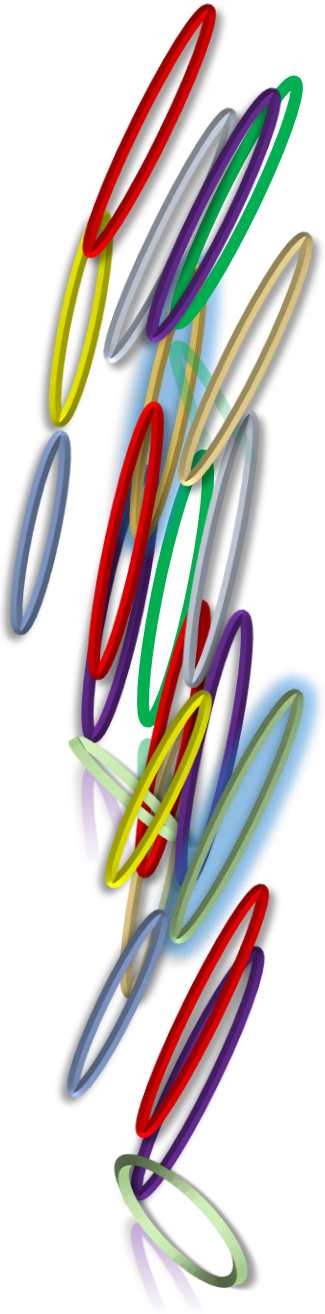

Señales en linux

Abril 2023

Docente: Mgter. Ing. Silvia Carrera

Resultados de aprendizaje:

- ✓ Definir señales para diferenciarlo de otros elementos de Linux.
- ✓ Identificar y caracterizar las señales.
- ✓ Terminología de las señales
- ✓ Envío y recepción de señales para construir código en C

CARACTERIZACIÓN GENERAL DE LAS SEÑALES

Las señales son la forma más simple de comunicación entre procesos que se dispone en Linux. Los procesos las utilizan para comunicarse entre sí, y el kernel las utiliza para comunicarse con cada uno de los procesos.

Una señal es una interrupción que recibe un proceso. ***Las señales son empleadas por el sistema operativo para acusar una situación excepcional en la ejecución de un programa***

Una señal acusa la ocurrencia de un suceso excepcional ejemplo:

- División entre cero.
- Un usuario manda con el teclado la terminación de un programa (Ctrl+C).
- La terminación de un proceso hijo.
- Expiración de un timer de alarma.
- Una llamada a kill.
- Hacer una operación de Entrada/Salida que no puede ser realizada (leer un pipe que no es de escritura).

Los usos de las señales pueden ser múltiples:

- comunicación entre usuario y proceso
- terminar o cancelar la ejecución de un proceso
- entre procesos
- entre sistema y proceso
- ejecución de instrucciones ilegales
- etc.

¿Cómo se identifican las señales?

Cada señal tiene un nombre que comienza con SIG, como ejemplo SIGTERM. Estos nombres se corresponden con constantes enteras positivas, denominadas número de señal, definidas en <signal.h>

¿Qué sucede cuando un proceso recibe una señal?

Pueden ocurrir una de 3 situaciones:

- Puede ignorar la señal.
- Puede permitir que ocurra la acción predeterminada asociada a la señal.
- Puede capturar o interceptar la señal, lo que implica que se ejecute un trozo de código, llamado handler (manipulador) de señal.

¿Cuántas señales hay?

Existen alrededor de 32 señales diferentes en linux, cada una de ellas asociada a un número entero único.

¿Cuántas señales hay?

Nº	Señal	Descripción	Acción
14	SIGALARM	Señal despertadora (timer signal) generada por la función alarm del sistema	Proceso termina
		Otros	

Peculiaridad con las señales

Todas las señales pueden ser ignoradas o bloqueadas, a excepción de SIGSTOP y SIGKILL, que son imposibles de ignorar.

Limitaciones con las señales:

- **no tienen prioridades relativas**

  Por ejemplo, si dos señales llegan al mismo tiempo a un proceso puede que sean tratadas en cualquier orden, no podemos asegurar la prioridad de una en concreto.

- **imposibilidad de tratar múltiples señales iguales**

  si llegan 8 señales SIGCONT a la vez, el proceso funcionará como si hubiera recibido sólo una.

Ejemplo

```
#include <signal.h>
#include <unistd.h>
void atrapa(int);//se encarga de capturar una señal
int main(int argc, char *argv[])
{    int t;      for(t=1;t<=64;t++) //capturamos todas las llamadas, 64
        signal(t, atrapa);
     printf("Identificación de proceso PID: %d\n", getpid() );
     pause(); /*se espera que llegue una señal con la función pause, por ejemplo pulsando control +C
*/
    printf("Continuando...\n");  return 0;}
 void atrapa(int sig)
{    signal(sig, atrapa);
    printf("Recibida la señal: %d\n", sig); // muestra el número 2}
```

BIBLIOGRAFIA:

- Programación en Linux, con ejemplos (año 2000). Autor: kurt Wall. Editorial: Prentice Hall.
- Aprendiendo C++ para Linux en 21 Días. Autores: Jesse Liberty y David B. Horvath. Prentice Hall (2001)

GRACIAS

Curso: 2R3

Docente: Mgter. Ing. Silvia Carrera