Universidad Nacional de General Sarmiento

Sistemas Operativos y Redes I
Comisión 2 Mañana

Trabajo Práctico

Sistemas Operativos y Redes I
TP2 1er Cuatrimestre 2019

Andrés Rojas Paredes

Hvara Ocar

APELLIDO Y NOMBRE	LEGAJO	EMAIL
Tula, Ignacio Mariano	35.226.620/2014	itula@logos.net.ar itula@ungs.edu.ar

Capa Física

Ejercicio 1. Desde la línea de comandos investigar qué interfaces de red tiene disponibles. Que es una interfaz de red?

Se ejecutó el siguiente comando para guardar la salida del comando ifconfig. Dicho resultado se encuentra en ***capaFisica/ipconfig.txt***

```
$: ifconfig > ipconfig.txt
```

```
br-524f6e073a24: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.18.0.1  netmask 255.255.0.0  broadcast 172.18.255.255
        ether 02:42:05:6b:20:ed  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:6f:c7:a7:94  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eno1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.64  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::101e:2730:984b:c52a  prefixlen 64  scopeid 0x20<link>
        ether 48:ba:4e:59:50:0d  txqueuelen 1000  (Ethernet)
        RX packets 42198  bytes 36122226 (36.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 57701  bytes 64148961 (64.1 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Bucle local)
        RX packets 1515  bytes 211409 (211.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1515  bytes 211409 (211.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.65  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::b358:5e1f:573:b8d  prefixlen 64  scopeid 0x20<link>
        ether d4:6a:6a:76:97:ff  txqueuelen 1000  (Ethernet)
        RX packets 1733  bytes 222738 (222.7 KB)
        RX errors 0  dropped 7  overruns 0  frame 0
        TX packets 337  bytes 44282 (44.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Mi computadora (al ser una pc portátil) tiene dos interfaces físicas: “wlo1” que corresponde a la tarjeta de red inalámbrica, y “eno1” que corresponde a la interfaz de red cableada.

Por otro lado se encuentra la típica interfaz virtual “lo” propia de GNU/Linux que sirve para cuando cuando el host al que se quiere comunicar es la misma.

Luego se encuentran dos interfaces más, que se habrán creado en el proceso (de realizar el presente TP) de instalar docker y poner en funcionamiento imágenes del mismo. Una de ellas con prefijo “br” de bridge o puente.

Realizando la investigación para el presente trabajo práctico, se han encontrado dos acepciones para el término “interfaz de red”.

La primera interpretación, tiene que ver con el hardware, con la placa o plaqueta física que permite unir una computadora con una red, y que cumple ciertas funciones como 1)La transmisión y recepción de datos. 2) Acceder al conector o al cable de red. 3) Codificar y decodificar las señales de los cable en otras que puedan ser comprendidas por las capas superiores del modelo OSI.

Por último, se encontró que la “interfaz de red TCP/IP” es en realidad el software específico que se comunica con el controlador (driver) del dispositivo físico de red y la capa IP. Esto es con el fin de que la capa IP tenga una interfaz común con todos las placas o hardware de red que puedan estar instaladas en el mismo host.[1]

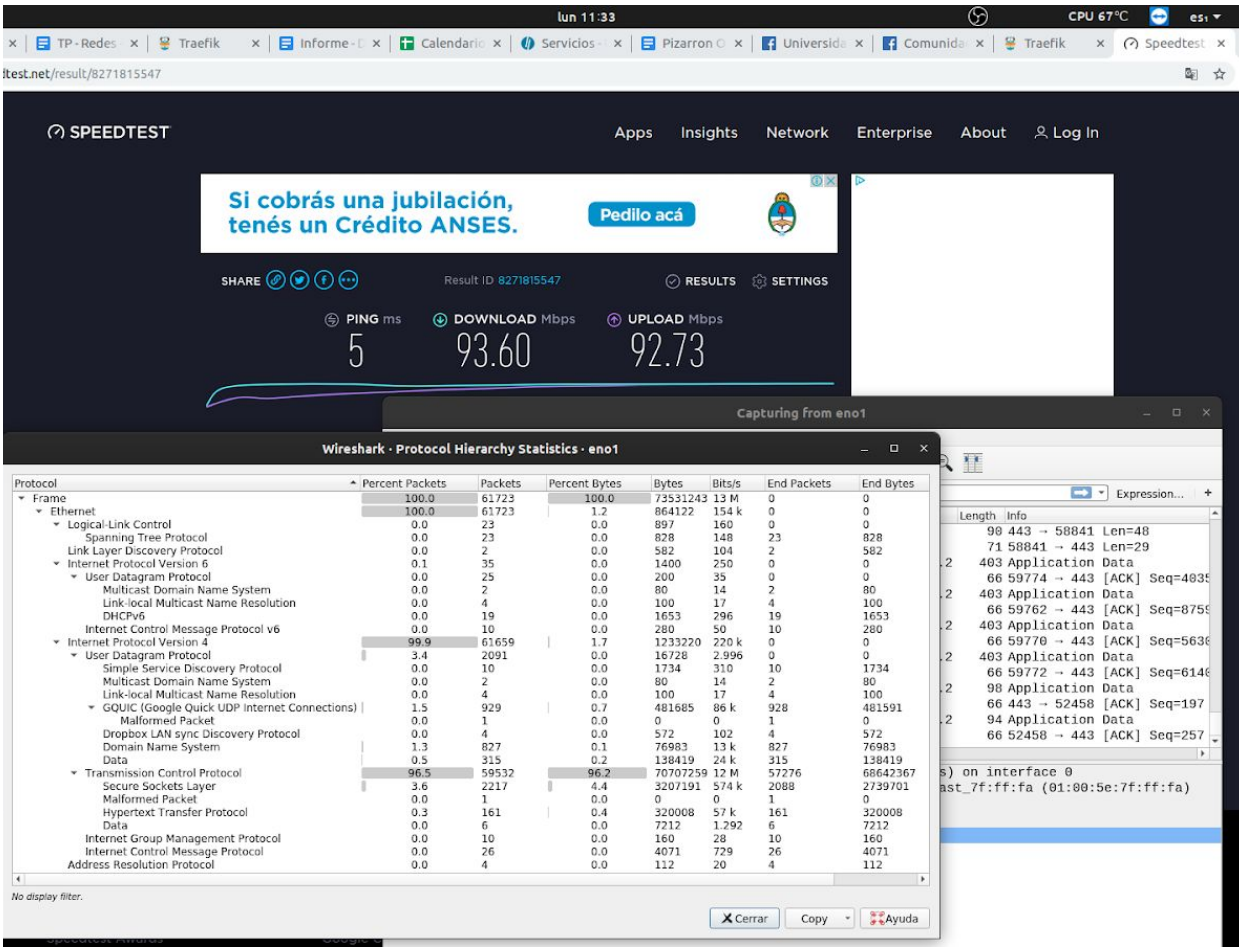
Ejercicio 2. Ejecutar wireshark y escuchar el tráfico en diferentes interfaces de red.

La captura se encuentra guardada en “capaFisica/CapturaWireshark.pcapng”.

Se puede ver el despliegue de la captura en https://youtu.be/Yy8rfpQ_Oo8

[1] https://www.ibm.com/support/knowledgecenter/es/ssw_aix_72/com.ibm.aix.networkcomm/tcpip_interfaces.htm

Ejercicio 3. Determinar el ancho de banda digital de su conexión a internet. Puede usar wireshark y la opción del menú statistics>protocol hierarchy. También puede graficar junto con la opción statistics>IO graph.



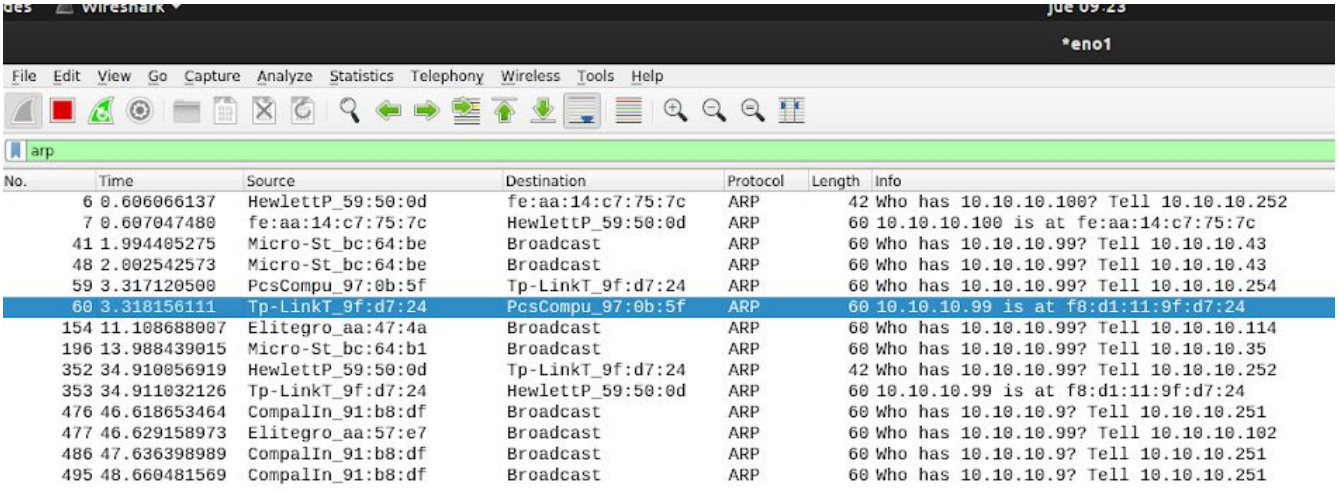
Capa de Enlace

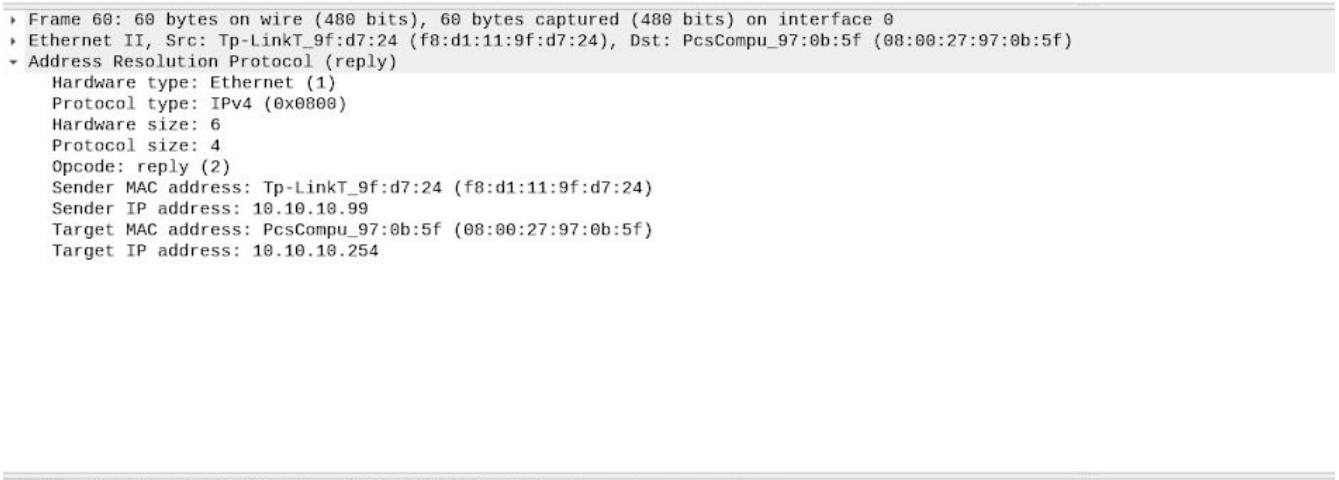
Ejercicio 1. Investigar el protocolo ARP y su relación con las direcciones MAC.

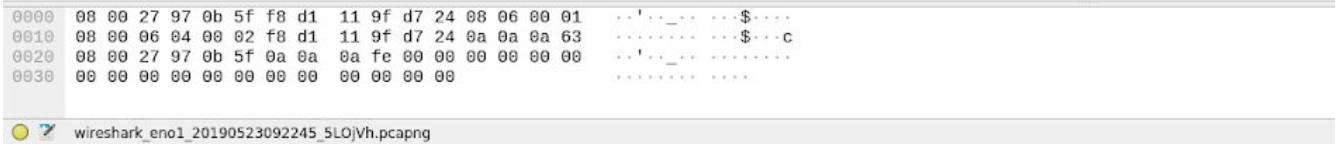
El protocolo ARP que en español significa “Protocolo de Resolución de Direcciones” se utiliza justamente para obtener la dirección MAC de un host con determinada dirección IP. Para ello realiza un broadcast en la red solicitando quién tiene la MAC que corresponde a la IP indicada.

Este protocolo es utilizado cuando el host que pregunta no tiene guardada en su tabla ARP la MAC correspondiente a dicha IP.

Ejercicio 2. Utilizando wireshark realizar una captura de tal forma de identificar un envío ARP y su respuesta.







Ejercicio 3. Sobre su captura identificar los siguientes datos: MAC del origen, IP del destino, MAC del destino.

Se identifican los siguientes datos:

```
MAC Origen:  F8:D1:11:9F:D7:24
MAC Destino: 08:00:27:97:0B:5F
IP Destino:  10.10.10.254
```

Ejercicio 4. Cuando se envía la pregunta “who has ip ….. ” porqué se realiza un broadcast? En ese momento qué datos son desconocidos para el que envía?

En ese momento el dato desconocido por el emisor es la dirección MAC que está asociada con la dirección IP que está consultando. En ese momento, no se encontró en la tabla ARP del dispositivo o del host, dicha entrada conteniendo esa relación.

Se realiza un broadcast para que esa pregunta (¿Quién tiene la IP xxx?) llegue a cada host de la red, para que el quien es aludido responda. Al justamente no saber quien es el host específico ni su dirección MAC, es necesario preguntarlo a toda la red.

Ejercicio 5. Cuando llega la respuesta “ ip ….. is ……” qué nuevo dato se aprende?

En el Wireshark se encontraron respuestas del tipo “xxx.xxx.xxx.xxx” at “xx:xx:xx:xx:xx:xx”. Lo que se aprende es que la dirección IP está asociada con el dispositivo cuya dirección física es la MAC indicada en la respuesta.

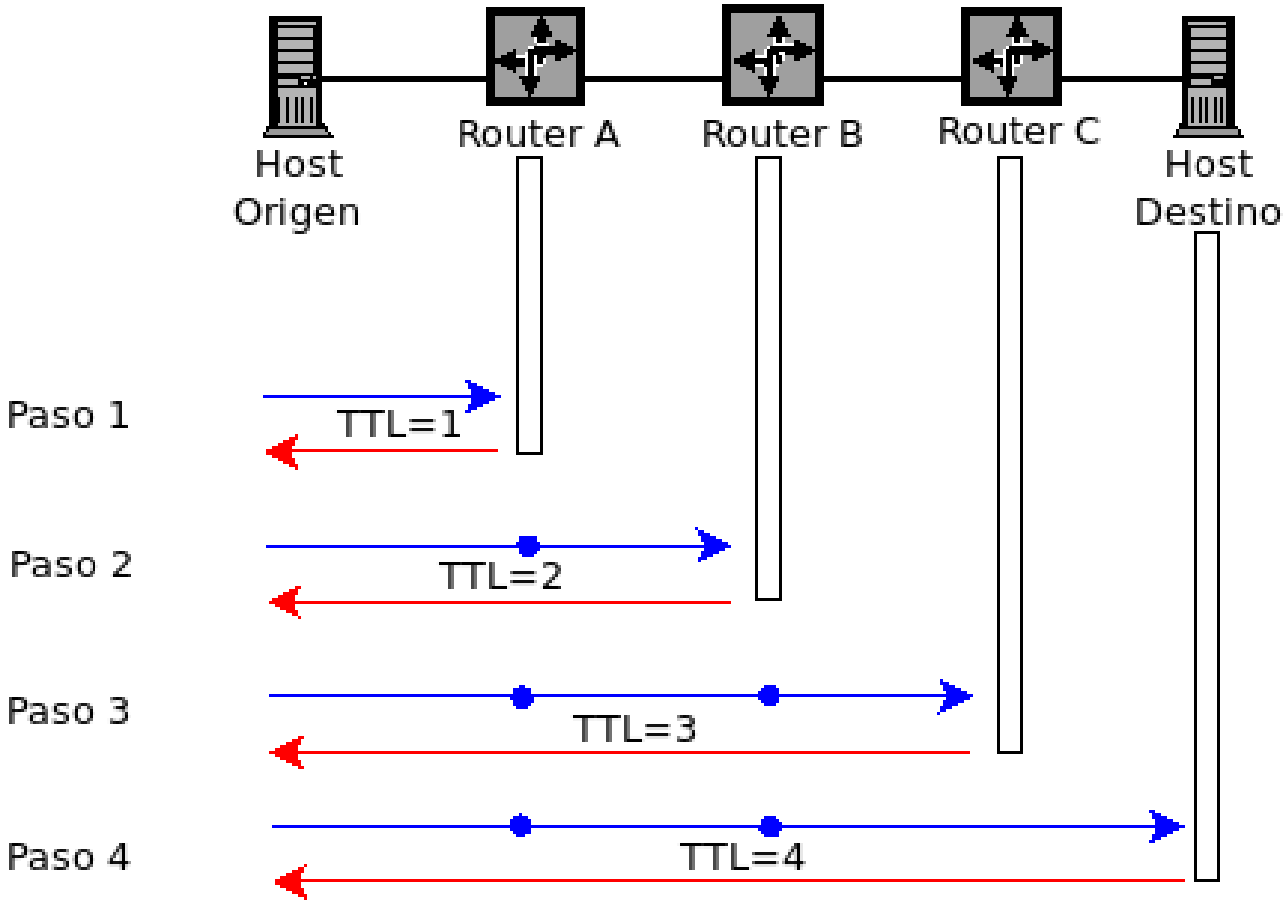
Capa de Red

Ejercicio 1. Investigar cómo funciona traceroute y realizar un diagrama explicativo.

Traceroute funciona utilizando de forma estratégica el campo TTL (Time to live) de los paquetes que envía, sea UDP o ICMP. De esta manera puede reconocer los saltos o hosts intermedios que existen entre el origen y el host de destino, teniendo en cuenta que por cada uno de estos, resta uno al TTL.

De forma iterativa, envía un paquete solicitando llegar al Host de Destino, pero este paquete inicial tendrá un TTL=1. Lo cual producirá en el primer host intermedio que este último devuelva un mensaje de error al emisor indicando que al mensaje enviado le ha expirado el TTL.

Así es como que traceroute, analiza este mensaje de error, para obtener datos como el nombre de host, la IP, el tiempo que tarda en llegar y volver el mensaje. Luego recomienza el proceso pero aumentando el valor de TTL en uno con la finalidad de obtener los mismos datos de los subsiguientes hosts intermedios.



En líneas azules se grafica el mensaje que envía traceroute, generalmente un paquete UDP, o un pedido de eco con ICMP. En líneas rojas, una respuesta del host indicando que hubo error, que se excedió el TTL del mensaje y que por ende se descartó el envío del mensaje azul.

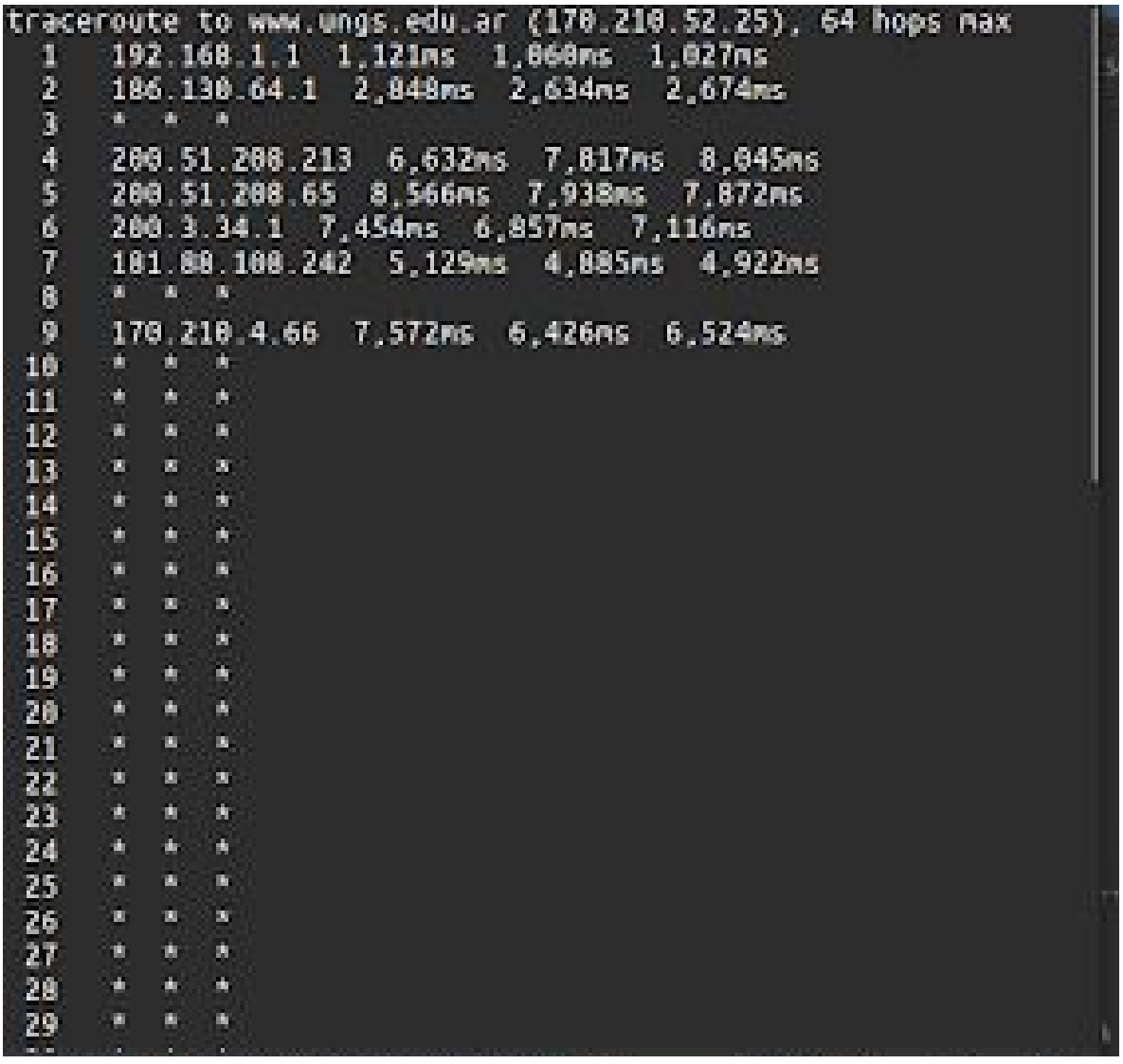
Ejercicio 2. Instalar traceroute: sudo apt-get install traceroute

Se instaló correctamente traceroute

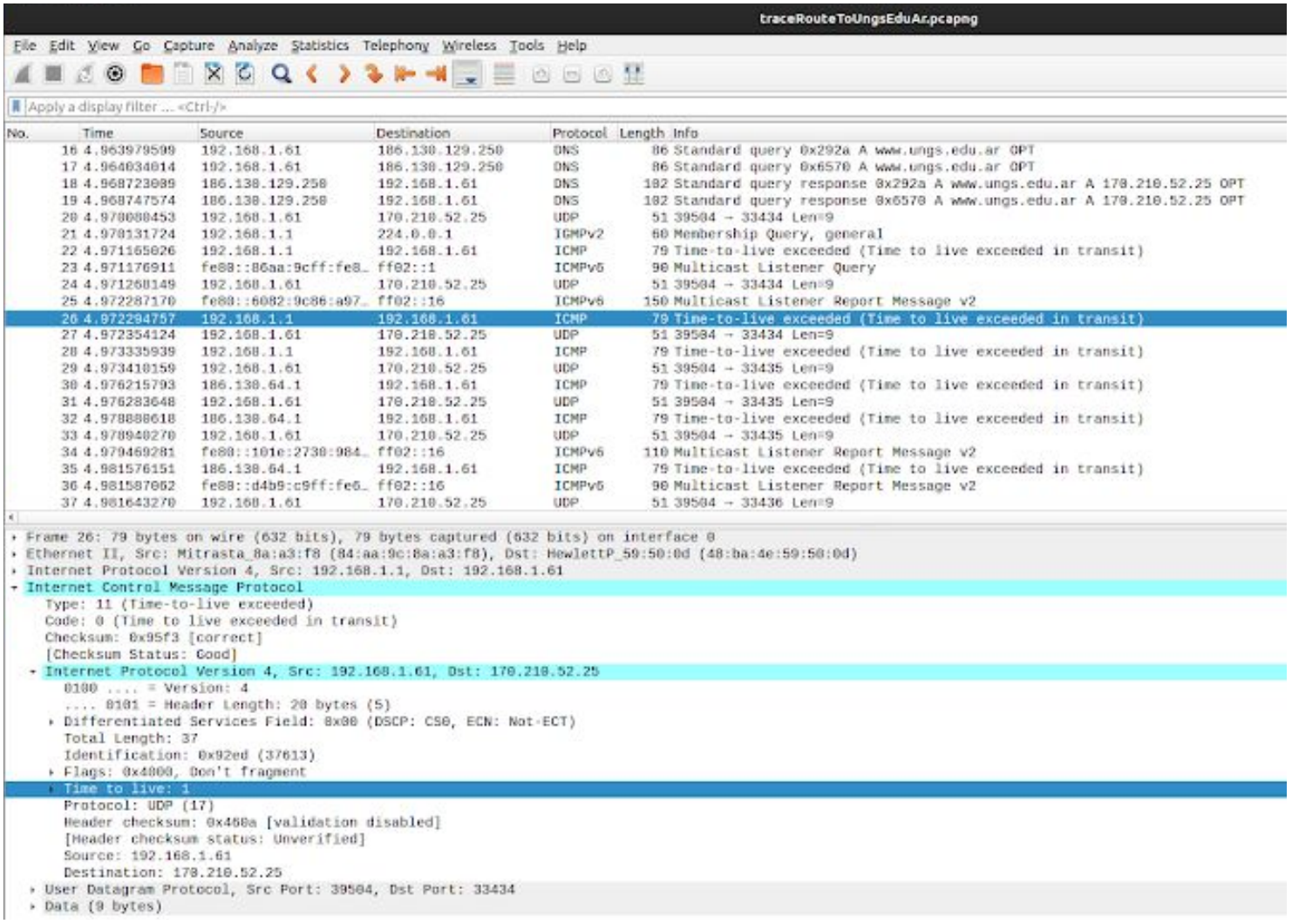
Ejercicio 3. Ejecutar traceroute mientras realiza una captura con wireshark. Que función cumple el protocolo ICMP?

Se ejecutó el siguiente comando para guardar la salida del comando traceroute. Dicho resultado se encuentra en ***"capaRed/Ejercicio 03/traceRouteToUngsEduAr.txt"***

```
$: traceroute www.ungs.edu.ar > traceRouteToUngsEduAr.txt
```



El resultado de la escucha de Wireshark se encuentra en ***"capaRed/Ejercicio 03/traceRouteToUngsEduAr.traceRouteToUngsEduAr.pcapng"***



El protocolo ICMP cumple funciones de información operativa de la capa de red. Es decir que sirve para avisar cuestiones del propio funcionamiento de la red. En este caso, sirve para avisar que el mensaje original no se entregó porque el TTL expiró. En otros casos, sirve para avisar que el host de destino no se puede alcanzar, etc.

Ejercicio 4. Ejecutar traceroute mientras realiza una captura con wireshark. Que función cumple el protocolo ICMP?

Qué direcciones ip aparecen en la columna Source o fuente, a qué hosts pertenecen estas direcciones ip?

En la columna “source” aparece la dirección IP de la computadora o host donde nos encontramos ejecutando el traceroute, y también aparecen la dirección IP de los host intermedios respondiendo el mensaje ICMP de error indicando la expiración del TTL.

Porque los mensajes tienen el campo info igual a “Time-to-live exceeded”?

Es el mensaje que espera traceroute para, con el TTL estratégico que se envió, obtener los datos de ese host que se encuentra en el camino hacia nuestro host de destino. Aprovechando “la solidaridad” del host intermedio en avisar que el mensaje con nuestro TTL expirado no llegará a destino, para obtener cierta información, como su dirección IP, nombre de host, etc.

Comparar estos datos con la salida standard de traceroute y marcar similitudes y diferencias, por ejemplo su salida puede ser:

La salida del TP

```
~$ traceroute google.com
traceroute to google.com (172.217.30.174), 30 hops max, 60 byte packets
 1  gateway (192.168.1.1)  2.250 ms  2.855 ms  4.311 ms
 2  186-38-31-104.mrse.com.ar (186.38.31.104)  10.893 ms  10.912 ms  12.482 ms
 3  201.251.0.234 (201.251.0.234)  13.445 ms  14.534 ms  16.488 ms
 4  rtrpc1.interbourg.com.ar (186.177.192.194)  17.678 ms  18.106 ms  19.825 ms
 5  200.51.235.29 (200.51.235.29)  29.175 ms  30.814 ms  31.544 ms
 6  74.125.52.126 (74.125.52.126)  33.249 ms  23.985 ms  23.042 ms
 7  108.170.248.241 (108.170.248.241)  23.367 ms  24.609 ms 108.170.248.225 (108.170.248.225)
 8  216.239.62.245 (216.239.62.245)  23.022 ms  24.215 ms  25.311 ms
 9  eze03s36-in-f14.1e100.net (172.217.30.174)  22.012 ms  23.715 ms  24.028 ms
```

Se ejecutó el comando:

```
$: traceroute google.com --resolve-hostnames > traceRouteToGoogleCom.txt
```

La salida de mi PC

```
~$ traceroute google.com --resolve-hostnames > traceRouteToGoogleCom.txt
traceroute to google.com (172.217.30.174), 64 hops max
   1   192.168.1.1 (_gateway)  1,241ms  1,088ms  0,939ms
   2   186.130.64.1 (186-130-64-1.speedy.com.ar)  2,676ms  2,612ms  2,426ms
   3   *  *  *
   4   200.51.208.213 (200.51.208.213)  8,810ms  9,369ms  7,787ms
   5   *  *  *
   6   200.51.240.181 (200.51.240.181)  7,203ms  5,027ms  7,876ms
   7   74.125.52.126 (74.125.52.126)  3,889ms  3,938ms  3,825ms
   8   108.170.248.241 (108.170.248.241)  4,847ms  4,758ms  4,756ms
   9   216.239.62.243 (216.239.62.243)  4,410ms  4,279ms  4,232ms
  10   172.217.30.174 (eze03s36-in-f14.1e100.net)  4,410ms  4,349ms  4,035ms
```

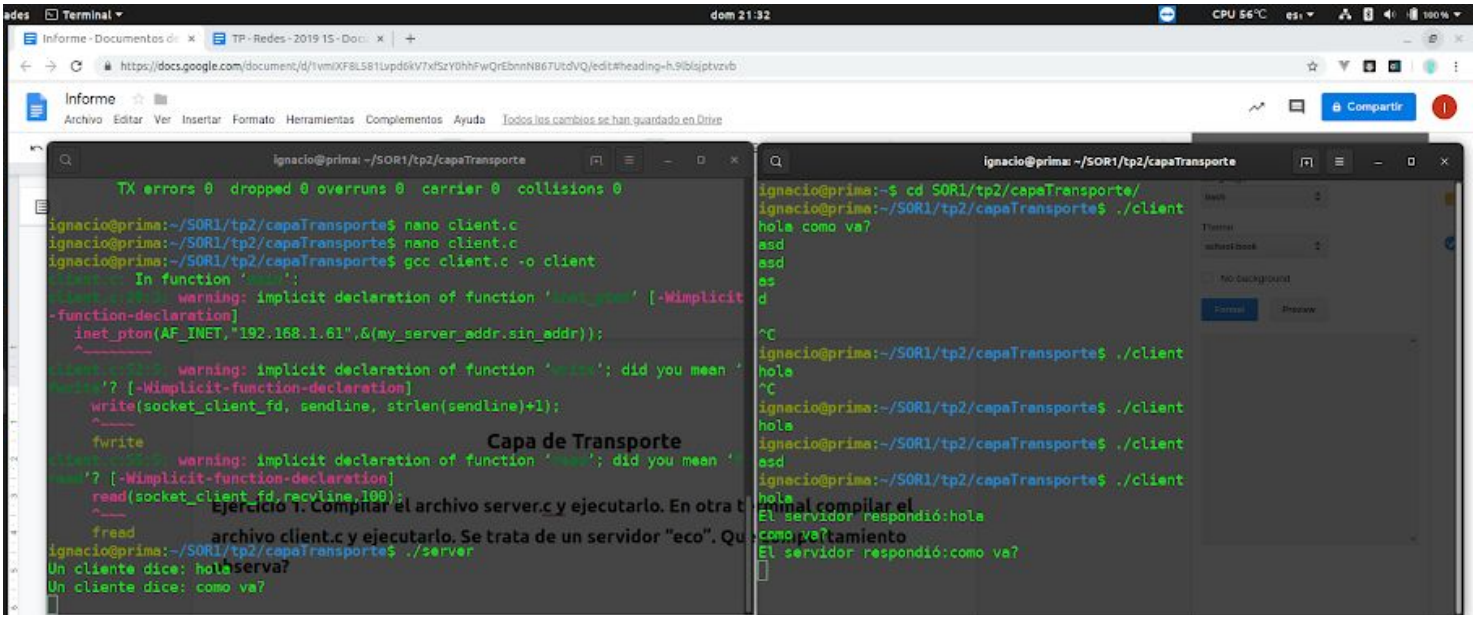
En verde, la similitud, tanto el ejemplo como en la red de mi hogar, las mismas están configuradas en la misma red 192.168.1.0 y ambas puertas de enlace a internet se encuentran en 192.168.1.1

En rojo las diferencias por poseer distintos proveedores de internet, en mi caso es Movistar Fibra (antes comercializada como Speedy), en el caso del TP utiliza la Cooperativa Telefónica de Grand Bourg.

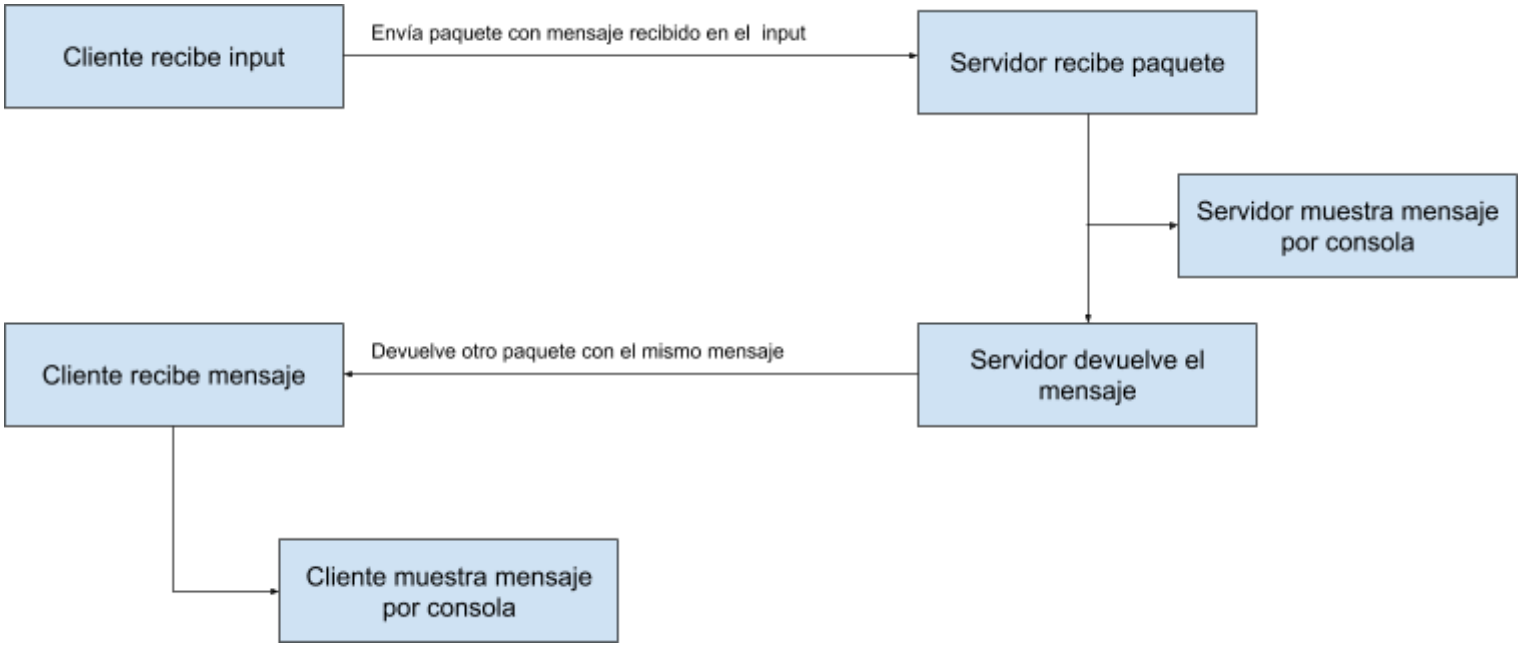
En azul las similitudes, pues ya nos encontramos dentro de la infraestructura de red de Google.

Capa de Transporte

Ejercicio 1. Compilar el archivo server.c y ejecutarlo. En otra terminal compilar el archivo client.c y ejecutarlo. Se trata de un servidor “eco”. Qué comportamiento observa?



Se observa que es necesario ejecutar el servidor primero, y luego el cliente.



Ejercicio 2. Modificar los valores del socket para que cliente y servidor se ejecuten en diferentes computadoras del laboratorio.

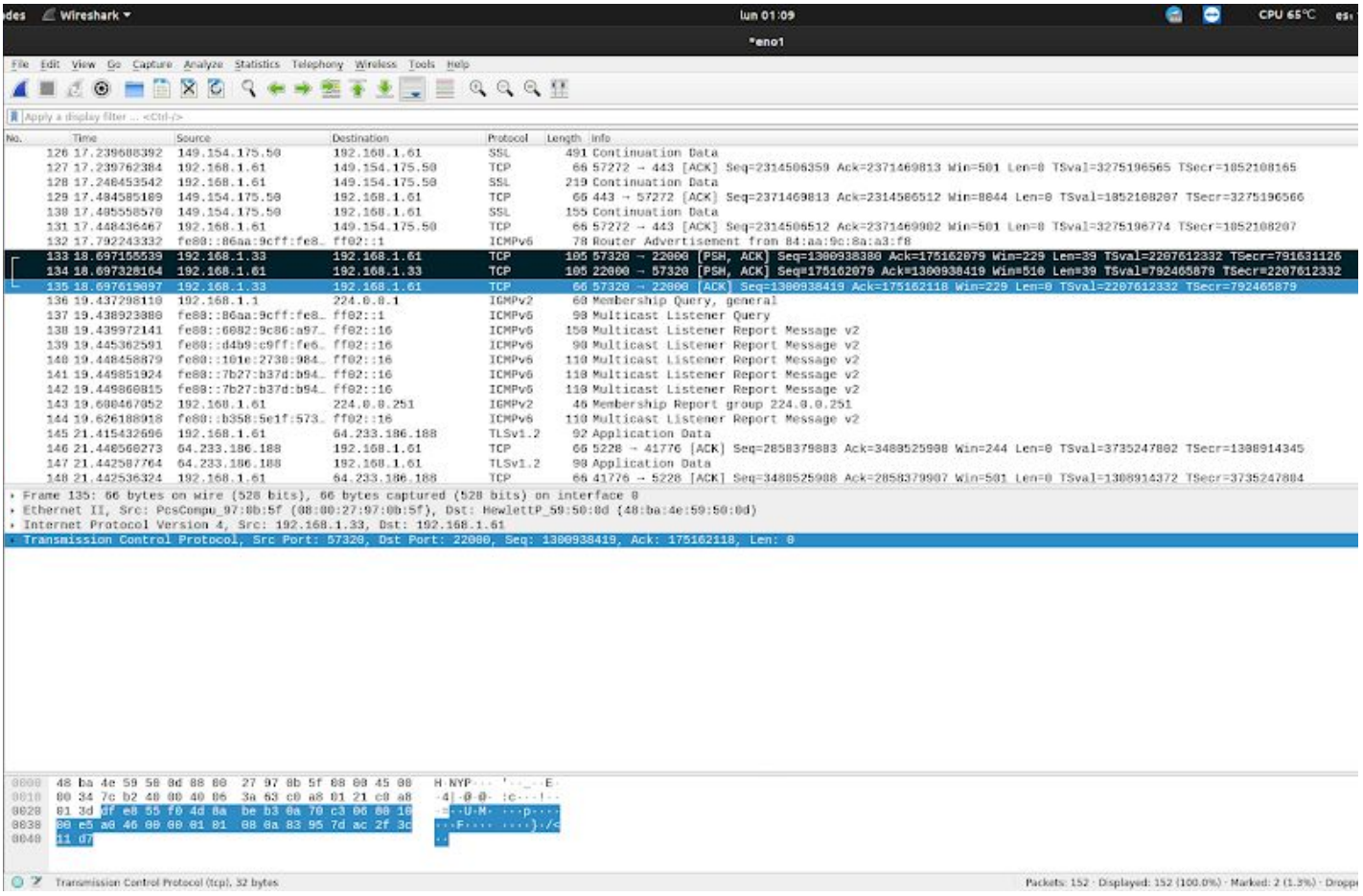
Se configuró una computadora virtual para realizar dichas conexiones. Se utiliza la interface cableada como dispositivo adaptador puente.

Ejercicio 3. Realizar una captura del tráfico entre el cliente y el servidor y recopilar los siguientes datos:

Elegir una interfaz de red sobre la cual escuchar: sobre el cliente o sobre el servidor.

Se analizó la interface del servidor.

Identificar el momento del three way handshake y detallar la información que se intercambia. Como se utiliza esta información para implementar la confiabilidad de TCP?



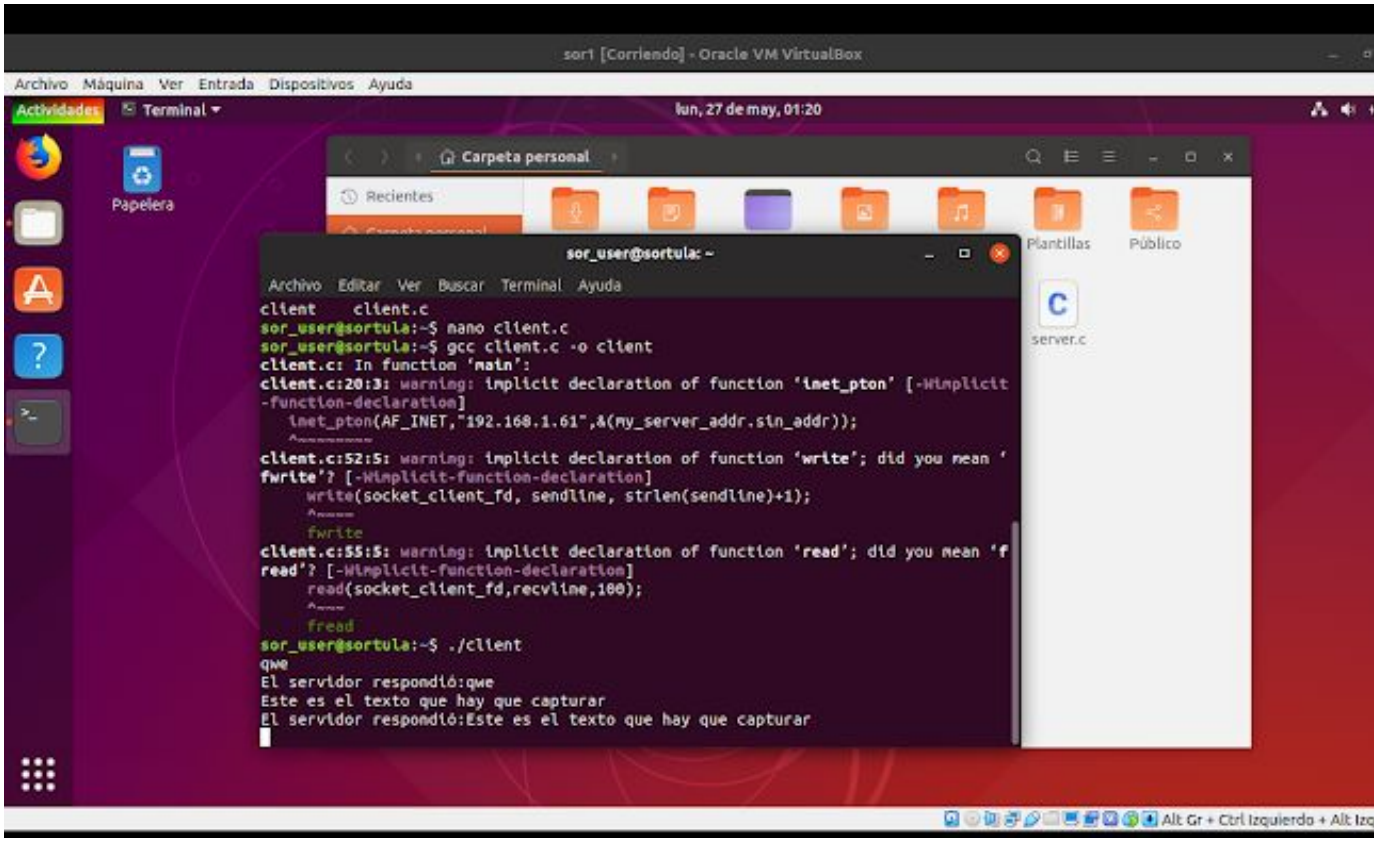
Los dos registros en negro y el tercero en azul es el momento del three way handshake

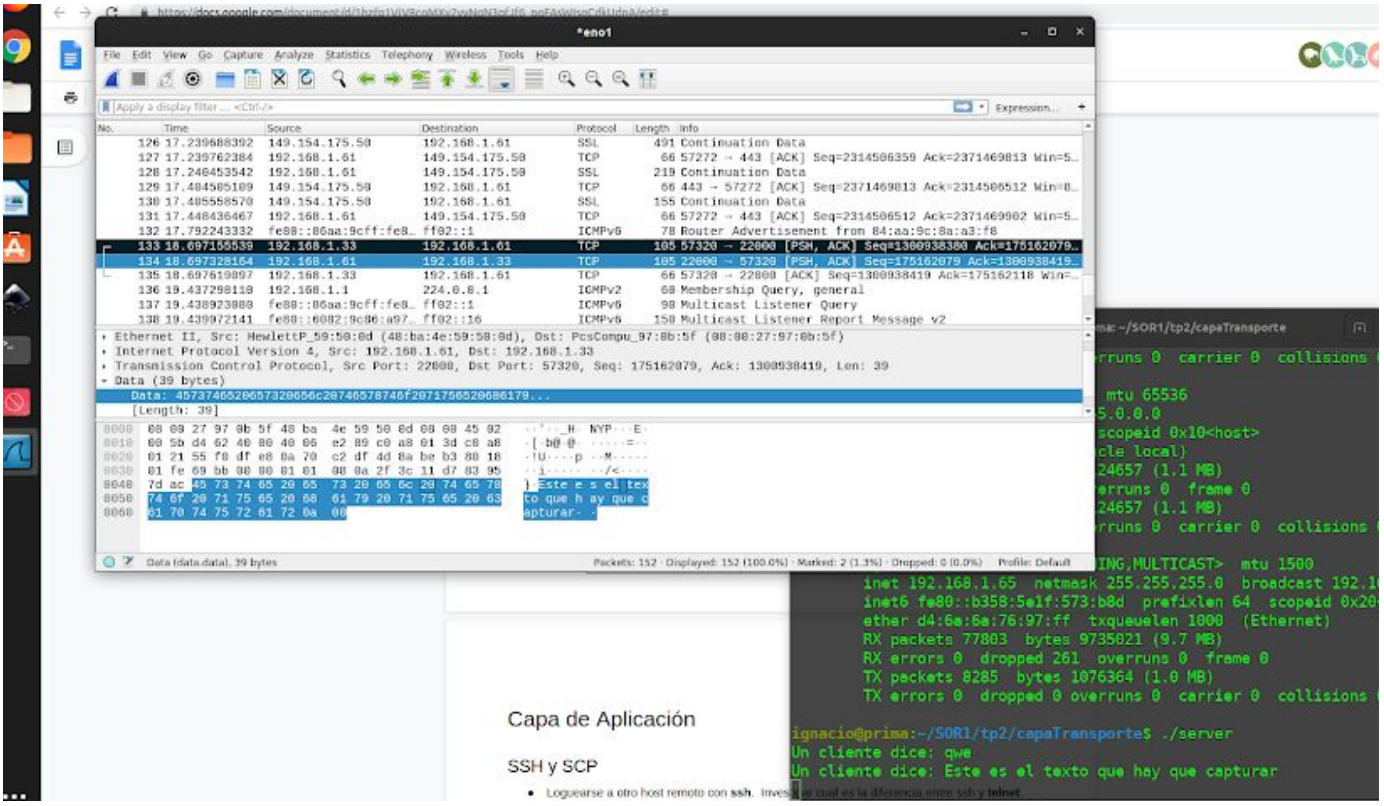
La confiabilidad, es decir el asegurar que los datos sean **recibidos**, **sin errores** y en el **orden enviado**, se implementa en el protocolo TCP, en esta serie de tres pasos, donde servidor y cliente se anuncian mutuamente.

El cliente en primer lugar, al iniciar la conexión a un puerto del servidor. El servidor luego respondiendo a la conexión inicial, y por último el cliente avisando al servidor que recibió el mensaje anterior.

1) El cliente envía un mensaje “ABC”, y le pide al servidor ¿Me confirma si recibió el mensaje?
2) El servidor le responde: “Le confirmo que recibí su mensaje ABC” y le pide al cliente “¿Me confirma que recibió mi confirmación de que recibí su mensaje?”
3) El cliente envía un último mensaje al servidor “Le confirmo, que recibí la confirmación de que recibió mi mensaje”.

Realizar un seguimiento del texto plano que se intercambió entre cliente y servidor. Puede usar la opción de wireshark Menu>Analize>Follow>TCPstream





Qué cambios piensa que se debe hacer en la comunicación para que la captura de wireshark no se pueda leer o entender como en el punto anterior?

El cambio que se debe realizar para evitar la lectura de cualquier persona en el medio leyendo la comunicación es implementar alguna mínima técnica de criptografía.

La más sencilla de implementar (pero no muy segura), y sin requerir técnica de programación mediante, es utilizar una clave de cifrado, previamente compartida entre cliente y servidor.

Esta clave, permitirá cambiar el texto original por un texto ilegible para los terceros que quieran leer, pero que volverá a tener significado cuando el emisor lo decodifique.

Completar la siguiente tabla:
En la red de mi hogar:

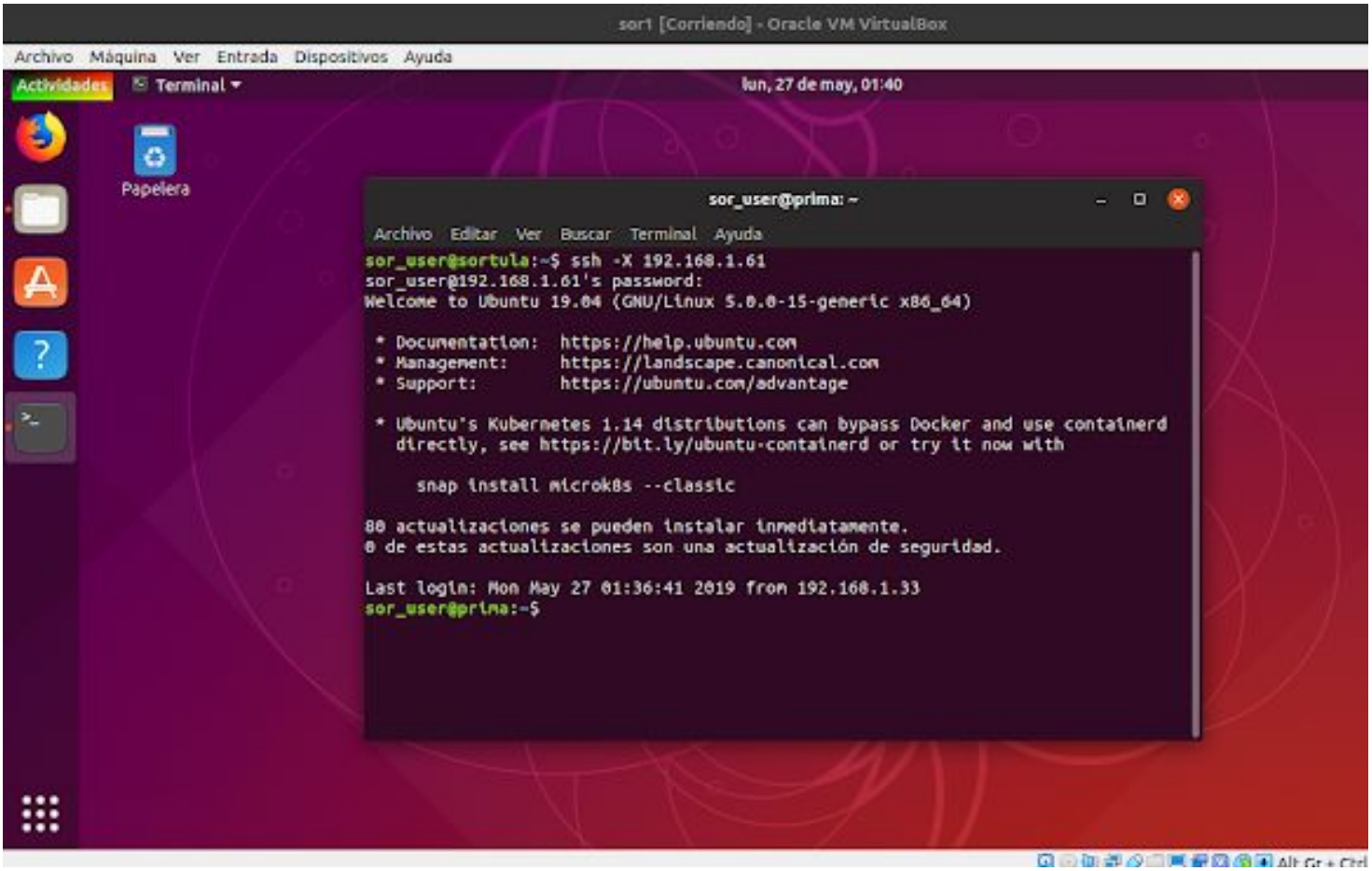
	Nro de IP	Nro de puerto
Socket cliente	192.168.1.33	57320
Socket servidor	192.168.1.61	22000

Capa de Aplicación

Ejercicio 1. Loguearse a otro host remoto con ssh. Investigar cual es la diferencia entre ssh y telnet. Realizar un login remoto con la posibilidad de lanzar aplicaciones gráficas. Una vez logueado, el usuario debería poder ejecutar programas de interfaz gráfica como gedit, emacs ó wireshark.

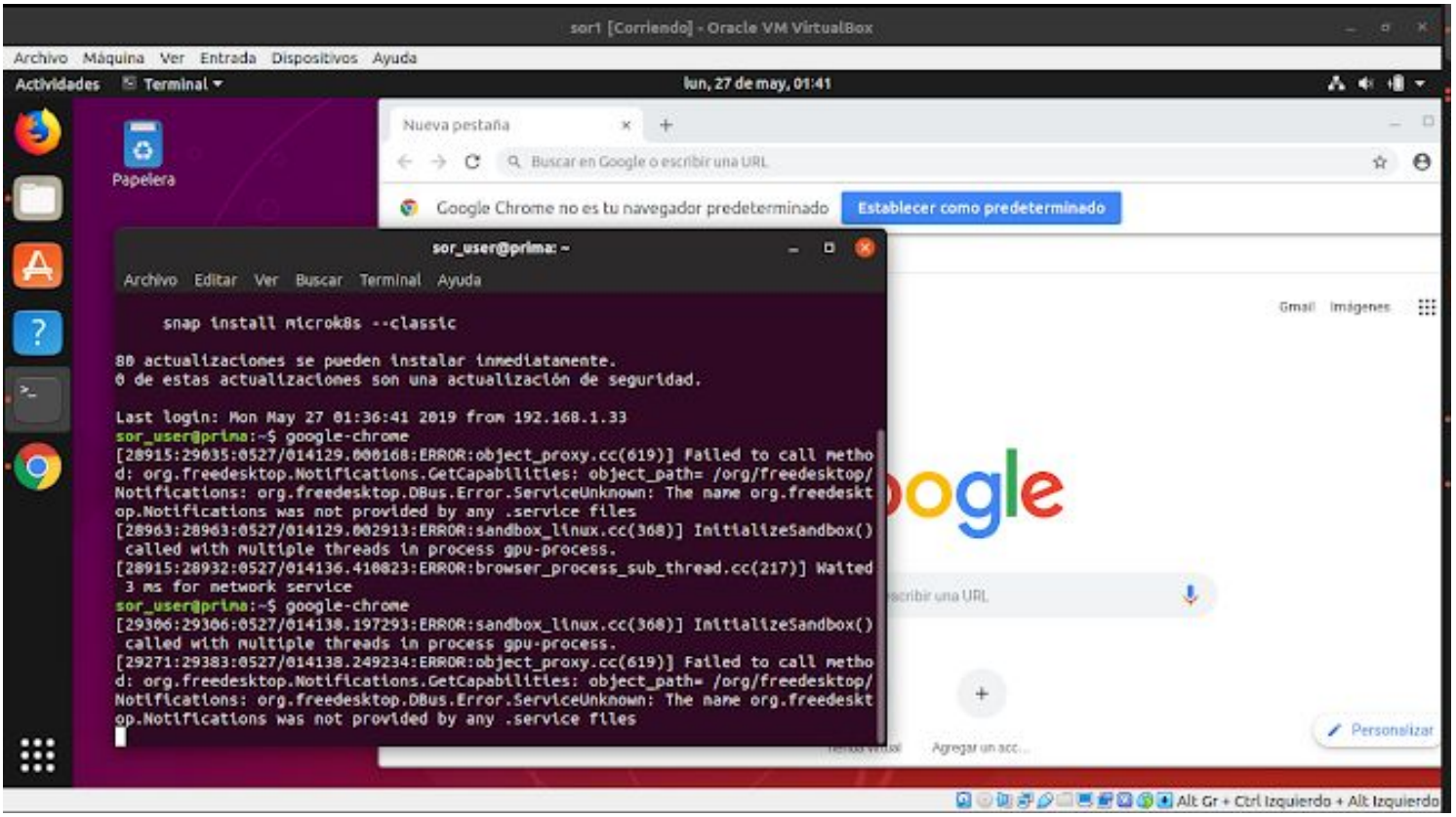
El login remoto se realiza desde la pc virtual hasta la computadora host física, utilizando el comando

```
ssh -X 192.168.1.61 -l sor_user
```



Sobre esta pc virtual con ubuntu 18.04 casi sin muchas configuraciones y donde no se ha instalado Google Chrome (a diferencia de la computadora física), se ejecutó el comando.

```
$: google-chrome
```



La diferencia que existe entre telnet y ssh son básicamente dos. La primera y maś sencilla: telnet utiliza el puerto 23 y ssh el puerto 22. La segunda y más compleja, es que ssh provee de comunicación encriptada entre cliente y servidor, mientras que en telnet la comunicación es en texto plano. SSH es el reemplazo de telnet.

Ejercicio 2. Realizar una transferencia de archivos entre diferentes hosts. Usar scp. Cual es la relación entre ssh y scp?

Se ejecutó el siguiente comando

```
$: scp movemeHaciaVirtual.txt sor_user@192.168.1.33:/home/sor_user/movidoDesdeFisica.txt
```

De esa forma se copió y renombró en la ubicación destino el archivo ubicado en “capaAplicacion/movemeHaciaVirtual.txt” hacia la pc virtual cómo “movidoDesdeFisica.txt”

La relación existente entre scp y ssh, es que el primero utiliza una conexión cifrada basada en ssh para realizar la transferencia de archivos.

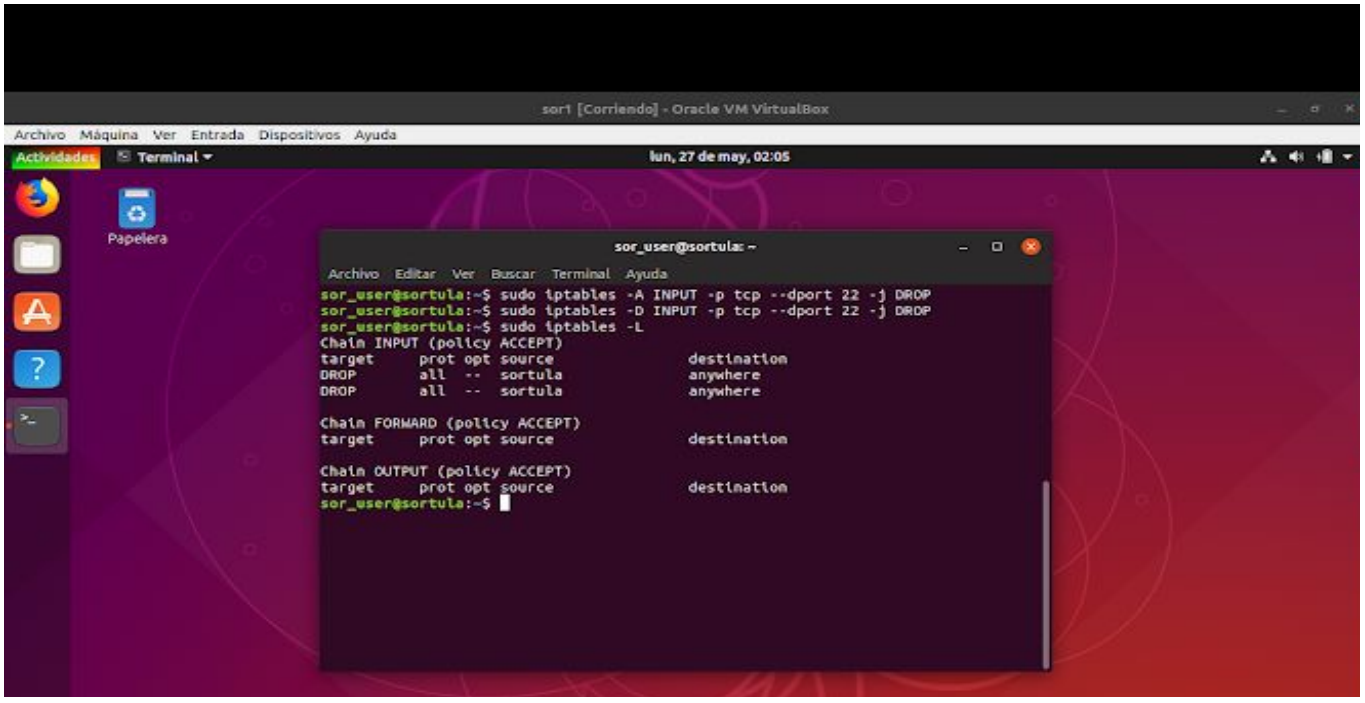
Ejercicio 3. Agregar restricciones de acceso a su host, por ejemplo con iptables

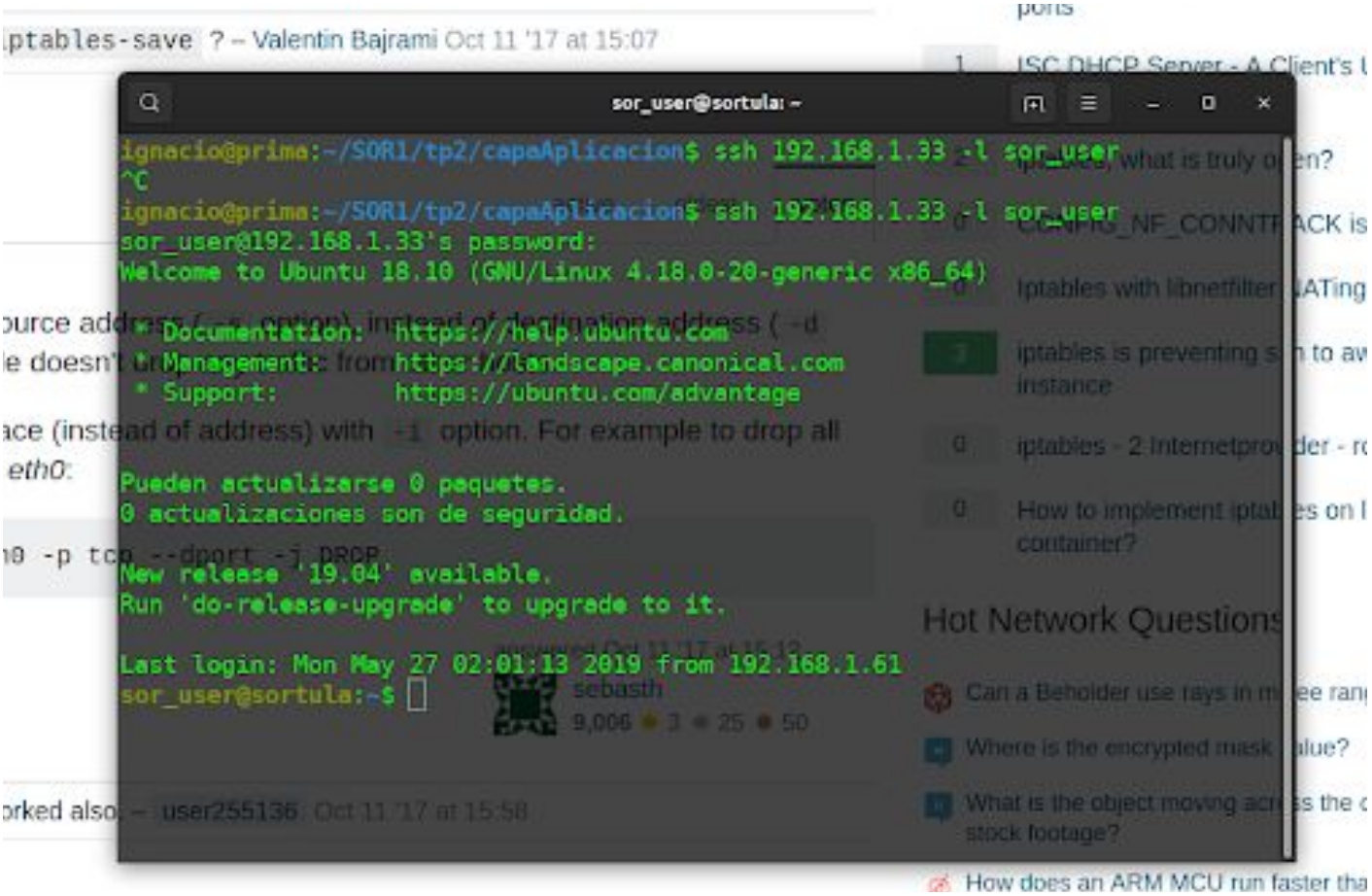
Para bloquear:

```
$: sudo iptables -A INPUT -p tcp --dport 22 -j DROP
```

Para eliminar el bloqueo

```
$: sudo iptables -D INPUT -p tcp --dport 22 -j DROP
```


sor1 [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
Actividades
Terminal
lun, 27 de may, 02:05
Papelera
sor_user@sortula: ~
Archivo Editar Ver Buscar Terminal Ayuda
sor_user@sortula:~$ sudo iptables -A INPUT -p tcp --dport 22 -j DROP
sor_user@sortula:~$ sudo iptables -D INPUT -p tcp --dport 22 -j DROP
sor_user@sortula:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target prot opt source destination
DROP all -- sortula anywhere
DROP all -- sortula anywhere
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
sor_user@sortula:~$


sor_user@sortula: ~
ignacio@prima:~/SOR1/tp2/capaAplicacion$ ssh 192.168.1.33 -l sor_user
^C
ignacio@prima:~/SOR1/tp2/capaAplicacion$ ssh 192.168.1.33 -l sor_user
sor_user@192.168.1.33's password:
Welcome to Ubuntu 18.10 (GNU/Linux 4.18.0-20-generic x86_64)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
Pueden actualizarse 0 paquetes.
0 actualizaciones son de seguridad.
New release '19.04' available.
Run 'do-release-upgrade' to upgrade to it.
Last login: Mon May 27 02:01:13 2019 from 192.168.1.61
sor_user@sortula:~$

Instalar Docker Community Edition (CE) (link)
Instalar Docker Compose (link)
Seguir las instrucciones del proxy reverso Traefik - Getting Started (link)
Ingresar al panel de traefik, que debería quedar funcionando en http://localhost:8080/dashboard/
Reproducir la imagen anterior y verificar que efectivamente se accede a diferentes servidores.

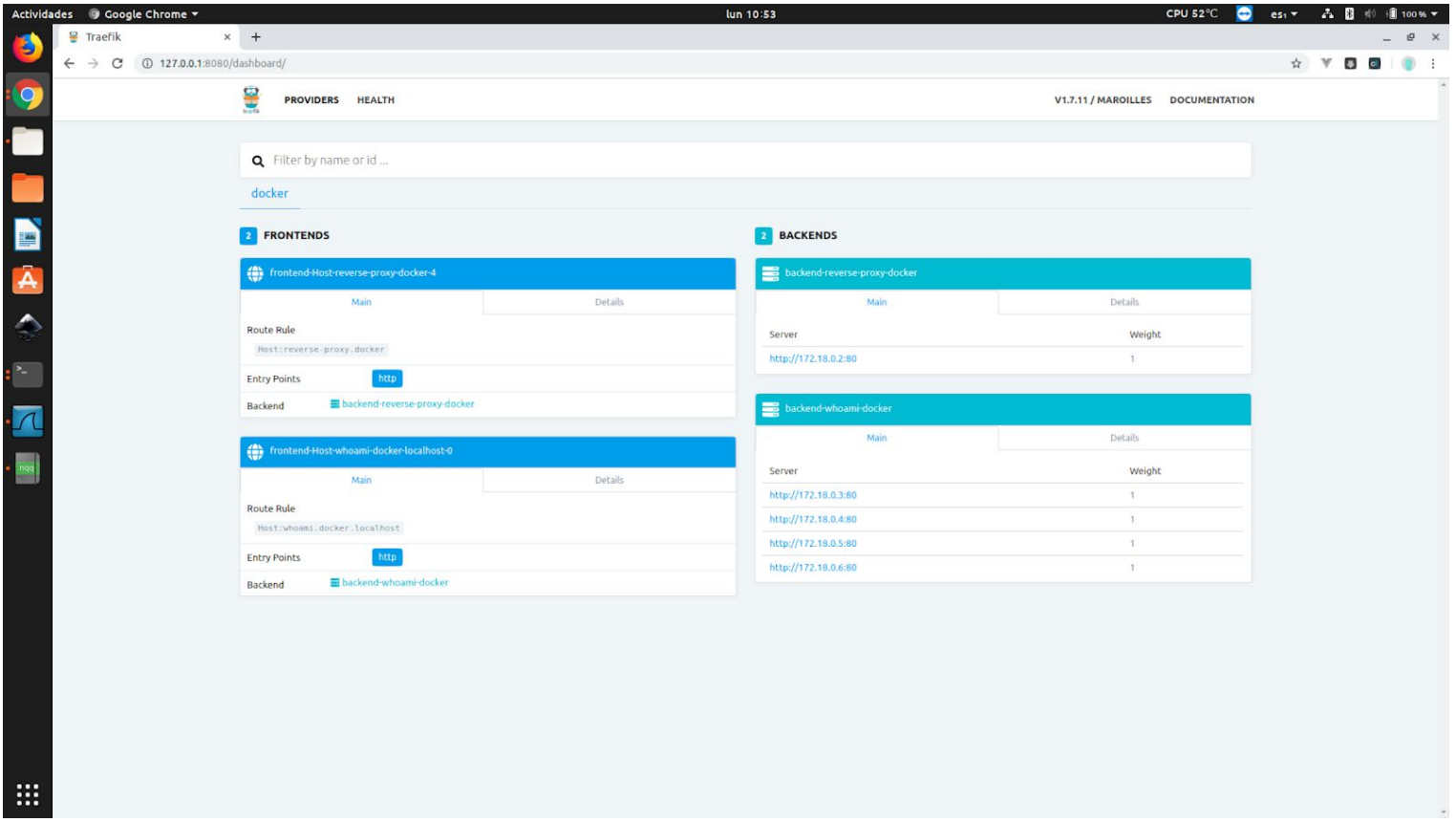
Se instaló docker.io que es la versión de Docker mantenida por canonical y Docker-Compose del github oficial.

Se siguieron las instrucciones de Traefik.

Se podrá encontrar en la ruta “capaAplicacion/docker” el “docker-compose.yml” correspondiente.

En dicha ubicación se ejecutó:

```
$: sudo service apache2 stop # tengo un servidor web apache corriendo, debo apagarlo
$: sudo docker-compose up -d reverse-proxy
$: sudo docker-compose up -d whoami
$: sudo docker-compose up -d whoami
$: sudo docker-compose up -d whoami
```



Se ejecutó varias veces el comando:

```
$: curl -H Host:whoami.docker.localhost http://127.0.0.1 > whoami.txt
```

Puede encontrarse el archivo whoami.txt en “capaAplicacion/docker/whoami.txt”

```
Hostname: c62f6c80abfa
IP: 127.0.0.1
IP: 172.18.0.4
GET / HTTP/1.1
Host: whoami.docker.localhost
User-Agent: curl/7.64.0
Accept: */*
Accept-Encoding: gzip
X-Forwarded-For: 172.18.0.1
X-Forwarded-Host: whoami.docker.localhost
X-Forwarded-Port: 80
X-Forwarded-Proto: http
X-Forwarded-Server: fcf897307c6d
X-Real-Ip: 172.18.0.1

Hostname: 47bcf3cb5509
IP: 127.0.0.1
IP: 172.18.0.5
GET / HTTP/1.1
Host: whoami.docker.localhost
User-Agent: curl/7.64.0
Accept: */*
Accept-Encoding: gzip
X-Forwarded-For: 172.18.0.1
X-Forwarded-Host: whoami.docker.localhost
X-Forwarded-Port: 80
X-Forwarded-Proto: http
X-Forwarded-Server: fcf897307c6d
X-Real-Ip: 172.18.0.1

Hostname: 5cd6b932d255
IP: 127.0.0.1
IP: 172.18.0.6
GET / HTTP/1.1
Host: whoami.docker.localhost
User-Agent: curl/7.64.0
Accept: */*
Accept-Encoding: gzip
X-Forwarded-For: 172.18.0.1
X-Forwarded-Host: whoami.docker.localhost
X-Forwarded-Port: 80
X-Forwarded-Proto: http
X-Forwarded-Server: fcf897307c6d
X-Real-Ip: 172.18.0.1

Hostname: 9480b60dc3d7
IP: 127.0.0.1
IP: 172.18.0.3
GET / HTTP/1.1
Host: whoami.docker.localhost
```

```
User-Agent: curl/7.64.0
Accept: */*
Accept-Encoding: gzip
X-Forwarded-For: 172.18.0.1
X-Forwarded-Host: whoami.docker.localhost
X-Forwarded-Port: 80
X-Forwarded-Proto: http
X-Forwarded-Server: fcf897307c6d
X-Real-Ip: 172.18.0.1

Hostname: c62f6c80abfa
IP: 127.0.0.1
IP: 172.18.0.4
GET / HTTP/1.1
Host: whoami.docker.localhost
User-Agent: curl/7.64.0
Accept: */*
Accept-Encoding: gzip
X-Forwarded-For: 172.18.0.1
X-Forwarded-Host: whoami.docker.localhost
X-Forwarded-Port: 80
X-Forwarded-Proto: http
X-Forwarded-Server: fcf897307c6d
X-Real-Ip: 172.18.0.1

Hostname: 47bcf3cb5509
IP: 127.0.0.1
IP: 172.18.0.5
GET / HTTP/1.1
Host: whoami.docker.localhost
User-Agent: curl/7.64.0
Accept: */*
Accept-Encoding: gzip
X-Forwarded-For: 172.18.0.1
X-Forwarded-Host: whoami.docker.localhost
X-Forwarded-Port: 80
X-Forwarded-Proto: http
X-Forwarded-Server: fcf897307c6d
X-Real-Ip: 172.18.0.1
```