



TECNOLÓGICO
NACIONAL DE MEXICO



INSTITUTO TECNOLÓGICO DE CANCÚN

ALUMNO: GONGORA JIMENEZ FRANCISCO DAVID.

PROFESOR: ISMAEL JIMÉNEZ SÁNCHEZ.

MATERIA: FUNDAMENTOS DE TELECOMUNICACIONES.

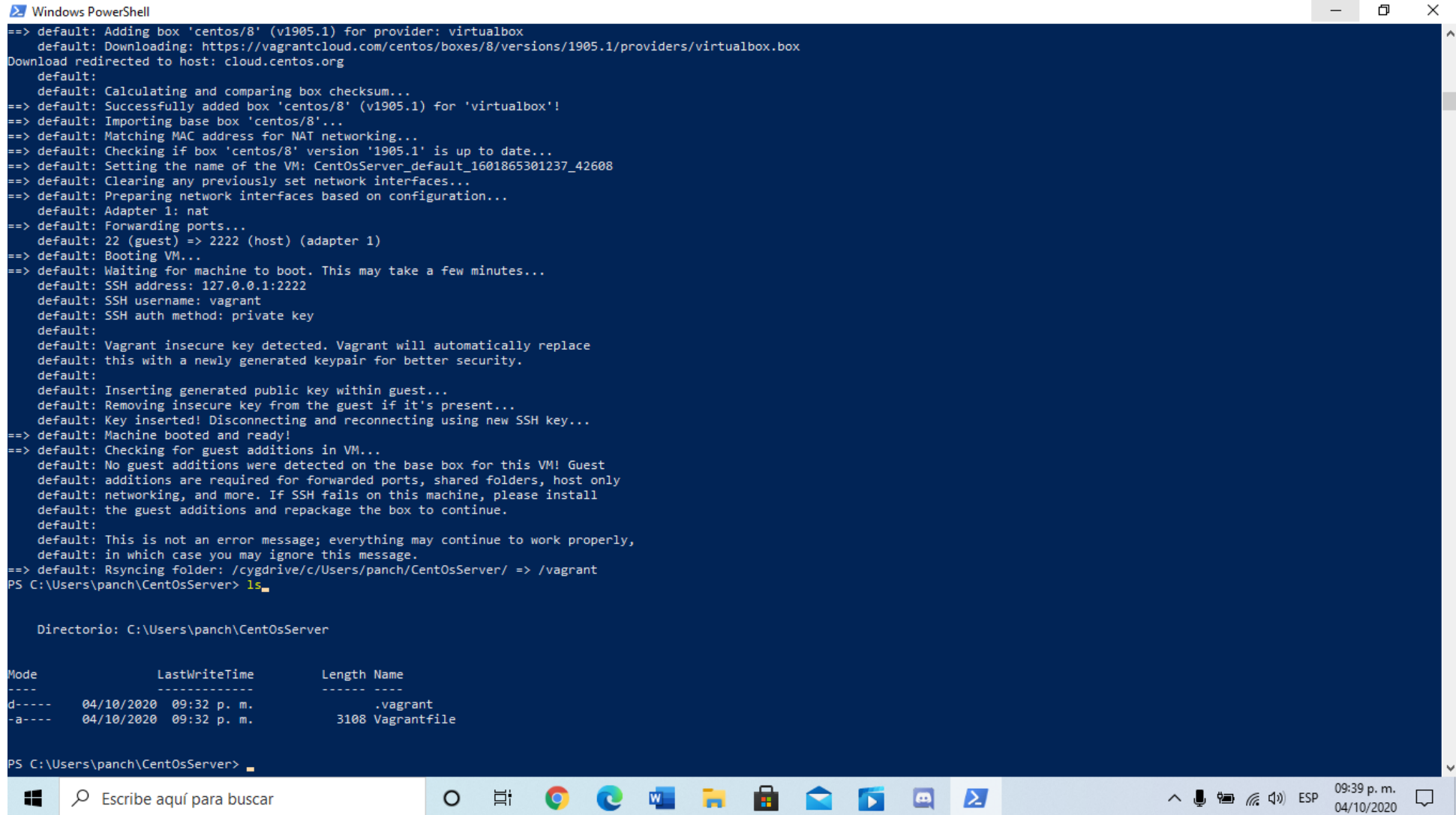
“PROYECTO: SISTEMAS DE COMUNICACIÓN”

HORARIO: 5PM-6PM.

FECHA DE ENTREGA:

26 DE OCTUBRE DEL 2020

FASE 1: Creando la carpeta donde se va a guardar el archivo donde se instalará el sistema operativo centos8



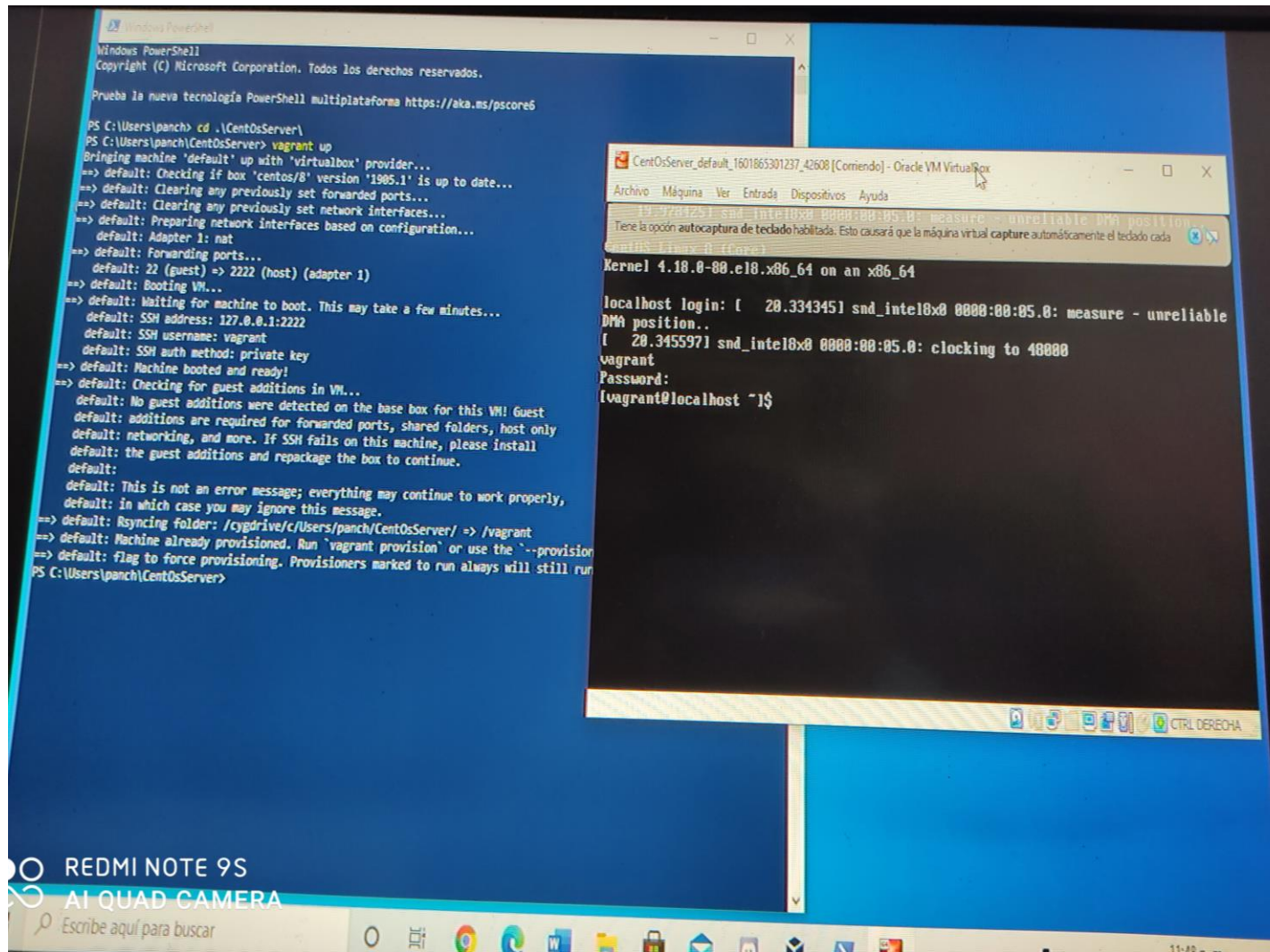
```
Windows PowerShell
==> default: Adding box 'centos/8' (v1905.1) for provider: virtualbox
default: Downloading: https://vagrantcloud.com/centos/boxes/8/versions/1905.1/providers/virtualbox.box
Download redirected to host: cloud.centos.org
default:
default: Calculating and comparing box checksum...
==> default: Successfully added box 'centos/8' (v1905.1) for 'virtualbox!'
==> default: Importing base box 'centos/8'...
==> default: Matching MAC address for NAT networking...
==> default: Checking if box 'centos/8' version '1905.1' is up to date...
==> default: Setting the name of the VM: CentOsServer_default_1601865301237_42608
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
default: Adapter 1: nat
==> default: Forwarding ports...
default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
default: SSH address: 127.0.0.1:2222
default: SSH username: vagrant
default: SSH auth method: private key
default:
default: Vagrant insecure key detected. Vagrant will automatically replace
default: this with a newly generated keypair for better security.
default:
default: Inserting generated public key within guest...
default: Removing insecure key from the guest if it's present...
default: Key inserted! Disconnecting and reconnecting using new SSH key...
==> default: Machine booted and ready!
==> default: Checking for guest additions in VM...
default: No guest additions were detected on the base box for this VM! Guest
default: additions are required for forwarded ports, shared folders, host only
default: networking, and more. If SSH fails on this machine, please install
default: the guest additions and repackage the box to continue.
default:
default: This is not an error message; everything may continue to work properly,
default: in which case you may ignore this message.
==> default: Rsyncing folder: /cygdrive/c/Users/panch/CentOsServer/ => /vagrant
PS C:\Users\panch\CentOsServer> ls

Directorio: C:\Users\panch\CentOsServer

Mode                LastWriteTime         Length Name
----                -
d-----          04/10/2020   09:32 p. m.         .vagrant
-a----          04/10/2020   09:32 p. m.         3108 Vagrantfile

PS C:\Users\panch\CentOsServer>
```

FASE 1: Finalizando la instalación del Centos8



FASE 2: Conectando las dos máquinas virtuales de CentOS en GNS3

The screenshot displays the GNS3 management console interface. The main workspace shows a network topology with three nodes: 'CentOsServer_default_1601865301237_42608-1' on the left, 'Hub1' in the center, and 'centos2server_default_1603568919671_76912-1' on the right. They are connected in a linear fashion. The right sidebar contains two panels: 'Topology Summary' and 'Servers Summary'. The 'Topology Summary' panel lists the nodes and their console connections. The 'Servers Summary' panel shows the CPU and RAM usage for the desktop environment. The bottom console window displays the GNS3 management console output, including a warning message about a virtual machine state.

Topology Summary

Node	Console
centos2server_default_1603568919671_76912-1	telnet localhost:5001
CentOsServer_default_1601865301237_42608-1	telnet localhost:5000
Hub1	none

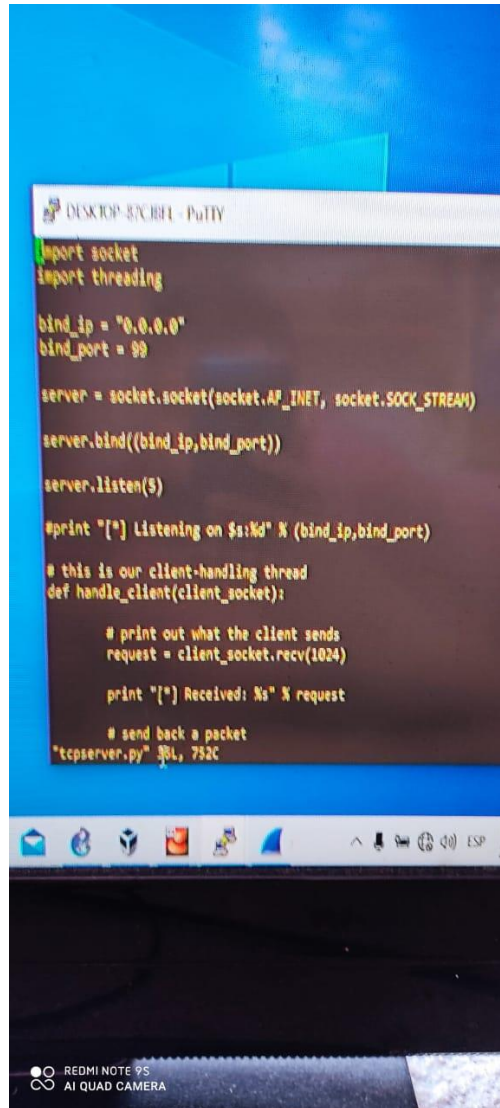
Servers Summary

- DESKTOP-87CIBFL CPU 39.2%, RAM 50.2%

Console

```
GNS3 management console.  
Running GNS3 version 2.2.14 on Windows (64-bit) with Python 3.6.8 Qt 5.12.1 and PyQt 5.12.  
Copyright (c) 2006-2020 GNS3 Technologies.  
Use Help -> GNS3 Doctor to detect common issues.  
  
=> VirtualBox VM 'CentOsServer_default_1601865301237_42608-1' is not powered off (current state is 'aborted')
```

Configurando los scripts de Python de tcpServer.py y tcpClient.py con direcciones ip correctos y los puertos



```
DESKTOP-87CIBFL - PuTTY
import socket
import threading

bind_ip = "0.0.0.0"
bind_port = 99

server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind((bind_ip, bind_port))
server.listen(5)

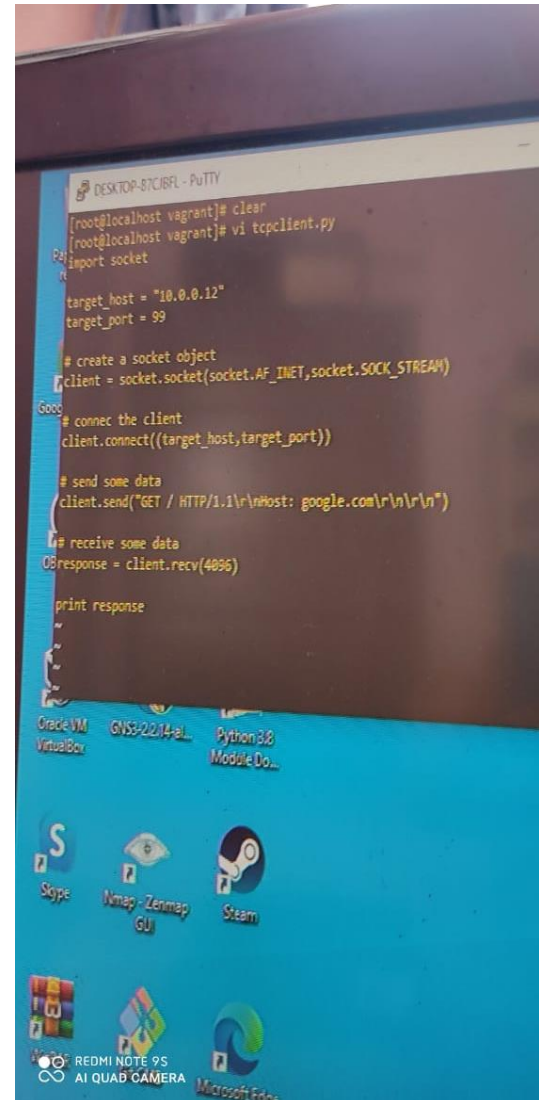
#print "[" Listening on %s:%d" % (bind_ip, bind_port)

# this is our client-handling thread
def handle_client(client_socket):

    # print out what the client sends
    request = client_socket.recv(1024)

    print "[" Received: %s" % request

    # send back a packet
    "tcpserver.py" % bind_ip, 752C
```



```
DESKTOP-87CIBFL - PuTTY
[root@localhost vagrant]# clear
[root@localhost vagrant]# vi tcpclient.py
import socket

target_host = "10.0.0.12"
target_port = 99

# create a socket object
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# connect the client
client.connect((target_host, target_port))

# send some data
client.send("GET / HTTP/1.1\r\nHost: google.com\r\n\r\n")

# receive some data
response = client.recv(4096)

print response
```

En esta parte se esta ejecutando el servidor mediante los scripts de tcp server y tcp client

The screenshot displays a Windows 10 desktop environment with four terminal windows open. The top two windows are Oracle VM VirtualBox instances of CentOS 8. The bottom-left window is a PuTTY terminal running a script that sends an HTTP GET request to google.com and receives a response. The bottom-right window is another PuTTY terminal showing the output of a netstat command, which lists listening ports and services. A console window in the background shows the output of an nmap scan for 10.0.0.5.

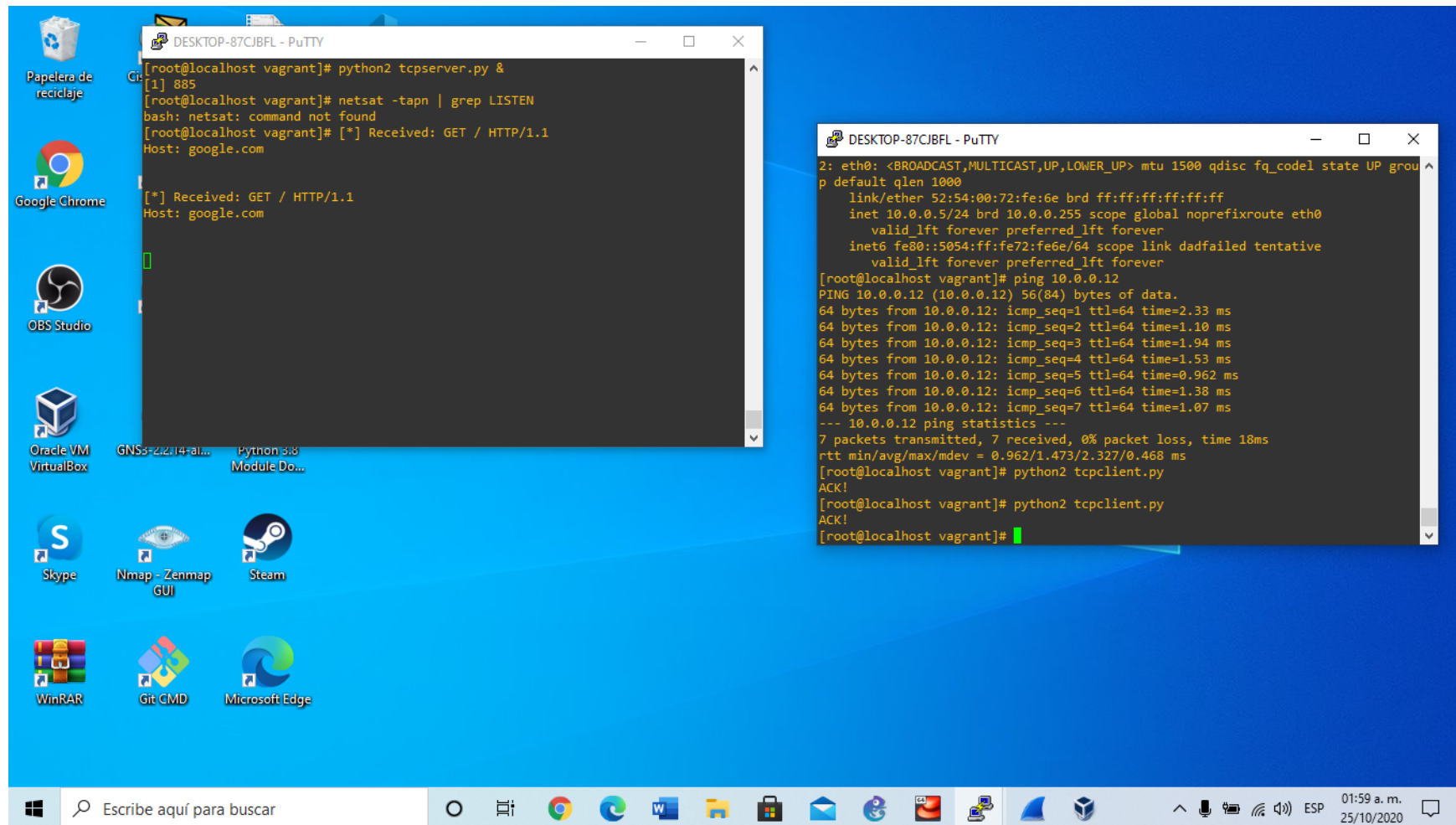
```
Probing EDD (edd=off to disable)... ok
CentOS Linux 8 (Core)
Kernel 4.18.0-80.el8.x86_64 on an x86_64

localhost login: [ 17.921137] snd_intel8x0 0000:00:05.0: measure - unreliable DMA position..
[ 18.292285] snd_intel8x0 0000:00:05.0: measure - unreliable DMA position..
[ 18.317210] snd_intel8x0 0000:00:05.0: clocking to 48000
vagrant
Password:
Last login: Sun Oct 25 00:50:50 on tty1
[vagrant@localhost ~]$ supersu
-bash: supersu: command not found
[vagrant@localhost ~]$ supersu
-bash: supersu: command not found
[vagrant@localhost ~]$ super su
-bash: super: command not found
[vagrant@localhost ~]$ super su
-bash: super: command not found
[vagrant@localhost ~]$
```

```
[1] 916
[root@localhost vagrant]# netstat -tavn | grep LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
594/sshd
tcp        0      0 0.0.0.0:650            0.0.0.0:*               LISTEN
916/python2
tcp        0      0 0.0.0.0:111            0.0.0.0:*               LISTEN
1/systemd
tcp6       0      0 :::22                  :::*                     LISTEN
594/sshd
tcp6       0      0 :::111                  :::*                     LISTEN
1/systemd
[root@localhost vagrant]# netstat -tavn | grep LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      594/sshd
tcp        0      0 0.0.0.0:650            0.0.0.0:*               LISTEN      916/python2
tcp        0      0 0.0.0.0:111            0.0.0.0:*               LISTEN      1/systemd
tcp6       0      0 :::22                  :::*                     LISTEN      594/sshd
tcp6       0      0 :::111                  :::*                     LISTEN      1/systemd
[root@localhost vagrant]#
```

```
[root@localhost vagrant]# nmap -f 10.0.0.5
Starting Nmap 7.70 ( https://nmap.org ) at 2020-10-25 02:27 UTC
Running Nmap scan report for 10.0.0.5
Host is up (0.000073s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
111/tcp   open  rpcbind
Nmap done: 1 IP address (1 host up) scanned in 14.84 seconds
[root@localhost vagrant]#
```

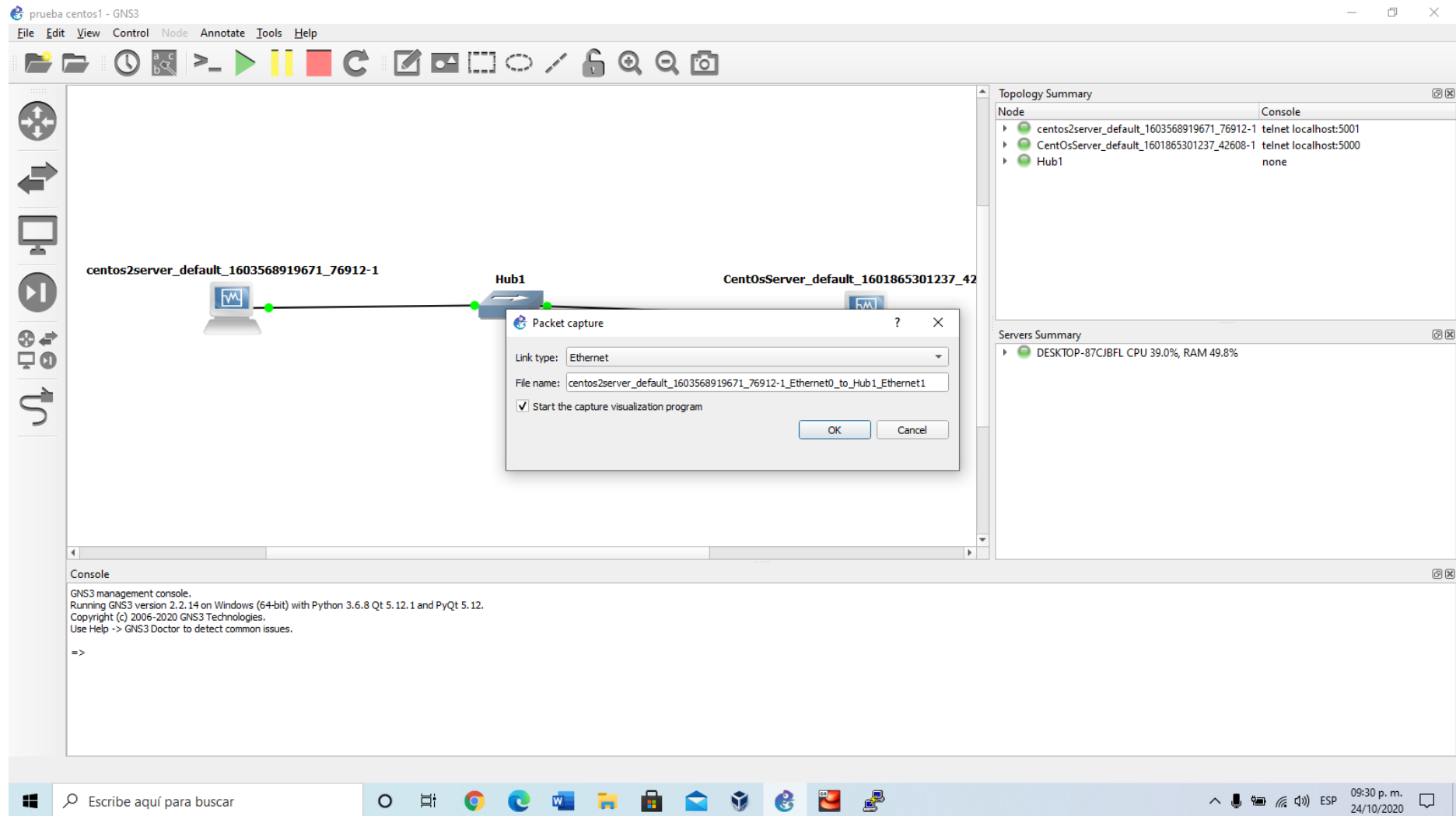

FASE 3: aquí se hizo un ping para comprobar que este funcionando bien las ip configuradas a las máquinas virtuales y también se ejecuto el comando python2 tcpserver.py & y el comando python2 tcpclient.py para que se pueda comenzar a mostrar el trafico de la comunicación de las maquinas virtuales.



```
[root@localhost vagrant]# python2 tcpserver.py &
[1] 885
[root@localhost vagrant]# netsat -tapn | grep LISTEN
bash: netsat: command not found
[root@localhost vagrant]# [*] Received: GET / HTTP/1.1
Host: google.com
[*] Received: GET / HTTP/1.1
Host: google.com

2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP grou
p default qlen 1000
    link/ether 52:54:00:72:fe:6e brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.5/24 brd 10.0.0.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::5054:ff:fe72:fe6e/64 scope link dadfailed tentative
        valid_lft forever preferred_lft forever
[root@localhost vagrant]# ping 10.0.0.12
PING 10.0.0.12 (10.0.0.12) 56(84) bytes of data.
64 bytes from 10.0.0.12: icmp_seq=1 ttl=64 time=2.33 ms
64 bytes from 10.0.0.12: icmp_seq=2 ttl=64 time=1.10 ms
64 bytes from 10.0.0.12: icmp_seq=3 ttl=64 time=1.94 ms
64 bytes from 10.0.0.12: icmp_seq=4 ttl=64 time=1.53 ms
64 bytes from 10.0.0.12: icmp_seq=5 ttl=64 time=0.962 ms
64 bytes from 10.0.0.12: icmp_seq=6 ttl=64 time=1.38 ms
64 bytes from 10.0.0.12: icmp_seq=7 ttl=64 time=1.07 ms
--- 10.0.0.12 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 18ms
rtt min/avg/max/mdev = 0.962/1.473/2.327/0.468 ms
[root@localhost vagrant]# python2 tcpclient.py
ACK!
[root@localhost vagrant]# python2 tcpclient.py
ACK!
[root@localhost vagrant]#
```

En esta parte se esta iniciando la captura del trafico de comunicación de ambas máquinas virtuales con el gns3 y se mostrara en wireshark



FASE 4: Aquí ya se está mostrando como se esta haciendo la captura del trafico de la comunicación en ambas maquinas virtuales con los scripts mediante wireshark

The screenshot displays a Windows desktop environment with a blue background. On the left side, there are several desktop icons: 'Papelera de reciclaje', 'Google Chrome', 'OBS Studio', 'Oracle VM VirtualBox', 'Skype', 'Nmap - Zenmap GUI', 'WinRAR', and 'Git CMD'. The taskbar at the bottom shows the Start button, a search bar with the text 'Escribe aquí para buscar', and several application icons including Chrome, Edge, Word, File Explorer, Task Manager, and others. The system clock in the bottom right corner indicates '01:57 a. m. 25/10/2020'.

In the center of the screen, a Wireshark window is open, titled 'Capturing from - [CentOSServer_default_1601865301237_42608-1 Ethernet0 to Hub1 Ethernet0]'. The interface shows a list of captured packets. The selected packet is number 4, which is an HTTP GET request. The packet details pane shows the following information:

- Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface -, id 0
- Ethernet II, Src: RealtekU_72:fe:6e (52:54:00:72:fe:6e), Dst: RealtekU_72:fe:6e (52:54:00:72:fe:6e)
- Internet Protocol Version 4, Src: 10.0.0.5, Dst: 10.0.0.12
- Transmission Control Protocol, Src Port: 50960, Dst Port: 99, Seq: 0, Len: 0

The packet bytes pane shows the raw data in hexadecimal and ASCII format.

In the background, a terminal window titled 'DESKTOP-87CJBFL - PuTTY' is open. It shows the execution of a Python script named 'tcpserver.py' and the output of the 'netsat' command, which is listening on port 99.

FASE 4: Aquí se muestra mas datos sobre la captura de la comunicación de ambas maquinas virtuales con los scripts mediante Wireshark

The screenshot displays the Wireshark interface with a packet capture from a CentOsServer. The packet list shows various network protocols including ARP, ICMPv6, and TCP/HTTP. The packet details pane shows the raw data for frame 39, which is a TCP segment. The background shows a Windows 10 desktop with a blue wallpaper and a taskbar at the bottom.

No.	Time	Source	Destination	Protocol	Length	Info
28	1234.589125	RealtekU_72:fe:6e	RealtekU_72:fe:6e	ARP	60	10.0.0.12 is at 52:54:00:72:fe:6e
29	1234.827270	RealtekU_72:fe:6e	RealtekU_72:fe:6e	ARP	60	Who has 10.0.0.5? Tell 10.0.0.12
30	1234.828246	RealtekU_72:fe:6e	RealtekU_72:fe:6e	ARP	60	10.0.0.5 is at 52:54:00:72:fe:6e
31	1299.851470	fe80::5054:ff:fe72:...	ff02::2	ICMPv6	70	Router Solicitation from 52:54:00:72:fe:6e
32	1420.843184	10.0.0.5	10.0.0.12	TCP	74	50964 → 99 [SYN] Seq=0 Win=29200 Len=0 ...
33	1420.844187	10.0.0.12	10.0.0.5	TCP	74	99 → 50964 [SYN, ACK] Seq=0 Ack=1 Win=2...
34	1420.845165	10.0.0.5	10.0.0.12	TCP	66	50964 → 99 [ACK] Seq=1 Ack=1 Win=29248 ...
35	1420.846113	10.0.0.5	10.0.0.12	HTTP	102	GET / HTTP/1.1
36	1420.846113	10.0.0.12	10.0.0.5	TCP	66	99 → 50964 [ACK] Seq=1 Ack=37 Win=28992...
37	1420.848092	10.0.0.12	10.0.0.5	TCP	70	99 → 50964 [PSH, ACK] Seq=1 Ack=37 Win=...
38	1420.848092	10.0.0.5	10.0.0.12	TCP	66	50964 → 99 [ACK] Seq=37 Ack=5 Win=29248...
39	1420.849040	10.0.0.12	10.0.0.5	TCP	66	99 → 50964 [FIN, ACK] Seq=5 Ack=37 Win=...
40	1420.851969	10.0.0.5	10.0.0.12	TCP	66	50964 → 99 [FIN, ACK] Seq=37 Ack=6 Win=...
41	1420.851969	10.0.0.12	10.0.0.5	TCP	66	99 → 50964 [ACK] Seq=6 Ack=38 Win=28992...
42	1426.075519	RealtekU_72:fe:6e	RealtekU_72:fe:6e	ARP	60	Who has 10.0.0.12? Tell 10.0.0.5
43	1426.076492	RealtekU_72:fe:6e	RealtekU_72:fe:6e	ARP	60	10.0.0.12 is at 52:54:00:72:fe:6e
44	1426.314638	RealtekU_72:fe:6e	RealtekU_72:fe:6e	ARP	60	Who has 10.0.0.5? Tell 10.0.0.12
45	1426.315612	RealtekU_72:fe:6e	RealtekU_72:fe:6e	ARP	60	10.0.0.5 is at 52:54:00:72:fe:6e

Frame 39: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface -, id 0

```
0000 52 54 00 72 fe 6e 52 54 00 72 fe 6e 08 00 45 00  RT.r.nRT.r.n.E
0010 00 34 17 ba 40 00 40 06 0e fa 0a 00 00 0c 0a 00  4.@.@ .....
0020 00 05 00 63 c7 14 a6 05 cf 1f 81 03 fd cf 80 11  ...c.....
0030 01 c5 83 21 00 00 01 01 08 0a 2a 80 95 c5 5d 4a  ...!.....*...J
0040 04 c5 ..
```

Ready to load or capture | Packets: 45 · Displayed: 45 (100.0%) | Profile: Default

FASE 5: REPORTE DE CONCLUSIONES

Después de realizar este proyecto, entendí que la imagen de arriba para entenderla es que se relaciona al tema triple handshake en el cual consiste de la forma que los dispositivos se organicen o se pongan de acuerdo para iniciar la conversación, en este caso es como si el cliente le dice al servidor, ¿disculpa necesito un favor tuyo y el servidor le responde necesitas un favor mío? Y el cliente le responde si necesito un favor tuyo., en este caso tomaremos como ejemplo la imagen que se encuentra arriba de esta hoja , el cual cliente que en este caso sería la dirección ip 10.0.0.5 quiere comunicarse con el servidor lo primero que se hace es enviarle dentro del paquete de red un tcp donde ese va a activar una bandera o un aviso al servidor (en este caso el servidor sería la dirección ip 10.0.0.12) donde el cliente quiere iniciar una conversación en donde a esa bandera se le conoce como SYN el cual se refiere a la sincronización, una vez que el servidor recibe el “SYN” del cliente, el servidor procede a responderle con un “SYN,ACK”, indicando que recibió la solicitud para comunicarse con el servidor y también le manda el “ACK” activado indicando que reconoce que el cliente quiere iniciar una conversación con el (servidor), entonces el cliente procede a responderle y le avisa al servidor esta bien y le devuelve un “ACK”, entonces ese último(tercer) mensaje le dicen esta bien, estamos ambos listos para iniciar la conversación, y el ultimo paquete son los datos que uno quería enviar, después de haber terminado la conversación el servidor manda un bandera “FIN,ACK” indicando que ya termino de recibir toda la información, y el ack indicando que le confirme que si es correcto, entonces esa es la forma que entendí de cómo se captura el tráfico de telecomunicaciones entre las dos máquinas virtuales y es bastante interesante para mi saber y aprender todo esto.