Instituto Tecnológico de Costa Rica

Principio de Sistemas Operativos

Prof. Erika Marín

Estudiantes:

Ariana Bermúdez Venegas.

2015019596

Ximena Bolaños Fonseca.

2015073844

Grupo #1

II Semestre

18 de setiembre del 2017

Documentación Primera Tarea Programada

Introducción

El siguiente documento, consiste en la documentación sobre lo que es la primera tarea programa de principios de sistemas operativos. En donde, se presenta un análisis de resultados sobre lo que salió bien y mal en su desarrollo; casos de prueba en donde se representa que se va a utilizar para la comprobación del buen funcionamiento del programa; experiencias que consiste en qué tal se vio la tarea programada; una breve investigación sobre los hilos, sus bibliotecas y plataformas; el funcionamiento de la tarea programa y su compilación, por último las conclusiones sobre que se aprendió en la tarea programada y como se puede dar un mejoramiento para la próxima ocasión.

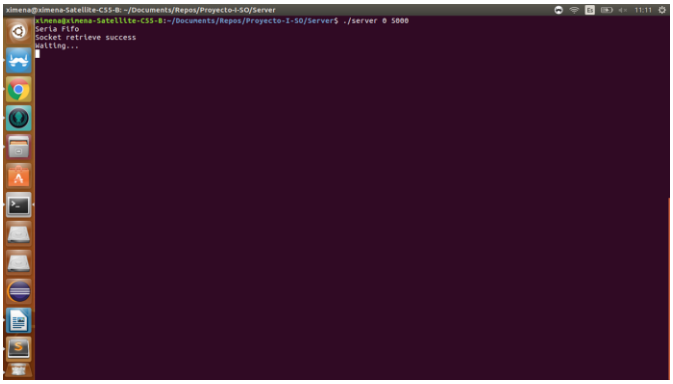
Análisis de Resultados del Programa

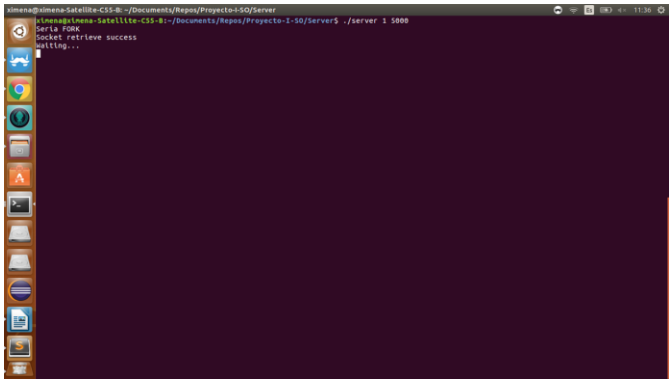
Para ver la funcionalidad de que sirve y qué no sirve en la tarea programada, se harán un listado de aspectos fundamentales para el funcionamiento del proyecto. Además, de un cierto porcentaje de completitud de cada uno de estos aspectos.

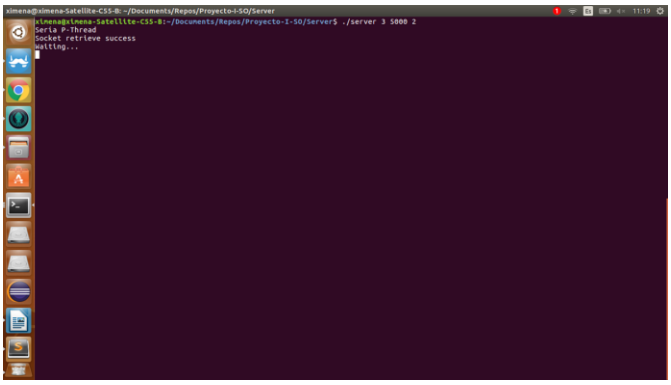
Aspecto a Implementado	**Completitud**
Conexión del Servidor	100%
Conexión con el Cliente	100%
Conexión con el Browser	100%
Envío de Archivo al Cliente	100%
Envío de Archivo al Browser	100%
Implementación del FIFO	100%
Implementación del FORK	100%
Implementación del Thread	100%
Implementación del P-Thread	100%
Mensajes de Error	100%

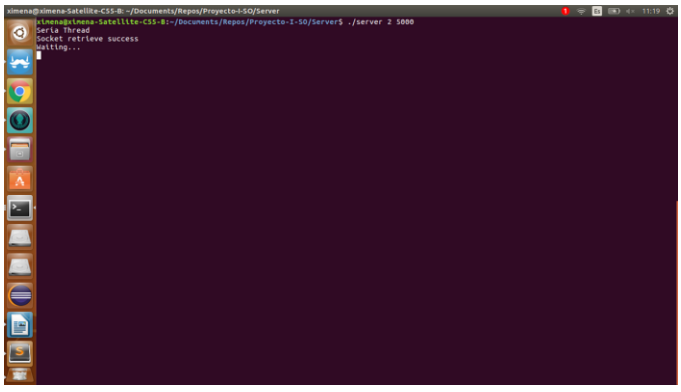
Casos de Prueba

En primera instancia se probó lo que era que el servidor sirviera en lo que son los diferentes casos, ya sea en Fifo, Fork, Thread y Pthread, para comprobar que realmente se conectara con cada uno de estos.

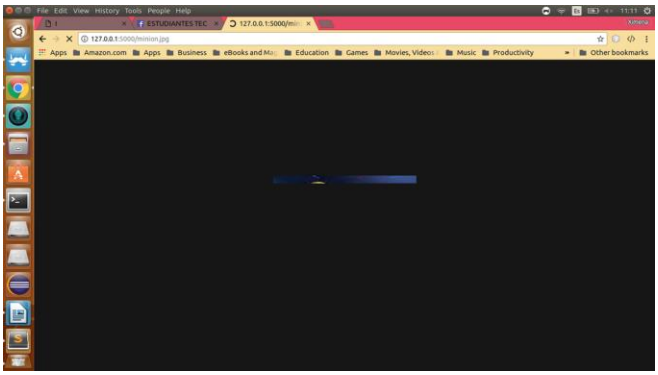


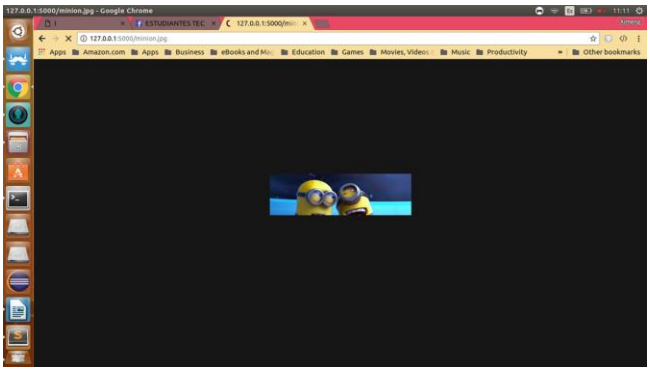


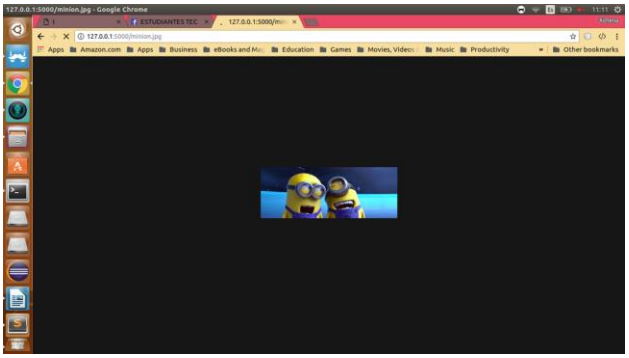


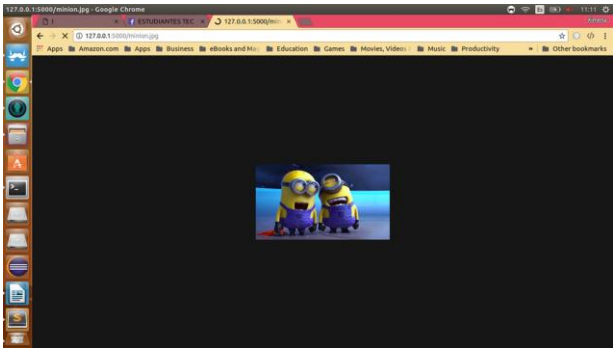


Una vez que se denoto que si se conectaba cada uno de los servidores de diferente manera con cada uno se probó que sirviera el envío de archivos a través del navegador. Esta se puede apreciar con las imágenes siguientes:

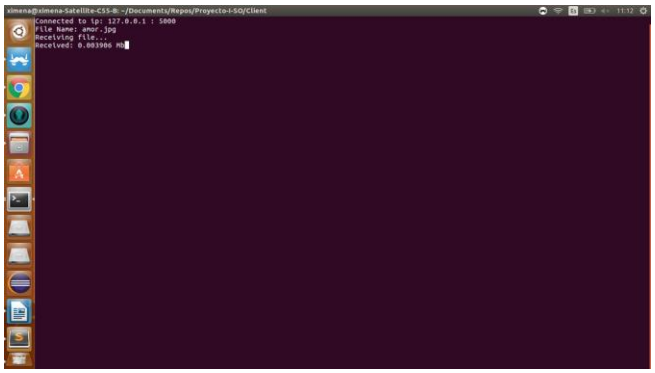


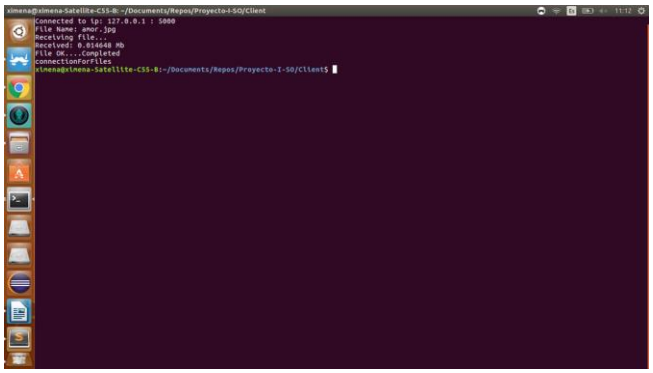




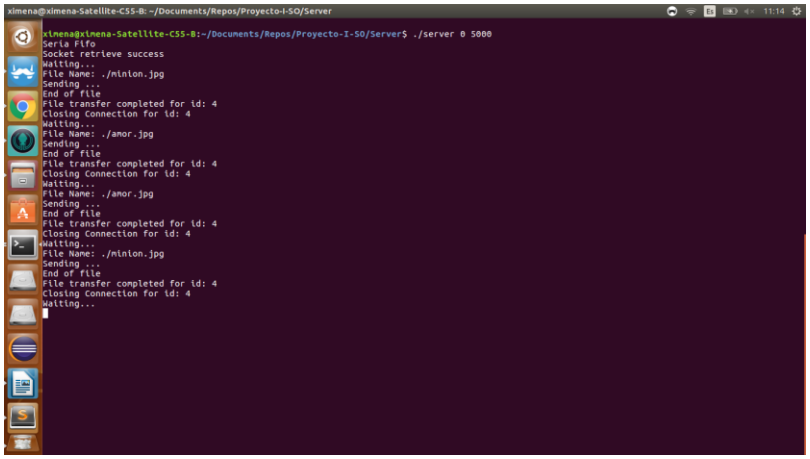


En caso de ser por medio del cliente se puede apreciar de la siguiente manera. Mostrando el modo de ejecución, mientras carga y una vez finalizado el proceso de envío.

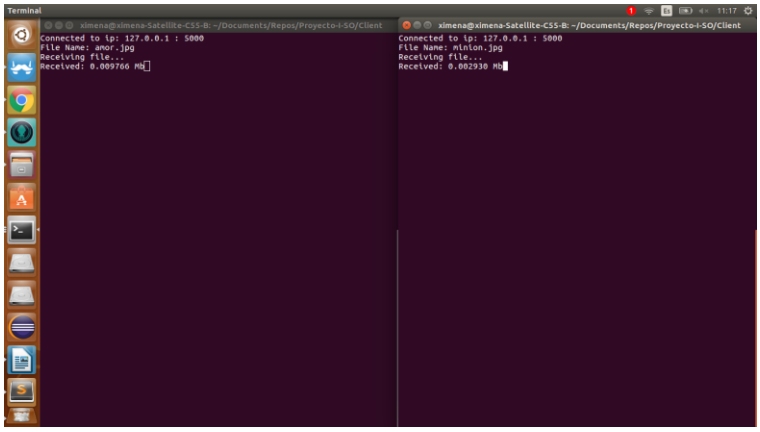




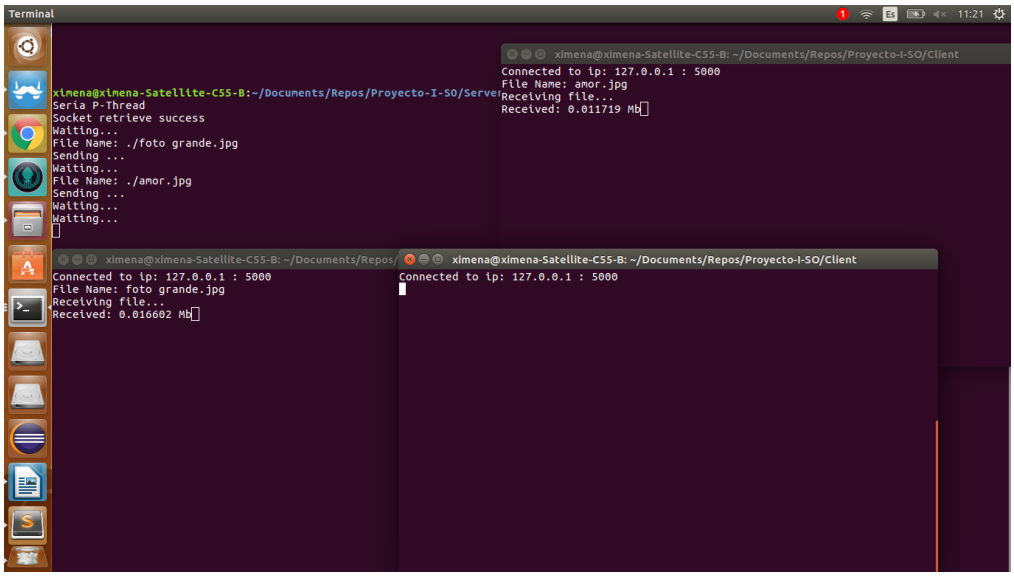
En el servidor ambos procesos se verían mostrando el el puerto y el número de conexión que sería, el archivo que se va a enviar y una vez finalizado cada proceso se manda un mensaje.



En caso de querer llamar varios archivos desde el cliente, se debe de poner el nombre de los archivos separados por comas, en caso de ser Fork, Thread o Pthread se van a ver como corren en paralelo los procesos de la siguiente manera. En caso de ser Fifo va de uno en uno dejando en espera los más nuevos.



En el caso de ser Pthread o FIFO la espera se mostraría de la siguiente manera, en donde se ven que dos corren a la vez, pero el otro espera, ya que solo se han activado dos hilos en el pool.



Experiencias

Entre lo que se puede decir que es mejor sería el PThread como servidor debido a que se tiene un límite establecido por la persona y no gasta más memoria de lo establecido desde el inicio. Además, estos pueden compartir memoria. Se ve donde cada uno de estos trabajan en paralelo sin afectar al otro. A diferencia del Thread y Fork no crea por cada uno un hilo o un proceso por cada petición, si no que ya viene preestablecido la cantidad de hilos que van a trabajar. El Fifo por otro lado no puede trabajar en paralelo, ya que trabaja como una cola.

Entre lo que fue la implementación del Fifo fue bastante sencilla, ya que el servidor por default trabaja de esa manera.

Para lo que es el Thread se trabajó con la librería “pthread.h”. Esta implementación fue realmente sencilla al igual que el Fork pero en este caso con la librería “unistd.h”.

El más complicado fue el Pthread porque se debían de crear los hilos y estarlos bloqueando y desbloqueando. Para ello se utilizó la librería “pthreadpool.h”

Investigación

La creación de hilos y procesos en cualquier sistema operativo de Unix, se realiza de una manera muy sencilla gracias a la librería “pthead.h”. Actualmente, es la librería más común para lo que son la creación de hilos en Linux. Este al ser de Unix permite su uso tanto en computadoras Mac como las que contienen Linux. Por otro lado, se tiene a Windows de una manera independiente y para la programación de lo que son hilos de Windows se utiliza el API y se llamaría por medio de un “#include <windows.h>”. Es por ello por lo que hay que tener mucho cuidado con la plataforma que se está utilizando para el uso de estos en C.

Compilación y Ejecución

Para la ejecución y compilación del programa se debe de en primera instancia manejarse por medio del sistema operativo Linux, en la carpeta tanto del cliente como del servidor se debe de abrir la terminar y poner en la línea de comando “make” para la creación del ejecutable. Una vez creados los ejecutables primero se debe de ejecutar el servidor y luego los clientes y el browser. Para su ejecución se debe:

- Servidor: se debe de abrir una terminal desde la carpeta de ejecución, poner en la línea de comando “./server #tipo #puerto #threads(únicamente con Pthreads)”
    - Tipo:
        - FIFO:0 y por default.
        - FORK:1
        - THREAD:2
        - PTHREAD:3
- Cliente: se debe de abrir una terminal desde la carpeta de ejecución, poner en la línea de comando “./client ip #puerto archivo ”
- Browser: se debe de ingresar en el navegador “http://#ip:#puerto/archivo”

Conclusiones

Se puede concluir que el manejo de la memoria es realmente importante a la hora de manejar tanto hilos como procesos, ya que, si hay dos hilos que utilizan un mismo “buffer” de memoria para el envío de una imagen, este puede alterar completamente el resultado.

Se pudo denotar desde un manejo muy sencillo lo que es el funcionamiento de un sistema operativo a la hora de transmitir archivos o bien un poco del funcionamiento de la nube.

Finalmente, se puede concluir que a la hora de trabajar con servidores, procesos e hilos se debe de tener mucho orden y tener una idea clara de que es lo que se ocupa enviar para no sobrecargar el servidor.

Referencias

UM, 2012. *Procesos e Hilos en C.* Recuperado de http://www.um.es/earlyadopters/actividades/a3/PCD_Activity3_Session1.pdf

Rodríguez G., 2014. *Programación Avanzada.T3-Uso de Threads en C.* Recuperado de https://www.driverlandia.com/programacion-c-avanzada-t3-uso-de-threads-en-c/