

## TAREA # 3

### ARQUITECTURA CLIENTE-SERVIDOR

#### ACTIVIDAD 1

Del sitio web del Tema 1 toma los programas 20 y 21

[http://profesores.fi-b.unam.mx/carlos/acs/Tema-01-Linux\\_y\\_Procesos/programa20\\_myecho.c](http://profesores.fi-b.unam.mx/carlos/acs/Tema-01-Linux_y_Procesos/programa20_myecho.c)

[http://profesores.fi-b.unam.mx/carlos/acs/Tema-01-Linux\\_y\\_Procesos/programa21\\_execve.c](http://profesores.fi-b.unam.mx/carlos/acs/Tema-01-Linux_y_Procesos/programa21_execve.c)

- a) comenta detalladamente ambos programas
- b) comprueba que el programa 20 (myecho) se puede ejecutar desde linea de comandos, pasando los argumentos que gustes.

```
132.248.59.6.tlp - carlos@132.248.59.6:22 - Bitvise xterm - carlos@odin:~/sitio_web/acs/Tema-01-Linux_y_Procesos
[carlos@odin Tema-01-Linux_y_Procesos]$ ./programa20_myecho hola mundo buen dia
argv[0]: ./programa20_myecho
argv[1]: hola
argv[2]: mundo
argv[3]: buen
argv[4]: dia
[carlos@odin Tema-01-Linux_y_Procesos]$
```

- c) En una segunda ejecución, pasa tu nombre en formato apellidopaterno apellidomaterno nombre(s). Por ejemplo:

```
132.248.59.6.tlp - carlos@132.248.59.6:22 - Bitvise xterm - carlos@odin:~/sitio_web/acs/Tema-01-Linux_y_Procesos
[carlos@odin Tema-01-Linux_y_Procesos]$ ./programa20_myecho Perez Perez Juan
argv[0]: ./programa20_myecho
argv[1]: Perez
argv[2]: Perez
argv[3]: Juan
[carlos@odin Tema-01-Linux_y_Procesos]$
```

- d) Al comentar el programa 21, modifica la llamada a exec para que quede en un bloque if, como lo vimos en los programas de clase, con una sentencia exit(EXIT\_FAILURE):

```
// ESTA SENTENCIA ES MUY IMPORTANTE!
// QUE HACE ESTA SENTENCIA?
newargv[0] = argv[1];

execve( argv[1], newargv, NULL );
// AUNQUE LA SIG SENTENCIA DICE QUE NO SE EJECUTA,
// ASI COMO ESTA EL CODIGO, SI PODRIA EJECUTARSE, SI EL execve FALLA
// MODIFICA ESTE BLOQUE PARA QUE AUNQUE FALLE LA LLAMADA A execve
// LA SIGUIENTE SENTENCIA NO SE EJECUTE
printf("Esta sentencia no se ejecuta\n");
perror("execve");
exit( EXIT_FAILURE );
}
```

e) Ejecuta el programa 20 llamándolo desde el programa 21

```
./programa21_execve    ./programa20_myecho
```

Agrega la captura de pantalla de la ejecución

## ACTIVIDAD 2

Toma el programa 23 y coméntalo detalladamente:

[http://profesores.fi-b.unam.mx/carlos/acs/Tema-01-Linux\\_y\\_Procesos/programa23\\_fork-exec.c](http://profesores.fi-b.unam.mx/carlos/acs/Tema-01-Linux_y_Procesos/programa23_fork-exec.c)

En la función spawn modifica el bloque de la función execvp para que quede en un bloque if y termine con exit(EXIT\_FAILURE):

```
int spawn (char* program, char** arg_list)
{
    pid_t child_pid;
    child_pid = fork ();
    if (child_pid != 0)
        return child_pid;
    else
    {
        // RECUERDEN: Si execvp() se ejecuta exitosamente
        // el proceso "parent" termina en ese momento,
        // pero termina con éxito
        execvp (program, arg_list);
        printf ("ocurrió un error al ejecutar execvp(...)\n");
        // alternativa al printf:
        // fprintf (stderr, "ocurrió un error al ejecutar execvp(...)\n");
        return 1; // termina con error
    }
}
```

Compila y ejecuta. Agrega la captura de pantalla de la ejecución.