TCP (Stop and wait)

Autor: Joaquin Lopez

Estado y como ejecutar

En el estado actual, el codigo para crear sockets TCP corre todos los tests provistos en los codigos test_clientTCP.py y test_serverTCP.py, con y sin inducir perdidas via netem.

Para las pruebas, se decidio hacer un ejemplo simple de una sola llamada de server y cliente en los archivos server.py y client.py

El protocolo de comunicaciones a usar, es que espera un '\n' al final del mensaje, el mismo de la actividad 0.

Se usaron las mismas funciones para recibir la completitud del mensaje.

Para ejecutar las pruebas, se debe de ejecutar primero el server

```
python3 server.py <ip> <port>
```

y luego el cliente que recibe un archivo de texto

```
python3 client.py <ip> <port> < <algunarchivo.txt>
```

Si se ejecuta el cliente sin redirigir un archivo a la salida estandar, se puede ingresar un mensaje tambien.

Se probo con varios buffsizes incluyendo impares.

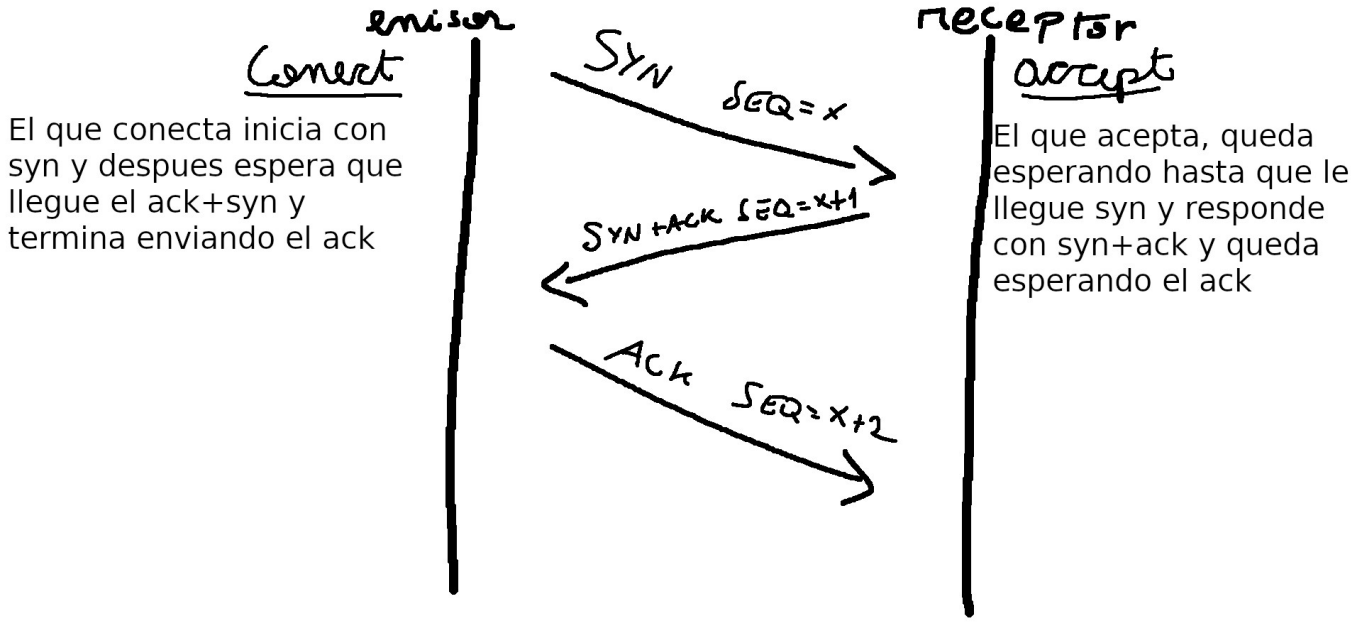
Quitar modo debug

Para quitar el modo debug solo basta con cambiar la linea de INFO a WARN al inicio del archivo sockettcp

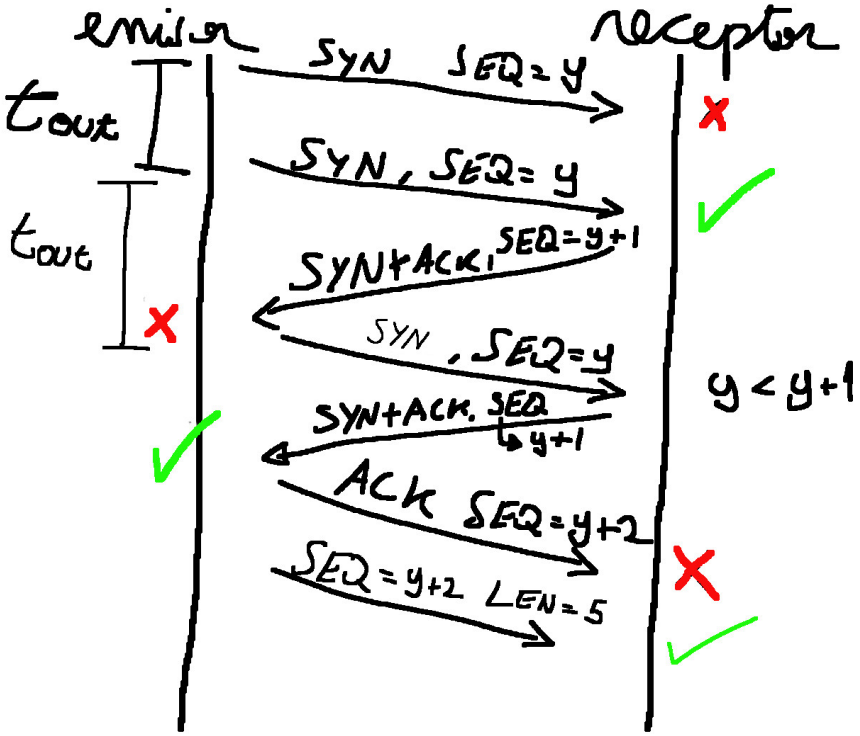
```
logging.basicConfig(format="%(levelname)s -> %(message)s",
level=logging.WARN)
```

Diagramas y desiciones de diseño

Para la seccion 4 se adjunta la imagen junto a la explicacion.



Tambien el diagrama para la seccion 7



El diagrama de la seccion 7 es el 3 way handshake pero con stop & wait, explicando mas el diagrama, se tomo la decisión de:

Si se pierde el ultimo ACK, podemos arreglarlo si permitimos que el mensaje de datos que nos llegue en el futuro, sea el mensaje que permita hacer ACK de que si estamos comunicados.

Esto lo podemos ver al final del diagrama de la seccion 7, al receptor le llega un mensaje con datos y como toma ese como un ACK, sale de la funcion de `accept` y entra a la funcion para `recibir` y de ahí recibe el

largo.

Decisiones de diseño Seccion 5

Para la clase socket tcp, se decidio usar los headers recomendados y se decidio que todos los buffers de los sockets UDP, independiente si tiene datos o no, vengan con el largo de los headers + el buffer de datos, el porque es, por lo dicho anteriormente en la seccion 7.