

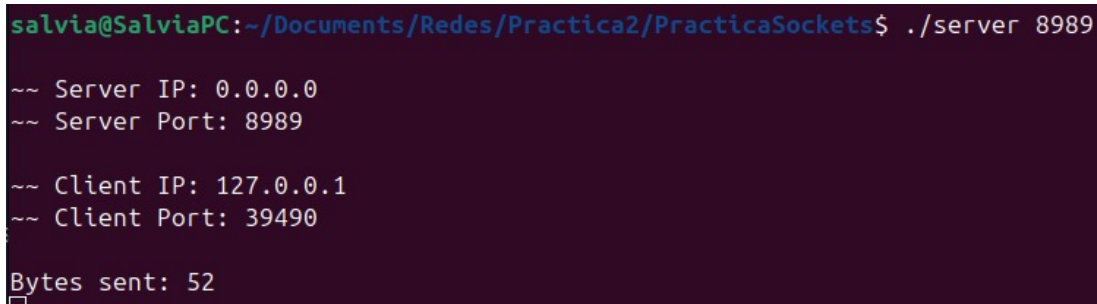
Informe Practica 2 - Redes

Salvia Sisa Cortés, Sergio Souto Mourelle

Apartado 1.C

Comenzamos haciendo cambios en el código del servidor creado en los apartados anteriores, añadimos únicamente otra función `send()` para así enviar 2 mensajes distintos al cliente, en el cual metemos un `sleep(5)` antes de la función `recv()`. Esto hace que el cliente espere a recibir los mensajes y así nos aseguramos que se envían los dos y no recibe el primer mensaje antes de enviar el segundo. El tamaño en bytes que recibe es la suma de los bytes de ambos mensajes.

Cuando quitamos el `sleep()` en el cliente es cuestión de suerte que se reciban ambos mensajes o no, ya que a veces al servidor no le da tiempo a mandar ambos mensajes antes de que el cliente cierre la conexión después de recibir el primero, aunque depende de factores externos el hecho de que reciba ambos antes de cerrar conexión o no.

A terminal window with a dark purple background. The prompt is 'salvia@SalviaPC:~/Documents/Redes/Practica2/PracticaSockets\$'. The command './server 8989' has been executed. The output shows server and client IP/port information and the number of bytes sent.

```
salvia@SalviaPC:~/Documents/Redes/Practica2/PracticaSockets$ ./server 8989
~~ Server IP: 0.0.0.0
~~ Server Port: 8989

~~ Client IP: 127.0.0.1
~~ Client Port: 39490

Bytes sent: 52
```

(52 es la suma de los bytes de ambos mensajes)

Apartado 1.D

Al usar un bucle `while`, el cliente va a seguir recibiendo mensajes hasta que el servidor cierre la conexión, y al cambiar el número de bytes a recibir por un número menor al que ocupa el mensaje, ocurre que este se segmenta en partes del tamaño escrito, se reciben consecutivamente por el cliente, y luego cierra la conexión. Dependiendo del número de bytes a recibir es posible que el programa reciba caracteres a medias, dando en una impresión del mensaje un tanto extraña. Aquí un ejemplo:

```
Received message: Hola!  
Bytes received: 1  
Received message: Soy  
Bytes received: 1  
Received message: el se  
Bytes received: 1  
Received message: rvido  
Bytes received: 1  
Received message: r :)  
Bytes received: 1  
Counter: 5
```

Apartado 3

Para que varios clientes se pudieran conectar y el servidor los atendiera secuencialmente tuvimos que cambiar el bucle, añadimos el `accept()` a dentro de un bucle junto al resto de comandos relacionados con el recibir y mandar mensajes, que ya estaban dentro de un bucle. Al salir del bucle de recibir y mandar mensajes, vuelve a repetir el `accept()`, así que puede luego aceptar al segundo cliente y realizar las acciones consecuentes.