Práctica 6 - Redes y Comunicaciones

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.**

Protocolo	Puerto
Web HTTP	80
SSH	22
DNS	53
Web Seguro HTTPS	443
POP3	110
IMAP	143
SMTP	25

- **Ubicación de la configuración en Linux:**
  - En Linux la configuración está en el archivo /etc/services aunque servicios como SSH tienen archivos de configuración específicos, por ejemplo, /etc/ssh/sshd_config para SSH.
- **Ubicación de la configuración en Windows:**
  - En Windows la configuración está en el archivo C:\Windows\System32\drivers\etc\services.

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?**

- **Multicast** es un método de transmisión de datos en redes donde un solo paquete se envía a múltiples destinatarios interesados de manera simultánea. A diferencia de unicast (un solo emisor a un solo receptor) y broadcast (un emisor a todos los dispositivos en una red), multicast permite que la información se distribuya sólo a los dispositivos que han expresado interés en recibirla.
  Multicast funciona principalmente sobre **UDP (User Datagram Protocol)**. La razón es que UDP es un protocolo sin conexión y de bajo costo computacional, lo que facilita la transmisión de paquetes a múltiples destinatarios sin establecer conexiones individuales con cada uno.
  No es práctico ni común adaptar multicast para funcionar sobre **TCP (Transmission Control Protocol)** debido a las siguientes razones:

- **Conexiones individuales:** TCP es un protocolo orientado a conexión, lo que significa que establece una conexión punto a punto entre el emisor y cada receptor. Implementar multicast sobre TCP requeriría establecer una conexión separada con cada destinatario, lo que anula la eficiencia de multicast.
- **Control de flujo y congestión:** TCP proporciona confiabilidad, control de flujo y detección de congestión. Sin embargo, en un escenario multicast, gestionar estas características para múltiples destinatarios con diferentes velocidades y capacidades sería muy complejo. Si un destinatario tiene problemas de red, retrasaría la transmisión para todos los demás.
- **Garantía de entrega:** La naturaleza de TCP implica la necesidad de confirmar la recepción de paquetes. Si varios receptores tuvieran diferentes latencias o pérdidas de paquetes, coordinar las retransmisiones y confirmaciones se volvería extremadamente complicado.

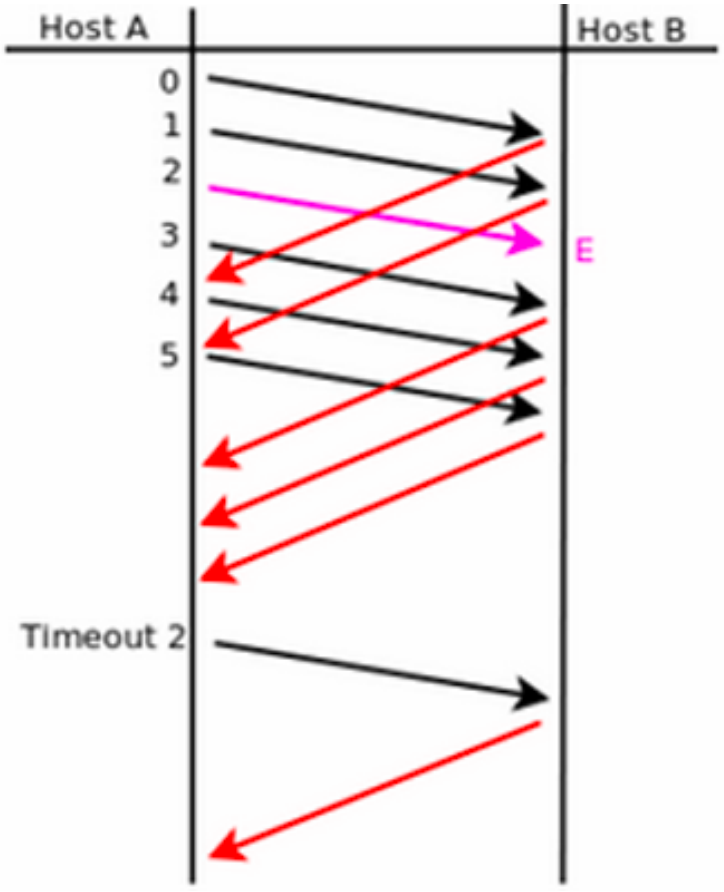
3. **Investigue cómo funciona el protocolo de aplicación FTP teniendo en cuenta las diferencias en su funcionamiento cuando se utiliza el modo activo de cuando se utiliza el modo pasivo ¿En qué se diferencian estos tipos de comunicaciones del resto de los protocolos de aplicación vistos?**

- El **File Transfer Protocol (FTP)** es un protocolo de aplicación utilizado para transferir archivos entre un cliente y un servidor en una red TCP/IP. FTP opera en la capa de aplicación del modelo OSI y utiliza el **puerto 21** para el control de comandos y el **puerto 20** para la transferencia de datos en modo activo.
    - **Modo Activo:**
        - **Conexión de control:** El cliente inicia una conexión TCP desde un puerto efímero (por ejemplo, 1025) al puerto 21 del servidor FTP. Esta conexión se utiliza para enviar comandos.
        - **Conexión de datos:** Cuando el cliente solicita una transferencia de datos (como listar directorios o descargar un archivo), el servidor inicia una conexión TCP desde su puerto 20 al puerto efímero del cliente. **Importante:** El cliente debe permitir conexiones entrantes en este puerto, lo que puede ser problemático si hay un firewall o NAT que bloquea conexiones entrantes no solicitadas.
    - **Modo Pasivo:**
        - **Conexión de control:** El cliente establece la conexión de control desde un puerto efímero al puerto 21 del servidor FTP, como en el modo activo.
        - **Conexión de datos:** El cliente envía un comando PASV al servidor. El servidor responde con un puerto de datos (generalmente dinámico, en el rango 1024-65535). El cliente inicia una conexión desde un puerto efímero al puerto indicado por el servidor.
    - **Diferencias con otros protocolos de aplicación**
        - **Conexiones múltiples:** FTP requiere dos conexiones separadas: una para el control y otra para la transferencia de datos. La mayoría de los

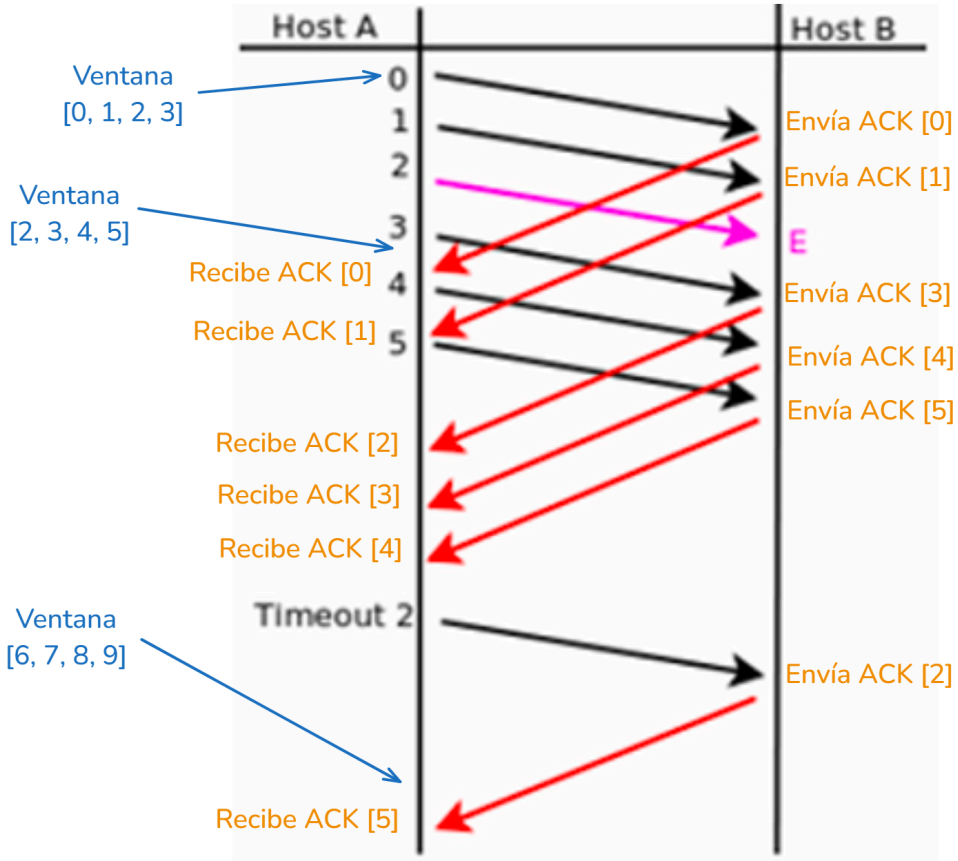
protocolos de aplicación (como HTTP, SMTP, o IMAP) utilizan una sola conexión TCP para controlar y transferir datos.

- **Modo activo vs. pasivo:** FTP es uno de los pocos protocolos que ofrece estos dos modos distintos para la gestión de conexiones, lo que es necesario debido a las restricciones de red y seguridad modernas (como firewalls y NATs).
- **Estado de la conexión:** FTP es un protocolo con estado (stateful), lo que significa que mantiene una sesión activa entre el cliente y el servidor durante la transferencia de archivos. En contraste, protocolos como HTTP son sin estado (stateless) y cada solicitud es independiente.

4. **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.**



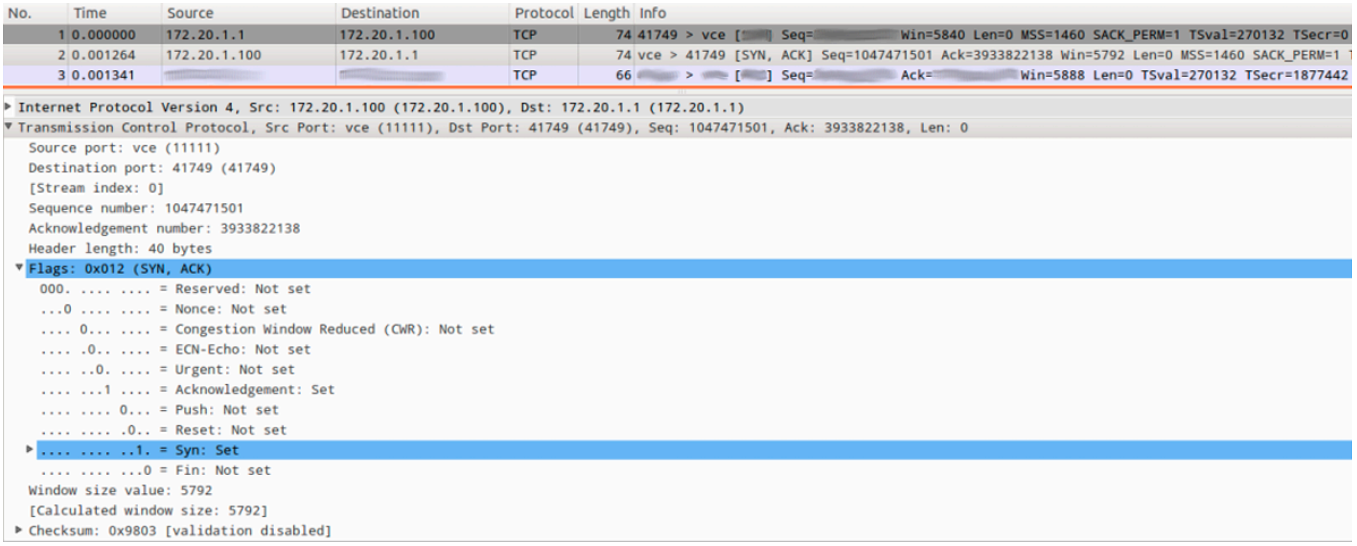
- **Resolución:**



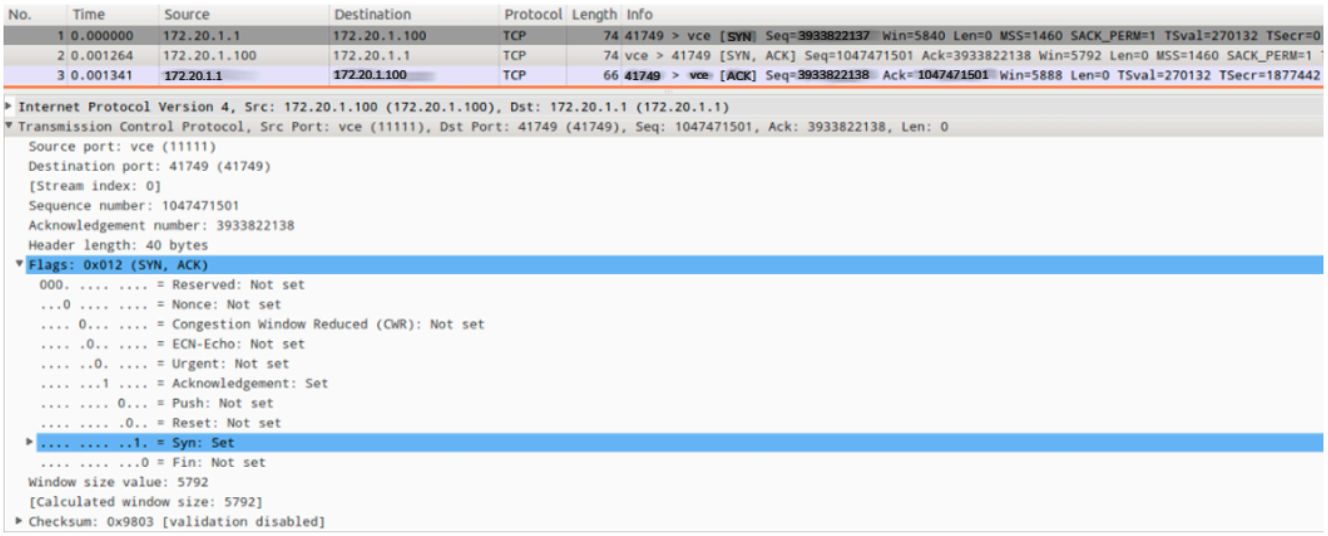
5. **¿Qué restricción existe sobre el tamaño de ventanas en el protocolo Selective Repeat?**

- El tamaño máximo de la ventana de transmisión (W) debe cumplir la siguiente condición: **W <= <(N / 2)**. Donde:
    - **W** es el tamaño de la ventana (tanto de envío como de recepción).
    - **N** es el rango total de números de secuencia disponibles.

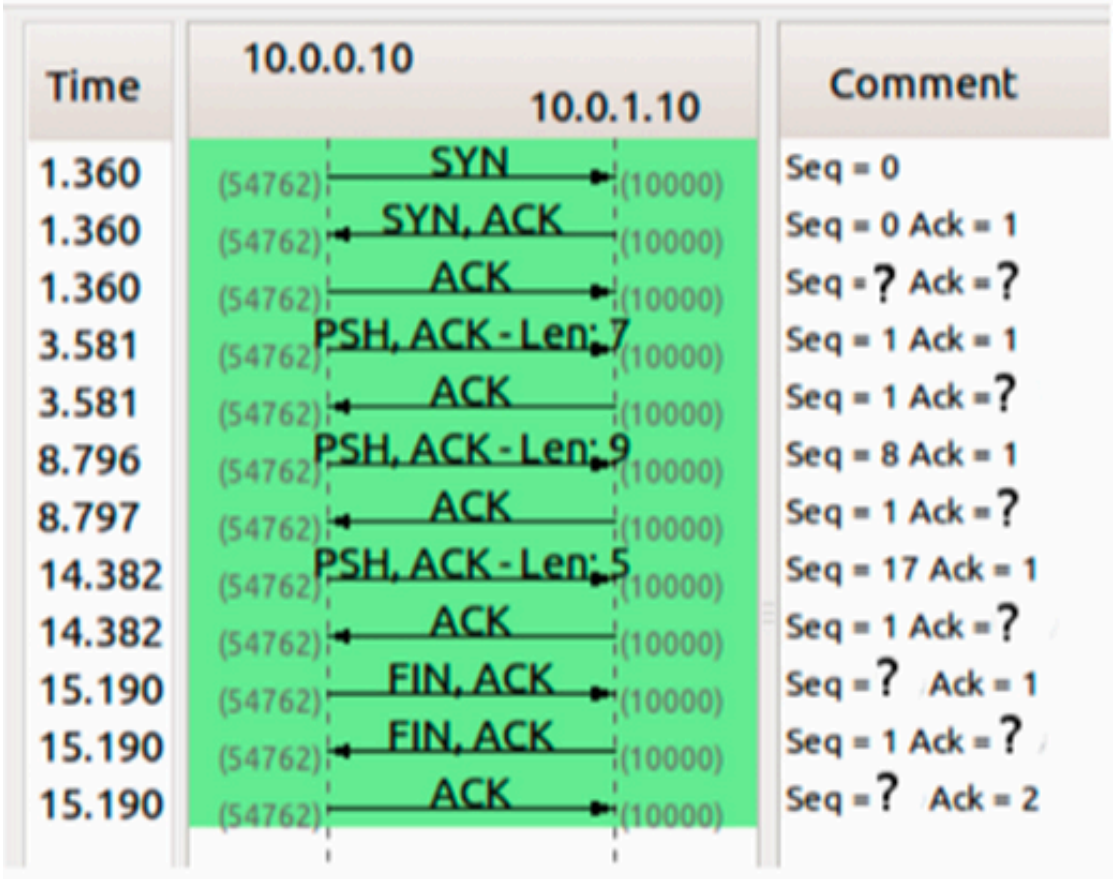
6. **De acuerdo a la captura TCP de la siguiente figura, indique los valores de los campos borroneados.**



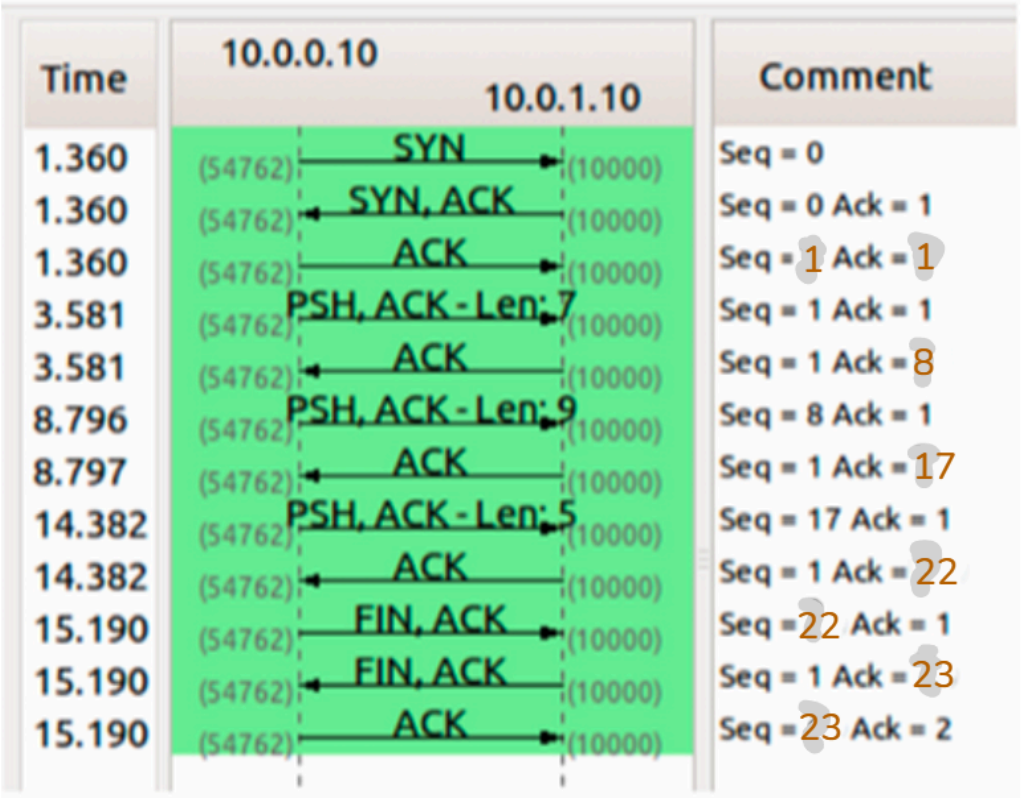
- **Resolución:**



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


Time
10.0.0.10
10.0.1.10
Comment
1.360 (54762) SYN (10000) Seq = 0
1.360 (54762) SYN, ACK (10000) Seq = 0 Ack = 1
1.360 (54762) ACK (10000) Seq = ? Ack = ?
3.581 (54762) PSH, ACK - Len: 7 (10000) Seq = 1 Ack = 1
3.581 (54762) ACK (10000) Seq = 1 Ack = ?
8.796 (54762) PSH, ACK - Len: 9 (10000) Seq = 8 Ack = 1
8.797 (54762) ACK (10000) Seq = 1 Ack = ?
14.382 (54762) PSH, ACK - Len: 5 (10000) Seq = 17 Ack = 1
14.382 (54762) ACK (10000) Seq = 1 Ack = ?
15.190 (54762) FIN, ACK (10000) Seq = ? Ack = 1
15.190 (54762) FIN, ACK (10000) Seq = 1 Ack = ?
15.190 (54762) ACK (10000) Seq = ? Ack = 2

- **Resolución:**


Time
10.0.0.10
10.0.1.10
Comment
1.360 (54762) SYN (10000) Seq = 0
1.360 (54762) SYN, ACK (10000) Seq = 0 Ack = 1
1.360 (54762) ACK (10000) Seq = 1 Ack = 1
3.581 (54762) PSH, ACK - Len: 7 (10000) Seq = 1 Ack = 1
3.581 (54762) ACK (10000) Seq = 1 Ack = 8
8.796 (54762) PSH, ACK - Len: 9 (10000) Seq = 8 Ack = 1
8.797 (54762) ACK (10000) Seq = 1 Ack = 17
14.382 (54762) PSH, ACK - Len: 5 (10000) Seq = 17 Ack = 1
14.382 (54762) ACK (10000) Seq = 1 Ack = 22
15.190 (54762) FIN, ACK (10000) Seq = 22 Ack = 1
15.190 (54762) FIN, ACK (10000) Seq = 1 Ack = 23
15.190 (54762) ACK (10000) Seq = 23 Ack = 2

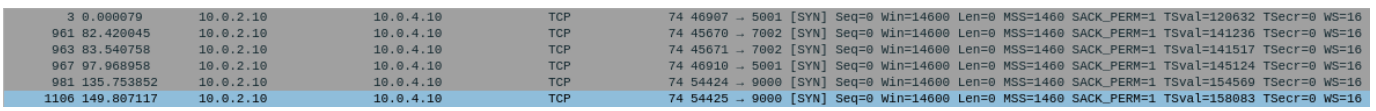
8. **¿Qué es el RTT y cómo se calcula? Investigue la opción TCP timestamp y los campos TSval y TSecr.**

- **RTT (Round Trip Time):**
    - Representa el tiempo que tarda un paquete de datos en viajar desde el emisor hasta el receptor y volver con la confirmación (ACK). En otras palabras, es la **suma del tiempo de transmisión, el tiempo de propagación en la red y el tiempo de procesamiento en el receptor.** Se calcula a medida que se envían datos y se reciben ACK.
    - Es un factor **fundamental** para **determinar el valor del RTO.**
    - **Un RTT alto** puede afectar negativamente el rendimiento del control de errores, especialmente en conexiones de alta velocidad y con un gran ancho de banda.
    - **Cálculo del RTT:**
        - **Envío del paquete:** Se registra el momento en que el emisor envía un paquete.
        - **Recepción del acuse de recibo (ACK):** Se registra el momento en que el emisor recibe la respuesta del receptor.
        - **Fórmula:** RTT = Tiempo de recepción del ACK - Tiempo del envío del paquete.
- **Opción con TimeStamps:**
    - La opción de TimeStamp permite una **estimación más precisa del RTT al incluir marcas de tiempo en los segmentos TCP (Timestamp Value TSval).**
        - Los valores **TSval** se repiten en el lado opuesto de la conexión en el campo **Timestamp Echo Reply TSecr** . Entonces, cuando se confirma un segmento, el remitente de ese segmento puede simplemente restar su marca de tiempo actual del valor **TSecr** para **calcular una medición precisa del tiempo de ida y vuelta (RTT) → RTT = TSecr - TSval.**
    - **Beneficios:**
        - Cálculo de RTT sin temporizadores individuales por segmento.
        - Protección contra Wraparounds de números de secuencia (PAWS).

9. **Para la captura tcp-captura.pcap, responder las siguientes preguntas.**

    a. **¿Cuántos intentos de conexiones TCP hay?**

    - Hay 6 intentos de conexiones TCP:

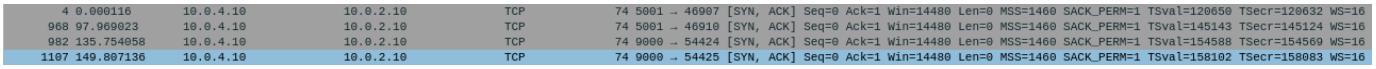


3	0.000079	10.0.2.10	10.0.4.10	TCP	74 46907 → 5001 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=120632 TSecr=0 WS=16
961	82.420045	10.0.2.10	10.0.4.10	TCP	74 45670 → 7002 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=141236 TSecr=0 WS=16
963	83.540758	10.0.2.10	10.0.4.10	TCP	74 45671 → 7002 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=141517 TSecr=0 WS=16
967	97.968958	10.0.2.10	10.0.4.10	TCP	74 46910 → 5001 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=145124 TSecr=0 WS=16
981	135.753852	10.0.2.10	10.0.4.10	TCP	74 54424 → 9000 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=154569 TSecr=0 WS=16
1106	149.807117	10.0.2.10	10.0.4.10	TCP	74 54425 → 9000 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=158083 TSecr=0 WS=16

    b. **¿Cuáles son la fuente y el destino (IP:port) para c/u?**

(IP:port) fuente	(IP:port) destino
10.0.2.10:46907	10.0.4.10:5001
10.0.2.10:45670	10.0.4.10:7002
10.0.2.10:45671	10.0.4.10:7002
10.0.2.10:46910	10.0.4.10:5001
10.0.2.10:54424	10.0.4.10:9000
10.0.2.10:54425	10.0.4.10:9000

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?



Las exitosas tienen los flags SYN/ACK en 1, las fallidas tienen los flags RST/ACK en 1

d. Dada la primera conexión exitosa responder:

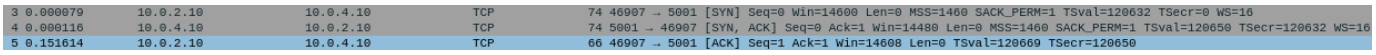
i. ¿Quién inicia la conexión?

- La conexión es iniciada por 10.0.2.10:46907.

ii. ¿Quién es el servidor y quién el cliente?

- **Servidor:** 10.0.4.10:5001.
- **Cliente:** 10.0.2.10:46907.

iii. ¿En qué segmentos se ve el 3-way handshake?



iv. ¿Cuáles ISNs se intercambian?

- **Segmento 3:**
  - **ISN Relativo:** 0.
  - **ISN raw:** 2218428254.
- **Segmento 4:**
  - **ISN Relativo:** 0.
  - **ISN raw:** 1292618479.
- **Segmento 5:**
  - **ISN Relativo:** 1.
  - **ISN raw:** 2218428255.

v. **¿Cuál MSS se negoció?**

- Se negoció como **MSS** 1460.

vi. **¿Cuál de los dos hosts envía la mayor cantidad de datos (IP:port)?**

- El host que más envía datos es 10.0.2.10:46907, esto lo podemos ver si vamos examinado los números de secuencia de los segmentos que envía, este número va creciendo muchísimo a lo largo de la comunicación.

e. **Identificar el primer segmento de datos (origen, destino, tiempo, número de fila y número de secuencia TCP).**

i. **¿Cuántos datos lleva?**

- Lleva 24 bytes de datos, se ve en el Len.

ii. **¿Cuándo es confirmado (tiempo, número de fila y número de secuencia TCP)?**

iii. **La confirmación, ¿qué cantidad de bytes confirma?**

- Se confirman 24 bytes, se espera el 25.

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)?**

10. Responda las siguientes preguntas respecto del mecanismo de control de flujo.

a. **¿Quién lo activa? ¿De qué forma lo hace?**

- El control de flujo lo activa el receptor enviando ventanas más chicas. Esto deja en evidencia que el receptor tiene poco espacio (o no tiene más lugar) para seguir recibiendo datos. Esto se realiza a través del campo de tamaño de ventana en los encabezados de los segmentos TCP.

b. **¿Qué problema resuelve?**

- El control de flujo resuelve el problema de sobrecarga del receptor de modo que este no sobrecargue su buffer y se pierdan datos. Sin este mecanismo:
    - El emisor podría enviar datos a una velocidad superior a la que el receptor puede procesar.
    - Esto llevaría a la pérdida de datos o a un mal funcionamiento del receptor, especialmente si sus búferes se llenan.

c. ¿Cuánto tiempo dura activo y qué situación lo desactiva?

- **Duración:** El control de flujo permanece activo durante toda la duración de la conexión, ya que la capacidad del receptor puede cambiar dinámicamente.
- **Desactivación:** Se "desactiva" temporalmente cuando el receptor indica que tiene suficiente espacio en sus búferes, aumentando la ventana de recepción (RWND). Esto permite al emisor reanudar el envío de datos a una velocidad normal o ajustada.

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?

- El mecanismo de control de congestión es **activado por el emisor** (en TCP) y se basa en la detección de posibles problemas de congestión en la red.
  Posibles disparadores:
    - **Fin de Temporización:** La expiración del temporizador asociado con el envío de un segmento TCP puede ser interpretada como una señal de pérdida, indicando posiblemente congestión en la ruta.
    - **Recepción de TRES ACK Duplicados:** La recepción de paquetes ACK duplicados procedentes del receptor también se interpreta como un suceso de pérdida. Este evento puede sugerir la pérdida de un paquete en la red debido a la congestión.

b. ¿Qué problema resuelve?

- El control de congestión resuelve el problema de **sobrecarga de la red**, donde múltiples transmisiones simultáneas pueden saturar los recursos compartidos (routers, enlaces, etc.). Sin este mecanismo:
    - Los routers pueden saturarse, causando la pérdida de paquetes.
    - Se produce una disminución en el rendimiento general de la red.
    - La transmisión de datos puede volverse ineficiente e inestable.

c. Diferencie slow start de congestion-avoidance.

Característica	Slow Start	Congestion Avoidance

Objetivo	Descubrir rápidamente la capacidad de la red	Mantener un flujo estable y evitar congestión
Crecimiento de cwnd	Exponencial (duplica cada RTT)	Lineal (incrementa gradualmente)
Inicio	Desde el inicio de la conexión o después de una pérdida (Expira RTO)	Después de alcanzar el umbral ssthresh o recepción de 3 ACK's duplicados
Finalización	Al alcanzar el ssthresh o detectar congestión	Continúa hasta detectar congestión

12. Para la captura udp-captura.pcap, responder las siguientes preguntas.

a. ¿Cuántas comunicaciones (srcIP,srcPort,dstIP,dstPort) UDP hay en la captura?

- En un principio son 9, pero hay algunas que se tratan de la misma conversación, quedando 6:

Fuente	Destino
10.0.2.10:0	10.0.30.10:8003
10.0.2.10:9004	10.0.3.10:9045
10.0.2.10:9004	1.1.1.1:9045
10.0.2.10:53300	10.0.4.10:9045
10.0.2.10:59053	10.0.4.10:8003
10.0.2.10:8003	10.0.4.10:8003

b. ¿Cómo se podrían identificar las exitosas de las que no lo son?

- Se pueden diferenciar con los mensajes ICMP, estos se envían a las no exitosas.

c. ¿UDP puede utilizar el modelo cliente/servidor?

- Como no se establece una conexión, no sigue ningún modelo en particular, no tiene una estructura interna para definir roles específicos de cliente o servidor. Aun así, la mayoría de las aplicaciones que utilizan UDP suelen

adoptar un modelo cliente/servidor según las necesidades del servicio que están proporcionando.

d. ¿Qué servicios o aplicaciones suelen utilizar este protocolo?¿Qué requerimientos tienen?

- **Aplicaciones que Suelen Utilizar UDP**
    - **DNS (Domain Name System):** DNS utiliza UDP para las consultas de nombres de dominio a direcciones IP. Esta elección se debe a que las consultas DNS son generalmente pequeñas y no requieren la fiabilidad que ofrece TCP. Además, la velocidad es crucial en las consultas DNS, y el mecanismo de control de congestión de TCP podría agregar latencia.
    - **Streaming multimedia (audio y video):** Aplicaciones como la telefonía por Internet y la videoconferencia a menudo utilizan UDP debido a sus requisitos de baja latencia. Aunque la pérdida de algunos paquetes puede degradar la calidad, la retransmisión de los mismos (como en TCP) introduciría un retardo inaceptable. Sin embargo, en la Internet actual, el tráfico de voz y video se envía cada vez más a través de TCP debido a las políticas de gestión de tráfico de los ISP, que a menudo favorecen a TCP sobre UDP.
    - **Juegos en línea:** Los juegos en línea que requieren respuestas rápidas a menudo utilizan UDP para minimizar la latencia. La pérdida ocasional de paquetes es generalmente tolerable en este contexto.
    - **SNMP (Simple Network Management Protocol):** SNMP, un protocolo utilizado para la gestión de dispositivos de red, típicamente utiliza UDP para el envío de mensajes de control y monitoreo.
    - **Protocolos de enrutamiento:** Algunos protocolos de enrutamiento, como RIP (Routing Information Protocol), utilizan UDP para el intercambio de información de enrutamiento entre routers.
      Requerimientos de las Aplicaciones que Usan UDP
    - **Tolerancia a la pérdida de datos:** Las aplicaciones pueden tolerar cierta pérdida de paquetes sin comprometer su funcionalidad.
    - **Baja latencia:** La velocidad es crucial para estas aplicaciones, y la sobrecarga de una conexión orientada a la conexión como TCP sería perjudicial.
    - **Control de flujo a nivel de aplicación:** Si se necesita la fiabilidad, la aplicación debe implementar sus propios mecanismos de control de flujo y retransmisión de datos.

e. ¿Qué hace el protocolo UDP en relación al control de errores?

- El protocolo UDP (User Datagram Protocol) proporciona un mecanismo básico de detección de errores, pero no implementa mecanismos de recuperación.

f. Con respecto a los puertos vistos en las capturas, ¿observa algo particular que lo diferencie de TCP?

- En UDP el puerto origen puede ser 0 si no necesita una respuesta, puede ser simplemente un envío.

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?

- 10.0.2.1:9004.

ii. ¿Cuántos datos se envían en un sentido y en el otro?

- Se envían 9 y 12 bytes.

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))
```

a. 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.

i. hping3 -p 3306 –udp 10.100.25.135

- Te devuelve ICMP Port Unreachable porque el puerto está escuchando TCP.

ii. hping3 -S -p 25 10.100.25.135

- Te devuelve un segmento TCP RST/ACK ya no hay nadie escuchando en el puerto 25 con ip 10.100.25.135.

iii. **hping3 -S -p 22 10.100.25.135**

- Te devuelve un segmento TCP SYN/ACK ya que hay alguien escuchando en el puerto 22.

iv. **hping3 -S -p 110 10.100.25.135**

- Te devuelve un segmento TCP RST/ACK ya no hay nadie escuchando en el puerto 110.

b. ¿Cuántas conexiones distintas hay establecidas? Justifique.

- Hay 3 conexiones distintas establecidas ya que la primera y la segunda línea de las conexiones representan los dos extremos de la misma conexión local.