# name Investigation of hardware Week 7

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)

Describe the steps you took and provide screenshot of the started node.

1. Download netkit files

Go to <http://wiki.netkit.org/index.php/Download_Official>

* “netkit-2.8.tar.bz2”
* “netkit-filesystem-i386-F5.2.tar.bz2”
* “netkit-kernel-i386-K2.8.tar.bz2”

1. Open terminal and unpack downloaded files using following commands (assuming you’re in the “~$” directory)

* Use this command “tar -xjSf Downloads/netkit-2.8.tar.bz2”
* Use this command “tar -xjSf Downloads/netkit-filesystem-i386-F5.2.tar.bz2”
* Use this command “tar -xjSf Downloads/netkit-kernel-i386-K2.8.tar.bz2”

1. Configuration of variables (assuming you’re still in the “~$” directory)

* The “pwd” command results in current path: /home/<your linux username>
* Use this command “export NETKIT\_HOME=/home/<your username>/netkit”
* Use this command “export MANPATH=:$NETKIT\_HOME/man”
* Use this command “export PATH=$NETKIT\_HOME/bin:$PATH”
* Use this command “. $NETKIT\_HOME/bin/netkit\_bash\_completion”
* Use this command “gedit .bashrc/”. This command opens the bash shell file to save the following variables:
* Insert “export NETKIT\_HOME=/home/<your username>/netkit” at the end of the bashrc file.
* Insert “export MANPATH=:$NETKIT\_HOME/man” at the end of the bashrc file.
* Insert “export PATH=$NETKIT\_HOME/bin:$PATH” at the end of the bashrc file.
* Insert “. $NETKIT\_HOME/bin/netkit\_bash\_completion” at the end of the bashrc file.

1. Check your configuration (assuming you’re still in the “~$” directory)

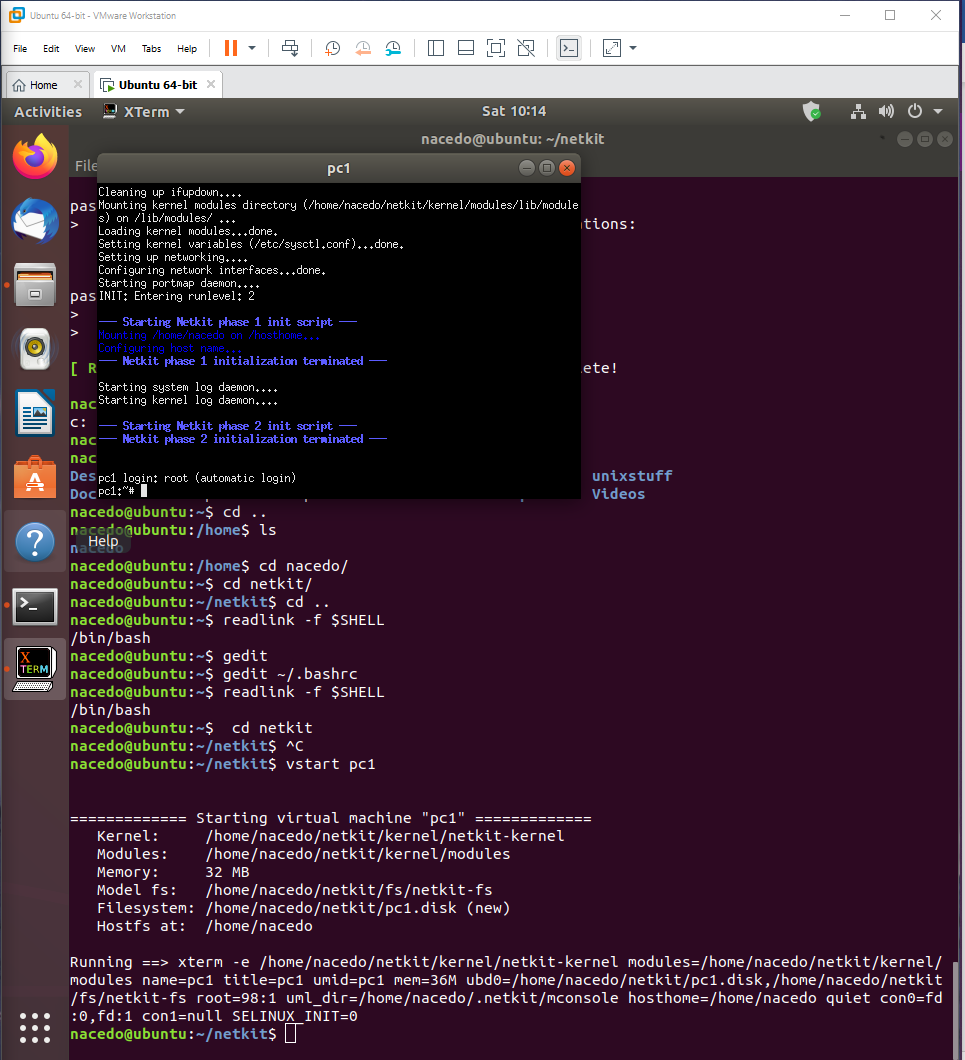
* “cd netkit”
* Use the command “./check\_configuration.sh” in the terminal
* Use the command “sudo apt-get install lib32ncurses5”. Press key “y” on keyboard. If you see this message “Do you want to continue? [Y(es)/n(o)]”.
* Use the command “sudo apt-get install libc6-i386”. Press key “y” on keyboard. If you see this message “Do you want to continue? [Y(es)/n(o)]”
* Use the command “sudo apt-get install xterm”. Press key “y” on keyboard. If you see this message “Do you want to continue? [Y(es)/n(o)]”
* Use the command “./check\_configuration.sh” to check if netkit has install successfully

1. Run netkit (assuming you’re still in the “~/netkit$” directory)

Use the command “vstart pc1” (starts virtual machine)

Use the command “vlist” (lists all virtual machines)

Use the command “vhalt -r pc1” (should stop the virtual machine)



Task 2: TCP/IP Layers in Wireshark

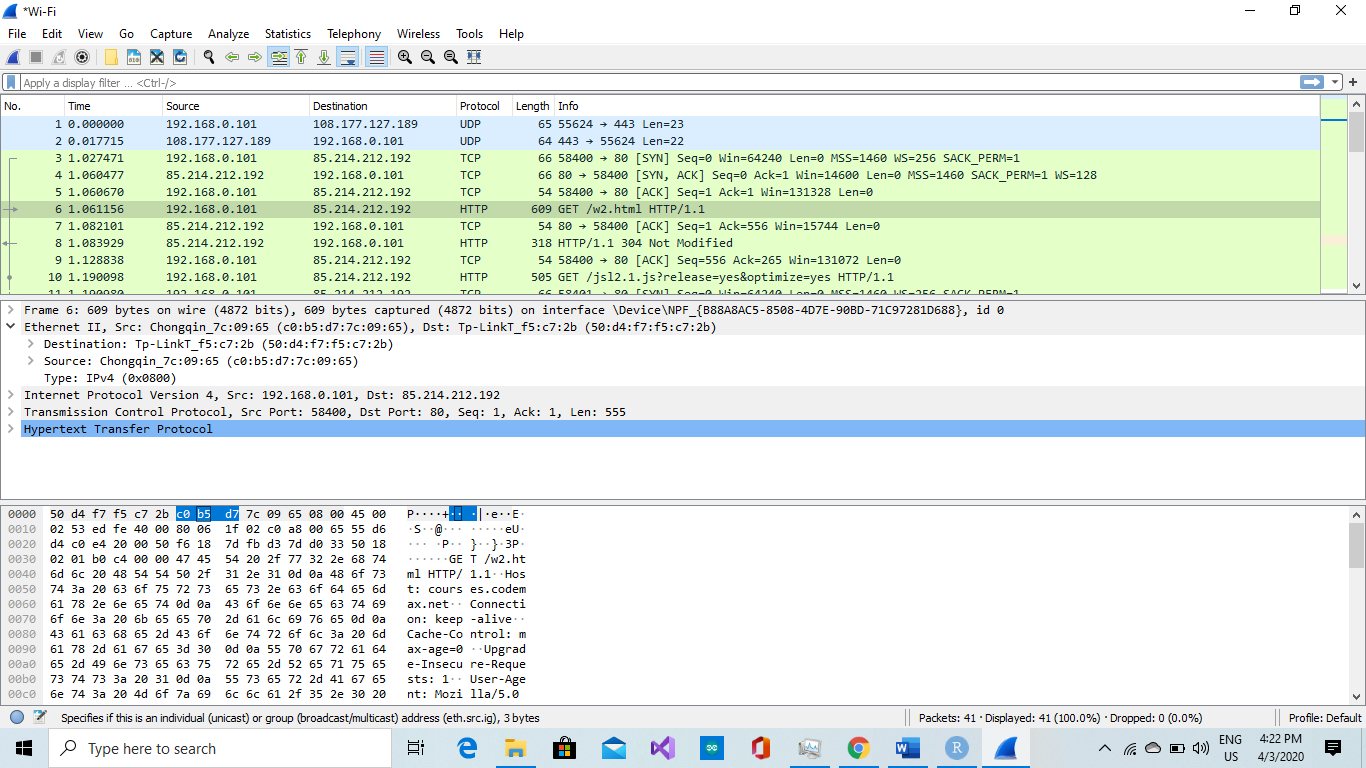
Find a Wireshark Tutorial on the web. Run Wireshark.

Start capturing the network traffic. To generate HTTP traffic, go to <http://courses.codemax.net/w2.html> web browser. 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?

Source MAC address: c0:b5:d7:7c:09:65  
Destination MAC address: 50:d4:f7:f5:c7:2b

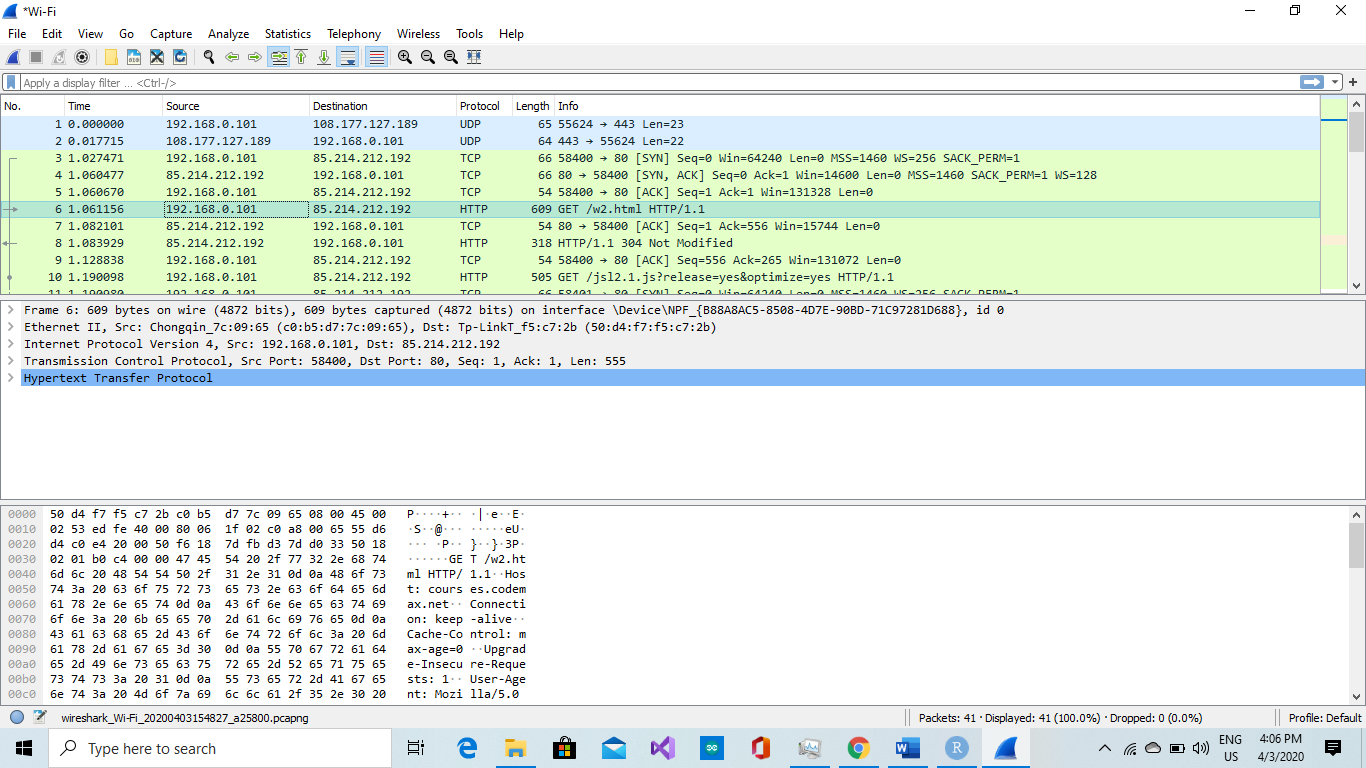
Provide a screenshot below with the Wireshark snapshot and highlight these addresses:



* What is the source and destination IP address of this HTTP packet?

Source IP address : 192.168.0.101  
Destination IP address : 85.214.212.192

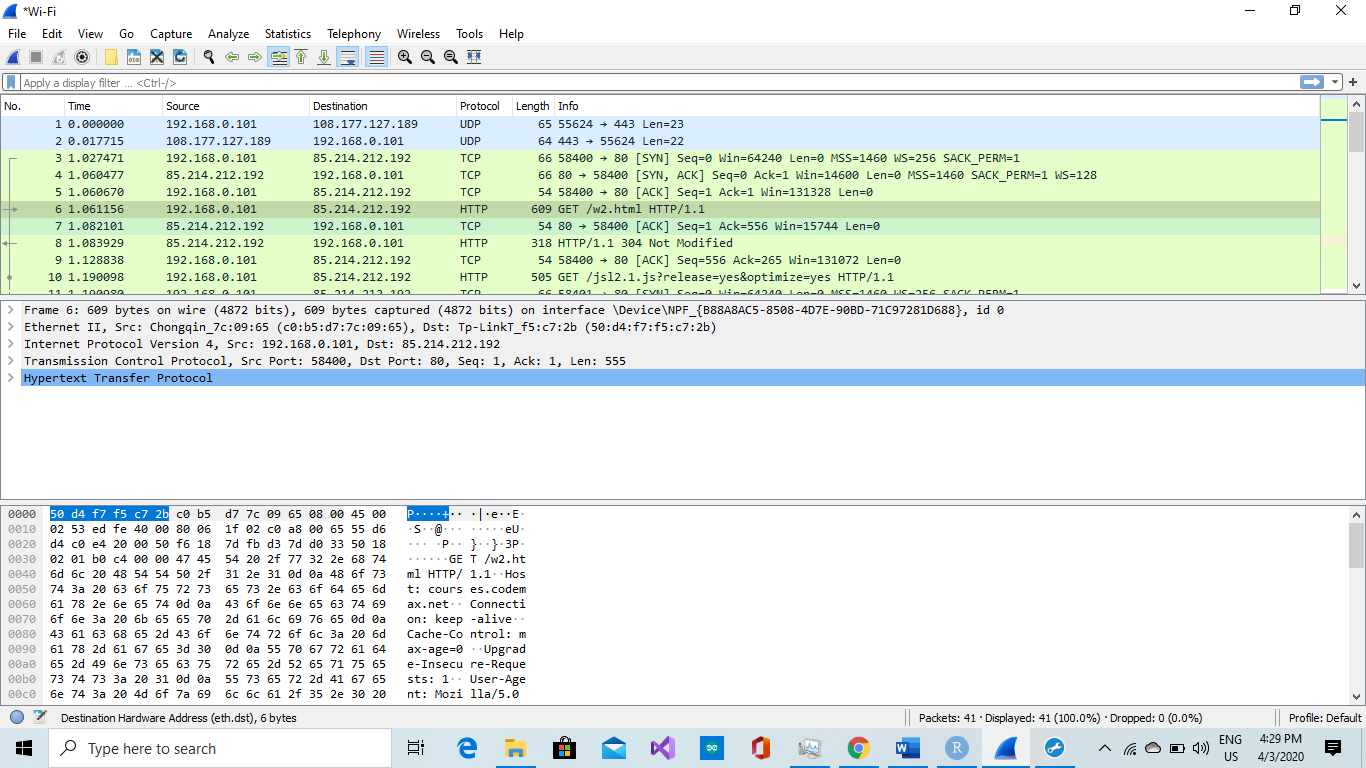
Provide a screenshot below with the Wireshark snapshot and highlight these addresses:



* What is the source and destination port of this HTTP packet? Provide a screenshot to prove it

Source port : 58400  
Destination port: 80

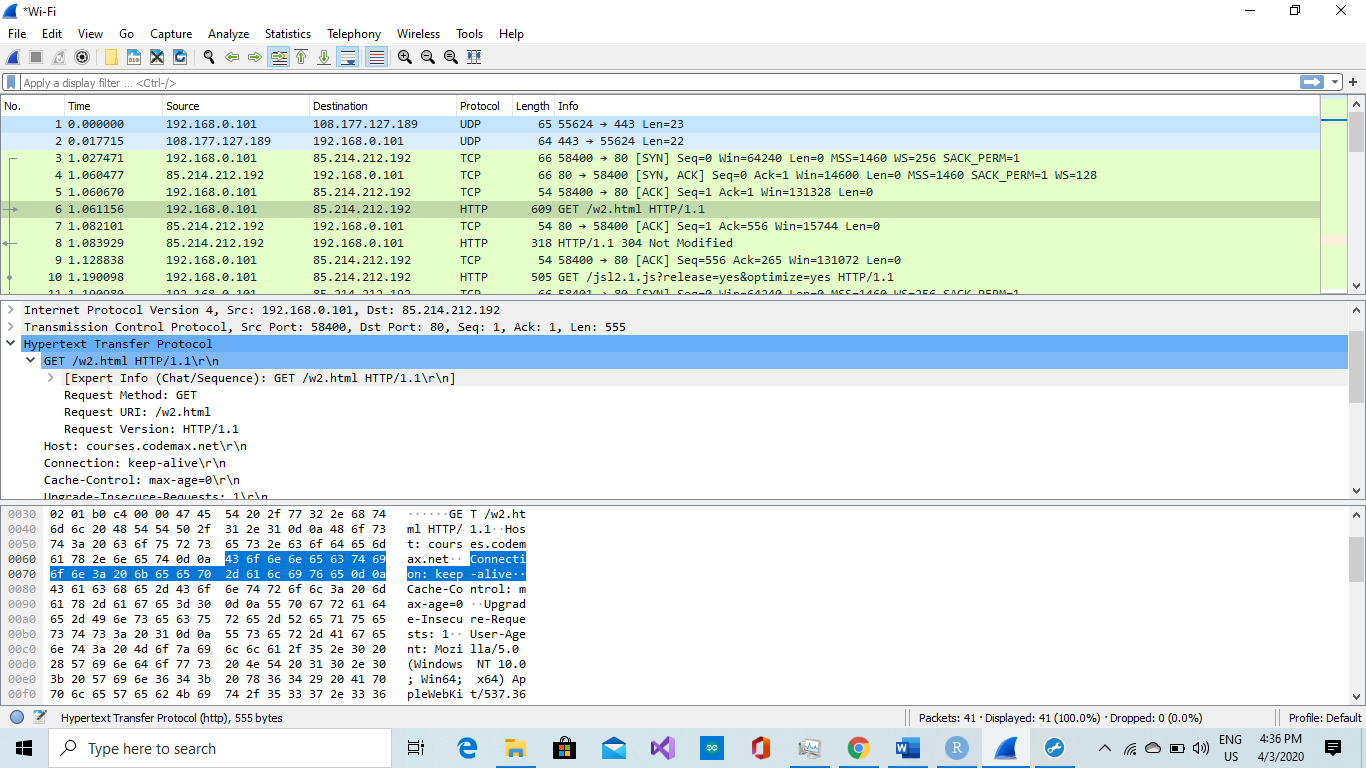
Provide a screenshot below with the Wireshark snapshot and highlight these addresses:



* What is the host name of this HTTP Get packet?

Host name: courses.codemax.net

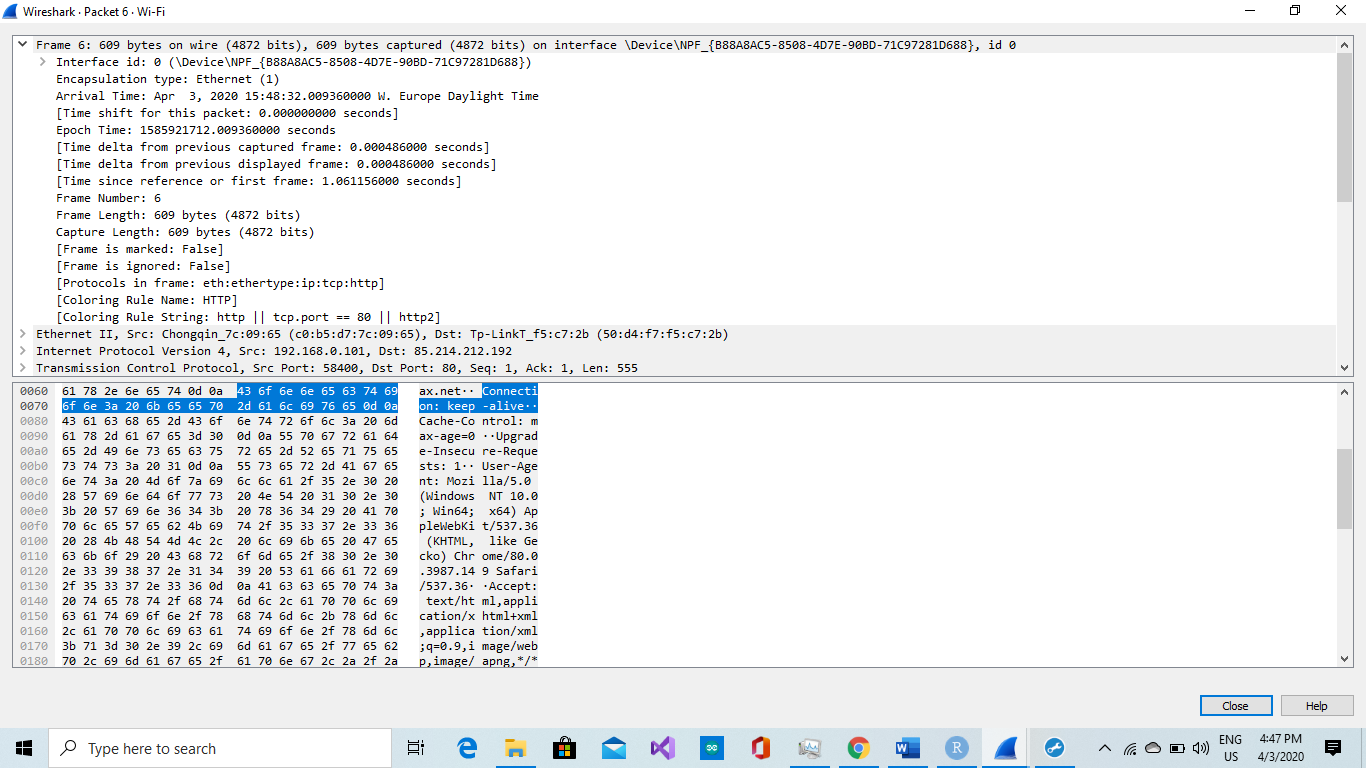
Provide a screenshot below with the Wireshark snapshot and highlight the host name:



* Find the HTTP Response belonging to the HTTP Get packet. How much time elapsed between the HTTP Get and HTTP response?

Time elapsed: 1.061156000 seconds

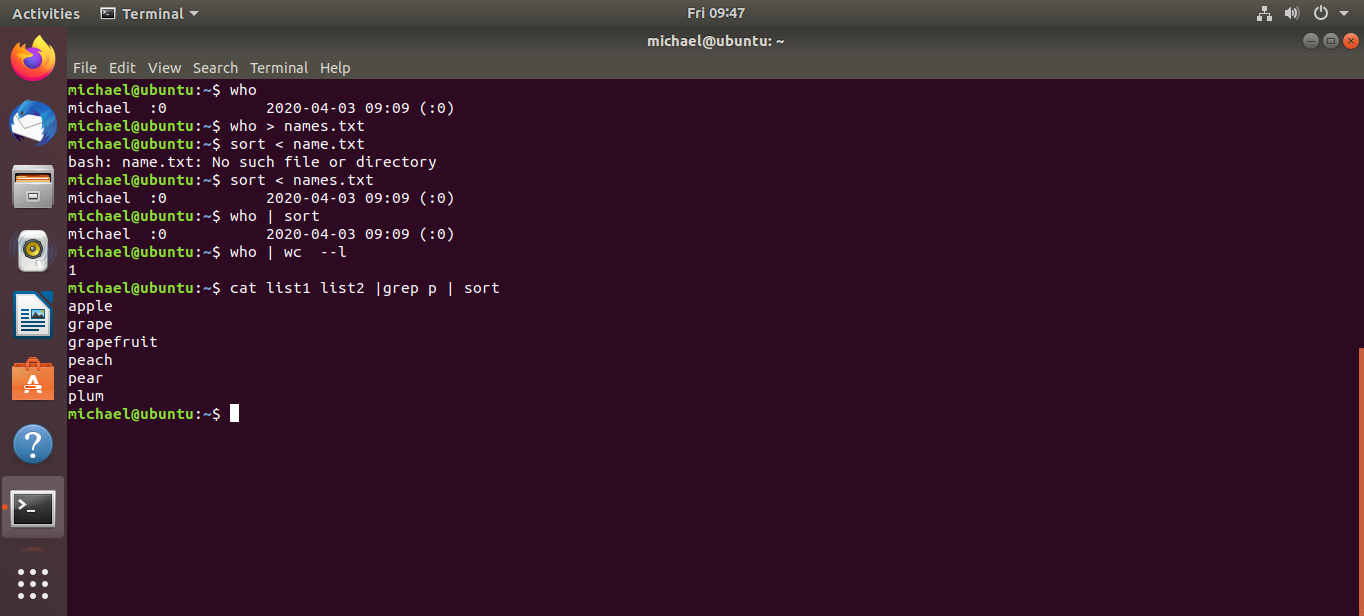
Provide a screenshot below with the Wireshark snapshot and highlight the elapsed time:

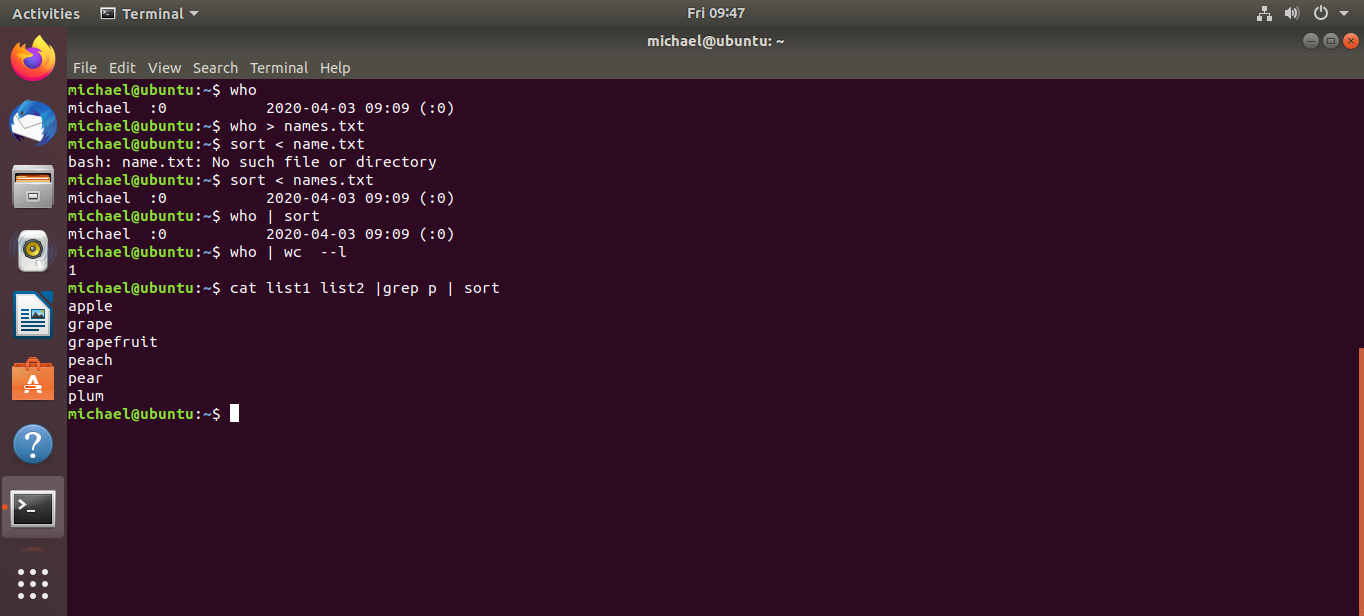


Task 3: Do Linux Tutorial

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



Exercise 3b

# Linux, Static IP address / Subnet configuration Week 8

Linux, Static IP address/subnets configuration

**Task 1a**: Do Linux Tutorial

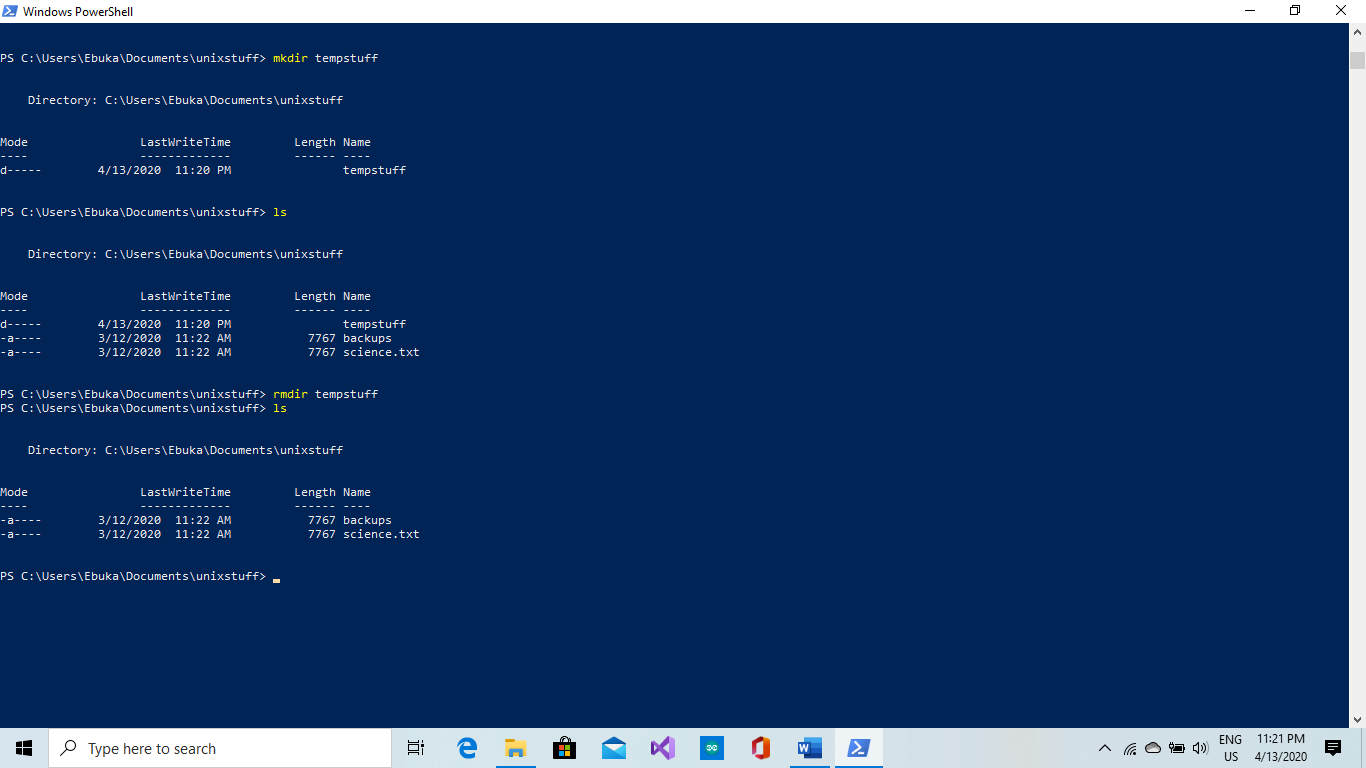
Go to <http://www.ee.surrey.ac.uk/Teaching/Unix/unix2.html> and do the 2nd basic Unix tutorial.

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.



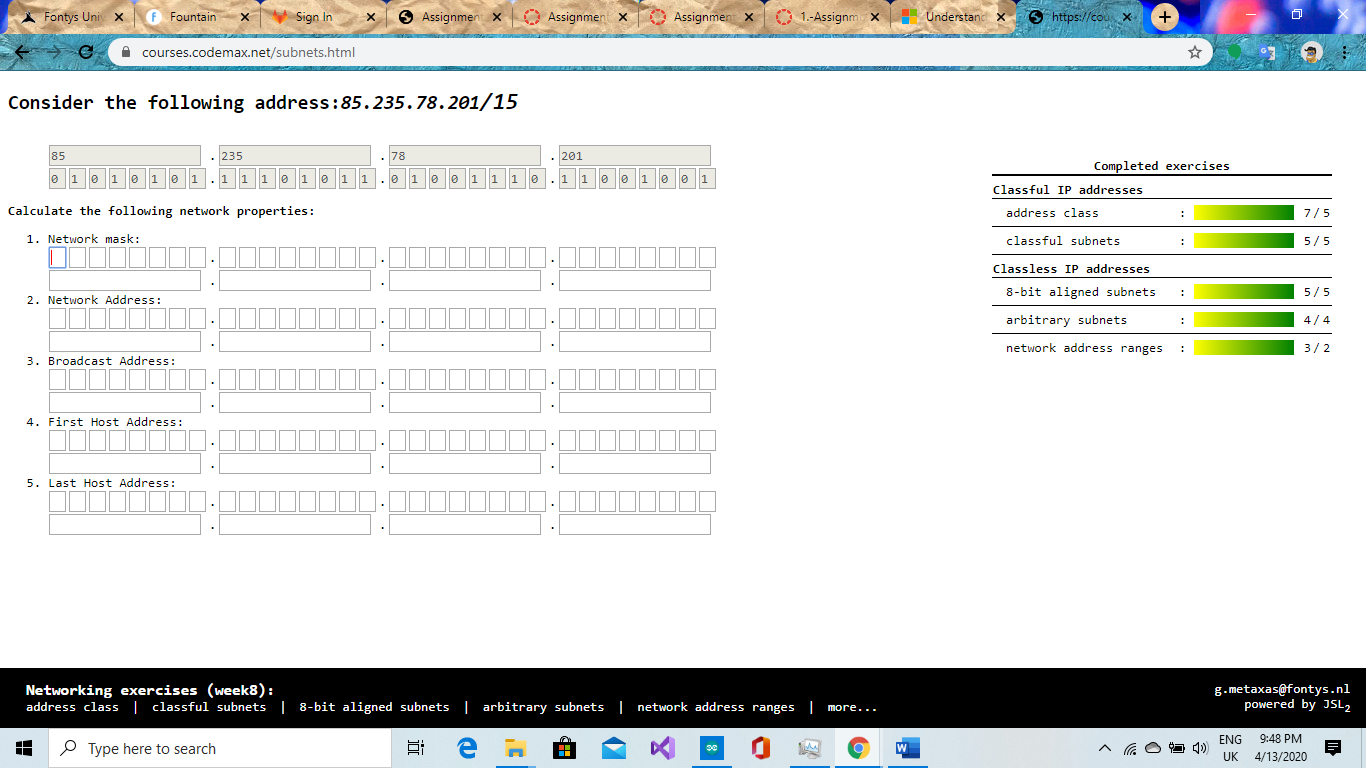
2b.



**Task 1b**: Networking exercise

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

Provide screenshots of all exercises.



**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.

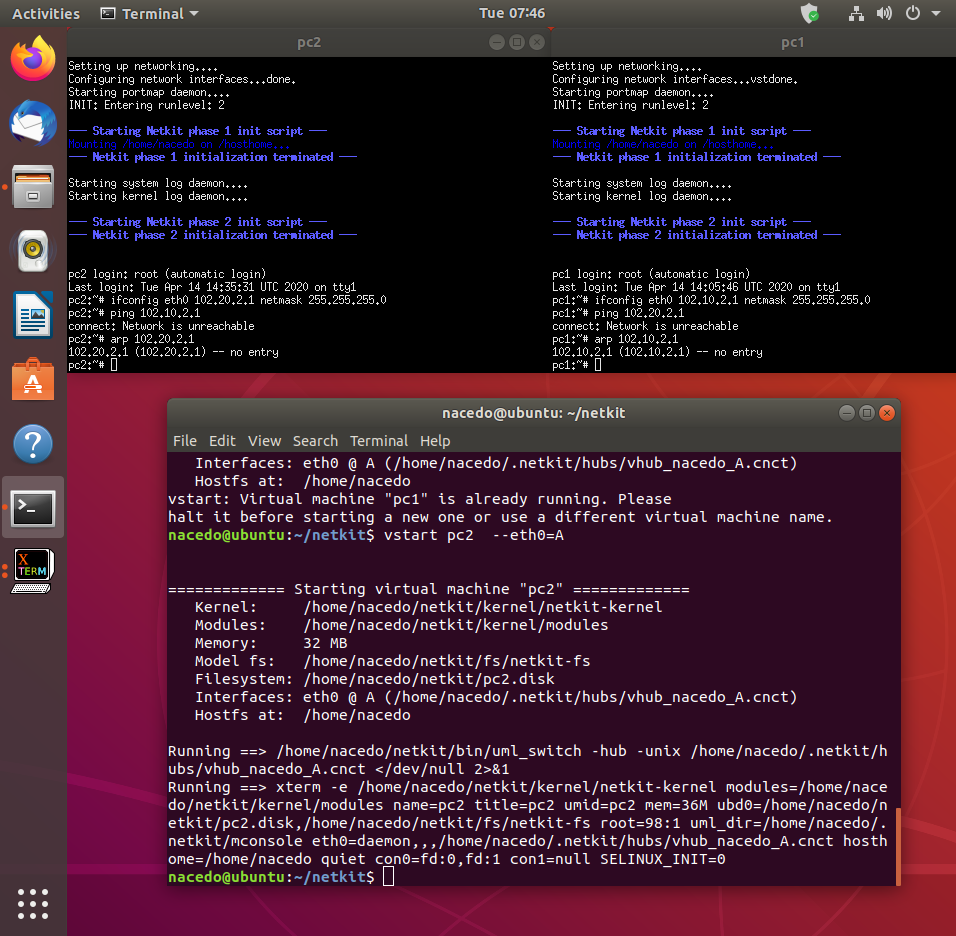
Try the following configurations:

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.



The ping wasn’t successful because the ip addresses are not in the same range.

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

“Arp 102.20.2.1” , “Arp 102.10.2.1”. The arp entries are sill empty because there was no connection established between the nodes.

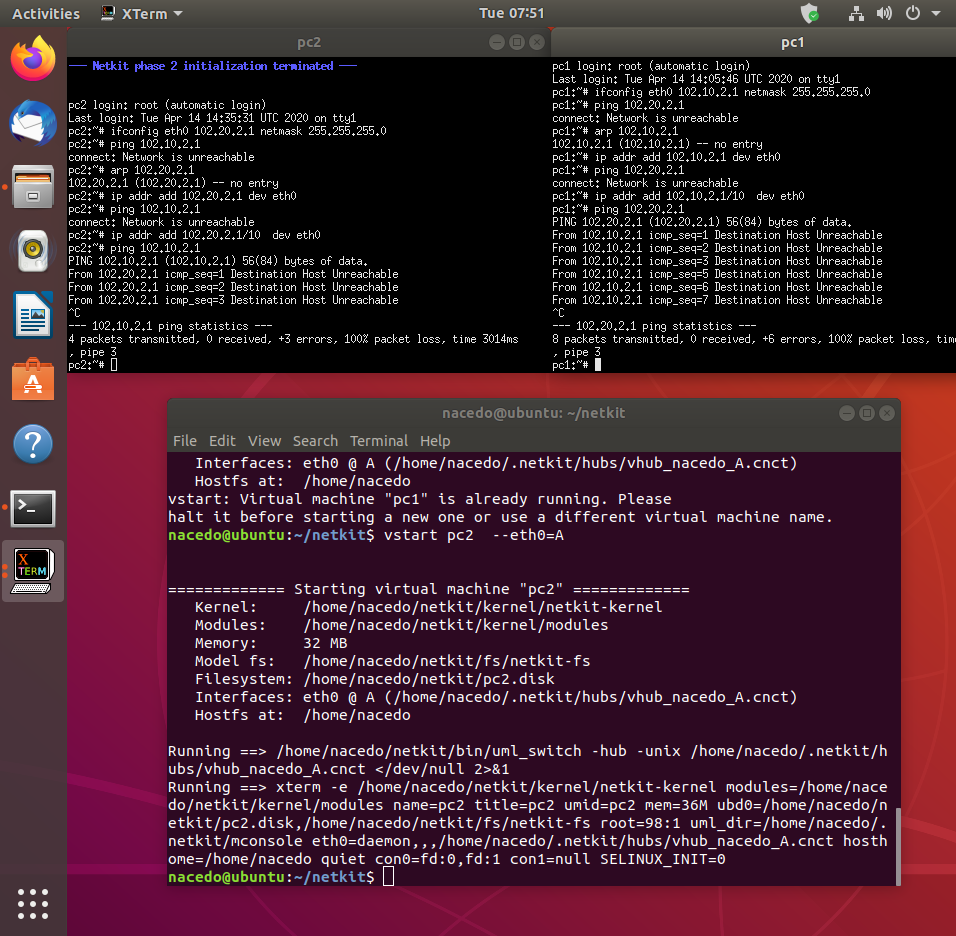
B) Configure the IP addresses of the 2 nodes by using the “ip” command explained in the theory lesson.

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.

What is the result of the ping? Can you explain it? Provide a screenshot of your configured interfaces.

The ranges of the ip addresses are wrong that’s why the nodes cannot communicate.

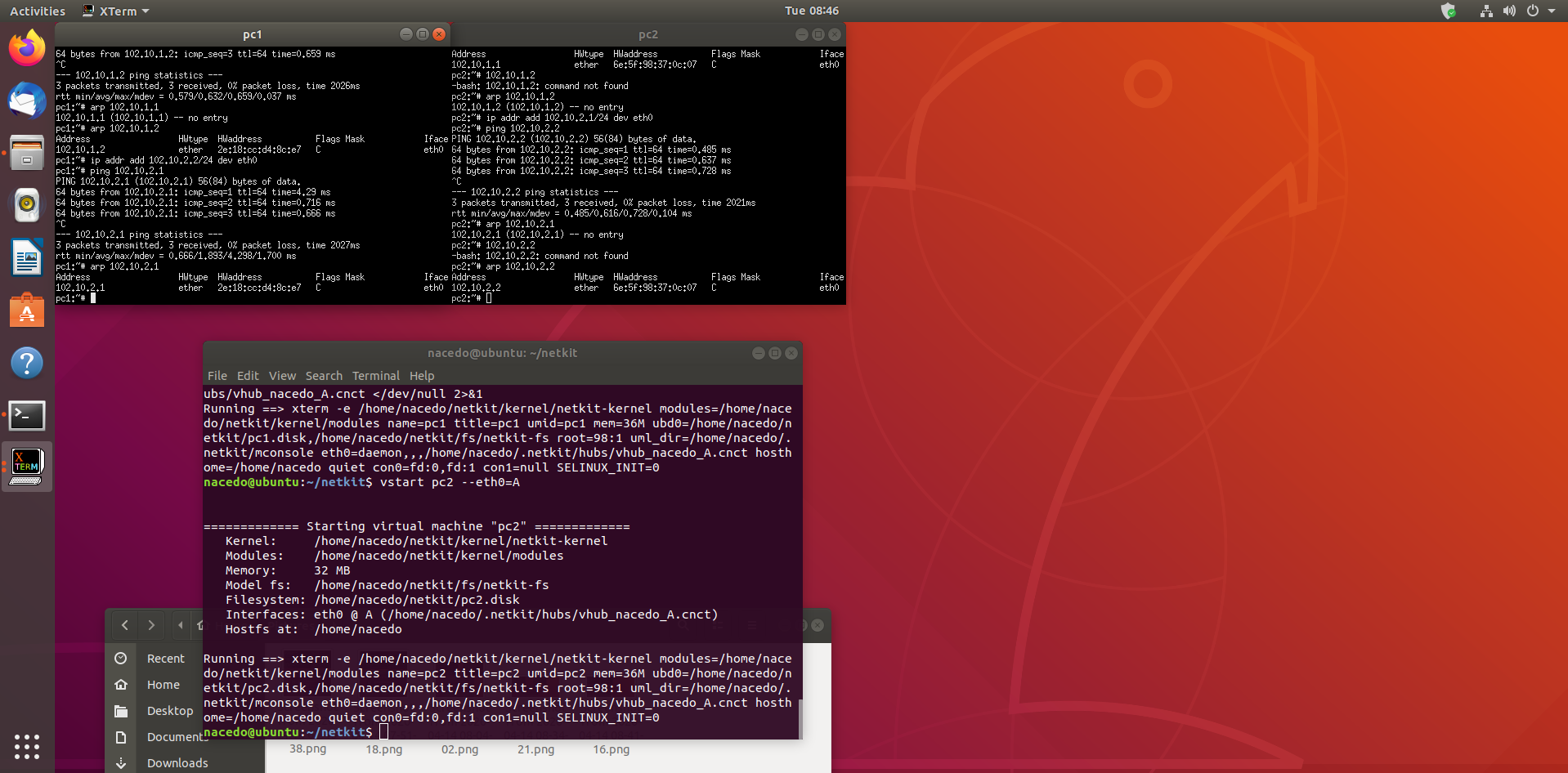


1. Look at the ARP entries of your Node1 and Node2. Which ARP entries are there?

The arp entries are registered as incomplete. The ip addresses “102.20.2.1”, “102.10.2.1” are shown in the register.

C) Configure both nodes to have a subnet mask 255.255.255.0, and change the IP address of Node2 in such a way that the ping between them is successful.

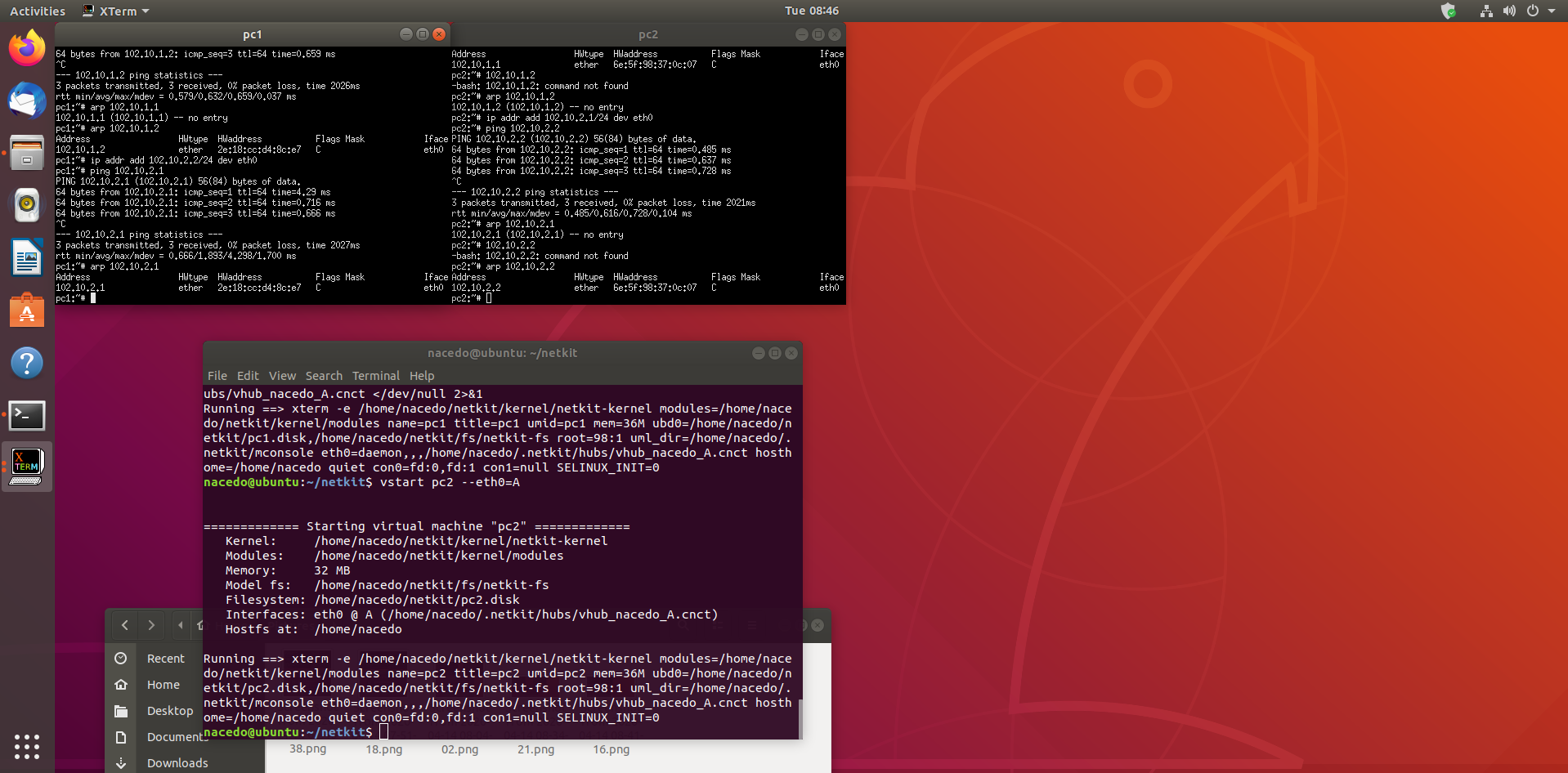
1. Provide a screenshot of your configuration and successful ping.



1. 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?

The communication between both nodes was successful because the arp situation is not incomplete anymore.

“arp -d 102.10.2.2” and “arp -d 102.10.2.1”



**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.

1. 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 LANA
2. PC1B, PC2B and RouterBC are connected to LANB
3. PC1C, PC2C, RouterBC, RouterAC and Gateway are connected to LANC
4. Gateway is connected to LANC 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.



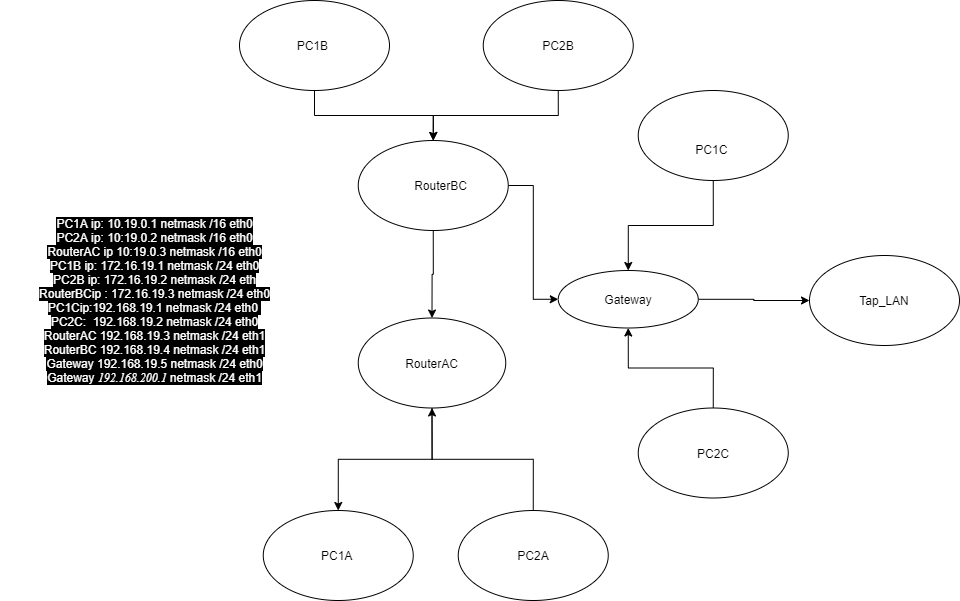
**Provide the network drawing** of your lab network you can use <https://app.diagrams.net/> and screenshots of the pings which are possible WITHIN LANA, LANB and LANC (PC1A to PC2A, PC1B to PC2B and so on). When creating the network drawing, don’t forget to mention the IP addresses/subnet masks for all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

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.

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 |





Every node can connect to the nodes within the same collision domain.

**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 : 122.33.196.0/24
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 : 163.249.223.128/25
2. First Host : 163.249.223.129
3. Last Host :163.249.223.254
4. Broadcast Address :163.249.223.255

**WEEK 9**

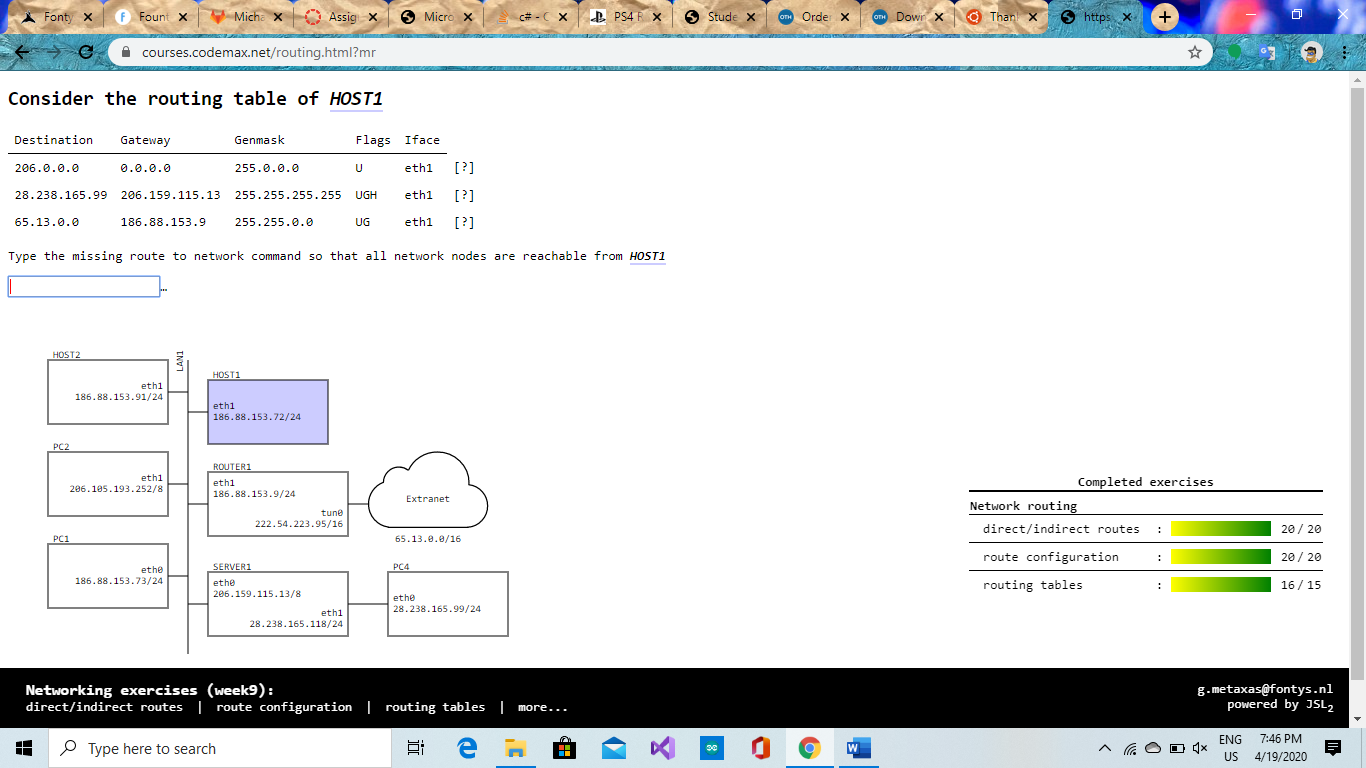
# IP Routing

Task 1a: Online exercises

Complete **all** the online exercises in the following URL and provide a screenshot as evidence

<https://courses.codemax.net/w9.html>

[Place here the screenshot]



Task 1b: 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.

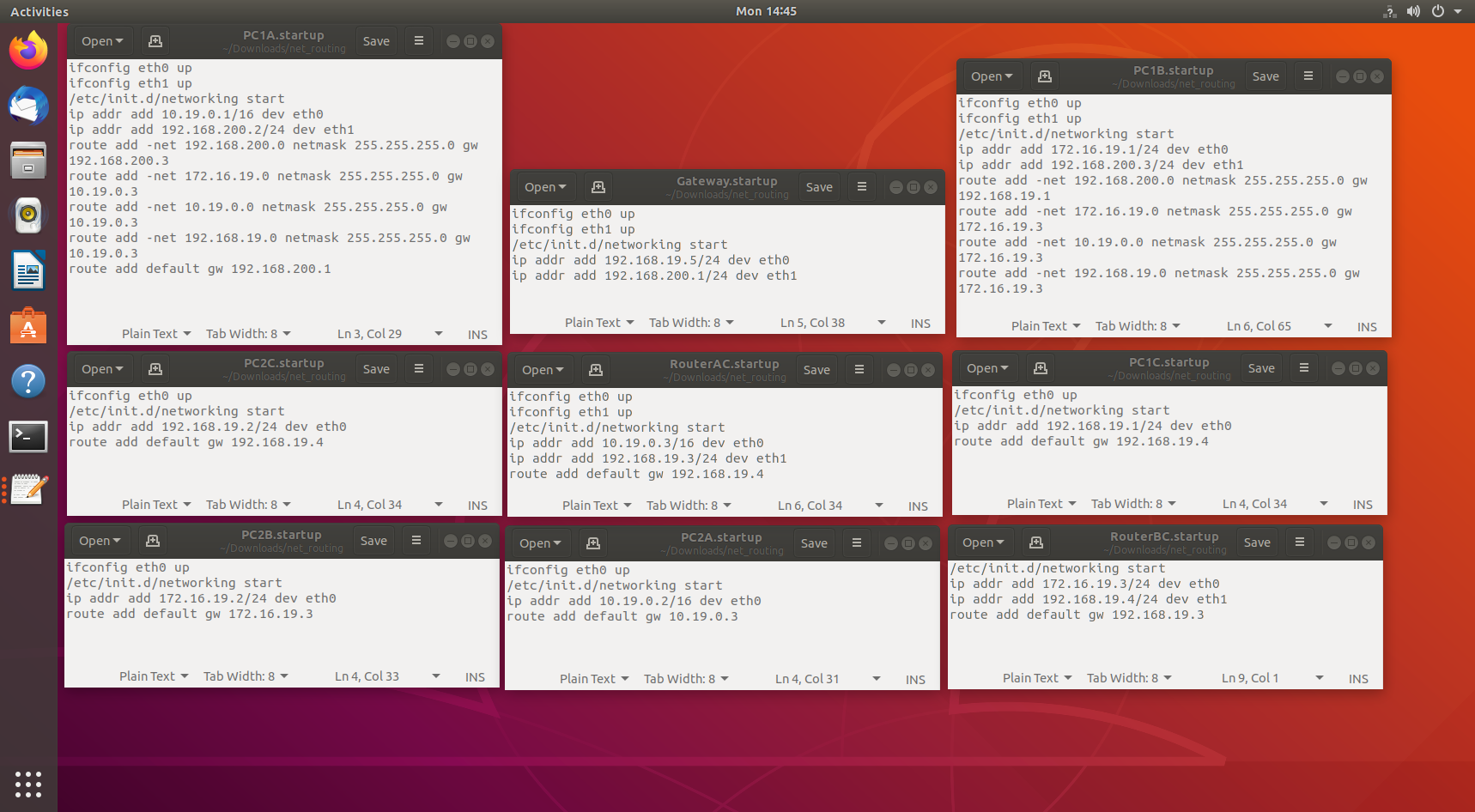
Your task is adding routing information to your nodes in such a way, that every node of your network should be able to ping any other node of your network. The routes should be optimal, so the shortest path from node to node should be used. To implement routing, you’ll have to use different types of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

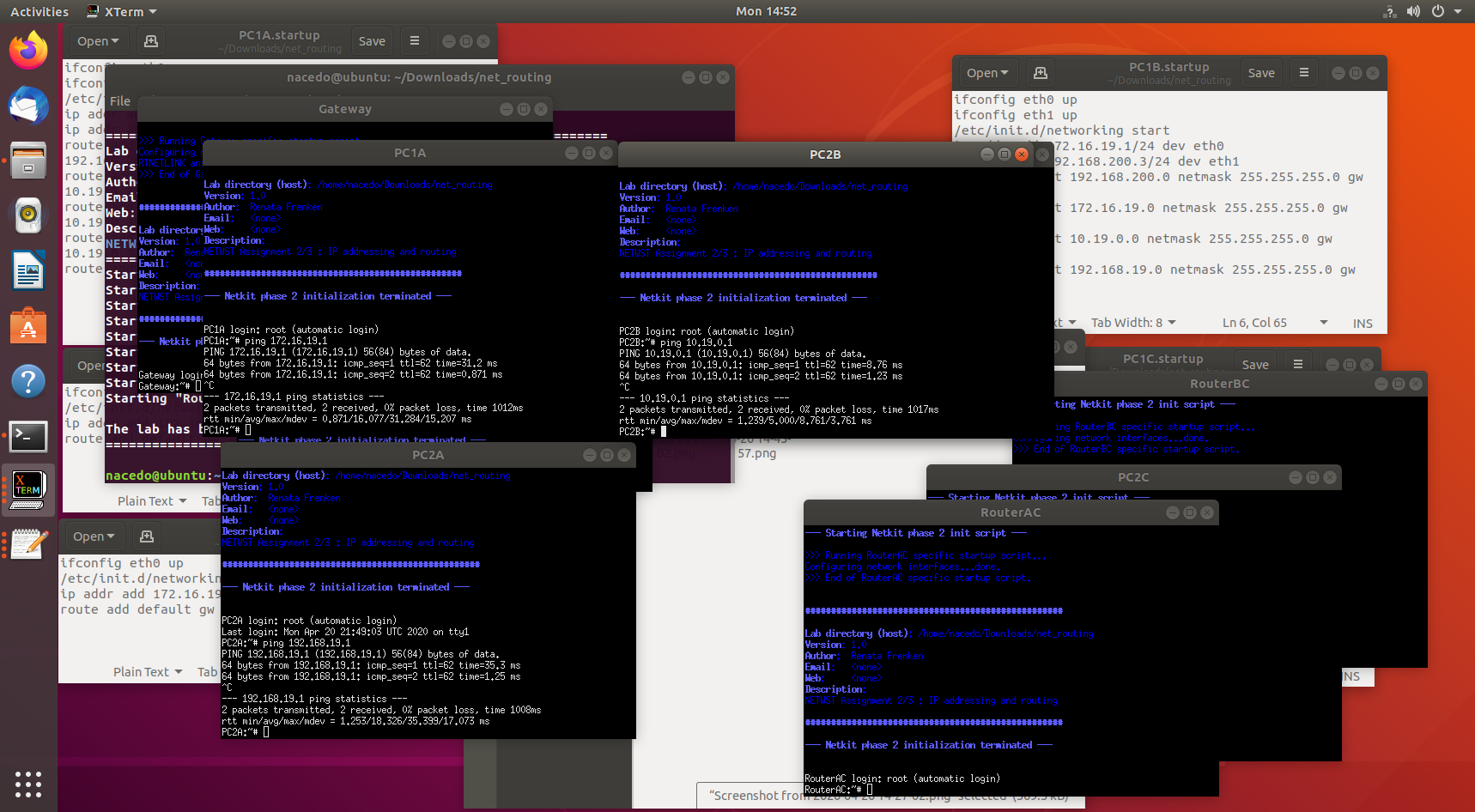
Tip: Use the network drawing from the last week assignment (week 8 ) and first think about the way you’re going to route. Use **tcpdump** and **traceroute** commands to debug your routing.

Provide screenshots of the following pings:

1. PC1A to PC1B
2. PC2B to PC2A
3. PC2A to PC1C

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





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:

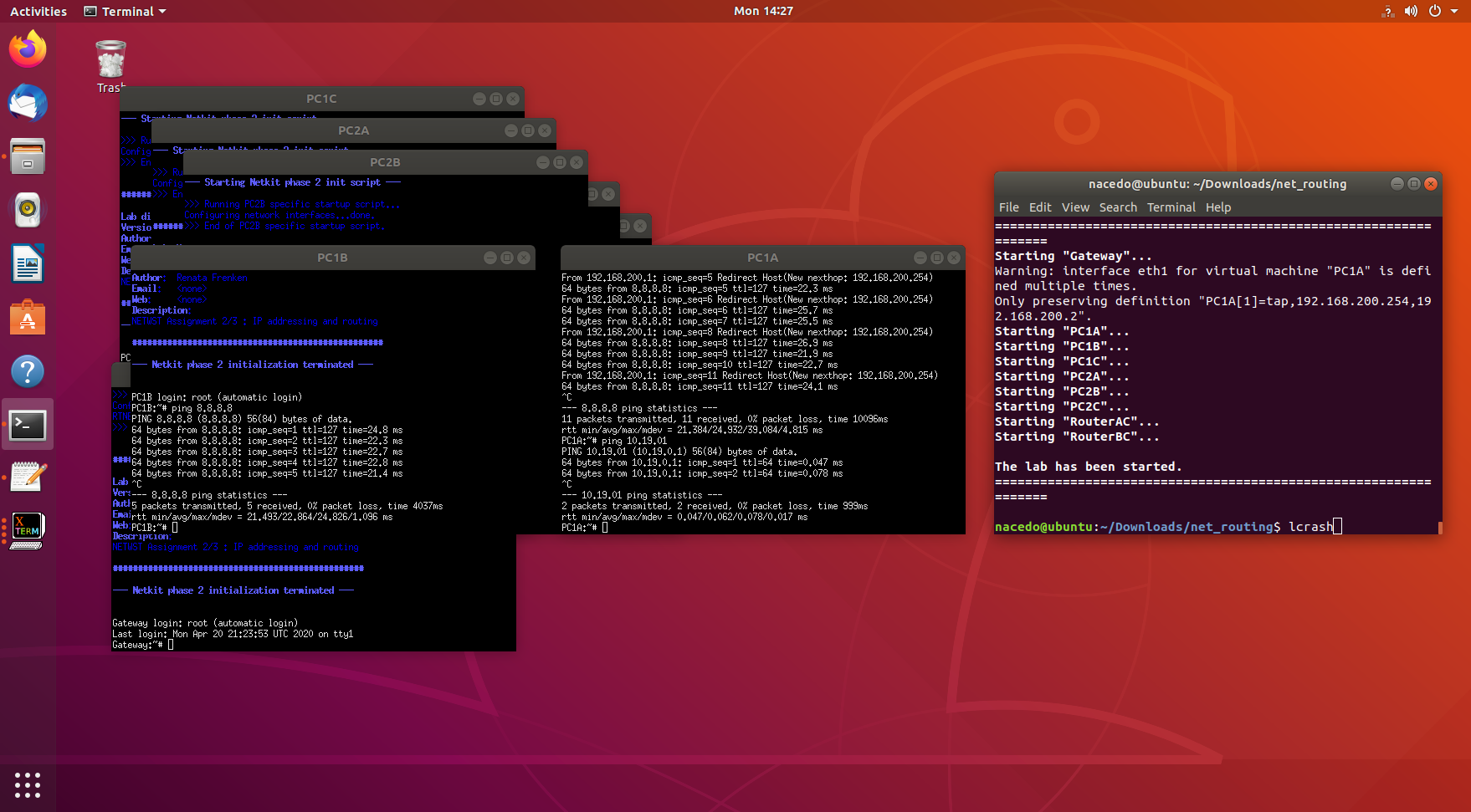


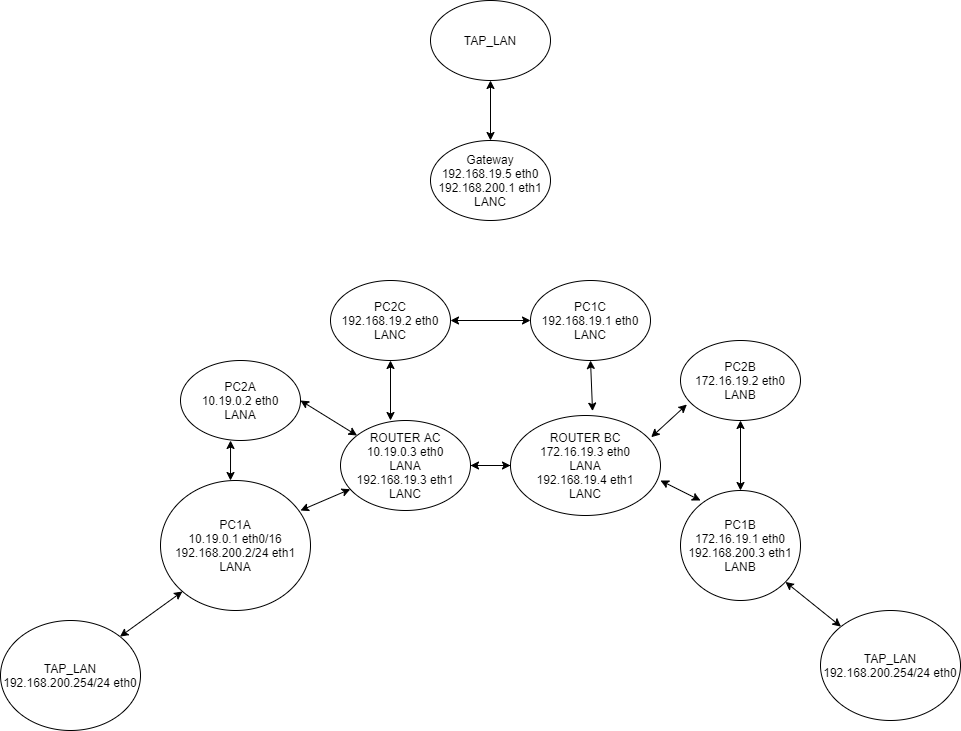
Configure your network in such a way that you can reach a node on Internet.

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





I know our way of connecting to the internet is not the most efficient. We could improve on that.

Table 1 : IPv4 address ranges per pair

|  |  |  |  |
| --- | --- | --- | --- |
| Pair | 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 |
| 3 | 10.3.0.0/16 | 172.16.3.0/24 | 192.168.3.0/24 |
| 4 | 10.4.0.0/16 | 172.16.4.0/24 | 192.168.4.0/24 |
| 5 | 10.5.0.0/16 | 172.16.5.0/24 | 192.168.5.0/24 |
| 6 | 10.6.0.0/16 | 172.16.6.0/24 | 192.168.6.0/24 |
| 7 | 10.7.0.0/16 | 172.16.7.0/24 | 192.168.7.0/24 |
| 8 | 10.8.0.0/16 | 172.16.8.0/24 | 192.168.8.0/24 |
| 9 | 10.9.0.0/16 | 172.16.9.0/24 | 192.168.9.0/24 |
| 10 | 10.10.0.0/16 | 172.16.10.0/24 | 192.168.10.0/24 |
| 11 | 10.11.0.0/16 | 172.16.11.0/24 | 192.168.11.0/24 |
| 12 | 10.12.0.0/16 | 172.16.12.0/24 | 192.168.12.0/24 |
| 13 | 10.13.0.0/16 | 172.16.13.0/24 | 192.168.13.0/24 |
| 14 | 10.14.0.0/16 | 172.16.14.0/24 | 192.168.14.0/24 |
| 15 | 10.15.0.0/16 | 172.16.15.0/24 | 192.168.15.0/24 |

# 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>**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.

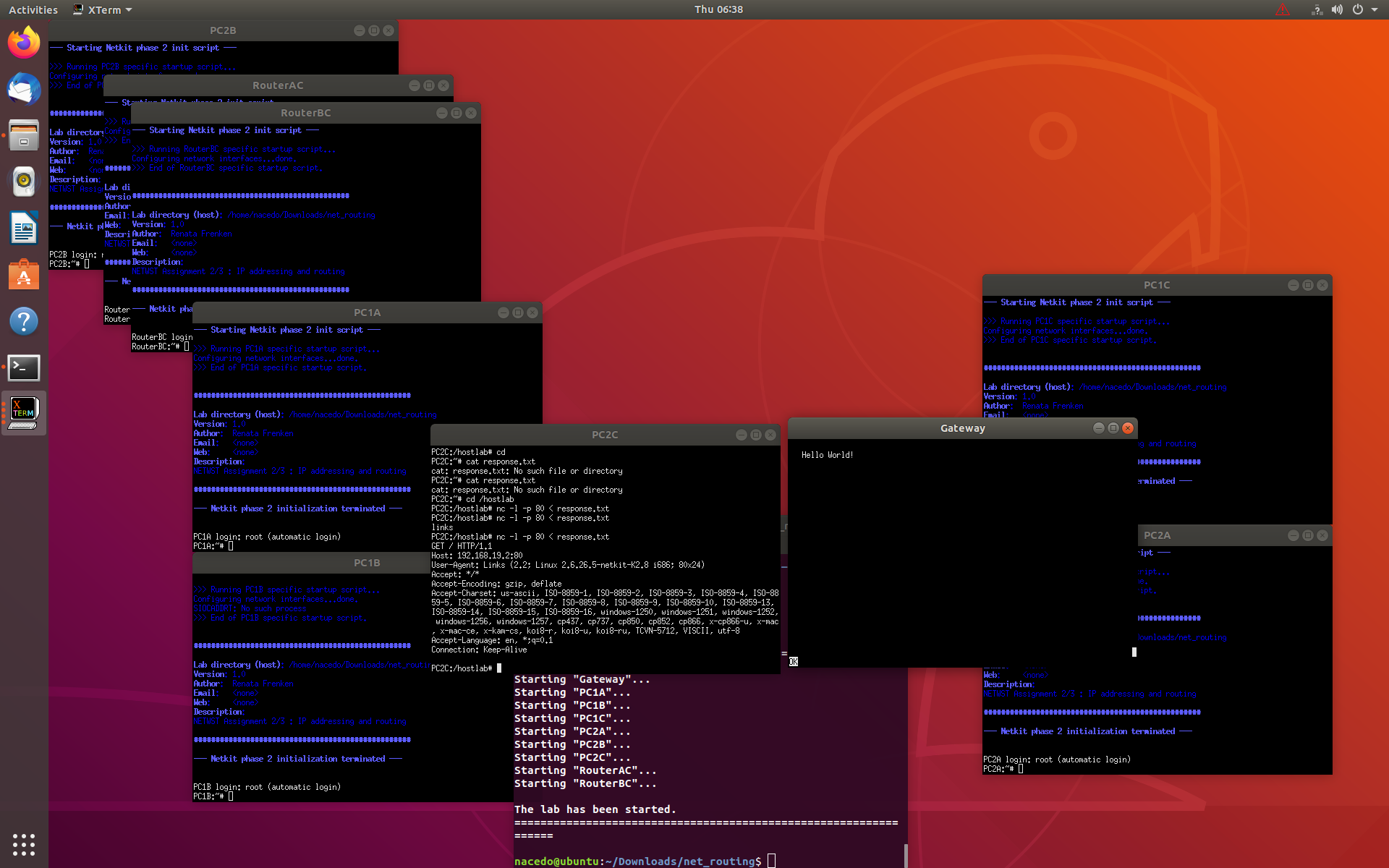
Note1: You can start links browser by issuing this command:

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.

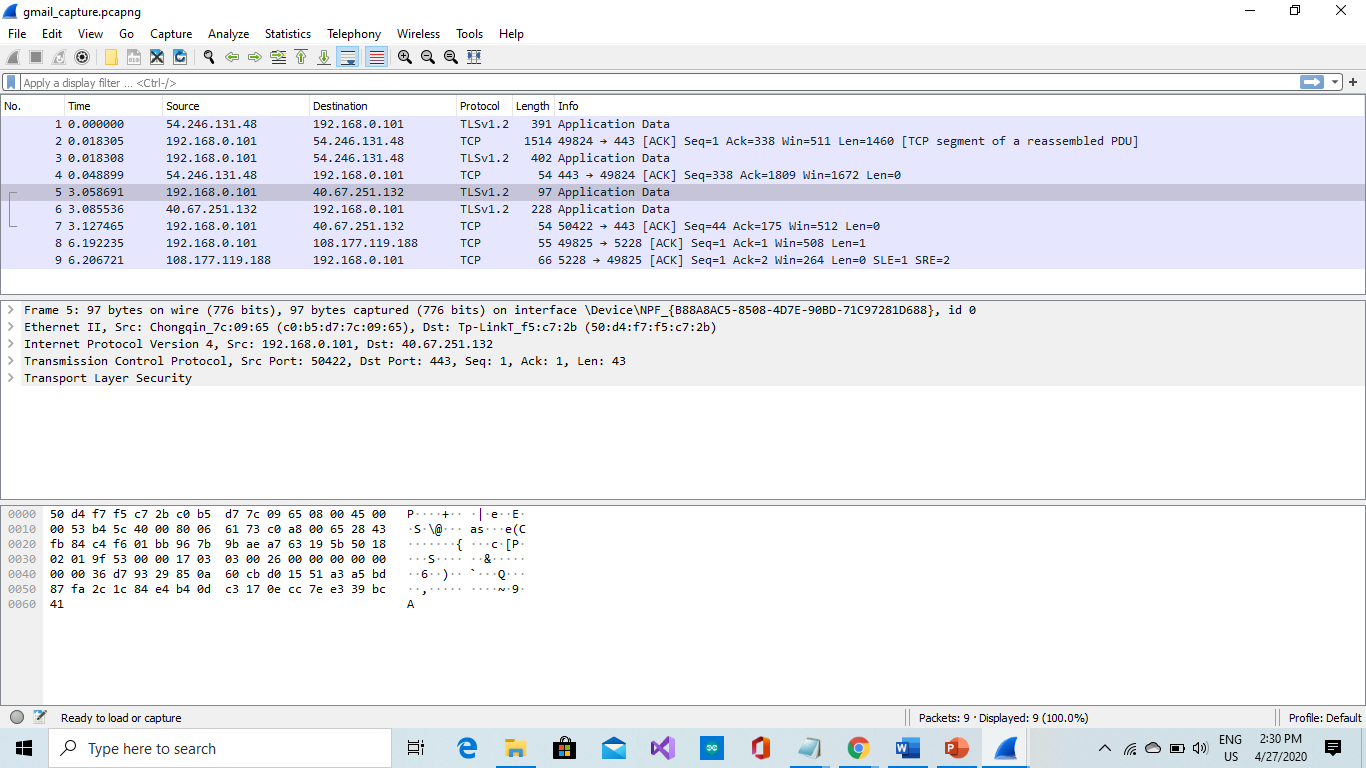


Task 2: Find 2 TCP uses

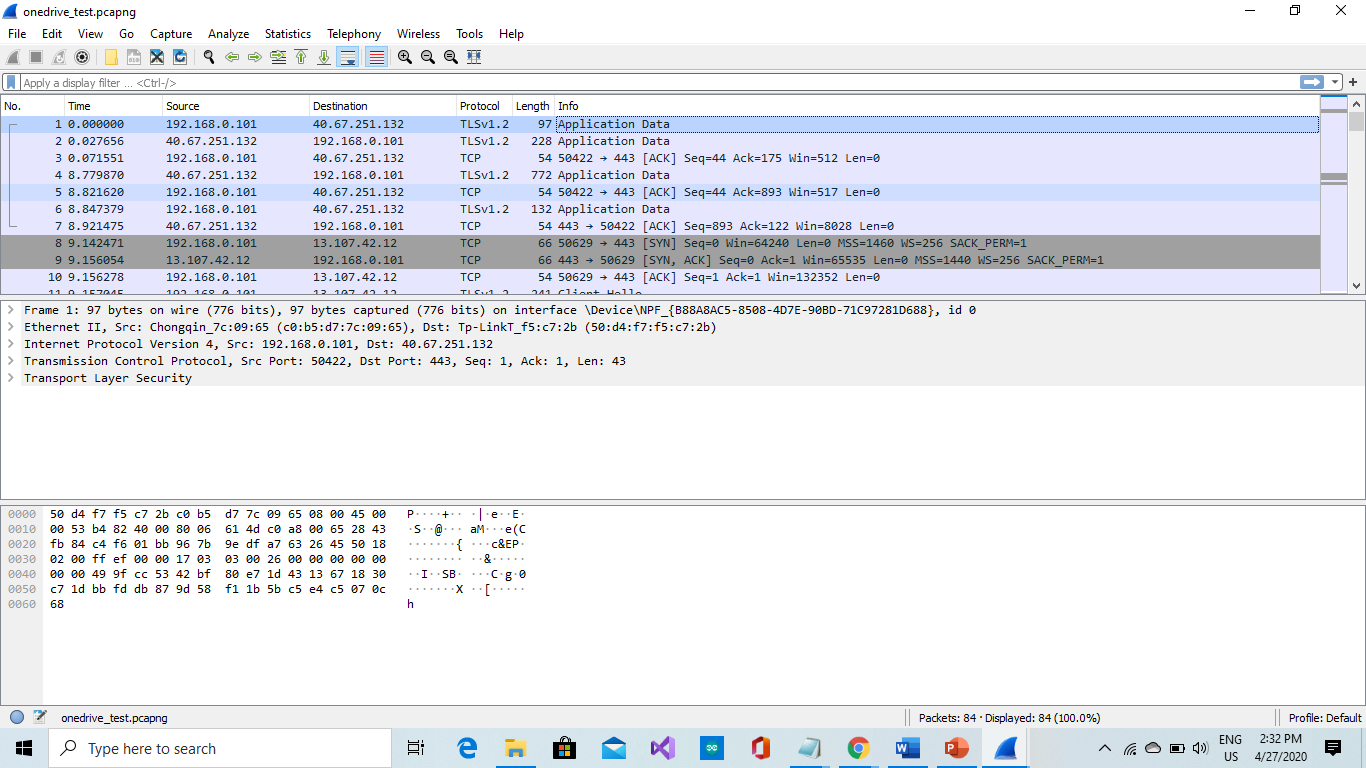
Think about two different scenarios for TCP use that you can simulate (you can do this on your own PC, so you don’t need Ubuntu for this). Start a Wireshark trace for both scenarios.

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

1. Sending a mail through Gmail app. The Gmail uses an SMTP server(Simple Mail Transfer Protocol) which is run by TCP.

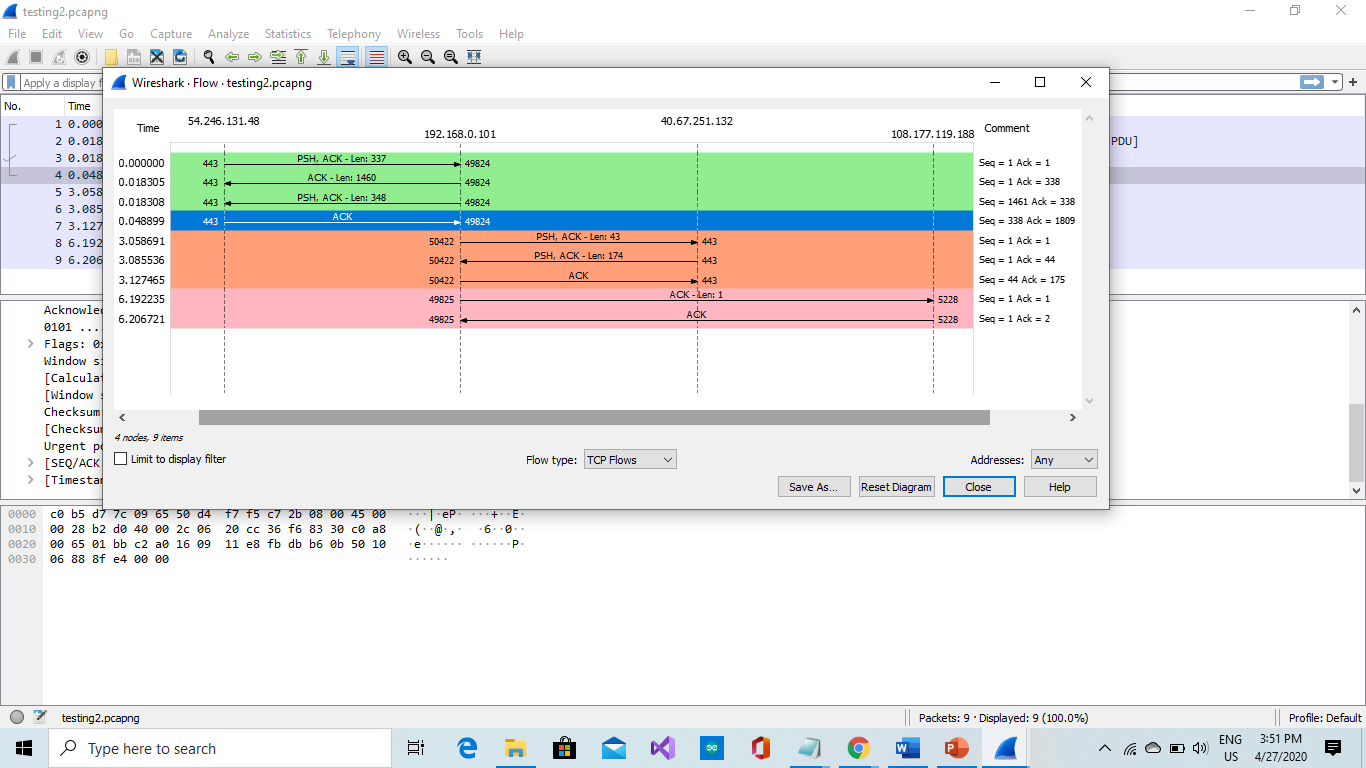


1. Sending a file from your phone to your laptop using the OneDrive app.



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 duringvarious stages (begin, middle, end) of the communication. Explain SYN, SYNACK and ACK. Explain the Len, Seq and Ack numbers.

At the beginning(the green part),the port 443 sends an acknowledgement to the port 49824 indicating a receipt of some data and also transmitting some more data,with a TCP payload of 337 bytes.Then the port 49824 sends an acknowledgement to the port 443 indicating that it has received the data which was sent ,with a TCP payload of 1460 bytes. This same port sends acknowledgement again to the port 443 indicating that it ha s received the data and is also transmitting more of the data, with a TCP payload of 348 bytes.

At the middle, a new port 50422 comes in sending an acknowledgement to the port 443 indicating a receipt of some data and also transmitting some more data,with a TCP payload of 43 bytes. The port 443 does the same back to the port 50422 but with 174 bytes. Then the port 50442 sends acknowledgement again to the port 443 indicating receipt of full data

At the end, two ports interact. These ports are 49825 and 5228.The port 49825 send an acknowledgment of data to the port 5228, with a length(TCP payload) of 1 byte. The port 5228 then sends an acknowledgement message that it has received full data from the port 49825 .

SYN – This signifies the start of a TCP session.

SYN,ACK – This is a combination of a SYN message together with an acknowledgment of a packet.

ACK – This is basically a message which a machine sends to another machine indicating that it has received the data which was sent

Len – This is the data portion of a packet. It can also be called TCP payload. It is measured in bytes.

Seq- This is a sequence number which allows you to keep track of the conversation between both systems.

ACK number – This is a number which shows/indicates that data has been acknowledged from one system to another.

Task 3: Find 2 UDP uses

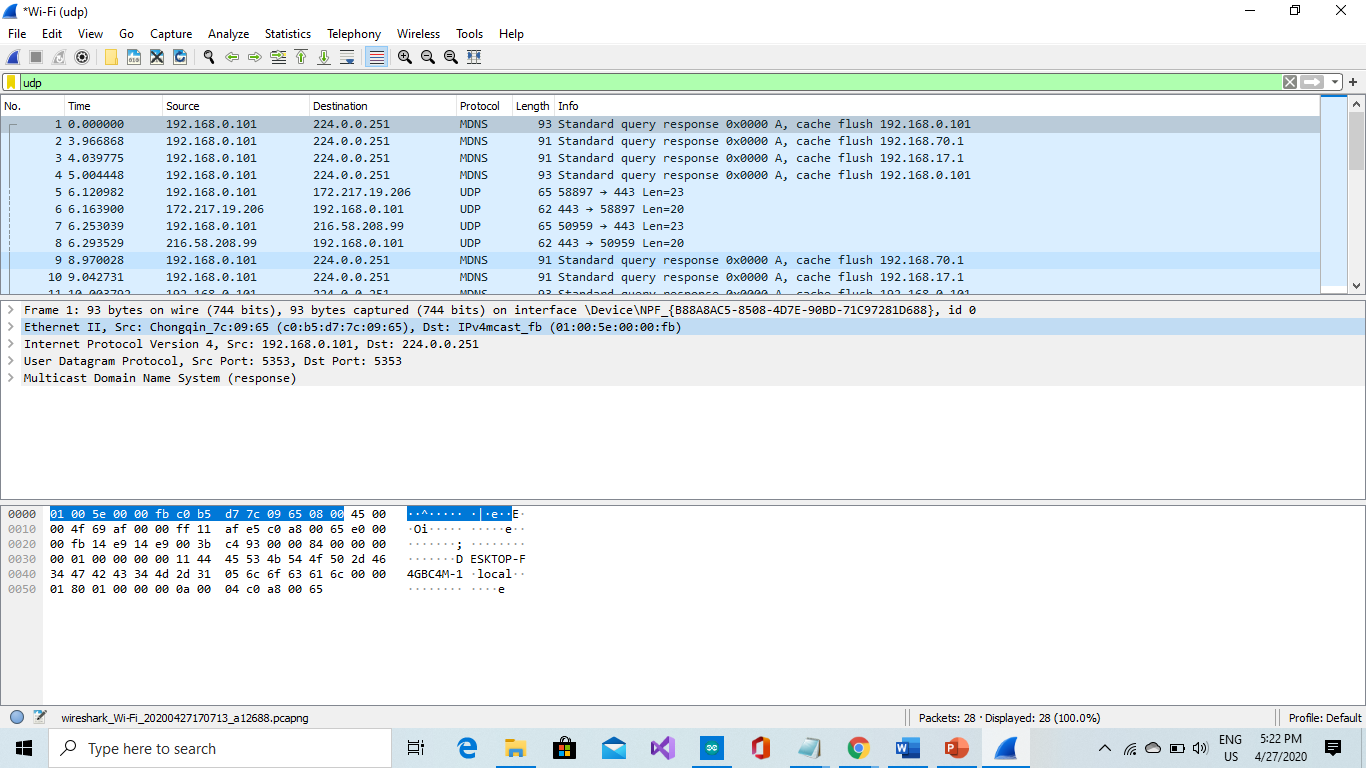
Think about two different scenarios for UDP use that you can simulate (you can do this on your own PC, so you don’t need Ubuntu for this). Start a Wireshark trace for both scenarios.

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

1. One of the scenarios for UDP use is watching a live stream video on Twitch.



1. Playing an online multiplayer game with your friends e.g(Fifa,COD)



Task 4 (Optional): TCP SYN Flooding

Read an explanation of TCP SYN Flooding at <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.