

Trabajo Práctico

TPL 2 - Aplicaciones 1: Cliente/Servidor – Telnet

Fecha de entrega: 22/04/2020

Franco, Juan Martín 149.615

juanmartin_franco@hotmail.com

Primer parte: Creación de un modelo simple Cliente/Servidor

Para dar comienzo con la resolución del TP N° 2, primero es necesario configurar el laboratorio como lo hicimos en el TP N° 1.

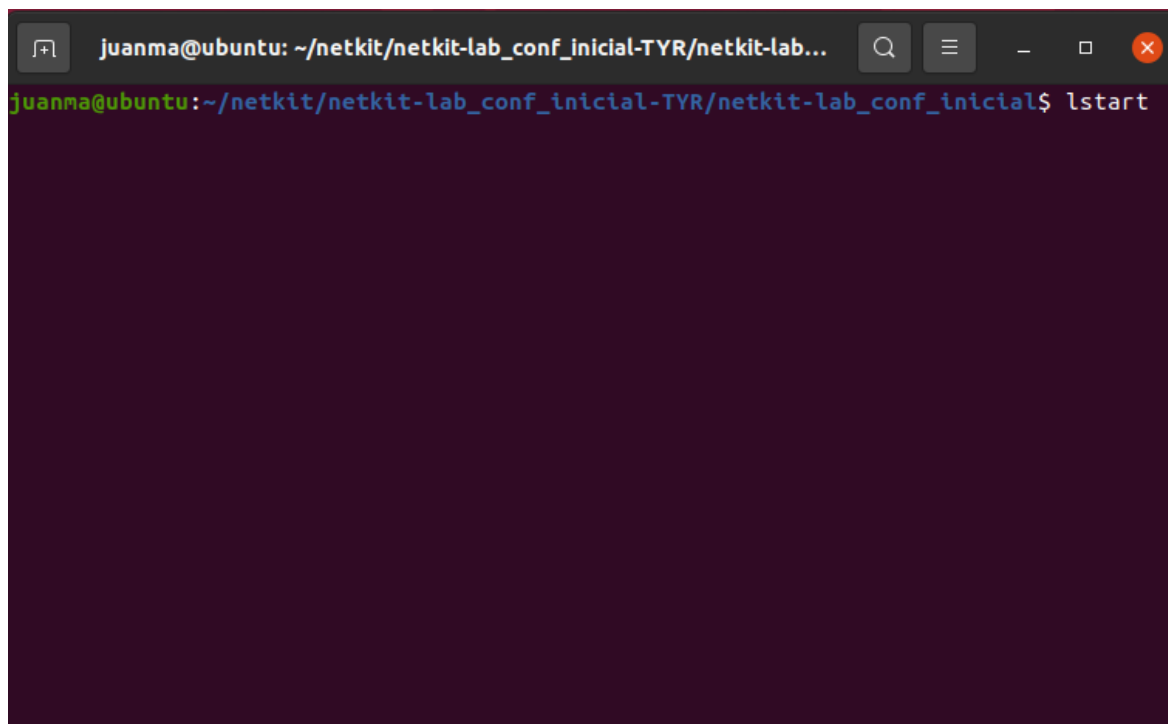
Para esto, primero debo iniciar la máquina virtual.

Una vez iniciada, procedo a configurar todo como en el TP N° 1.

Primero, debo iniciar el laboratorio inicial, posicionándome en la carpeta contenedora del mismo y utilizando el comando

```
lstart
```

(todo esto desde la terminal)

A screenshot of a terminal window. The title bar shows the user 'juanma@ubuntu' and the current directory '~/netkit/netkit-lab_conf_inicial-TYR/netkit-lab...'. The terminal prompt is 'juanma@ubuntu:~/netkit/netkit-lab_conf_inicial-TYR/netkit-lab_conf_inicial\$'. The command 'lstart' has been entered and executed, as indicated by the green prompt color. The terminal background is dark purple.

Una vez ejecutado el comando, se iniciará el laboratorio de pruebas, el cual como ya sabemos, incluye a pc1, pc2 y r1.

Cuando finaliza la ejecución del mismo obtendremos lo siguiente:

```
#####
Lab directory (host): /home/juanma/netkit/netkit-lab_conf_inicial-TYR/netkit-lab
conf_inicial
Version: 1
Author: F. Lorge, M. Meloni
Email: [florge]@meloni@unlu.edu.ar
Web: http://www.labredes.unlu.edu.ar/ https://github.com/redesunlu/netkit-lab
Description:
Configuraci3n inicial del laboratorio
#####
--- Netkit phase 2 initialization terminated ---
[ ok ] Starting enhanced syslogd: rsyslogd.
tyr11 login: root (automatic login)
Last login: Fri Apr 9 23:40:12 ART 2016 on tty0
Linux tyr11 3.2.86-netkit-ng-K3.2 #2 Wed Mar 29 09:08:55 ART 2017 i686
Welcome to Netkit
root@tyr11:~#

#####
Lab directory (host): /home/juanma/netkit/netkit-lab_conf_inicial-TYR/netkit-lab
conf_inicial
Version: 1
Author: F. Lorge, M. Meloni
Email: [florge]@meloni@unlu.edu.ar
Web: http://www.labredes.unlu.edu.ar/ https://github.com/redesunlu/netkit-lab
Description:
Configuraci3n inicial del laboratorio
#####
--- Netkit phase 2 initialization terminated ---
[ ok ] Starting enhanced syslogd: rsyslogd.
r1 login: root (automatic login)
Last login: Fri Apr 9 23:41:22 ART 2021 on tty0
Linux r1 3.2.86-netkit-ng-K3.2 #2 Wed Mar 29 09:08:55 ART 2017 i686
Welcome to Netkit
root@r1:~#
```

Ahora, debo activar las interfaces utilizando el comando

`ip link set dev INTERFAZ up`

Siendo INTERFAZ el nombre de la interfaz a activar.

En este caso, al querer activar la interfaz `eth0`, debo ejecutar el comando:

`ip link set eth0 up`

```
pc1
root@tyr11:~# ip link set dev eth0 up
root@tyr11:~# ip a s
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOW
    link/ether 2e:44:cd:f9:b9:e7 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::2c44:cdff:fe9:b9e7/64 scope link
        valid_lft forever preferred_lft forever
root@tyr11:~#
```

Luego, repito el mismo proceso para las máquinas restantes del laboratorio.

```
pc2
root@tyr12:~# ip link set dev eth0 up
root@tyr12:~# ip a s
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOW
    link/ether e6:c6:b2:40:fc:8e brd ff:ff:ff:ff:ff:ff
    inet6 fe80::e4c6:b2ff:fe40:fc8e/64 scope link
        valid_lft forever preferred_lft forever
root@tyr12:~#
```

```
r1
root@r1:~# ip a s
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOW
    link/ether 76:6a:7e:26:c9:22 brd ff:ff:ff:ff:ff:ff
    inet 10.4.11.30/24 scope global eth0
    inet6 fe80::746a:7eff:fe26:c922/64 scope link
        valid_lft forever preferred_lft forever
root@r1:~#
```

(r1 viene con la interfaz eth0 activada por defecto)

Una vez activadas las interfaces nombradas anteriormente, procedo a asignar la dirección ip que le había asignado en el TP N° 1.

Esto se hace mediante el comando:

```
ip address add DIRECCIÓN_IP/PREFIJO_MÁSCARA dev INTERFAZ
```

En este caso, ejecuto:

```
ip address add 10.4.11.11/24 dev eth0 (pc1)
```

```
pc1
root@tyr11:~# ip address add 10.4.11.11/24 dev eth0
root@tyr11:~# ip a s
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOW
    link/ether 2e:44:cd:f9:b9:e7 brd ff:ff:ff:ff:ff:ff
    inet 10.4.11.11/24 scope global eth0
    inet6 fe80::2c44:cdff:fe9:b9e7/64 scope link
        valid_lft forever preferred_lft forever
root@tyr11:~#
```

```
ip address add 10.4.11.12/24 dev eth0 (pc2)
```

```
pc2
root@tyr12:~# ip address add 10.4.11.12/24 dev eth0
root@tyr12:~# ip a s
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOW
    link/ether e6:c6:b2:40:fc:8e brd ff:ff:ff:ff:ff:ff
    inet 10.4.11.12/24 scope global eth0
    inet6 fe80::e4c6:b2ff:fe40:fc8e/64 scope link
        valid_lft forever preferred_lft forever
root@tyr12:~#
```

Por último, debo probar que puedo contactarme con los demás dispositivos de la red, tanto por su dirección ip como por su nombre de host.

Para esto, utilizo los siguientes comandos:

```
ping 10.4.11.12
```

o

```
ping tyr12
```

(ambos son equivalentes, los 2 los ejecuto desde pc1)

```
pc1
root@tyr11:~# ping 10.4.11.12
PING 10.4.11.12 (10.4.11.12) 56(84) bytes of data.
64 bytes from 10.4.11.12: icmp_req=1 ttl=64 time=1.03 ms
64 bytes from 10.4.11.12: icmp_req=2 ttl=64 time=0.715 ms
64 bytes from 10.4.11.12: icmp_req=3 ttl=64 time=1.11 ms
^C
--- 10.4.11.12 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2006ms
rtt min/avg/max/mdev = 0.715/0.956/1.119/0.175 ms
root@tyr11:~# ping tyr12
PING tyr12 (10.4.11.12) 56(84) bytes of data.
64 bytes from tyr12 (10.4.11.12): icmp_req=1 ttl=64 time=0.457 ms
64 bytes from tyr12 (10.4.11.12): icmp_req=2 ttl=64 time=0.960 ms
64 bytes from tyr12 (10.4.11.12): icmp_req=3 ttl=64 time=1.07 ms
^C
--- tyr12 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2012ms
rtt min/avg/max/mdev = 0.457/0.830/1.074/0.269 ms
root@tyr11:~#
```

Y repito el mismo proceso pero utilizando la ip y el nombre de host de la pc1 desde pc2:

```
pc2
root@tyr12:~# ping 10.4.11.11
PING 10.4.11.11 (10.4.11.11) 56(84) bytes of data.
64 bytes from 10.4.11.11: icmp_req=1 ttl=64 time=0.750 ms
64 bytes from 10.4.11.11: icmp_req=2 ttl=64 time=1.34 ms
64 bytes from 10.4.11.11: icmp_req=3 ttl=64 time=0.921 ms
^C
--- 10.4.11.11 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2015ms
rtt min/avg/max/mdev = 0.750/1.006/1.349/0.254 ms
root@tyr12:~# ping tyr11
PING tyr11 (10.4.11.11) 56(84) bytes of data.
64 bytes from tyr11 (10.4.11.11): icmp_req=1 ttl=64 time=0.654 ms
64 bytes from tyr11 (10.4.11.11): icmp_req=2 ttl=64 time=1.20 ms
64 bytes from tyr11 (10.4.11.11): icmp_req=3 ttl=64 time=0.865 ms
^C
--- tyr11 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2021ms
rtt min/avg/max/mdev = 0.654/0.907/1.203/0.227 ms
root@tyr12:~#
```

Lo cual, me permite sacar la conclusión de que fue posible contactar ambos equipos en la red.

Aclaración: No hizo falta configurar el archivo hosts ubicado en /etc/hosts ya que, como bien se detalla en el TP N° 1, el cambio realizado es permanente, por lo que se mantiene la configuración realizada.

Defina un número de puerto para el proceso servidor (superior a 1024).

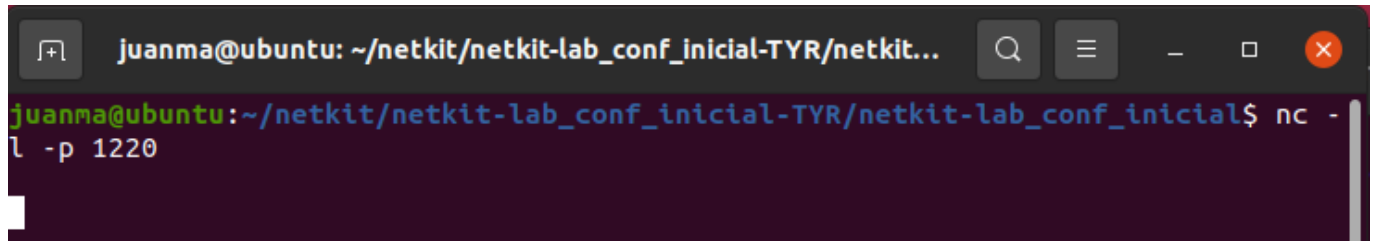
En este caso, para el proceso servidor opté por elegir el puerto 1220, el cual es un puerto registrado y puede ser utilizado por cualquier aplicación de usuario.

Para hacer esto, utilizo el comando:

```
nc -l [PuertoElegido]
```

Por ejemplo:

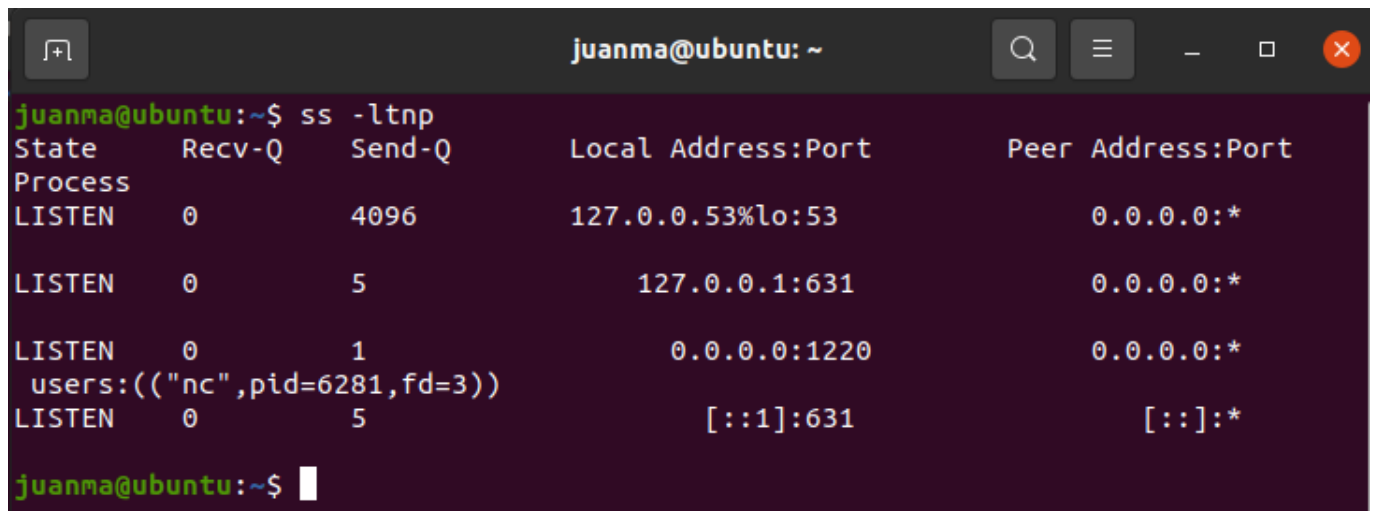
```
nc -l -p 1220
```



```
juanma@ubuntu: ~/netkit/netkit-lab_conf_inicial-TYR/netkit...
juanma@ubuntu:~/netkit/netkit-lab_conf_inicial-TYR/netkit-lab_conf_inicial$ nc -l -p 1220
```

Y luego, verifico que efectivamente el puerto está en escucha con el comando:

```
ss -ltnp
```



```
juanma@ubuntu: ~
juanma@ubuntu:~$ ss -ltnp
State      Recv-Q    Send-Q    Local Address:Port    Peer Address:Port
Process
LISTEN     0         4096      127.0.0.53%lo:53      0.0.0.0:*
LISTEN     0         5        127.0.0.1:631        0.0.0.0:*
LISTEN     0         1        0.0.0.0:1220        0.0.0.0:*
users:((("nc",pid=6281,fd=3))
LISTEN     0         5        [::1]:631          [::]:*
```

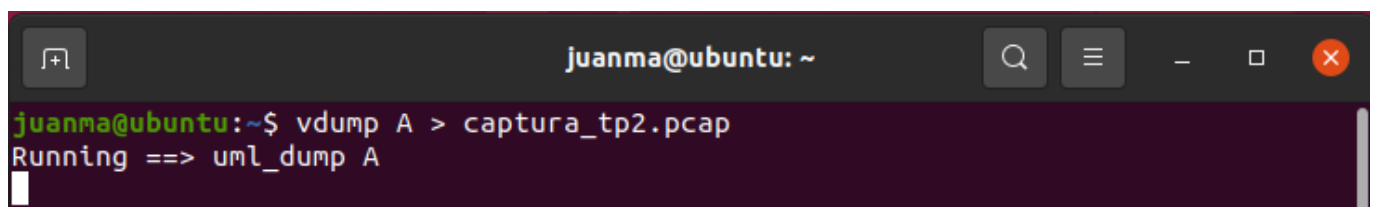
Realice una captura de todo el proceso utilizando la herramienta vdump, guardándola en un archivo en formato pcap para su posterior análisis.

Para realizar una captura utilizando vdump, ejecuto el comando:

```
vdump A > nombre_archivo.pcap
```

Por ejemplo:

```
vdump A > captura_tp2.pcap
```



```
juanma@ubuntu: ~
juanma@ubuntu:~$ vdump A > captura_tp2.pcap
Running ==> uml_dump A
```

Luego de ejecutar esto, vdump comenzará a capturar el tráfico.

En el host **pc1** deberá ejecutar la utilidad **nc** actuando como servidor, indicando como parámetro el número de puerto elegido. Una vez iniciado, este servicio quedará en modo de *escucha o listening*.

En el otro host (**pc2**) ejecute la utilidad **nc** como cliente indicando como parámetros la IP del servidor y número de puerto.

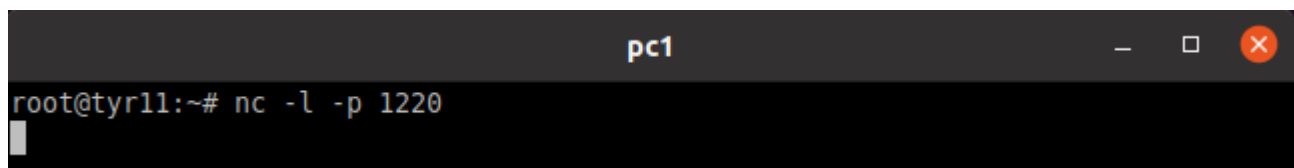
Como bien lo indica el enunciado, debo ejecutar dentro de pc1 la utilidad nc para que este cumpla el rol de servidor.

Para hacer esto, utilizo el comando:

```
nc -l -p <NRO_DE_PUERTO>
```

En este caso:

```
nc -l -p 1220
```

A terminal window titled 'pc1' with standard window controls. The prompt is 'root@tyr11:~#'. The command 'nc -l -p 1220' has been entered and is being executed, as indicated by a cursor on the line below.

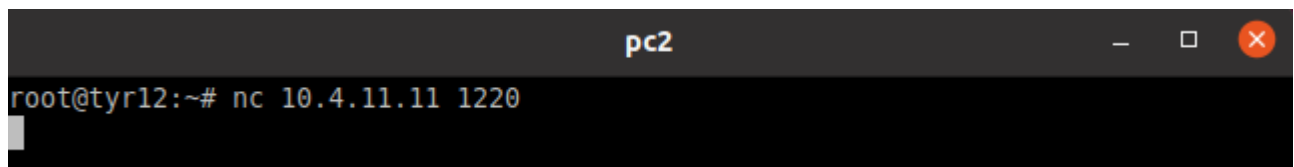
Ahora, debo ejecutar la utilidad netcat como cliente dentro de pc2.

Para hacer esto, utilizo el comando:

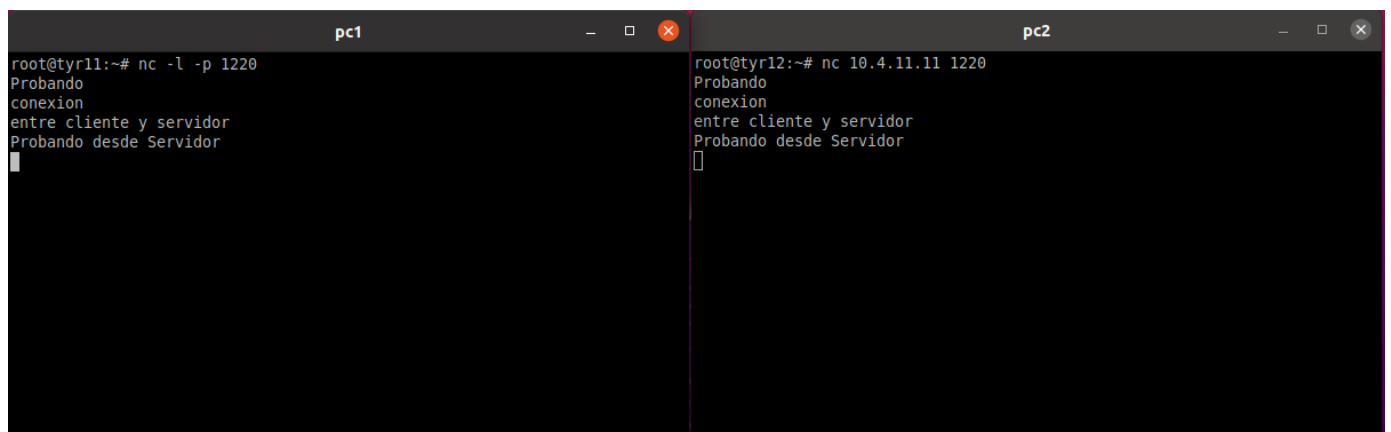
```
nc <IP_HOST_SERVIDOR> <NRO_DE_PUERTO_SERVIDOR>
```

En este caso:

```
nc 10.4.11.11 1220
```

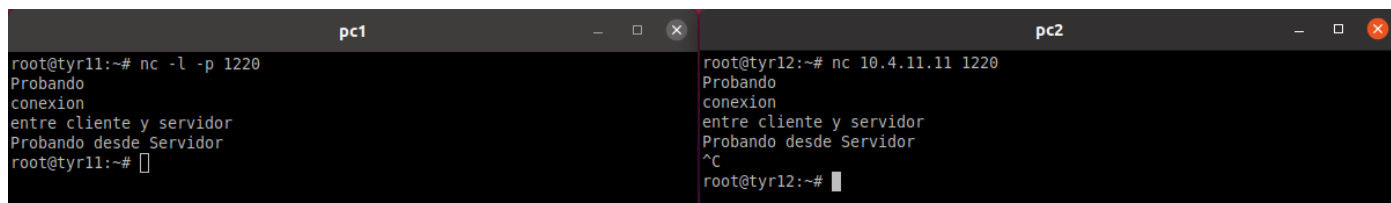
A terminal window titled 'pc2' with standard window controls. The prompt is 'root@tyr12:~#'. The command 'nc 10.4.11.11 1220' has been entered and is being executed, as indicated by a cursor on the line below.

Una vez ejecutados ambos comandos, se establece la conexión y se ve una especie de chat en el cual puedo intercambiar mensajes.

Two terminal windows are shown side-by-side. The left window is titled 'pc1' and the right is titled 'pc2'. Both show the same sequence of messages: 'root@tyr11:~# nc -l -p 1220' followed by 'Probando conexion entre cliente y servidor' and 'Probando desde Servidor'. The right window also shows the command 'root@tyr12:~# nc 10.4.11.11 1220' followed by the same messages. This indicates a successful connection has been established between the two hosts.

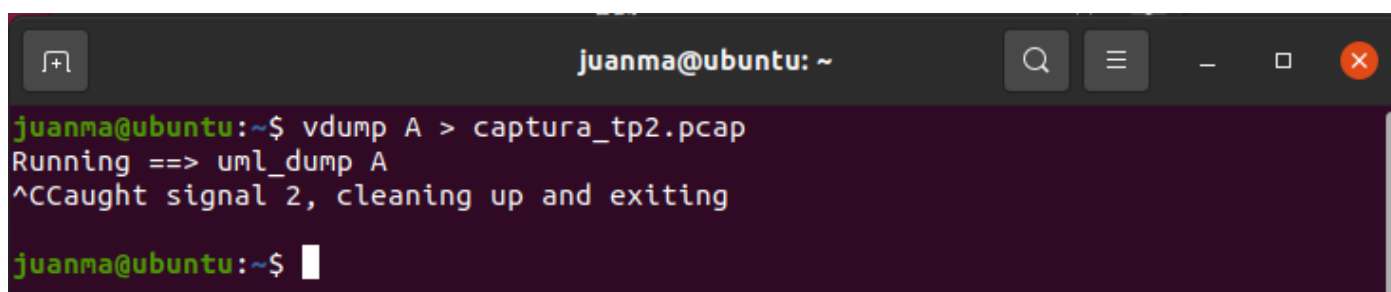
En la imagen no se puede apreciar, pero los primeros 3 mensajes fueron enviados desde pc2 (cliente) y el restante fue enviado desde pc1 (servidor), corroborando que efectivamente el chat funciona de manera correcta y bidireccionalmente.

Ahora, debo proceder a finalizar la conexión utilizando CTRL + C en cualquiera de las 2 máquinas virtuales.



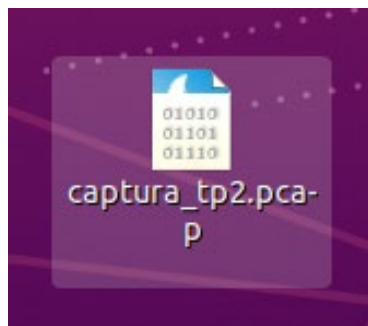
```
pc1                                     pc2
root@tyr11:~# nc -l -p 1220             root@tyr12:~# nc 10.4.11.11 1220
Probando                               Probando
conexion                               conexion
entre cliente y servidor               entre cliente y servidor
Probando desde Servidor               Probando desde Servidor
root@tyr11:~#                          ^C
root@tyr12:~#
```

Y luego, detengo la captura (fuera del laboratorio) también con CTRL + C.



```
juanma@ubuntu: ~
juanma@ubuntu:~$ vdump A > captura_tp2.pcap
Running ==> uml_dump A
^C Caught signal 2, cleaning up and exiting
juanma@ubuntu:~$
```

Ahora, ya finalizada la conexión y la captura, tengo el archivo `captura_tp2.pcap`.



Analice la captura almacenada en el archivo utilizando tshark y diversos parámetros de visualización (consulte la guía de comandos provista por la materia).

Para analizar la captura utilizando tshark, debo utilizar el comando provisto por la hoja de comandos, el cual es:

```
tshark -r captura_tp2.pcap
```

Obteniendo el siguiente resultado:


```

juanma@ubuntu:~/Desktop$ tshark -r captura_tp2.pcap
 1  0.000000 2e:49:77:10:28:47 → Broadcast  ARP 42 Who has 10.4.11.11? Tell 10.4.11.12
 2  0.000241 96:a5:ea:86:f8:c6 → 2e:49:77:10:28:47 ARP 42 10.4.11.11 is at 96:a5:ea:86:f8:c6
 3  0.000533 10.4.11.12 → 10.4.11.11  TCP 74 42754 → 1220 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=109176 TSecr=0 WS=4
 4  0.000555 10.4.11.11 → 10.4.11.12  TCP 74 1220 → 42754 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=113540 TSecr=109176 WS=4
 5  0.000560 10.4.11.12 → 10.4.11.11  TCP 66 42754 → 1220 [ACK] Seq=1 Ack=1 Win=14600 Len=0 TSval=109176 TSecr=113540
 6  5.016933 96:a5:ea:86:f8:c6 → 2e:49:77:10:28:47 ARP 42 Who has 10.4.11.12? Tell 10.4.11.11
 7  5.017089 2e:49:77:10:28:47 → 96:a5:ea:86:f8:c6 ARP 42 10.4.11.12 is at 2e:49:77:10:28:47
 8  58.190722 10.4.11.12 → 10.4.11.11  TCP 75 42754 → 1220 [PSH, ACK] Seq=1 Ack=1 Win=14600 Len=9 TSval=115002 TSecr=113540
 9  58.191495 10.4.11.11 → 10.4.11.12  TCP 66 1220 → 42754 [ACK] Seq=1 Ack=10 Win=14480 Len=0 TSval=119359 TSecr=115002
10  63.195593 2e:49:77:10:28:47 → 96:a5:ea:86:f8:c6 ARP 42 Who has 10.4.11.11? Tell 10.4.11.12
11  63.196195 96:a5:ea:86:f8:c6 → 2e:49:77:10:28:47 ARP 42 10.4.11.11 is at 96:a5:ea:86:f8:c6
12  69.517583 10.4.11.12 → 10.4.11.11  TCP 75 42754 → 1220 [PSH, ACK] Seq=10 Ack=1 Win=14600 Len=9 TSval=116135 TSecr=119359
13  69.517943 10.4.11.11 → 10.4.11.12  TCP 66 1220 → 42754 [ACK] Seq=1 Ack=19 Win=14480 Len=0 TSval=120491 TSecr=116135
14  74.520553 96:a5:ea:86:f8:c6 → 2e:49:77:10:28:47 ARP 42 Who has 10.4.11.12? Tell 10.4.11.11
15  74.520604 2e:49:77:10:28:47 → 96:a5:ea:86:f8:c6 ARP 42 10.4.11.12 is at 2e:49:77:10:28:47
16  75.480226 10.4.11.12 → 10.4.11.11  TCP 91 42754 → 1220 [PSH, ACK] Seq=19 Ack=1 Win=14600 Len=25 TSval=116731 TSecr=120491
17  75.480737 10.4.11.11 → 10.4.11.12  TCP 66 1220 → 42754 [ACK] Seq=1 Ack=44 Win=14480 Len=0 TSval=121088 TSecr=116731
18  171.930121 10.4.11.11 → 10.4.11.12  TCP 90 1220 → 42754 [PSH, ACK] Seq=1 Ack=44 Win=14480 Len=24 TSval=130733 TSecr=116731
19  171.930197 10.4.11.12 → 10.4.11.11  TCP 66 42754 → 1220 [ACK] Seq=44 Ack=25 Win=14600 Len=0 TSval=126376 TSecr=130733
20  176.9421974 2e:49:77:10:28:47 → 96:a5:ea:86:f8:c6 ARP 42 Who has 10.4.11.11? Tell 10.4.11.12
21  176.942020 96:a5:ea:86:f8:c6 → 2e:49:77:10:28:47 ARP 42 10.4.11.11 is at 96:a5:ea:86:f8:c6
22  355.110332 10.4.11.12 → 10.4.11.11  TCP 66 42754 → 1220 [FIN, ACK] Seq=44 Ack=25 Win=14600 Len=0 TSval=144694 TSecr=130733
23  355.113091 10.4.11.11 → 10.4.11.12  TCP 66 1220 → 42754 [FIN, ACK] Seq=25 Ack=45 Win=14480 Len=0 TSval=149051 TSecr=144694
24  355.113243 10.4.11.12 → 10.4.11.11  TCP 66 42754 → 1220 [ACK] Seq=45 Ack=26 Win=14600 Len=0 TSval=144695 TSecr=149051
25  360.131625 96:a5:ea:86:f8:c6 → 2e:49:77:10:28:47 ARP 42 Who has 10.4.11.12? Tell 10.4.11.11
26  360.144490 2e:49:77:10:28:47 → 96:a5:ea:86:f8:c6 ARP 42 Who has 10.4.11.11? Tell 10.4.11.12
27  360.158642 2e:49:77:10:28:47 → 96:a5:ea:86:f8:c6 ARP 42 10.4.11.12 is at 2e:49:77:10:28:47
28  360.159018 96:a5:ea:86:f8:c6 → 2e:49:77:10:28:47 ARP 42 10.4.11.11 is at 96:a5:ea:86:f8:c6
juanma@ubuntu:~/Desktop$

```

Otra forma de analizar la captura es utilizando el comando:

```
tshark -r captura_tp2.pcap -nV
```

El cual detalla más a fondo cada frame.

```

juanma@ubuntu:~/Desktop$ tshark -r captura_tp2.pcap -nV
Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits)
  Encapsulation type: Ethernet (1)
    Arrival Time: Apr 20, 2021 19:44:31.651617000 PDT
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1618973071.651617000 seconds
    [Time delta from previous captured frame: 0.000000000 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 0.000000000 seconds]
    Frame Number: 1
    Frame Length: 42 bytes (336 bits)
    Capture Length: 42 bytes (336 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:arp]
  Ethernet II, Src: 2e:49:77:10:28:47, Dst: ff:ff:ff:ff:ff:ff
    Destination: ff:ff:ff:ff:ff:ff
      Address: ff:ff:ff:ff:ff:ff
        .... ..1. .... = LG bit: Locally administered address (this is NOT the factory default)
        .... ...1 .... = IG bit: Group address (multicast/broadcast)
    Source: 2e:49:77:10:28:47
      Address: 2e:49:77:10:28:47
        .... ..1. .... = LG bit: Locally administered address (this is NOT the factory default)
        .... ...0 .... = IG bit: Individual address (unicast)
    Type: ARP (0x0806)
  Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: 2e:49:77:10:28:47
    Sender IP address: 10.4.11.12

```

a) “Extraiga” de la captura solamente los datos intercambiados a nivel aplicación y remítalos.

Para extraer, dentro de la captura, los datos referentes a nivel aplicación, debo utilizar el siguiente comando:

```
tshark -r NOMBRE_ARCHIVO.PCAP -nqz follow,tcp,hex,0
```


En este caso:

```
tshark -r captura_tp2.pcap -nqz follow,tcp,hex,0
```

Obteniendo el siguiente resultado:

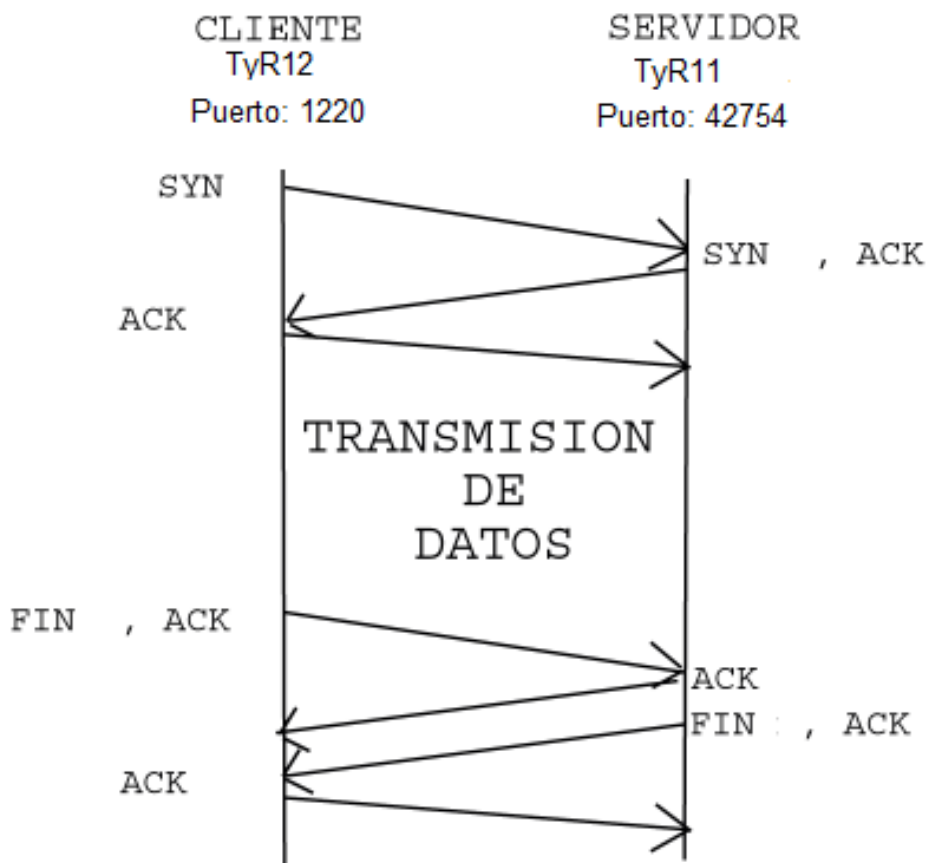
```
juanma@ubuntu:~/Desktop$ tshark -r captura_tp2.pcap -nqz follow,tcp,hex,0
=====
Follow: tcp,hex
Filter: tcp.stream eq 0
Node 0: 10.4.11.12:42754
Node 1: 10.4.11.11:1220
00000000  50 72 6f 62 61 6e 64 6f 0a          Probando .
00000009  63 6f 6e 65 78 69 6f 6e 0a          conexion .
00000012  65 6e 74 72 65 20 63 6c 69 65 6e 74 65 20 79 20  entre cl iente y
00000022  73 65 72 76 69 64 6f 72 0a          servidor .
          00000000  50 72 6f 62 61 6e 64 6f 20 64 65 73 64 65 20 53  Probando desde S
          00000010  65 72 76 69 64 6f 72 0a          ervidor.
=====
```

Según la documentación de Tshark, este filtro se utiliza para mostrar el contenido del primer flujo TCP en formato hexadecimal.

Example: **-z "follow,tcp,hex,1"** will display the contents of the second TCP stream (the first is stream 0) in "hex" format.

```
=====
Follow: tcp,hex
Filter: tcp.stream eq 1
Node 0: 200.57.7.197:32891
Node 1: 200.57.7.198:2906
00000000  00 00 00 22 00 00 00 07 00 0a 85 02 07 e9 00 02  ..."....
00000010  07 e9 06 0f 00 0d 00 04 00 00 00 01 00 03 00 06  ....
00000020  1f 00 06 04 00 00                                ....
00000000  00 01 00 00                                ....
00000026  00 02 00 00
```

b) Realice un diagrama representando el intercambio de tramas indicando las que corresponden al establecimiento de la conexión TCP, a las de transmisión de datos a nivel aplicación, y a las del cierre de la conexión TCP.



c) ¿Todas las tramas en las que identifica el protocolo TCP transportan datos de aplicación?. ¿Si no es así puede explicar por qué?

No, no todas las tramas transportan datos de aplicación, ya que por ejemplo, las primeras 7 tramas (en mi caso) de la captura realizada transportan datos referentes a la conexión TCP, más específicamente al Three-Way Handshake (SYN, SYN/ACK y ACK).

Por último, están las tramas referidas al cierre de la conexión entre cliente y servidor (más precisamente las que contiene las flags FIN/ACK y ACK).

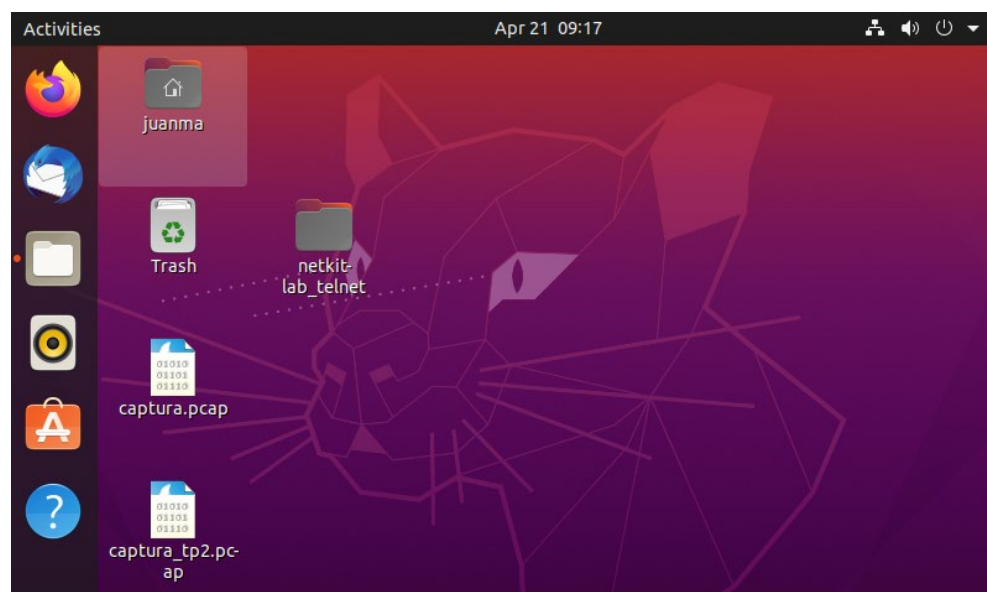
Como TCP es un protocolo Orientado a la Conexión, primero se debe confirmar que se estableció la conexión de manera exitosa mediante el envío de estas flags nombradas con anterioridad.

Segunda parte: Protocolo de acceso remoto TELNET

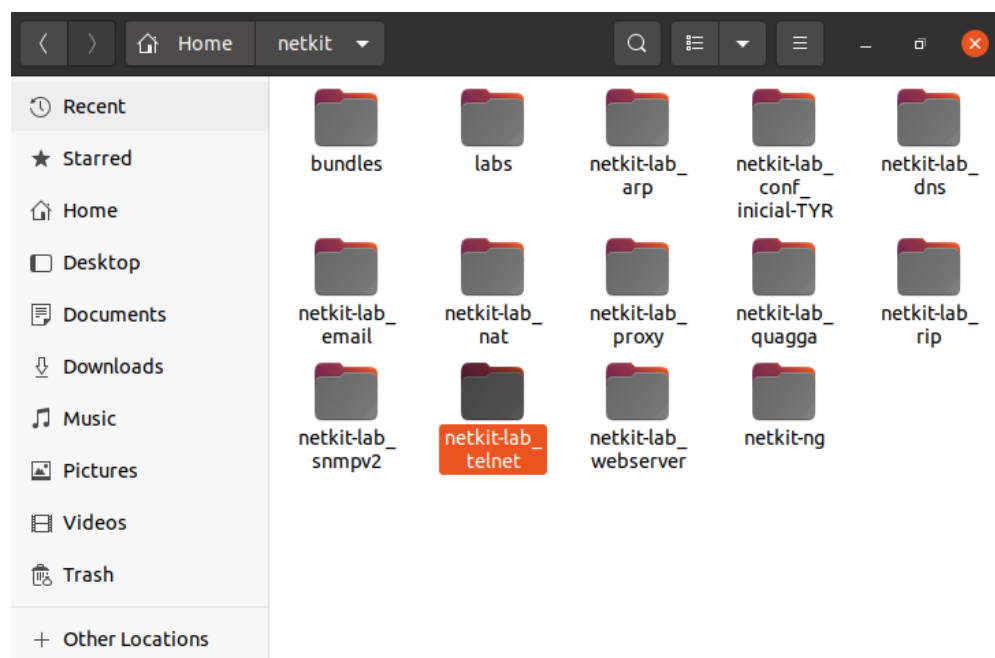
Instale e inicie en netkit el laboratorio de Telnet provisto por los docentes, disponible en https://github.com/redesunlu/netkit-labs/raw/master/tarballs/netkit-lab_telnet-TYR.tar.gz

Para dar comienzo con la segunda parte del TP N° 2, es necesario acceder al link para poder descargar el laboratorio de Telnet.

En mi caso, lo extraje en el Escritorio.



Aunque luego, por cuestiones de comodidad, decidí moverlo a la carpeta donde están ubicados los demás laboratorios de Netkit.



Una vez ubicado en mi carpeta de preferencia, desde la terminal me posiciono en la misma y ejecuto el siguiente comando:

```
lstart
```

Para iniciar el laboratorio y verificar que funcione con normalidad.

```
juanna@ubuntu:~/netkit/netkit-lab_telnet$ lstart
===== Starting lab =====
Lab directory: /home/juanna/netkit/netkit-lab_telnet
Version: 1.2
Author: Mauro A. Meloni
Email: maurom@unlu.edu.ar
Web: https://github.com/redesunlu/netkit-labs/

=====
Lab directory (host): /home/juanna/netkit/netkit-lab_telnet
Version: 1.2
Author: Mauro A. Meloni
Email: maurom@unlu.edu.ar
Web: https://github.com/redesunlu/netkit-labs/
Description:
A simple lab requiring the use of a telnet session to perform a task on a remote host

----- Netkit phase 2 initialization terminated -----

[ ok ] Starting enhanced syslogd: rsyslogd.

client login: root (automatic login)
Linux rootstrap 3.2.86-netkit-ng-K3.2 #2 Wed Mar 29 09:08:55 ART 2017 i686
Welcome to Netkit

root@client:~#

=====
Lab directory (host): /home/juanna/netkit/netkit-lab_telnet
Version: 1.2
Author: Mauro A. Meloni
Email: maurom@unlu.edu.ar
Web: https://github.com/redesunlu/netkit-labs/
Description:
A simple lab requiring the use of a telnet session to perform a task on a remote host

----- Netkit phase 2 initialization terminated -----

=====
Este servidor se encuentra en la Antártida y como usted no se encuentra allí, no tiene manera de acceder al teclado físico del equipo.

Pero puede acceder en forma remota mediante protocolo telnet.
Utilice el nombre de usuario 'root' y la clave 'ultrasecreta'.

Tenga en cuenta dada la distancia entre el cliente y el servidor,
habrá cierta demora al ingresar comandos y recibir respuestas.
=====
```

Efectivamente, como se aprecia en la imagen, mediante la ejecución del comando se inició el laboratorio de manera correcta, abriendo una máquina llamada client y una llamada remote.

Asigne una dirección IP al host cliente dentro de la red 172.16.0.0/24.

Para hacer esto, dentro de la terminal de la máquina client, primero debo activar la interfaz eth0 mediante el uso del comando:

```
ip link set dev eth0 up
```

```
client

root@client:~# ip link set dev eth0 up
root@client:~# ip a s
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16384 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN qlen 1000
    link/ether ca:f9:17:14:f0:5a brd ff:ff:ff:ff:ff:ff
    inet6 fe80::c8f9:17ff:fe14:f05a/64 scope link
        valid_lft forever preferred_lft forever
root@client:~#
```

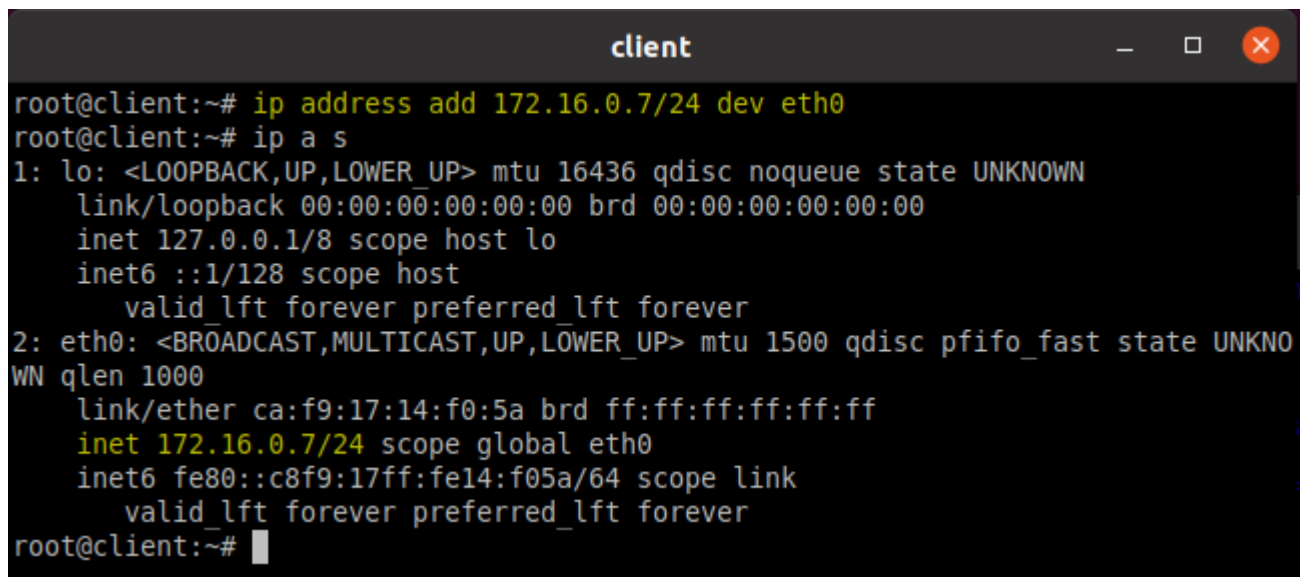
Para asignar la dirección IP al host cliente debo utilizar el comando:

```
ip address add 172.16.0.[0-254] dev eth0
```

En mi caso, opté por el número 7, por lo que ejecuté el siguiente comando:

```
ip address add 172.16.0.7/24 dev eth0
```

El cual pertenece a la red 172.16.0.0/24.

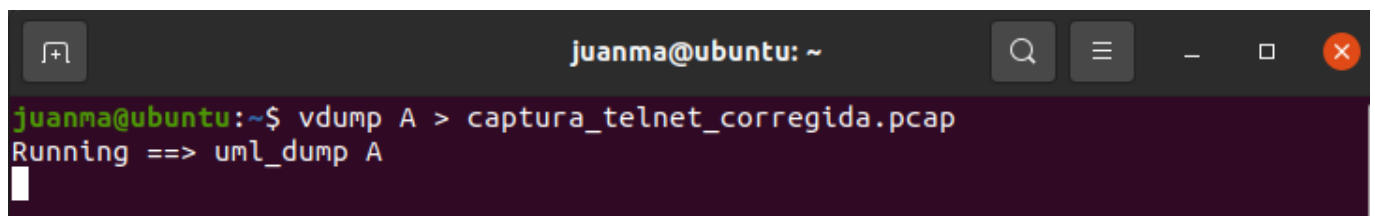


```
client
root@client:~# ip address add 172.16.0.7/24 dev eth0
root@client:~# ip a s
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOW
    link/ether ca:f9:17:14:f0:5a brd ff:ff:ff:ff:ff:ff
    inet 172.16.0.7/24 scope global eth0
    inet6 fe80::c8f9:17ff:fe14:f05a/64 scope link
        valid_lft forever preferred_lft forever
root@client:~#
```

Inicie una captura de tráfico en el enlace A.

Para realizar una captura de tráfico en el enlace A, puedo utilizar la herramienta vdump con el siguiente comando:

```
vdump A > captura_telnet_corregida.pcap
```



```
juanma@ubuntu: ~
juanma@ubuntu:~$ vdump A > captura_telnet_corregida.pcap
Running ==> uml_dump A
```

En la terminal del host cliente, conéctese mediante telnet al host remoto, cuya dirección IP es 172.16.0.10 . Utilice el nombre de usuario root y la clave ultrasecreta .

Ahora, para conectarme al host remoto mediante telnet, desde el host cliente debo utilizar el comando:

```
telnet IP_HOST_SERVIDOR
```

En este caso:

```
telnet 172.16.0.10
```

```
client
root@client:~# telnet 172.16.0.10
Trying 172.16.0.10...
Connected to 172.16.0.10.
Escape character is '^]'.
Debian GNU/Linux 7
remote login: █
```

Como se aprecia en la imagen, se estableció una conexión con 172.16.0.10.

Ahora, debo ingresar el nombre de usuario y la contraseña provista por los profesores, las cuales son:

Nombre de usuario: root

Contraseña: ultrasecreta

```
client
root@client:~# telnet 172.16.0.10
Trying 172.16.0.10...
Connected to 172.16.0.10.
Escape character is '^]'.
Debian GNU/Linux 7
remote login: root
Password:
Linux rootstrap 3.2.86-netkit-ng-K3.2 #2 Wed Mar 29 09:08:55 ART 2017 i686
=====
Bienvenido al servidor en la Antártida.

Continúe el ejercicio ejecutando el comando:

    who && who | openssl dgst

Copie y pegue la salida de este en la resolución de su práctica.
=====

root@remote:~# █
```

Efectivamente, se pudo ingresar de manera exitosa al host remoto.

Con la sesión iniciada en remoto, ejecute el siguiente comando respetando la sintaxis: who && who | openssl dgst.

Copie la salida de dicho comando como resolución de este ejercicio (como texto). Añada además todos los comandos que ejecutó para lograr dicho resultado.

Ahora, debo ejecutar el siguiente comando:

```
who && who | openssl dgst
```

Obteniendo el siguiente resultado:

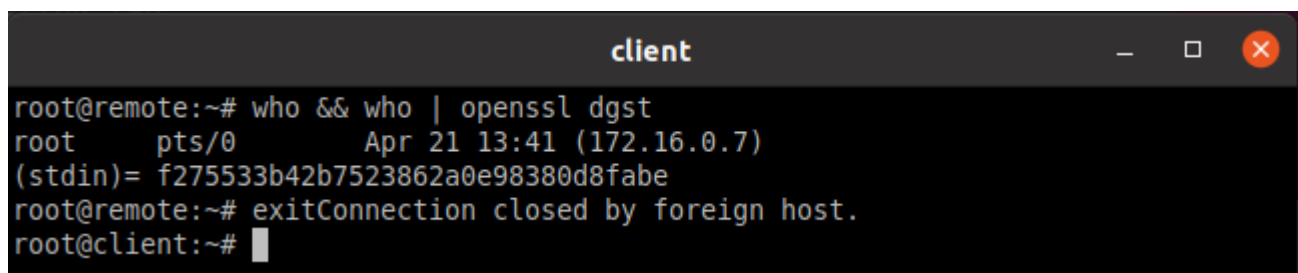
```
root@remote:~# who && who | openssl dgst
root      pts/0      Apr 21 13:41 (172.16.0.7)
(stdin)= f275533b42b7523862a0e98380d8fabe
root@remote:~#
```

El resultado como texto:

```
root@remote:~# who && who | openssl dgst
root      pts/0      Apr 21 13:41 (172.16.0.7)
(stdin)= f275533b42b7523862a0e98380d8fabe
```

Salga del host remoto escribiendo el comando exit.

Para salir del host, ejecuto el comando `exit` y obtengo el siguiente mensaje:

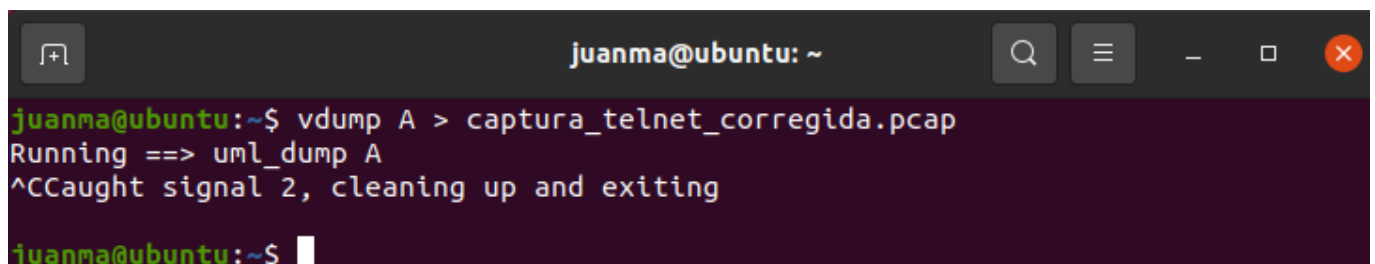


```
client
root@remote:~# who && who | openssl dgst
root      pts/0      Apr 21 13:41 (172.16.0.7)
(stdin)= f275533b42b7523862a0e98380d8fabe
root@remote:~# exit
Connection closed by foreign host.
root@client:~#
```

Lo que me indica que se cerró la conexión.

Detenga y guarde la captura de tráfico. Remítala como parte de la tarea.

Ahora, para detener la captura, basta con apretar la combinación de teclas CTRL + C en la terminal donde la inicié.



```
juanma@ubuntu: ~
juanma@ubuntu:~$ vdump A > captura_telnet_corregida.pcap
Running ==> uml_dump A
^CCaught signal 2, cleaning up and exiting
juanma@ubuntu:~$
```

Analice la captura:

a) Identifique e indique las tramas que corresponden a la transmisión de datos a nivel aplicación, cuáles a protocolos auxiliares (si existen) y al establecimiento y cierre de la conexión TCP. (referenciando por número de trama en la captura)

Una vez finalizada la captura, para analizarla debo utilizar el comando:

```
tshark -r captura_telnet_corregida.pcap
```

Obteniendo el siguiente resultado:


```

juanna@ubuntu:~$ tshark -r captura_telnet_corregida.pcap
  1  0.000000  be:65:eb:70:76:72 → Broadcast    ARP 42 Who has 172.16.0.10? Tell 172.16.0.7
  2  0.252778  aa:8c:94:71:c2:23 → be:65:eb:70:76:72 ARP 42 172.16.0.10 is at aa:8c:94:71:c2:23
  3  0.253426  172.16.0.7 → 172.16.0.10    TCP 74 43738 → 23 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=4294950599 TSecr=0 WS=4
  4  0.602970  172.16.0.10 → 172.16.0.7    TCP 74 23 → 43738 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=4294948299 TSecr=4294950599 WS=4
  5  0.603100  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=1 Ack=1 Win=14600 Len=0 TSval=4294950659 TSecr=4294948299
  6  0.612046  172.16.0.7 → 172.16.0.10    TELNET 90 Telnet Data ...
  7  0.866816  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=1 Ack=25 Win=14480 Len=0 TSval=4294948334 TSecr=4294950659
  8  1.928867  172.16.0.10 → 172.16.0.7    TELNET 78 Telnet Data ...
  9  1.929739  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=25 Ack=13 Win=14600 Len=0 TSval=4294950792 TSecr=4294948440
 10  1.931044  172.16.0.7 → 172.16.0.10    TELNET 69 Telnet Data ...
 11  2.187450  172.16.0.10 → 172.16.0.7    TELNET 81 Telnet Data ...
 12  2.187564  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=28 Ack=28 Win=14480 Len=0 TSval=4294948467 TSecr=4294950792
 13  2.196975  172.16.0.7 → 172.16.0.10    TELNET 75 Telnet Data ...
 14  2.449901  172.16.0.10 → 172.16.0.7    TELNET 84 Telnet Data ...
 15  2.451903  172.16.0.7 → 172.16.0.10    TELNET 100 Telnet Data ...
 16  2.712054  172.16.0.10 → 172.16.0.7    TELNET 69 Telnet Data ...
 17  2.713494  172.16.0.7 → 172.16.0.10    TELNET 69 Telnet Data ...
 18  2.994877  172.16.0.10 → 172.16.0.7    TELNET 69 Telnet Data ...
 19  2.995648  172.16.0.7 → 172.16.0.10    TELNET 69 Telnet Data ...
 20  3.254309  172.16.0.10 → 172.16.0.7    TELNET 100 Telnet Data ...
 21  3.288881  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=77 Ack=86 Win=14600 Len=0 TSval=4294950928 TSecr=4294948573
 22  11.825329  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 23  12.087932  172.16.0.10 → 172.16.0.7    TELNET 67 Telnet Data ...
 24  12.088930  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=78 Ack=87 Win=14600 Len=0 TSval=4294951808 TSecr=4294949456
 25  12.163196  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 26  12.420909  172.16.0.10 → 172.16.0.7    TELNET 67 Telnet Data ...
 27  12.421606  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=79 Ack=88 Win=14600 Len=0 TSval=4294951841 TSecr=4294949490
 28  12.444971  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 29  12.708503  172.16.0.10 → 172.16.0.7    TELNET 67 Telnet Data ...
 30  12.754481  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=80 Ack=89 Win=14600 Len=0 TSval=4294951874 TSecr=4294949518
 31  12.883030  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 32  13.137449  172.16.0.10 → 172.16.0.7    TELNET 67 Telnet Data ...
 33  13.138399  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=81 Ack=90 Win=14600 Len=0 TSval=4294951913 TSecr=4294949562
 34  13.805507  172.16.0.7 → 172.16.0.10    TELNET 68 Telnet Data ...
 35  14.067758  172.16.0.10 → 172.16.0.7    TELNET 68 Telnet Data ...
 36  14.068259  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=83 Ack=92 Win=14600 Len=0 TSval=4294952006 TSecr=4294949654
 37  14.329562  172.16.0.10 → 172.16.0.7    TELNET 76 Telnet Data ...
 38  14.330144  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=83 Ack=102 Win=14600 Len=0 TSval=4294952032 TSecr=4294949680
 39  15.719327  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 40  16.011905  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=102 Ack=84 Win=14480 Len=0 TSval=4294949849 TSecr=4294952171
 41  16.093670  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 42  16.345548  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=102 Ack=85 Win=14480 Len=0 TSval=4294949883 TSecr=4294952208
 43  16.697718  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 44  16.949190  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=102 Ack=86 Win=14480 Len=0 TSval=4294949943 TSecr=4294952269
 45  17.022131  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 46  17.275444  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=102 Ack=87 Win=14480 Len=0 TSval=4294949976 TSecr=4294952301
 47  17.551927  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 48  17.803815  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=102 Ack=88 Win=14480 Len=0 TSval=4294950029 TSecr=4294952354
 49  17.980714  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 50  18.239777  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=102 Ack=89 Win=14480 Len=0 TSval=4294950072 TSecr=4294952397
 51  18.251684  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 52  18.504279  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=102 Ack=90 Win=14480 Len=0 TSval=4294950099 TSecr=4294952424
 53  18.571475  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 54  18.832994  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=102 Ack=91 Win=14480 Len=0 TSval=4294950131 TSecr=4294952456
 55  18.968918  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 56  19.221044  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=102 Ack=92 Win=14480 Len=0 TSval=4294950170 TSecr=4294952496
 57  19.244515  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 58  19.498782  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=102 Ack=93 Win=14480 Len=0 TSval=4294950198 TSecr=4294952523
 59  19.501439  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 60  19.763144  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=102 Ack=94 Win=14480 Len=0 TSval=4294950224 TSecr=4294952549
 61  19.851343  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 62  20.102009  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=102 Ack=95 Win=14480 Len=0 TSval=4294950259 TSecr=4294952584
 63  20.130225  172.16.0.7 → 172.16.0.10    TELNET 68 Telnet Data ...
 64  20.491947  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=102 Ack=97 Win=14480 Len=0 TSval=4294950286 TSecr=4294952612
 65  20.492208  172.16.0.10 → 172.16.0.7    TELNET 68 Telnet Data ...
 66  20.492389  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=97 Ack=104 Win=14600 Len=0 TSval=4294952648 TSecr=4294950289
 67  20.767654  172.16.0.10 → 172.16.0.7    TELNET 540 Telnet Data ...
 68  20.767867  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=97 Ack=578 Win=14600 Len=0 TSval=4294952676 TSecr=4294950323
 69  21.314820  172.16.0.10 → 172.16.0.7    TELNET 81 Telnet Data ...
 70  21.315553  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=97 Ack=593 Win=14600 Len=0 TSval=4294952730 TSecr=4294950370
 71  35.305062  172.16.0.7 → 172.16.0.10    TELNET 91 Telnet Data ...
 72  35.574468  172.16.0.10 → 172.16.0.7    TELNET 91 Telnet Data ...
 73  35.575189  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=122 Ack=618 Win=14600 Len=0 TSval=4294954156 TSecr=4294951804
 74  36.790766  172.16.0.7 → 172.16.0.10    TELNET 68 Telnet Data ...
 75  37.192945  172.16.0.10 → 172.16.0.7    TELNET 68 Telnet Data ...
 76  37.193115  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=124 Ack=620 Win=14600 Len=0 TSval=4294954318 TSecr=4294951953
 77  37.452557  172.16.0.10 → 172.16.0.7    TELNET 173 Telnet Data ...
 78  37.453229  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=124 Ack=727 Win=14600 Len=0 TSval=4294954344 TSecr=4294951993
 79  49.386240  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 80  49.639584  172.16.0.10 → 172.16.0.7    TELNET 67 Telnet Data ...
 81  49.640411  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=125 Ack=728 Win=14600 Len=0 TSval=4294955563 TSecr=4294953212
 82  50.048262  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 83  50.301228  172.16.0.10 → 172.16.0.7    TELNET 67 Telnet Data ...
 84  50.302230  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=126 Ack=729 Win=14600 Len=0 TSval=4294955629 TSecr=4294953278
 85  50.505464  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 86  50.766131  172.16.0.10 → 172.16.0.7    TELNET 67 Telnet Data ...
 87  50.767117  172.16.0.7 → 172.16.0.10    TELNET 67 Telnet Data ...
 88  51.019259  172.16.0.10 → 172.16.0.7    TELNET 67 Telnet Data ...
 89  51.063261  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [ACK] Seq=128 Ack=731 Win=14600 Len=0 TSval=4294955705 TSecr=4294953350
 90  51.819600  172.16.0.7 → 172.16.0.10    TELNET 68 Telnet Data ...
 91  52.097900  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [FIN, ACK] Seq=731 Ack=130 Win=14480 Len=0 TSval=4294953455 TSecr=4294955781
 92  52.098988  172.16.0.7 → 172.16.0.10    TCP 66 43738 → 23 [FIN, ACK] Seq=130 Ack=732 Win=14600 Len=0 TSval=4294955809 TSecr=4294953455
 93  52.351217  172.16.0.10 → 172.16.0.7    TCP 66 23 → 43738 [ACK] Seq=732 Ack=131 Win=14480 Len=0 TSval=4294953483 TSecr=4294955809

```

Tramas referidas al establecimiento de la conexión:

Tramas 3 – 4 – 5

```
3 0.253426 172.16.0.7 → 172.16.0.10 TCP 74 43738 → 23 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=4294950599 TSecr=0 WS=4
4 0.602970 172.16.0.10 → 172.16.0.7 TCP 74 23 → 43738 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1460 SACK_PERM=1 TSval=4294948299 TSecr=4294950599 WS=4
5 0.603100 172.16.0.7 → 172.16.0.10 TCP 66 43738 → 23 [ACK] Seq=1 Ack=1 Win=14600 Len=0 TSval=4294950659 TSecr=4294948299
```

En ellas se puede apreciar el famoso Three-Way Handshake (SYN, SYN/ACK, ACK).

Tramas referidas al cierre de la conexión:

Tramas 91 – 92 – 93

```
91 52.097900 172.16.0.10 → 172.16.0.7 TCP 66 23 → 43738 [FIN, ACK] Seq=731 Ack=130 Win=14480 Len=0 TSval=4294953455 TSecr=4294955781
92 52.098988 172.16.0.7 → 172.16.0.10 TCP 66 43738 → 23 [FIN, ACK] Seq=130 Ack=732 Win=14600 Len=0 TSval=4294955809 TSecr=4294953455
93 52.351217 172.16.0.10 → 172.16.0.7 TCP 66 23 → 43738 [ACK] Seq=732 Ack=131 Win=14480 Len=0 TSval=4294953483 TSecr=4294955809
```

En ellas se puede apreciar el Four-Way Handshake referente al cierre de la conexión en TCP (FIN/ACK, FIN/ACK)

(Me queda la duda sobre el último ACK, si pertenece al cierre o no).

Tramas referentes a datos de nivel de aplicación:

```
6 0.612046 172.16.0.7 → 172.16.0.10 TELNET 90 Telnet Data ...
7 0.866816 172.16.0.10 → 172.16.0.7 TCP 66 23 → 43738 [ACK] Seq=1 Ack=25 Win=14480 Len=0 TSval=4294948334 TSecr=4294950659
8 1.928867 172.16.0.10 → 172.16.0.7 TELNET 78 Telnet Data ...
9 1.929739 172.16.0.7 → 172.16.0.10 TCP 66 43738 → 23 [ACK] Seq=25 Ack=13 Win=14600 Len=0 TSval=4294950792 TSecr=4294948440
10 1.931044 172.16.0.7 → 172.16.0.10 TELNET 69 Telnet Data ...
11 2.187450 172.16.0.10 → 172.16.0.7 TELNET 81 Telnet Data ...
12 2.187564 172.16.0.10 → 172.16.0.7 TCP 66 23 → 43738 [ACK] Seq=28 Ack=28 Win=14480 Len=0 TSval=4294948467 TSecr=4294950792
13 2.196975 172.16.0.7 → 172.16.0.10 TELNET 75 Telnet Data ...
14 2.449901 172.16.0.10 → 172.16.0.7 TELNET 84 Telnet Data ...
15 2.451903 172.16.0.7 → 172.16.0.10 TELNET 100 Telnet Data ...
16 2.712054 172.16.0.10 → 172.16.0.7 TELNET 69 Telnet Data ...
17 2.713494 172.16.0.7 → 172.16.0.10 TELNET 69 Telnet Data ...
18 2.994877 172.16.0.10 → 172.16.0.7 TELNET 69 Telnet Data ...
19 2.995648 172.16.0.7 → 172.16.0.10 TELNET 69 Telnet Data ...
20 3.254309 172.16.0.10 → 172.16.0.7 TELNET 100 Telnet Data ...
21 3.288881 172.16.0.7 → 172.16.0.10 TCP 66 43738 → 23 [ACK] Seq=77 Ack=86 Win=14600 Len=0 TSval=4294950928 TSecr=4294948573
22 11.825329 172.16.0.7 → 172.16.0.10 TELNET 67 Telnet Data ...
23 12.087932 172.16.0.10 → 172.16.0.7 TELNET 67 Telnet Data ...
24 12.088930 172.16.0.7 → 172.16.0.10 TCP 66 43738 → 23 [ACK] Seq=78 Ack=87 Win=14600 Len=0 TSval=4294951808 TSecr=4294949456
25 12.163196 172.16.0.7 → 172.16.0.10 TELNET 67 Telnet Data ...
26 12.420909 172.16.0.10 → 172.16.0.7 TELNET 67 Telnet Data ...
27 12.421606 172.16.0.7 → 172.16.0.10 TCP 66 43738 → 23 [ACK] Seq=79 Ack=88 Win=14600 Len=0 TSval=4294951841 TSecr=4294949490
28 12.444971 172.16.0.7 → 172.16.0.10 TELNET 67 Telnet Data ...
29 12.708503 172.16.0.10 → 172.16.0.7 TELNET 67 Telnet Data ...
30 12.754481 172.16.0.7 → 172.16.0.10 TCP 66 43738 → 23 [ACK] Seq=80 Ack=89 Win=14600 Len=0 TSval=4294951874 TSecr=4294949518
31 12.883030 172.16.0.7 → 172.16.0.10 TELNET 67 Telnet Data ...
32 13.137449 172.16.0.10 → 172.16.0.7 TELNET 67 Telnet Data ...
33 13.138399 172.16.0.7 → 172.16.0.10 TCP 66 43738 → 23 [ACK] Seq=81 Ack=90 Win=14600 Len=0 TSval=4294951913 TSecr=4294949562
34 13.805507 172.16.0.7 → 172.16.0.10 TELNET 68 Telnet Data ...
35 14.067758 172.16.0.10 → 172.16.0.7 TELNET 68 Telnet Data ...
36 14.068259 172.16.0.7 → 172.16.0.10 TCP 66 43738 → 23 [ACK] Seq=83 Ack=92 Win=14600 Len=0 TSval=4294952006 TSecr=4294949654
37 14.329562 172.16.0.10 → 172.16.0.7 TELNET 76 Telnet Data ...
38 14.330144 172.16.0.7 → 172.16.0.10 TCP 66 43738 → 23 [ACK] Seq=83 Ack=102 Win=14600 Len=0 TSval=4294952032 TSecr=4294949680
```

Se las puede identificar por la presencia de “Telnet Data” en su descripción.

Tramas referentes a protocolos auxiliares:

Tramas 1 – 2

```
1 0.000000 be:65:eb:70:76:72 → Broadcast ARP 42 Who has 172.16.0.10? Tell 172.16.0.7
2 0.252778 aa:8c:94:71:c2:23 → be:65:eb:70:76:72 ARP 42 172.16.0.10 is at aa:8c:94:71:c2:23
```

Se puede identificar el protocolo auxiliar ARP.

b) Comente las características de la información en tránsito con respecto a la confidencialidad.

La gran desventaja que pude apreciar sobre Telnet es su bajo nivel de seguridad respecto a la confidencialidad, ya que, sus datos no viajan de manera cifrada (incluida el nombre de usuario y la contraseña) lo cual es gravísimo, ya que cualquier persona que esté “sniffeando” el tráfico de la red puede acceder a su usuario y contraseña, y, por ende, a la máquina remota.

Según mi opinión, esta desventaja es la gran causante de que Telnet casi no se utilice en la actualidad.