Proyecto 02 - Principio de Sistemas Operativos IC6600

Integrantes


```
David Jose Espinoza Soto - 2016012024
```


Introducción

Un problema que se presenta con el servicio FTP (file transfer protocol), utilizado tradicionalmente para la transferencia de archivos entre máquinas, es la tarea a veces complicada de instalar un programa servidor en la máquina a la que se desean transferir/recuperar archivos (modelo cliente-servidor), ya que se requieren privilegios de administrador.

Un enfoque más sencillo para implementar un servicio FTP sería simplemente que dos programas clientes se comuniquen entre sí y transfieran archivos entre ellos sin necesidad de instalar un servidor (modelo peer-to-peer).

En este proyecto se pretende desarrollar un cliente FTP que funcione de ese modo, es decir, que permita transferir y recibir archivos entre programas clientes que se encuentren instalados en máquinas remotas.

Comandos ftp

El programa btfp reconocerá un subconjunto de instrucciones de FTP que permitirán la conexión y transferencia de archivos:

```
open <dirección-ip>: establece una conexión remota
close: cierra la conexión actual
quit: termina el programa
cd <directorio>: cambia de directorio remoto
get <archivo>: recupera un archivo remoto
lcd <directorio>: cambia de directorio local
ls: lista los archivos del directorio remoto
put <archivo>: envía un archivo a la máquina remota
pwd: muestra el directorio activo remoto
```

La transferencia de archivos realizará una copia del archivo en cuestión desde la máquina remota hasta la máquina local (o viceversa) almacenando éste con el mismo nombre que se encontraba en la otra máquina.

Modo de operación

El programa bftp se debe ejecutar en al menos dos máquinas al mismo tiempo, y quedarán esperando conexiones en el puerto 8889. Posteriormente, permitirá que el usuario ingrese comandos en la terminal y los ejecutará. Note que un usuario solo puede establecer una conexión con una máquina a un tiempo, pero múltiples máquinas remotas pueden establecer conexión con la máquina de dicho usuario al mismo tiempo.

Internamente cada programa se deberá comportar con un servidor y un cliente al mismo tiempo. Por eso se debe utilizar un mecanismo (tareas) que le permita al programa estar haciendo varias cosas al mismo tiempo: recibir comandos del usuario, atender todas las conexiones entrantes y ejecutar sus comandos. El programa debe utilizar tanto sockets clientes como un socket servidor para realizar las conexiones adecuadamente.

Estructuras

Originalmente tenia planeado crear un servidor que tambien funcionara de cliente pero tuve complicación, ya que mientras esta esperando por la entrada de datos, el servidor no permite otra interacción, asi que pense hacer un servidor que tubiera las opciones de los comandos btfp y el cliente tendria las opciones de lo que puede escoger y le pasa al servidor el numero y una cadena de caracter.

El servidor tendria una cola con n hilos para atender a un maximo de n clientes.

Concluciones

Al final no pude avanzar mucho, tuve problemas para arracar los ejemplos de sockets y se me fue el tiempo en otros cursos, necesito gestionar mejor mi tiempo de desarrollo, adjunto le envio los codigos de ejemplo que estaba usando. EL unico que logre hacer funcionar fueron example-socket06.c y example-socket07.c.