SISTEMAS OPERATIVOS II

Tercer curso Ingeniería Informática . Curso 2008-2009

Práctica 3: Procesos en UNIX: señales

Continuar la codificación de un intérprete de comandos (shell) en UNIX. Al igual que en la práctica anterior:

- Los argumentos entre corchetes [] son opcionales.
- Los argumentos separados por `|` indican que debe ir uno u otro, pero no ambos simultaneamente.
- No debe dilapidar memoria (ejemplo: variable que se asigna cada vez que se llama a una función y no se libera).
- Cuando el shell no pueda ejecutar una acción por algún motivo, debe indicarlo con un mensaje como el que se obtiene con `sys_errlist[errno]` o con *perror()* (por ejemplo, si no puede cambiar de directorio debe indicar por qué).
- En ningún caso debe producir un error de ejecución (segmentation, bus error . . .), salvo que se diga explicitamente
- Las direcciones de memoria deben mostrase en **hexadecimal**.
- La información que se nuestra en pantalla no debe incluir en ningun caso líneas en blanco.
- El shell leerá de su entrada estándar y escribirá en su salida estándar, de manera que podría ser ejecutado un archivo de comandos invocando al shell con su entrada estándar redireccionada a dicho archivo.

En esta práctica el shell incorporará las funciones para comprobar el funcionamiento de los manejadores de señales, así como la posibilidad de enmascararlas e ignorarlas. Todo el manejo de señales se hará con la interface de System V R4 (*sigaction, sigprocmask . . .*). Los nombres de la señales se especificarán sin el SIG. Pueden usarse las funciones que se suministran para convertir entre nombre de una señal y su valor como entero.

`http://www.dc.fi.udc.es/~afyanez/Practicas/sources/senales.c`

En programa ha de llevar un contador para cada señal que indica cuántas veces se ha ejecutado el manejador asociado a dicha señal. En comando *count* muestra o inicializa dicho contador.

Considérese que el número de señales es 32 (aun cuando NSIG es 64 en linux y 48 en solaris).

- **sigstack [-*t tam*] [-*a addr*]**
  Establece una pila alternativa para la ejecución de los manejadores de las señales (los que se instalen con el flag SA_ONSTACK). *tam* representa el tamaño y *addr* la dirección. Si se especifica solo el tamaño la dirección se obtendrá asignándola con *malloc()*. Si se especifica sólo la dirección y no el tamaño se asumirá un tamaño de 16K. En caso de no suministrar argumentos, o de hacerlo manera incorrecta nos indicará el tamaño de la pila alternativa y de su dirección. Nótese que en caso de especificar la dirección no hay que llamar a *malloc* y dependiendo de cuál sea esa dirección podría producir error al ejecutar el manejador.
- **signal [SEN1] [SEN2] ...**
  No da información del estado de cada señal: manejada (con la dirección del menjador, los flags y la máscara asociada al manejador), ignorada o acción por defecto, asi como de si está enmascarada o no. Si no se especifican señales nos informa de todas. Ejemplo

```
#signal INT HUP SEGV
INT  Enmacarada     manejador 0x30045a00 SA_NODEFER. mascara asoc. SIGUSR1
HUP  No enmascarada Accion por defecto
SEGV No emmascarada Ignorada
```

- **sigmask [SEN1] [SEN2] ...**
  Enmascara (mediante *sigprocmask*) las señales SEN1 SEN2 .... Si no se especifican señales nos informa de como están TODAS las señales (igual que el comando signal sin argumentos).
- **sigunmask [SEN1] [SEN2] ...**
  Desenmascara (mediante *sigprocmask*) las señales SEN1 SEN2 .... Si no se especifican señales nos informa de como están TODAS las señales (igual que el comando signal sin argumentos).
- **sigignore [SEN1] [SEN2] ...**
  Ignora las señales SEN1 SEN2 .... Si no se especifican señales nos informa de como están las señales (igual que el comando signal sin argumentos).
- **sigdefault [SEN1] [SEN2] ...**
  Pone las señales SEN1 SEN2 ... a su acción por defecto. Si no se especifican señales nos informa de como están las señales (igual que el

comando signal sin argumentos).

- **count [-r] [SEN1] [SEN2] ...**
Nos informa de los contadores de la señales SEN1 SEN2 ..., con -r resetea los contadores de dichas señales. Si no se especifican señales lo hace para todas (tanto mostrar los contadores como resetearlos en su caso)

- **handler[-v] [-r] [-p] [-t] [-f]] [-dN] [-mSEN1] [-mSEN2] ... SIG [SIG2] ....**
Instala un manejador para las señales SIG1, (SIG2 ...) (Recordar que el manejador es la función que se ejecuta CUANDO llega la señal). El manejador incrementará el contador que indica cuantas veces se ha ejecutado. Además se admitirán las siguientes opciones

-mSEN Se añade SEN al conjunto de señales enmascaradas DURANTE la ejecución del manejador (miempro sa_mask de la estructura *sigaction*)

-dN El manejador se queda en espera N segundos (con la llamada *sleep*). Dicha espera debe ser la última instrucción dentro del manejador. Pede ser distinto el tiemo de espera para manejadors instalados para distintas señales

-v El manejador debe imprimir en pantalla la señal que se ha recibido, cuantas veces se ha ejecutado dicho manejador (el valor del contador) y la dirección del parámetro que recibe. De no indicarse -v el manejador NO DEBE IMPRIMIR NADA en pantalla.

-r El manejador reenvía al proceso la señal para la cual es manejador. En caso de reenviarse la señal, debe hacerse después de imprimir en pantalla (en caso de que se imprima) y antes de quedarse en espera (en caso de que se haya especificado -sN)

-t El manejador es temporal: se instala con el flag SA_RESETHAND

-p El manejador se ejecutará en la pila alternativa: se instala con el flag SA_ONSTACK

-f El manejador se instala con el flag SA_NODEFER

- **maxhandler [*n*]**
Establece el número máximo de veces que el manejador instalado con *-r* se reenvía la señal (se utilizará para ello el contador de señales recibidas). Si no se especifica *n* se indicará a cuanto está establecido dicho valor. Un valor 0 indica que no hay límite predefinido.

- **bucle**
  Hace que el shell entre en un bucle infinito. Instala un manejador para SIGINT que permite salir del bucle pulsando control-c para seguir ejecutando el shell.
- **segmentation**
  Produce un fallo de segmentación en el shell. (No vale enviar SIGSEGV, tiene que ser un fallo de segmentación de verdad).
- **fpe**
  Produce una excepción de la unidad en punto flotante en el shell. (No vale enviar SIGFPE, tiene que ser una excepción de la unidad en punto flotante de verdad).

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

FORMA DE ENTREGA Como en prácticas anteriores

FECHA DE ENTREGA Viernes 15 Mayo 2009