Sistemas Operativos

Formulario de auto-evaluación

Modulo 2. Sesión 7. Construcción de un spool de impresión

Nombre y apellidos:

Néstor Rodríguez Vico

a) Cuestionario de actitud frente al trabajo.

El tiempo que he dedicado a la preparación de la sesión antes de asistir al laboratorio ha sido de ….60.. minutos.

1. He resuelto todas las dudas que tenía antes de iniciar la sesión de prácticas: …Si… (si/no). En caso de haber contestado “no”, indica los motivos por los que no las has resuelto:

2. Tengo que trabajar algo más los conceptos sobre:

3. Comentarios y sugerencias:

b) Cuestionario de conocimientos adquiridos.

El código de mi programa **servidor.c** ha sido:

```
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/wait.h>
#include <fcntl.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <signal.h>

static void signal_handler(int sigNum){
	pid_t pid;
	// Capturar pid del hijo y eliminar su archivo fifo asociado.
	pid = wait(NULL);
}

int main(int argc, char *argv[]){
	int fde, fds, fdc, tmp_cli, leidos, proxypid;
	char nombrefifoe[50], nombrefifos[50], fifoproxy[50];
	pid_t pid;

	// Comprobar uso correcto
	if(argc != 2) {
		printf("Uso: ./servidor <nombre_fifo>");
		exit(-1);
	}

	signal_handler(SIGCHLD);
```

```
// Darle nombre a los fifos
sprintf(nombrefifoe,"%se",argv[1]);
sprintf(nombrefifos,"%ss",argv[1]);

// Borrar los archivos fifo por si existieran
unlink(nombrefifoe);
unlink(nombrefifos);

//Crear los fifos dandoles permisos
umask(0);
mkfifo(nombrefifoe, 0666);
mkfifo(nombrefifos, 0666);

// Abrir los fifos en RDWR
if((fde = open(nombrefifoe, O_RDWR)) == -1){
        perror("Error al abrir el fifo de entrada");
        exit(1);
}

if((fds = open(nombrefifos, O_RDWR)) == -1){
        perror("Error al abrir el fifo de salida");
        exit(1);
}

// Crear archivo bloqueo que usan los proxys
if((fdc=open("bloqueo", O_RDWR|O_CREAT, S_IRWXU)) == -1){
        perror("Error al crear el archivo de bloqueo.");
        exit(1);
}
```

```
// Leemos de nombrefifoe y se lanza un hijo
// que será el proxy para este cliente
while((leidos = read(fde, &tmp_cli, sizeof(int))) != 0){
        // Creo un hijo
        pid = fork();

        // Soy hijo
        if(pid == 0){

                // Obtener pid del nuevo proceso
                proxypid = getpid();

                sprintf(fifoproxy, "fifo.%d", proxypid);
                umask(0);
                mkfifo(fifoproxy, S_IRWXU);

                // Escribe el pid del proxy en fifos
                write(fds, &proxypid, sizeof(int));

                // Abrir fifo para lectura
                int fifo = open(fifoproxy, O_RDONLY);

                // Redirigir entrada a archivofifo
                dup2(fifo, STDIN_FILENO);

                // Ejecutar ./proxy
                execlp("./proxy", "proxy", NULL);
                exit(0);
        }
}
```

```
	unlink(nombrefifoe);
	unlink(nombrefifos);
	unlink("bloqueo");

	return(0);
}
```

El código de mi programa **proxy.c** ha sido:

```
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>

void bloqueardesbloquear(int entero, int orden){
	struct flock cerrojo;
	cerrojo.l_type = orden;
	cerrojo.l_whence = SEEK_SET;
	cerrojo.l_start = 0;
	cerrojo.l_len = 0;
	if(fcntl(entero, F_SETLKW, &cerrojo) == -1){
		perror("Error al bloquear en proxy.");
		exit(1);
	}
}
```

```
int main(int argc, char *argv[]){
	char buffer[1024];
	int dbloqueo, nbytes;
	FILE *tmp = tmpfile();
	char fifoproxy[256];

	//Leemos de STDIN_FILENO y escribimos en el temporal
	while((nbytes = read(STDIN_FILENO,buffer,1024)) > 0){
		fwrite(buffer,sizeof(char),nbytes,tmp);
	}

	//Ganar EM en pantalla
	if ((dbloqueo = open("bloqueo",O_RDWR)) == -1){
		perror("Error al abrir bloqueo.");
		exit(1);
	}

	//Bloquear pantalla
	bloqueardesbloquear(dbloqueo, F_WRLCK);

	//Leer de temporal y escribir en la pantalla
	while(!feof(tmp)){
		//Leemos de temporal
		fread(buffer,sizeof(char),1024,tmp);
		//Escribimos en pantalla
		write(STDOUT_FILENO,buffer,1024);
	}

	//Desbloqueamos
	bloqueardesbloquear(dbloqueo, F_UNLCK);
```

```
        // Eliminar fifo
        sprintf(fifoproxy,"fifo.%d", getpid());
        unlink(fifoproxy);

        exit(0);
}
```