



TECNOLÓGICO  
NACIONAL DE MÉXICO



INSTITUTO TECNOLÓGICO DE CANCÚN

INGENIERÍA EN  
SISTEMAS COMPUTACIONALES  
FUNDAMENTOS DE TELECOMUNICACIONES

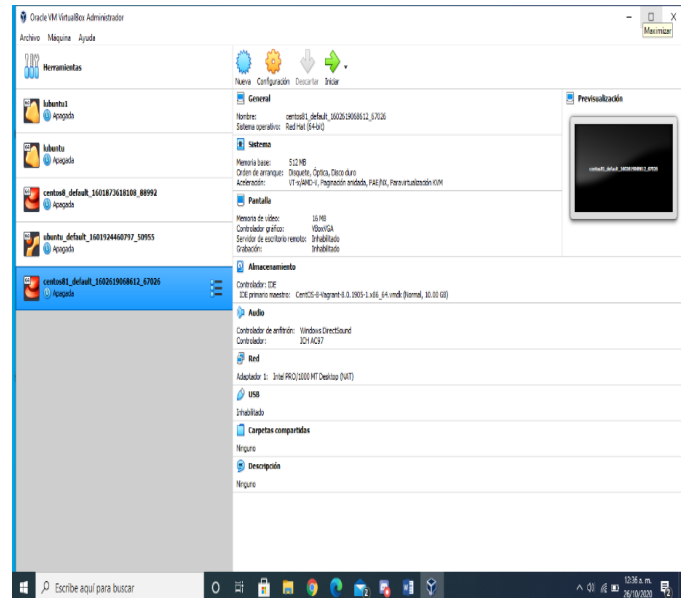
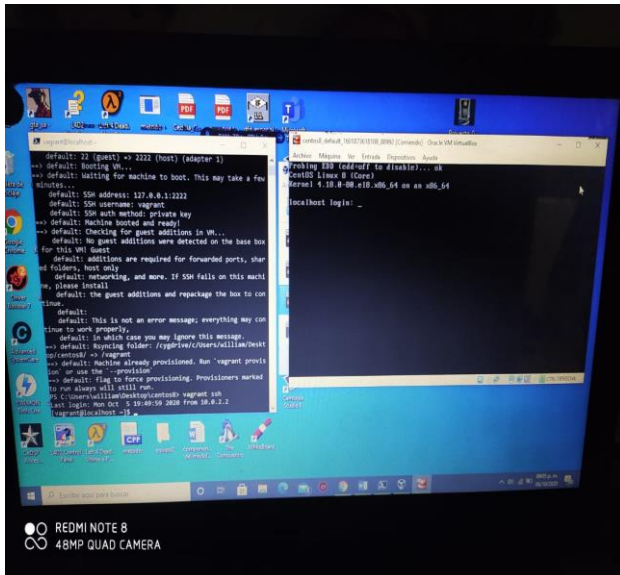
NOMBRE DEL ALUMNO:  
COLLÍ CHEL WILLIAM BLADIMIR

HORARIO  
LUNES A JUEVES  
5:00 PM – 6:00 PM

PROFESOR  
ISMAEL JIMENEZ SANCHEZ

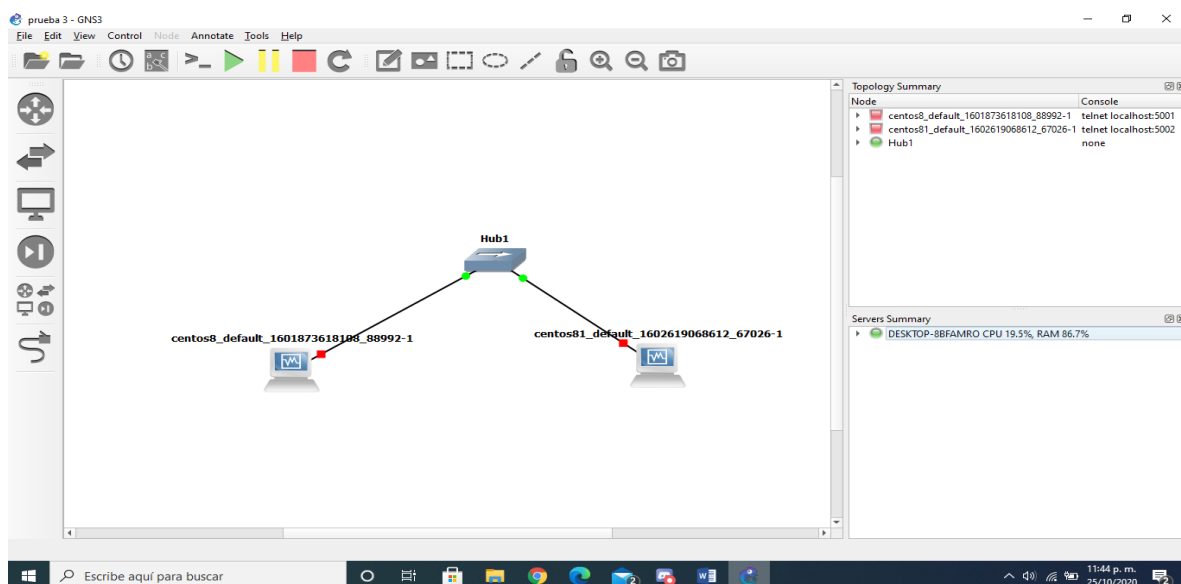
## Fase 1: instalar dos centos8 en virtual box usando vagrant.

Se tuvo que instalar dos centos8 por medio de vagrant y powershell como se ve en la imagen y darle vagrant up para poder iniciar las maquinas.



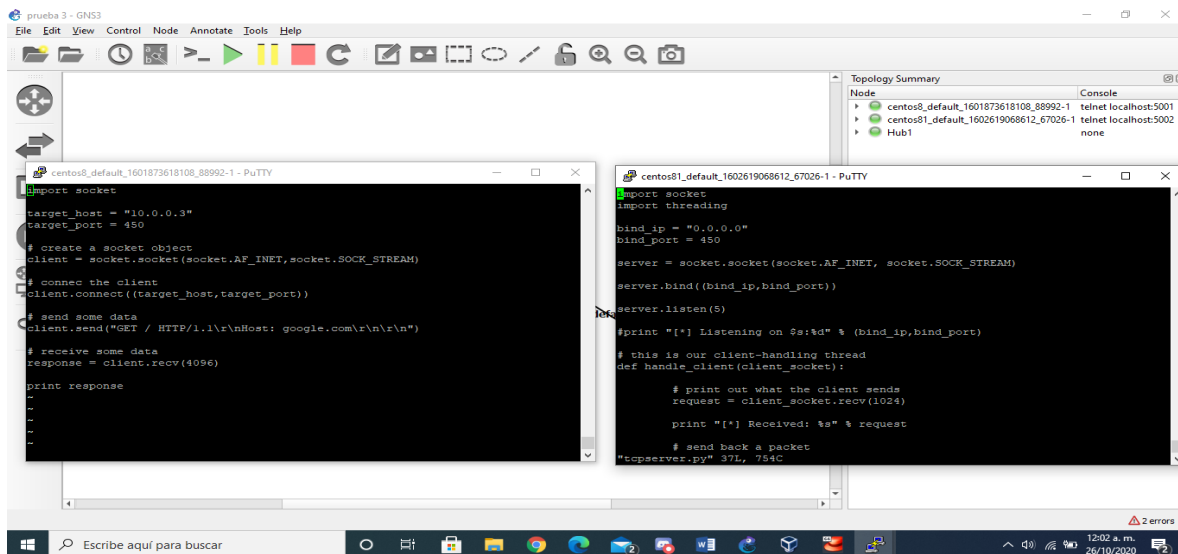
## Fase 2: conectar en gns3 los dos centos de virtual box con Ethernet hub.

Es esta fase se conectan los dos centos en gns3 como en la imagen y con un Ethernet hub.



### Fase 3: Usar los scripts de python para conectar las dos VMs usando sockets.

En esta fase se tuvieron que configurar las ip y por medio del comando nmtui una vez configurado las ip se tiene que poner los script de Python de cliente y servidor como se ve en la imagen y se guardan



The screenshot shows the GNS3 interface with two VMs. The left VM, 'centos8\_default\_1601873618108\_88992-1', has a PuTTY window with a Python client script. The right VM, 'centos81\_default\_1602619068612\_67026-1', has a PuTTY window with a Python server script. A 'Topology Summary' panel on the right shows the network connections between the VMs and a central 'Hub1'.

```
centos8_default_1601873618108_88992-1 - PuTTY
import socket
target_host = "10.0.0.3"
target_port = 450

# 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

centos81_default_1602619068612_67026-1 - PuTTY
import socket
import threading

bind_ip = "0.0.0.0"
bind_port = 450

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):

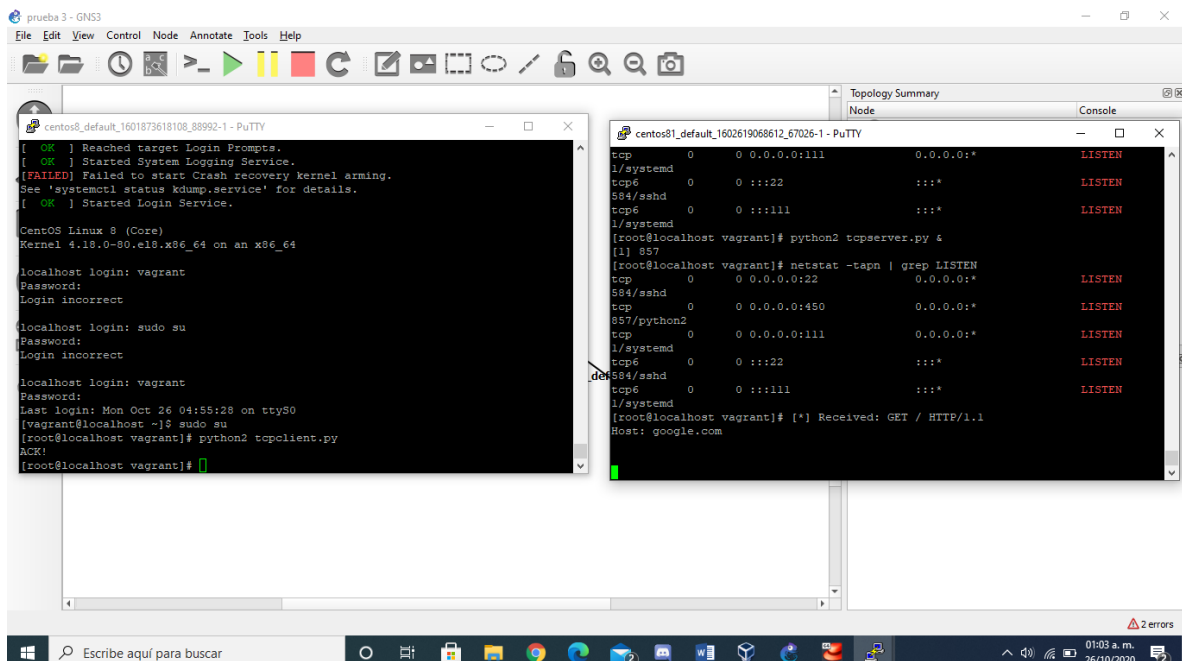
    # print out what the client sends
    request = client_socket.recv(1024)

    print "[*] Received: %s" % request

    # send back a packet
    "tcpserver.py" 37L, 754C
```

### Fase 4: Capturar el tráfico de la comunicación entre las dos VMs.

al momento de utilizar los script aquí levantamos el servidor y después el cliente para poder tener el tráfico de comunicación de los script como en la imagen.



The screenshot shows the GNS3 interface with the same two VMs. The left VM's PuTTY window shows the execution of the client script, which sends a GET request to 10.0.0.3. The right VM's PuTTY window shows the execution of the server script, which is listening on 0.0.0.0:450 and has received the GET request. A 'netstat -tbn | grep LISTEN' command is run on the right VM, showing that ports 22, 450, and 857 are listening. A 'tcpdump -i eth0' command is run on the right VM to capture traffic.

```
centos8_default_1601873618108_88992-1 - PuTTY
[ OK ] Reached target Login Prompts.
[ OK ] Started System Logging Service.
[FAILED] Failed to start Crash recovery kernel arming.
See 'systemctl status kdump.service' for details.
[ OK ] Started Login Service.

CentOS Linux 8 (Core)
Kernel 4.18.0-80.el8.x86_64 on an x86_64

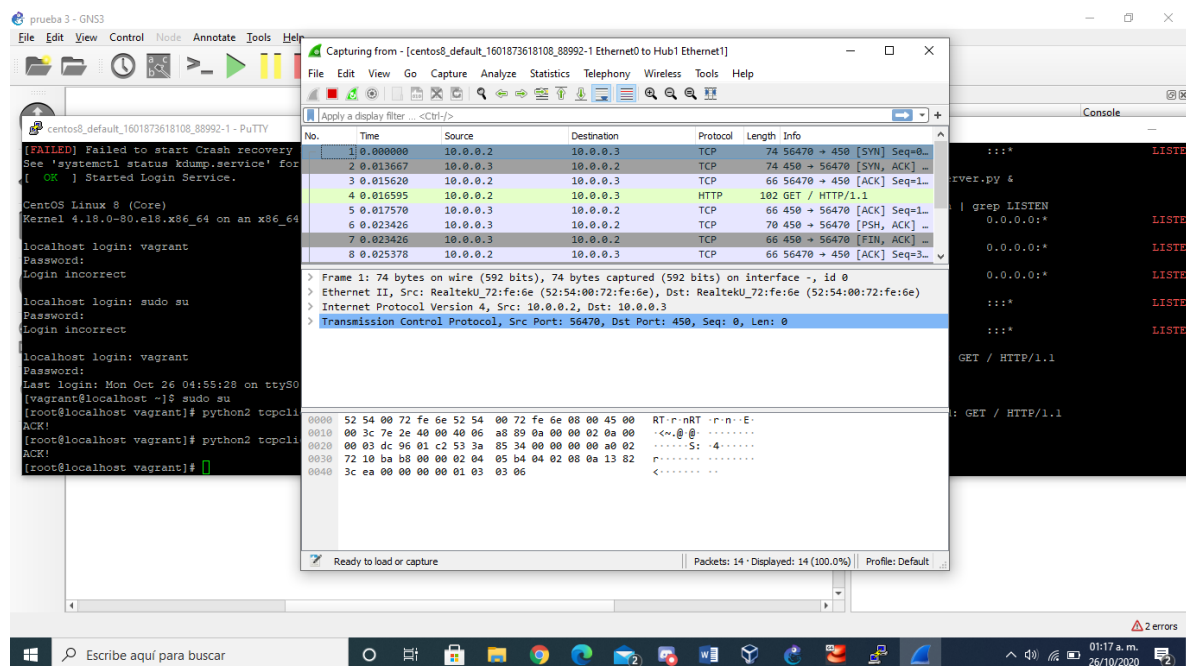
localhost login: vagrant
Password:
Login incorrect

localhost login: sudo su
Password:
Login incorrect

localhost login: vagrant
Password:
Last login: Mon Oct 26 04:55:28 on ttyS0
[vagrant@localhost ~]$ sudo su
[root@localhost vagrant]# python2 tcpclient.py
ACK!
[root@localhost vagrant]#

centos81_default_1602619068612_67026-1 - PuTTY
tcp 0 0 0.0.0.0:111 0.0.0.0:* LISTEN
1/systemd
tcp6 0 0 :::22 :::* LISTEN
584/sshd
tcp6 0 0 :::111 :::* LISTEN
1/systemd
[root@localhost vagrant]# python2 tcpserver.py 4
[1] 857
[root@localhost vagrant]# netstat -tbn | grep LISTEN
tcp 0 0 0.0.0.0:22 0.0.0.0:* LISTEN
584/sshd
tcp 0 0 0.0.0.0:450 0.0.0.0:* LISTEN
857/python2
tcp 0 0 0.0.0.0:111 0.0.0.0:* LISTEN
1/systemd
tcp6 0 0 :::22 :::* LISTEN
584/sshd
tcp6 0 0 :::111 :::* LISTEN
1/systemd
[root@localhost vagrant]# [*] Received: GET / HTTP/1.1
Host: google.com
```

Y después se abre wireshark para poder el tráfico de comunicación de servidor y cliente como en la imagen.



## Fase 5: reporte de conclusiones:

En este reporte sobre gns3 y los scripts de Python y el Wireshark para poder capturar el tráfico de ambos como se puede observar en la imagen está capturando el tráfico de servidor a cliente por medio de Wireshark y nos podemos dar cuenta que el tráfico de datos que están llevando y se puede percatar que se muestra un triplehand que quiere decir eso que los paquetes de datos syn inicia la comunicación con ack quiere decir que está esperando las respuestas del server ya cuando el server y cliente se saludan se termina el proceso con una despedida con fin,ack bueno aparte todo esto que se llevó acabo con los script de Python mi conclusión de este fue que es muy interesante porque te puedes percatar como se comunican ambas máquinas de cliente a servidor y poder ver el tráfico de comunicación que llevan.