# Practica 6 TRANSPORTE Pt2

## Dudas

2 - ~~¿Es correcta mi justificacion de porque no se podria implementar Multicast en TCP?~~ →Si

3 - ~~¿La diferencia con otros protocolos es que FTP mantiene 2 conexiones?~~ →Si

4 - ~~Revisar grafico~~ → esta mal, mi grafico representa el go back N, en SR se siguen recibiendo los demas paquetes y solo se retransmite el que no llego/se pierdio la confirmacion

5 - ¿como se cual es el numero total de secuencias de segmentos disponibles? Es la cantidad de segmentos totales que tengo que enviar

6 - Revisar → Prestar atencion a los ack y seg de ambos extremos

7 - Revisar → Bien

12 - ¿Como se cuantas comunicaciones tengo en una captura wireshark? → Ni idea pero comunicaciones como tal en UDP no tengo

13 - Revisar → Oremos que este bien

## Ejercicio 1 ✅

¿Cuál es el puerto por defecto que se utiliza en los siguientes servicios?
Web / SSH / DNS / Web Seguro / POP3 / IMAP / SMTP
Investigue en qué lugar en Linux y en Windows está descrita la asociación utilizada por
defecto para cada servicio.

Web (entiendo por web a HTTPS) → 80

SSH → 22

DNS → 53

Web seguro (entiendo por web a HTTPS) → 443

POP3 → 110

IMAP → 143

SMTP → 25

Linux → /etc/services

Windows → c/windows/system32/drivers/etc/servers

## Ejercicio 2 ✅

Investigue qué es multicast. ¿Sobre cuál de los protocolos de capa de transporte funciona? ¿Se podría adaptar para que funcione sobre el otro protocolo de capa de transporte? ¿Por qué?

El **multicast** es una tecnica de transmisión de datos que permite enviar información a un grupo determinado de destinatarios (dispositivos), en lugar de **broadcast** que trasmite datos a todos los dispositivos, o **unicast** que transmite datos a un único dispositivo. Los routers de la red se encargan de **reenviar los paquetes solo a las interfaces** que tienen miembros interesados, optimizando el uso del ancho de banda.

Funciona sobre UDP ya que permite la trasmisión sin conexión y tambien debido a su simplicidad y bajo overhead.
Se podría adaptar para TCP pero no es eficiente, ya que TCP es orientado a conexion, por lo tanto
necesita conectarse con el host destino (1:1) y en multicast debería conectarse a varios hosts del grupo (1:Grupo).
Supongamos que un emisor multicast envía un paquete a 1000 receptores. Si un solo paquete se pierde,
cada receptor que no lo reciba podría enviar un ACK negativo solicitando la retransmisión. Esto genera → **ACK Implosión y Retransmisión ineficiente**

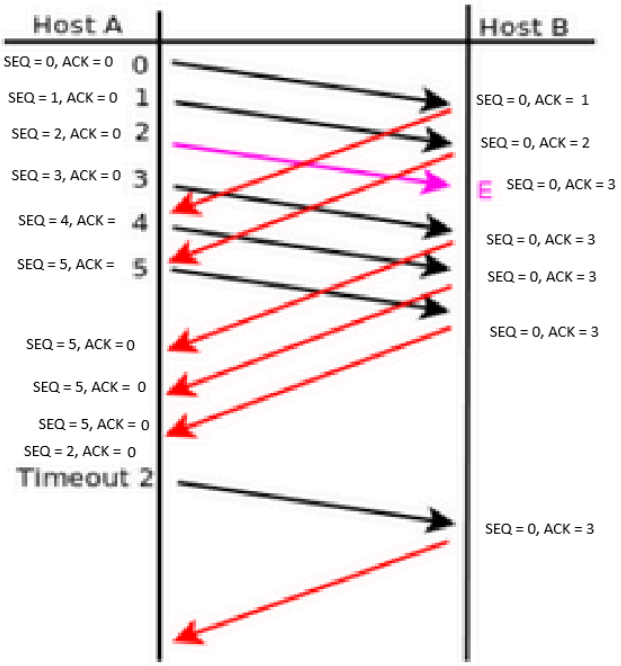
Ademas TCP ajusta la velocidad de transmisión según el receptor más lento (Ventana de Recepción).
En multicast,
cada receptor puede tener **diferentes capacidades de red**. Si TCP aplicara control de flujo para el receptor más lento, **todos los demás se verían afectados**.

## Ejercicio 3

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 elmodo pasivo ¿En qué se diferencian estos tipos de comunicaciones del resto de los protocolos de aplicación vistos?

La diferencia entre FTP y los demás protocolos de la capa de aplicación es que FTP pose 2 conexiones, una de control y otra de datos, los demás protocolos aplican estos mismos conceptos pero sobre la misma conexión.

## Ejercicio 4

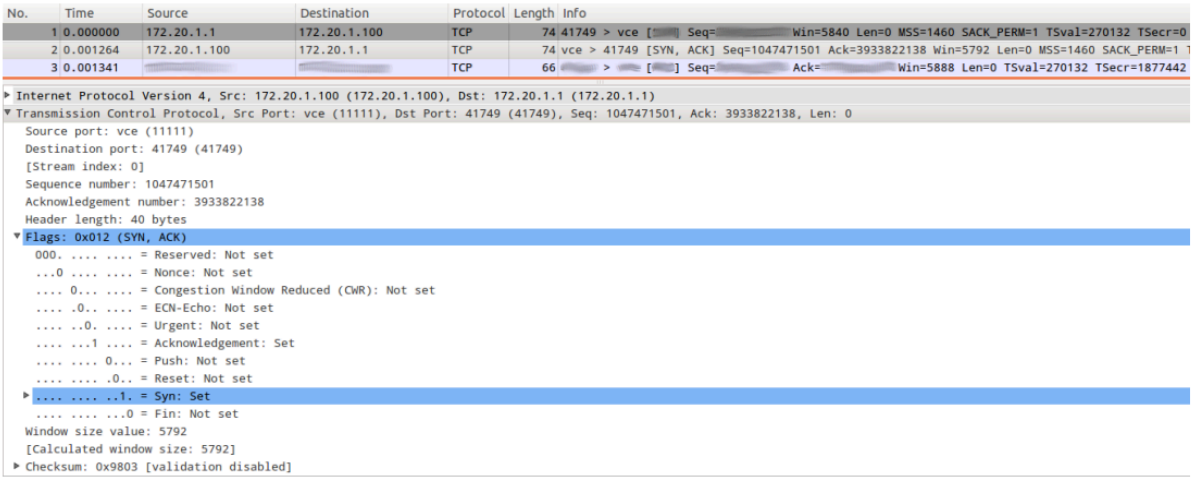



## Ejercicio 5

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

El tamaño de la ventana W debe ser menor que la mitad del numero  numero total de secuencias de segmentos M disponibles, es decir W<=M - 1/2

## Ejercicio 6

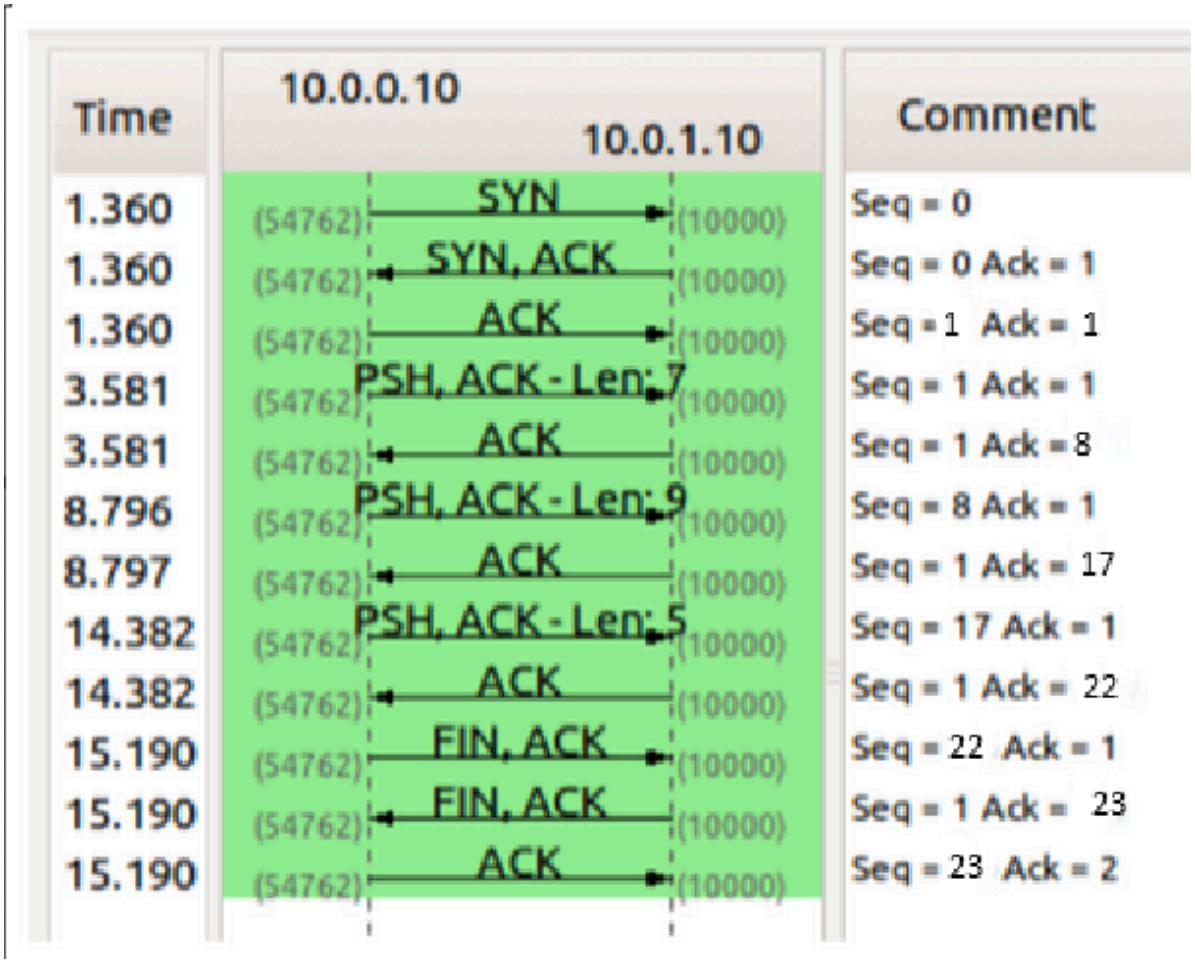



De acuerdo a la captura TCP de la siguiente figura, indique los valores de los campos
borroneados.

| Source | Destination | Info |
|---|---|---|
| 172.20.1.1 | 172.20.1.100 | [SYN] Seq = 1047471500 |
| 172.20.1.100 | 172.20.1.1 | [SYN, ACK] Seq =1047471501 |
| 172.20.1.1 | 172.20.1.100 | 41749 > vce [ACK] Seq = 1047471502 |

## Ejercicio 7

Dada la sesión TCP de la figura, completar los valores marcados con un signo de interrogación.




## Ejercicio 8 ✅

¿Qué es el RTT y cómo se calcula? Investigue la opción TCP timestamp y los campos
TSval y TSecr.

El RTT (Round Time Trip) es el tiempo que tarda un mensaje en ser enviado por un Emisor, que llegue al Receptor y vuelva al Emisor, es el tiempo de ida y vuelta de un paquete.
Este nos permite evaluar la letencia en la red, ajustar tiempos de retransmisión de paquetes y optimizar el rendimiento en la transmisión.
Se calcula de la siguiente forma: Tiempo_Recepcion - Tiempo_Envio

El TCP Timestamp se encuentra en el apartado Options de un segmento TCP y consiste en una opcion para medir el RTT de una manera mas precisa. Esta opcion es util para evitar suposiciones erroneas, mejora la fiabilidad y el rendimiento del protocolo TCP.

SACA UNA “FOTO” QUE SE COPIA CUANDO SE ENVIA EL SEGMENTO Y DE ESTA FORMA SIMPLIFICA LA COMPARACION, O SEA ES PARTE DE UN

CONTADOR PARA TODOS LOS SEGMENTOS, ESTA OPCION ES MAS SIMPLE QUE IMPLEMENTAR UN CONTADOR PARA CADA UNA DE LAS TRANSMISIONES DE PAQUETE

Timestamp posee 2 campos, TSval y TSecr
TSval (Timestamp Value): Es el tiempo actual en el emisor en el momento de enviar el segmento, se incluye en el encabezado TCP cuando se envía un segmento y contiene el tiempo actual

TSecr (Timestamp Echo Replay): Cuando el receptor recibe un segmento TSVal, incluye ese mismo valor en el campo TSecr cuando envia un ACK de vuelta al emisor

Ejemplo

**Envío del segmento desde el emisor:**

- El emisor envía un segmento TCP con un valor **TSval**.
- Este valor representa el **tiempo actual** en el emisor. Puede ser un contador de ticks, un temporizador en milisegundos, etc.
- En este momento, el campo **TSecr** está vacío o sin valor significativo, ya que es un segmento de datos, no un ACK.

**Ejemplo:**

- TSval = 100
- TSecr = (vacío o sin valor)

---

**Recepción del segmento en el receptor:**

- El receptor recibe el segmento TCP con el **TSval** del emisor.
- El receptor **no modifica el TSval**. Solo lo guarda temporalmente.

---

**Envío del ACK desde el receptor:**

- Cuando el receptor envía un **ACK**, incluye en el campo **TSecr** el valor **TSval** que recibió del emisor.
- En otras palabras, **TSecr es una copia exacta del TSval recibido previamente**.
- Además, el receptor también envía su propio **TSval**, que representa el tiempo actual en el receptor al momento de enviar el ACK.

**Ejemplo:**

- TSval = 500 (tiempo actual del receptor)
- TSecr = 100 (copia del TSval recibido previamente del emisor)

## Ejercicio 9 ✅

Para la captura tcp-captura.pcap, responder las siguientes preguntas.
a. ¿Cuántos intentos de conexiones TCP hay?
b. ¿Cuáles son la fuente y el destino (IP:port) para c/u?
c. ¿Cuántas conexiones TCP exitosas hay en la captura? ¿Cómo diferencia las exitosas de las que no lo son? ¿Cuáles flags encuentra en cada una?
d. Dada la primera conexión exitosa responder:
i. ¿Quién inicia la conexión?
ii. ¿Quién es el servidor y quién el cliente?
iii. ¿En qué segmentos se ve el 3-way handshake?
iv. ¿Cuáles ISNs se intercambian?
v. ¿Cuál MSS se negoció?
vi. ¿Cuál de los dos hosts envía la mayor cantidad de datos (IP:port)?

e. Identificar primer segmento de datos (origen, destino, tiempo, número de fila y número de secuencia TCP).
i. ¿Cuántos datos lleva?
ii. ¿Cuándo es confirmado (tiempo, número de fila y número de secuencia TCP)?
iii. La confirmación, ¿qué cantidad de bytes confirma?
f. ¿Quién inicia el cierre de la conexión? ¿Qué flags se utilizan? ¿En cuáles segmentos se ve (tiempo, número de fila y número de secuencia TCP)?

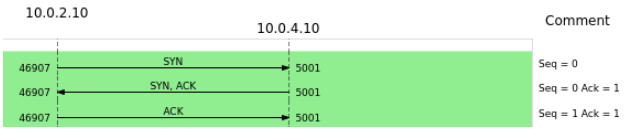
a. 2 intentos de conexiones TCP

b.

| Conexion | Origen | Destino |
|---|---|---|
| 1 | 10.0.2.10:46907 | 10.0.4.10:5001 |
| 2 (intento 1) | 10.0.2.10:45670 | 10.0.4.10:7002 |
| 3 (intento 2) | 10.0.2.10:45671 | 10.0.4.10:7002 |
| 4 | 10.0.2.10:46910 | 10.0.4.10:5001 |
| 5 | 10.0.2.10:54424 | 10.0.4.10:9000 |
| 6 | 10.0.2.10:54425 | 10.0.4.10:9000 |

c. Conexiones exitosas son las que completaron el 3 way handshake correctamente (es decir SYN, SYN-ACK,ACK) , estas son 4
Mientras que conexiones fallidas son 2, podemos identificarlas por el flag RST

d.
i. Inicia la conexion el host 10.0.2.10 en el puerto 46907
ii. 10.0.2.10 : 46907 Cliente - 10.0.4.10 : 5001 Servidor
iii.




iv.
SYN → 221842854 (Sequence Number (raw))
SYN, ACK → 1292618479 (Sequence Number (raw))
ACK → 221842955 (Sequence Number (raw))

v. 1460 bytes
vi. 10.0.2.10:46907 envio 786458 bytes

e. El primer envio de datos se hace desde 10.0.2.10:46907 a 10.0.4.10:5001, fila 4, tiempo 0.151826 num de secuencia TCP 6

i. lleva 24 bytes
ii. Es confirmado en el numero de secuencia 7 tiempo 0.151925 y numero de fila 5
iii. Confirma 25 bytes indicando el proximo valor esperado por el emisor.
iv. El cierre es iniciado por el host 10.0.2.10:46907 en el numero de secuencia 958, fila 955, tiempo 76,090196 y dura hasta el numero de secuencia 960, fila 957, tiempo 75.247457, y se envian los flags de FIN ,PSH, ACK al receptor, el receptor contesta con FIN, ACK para confirmar el cierre de su parte, y por ultimo el host que inicio el cierre envia el ACK final.

## Ejercicio 10 ✅

Responda las siguientes preguntas respecto del mecanismo de control de flujo.
a. ¿Quién lo activa? ¿De qué forma lo hace?
b. ¿Qué problema resuelve?
c. ¿Cuánto tiempo dura activo y qué situación lo desactiva?

a. El control de flujo lo activa el RECEPTOR cuando detecta que esta recibiendo mas datos de los que puede procesar este ajusta su ventana de recepción dinámicamente y de esta forma limita la cantidad de bytes que puede recibir del EMISOR, este ultimo lee el valor de la ventana y envia la cantidad de paquetes acorde a la ventana.

b. Resuelve el problema de la saturacion de los buffers del receptor.

c. El control en si, esta activo durante toda la conexión TCP, solamente se ajusta dinámicamente a medida que el RECEPTOR se queda sin espacio en el buffer para nuevos paquetes. Puede pasar que si esta ventana tiende a 0 el EMISOR, durante un periodo de tiempo, no envía paquetes hasta que el buffer del RECEPTOR no se libere

## Ejercicio 11 ✅

Responda las siguientes preguntas respecto del mecanismo de control de congestión.
a. ¿Quién activa el mecanismo de control de congestión? ¿Cuáles son los posibles
disparadores?
b. ¿Qué problema resuelve?
c. Diferencie slow start de congestion-avoidance.

a. El control de congestión es activado por el emisor al detectar 3 ACK duplicados indicando
pérdida de un segmento intermedio) o la expiración de un temporizador RTO (Timeout) sin haber recibido una confirmación (ACK).

b. El control de congestión resuelve el problema de la congestión en la red, ajustando dinámicamente la cantidad de datos que se pueden enviar para no saturar la red.

c. Slow Start es la primera fase del control de congestion y consiste en enviar una poca cantidad de datos con el fin de detectar cual es el limite de la red, es decir, el punto donde se produce la primera perdida de paquetes o ack duplicados. En esta fase la ventana de congestion o CongWin es pequeña, de 1 MSS (Maximum Segment Size) y va incrementándose con cada ACK recibido.
Congestion Avoidance es la segunda etapa que se dispara cuando Slow Start supera al limite (ssthresh), consiste en un creciemiento de la ventana

CongWin de manera lineal (mientras que en Slow Start es una fase exponencial) donde el objetivo no es ver un limite si no evitar congestionar la red.

## Ejercicio 12 ✅

Para la captura udp-captura.pcap, responder las siguientes preguntas.
a. ¿Cuántas comunicaciones (srcIP,srcPort,dstIP,dstPort) UDP hay en la captura?
b. ¿Cómo se podrían identificar las exitosas de las que no lo son?
c. ¿UDP puede utilizar el modelo cliente/servidor?
d. ¿Qué servicios o aplicaciones suelen utilizar este protocolo?¿Qué requerimientos tienen?
e. ¿Qué hace el protocolo UDP en relación al control de errores?
f. Con respecto a los puertos vistos en las capturas, ¿observa algo particular que lo diferencie de TCP?
g. Dada la primera comunicación en la cual se ven datos en ambos sentidos (identificar el primer datagrama):
i. ¿Cuál es la dirección IP que envía el primer datagrama?,¿desde cuál puerto?
ii. ¿Cuántos datos se envían en un sentido y en el otro?

a. 9 comunicaciones totales en UDP (sacado mediante un filtro de wireshark)
b. No podemos diferencia comunicaciones a simple vista ya que no tenemos establecimientos de conexion ni cierres, lo que si podemos identificar es cuando un datagrama udp se pierde, mostrando los mensajes ICMP como “Port Unreachable” o Time to Live Exceded”
c. Sí, **UDP (User Datagram Protocol)** puede utilizar el modelo **cliente/servidor**. Aunque UDP no tiene la misma fiabilidad que TCP, se puede usar perfectamente en aplicaciones que implementen este tipo de modelo de comunicación
d. Aplicaciones que priorizan mucha cantidad de mensajes por segundo por sobre la fidelidad o la confirmación de llegada de los mismos, es decir, si llegan o no no es un problema relevante.
e. En principio UDP no, pero si el protocolo ICMP el cual puede contestar situaciones como cuando se envie un datagrama a un puerto donde no hay ningun servicio escuchando ICMP nos conteste con un Port Unrechable o un Time to Live Exceded en el caso donde el tiempo de vida de un datagrama expira debido a que se perdió.

f. Son puertos muy especificos, el RTT es bajo por lo tanto hay mas cantidad de datagramas por segundo, (REVISAR)

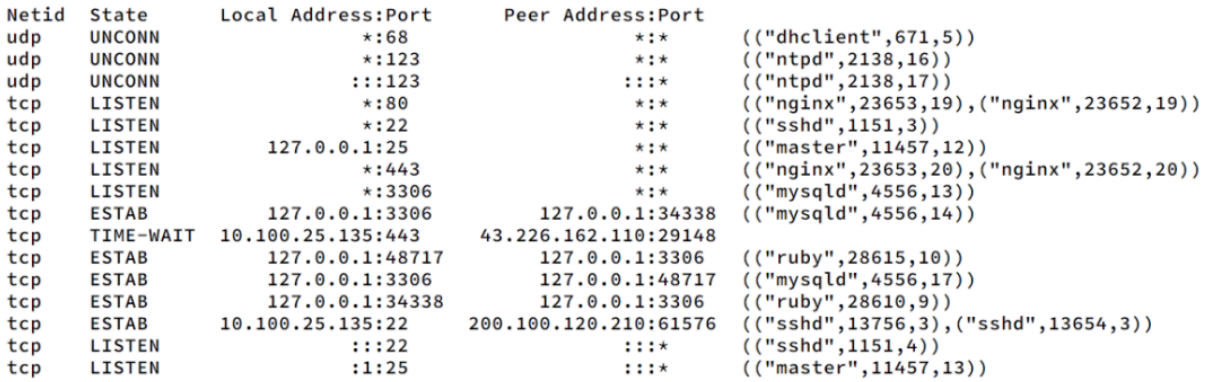
g.

i. 10.0.2.10:0

ii. Se envían 8 bytes de datos en un sentido y en el otro.

## Ejercicio 13

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
b. hping3 -S -p 25 10.100.25.135
c. hping3 -S -p 22 10.100.25.135
d. hping3 -S -p 110 10.100.25.135
● ¿Cuántas conexiones distintas hay establecidas? Justifique.

a. Envia un datagrama UDP al puerto 3306 → Responde con un mensaje ICMP puerto inalcanzable ya que no esta escuchando datagramas UDP

b. Envia un paquete con el flag SYN al puerto 25 → Responde con un RST indicando que no es posible establecer una conexion ya que el puerto 25 esta escuchando con el localhost ipv4 e ipv6 (loopback)

c. Envia un paquete con el flag SYN al puerto 22 → Responde con el flag SYN-ACK (SA) indicando que recibio el mensaje SYN y tambien esta

disponible para aceptar conexiones luego el proceso del 3 Way Handshake continua

d. Envia un paquete con el flag SYN al puerto 110 (POP3), este no figura en la tabla por lo tanto probablemente obtengamos un flag RESET (R) como respuesta.

hay 4 conexiones distintas establecidas. hay 2 conexiones particulares

127.0.0.1:48717 → 127.0.0.1:3306
127.0.0.1:3306→ 127.0.0.1:48717

Si bien ambas conexiones estan dentro del mismo host, las conexiones SON distintas ya que los puertos son diferentes

NOTAS DEL EJERCICIO

si el puerto aparece junto al Local Address significa que esta escuchando en ese puerto

mientras si que si el puerto aparece junto al Peer Address significa que esta enviando/recibiendo datos a ese puerto

si envias un datagrama a un puerto que solo esta escuchando TCP te contesta un mensaje ICMP diciendo puerto inalcanzable

UNCONN es como muestran los puertos abiertos que escuchan UDP

si aparece un “:::22” es una forma abreviada de IPV6 0:0:0:0:0:0:0 es igual a “*:22” en IPV4

si aparece un “:1:25” es una forma equivalente a 127.0.0.1:25

El
**ACK flag** se envía con cada segmento de datos para **mantener la sincronización** de la conexión y confirmar la recepción. (PSH, ACK - Len 24)