# Laboratorio II – parte 2

## Instrucciones

- Esta práctica deberá de ser realizada en los mismos grupos con los que se elaboró el laboratorio II.
- En cuanto a requerimientos, se utilizará la misma máquina virtual que en el laboratorio II pero con el procedimiento de compilación ya terminado.
- Para calificar la práctica, cada integrante del grupo deberá de mostrar su avance hasta completar esta práctica, 10 minutos antes de que termine el período de clase acompañado de las preguntas de este laboratorio, debidamente respondidas (horario en el que cierra el portal).
- Nota: capturas de pantalla en este laboratorio han sido tomados usando Ubuntu 17.04.

## Recomendaciones previas a la práctica:

Tener actualizado el sistema

Tener GCC instalado, libncurses5-dev, libncursesw5-dev, haber terminado el proceso de compilación del kernel.

## Parte II – Uso de la nueva system call

Para la realización de esta practica se necesitará tener un kernel ya recompilado con los pasos del laboratorio anterior. Para saber que el proceso ha terminado, el proceso de compilación devuelve el control en la terminal:

```
root@jeffOS:/usr/src/linux-4.10.13#
```

Instalar el kernel en el sistema operativo actual:

```
root@jeffOS:/usr/src/linux-4.10.13# make modules_install install
```

Al finalizar, se mostrará un mensaje de done y nuevamente devolverá el control sobre la terminal.

Reiniciar la máquina virtual:

```
root@jeffOS:/usr/src/linux-4.10.13# shutdown -r now
```

Una vez reiniciada, el kernel deberá de ser el que se había descargado al usar “uname -r”.

Cree una nueva carpeta y cree un documento .C en ella con el siguiente código:

```
root@jeffOS:~# cd Documents/
root@jeffOS:~/Documents# mkdir syscall_test
```

```c
#include <stdio.h>
#include <linux/kernel.h>
#include <sys/syscall.h>
#include <unistd.h>

int main() {
        long int s = syscall(332);
        printf("Prueba de system Call .: display_msg :. ret %1d \n", s);
        return 0;
}
```

En mi caso, este código se ha guardado en Documents/syscall_test/my_display.c

Tenga en cuenta que en el código se usa “syscall(332)” pero este número es el id que se usó cuando se agregó la systemcall al archivo de las systemcalls en el laboratorio anterior.

Compile con gcc:

```
root@jeffOS:~/Documents/syscall_test# gcc my_display.c
```

Esto creará un archivo .out

Ejecute el archivo para probar el código:

```
root@jeffOS:~/Documents/syscall_test# ./a.out
```

Con esto podrá ver el mensaje en pantalla que configuró en el código. En este caso el “Prueba de system call”. En el código también se tiene “%1d” return 0. Lo cual es lo que se espera si funciona de forma correcta. En caso no fuere “0” este valor, estaría devolviendo un código de error.

También se puede ver el listado de mensajes de kernel y deberá de salir el llamado que se ha hecho con el código en la instrucción anterior

```
root@jeffOS:~/Documents/syscall_test# dmesg
```

## Parte II – Preguntas

- Por qué se deben de incluir las 4 librerias usadas en el código C de esta practica, que hace cada una?
- En sus palabras, que ha estado haciendo durante la práctica.
- En sus palabras, al hacer una systemcall que haga otras funciones, eso debería de hacerse en el código de la systemcall o en el código C como el que se ha compilado en esta práctica?.