168.198.128	54.192.87.72	HTTP	438	HTTP/1.1 200 OK (text/plain)
439	1.316161911	192.168.198.128	54.192.87.72	TCP	54	35656 → 80 [ACK] Seq=302 Ack=385 Win=63856 Len=0
440	1.316161911	192.168.198.128	54.192.87.72	TCP	74	51636 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2994075617 TSecr=0 WS=128
441	1.316161911	54.192.87.72	192.168.198.128	TCP	60	443 → 51636 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
442	1.316161911	192.168.198.128	54.192.87.72	TCP	54	51636 → 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
443	1.316161911	192.168.198.128	54.192.87.72	TLSv1.2	571	Client Hello
444	1.316161911	54.192.87.72	192.168.198.128	TCP	60	443 → 51636 [ACK] Seq=1 Ack=518 Win=64240 Len=0
445	1.316161911	54.192.87.72	192.168.198.128	TLSv1.2	514	Server Hello
446	1.316161911	192.168.198.128	54.192.87.72	TCP	54	51636 → 443 [ACK] Seq=518 Ack=518 Win=62780 Len=0
447	1.316161911	192.168.198.128	54.192.87.72	TLSv1.2	2408	Certificate, Certificate Status, Server Key Exchange, Server Hello Done
448	1.316161911	192.168.198.128	54.192.87.72	TCP	54	51636 → 443 [ACK] Seq=518 Ack=595 Win=62780 Len=0
449	1.316161911	192.168.198.128	54.192.87.72	TLSv1.2	590	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
450	1.316161911	192.168.198.128	54.192.87.72	TCP	60	443 → 51636 [ACK] Seq=595 Ack=1016 Win=64240 Len=0
451	1.316161911	192.168.198.128	54.192.87.72	TLSv1.2	426	Application Data
452	1.316161911	192.168.198.128	54.192.87.72	TCP	60	443 → 51636 [ACK] Seq=595 Ack=1016 Win=64240 Len=0
453	1.316161911	192.168.198.128	54.192.87.72	TLSv1.2	224	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
454	1.316161911	192.168.198.128	54.192.87.72	TLSv1.2	54	51636 → 443 [ACK] Seq=1016 Ack=6075 Win=62780 Len=0
455	1.316161911	192.168.198.128	54.192.87.72	TLSv1.2	916	Application Data, Application Data
456	1.316161911	192.168.198.128	54.192.87.72	TCP	54	51636 → 443 [ACK] Seq=1016 Ack=4037 Win=62780 Len=0

Choose one of the 2 scenarios traces and browse it in the Wireshark. Select 'Statistics > Flow Graph' and then choose flowtype 'TCP flow' to draw a Sequence Diagram of the TCP message interaction that you see in Wireshark.

Provide a screenshot of this Flow Graph.



Explain what is happening during various stages (begin, middle, end) of the communication. Explain SYN, SYNACK and ACK. Explain the Len, Seq and Ack numbers.

I captured packets from connecting to <https://www.cnet.com/news/>, the connection was started by three way handshake and it was also terminated by three way handshake

### Task 3: Find 2 UDP uses

Think about two different scenarios for UDP use that you can simulate (you can do this on your own PC, so you don't need Ubuntu for this). Start a Wireshark trace for both scenarios.

Describe the chosen scenarios and a proof of UDP use in them by attaching a Wireshark trace showing UDP packets.

- A. I went to Youtube and I got a DNS server request to open the url [www.youtube.com](http://www.youtube.com)  
DNS server uses UDP packets to execute requests

2781	35.37491878	192.168.19.132	192.168.19.2	DNS	100 Standard query 0xea10 AAAA pagead2.googlesyndication.com OPT
2782	35.377621476	192.168.19.132	192.168.19.2	DNS	86 Standard query 0xd8f1 A www.youtube.com OPT
2783	35.377529232	192.168.19.132	192.168.19.2	DNS	86 Standard query 0x5a4e AAAA www.youtube.com OPT
2784	35.379629975	192.168.19.132	216.58.208.110	TLSv1.3	155 Application Data
2785	35.379881370	216.58.208.110	192.168.19.132	TCP	60 443 → 35130 [ACK] Seq=2118576 Ack=5200 Win=64240 Len=0
2786	35.392716231	192.168.19.2	192.168.19.132	DNS	116 Standard query response 0x5a85 A pagead2.googlesyndication.com A 0.0.0
2787	35.393410773	192.168.19.2	192.168.19.132	DNS	128 Standard query response 0xea10 AAAA pagead2.googlesyndication.com AAAA
2788	35.394218519	192.168.19.2	192.168.19.132	DNS	235 Standard query response 0xd8f1 A www.youtube.com CNAME youtube-ui.l.go
2789	35.397698864	192.168.19.2	192.168.19.132	DNS	151 Standard query response 0x5a4e AAAA www.youtube.com CNAME youtube-ui.l
2790	35.415231440	216.58.208.110	192.168.19.132	TLSv1.3	208 Application Data

- B. Also the NTP client uses UDP packets to execute requests on clock transactions

4	6.007639764	91.189.89.198	192.168.19.132	NTP	90 NTP Version 4, server
5	8.173267247	fe80::d7f2:dd28:a09...	ff02::fb	MDNS	107 Standard query 0x0000 PTR _ipps._tcp.local, "QM" question
6	8.173893896	192.168.19.132	224.0.0.251	MDNS	87 Standard query 0x0000 PTR _ipps._tcp.local, "QM" question
12	16.570707913	192.168.19.132	192.168.19.2	DNS	95 Standard query 0xa38b A detectportal.firefox.com OPT

### Task 4 (Optional): TCP SYN Flooding

Read an explanation of TCP SYN Flooding at [http://en.wikipedia.org/wiki/SYN\\_flood](http://en.wikipedia.org/wiki/SYN_flood) or from some other source.

In this task you're going to simulate this kind of DDOS attack that uses vulnerability of TCP protocol.

For this experiment you can reuse net\_routing lab. You can use for example the PC1B node as the victim and RouterAC node as an attacker.

To be able to wait for the TCP connections, use netcat command to wait for the TCP connections on a specific port at the victim node.

To simulate TCP SYN flood traffic from the attacker node, you can use the "hping3" tool which is part of your netkit nodes.

*Before you start the attacker command, don't forget to sniff the traffic with tcpdump command and write the output to a pcap file:*

```
tcpdump -w <filename>.pcap -s 0
```

*Tip: If you want to run tcpdump or any other command in the background you can do it by specifying "&" at the end of the command. In this way you can use your Linux prompt again. To see all your background processes use "jobs" command and to put a job in foreground again use : "fg <job\_number>".*

*A command to be issued at the attacker node can look like:*

```
hping3 --rand-source <IP_ADDRESS_OF_VICTIM> --flood -S -L 0 -p  
<PORT_NR_OF_VICTIM>.
```

*Wait about 10 seconds, stop hping3 and tracing.*

*Now you should be able to analyze the trace. You should be able to see spoofed source IP address.*

*Analyze your trace. Find out how many SYNs, SYN+ACKs and ACKs you can see. Explain what do these numbers tell you about SYN attack.*

*Consult internet to find out how another transport protocol – SCTP - solves TCP SYN flooding problem. Give a short explanation of how it is implemented in SCTP.*

## **CHAPTER SIX :**

### **CONCLUSION**

My week seven to week Ten study assignments for infrastructure are all compiled and contained in this document. I enjoyed working together with my groupmate Robertas and Group 34. It has been not easy to meet up as it would be if school was open but I am glad we always made time to go into teams and do the assignments and I am grateful for Mr. Vladimir who have been always there to help us whenever we had questions to ask him, or when we sent him emails he was responsive. I appreciate being introduced to the basics of networking in this semester and I am thankful for my good teacher who was still available in this Corona era where classes have been closed.

## **CHAPTER SEVEN :**

### **PERSONAL REFRACTION**

I appreciate so much having been introduced to the basics of networking , but I wish there was no corona all over the world , and I get the basics more practically , we used to do the work using Wireshark and Ubuntu and wished if the school laboratories were open for us to go in there and have a more feel of what I was doing. I have learnt new things I never knew but I still miss that laboratory feel. But it stands now I say I am no more illiterate about networking at least I have the beginners knowledge and I am proud of myself about that.