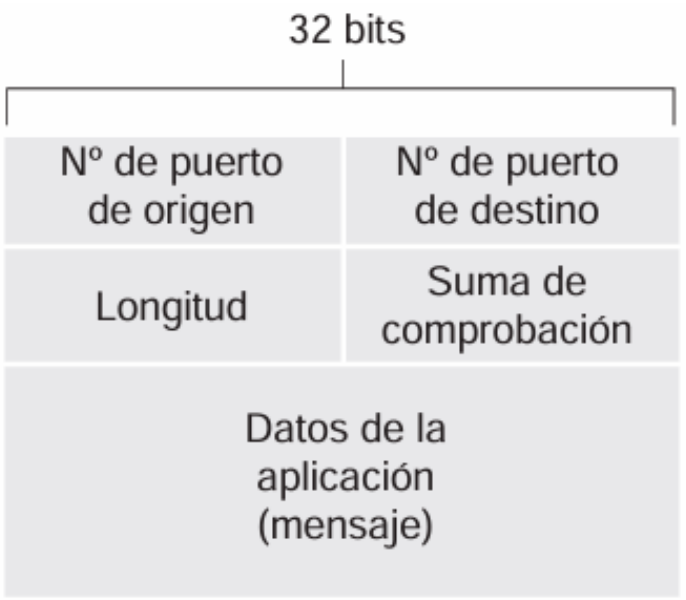
Práctica 5 - Redes y Comunicaciones

1. **¿Cuál es la función de la capa de transporte?**

- Multiplexación y Demultiplexación de conversaciones.
- Encapsulación:
    - Define PDU (Protocol Data Unit) donde se envían los mensajes de la aplicación.
- Soporte de datos de tamaños arbitrarios.
- Control y Detección de Errores (pérdida, duplicación, corrupción. etc).
- Control de Flujo y Control de Congestión (TCP).

2. **Describa la estructura del segmento TCP y UDP.**

- **Header UDP:**
    - La cabecera UDP solo tiene cuatro campos, cada uno con una longitud de dos bytes:
        - Números de puerto de origen y de destino: Permiten al host de destino pasar los datos de la aplicación al proceso apropiado que está ejecutándose en el sistema terminal de destino (demultiplexación).
        - Longitud: Especifica el número de bytes del segmento UDP (la cabecera más los datos). Es necesario un valor de longitud explícito ya que el tamaño del campo de datos puede variar de un segmento UDP al siguiente.
        - Suma de Comprobación/Checksum: El host receptor utiliza la suma de comprobación para detectar si se han introducido errores en el segmento.
    - Provee Multiplexación y Demultiplexación de aplicación y detección de errores (no obligatorio).
- **Segmento UDP:**
    - Un segmento/datagrama UDP se conforma de la siguiente manera:
        - Una cabecera UDP.
        - Datos de la aplicación (mensaje): Estos datos ocupan el campo de datos del segmento UDP.

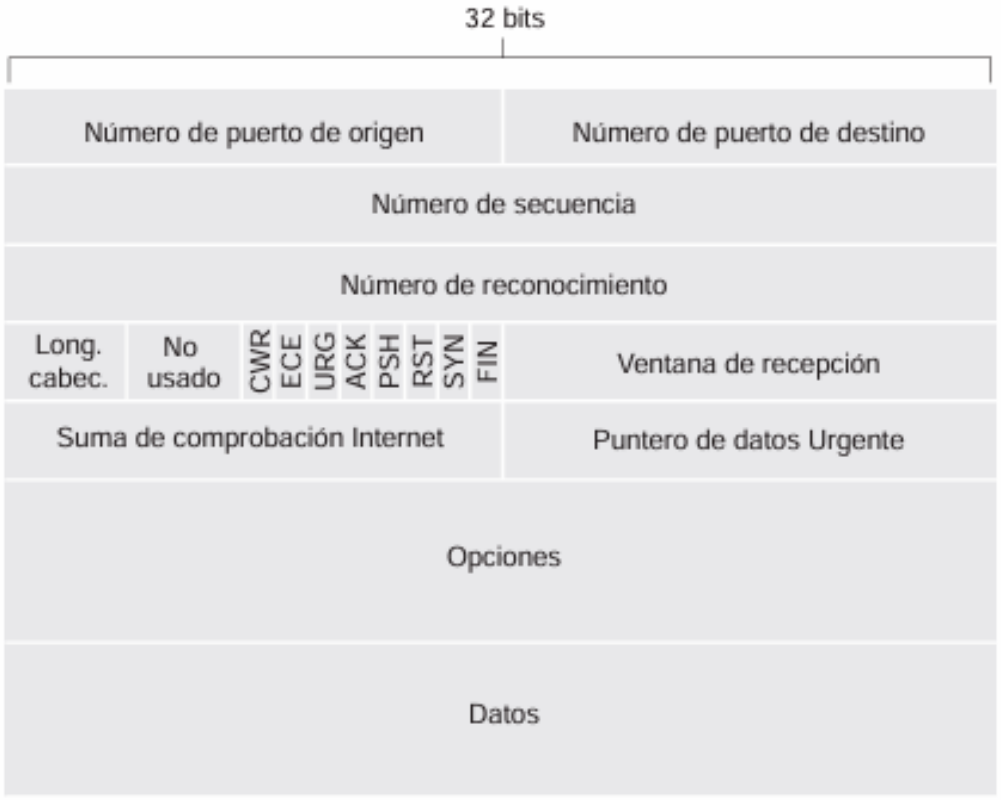


- **Header TCP:**
  - La cabecera TCP es de 20 bytes (12 más que la UDP) e incluye los siguientes campos:
    - Número de puerto de origen y de destino: Se utilizan para multiplexar y demultiplexar los datos de y para las aplicaciones de la capa superior.
    - Suma de comprobación/Checksum.
    - Número de secuencia y Número de reconocimiento: Ambos campos de 32 bits, son utilizados por el emisor y el receptor de TCP para implementar un servicio de transferencia de datos fiable.
    - Ventana de recepción: Campo de 16 bits que se utiliza para el control de flujo (se emplea para indicar el número de bytes que un receptor está dispuesto a aceptar).
    - Longitud de cabecera: Campo de 4 bits que especifica la longitud de la cabecera TCP en palabras de 32 bits. La cabecera TCP puede tener una longitud variable a causa del campo opciones de TCP (normalmente, este campo está vacío, por lo que la longitud de una cabecera TCP típica es de 20 bytes).
    - Opciones: Campo opcional y de longitud variable. Se utiliza cuando un emisor y un receptor negocian el tamaño máximo de segmento (MSS) o como un factor de escala de la ventana en las redes de alta velocidad.
    - Indicador: Campo de 6 bits → En la práctica el PSH, URG no se utilizan:
      - El bit ACK se utiliza para indicar que el valor transportado en el campo de reconocimiento es válido; es decir, el segmento contiene un reconocimiento para un segmento que ha sido recibido correctamente.

- Los bits RST, SYN y FIN se utilizan para el establecimiento y cierre de conexiones.
- Los bits CWR y ECE se emplean en la notificación de congestión explícita.
- La activación del bit PSH indica que el receptor deberá pasar los datos a la capa superior de forma inmediata.
- El bit URG se utiliza para indicar que hay datos en este segmento que la entidad de la capa superior del lado emisor ha marcado como “urgentes”.

    - La posición de este último byte de estos datos urgentes se indica mediante el campo puntero de datos urgentes de 16 bits. TCP tiene que informar a la entidad de la capa superior del lado receptor si existen datos urgentes y pasarle un puntero a la posición donde finalizan los datos urgentes → En la práctica no se usa.
    - El encabezado TCP provee: Multiplexación, Demultiplexación, detección de errores, sesiones, control de errores, control de flujo y control de congestión.

- **Segmento TCP:**
    - El segmento TCP consta de campos cabecera y campo de datos.
        - El Campo De Datos contiene un fragmento de los datos de la aplicación. El MSS limita el tamaño máximo del campo de datos de un segmento. Si se enviara un archivo muy grande, este se vería dividido en varios fragmentos de tamaño MSS.



3. **¿Cuál es el objetivo del uso de puertos en el modelo TCP/IP?**

- Los puertos son esenciales en el modelo TCP/IP, ya que permiten la comunicación entre múltiples aplicaciones en un mismo host. Son cruciales para distinguir entre diferentes aplicaciones que operan simultáneamente en un dispositivo, actuando como puntos finales de comunicación. Cada puerto se asocia con una aplicación específica, permitiendo que el sistema operativo entregue paquetes a la aplicación correspondiente según el número de puerto. En sí, un puerto es un número de 16 bits comprendido en el rango de 0 a 65535. Los números de puerto pertenecientes al rango de 0 a 1023 se conocen como números de puertos bien conocidos y son restringidos, lo que significa que están reservados para ser empleados por los protocolos de aplicación bien conocidos, como por ejemplo HTTP o FTP. Al desarrollar una nueva aplicación, es necesario asignar un número de puerto a la aplicación.

4. **Compare TCP y UDP en cuanto a:**

    a. **Confiabilidad.**

    - TCP es un protocolo confiable, es decir, garantiza que los datos se entreguen en el orden correcto y sin errores a través de técnicas como la retransmisión de datos perdidos y la detección y corrección de errores utilizando sumas de verificación. UDP es un protocolo no confiable.

    b. **Multiplexación.**

    - Si bien ambos protocolos poseen multiplexación y demultiplexación, varía la forma en que estas se hacen:
        - UDP solamente utiliza la dirección IP de destino y un número de puerto de destino.
        - TCP utiliza más campos lo que la hace más compleja y más precisa, los campos que utiliza son: una dirección IP de origen, número de puerto de origen, dirección IP de destino y número de puerto de destino.

    c. **Orientado a la conexión.**

    - TCP es orientado a conexión, esto quiere decir que antes de poder enviar datos y tener una comunicación, 2 hosts deben establecer una conexión lógica entre ellos lo que hace que tenga más Overhead que UDP que no es orientado a conexión.

    d. **Controles de congestión.**

    - TCP implementa mecanismo para el control de congestión mientras que UDP no. El control de congestión permite que las aplicaciones no saturen la

capacidad de la red y tiene como objetivo controlar el tráfico (cantidad de datos que un emisor inyecta en la red) para evitar congestionar la red y, por ende, la pérdida de paquetes y la necesidad de retransmisiones.

e. Utilización de puertos.

- Ambos protocolos hacen uso de los puertos para identificar las aplicaciones que se están comunicando, la diferencia sería que TCP hace una conexión punto a punto donde solo participan 2 procesos mientras que UDP permite que muchos clientes o procesos envíen datos por el mismo socket.

5. La PDU de la capa de transporte es el segmento. Sin embargo, en algunos contextos suele utilizarse el término datagrama. Indique cuando.

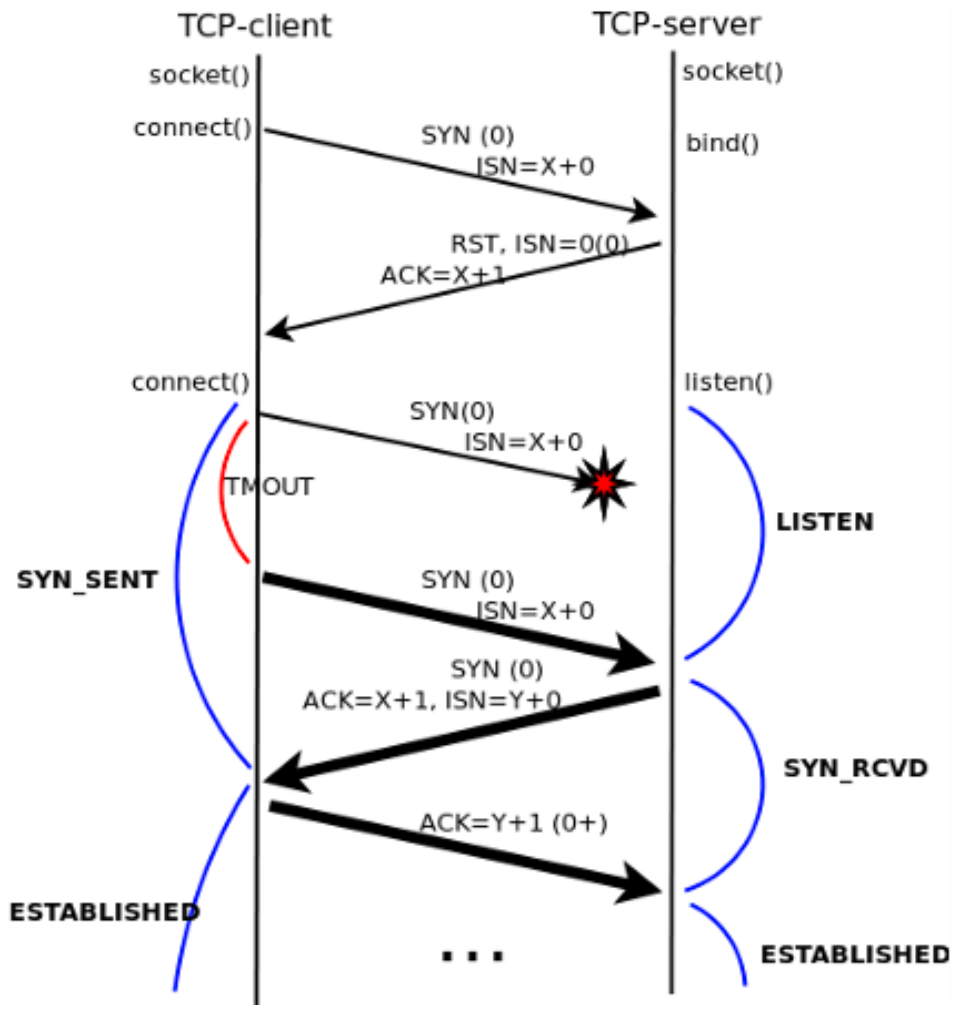
- Cuando nos referimos al protocolo UDP utilizamos el término datagrama de usuario.

6. Describa el saludo de tres vías de TCP. ¿UDP tiene esta característica?

- **Saludo de 3 vías (UDP no lo tiene):**
    - El three-way handshake es el proceso que TCP utiliza para establecer una conexión fiable entre el cliente y el servidor. Este proceso garantiza que ambos extremos estén listos para recibir y enviar datos.
    - Durante este proceso se intercambian segmentos SYN y ACK antes de que comience la comunicación de datos real.
        - Paso 1 (SYN)
            - El cliente envía un segmento con el bit SYN activado y el número de secuencia inicial (ISN), este último es un número aleatorio que servirá como punto de partida para la transmisión de datos, además, ayuda a rastrear y ordenar los segmentos de datos enviados en la conexión.
        - Paso 2 (SYN-ACK)
            - El servidor recibe el segmento SYN y responde con un segmento que tiene los bits SYN y ACK activados, a su vez, el servidor también envía su propio número de secuencia inicial (ISN).
            - La parte ACK del segmento enviado contiene el número de secuencia del cliente incrementado en uno (ISN del cliente + 1) para indicar que se recibió correctamente el SYN del cliente.
            - Se envía RST si no hay proceso en estado LISTEN.
        - Paso 3 (ACK)
            - Cuando el cliente recibe el segmento SYN-ACK del servidor, responde con un segmento que tiene el bit ACK activado. En este segmento ya se puede enviar información.
            - La parte ACK del segmento enviado contiene el número de secuencia del servidor incrementado en uno (ISN del servidor

+ 1) para indicar que se recibió correctamente el SYN-ACK del servidor.

- En este punto, ambos lados han confirmado la conexión, y la conexión TCP se considera establecida.



7. **Investigue qué es el ISN (Initial Sequence Number). Relaciónelo con el saludo de tres vías.**

- El ISN (Initial Sequence Number), o Número de Secuencia Inicial, es un número aleatorio de 32 bits que se utiliza en el protocolo TCP para identificar y numerar cada byte en un flujo de datos. Este número se genera al establecer una conexión TCP y juega un papel crucial en la fiabilidad del protocolo, permitiendo al receptor detectar segmentos perdidos, duplicados o desordenados. El ISN se relaciona con el saludo de 3 vías ya que en este saludo ambos, cliente y servidor, intercambian sus ISN.

8. **Investigue qué es el MSS. ¿Cuándo y cómo se negocia?**

- El MSS (Maximum Segment Size), o Tamaño Máximo de Segmento, es un parámetro en el protocolo TCP que define la cantidad máxima de datos de la capa de aplicación

que puede contener un único segmento TCP. Este valor se especifica en bytes y se negocia durante el saludo de tres vías de TCP, lo que asegura que ambos extremos, cliente y servidor, estén de acuerdo en el tamaño de los segmentos que intercambiarán. Esta negociación se da en el segundo paso del saludo de tres vías, cada extremo propone el MSS que admite y para la conexión se elige el mínimo entre los 2 MSS.

9. **Utilice el comando ss (reemplazo de netstat) para obtener la siguiente información de su PC:**

   a. **Para listar las comunicaciones TCP establecidas.**

State	Recv-Q	Send-Q	Local Address:Port	Peer Address:Port	Process
LISTEN	0	5	127.0.0.1:4038	0.0.0.0:*	
LISTEN	0	128	0.0.0.0:ssh	0.0.0.0:*	
LISTEN	0	128	127.0.0.1:ipp	0.0.0.0:*	
ESTAB	0	0	10.0.2.15:53984	142.250.79.74:https	
TIME-WAIT	0	0	10.0.2.15:49314	200.42.33.137:http	
ESTAB	0	0	10.0.2.15:34630	142.251.133.234:https	
ESTAB	0	0	10.0.2.15:41524	200.42.33.146:http	
ESTAB	0	0	10.0.2.15:54406	34.107.221.82:http	
TIME-WAIT	0	0	10.0.2.15:35052	142.251.134.67:http	
TIME-WAIT	0	0	10.0.2.15:35060	142.251.134.67:http	
ESTAB	0	0	10.0.2.15:47796	34.117.121.53:https	
ESTAB	0	0	10.0.2.15:42268	34.107.221.82:http	
ESTAB	0	0	10.0.2.15:40328	35.201.103.21:https	
ESTAB	0	0	10.0.2.15:54710	216.58.202.110:https	
ESTAB	0	0	10.0.2.15:39760	34.149.100.209:https	
ESTAB	0	0	10.0.2.15:50538	142.251.133.202:https	
ESTAB	0	0	10.0.2.15:33148	200.42.33.137:http	
TIME-WAIT	0	0	10.0.2.15:35066	142.251.134.67:http	
ESTAB	0	0	10.0.2.15:56262	142.251.133.206:https	
ESTAB	0	0	10.0.2.15:51402	34.160.144.191:https	
TIME-WAIT	0	0	10.0.2.15:56518	200.42.33.146:http	
TIME-WAIT	0	0	10.0.2.15:56532	200.42.33.146:http	
TIME-WAIT	0	0	10.0.2.15:32856	142.251.134.67:http	
ESTAB	0	0	10.0.2.15:54662	34.98.75.36:https	
TIME-WAIT	0	0	10.0.2.15:49300	200.42.33.137:http	
ESTAB	0	0	10.0.2.15:38606	34.107.243.93:https	
LISTEN	0	4096	[::1]:50051	[::]:*	
LISTEN	0	4096	[::ffff:127.0.0.1]:50051	*:*	
LISTEN	0	128	[::]:ssh	[::]:*	
LISTEN	0	128	[::1]:ipp	[::]:*	

ss -t -a

   b. **Para listar las comunicaciones UDP establecidas.**

State	Recv-Q	Send-Q	Local Address:Port	Peer Address:Port	Process
UNCONN	0	0	0.0.0.0:631	0.0.0.0:*	
UNCONN	0	0	0.0.0.0:45843	0.0.0.0:*	
UNCONN	0	0	0.0.0.0:56488	0.0.0.0:*	
UNCONN	0	0	0.0.0.0:mdns	0.0.0.0:*	
UNCONN	0	0	127.0.0.1:4038	0.0.0.0:*	
UNCONN	0	0	0.0.0.0:40985	0.0.0.0:*	
ESTAB	0	0	10.0.2.15%enp0s3:bootpc	10.0.2.2:bootps	
UNCONN	0	0	[::]:54043	[::]:*	
UNCONN	0	0	[::]:mdns	[::]:*	

ss -u -a

   c. **Obtener sólo los servicios TCP que están esperando comunicaciones.**

```
State            Recv-Q           Send-Q                     Local Address:Port                     Peer Address:Port           Process
LISTEN           0                5                              127.0.0.1:4038                          0.0.0.0:*
LISTEN           0                128                              0.0.0.0:ssh                           0.0.0.0:*
LISTEN           0                128                            127.0.0.1:ipp                           0.0.0.0:*
LISTEN           0                4096                             [::1]:50051                              [::]:*
LISTEN           0                4096                  [::ffff:127.0.0.1]:50051                                *:*
LISTEN           0                128                              [::]:ssh                              [::]:*
LISTEN           0                128                             [::1]:ipp                              [::]:*
```

ss -t -l

d. Obtener sólo los servicios UDP que están esperando comunicaciones.

```
State            Recv-Q           Send-Q                     Local Address:Port                     Peer Address:Port           Process
UNCONN           0                0                                0.0.0.0:631                           0.0.0.0:*
UNCONN           0                0                                0.0.0.0:45843                         0.0.0.0:*
UNCONN           0                0                                0.0.0.0:mdns                          0.0.0.0:*
UNCONN           0                0                                0.0.0.0:38754                         0.0.0.0:*
UNCONN           0                0                              127.0.0.1:4038                          0.0.0.0:*
UNCONN           0                0                                0.0.0.0:40985                         0.0.0.0:*
UNCONN           0                0                                   [::]:54043                            [::]:*
UNCONN           0                0                                   [::]:mdns                             [::]:*
```

ss -u -l

e. Repetir los anteriores para visualizar el proceso del sistema asociado a la conexión.

```
State        Recv-Q       Send-Q                 Local Address:Port                  Peer Address:Port        Process
LISTEN       0            5                          127.0.0.1:4038                       0.0.0.0:*
LISTEN       0            128                          0.0.0.0:ssh                        0.0.0.0:*
LISTEN       0            128                        127.0.0.1:ipp                        0.0.0.0:*
ESTAB        0            0                         10.0.2.15:38836                35.244.181.201:https        users:(("firefox-esr",pid=3261,fd=112))
ESTAB        0            0                         10.0.2.15:42984               172.217.173.234:https        users:(("firefox-esr",pid=3261,fd=110))
ESTAB        0            0                         10.0.2.15:40604                34.160.144.191:https        users:(("firefox-esr",pid=3261,fd=74))
ESTAB        0            0                         10.0.2.15:47174                151.101.65.91:https         users:(("firefox-esr",pid=3261,fd=44))
ESTAB        0            0                         10.0.2.15:56230                 192.16.49.85:http          users:(("firefox-esr",pid=3261,fd=117))
ESTAB        0            0                         10.0.2.15:56222                 192.16.49.85:http          users:(("firefox-esr",pid=3261,fd=115))
ESTAB        0            0                         10.0.2.15:38606                34.107.243.93:https         users:(("firefox-esr",pid=3261,fd=128))
LISTEN       0            4096                           [::1]:50051                        [::]:*
LISTEN       0            4096                [::ffff:127.0.0.1]:50051                          *:*
LISTEN       0            128                            [::]:ssh                           [::]:*
LISTEN       0            128                           [::1]:ipp                           [::]:*
```

ss -t -p -a

```
State          Recv-Q         Send-Q                 Local Address:Port                  Peer Address:Port          Process
UNCONN         0              0                            0.0.0.0:631                        0.0.0.0:*
UNCONN         0              0                            0.0.0.0:45843                      0.0.0.0:*
UNCONN         0              0                            0.0.0.0:mdns                       0.0.0.0:*
UNCONN         0              0                            0.0.0.0:34100                      0.0.0.0:*           users:(("firefox-esr",pid=3261,fd=38))
UNCONN         0              0                            0.0.0.0:36311                      0.0.0.0:*           users:(("firefox-esr",pid=3261,fd=43))
UNCONN         0              0                          127.0.0.1:4038                       0.0.0.0:*
UNCONN         0              0                            0.0.0.0:40985                      0.0.0.0:*           users:(("firefox-esr",pid=3261,fd=83))
ESTAB          0              0                10.0.2.15%enp0s3:bootpc                10.0.2.2:bootps
UNCONN         0              0                             [::]:54043                         [::]:*
UNCONN         0              0                             [::]:mdns                          [::]:*
```

ss -u -p -a

```
State            Recv-Q           Send-Q                     Local Address:Port                     Peer Address:Port           Process
LISTEN           0                5                              127.0.0.1:4038                          0.0.0.0:*
LISTEN           0                128                              0.0.0.0:ssh                           0.0.0.0:*
LISTEN           0                128                            127.0.0.1:ipp                           0.0.0.0:*
LISTEN           0                4096                             [::1]:50051                              [::]:*
LISTEN           0                4096                  [::ffff:127.0.0.1]:50051                                *:*
LISTEN           0                128                              [::]:ssh                              [::]:*
LISTEN           _ 0              128                             [::1]:ipp                              [::]:*
```

ss -t -p -l

```
State          Recv-Q         Send-Q                 Local Address:Port                  Peer Address:Port          Process
UNCONN         0              0                            0.0.0.0:631                        0.0.0.0:*
UNCONN         0              0                            0.0.0.0:45843                      0.0.0.0:*
UNCONN         0              0                            0.0.0.0:56355                      0.0.0.0:*           users:(("firefox-esr",pid=3261,fd=41))
UNCONN         0              0                            0.0.0.0:mdns                       0.0.0.0:*
UNCONN         0              0                            0.0.0.0:48429                      0.0.0.0:*           users:(("firefox-esr",pid=3261,fd=35))
UNCONN         0              0                          127.0.0.1:4038                       0.0.0.0:*
UNCONN         0              0                            0.0.0.0:40985                      0.0.0.0:*           users:(("firefox-esr",pid=3261,fd=83))
UNCONN         0              0                             [::]:54043                         [::]:*
UNCONN         0              0                             [::]:mdns                          [::]:*
```

ss -u -p -l

f. Obtenga la misma información planteada en los items anteriores usando el comando netstat.

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 localhost:4038          0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN
tcp        0      0 localhost:ipp           0.0.0.0:*               LISTEN
tcp        0      0 10.0.2.15:38606         34.107.243.93:https     ESTABLISHED
tcp6       0      0 localhost:50051         [::]:*                  LISTEN
tcp6       0      0 127.0.0.1:50051         [::]:*                  LISTEN
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN
tcp6       0      0 localhost:ipp           [::]:*                  LISTEN
```

netstat -t -a

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
udp        0      0 0.0.0.0:631             0.0.0.0:*
udp        0      0 0.0.0.0:45843           0.0.0.0:*
udp        0      0 0.0.0.0:mdns            0.0.0.0:*
udp        0      0 0.0.0.0:54997           0.0.0.0:*
udp        0      0 0.0.0.0:55021           0.0.0.0:*
udp        0      0 localhost:4038          0.0.0.0:*
udp        0      0 0.0.0.0:40985           0.0.0.0:*
udp        0      0 10.0.2.15:bootpc        10.0.2.2:bootps         ESTABLISHED
udp6       0      0 [::]:54043              [::]:*
udp6       0      0 [::]:mdns               [::]:*
```

netstat -u -a

```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 localhost:4038          0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN
tcp        0      0 localhost:ipp           0.0.0.0:*               LISTEN
tcp6       0      0 localhost:50051         [::]:*                  LISTEN
tcp6       0      0 127.0.0.1:50051         [::]:*                  LISTEN
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN
tcp6       0      0 localhost:ipp           [::]:*                  LISTEN
```

netstat -t -l

```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
udp        0      0 0.0.0.0:631             0.0.0.0:*
udp        0      0 0.0.0.0:45843           0.0.0.0:*
udp        0      0 0.0.0.0:mdns            0.0.0.0:*
udp        0      0 0.0.0.0:55021           0.0.0.0:*
udp        0      0 localhost:4038          0.0.0.0:*
udp        0      0 0.0.0.0:40985           0.0.0.0:*
udp        0      0 0.0.0.0:43231           0.0.0.0:*
udp6       0      0 [::]:54043              [::]:*
udp6       0      0 [::]:mdns               [::]:*
```

netstat -u -l

10. **¿Qué sucede si llega un segmento TCP con el flag SYN activo a un host que no tiene ningún proceso esperando en el puerto destino de dicho segmento (es decir, el puerto destino no está en estado LISTEN)?**

- Si no hay proceso en estado LISTEN en el puerto destino, el host enviará un segmento con la flag RST (reset) activado para indicar que la conexión no se puede establecer en dicho puerto. De esta manera informará al remitente que no hay no se puede establecer la conexión en ese momento para que no siga enviando segmentos.

    a. **Utilice hping3 para enviar paquetes TCP al puerto destino 22 de la máquina virtual con el flag SYN activado.**

```
HPING localhost (lo 127.0.0.1): S set, 40 headers + 0 data bytes
len=44 ip=127.0.0.1 ttl=64 DF id=0 sport=22 flags=SA seq=0 win=65495 rtt=3.4 ms
len=44 ip=127.0.0.1 ttl=64 DF id=0 sport=22 flags=SA seq=1 win=65495 rtt=10.9 ms
len=44 ip=127.0.0.1 ttl=64 DF id=0 sport=22 flags=SA seq=2 win=65495 rtt=14.2 ms
len=44 ip=127.0.0.1 ttl=64 DF id=0 sport=22 flags=SA seq=3 win=65495 rtt=4.1 ms
len=44 ip=127.0.0.1 ttl=64 DF id=0 sport=22 flags=SA seq=4 win=65495 rtt=11.0 ms
len=44 ip=127.0.0.1 ttl=64 DF id=0 sport=22 flags=SA seq=5 win=65495 rtt=4.2 ms
len=44 ip=127.0.0.1 ttl=64 DF id=0 sport=22 flags=SA seq=6 win=65495 rtt=4.8 ms
len=44 ip=127.0.0.1 ttl=64 DF id=0 sport=22 flags=SA seq=7 win=65495 rtt=9.3 ms
^C
--- localhost hping statistic ---
8 packets transmitted, 8 packets received, 0% packet loss
round-trip min/avg/max = 3.4/7.7/14.2 ms
```

sudo hping3 -S localhost -p 22

    b. **Utilice hping3 para enviar paquetes TCP al puerto destino 40 de la máquina virtual con el flag SYN activado.**

```
HPING localhost (lo 127.0.0.1): S set, 40 headers + 0 data bytes
len=40 ip=127.0.0.1 ttl=64 DF id=0 sport=40 flags=RA seq=0 win=0 rtt=2.8 ms
len=40 ip=127.0.0.1 ttl=64 DF id=0 sport=40 flags=RA seq=1 win=0 rtt=1.8 ms
len=40 ip=127.0.0.1 ttl=64 DF id=0 sport=40 flags=RA seq=2 win=0 rtt=9.4 ms
len=40 ip=127.0.0.1 ttl=64 DF id=0 sport=40 flags=RA seq=3 win=0 rtt=6.2 ms
len=40 ip=127.0.0.1 ttl=64 DF id=0 sport=40 flags=RA seq=4 win=0 rtt=5.0 ms
len=40 ip=127.0.0.1 ttl=64 DF id=0 sport=40 flags=RA seq=5 win=0 rtt=6.9 ms
len=40 ip=127.0.0.1 ttl=64 DF id=0 sport=40 flags=RA seq=6 win=0 rtt=2.0 ms
len=40 ip=127.0.0.1 ttl=64 DF id=0 sport=40 flags=RA seq=7 win=0 rtt=2.9 ms
len=40 ip=127.0.0.1 ttl=64 DF id=0 sport=40 flags=RA seq=8 win=0 rtt=7.7 ms
^C
--- localhost hping statistic ---
9 packets transmitted, 9 packets received, 0% packet loss
round-trip min/avg/max = 1.8/5.0/9.4 ms
```

sudo hping3 -S localhost -p 40

c. **¿Qué diferencias nota en las respuestas obtenidas en los dos casos anteriores? ¿Puede explicar a qué se debe? (Ayuda: utilice el comando ss visto anteriormente).**

- La diferencia está en los flags retornados, para el puerto 22 se retorna SA lo que significa que el puerto está abierto, es decir, los flags SYN/ACK están activados y se pudo establecer la comunicación. En el caso del puerto 40 se retorna RA lo que significa que el puerto está cerrado, es decir, RST/ACK están activados y no se pudo establecer la comunicación.

11. **¿Qué sucede si llega un datagrama UDP a un host que no tiene ningún proceso esperando en el puerto destino de dicho datagrama (es decir, que dicho puerto no está en estado LISTEN)?**

- Si no hay proceso en estado LISTEN en el puerto destino se responderá con un ICMP “Destination Unreachable”. Este mensaje ICMP indica que el puerto o el host destino no están disponibles.

a. **Utilice hping3 para enviar datagramas UDP al puerto destino 5353 de la máquina virtual.**

```
HPING localhost (lo 127.0.0.1): udp mode set, 28 headers + 0 data bytes
^C
--- localhost hping statistic ---
4 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

sudo hping3 –udp localhost -p 5353

b. **Utilice hping3 para enviar datagramas UDP al puerto destino 40 de la máquina virtual.**

```
HPING localhost (lo 127.0.0.1): udp mode set, 28 headers + 0 data bytes
ICMP Port Unreachable from ip=127.0.0.1 name=localhost
status=0 port=2551 seq=0
ICMP Port Unreachable from ip=127.0.0.1 name=localhost
status=0 port=2552 seq=1
ICMP Port Unreachable from ip=127.0.0.1 name=localhost
status=0 port=2553 seq=2
ICMP Port Unreachable from ip=127.0.0.1 name=localhost
status=0 port=2554 seq=3
ICMP Port Unreachable from ip=127.0.0.1 name=localhost
status=0 port=2555 seq=4
ICMP Port Unreachable from ip=127.0.0.1 name=localhost
status=0 port=2556 seq=5
^C
--- localhost hping statistic ---
6 packets transmitted, 6 packets received, 0% packet loss
round-trip min/avg/max = 2.4/7.6/22.6 ms
```

sudo hping3 –udp localhost -p 40

c. **¿Qué diferencias nota en las respuestas obtenidas en los dos casos anteriores? ¿Puede explicar a qué se debe? (Ayuda: utilice el comando ss visto anteriormente).**

- La diferencia es que en el puerto 5353 hay un proceso escuchando, aunque no se dé ninguna indicación de que los datagramas llegan o no, mientras que en el puerto 40 no hay ningún proceso escuchando y por lo tanto se nos responde con un ICMP Unreachable port.

12. **Investigue los distintos tipos de estados que puede tener una conexión TCP. Ver https://users.cs.northwestern.edu/~agupta/cs340/project2/TCPIP_State_Transition_Diagram .pdf**

- CLOSED: Es la ausencia total de una conexión activa.
- LISTEN: Este estado representa la espera de una solicitud de conexión precedente de cualquier TCP remoto y puerto.
- SYN_SENT: Representa la espera de una solicitud de conexión coincidente después de haber enviado una solicitud de conexión.
- SYN-RECEIVED: Representa la espera de una confirmación de solicitud de conexión después de haber recibido y enviado una solicitud de conexión.
- ESTABLISHED: Representa una conexión abierta, lo que permite que los datos recibidos sean entregados al usuario. Es el estado normal durante la fase de transferencia de datos de la conexión.
- FIN-WAIT-1: Representa la espera de una solicitud de terminación de conexión procedente del TCP remoto, o la confirmación de la solicitud de terminación de conexión previamente enviada.
- FIN-WAIT-2: Representa la espera de una solicitud de terminación de conexión procedente del TCP remoto.

- CLOSE-WAIT: Representa la espera de una solicitud de terminación de conexión procedente del usuario local.
- CLOSING: Representa la espera de una confirmación de la solicitud de terminación de conexión procedente del TCP remoto.
- LAST-ACK: Representa la espera de una confirmación de la solicitud de terminación de conexión previamente enviada al TCP remoto (la cual incluye una confirmación de su solicitud de terminación de conexión).
- TIME-WAIT: Representa la espera de suficiente tiempo para asegurar que el TCP remoto ha recibido la confirmación de su solicitud de terminación de conexión.

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

```
Netid State      Recv-Q Send-Q Local Address:Port        Peer Address:Port       Process
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-WAI         T 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 9 conexiones establecidas (ESTAB).

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

- Hay 5 puertos abiertos a la espera de nuevas conexiones (LISTEN).

c. El cliente y el servidor de las comunicaciones HTTPS (puerto 443), ¿residen en la misma máquina?

- No, no residen en la misma máquina, si los 2 estuvieran en la misma máquina deberían de coincidir las IP's.

d. El cliente y el servidor de la comunicación SSH (puerto 22), ¿residen en la misma máquina?

- Si, ambos residen en la misma máquina ya que comparten la misma 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.

Puerto	Protocolo	Proceso - PID	Rol
22	TCP	sshd - 468	Servidor
80	TCP	apache2 - 991	Servidor
53	UDP	named - 452	Servidor
443	TCP	x-www-browser - 1079	Cliente
443	TCP	x-www-browser - 1079	Cliente
443	TCP	x-www-browser - 1079	Cliente
443	TCP	x-www-browser - 1079	Cliente
443	TCP	x-www-browser - 1079	Cliente
25	TCP	postfix - 627	Servidor
22	TCP	sshd -1418 sshd - 1416	Servidor
22	TCP	ssh - 1415	Cliente
443	TCP	x-www-browser - 1079	Cliente
443	TCP	x-www-browser - 1079	Cliente
443	TCP	x-www-browser - 1079	Cliente
443	TCP	x-www-browser - 1079	Cliente
443	TCP	x-www-browser - 1079	Cliente
9500	TCP	x-www-browser - 1079	Cliente
53	UDP	named - 452	Servidor

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

- Las conexiones que estén en estado TIME-WAIT son las que el cierre fue iniciado por el host local, las que estén en CLOSE-WAIT por el remoto.

g. **¿Cuántas conexiones están aún pendientes por establecerse?**

- Una sola (SYN-SENT).

14. 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 cliente envió un segmento SYN al servidor y este fue recibido ya que el servidor está en estado SYN-RECV, el servidor, al alcanzar ese estado lo que hace es recibir el SYN y enviar el SYN-ACK al cliente pero este último todavía está en estado SYN-SENT, por lo tanto el segmento con el SYN-ACK se perdió en la red.

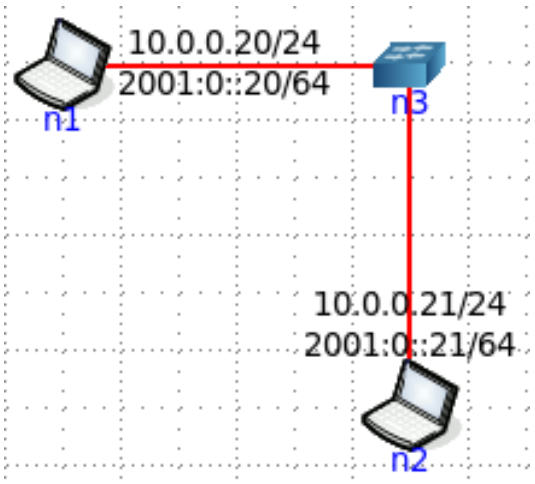
b. **¿A qué protocolo de capa de aplicación y de transporte se está intentando conectar el cliente?**

- El cliente se está queriendo conectar al protocolo de aplicación POP3 ya que envía un segmento al puerto 110, a su vez, se está queriendo conectar usando TCP como protocolo de transporte.

c. **¿Qué flags tendría seteado el segmento perdido?**

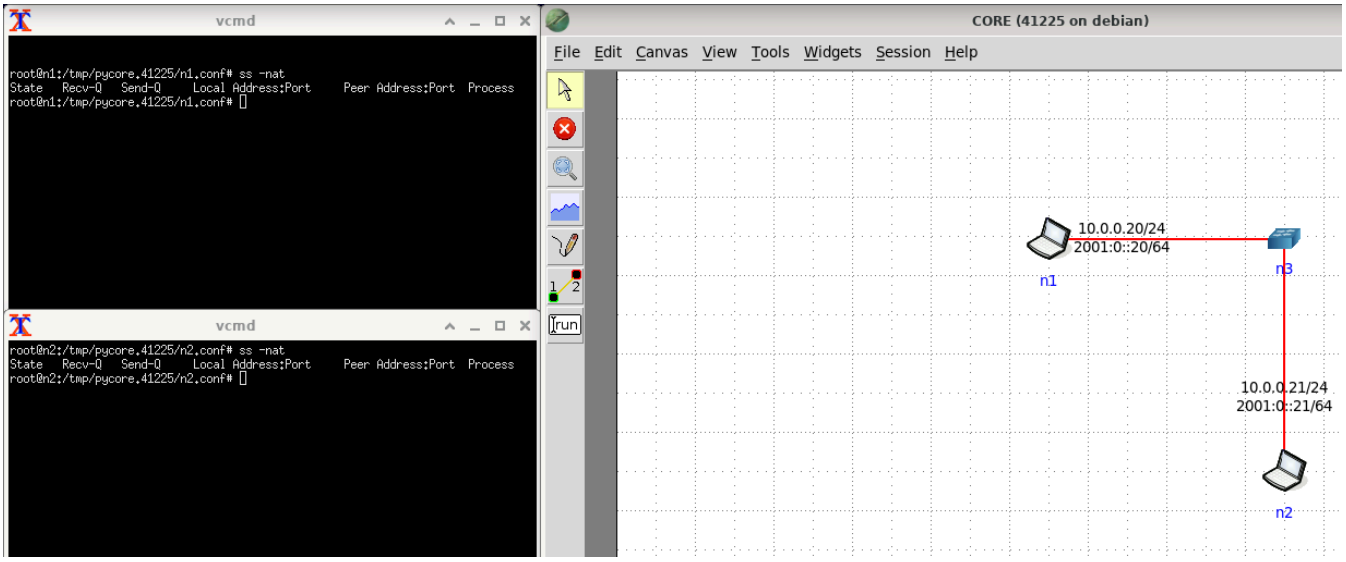
- El segmento perdido tendría los flags SYN y ACK seteados.

15. Use CORE para armar una topología como la siguiente, sobre la cual deberá realizar:



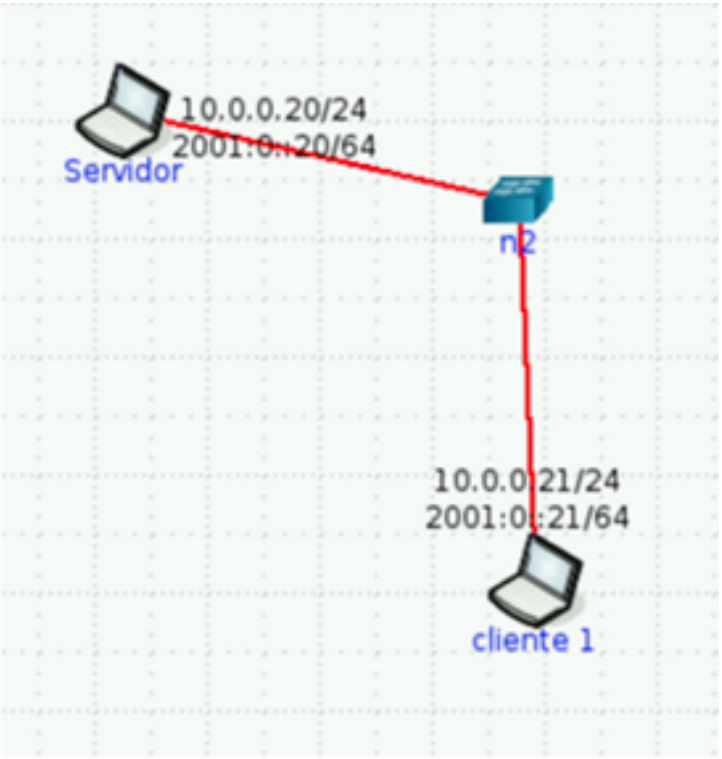
mi topologìa

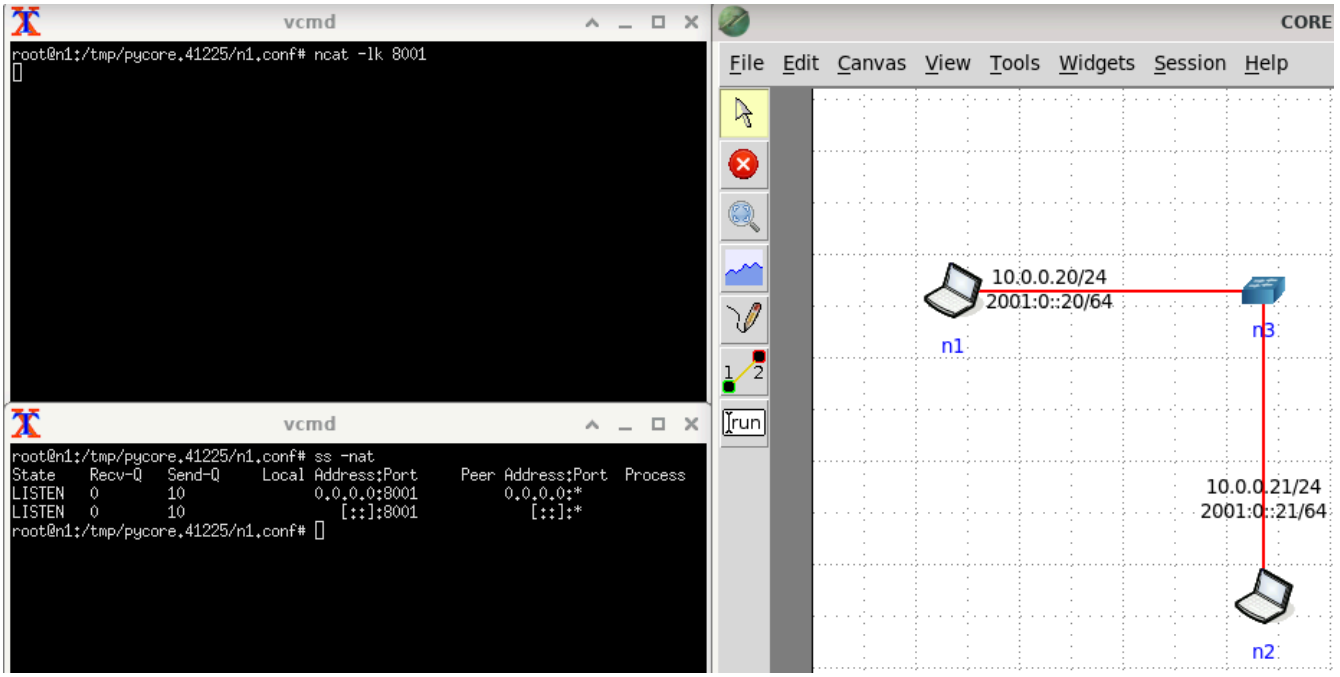
a. **En ambos equipos inspeccionar el estado de las conexiones y mantener abiertas ambas ventanas con el comando corriendo para poder visualizar los cambios a medida que se realiza el ejercicio. Ayuda: watch-n1 'ss-nat'.**



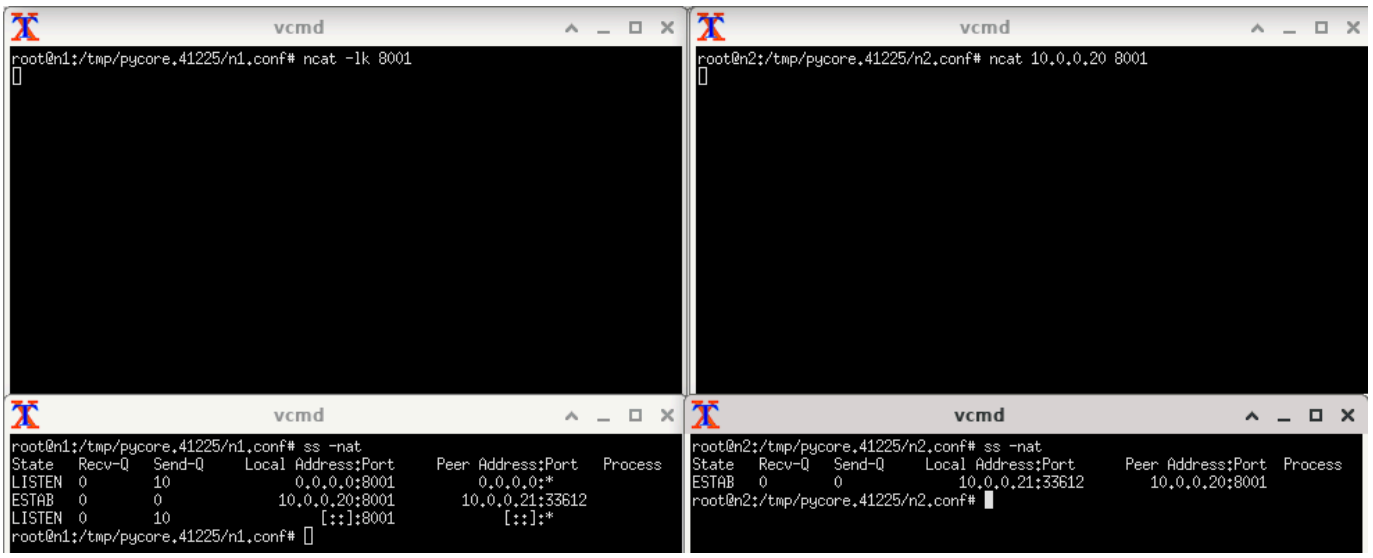
tuve que usar ss -nat

b. **En Servidor, utilice la herramienta ncat para levantar un servicio que escuche en el puerto 8001/TCP. Utilice la opción -k para que el servicio sea persistente. Verifique el estado de las conexiones.**

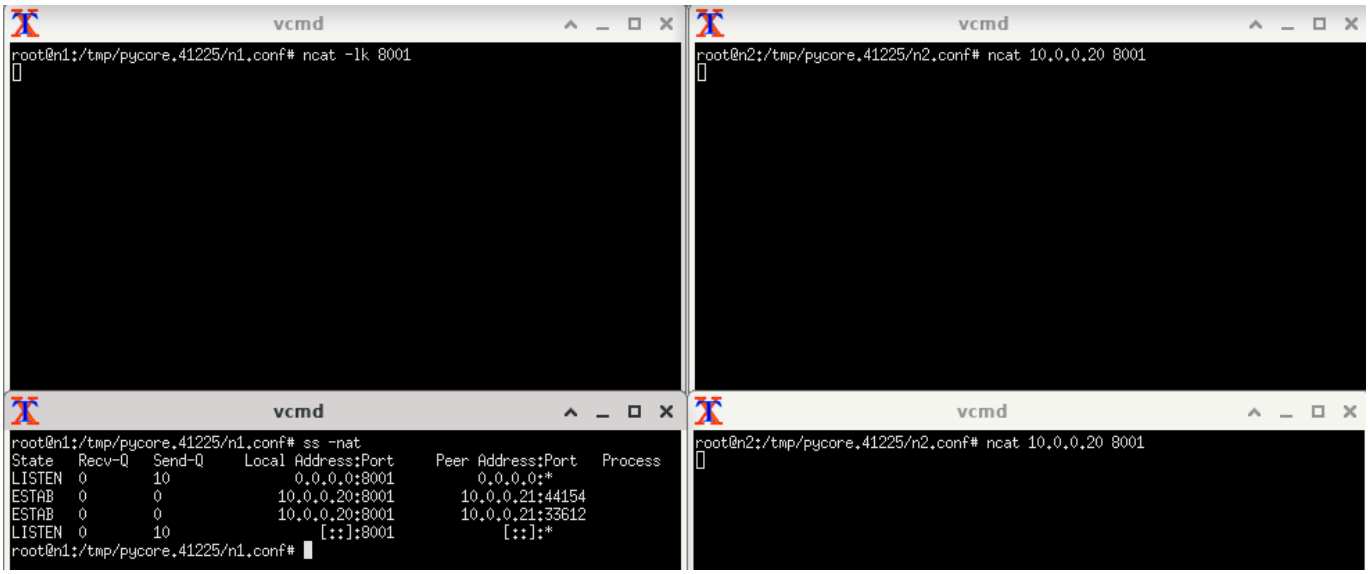




c. **Desde CLIENTE1 conectarse a dicho servicio utilizando también la herramienta ncat. Inspeccione el estado de las conexiones.**



d. Iniciar otra conexión desde CLIENTE1 de la misma manera que la anterior y verificar el estado de las conexiones. ¿De qué manera puede identificar cada conexión?



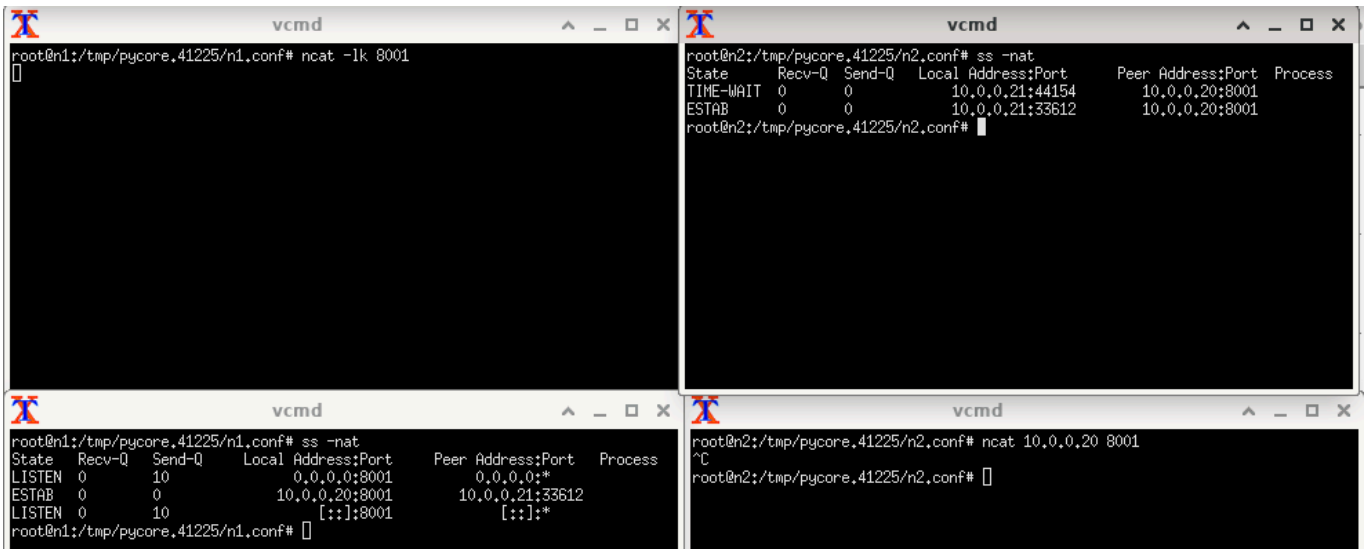
las conexiones las podemos identificar debido a los nùmeros de puerto que son distintos

e. En base a lo observado en el ítem anterior, ¿es posible iniciar más de una conexión desde el cliente al servidor en el mismo puerto destino? ¿Por qué? ¿Cómo se garantiza que los datos de una conexión no se mezclarán con los de la otra?

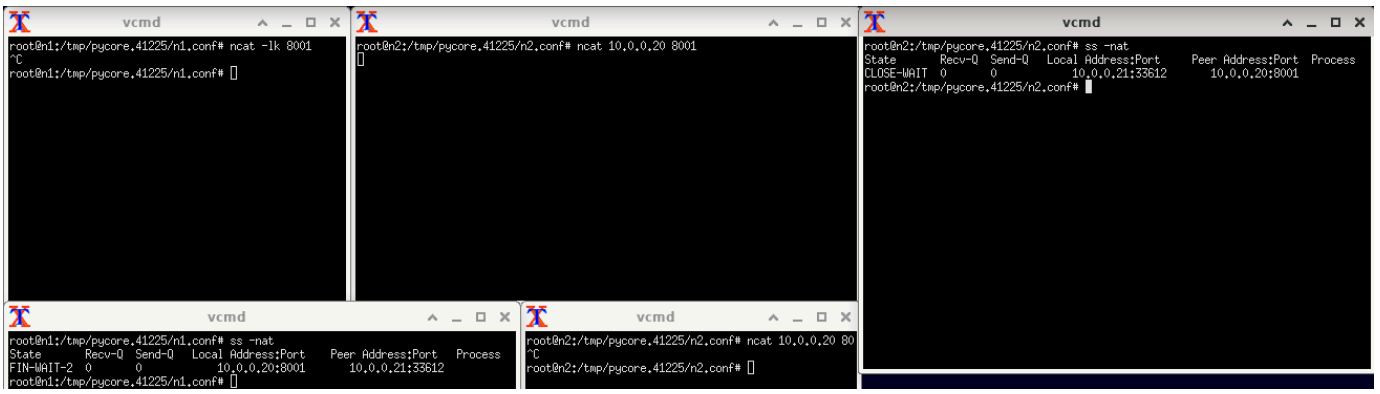
- Si, es posible ya que cada cliente tiene distinto puerto origen, por lo tanto, las identificaciones en base a las IP's de origen y de destino y los números de puerto de origen y destino siguen siendo únicas. Cada conexión se va a poder gestionar individualmente y sin mezclarse debido a que los identificadores son únicos.

f. **Analice en el tráfico de red, los flags de los segmentos TCP que ocurren cuando:**

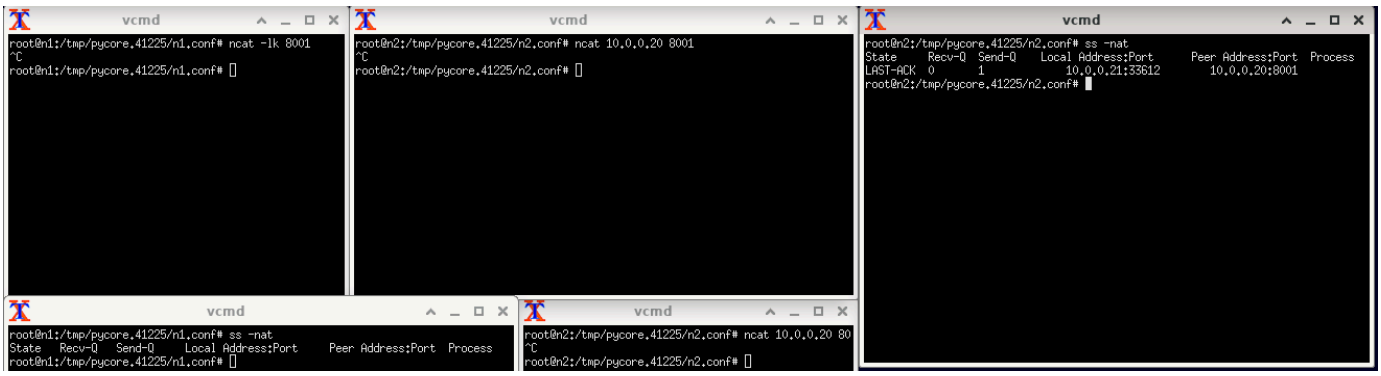
i. **Cierra la última conexión establecida desde CLIENTE1. Evalúe los estados de las conexiones en ambos equipos.**



ii. **Corta el servicio de ncat en el servidor (Ctrl+C). Evalúe los estados de las conexiones en ambos equipos.**



iii. **Cierra la conexión en el cliente. Evalúe nuevamente los estados de las conexiones.**


vcmd
root@n1:/tmp/pycore.41225/n1.conf# ncat -lk 8001
^C
root@n1:/tmp/pycore.41225/n1.conf#
vcmd
root@n2:/tmp/pycore.41225/n2.conf# ncat 10.0.0.20 8001
^C
root@n2:/tmp/pycore.41225/n2.conf#
vcmd
root@n2:/tmp/pycore.41225/n2.conf# ss -nat
State Recv-Q Send-Q Local Address:Port Peer Address:Port Process
LAST-ACK 0 1 10.0.0.21:33612 10.0.0.20:8001
root@n2:/tmp/pycore.41225/n2.conf#
vcmd
root@n1:/tmp/pycore.41225/n1.conf# ss -nat
State Recv-Q Send-Q Local Address:Port Peer Address:Port Process
root@n1:/tmp/pycore.41225/n1.conf#
vcmd
root@n2:/tmp/pycore.41225/n2.conf# ncat 10.0.0.20 80
^C
root@n2:/tmp/pycore.41225/n2.conf#