

Universidad Nacional de General Sarmiento

Sistemas Operativos y Redes

Trabajo Práctico nº2 "Redes"

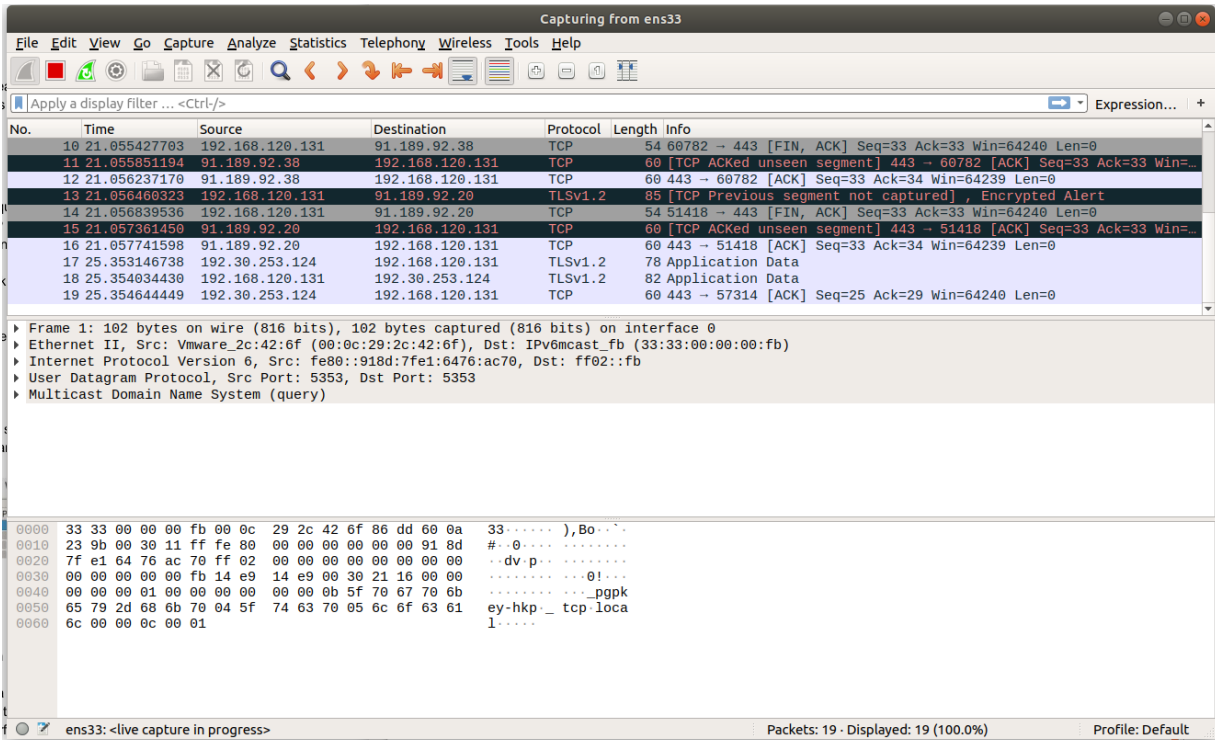
INFORME

Alumna: María Sol Hoyos

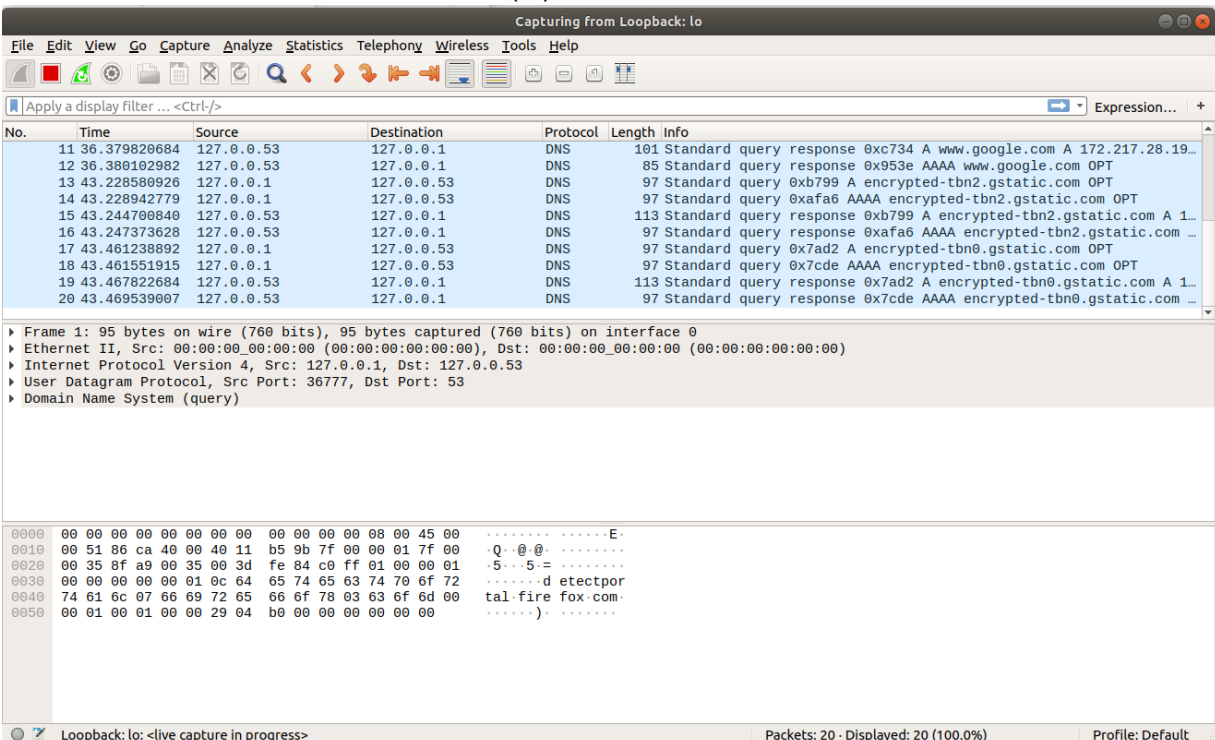
CAPA FÍSICA

Una interfaz de red o tarjeta de red es un componente de hardware que conecta una computadora a una red informática y que posibilita compartir recursos entre dos o más computadoras. Para comprobar las interfaces de red se utiliza el comando: ifconfig. Encontré las siguientes interfaces:

Tráfico de la interfaz de red Ethernet (ens33):



Traffic de la interfaz de red Localhost (lo):



El ancho de banda digital de mi conexión a internet es: 138 Bits/s

Wireshark · Protocol Hierarchy Statistics · ens33								
Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
▼ Frame	100.0	403	100.0	230603	5,678	0	0	0
▼ Ethernet	100.0	403	2.4	5642	138	0	0	0
▼ Internet Protocol Version 6	2.2	9	0.2	360	8	0	0	0
▼ User Datagram Protocol	1.2	5	0.0	40	0	0	0	0
Multicast Domain Name System	0.7	3	0.1	117	2	3	117	2

CAPA DE ENLACE

El protocolo ARP (Protocolo de resolución de direcciones) es un protocolo de comunicaciones que se encuentra entre la capa de enlace y la capa de red. Su principal objetivo es conocer la dirección física (MAC) de una tarjeta de interfaz de red correspondiente a una dirección IP.

Envío ARP y su respuesta:

Wireshark capture showing ARP request and response. The packet list shows a request from 192.168.120.2 to 192.168.120.132. The packet details show the request structure with sender and target MAC and IP addresses. The packet bytes show the raw data.

No.	Time	Source	Destination	Protocol	Length	Info
15	13.450297009	Vmware_f8:32:b6	Broadcast	ARP	60	Who has 192.168.120.132? Tell 192.168.120.2
16	13.450333211	Vmware_2c:42:6f	Vmware_f8:32:b6	ARP	42	192.168.120.132 is at 00:0c:29:2c:42:6f
33	131.592484417	Vmware_2c:42:6f	Vmware_f8:32:b6	ARP	42	Who has 192.168.120.2? Tell 192.168.120.132
34	131.592706877	Vmware_f8:32:b6	Vmware_2c:42:6f	ARP	60	192.168.120.2 is at 00:50:56:f8:32:b6
56	313.668546696	Vmware_f8:32:b6	Broadcast	ARP	60	Who has 192.168.120.132? Tell 192.168.120.2
57	313.668580867	Vmware_2c:42:6f	Vmware_f8:32:b6	ARP	42	192.168.120.132 is at 00:0c:29:2c:42:6f
81	613.883581360	Vmware_f8:32:b6	Broadcast	ARP	60	Who has 192.168.120.132? Tell 192.168.120.2
82	613.883596593	Vmware_2c:42:6f	Vmware_f8:32:b6	ARP	42	192.168.120.132 is at 00:0c:29:2c:42:6f

Frame 15: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
Ethernet II, Src: Vmware_f8:32:b6 (00:50:56:f8:32:b6), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Address Resolution Protocol (request)
Hardware type: Ethernet (1)
Protocol type: IPv4 (0x0800)
Hardware size: 6
Protocol size: 4
Opcode: request (1)
Sender MAC address: Vmware_f8:32:b6 (00:50:56:f8:32:b6)
Sender IP address: 192.168.120.2
Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
Target IP address: 192.168.120.132

0000 ff ff ff ff ff ff 00 50 56 f8 32 b6 08 06 00 01P V.2....
0010 08 00 06 04 00 01 00 50 56 f8 32 b6 c0 a8 78 02P V.2...x.
0020 00 00 00 00 00 00 c0 a8 78 84 00 00 00 00 00 00x.....
0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
wireshark_ens33_20190429234242_Krus7e.pcapng Packets: 1030 · Displayed: 28 (2.7%) Profile: Default

MAC del origen: Vmware_f8:32:b6 (00:50:56:f8:32:b6)

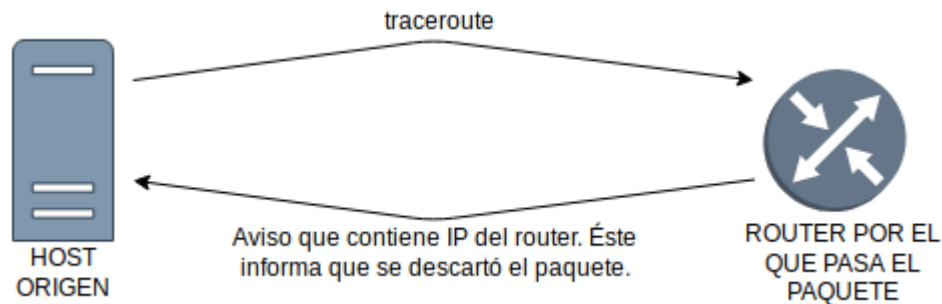
IP del destino: 192.168.120.132

MAC del destino: 00:00:00_00:00:00

El host con dirección IP 192.168.120.2 quiere enviar información. Éste sabe que debe enviar la información a 192.168.120.132, pero la dirección MAC de la ip consultada es el dato desconocido. Por lo tanto, este es el nuevo dato que se aprende en la respuesta.

CAPA DE RED

Traceroute crea paquetes y los envía al host correspondiente. El algoritmo de ruteo inspecciona el paquete que va a ser enviado. Si el paquete es igual a cero, se descarta el paquete y se da aviso al host de origen (para ello, se aplica el protocolo ICMP). De lo contrario, continúa el ruteo normalmente. El único parámetro que debe incluir cuando ejecuta el comando traceroute, es el nombre de host o la dirección IP del destino.



Ejecución de traceroute:

La función del protocolo ICMP es enviar mensajes de error e información operativa para que la fuente original pueda evitar o corregir el problema detectado. Puede indicar que un host no puede ser localizado o que un servicio que se ha solicitado no está disponible. Estos mensajes del protocolo ICMP se envían a la dirección IP de origen del paquete.

La captura en Wireshark:

No.	Time	Source	Destination	Protocol	Length	Info
60	6.170768651	192.168.1.1	192.168.1.33	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
62	6.171281232	192.168.1.1	192.168.1.33	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
63	6.171305752	192.168.1.1	192.168.1.33	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
70	6.174986462	181.25.255.255	192.168.1.33	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
71	6.176211130	181.25.255.255	192.168.1.33	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
72	6.176241538	181.25.255.255	192.168.1.33	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
77	6.183835087	200.51.208.213	192.168.1.33	ICMP	186	Time-to-live exceeded (Time to live exceeded in transit)
78	6.183874303	200.51.208.213	192.168.1.33	ICMP	186	Time-to-live exceeded (Time to live exceeded in transit)
79	6.183882672	200.51.208.213	192.168.1.33	ICMP	186	Time-to-live exceeded (Time to live exceeded in transit)
80	6.184890633	200.51.240.220	192.168.1.33	ICMP	186	Time-to-live exceeded (Time to live exceeded in transit)
82	6.188378523	200.51.208.65	192.168.1.33	ICMP	182	Time-to-live exceeded (Time to live exceeded in transit)
83	6.189907152	200.51.240.253	192.168.1.33	ICMP	186	Time-to-live exceeded (Time to live exceeded in transit)
84	6.190331109	200.3.34.1	192.168.1.33	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
101	6.212898665	181.88.108.242	192.168.1.33	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
102	6.213659466	200.3.34.1	192.168.1.33	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
103	6.213705922	200.3.34.1	192.168.1.33	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
105	6.215085149	200.3.34.133	192.168.1.33	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
106	6.215403169	200.3.34.133	192.168.1.33	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
107	6.216846493	181.96.22.222	192.168.1.33	ICMP	182	Time-to-live exceeded (Time to live exceeded in transit)
108	6.217631492	170.210.4.66	192.168.1.33	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
110	6.218244802	181.88.168.59	192.168.1.33	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
111	6.218267183	170.210.4.66	192.168.1.33	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
118	6.221334729	181.88.168.223	192.168.1.33	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
121	6.227548469	170.210.4.66	192.168.1.33	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)

Las direcciones IP que aparecen en la columna Source pertenecen a los routers que se encuentran en el trayecto del paquete.

Los mensajes tienen el campo info igual a "Time-to-live exceeded" ya que esta es la respuesta al equipo origen con la que el protocolo ICMP indica que un paquete fue descartado debido a que el router recibió un paquete con ttl igual a 0.

Datos de salida standard de traceroute:

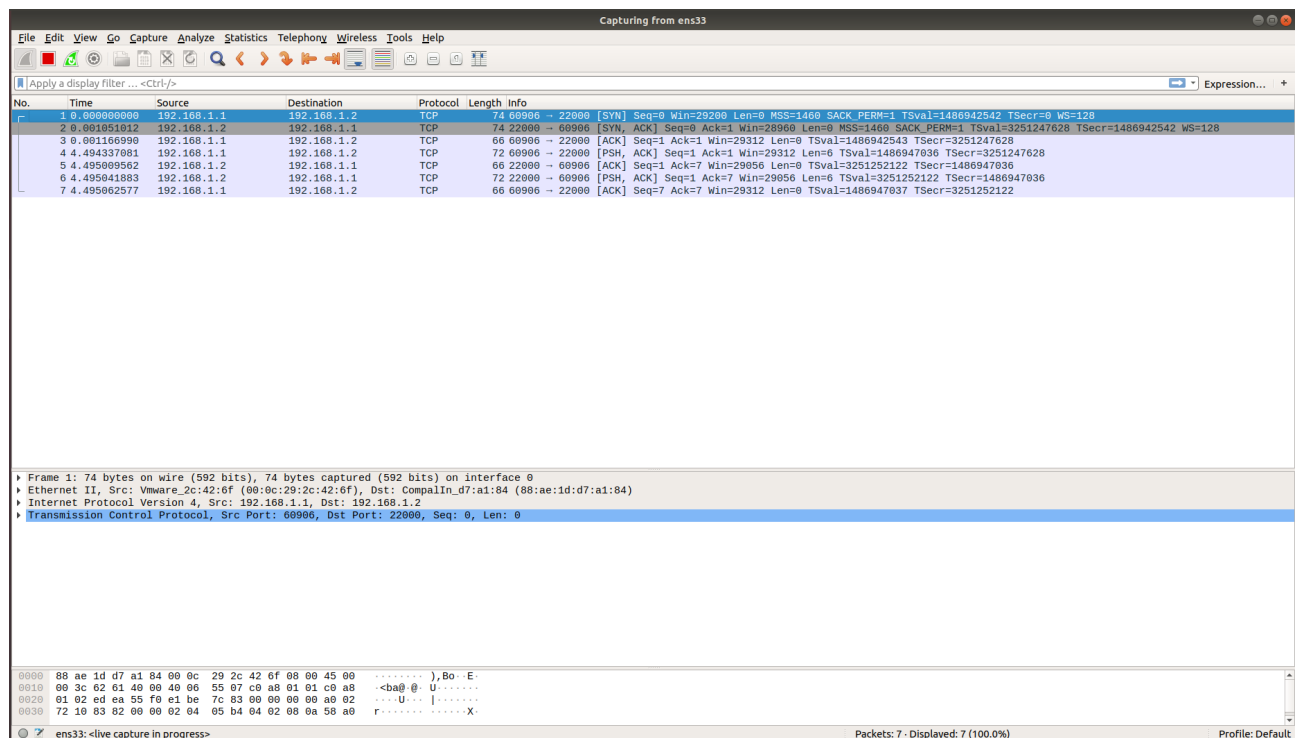
```
sol@ubuntu:~$ traceroute www.ungs.edu.ar
traceroute to www.ungs.edu.ar (170.210.52.25), 30 hops max, 60 byte packets
 1 _gateway (192.168.1.1)  2.592 ms  2.102 ms  2.877 ms
 2 181-25-255-255.speedy.com.ar (181.25.255.255)  5.819 ms  6.162 ms  5.824 ms
 3 * * *
 4 200.51.208.213 (200.51.208.213)  8.017 ms  11.020 ms  15.170 ms
 5 200.51.208.65 (200.51.208.65)  8.141 ms  7.825 ms  7.846 ms
 6 200.51.240.131 (200.51.240.131)  9.453 ms  11.506 ms  12.299 ms
 7 host133.200-3-34.telecom.net.ar (200.3.34.133)  14.148 ms  14.694 ms
host242.181-88-108.telecom.net.ar (181.88.108.242)  9.461 ms
 8 host209.190-225-249.telecom.net.ar (190.225.249.209)  10.373 ms * *
 9 host222.181-96-22.telecom.net.ar (181.96.22.222)  9.014 ms  7.891 ms  8.467 ms
10 host59.181-88-168.telecom.net.ar (181.88.168.59)  8.039 ms * *
```

Similitudes y diferencias:

Como similitud, en ambas salidas se muestra el ip de origen. En cuanto a las diferencias, la captura del wireshark muestra el ip de destino, el protocolo que interviene y la longitud del paquete, mientras que la salida del traceroute no lo hace. A su vez, la salida del traceroute muestra el nombre de dominio que interviene y la captura de wireshark no.

CAPA DE TRANSPORTE

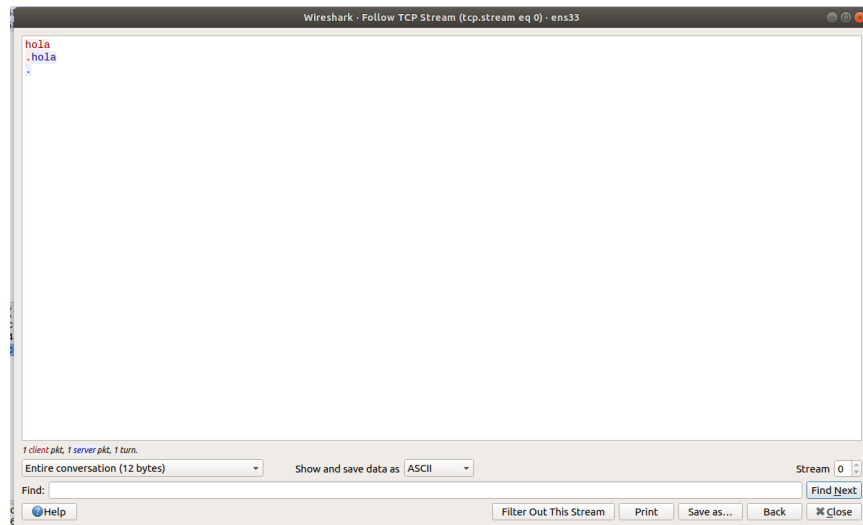
Escuando sobre la interfaz 192.168.1.1, sobre el cliente:



En el intercambio de mensajes, que se conoce como "establecer la conexión", se aclaran los siguientes datos:

- **SYN**: es un tipo de mensaje que significa "vamos a sincronizarnos"
 - **Win**: es el tamaño del buffer de quién está enviando el mensaje, es decir, la cantidad máxima que se está dispuesto a recibir.
 - **Ack**: es un tipo de mensaje que significa acuse de recibo. Pone un número que está relacionado al número de secuencia que recibió anteriormente indicando cuál es el número de secuencia que espera recibir a continuación.
 - **seq**: es el número de secuencia, identifica al índice de byte que se está enviando.
- En un envío típico bajo el protocolo TCP, el mensaje ACK Y el número de secuencia son los datos que garantizan el servicio confiable.

Seguimiento del texto plano:



Para que no se pueda leer o entender la captura de wireshark, se debería implementar algún método de cifrado del mensaje en texto plano desde el lado del cliente y un método para descifrar el mensaje recibido desde el lado del servidor.

	Número de IP	Número de puerto
Socket Cliente	192.168.1.1	60906
Socket Servidor	192.168.1.2	22000

CAPA DE APLICACIÓN

SSH y SCP:

Login remoto con ssh:

```
mariasol@ubuntu:~$ ssh sol@192.168.1.1
sol@192.168.1.1's password:
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-51-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

168 packages can be updated.
16 updates are security updates.

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your
Internet connection or proxy settings

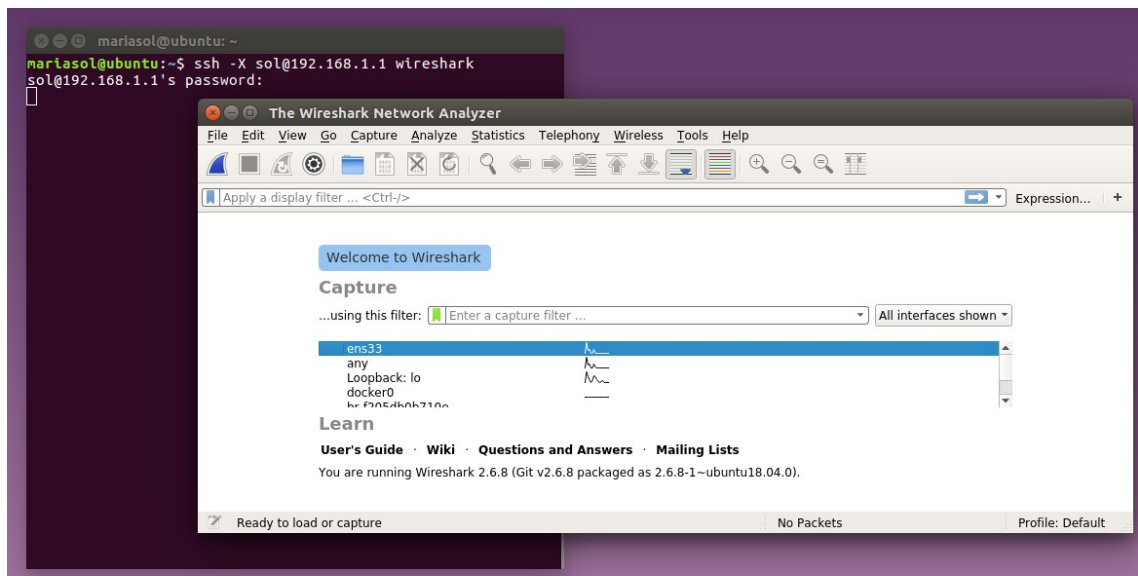
Last login: Thu Jun 13 00:24:08 2019 from 192.168.1.2
sol@ubuntu:~$ cd /home/sol
sol@ubuntu:~$ ls
Desktop  Downloads  Music      Public     TPISO      Videos
Documents  examples.desktop  Pictures   Templates  TP2REDES
```

SSH y Telnet son protocolos que sirven para conectarse remotamente a un servidor. La diferencia entre ellos es que con SSH utilizamos una conexión segura hacia nuestro servidor, con lo que la conexión viaja encriptada. En cambio, utilizando Telnet se expone toda la información que intercambiamos con el servidor. Es por esto que SSH ha sustituido al protocolo Telnet.

Puedo ejecutar programas de interfaz gráfica, como wireshark, con el siguiente comando:

```
$ ssh -X sol@192.168.1.1 wireshark
```

Con el parámetro -X indico que voy a ejecutar un programa con interfaz gráfica, luego indico el usuario y la ip del servidor:



Realicé la tranferencia de un archivo con SCP utilizando el siguiente comando:

```
$ scp sol@192.168.1.1:/home/sol/Documents/compartir /home/mariasol/Documents
```

Se transfirió el archivo "compartir" desde el host servidor hacia el host cliente:

```
mariasol@ubuntu:~/Documents$ ls
mariasol@ubuntu:~/Documents$ scp sol@192.168.1.1:/home/sol/Documents/compartir /home/mariasol/Documents
sol@192.168.1.1's password:
compartir
mariasol@ubuntu:~/Documents$ ls
compartir
mariasol@ubuntu:~/Documents$
```

SCP es un medio de transferencia segura de archivos entre un host local y otro remoto o entre dos hosts remotos, usando el protocolo SSH.

SSH y firewall con iptables:

Con el comando `$ sudo iptables -A INPUT -s 192.168.1.2 -j DROP` agrego una restricción de acceso para el host 192.168.1.2 y éste ya no puede conectarse:

```
mariasol@ubuntu: ~/Documents
mariasol@ubuntu:~/Documents$ ssh sol@192.168.1.1
```

Con el comando `$ sudo iptables -D INPUT -s 192.168.1.2 -j DROP` remuevo la restricción y ya puede volver a realizarse la conexión:

```
sol@ubuntu: ~  
marisol@ubuntu:~/Documents$ ssh sol@192.168.1.1  
^C  
marisol@ubuntu:~/Documents$ ssh sol@192.168.1.1  
sol@192.168.1.1's password:  
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-51-generic x86_64)  
  
 * Documentation:  https://help.ubuntu.com  
 * Management:    https://landscape.canonical.com  
 * Support:       https://ubuntu.com/advantage  
  
 * Canonical Livepatch is available for installation.  
   - Reduce system reboots and improve kernel security. Activate at:  
     https://ubuntu.com/livepatch  
  
168 packages can be updated.  
16 updates are security updates.  
  
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings  
  
Last login: Thu Jun 13 01:05:50 2019 from 192.168.1.2  
sol@ubuntu:~$
```

Con el comando `$ sudo iptables -L` verifico las reglas presentes:

```
sol@ubuntu:~$ sudo iptables -A INPUT -s 192.168.1.2 -j DROP
sol@ubuntu:~$ sudo iptables -D INPUT -s 192.168.1.2 -j DROP
sol@ubuntu:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source      destination
all -- -- 10.10.10.22 anywhere
all -- -- 10.10.10.22 anywhere

Chain FORWARD (policy DROP)
target     prot opt source      destination
DOCKER-USER all -- -- anywhere    anywhere
DOCKER-ISOLATION-STAGE-1 all -- -- anywhere    anywhere
ACCEPT all -- -- anywhere    anywhere
TABLISHED ctstate RELATED,ESTABLISHED
DOCKER all -- -- anywhere    anywhere
ACCEPT all -- -- anywhere    anywhere
ACCEPT all -- -- anywhere    anywhere
ACCEPT all -- -- anywhere    anywhere
TABLISHED ctstate RELATED,ESTABLISHED
DOCKER all -- -- anywhere    anywhere
ACCEPT all -- -- anywhere    anywhere
ACCEPT all -- -- anywhere    anywhere
ACCEPT all -- -- anywhere    anywhere
TABLISHED ctstate RELATED,ESTABLISHED
DOCKER all -- -- anywhere    anywhere
ACCEPT all -- -- anywhere    anywhere
ACCEPT all -- -- anywhere    anywhere
```

```
Chain OUTPUT (policy ACCEPT)
target      prot opt source                               destination

Chain DOCKER (3 references)
target      prot opt source                               destination

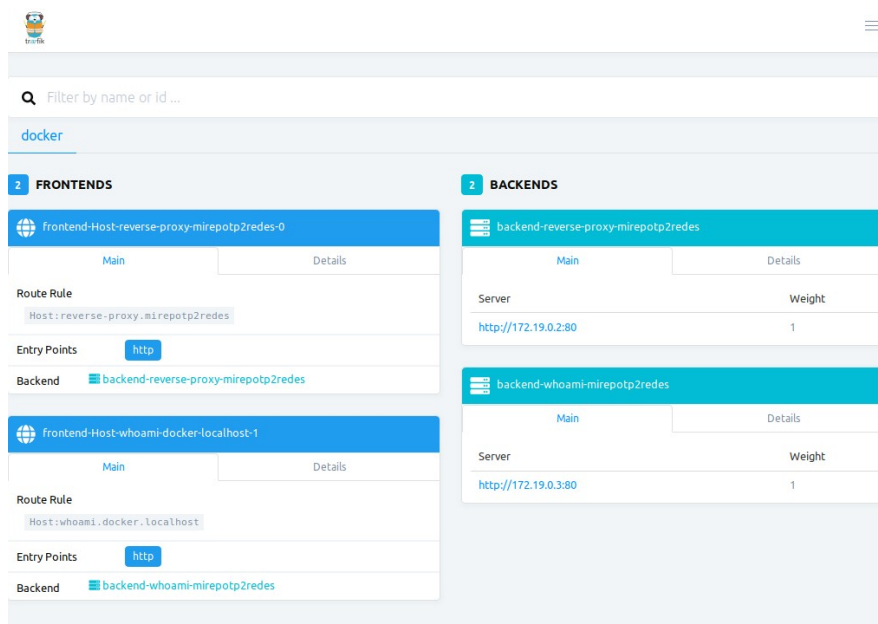
Chain DOCKER-ISOLATION-STAGE-1 (1 references)
target      prot opt source                               destination
DOCKER-ISOLATION-STAGE-2 all -- anywhere                          anywhere
DOCKER-ISOLATION-STAGE-2 all -- anywhere                          anywhere
DOCKER-ISOLATION-STAGE-2 all -- anywhere                          anywhere
RETURN      all -- anywhere                          anywhere

Chain DOCKER-ISOLATION-STAGE-2 (3 references)
target      prot opt source                               destination
DROP        all -- anywhere                          anywhere
DROP        all -- anywhere                          anywhere
DROP        all -- anywhere                          anywhere
RETURN      all -- anywhere                          anywhere

Chain DOCKER-USER (1 references)
target      prot opt source                               destination
RETURN      all -- anywhere                          anywhere
```


HTTP y Proxy:

Replica de la imagen del Docker:



Respuesta de la consola:

```
sol@ubuntu:~/TP2REDES/MiRepoTP2REDES$ curl -H Host:whoami.docker.localhost http://127.0.0.1
Hostname: 853c9b2ac80c
IP: 127.0.0.1
IP: 172.19.0.3
GET / HTTP/1.1
Host: whoami.docker.localhost
User-Agent: curl/7.58.0
Accept: */*
Accept-Encoding: gzip
X-Forwarded-For: 172.19.0.1
X-Forwarded-Host: whoami.docker.localhost
X-Forwarded-Port: 80
X-Forwarded-Proto: http
X-Forwarded-Server: a09f478e3831
X-Real-Ip: 172.19.0.1

sol@ubuntu:~/TP2REDES/MiRepoTP2REDES$ curl -H Host:whoami.docker.localhost http://127.0.0.1
Hostname: 853c9b2ac80c
IP: 127.0.0.1
IP: 172.19.0.3
GET / HTTP/1.1
Host: whoami.docker.localhost
User-Agent: curl/7.58.0
Accept: */*
Accept-Encoding: gzip
X-Forwarded-For: 172.19.0.1
X-Forwarded-Host: whoami.docker.localhost
X-Forwarded-Port: 80
X-Forwarded-Proto: http
X-Forwarded-Server: a09f478e3831
X-Real-Ip: 172.19.0.1

sol@ubuntu:~/TP2REDES/MiRepoTP2REDES$ curl -H Host:whoami.docker.localhost http://127.0.0.1
Hostname: 853c9b2ac80c
IP: 127.0.0.1
IP: 172.19.0.3
GET / HTTP/1.1
Host: whoami.docker.localhost
User-Agent: curl/7.58.0
Accept: */*
Accept-Encoding: gzip
X-Forwarded-For: 172.19.0.1
X-Forwarded-Host: whoami.docker.localhost
X-Forwarded-Port: 80
X-Forwarded-Proto: http
X-Forwarded-Server: a09f478e3831
X-Real-Ip: 172.19.0.1
```