SISTEMAS OPERATIVOS II

Tercer curso Ingeniería Informática. Curso 2003-2004

Práctica 4: Procesos en UNIX. Señales.

1. Realizar un programa, *kil*, que recibe como argumentos en nombre de una señal (sin el SIG), el pid de un proceso, el número de veces que ha de enviarle dicha señal (0 para indicarle que se la envie continuamente o hasta que el proceso termine) y (opcionalmente) el retardo (en segundos) entre que se la envía una vez y la siguiente. El programa NO DEBE IMPRIMIR NADA en pantalla. Ejemplo

   ```
   %kil INT 435 100 1
   ```

   Le envia al proceso con pid 435 100 señales SIGINT dejando pasar un segundo después de enviársela cada vez.
   En caso de que no se especifique el cuarto argumento se entenderá que es 0, y NO SE LLAMARA a *sleep* entre una llamada a *kill* y la siguiente.

2. Añadir al shell de la práctica anterior las siguientes funciones de manejo de las señales.

hand1 [-t][-v] SEN Instala un manejador para la señal SEN (sin el SIG). Cada vez que se ejecute el manejador incrementará un contador que indica cuantas veces se ha recibido la señal.

- -t el manejador es temporal (se instala con SA_RESETHAND)
- -v el manejador imprime un mensaje indicando que se ha recibido la señal. (Si no se especifica -v NO DEBE IMPRIMIR NADA).

hand2 [-f][-v] [-p] SEN Instala un manejador para la señal SEN (sin el SIG). Cada vez que se ejecute, el manejador incrementará un contador que indica cuantas veces se ha recibido la señal y además envía al propio proceso la señal para la cual es manejador

- -f el manejador se instala con SA_NODEFER.
- -v el manejador imprime un mensaje que indica que ha recibido la señal, la dirección de parámetro que recibe así como cuantas veces la ha recibido (el valor del contador para esa señal).
- -p el manejador se instala para ser ejecutado en la pila alternativa

pila [bytes] Establece una pila alternativa, de tamaño *bytes* para el manejo de las señales. Si no se especifica *bytes* informa de la pila actual.

maxhand2 [N] Establece el número máximo de veces que hand2 reenvia la señal. Si se especifica 0 lo hace indefinidamente. Si no se especifica N nos informa del valor establecido de N.

ignore SEN1 SEN2... Ignora las señales SEN1 SEN2...

mask [-u] SEN1 SEN2... Enmascara las señales SEN1 SEN2...Con -u las desenmascara.

default SEN1 SEN2... Pone las señales SEN1 SEN2...a su acción por defecto. Si no se especifican señales pone todas

info [SEN1 SEN2...] Para las señales SEN1 SEN2... indica si están enmascaradas, ignoradas, si tienen instalado un manejador, la dirección del manejador y los flags asociados a ese manejador. Si no se especifica ninguna sñal lo indicará para todas. Ejemplo

```
#senales INT HUP SEGV
SIGINT enmascarada manejador SI (0x12450) flags SA_RESETHAND
SIGHUP  enmascarada manejador NO (SIG_DFL) flags
SIGSEGV no enmascarada manejador SI (0x12700) flags SA_NODEFER
```

count [SEN1 SEN2...] Indica cuantas señales SEN1 SEN2...se han recibido. Si no se especifican parametros lo indica de todas.

clear [SEN1 SEN2...] Pone a 0 el contador de las señales SEN1 SEN2...Si no se especifican parámetros los pone todos a 0.

bucle Hace que el shell entre en un bucle infinito. Instala un manejador para SIGINT (dinstindo de los instalados por hand1 y hand2) que permite salir del bucle pulsando control-c para seguir ejecutando el shell (desenmascara SIGINT si es necesario).

fault Produce un fallo de segmentación en el shell. (No vale enviar SIGSEGV, tiene que ser un fallo de segmentación de verdad).

Información detallada de las llamadas al sistema y las funciones de la librería debe obternerse con man (sigaction, sigprocmask, str2sig

Los manejadores deben instalarse con *sigaction* y la señales enmascararse y desenmascararse con *sigprocmask*. SEN representa el nombre de una señal sin el SIG (ejemplo: INT, SEGV, HUP), es decir tal como lo hacen las funciones *strsignal, sig2str y str2sig*. El estado de las señales debe obtenerse con *si-*

gaction y *sigprocmask*, ya que almacenarlo en variables del programa podría hacer que tras una llamada a *fork()* o *exec* no fuese correcto.

Debe tenerse en cuenta que las señales interrumpen las llamadas al sistema (muy notorio en las funciones que leen de teclado) haciendo que éstas devuelvan -1 con errno=EINTR y sin realizar la tarea para la que fueron invocadas. Es necesario por tanto, que en el caso de que una llamada al sistema sea interrumpida por una señal la reiniciemos, bien directamente después de comprobar que ha sido interrumpida o bien indicándole al sistema que lo haga automáticamente (usando el flag adeciado en *sigaction*).

Pueden usarse las utilidades de /usr/proc/bin para comprobar la veracidad de lo reportado por el programa.

FORMA DE ENTREGA

Debe entregarse sólo el código correspondiente al intérprete de comandos. El programa auxiliar *kill* no debe entregarse. El código del intérprete de comandos se entregará como en prácticas anteriores.

FECHA DE ENTREGA VIERNES 28 MAYO 2004