Práctica 6

Capa de Transporte - Parte II

1. Utilizando la máquina virtual, use Wireshark para capturar paquetes enviados y recibidos en cada uno de los siguientes casos. Para ello, arranque la captura en la interfaz con IP 172.28.0.1 antes de realizar los incisos A, B, C y D.

a. Abra un navegador e ingrese a la URL: www.redes.unlp.edu.ar. Analice la secuencia de segmentos TCP que permiten la apertura del canal de comunicación por el cual posteriormente viajarán los mensajes HTTP intercambiados. ¿Con qué nombre se conoce a dicha secuencia? ¿Qué flags se utilizan en cada uno de los segmentos intercambiados? ¿Qué indica cada uno de estos flags?

Negociación en tres pasos o *Three-way handshake*
El lado cliente de una conexión realiza una apertura activa de un puerto enviando un paquete SYN inicial al servidor como parte de la negociación en tres pasos. En caso de que se encuentre abierto el puerto, el lado servidor respondería a la petición SYN válida con un paquete SYN/ACK. Finalmente, el cliente debería responderle al servidor con un ACK, completando así la negociación en tres pasos (SYN, SYN/ACK y ACK) y la fase de establecimiento de conexión.

Se utilizan los flags SYN en el primer mensaje del cliente al servidor, SYN y ACK en el segundo del servidor al cliente, y ACK en el tercero del cliente al servidor.
SYN se utiliza para solicitar el establecimiento de la conexion, y ACK se usa para la confirmacion del flag SYN.

b. Cierre el navegador. Analice la secuencia de segmentos TCP que ocurren al hacerlo ¿Cuál es el objetivo éstos? ¿Qué flags se utilizan en cada uno de dichos segmentos? ¿Qué indica cada uno de estos flags?

Negociación en cuatro pasos o four-way handshake
La fase de finalización de la conexión utiliza una negociación en cuatro pasos (*four-way handshake*), terminando la conexión desde cada lado independientemente. Sin embargo, es posible realizar la finalización de la conexión en 3 fases; enviando el segmento FIN y el ACK en uno solo. Cuando uno de los dos extremos de la conexión desea parar su "mitad" de conexión transmite un segmento con el *flag* FIN en 1, que el otro interlocutor asentirá con un ACK. Por tanto, una desconexión típica requiere un par de segmentos FIN y ACK desde cada lado de la conexión.

Una conexión puede estar "medio abierta" en el caso de que uno de los lados la finalice pero el otro no. El lado que ha dado por finalizada la conexión no puede enviar más datos pero la otra parte si podrá.

El flag FIN de quien lo envia significa que ya no enviara datos, pero todavia puede recibirlos. La respuesta ACK a cada flag de FIN significa que el flag fue recibido.

c. Para este ejercicio debe usar tanto el navegador Chromium como Iceweasel. Utilice Chromium para ingresar a la URL: www.redes.unlp.edu.ar y seguidamente utilice Iceweasel para ingresar nuevamente a la URL: www.redes.unlp.edu.ar

i. Observe la información de Puerto Origen y Puerto destino de cada una de las comunicaciones. En base a lo observado, responda ¿Es posible conectarse 2 veces en forma simultánea al mismo lugar? ¿Qué distingue una conexión de otra? Capture el tráfico de red si considera necesario para observar dicha información.

Lo que distingue cada conexion (conjunto ip_origen, porn_origen, ip_destino, port_destino) es el numero de puerto del cliente, ya que las ip distinguen a los host y el puerto del cliente es el mismo para todos los clientes ya que debe ser un puerto conocido y podrian estarse realizando multiples solicitudes desde el mismo host.

ii. Identifique lo observado en el punto anterior utilizando el comando ss.

ss -tua
Netid State Recv-Q Send-Q Local Address:***Port*** Peer Address:Port

d. Desde la consola use el servicio tftp.

i. Primero cree un archivo llamado prueba.txt, por ejemplo:
echo Prueba > prueba.txt

ii. Ejecute tftp 172.28.0.29 y copie el archivo anterior desde su PC al servidor, a través de la opción put.
redes@redes:~$ tftp 172.28.0.29
tftp> put prueba.txt
tftp> quit

iii. Cierre la conexión, borre el archivo de su PC (rm prueba.txt) y obténgalo ahora del servidor a través de la opción get:
redes@redes:~$ tftp 172.28.0.29
tftp> get prueba.txt
tftp> quit

e. ¿Qué diferencias encuentra en cuánto a mensajes intercambiados entre los puntos A, B respecto del punto D?

Los mensajes HTTP se transmiten por TCP, y los mensajes TFTP se envian por UDP, por lo que TFTP debe implementas un mecanismo para “establecer” las conexiones para evitar la perdida de datos.

f. ¿Qué diferencias encuentra en el punto D respecto a los anteriores respecto a utilización de puertos y protocolo de transporte utilizado?

Los mensajes HTTP se transmiten por TCP, y los mensajes TFTP se envian por UDP.
En los mensaje HTTP el puerto que se utiliza es siempre el 80, mientras que en TFTP el primer mensaje se envia al puerto 69 y en ese mensaje el cliente envia el numero de puerto que usara para la comunicación, y luego el servidor desde un puerto no privilegiado “establece” la conexión.

2. Investigue los distintos tipos de estado que puede tener una conexión TCP.
(Ver la página: https://thewalnut.io/app/release/73/#time=21*)*

Estados en TCP
CLOSED : No hay conexión activa ni pendiente.
LISTEN: El servidor espera una llamada.
SYN RCVD: Llegó una solicitud de conexión; espera ACK.
SYN SENT: La aplicación comenzó a abrir una conexión.
ESTABLISHED: Estado normal de transferencia de datos.
FIN WAIT 1: La aplicación dijo que ya terminó.
FIN WAIT 2: El otro lado acordó liberar.
TIMED WAIT: Espera a que todos los paquetes mueran.
CLOSING: Ambos lados intentaron cerrar simultáneamente.
CLOSE WAIT: El otro lado inició una liberación.
LAST ACK: Espera a que todos los paquetes mueran.

3. Dada la siguiente salida del comando ss, responda:

```
Netid  State      Recv-Q Send-Q   Local Address:Port       Peer Address:Port
tcp    LISTEN     0      128                *:22                    *:*      users:(("sshd",pid=468,fd=29))
tcp    LISTEN     0      128                *:80                    *:*      users:(("apache2",pid=991,fd=95))
udp    LISTEN     0      128       163.10.5.222:53                  *:*      users:(("named",pid=452,fd=10))
tcp    ESTAB      0      0         163.10.5.222:59736    64.233.163.120:443    users:(("x-www-browser",pid=1079,fd=51))
tcp    CLOSE-WAIT 0      0         163.10.5.222:41654     200.115.89.30:443    users:(("x-www-browser",pid=1079,fd=50))
tcp    ESTAB      0      0         163.10.5.222:59737    64.233.163.120:443    users:(("x-www-browser",pid=1079,fd=55))
tcp    ESTAB      0      0         163.10.5.222:33583     200.115.89.15:443    users:(("x-www-browser",pid=1079,fd=53))
tcp    ESTAB      0      0         163.10.5.222:45293     64.233.190.99:443    users:(("x-www-browser",pid=1079,fd=59))
tcp    LISTEN     0      128                *:25                    *:*      users:(("postfix",pid=627,fd=3))
tcp    ESTAB      0      0            127.0.0.1:22            127.0.0.1:41220  users:(("sshd",pid=1418,fd=3),
                                                                                       ("sshd",pid=1416,fd=3))
tcp    ESTAB      0      0         163.10.5.222:52952     64.233.190.94:443    users:(("x-www-browser",pid=1079,fd=29))
tcp    TIME-WAIT  0      0         163.10.5.222:36676     54.149.207.17:443    users:(("x-www-browser",pid=1079,fd=3))
tcp    ESTAB      0      0         163.10.5.222:52960     64.233.190.94:443    users:(("x-www-browser",pid=1079,fd=67))
tcp    ESTAB      0      0         163.10.5.222:50521     200.115.89.57:443    users:(("x-www-browser",pid=1079,fd=69))
tcp    SYN-SENT   0      0         163.10.5.222:52132        43.232.2.2:9500   users:(("x-www-browser",pid=1079,fd=70))
tcp    ESTAB      0      0            127.0.0.1:41220         127.0.0.1:22     users:(("ssh",pid=1415,fd=3))
udp    LISTEN     0      128          127.0.0.1:53                  *:*      users:(("named",pid=452,fd=9))
```

a. ¿Cuántas conexiones hay establecidas?
Hay establecidas 8 conexiones, ya que aparecen 9 conexiones que estan en estado de established, pero la conexión de 127.0.0.1:22 a 127.0.0.1:41220 y 127.0.0.1:41220 a 127.0.0.1:22 son en realidad la misma conexión pero aparece como doble porque una es la respuesta a la otra, asi que solamente cuenta como una.

b. ¿Cuántos puertos hay abiertos a la espera de posibles nuevas conexiones?
5

```
tcp    LISTEN    0    128          *:22
tcp    LISTEN    0    128          *:80
tcp    LISTEN    0    128    163.10.5.222:53
tcp    LISTEN    0    128          *:25
udp    LISTEN    0    128     127.0.0.1:53
```

c. El cliente y el servidor de las comunicaciones HTTPS (puerto 443), ¿residen en la misma máquina?
No, ya que la IP del cliente (163.10.5.222) es distinta a la de los servidores (64.233.163.120:443 y 200.115.89.30:443)

d. El cliente y el servidor de la comunicación SSH (puerto 22), ¿residen en la misma máquina?
El cliente y servidor de las comunicaciones SSH(22) residen en la misma máquina
puesto que tienen la misma dirección IP (127.0.0.1).

e. Liste los nombres de todos los procesos asociados con cada comunicación. Indique para cada uno si se trata de un proceso cliente o uno servidor.

Los servicios establecidos en los puertos tcp 22 son del proceso sshd (daemon de conexiones ssh), en el puerto tcp 80 el de servicio web (apache2), y en el puerto udp 53 son named, que corresponde a funciones de dns.
Los procesos del cliente se conectan al puerto tcp 443 corresponden al navegador.

f. ¿Cuáles conexiones tuvieron el cierre iniciado por el host local y cuáles por el remoto?

tcp CLOSE-WAIT 163.10.5.222:41654 200.115.89.30:443 users:(("x-www-browser",pid=1079,fd=50))
fue cerrada por el host remoto.
tcp TIME-WAIT 163.10.5.222:36676 54.149.207.17:443 users:(("x-www browser",pid=1079,fd=3))
fue cerrada por el host local.

CLOSE_WAIT indica que el extremo remoto (otro lado de la conexión) ha cerrado la conexión.
TIME_WAIT indica que el extremo local (de este lado) ha cerrado la conexión.

La conexión se mantiene alrededor de lo que cualquier retraso de los paquetes puede ser igualada a la conexión y se maneja adecuadamente. Las conexiones serán removidos cuando el tiempo de espera dentro de los cuatro minutos.

g. ¿Cuántas conexiones están aún pendientes por establecerse?
1, el cliente envio el primer mensaje de la negociacion en tres pasos, y se quedo en estado de SYN-SENT esperando la confirmacion del servidor

4. Dadas las salidas de los siguientes comandos ejecutados en el cliente y el servidor, responder:

servidor# ss -natu | grep 110
tcp LISTEN 0 0 *:110 *:*
tcp SYN-RECV 0 0 157.0.0.1:110 157.0.11.1:52843

cliente# ss -natu | grep 110
tcp SYN-SENT 0 1 157.0.11.1:52843 157.0.0.1:110

a. ¿Qué segmentos llegaron y cuáles se están perdiendo en la red?
El segmento que se está perdiendo en la red es el segmento SYN ACK del servidor.

Llego el SYN del cliente al servidor, y el cliente todavia no recibio el SYN, AKC, ya que el cliente esta en estado de SYN-SENT, y el servidor en SYN-RECV, por lo que respondio, pero no lo recibio el cliente y sigue esperando el SYN ACK.

b. ¿A qué protocolo de capa de aplicación y de transporte se está intentando conectar el cliente?
Capa de aplicacion --> POP (puerto 110)
Capa de transporte --> TCP

c. ¿Qué flags tendría seteado el segmento perdido?
Tendra seteado los flags SYN y ACK.

5. ¿Cual es el puerto por defecto que se utiliza en los siguientes servicios?

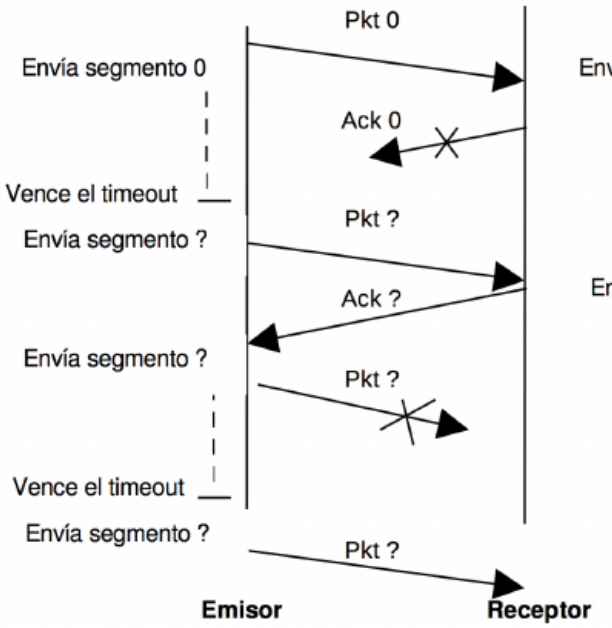
Web	→	80 / TCP
SSH	→	22 / TCP
DNS	→	53 / UDP
Web Seguro	→	443 / TCP
POP3	→	110 / TCP
IMAP	→	143 / TCP
SMTP	→	25 / TCP
FTP	→	20 / TCP (Data), 21 / TCP (Control)
TFTP	→	69 / UDP

Investigue en qué lugar en Linux y en Windows está descripta la asociación utilizada por defecto para cada servicio.

Sistemas operativos Linux y UNIX
    /etc/services
Sistemas operativos Windows
    %SystemRoot%\system32\drivers\etc\services

/etc/services : En cada línea de este fichero se especifican el nombre, número de puerto, protocolo utilizado y aliases de los servicios de red. Por ejemplo, para especificar que el servicio de smtp utilizará el puerto 25, el protocolo TCP y que un *alias* para él es mail, existirá una línea similar a la siguiente:
smtp 25/tcp mail
El fichero /etc/services es utilizado por los servidores y por los clientes para obtener el número de puerto en el que deben escuchar o al que deben enviar peticiones, de forma que se pueda cambiar (aunque no es lo habitual) un número de puerto sin afectar a las aplicaciones; de esta forma, podemos utilizar el nombre del servicio en un programa y la función getservicebyname() en lugar de utilizar el número del puerto.

6) Complete los ? de la siguiente secuencia
En caso de que se enviara un paquete más, volvería a comenzar desde el 0 puesto que en el stop and wait se van intercalando paquetes 0 y 1.



si, Pkt 0

si, Ack 0

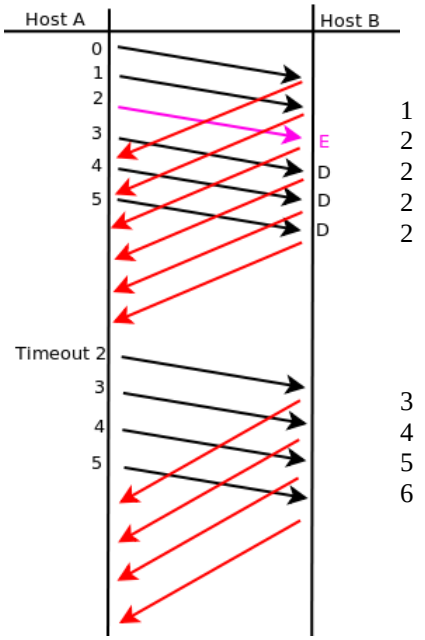
si, Pkt 1

si, Pkt 1

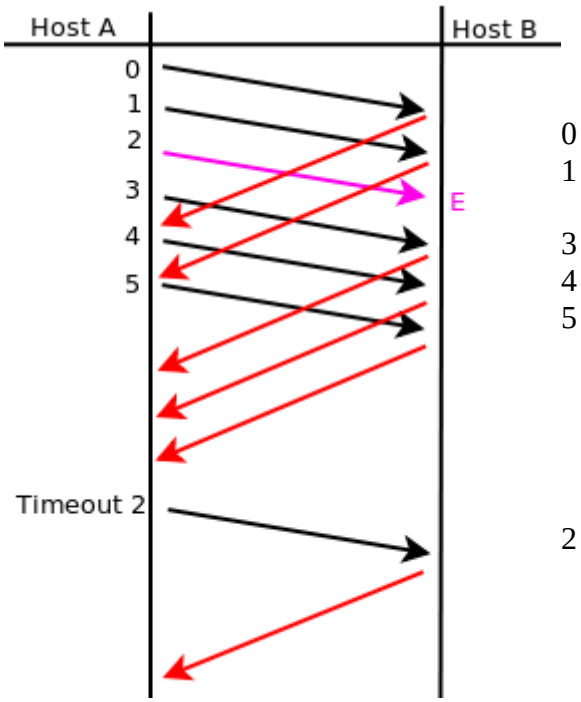
7. Explique la lógica de Go BackN.
En un protocolo GBN, el emisor puede transmitir varios paquetes (si estan disponibles) sin tener que esperar a que sean reconocidos, pero está restringido a no tener más de un número máximo permitido, N (para así no congestionar los enlaces) de paquetes no reconocidos en el canal. N suele denominarse tamaño de ventana y cuando el protocolo opera esta ventana se va desplazando hacia adelante, aumentando el número de segmentos que pueden ser enviados, por eso se lo denomina un protocolo de ventana deslizante.
En el protocolo GBN, el reconocimiento de un paquete es acumulativo, es decir que si se manda el reconocimiento de un paquete n, todos los paquetes con un número menor o igual que n han sido correctamente recibidos por el receptor. Si se produce un fin de temporización, el emisor reenvía todos los paquetes que no hayan sido reconocidos por el receptor.

8. Suponiendo Go Back N; tamaño de ventana 4 y sabiendo que E indica que el mensaje llegó con errores y que D significa que el mensaje será descartado por llegar fuera de secuencia. Indique en el siguiente gráfico, la numeración de los ACK que el host B envía al Host A.

Go back n puede confirmar con el proximo paquete que espera recibir, ya que cuando confirma un paquete entonces todos los anteriores se recibieron correctamente.



9. Suponiendo Selective Repeat; tamaño de ventana 4 y sabiendo que E indica que el mensaje llegó con errores. Indique en el siguiente gráfico, la numeración de los ACK que el host B envía al Host A.



10. ¿Qué restricción existe sobre el tamaño de ventanas en el protocolo Selective Repeat?

Cantidad de paquetes enviados menor a la cantidad de ventanas menos 1. Esto sucede ya que cuando el numero de secuencia llega al mayor valor posible, este se reinicia y comienza de 0. Si se envian mas paquetes que el limite establecido podria darse que lleguen 2 paquetes con igual numero de secuencia de distintas rafagas, y que no se pueda distinguir a cual corresponde. Al limitar la cantidad de paquetes enviados se imposibilita esta situacion, ya que no alcanzaran a cubrir todos los numeros de sequencia posibles si se usa esa cantidad menos un paquetes, porque la ventana no se movera hasta que los paquetes con numero de secuencia mas baja se confirmen.

11. Investigue cómo funciona el protocolo de aplicación FTP teniendo en cuenta las diferencias en su funcionamiento cuando se utiliza el modo activo de cuando se utiliza el modo pasivo ¿En qué se diferencian estos tipos de comunicaciones del resto de los protocolos de aplicación vistos?

FTP tiene la capacidad de trabajar de dos modos diferentes con respecto a la sesión de datos.

Activo: En el modo Activo, el cliente le debe indicar al servidor con el comando PORT en que dirección IP y puerto escuchará la conexión de datos. Es activo con respecto al servidor. Este puerto,se conectará al puerto indicado por el cliente para establecer la conexión de datos. Esto se realizará por cada operación que requiera una conexión de datos. Éste era el modo default.
El problema que presenta este modo es que se pueden encontrar complicaciones al realizar NAT (Network Address Translation) o al configurar un firewall que protege a los clientes, ya que comúnmente bloquean este tipo de solicitudes y corta la comunicación cliente – servidor.

Actualmente, no se recomienda su uso, pero existen algunos servidores viejos que solo soportan este modo.

Pasivo: En el modo Pasivo, en lugar de que el servidor se conecte al cliente, es el cliente el que se conecta al servidor. Es pasivo con respecto al servidor. El cliente debe indicarle al servidor que quiere trabajar con este modo usando el comando PASV. El servidor, como respuesta, indicará el puerto, con el mismo formato que el comando PORT, donde esperará la conexión. Es el modo recomendado actualmente [RFC-1579].

Diferencias con TFTP: Usa el puerto 69 y UDP como protocolo de transporte. TFTP usa comandos simples mientras que FTP usa comandos robustos. TFTP es un protocolo no orientado a la conexión, FTP si y utiliza dos conexiones, una para mantener el control de la conexión y otro para los datos. TFTP requiere menos memoria y esfuerzo de programación, FTP requiere más memoria y más esfuerzo en la programación.
Comandos FTP: mkdir, rmdir, ls, cd, get (recuperar archivos en posición local dondeestá la terminal) y put (colocar archivo desde posición actual en máquina local desde terminal).

12. Utilizando el Live CD conéctese al servidor ftp utilizando el comando ftp ftp.redes.unlp.edu.ar utilizando los siguientes datos:
a. Nombre de usuario: redes
b. Password: redes
c. Pruebe la transferencia de un archivo cualquiera hacia y desde el servidor.
d. Utilice Wireshark para obtener capturas de transferencias de archivos usando primero el modo activo y luego el modo pasivo.

```
@redes:~$ ftp ftp.redes.unlp.edu.ar
Connected to www.redes.unlp.edu.ar.
220 661850c39d3c FTP server (Version 6.4/OpenBSD/Linux-ftpd-0.17) ready.
Name (ftp.redes.unlp.edu.ar:redes): redes
331 Password required for redes.
Password:
230 User redes logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 PORT command successful.
150 Opening ASCII mode data connection for '/bin/ls'.
total 16
-rw-r--r-- 1 redes redes  220 Apr  9  2014 .bash_logout
-rw-r--r-- 1 redes redes 3637 Apr  9  2014 .bashrc
-rw-r--r-- 1 redes redes  675 Apr  9  2014 .profile
-rw-r----- 1 redes redes   11 Apr 19 22:24 prueba.txt
226 Transfer complete.
ftp> get .profile
local: .profile remote: .profile
200 PORT command successful.
150 Opening BINARY mode data connection for '.profile' (675 bytes).
226 Transfer complete.
675 bytes received in 0.00 secs (936.3348 kB/s)
ftp> send prueba.txt
local: prueba.txt remote: prueba.txt
200 PORT command successful.
150 Opening BINARY mode data connection for 'prueba.txt'.
226 Transfer complete.
```

11 bytes sent in 0.00 secs (429.6875 kB/s)
ftp> passive
Passive mode on.
ftp> send prueba.txt
local: prueba.txt remote: prueba.txt
227 Entering Passive Mode (172,28,0,50,188,57)
150 Opening BINARY mode data connection for 'prueba.txt'.
226 Transfer complete.
11 bytes sent in 0.00 secs (358.0729 kB/s)
ftp> sendport
Use of PORT cmds off.
ftp> get prueba.txt
local: prueba.txt remote: prueba.txt
227 Entering Passive Mode (172,28,0,50,168,84)
150 Opening BINARY mode data connection for 'prueba.txt' (11 bytes).
226 Transfer complete.
11 bytes received in 0.00 secs (51.3980 kB/s)
ftp> quit
221 Goodbye.

Programación de sockets

Resuelva los siguientes ejercicios utilizando el lenguaje de programación que prefiera (por simpleza, se recomiendan Python o Ruby).

13. Desarrolle un cliente y un servidor, donde el cliente envíe un mensaje al servidor y este último imprima en pantalla el contenido del mismo.

a. Utilizando UDP.

b. Utilizando TCP.

14. Compare ambas implementaciones. ¿Qué diferencia nota entre la implementación de cada una? ¿Cuál le parece más simple?

Ejercicio de parcial.

15. Dada la salida que se muestra en la imagen, responda los ítems debajo.

```
Netid  State      Local Address:Port      Peer Address:Port
udp    UNCONN               *:68                  *:*      (("dhclient",671,5))
udp    UNCONN               *:123                 *:*      (("ntpd",2138,16))
udp    UNCONN              :::123                :::*      (("ntpd",2138,17))
tcp    LISTEN               *:80                  *:*      (("nginx",23653,19),("nginx",23652,19))
tcp    LISTEN               *:22                  *:*      (("sshd",1151,3))
tcp    LISTEN        127.0.0.1:25                  *:*      (("master",11457,12))
tcp    LISTEN               *:443                 *:*      (("nginx",23653,20),("nginx",23652,20))
tcp    LISTEN               *:3306                *:*      (("mysqld",4556,13))
tcp    ESTAB         127.0.0.1:3306        127.0.0.1:34338  (("mysqld",4556,14))
tcp    TIME-WAIT  10.100.25.135:443     43.226.162.110:29148
tcp    ESTAB         127.0.0.1:48717       127.0.0.1:3306   (("ruby",28615,10))
tcp    ESTAB         127.0.0.1:3306        127.0.0.1:48717  (("mysqld",4556,17))
tcp    ESTAB         127.0.0.1:34338       127.0.0.1:3306   (("ruby",28610,9))
tcp    ESTAB      10.100.25.135:22     200.100.120.210:61576  (("sshd",13756,3),("sshd",13654,3))
tcp    LISTEN              :::22                 :::*      (("sshd",1151,4))
tcp    LISTEN              :1:25                 :::*      (("master",11457,13))
```

Suponga que ejecuta los siguientes comandos desde un host con la IP 10.100.25.90. Responda qué devuelve la ejecución de los siguientes comandos y, en caso que corresponda, especifique los flags.

a. hping3 -p 3306 –udp 10.100.25.135
Porn unracheable ICMP. El puerto 3306 abierto es TCP.

b. hping3 -S -p 25 10.100.25.135
El puerto es inaccesible, porque es para localhost.
Es local, no escucha hacia fuera. Si no tiene puerto escuchando R y A seteados.

c. hping3 -S -p 22 10.100.25.135
Acepta la conexión

d. hping3 -S -p 110 10.100.25.135
El puerto no esta escuchando (RA Recet Ack.)

¿Cuántas conexiones distintas hay establecidas? Justifique.

3, la conexión de 127.0.0.1:3306 a 127.0.0.1:48717, y la de 127.0.0.1:3306 a 127.0.0.1:34338, estan abiertas en ambos sentidos, pero son una misma conexión.

16. Complete en la columna Orden, el orden de aparición de los paquetes representados en cada fila.

Host A				Host B			Orden
Seq	ACK	Len		Seq	ACK	Len	
100	2421	0	->				4
308	2821	0	->				13
			<-	1419	100	1002	3
156	2780	64	->				7
220	2780	47	->				9
			<-	2821	308	1418	14
			<-	2780	220	0	8
			<-	1	100	1418	**1**
100	2780	56	->				6
			<-	2780	308	0	11
267	2780	41	->				10
			<-	2780	308	41	12
			<-	2421	100	359	5
100	1419	0	->				2