

Documentación primer proyecto

Índice

Compilación y ejecución.....	1
Metodología de pruebas.....	2
Pruebas con los servidores.....	3
FIFO.....	3
FORK.....	3
THREAD.....	4
PRE-THREAD.....	4
Conclusiones.....	5

Compilación y ejecución

Para poder correr el siguiente proyecto, se requiere un sistema operativo basado en UNIX, donde se posea GCC (GNU C Compiler). Se debe abrir una terminal y direccionarlo a la carpeta del proyecto.

Para compilar los servidores, se debe ir a la carpeta Servers y se debe digitar lo siguiente para cada servidor a compilar:

- **FIFO:** gcc -W -Wall -o fifo fifo.c
- **FORK:** gcc -W -Wall -o fork fork.c
- **THREAD:** gcc -W -Wall thread.c -pthread -o thread
- **PRETHREAD:** gcc -W -Wall pre-thread.c -pthread -o prethread

THREAD y PRETHREAD necesitan una flag extra (-pthread) para poder compilar satisfactoriamente.

Para ejecutar los servidores, se debe digitar lo siguiente:

- **FIFO:** ./fifo
- **FORK:** ./fork
- **THREAD:** ./thread
- **PRETHREAD:** ./prethread cantidadDeThreads (Este debe ser un número)

Para compilar el cliente, se debe ir a la carpeta Client y se debe digitar lo siguiente para compilar el cliente:

- gcc -W -Wall -o client client.c -pthread

Para ejecutar el cliente, se debe digitar lo siguiente:

- ./client ipADireccionar puerto archivo(s) (El primer parámetro debe tener formato de una IP (XXX.XXX.XXX.XXX), el segundo debe ser un número entero y el tercer es un string con los archivos que desea extraer, separados por una coma y sin espacios).

Metodología de pruebas

La metodología para probar cada servidor consiste en lo siguiente:

- Se utilizará un archivo formato .txt (Archivo sencillo de texto), un archivo formato .html (Un archivo de markup language) y un archivo formato .jpg (Archivo de imagen).
- Se probará transferir cada archivo desde el servidor al cliente. Se probará transferir primero todos los archivos juntos y después, se harán solicitudes por separado.
- Por último, se pedirá un archivo no perteneciente al server, para observar su manejo ante peticiones de archivos no existentes.

Los archivos que se probarán serán:

- text.txt
- index.html

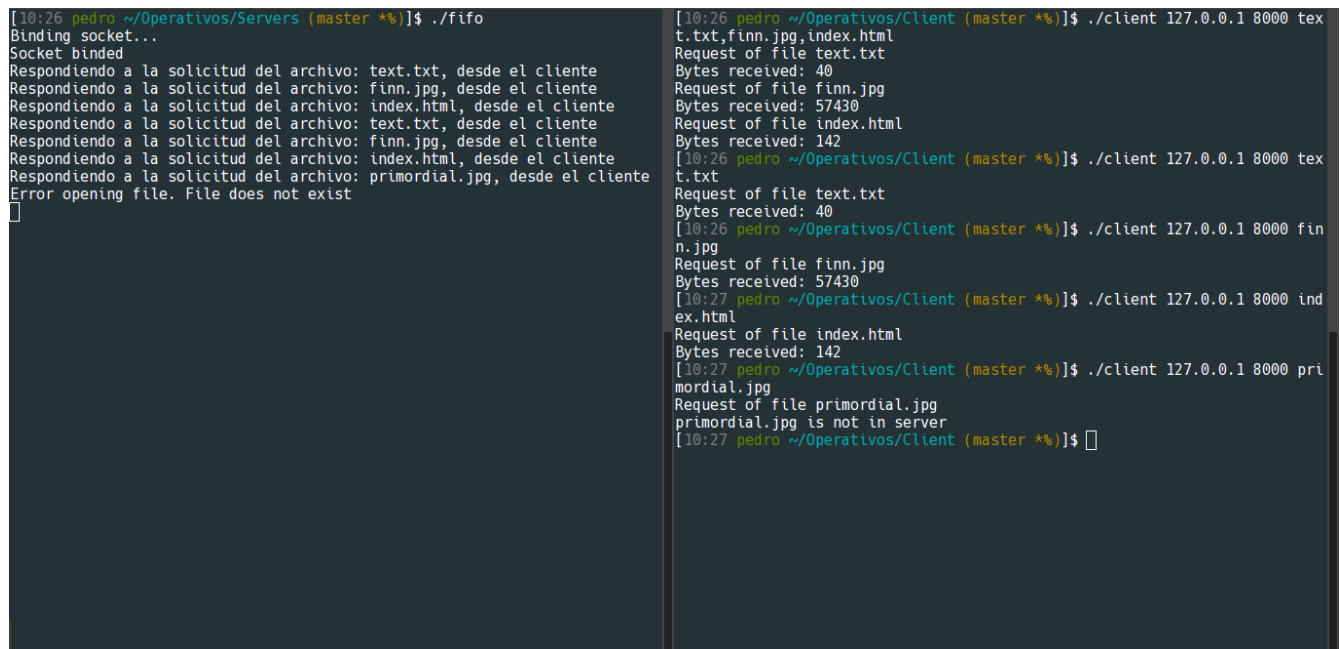
- finn.jpg
- primordial.jpg (no existe)

Como último detalle, se mostrará una imagen de la operación hecha en consola. Además, se soportarán archivos menores a 20Mb, por cuestiones del buffer.

Pruebas con los servidores

- **FIFO**

Este es el servidor más sencillo, porque la estructuración de este es el de un cliente servidor común. Al crearse el socket de servidor, se hace el procedimiento de conectarlo (bind) y luego, se espera a que el cliente mande la solicitud para verificar el archivo y que lo envíe.



```
[10:26 pedro ~/Operativos/Servers (master %)]$ ./fifo
Binding socket...
Socket binded
Respondiendo a la solicitud del archivo: text.txt, desde el cliente
Respondiendo a la solicitud del archivo: finn.jpg, desde el cliente
Respondiendo a la solicitud del archivo: index.html, desde el cliente
Respondiendo a la solicitud del archivo: text.txt, desde el cliente
Respondiendo a la solicitud del archivo: finn.jpg, desde el cliente
Respondiendo a la solicitud del archivo: index.html, desde el cliente
Respondiendo a la solicitud del archivo: primordial.jpg, desde el cliente
Error opening file. File does not exist
[10:26 pedro ~/Operativos/Client (master %)]$ ./client 127.0.0.1 8000 text.txt
Request of file text.txt
Bytes received: 40
[10:26 pedro ~/Operativos/Client (master %)]$ ./client 127.0.0.1 8000 finn.jpg
Request of file finn.jpg
Bytes received: 57430
[10:26 pedro ~/Operativos/Client (master %)]$ ./client 127.0.0.1 8000 index.html
Request of file index.html
Bytes received: 142
[10:26 pedro ~/Operativos/Client (master %)]$ ./client 127.0.0.1 8000 primordial.jpg
Request of file primordial.jpg
primordial.jpg is not in server
[10:27 pedro ~/Operativos/Client (master %)]$
```

- **FORK**

En este servidor, se debe tomar en cuenta que ahora el proceso padre crea procesos hijos para poder atender las transacciones solicitadas. Por lo tanto, el servidor imprime el Process ID (PID) de cada proceso, sea padre o hijo.

<pre>[10:31 pedro ~/Operativos/Servers (master %)]\$./fork Binding socket... Socket binded Id de proceso padre: 7370 Id de proceso hijo: 7381, del padre: 7370 para responder la solicitud del archivo: text.txt Id de proceso hijo: 7383, del padre: 7370 para responder la solicitud del archivo: finn.jpg Id de proceso hijo: 7385, del padre: 7370 para responder la solicitud del archivo: index.html Id de proceso hijo: 7410, del padre: 7370 para responder la solicitud del archivo: text.txt Id de proceso hijo: 7430, del padre: 7370 para responder la solicitud del archivo: finn.jpg Id de proceso hijo: 7461, del padre: 7370 para responder la solicitud del archivo: index.html Id de proceso hijo: 7482, del padre: 7370 para responder la solicitud del archivo: primordial.jpg Error opening file. File does not exist □</pre>	<pre>[10:38 pedro ~/Operativos/Client (master %)]\$./client 127.0.0.1 8000 tex t.txt,finn.jpg,index.html Request of file text.txt Bytes received: 40 Request of file finn.jpg Bytes received: 57430 Request of file index.html Bytes received: 142 [10:39 pedro ~/Operativos/Client (master %)]\$./client 127.0.0.1 8000 tex t.txt Request of file text.txt Bytes received: 40 [10:39 pedro ~/Operativos/Client (master %)]\$./client 127.0.0.1 8000 fin n.jpg Request of file finn.jpg Bytes received: 57430 [10:39 pedro ~/Operativos/Client (master %)]\$./client 127.0.0.1 8000 ind ex.html Request of file index.html Bytes received: 142 [10:39 pedro ~/Operativos/Client (master %)]\$./client 127.0.0.1 8000 pri mordial.jpg Request of file primordial.jpg primordial.jpg is not in server [10:39 pedro ~/Operativos/Client (master %)]\$ □</pre>
--	---

- **THREAD**

Este servidor funciona a base de hilos, por lo que cuando se recibe la solicitud, se debe invocar a una función encargada de hacer la transferencia y asignar la ejecución de dicha función al hilo a crearse (esto por el funcionamiento de los POSIX pthreads).

<pre>[10:41 pedro ~/Operativos/Servers (master %)]\$./thread Binding socket... Socket binded Se solicito el archivo: text.txt Se solicito el archivo: finn.jpg Se solicito el archivo: index.html Se solicito el archivo: text.txt Se solicito el archivo: finn.jpg Se solicito el archivo: index.html Se solicito el archivo: primordial.jpg Error opening file. File does not exist □</pre>	<pre>[10:42 pedro ~/Operativos/Client (master %)]\$./client 127.0.0.1 8000 tex t.txt,finn.jpg,index.html Request of file text.txt Bytes received: 40 Request of file finn.jpg Bytes received: 57430 Request of file index.html Bytes received: 142 [10:42 pedro ~/Operativos/Client (master %)]\$./client 127.0.0.1 8000 tex t.txt Request of file text.txt Bytes received: 40 [10:42 pedro ~/Operativos/Client (master %)]\$./client 127.0.0.1 8000 fin n.jpg Request of file finn.jpg Bytes received: 57430 [10:42 pedro ~/Operativos/Client (master %)]\$./client 127.0.0.1 8000 ind ex.html Request of file index.html Bytes received: 142 [10:42 pedro ~/Operativos/Client (master %)]\$./client 127.0.0.1 8000 pri mordial.jpg Request of file primordial.jpg primordial.jpg is not in server [10:43 pedro ~/Operativos/Client (master %)]\$ □</pre>
---	---

- **PRE-THREAD**

Pre-thread funciona de la misma manera que con el thread, pero este posee un número predeterminado de threads creados para poder manejar las solicitudes que están por

llegar. Entonces, si el thread está activo, se busca otro que se encuentre inactivo para hacer la transferencia.

```
[10:46 pedro ~/Operativos/Servers (master %)]$ ./prethread 5
Binding socket...
Socket binded

Se solicito el archivo: text.txt
Se solicito el archivo: finn.jpg
Se solicito el archivo: index.html
Se solicito el archivo: text.txt
Se solicito el archivo: index.html
Se solicito el archivo: finn.jpg
Se solicito el archivo: primordial.jpg
Error opening file. File does not exist
[]

[10:46 pedro ~/Operativos/Client (master %)]$ ./client 127.0.0.1 8000 tex
t.txt,finn.jpg,index.html
Request of file text.txt
Bytes received: 40
Request of file finn.jpg
Bytes received: 57430
Request of file index.html
Bytes received: 142
[10:48 pedro ~/Operativos/Client (master %)]$ ./client 127.0.0.1 8000 tex
t.txt
Request of file text.txt
Bytes received: 40
[10:48 pedro ~/Operativos/Client (master %)]$ ./client 127.0.0.1 8000 ind
ex.html
Request of file index.html
Bytes received: 142
[10:48 pedro ~/Operativos/Client (master %)]$ ./client 127.0.0.1 8000 fin
n.jpg
Request of file finn.jpg
Bytes received: 57430
[10:48 pedro ~/Operativos/Client (master %)]$ ./client 127.0.0.1 8000 pri
mordial.jpg
Request of file primordial.jpg
primordial.jpg is not in server
[10:48 pedro ~/Operativos/Client (master %)]$ []
```

Conclusiones

- Al principio, la integridad en la transferencia era violada por el hecho de mandar el archivo en pequeños paquetes desde el servidor hasta el cliente. Habían veces que se transferían menos bytes de lo que se debería.
- Destacar que el tamaño del buffer de transferencia, que se utilizará dentro de la función de envío de datos, influirá bastante en la transmisión, considerando el punto anterior, ya que si el buffer es pequeño, puede realizar transacciones más rápidas, pero con un riesgo mayor de inconsistencias. Si el buffer es mayor, podría enviar todo el archivo directamente (o simplemente más contenido), pero con una menor velocidad.
- Los servidores en prethread necesitan tener un mecanismo de validación de hilos para guardar la solicitud mientras todos los hilos están en actividad, ya que podría perderse la solicitud si no hubiera un mecanismo de ese tipo.
- Se recomienda usar un buffer por separado para confirmar si un archivo está disponible en el servidor o no, esto porque compartir un buffer para transferencia podría “ensuciarlo” y causar disparidad en la integridad del archivo transferido.

