SISTEMAS OPERATIVOS II

Tercer curso Ingeniería Técnica de Sistemas. Curso 2009-2010

Práctica 1: Recursos IPC en Unix: Memoria Compartida

Comenzar la codificación de un intérprete de comandos (shell) en UNIX. Nótese que los comandos aquí descritos deben interpretarse de la siguiente manera

- Los argumentos entre corchetes [] son opcionales.
- Los argumentos separados por `|` indican que debe ir uno u otro, pero no ambos simultaneamente.
- El intérprete de comandos debe aceptar y entender la sintaxis aquí propuesta, pero no tiene que forzarla. (por ejemplo, si hay varios argumentos deben aceptarse en el orden especificado, pero puede resultar mas cómodo de programar asumiendo que pueden ir en cualquier orden)

Además deben tenerse en cuenta las siguientes indicaciones

- **En ningún caso debe producir un error de ejecución (segmentation, bus error . . .)**. La práctica que produzca un error en tiempo de ejecución no será puntuada. Excepcionalmente se admitirá un error en tiempo de ejecución en algunos comandos (por ejemplo, la función recursiva que desborda la pila), en estos casos se indicará explicitamente (***)
- No debe dilapidar memoria (ejemplo: variable que se asigna cada vez que se llama a una función y no se libera). **NO SE REFIERE A DECLARAR LOS ARRAYS DE TAMAÑO PEQUEÑO**
- 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).
- Las direcciones de memoria deben mostrase en **hexadecimal**.
- La información que se nuestra en pantalla no debe incluir en ningun caso líneas en blanco (ni líneas de '*' ni de '=',. . .).
- 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.

El shell debe mantener en memoria una lista con las direcciones de memoria compartida que ha mapeado con la llamada *shmat* y las que se ha asignado

con el comando *malloc* (obviamente no se refiere a las asignadas con la funcion *malloc* para otros menesteres). Se guardará, además de la dirección, el tamaño del bloque de memoria y el instante en que se ha producido dicha asignación o encadenamiento; y en el caso de las de memoria compartida se almacenará tambien la clave.

Comandos a implementar en esta práctica

exit Termina la ejecución del intérprete de comandos.

autores Indica los nombres y los logins de los autores de la práctica.

getpid Muestra el pid del proceso y de su proceso padre

chdir [dir] Cambia el directorio actual del shell a *dir*. Si no se le suministra argumento informa del directorio actual.

malloc [-f] [tam] asigna en el shell la cantidad de memoria que se le especifica (utilizando la llamada *malloc*), y nos informa de la dirección de la memoria asignada, metiendola junto con el tamaño y el instante en la lista de direcciones asignadas. Si se especifica *-f* hace *free* de una de las zonas de tamaño *tam* asignadas **con el comando malloc** y la dirección de memoria correspondiente será eliminada de la lista. Si no se especifica tamaño los dará una lista de las direcciones de memoria asignadas **con el comando malloc**

read file dir Copia el fichero *file* en la dirección de memoria *dir* (podría producir error en caso de que *dir* no fuese adecuada)(***)

write [-f] file dir cont Copia desde la dirección de memoria *dir cont* bytes en el fichero *file*. -f especifica que se sobreesciba el fichero (si existe) (podría producir error en caso de que *dir* no fuese adecuada)(***)

display dir [cont] Muestra los contenidos de *cont bytes* a partir de la posición de memoria *dir*. Si no se especifica cont imprime 25 bytes. Para cada byte imprime, en distintas líneas el caracter asociado (en caso de no ser imprimible imprime un espacio en blanco) y su valor en hexadecimal. Imprime 25 bytes por línea. (podría producir error en caso de que *dir* no fuese adecuada)(***). Ejemplo

```
->malloc 100000
asignado 100000 en 0x8053088
->read shell.c 0x8053088
Leidos 57115 bytes
    57115 rwxr-xr-x Thu Nov 20 13:32:22 2008 shell.c
->display 0x8053088 300
  #  i  n  c  l  u  d  e     <  u  n  i  s  t  d  .  h  >     #  i  n  c  l
 23 69 6E 63 6C 75 64 65 20 3C 75 6E 69 73 74 64 2E 68 3E 0A 23 69 6E 63 6C
  u  d  e     <  s  t  d  i  o  .  h  >     #  i  n  c  l  u  d  e     <  s
```

```
75 64 65 20 3C 73 74 64 69 6F 2E 68 3E 0A 23 69 6E 63 6C 75 64 65 20 3C 73
 t  r  i  n  g  .  h  >     #  i  n  c  l  u  d  e     <  s  t  d  l  i  b
74 72 69 6E 67 2E 68 3E 0A 23 69 6E 63 6C 75 64 65 20 3C 73 74 64 6C 69 62
 .  h  >     #  i  n  c  l  u  d  e     <  s  y  s  /  t  y  p  e  s  .  h
2E 68 3E 0A 23 69 6E 63 6C 75 64 65 20 3C 73 79 73 2F 74 79 70 65 73 2E 68
 >     #  i  n  c  l  u  d  e     <  s  y  s  /  s  t  a  t  .  h  >     #
3E 0A 23 69 6E 63 6C 75 64 65 20 3C 73 79 73 2F 73 74 61 74 2E 68 3E 0A 23
```

detach [dir] Desmapea la zona de memoria compartida mapeada (con la llamada *shmdt*) en la dirección dir del espacio de direcciones del proceso y la elimina de la lista de direcciones de memoria compartida.

El shell es capaz de crear sistemas de ficheros en zonas de memoria (tanto compartida como privativa del proceso). Nótese que es un sistema con asignación contigua cuya estructura se detalla al final. Los siguientes comandos pueden acceder a dichos sistemas de ficheros

createfs cl tam Crea un sistema de ficheros virtual en una zona de memoria compartida de clave *cl* y tamaño tam. Si dicha zona de memoria ya existe producirá un error, si dicha zona de memoria no existe, creará la zona de memoria compartida, la mapeará (añadiéndola a la lista) e inicializará las estructuras del sistema de ficheros en el presentes.

makefs id_mem Crea un sistema de ficheros virtual en una zona de memoria ya existente identificada por id_mem. Esta operacion supondrá el *"formateo"* del sistema de ficheros. *id_mem* puede ser una clave, una dirección de memoria compartida o una dirección de memoria asignada con el comando malloc.

copy id_fich1 id_fich2 Copia el fichero *id_fich1* en *id_fich2*

move id_fich1 id_fich2 Mueve el fichero *id_fich1* en *id_fich2*. Es igual que copy salvo que elimina el original

delete id_fich Elimina el fichero *id_fich*

filestat id_fich Muestra información del fichero *id_fich*

listfs [-d dir |id_mem] Muestra información de los ficheros contenidos en la zona de memoria especificada por *id_mem* o en el directorio de disco*dir*. Si no se suministra *id_mem* o *dir* nos mostrará todas las zonas de memoria compartida mapeadas por el proceso

deletemen cl Elimina (mediante *shmctl(IPC_RMID...)* La zona de memoria especificada por *cl*. No necesita mapearla y NO DEBE DESMAPEARLA en caso de que esté mapeada (es más, ni siquiera necesita saber si está o no mapeada)

NOTAS SOBRE EL SISTEMA DE FICHEROS EN MEMORIA

El sistema de ficheros ha de usar asignación contigua y está definido por las estructuras que se muestran a continuación (deben usarse en el código) El miembro *desplazamiento* de los datos de un fichero se mide respecto al comienzo de la zona de datos del sistema de ficheros (es 0 para el fichero cuyos datos están al principio del area de datos). El miembro *tamano* del sistema de ficheros es el tamanño TOTAL (el de la zona de memoria compartida donde está dicho sistema) e incluye las estructuras de dicho sistema de ficheros. El miembro *elibre* es el espacio disponible para datos de los ficheros, es decir *tamano* menos lo que ocupen las estructuras del sistema de ficheros y ls datos de los ficheros en él contenidos en un instante dado. *nfich* es el número de ficheros en el sistema de ficheros: 0 representa que está vacio. Los miembros *reservado* son de uso libre, pero el resto de los items deben tener información coherente de manera que la información que se almacene en los miembros em reservados no sea imprescidible para el manejo del sistema de ficheros.

```
#define SFM_MARCA 0x2812aaff /*marca del sistema de ficheros*/
#define SFM_MAXFICH  128     /*numero maximo de ficheros*/
#define SFM_MAXNOMBRE 256    /*tamano maximo del nombre en este sistea ficheros*/
#define SFM_RESERVADOS 8

struct SB { /*info del sistema de ficheros*/
  unsigned marca; /*la marca, debe se SFM_MARCA*/
  unsigned tamano; /*tamano total sistema de ficheros*/
  unsigned elibre; /*espacio libre en sistema de ficheros*/
  unsigned nfich;  /*numero de ficheros en el sistema de ficheros*/
  unsigned reservado[SFM_RESERVADOS]; /*por si se quieren usar mas cosas*/
  };

struct NODOFICH{  /*informacion de un fichero */
  char nombre[SFM_MAXNOMBRE]; /*el nombre*/
  mode_t modo;                /*los permisos*/
  uid_t uid;                  /*propietario original*/
  off_t tamano;               /*tamano en bytes*/
  time_t fecha;               /*fecha en que se ha copiado al SF*/
  off_t desplazamiento;       /*desplazamiento a sus datos desde el principio*/
  unsigned enuso;             /*0 esta estructura NO contiene info de un fichero*/
  unsigned reservado [SFM_RESERVADOS];
  };

struct SF {    /*el sistema de ficheros*/
    struct SB sb;
    struct NODOFICH n[SFM_MAXFICH]; /*los nodos de los ficheros*/
    unsigned reservado[SFM_RESERVADOS];
```

```
char datos[1];        /*el area de datos */
};
```

NOTAS SOBRE LOS IDENTIFICADORES DE MEMORIA *id_mem* Y LOS DE FICHEROS *id_fich*

id_mem puede representar una dirección de memoria o una clave. Como los comandos que acceden al sistema de ficheros necesitan una dirección de memoria se procede de la siguiente manera: se supone que es una dirección y se comprueba si está en la lista de direcciones asignadas con *malloc* o *shmat*; si está, es una dirección válida y se usa; si no está, se supone que es una clave y se vuelve a obtener un identificador con *shmget* y una dirección con *shmat* y se añade a la lista (evidentemente esto puede suponer tener mapeada la misma zona de memoria compartida en varios sitios). **SI SE SUMINISTRA UNA CLAVE SE VUELVE A MAPEAR: NO DEBE BUSCARSE EN LA LISTA POR LA CLAVE**. Tambien debe tenerse en cuenta que un puntero a caracter no se almacena en la direccion de memoria representada por la cadena por el apuntada. (`char *p="0x004b0000"` no quiere decir que la cadena apuntada por p esté en la dirección de memoria 0x004b0000). Un posible código para obtener una dirección a partir de un identificador pordría ser:

```
void * ObtenerMemoria (char * idmem)
{
  void * p;
  key_t cl;

  p=(void *) strtoull(idmem,NULL,16);

  if (EstaEnListaMallocs (p) || EstaEnListaShareds(p))
   return p;
  cl=(key_t) strtoul (idmem,NULL,10);
  return (ObtenerMemoriaShmget(cl));
}
```

id_fich La identificación de un fichero se supone que es de la forma `id_mem:nombre_fichero`, donde *id_mem* es un identificador de memoria tal como acaba de describirse y *nombre_fichero* es el nombre del fichero en el sistema de ficheros representado por *id_mem*. Si no se especifica *id_mem* se entiende que es un fichero de disco.
Ejemplos 10:p1.c; 0xd7ac0000:p2.c, p4.c ...

Información detallada de las llamadas al sistema y las funciones de la librería debe obternerse con man (shmget, shmat, shmdt,

shmctl, read, write, stat, malloc, free, . . .)

FORMA DE ENTREGA Va a ser utilizado el servicio de recogida de prácticas suministrado por el Centro de Cálculo de esta Facultad y parte del proceso de corrección de las prácticas va a ser automtíco (compilación, listado de practicas entregadas etc) por lo cual deben entregarse **exactamente** como se indica a continuación:

- Se colocará el código fuente de la práctica en el directorio asignado para ello antes de la fecha tope de entrega de la práctica.
- Se entregará UN SOLO fichero fuente por práctica, de nombre pN.c (N el número de práctica). Por ejemplo, para esta práctica será p1.c (en minúsculas).
- Los grupos de prácticas son de **2 (DOS)** alumnos. La práctica SOLO DEBE SER ENTREGADA POR UNO DE LOS MIEMBROS DEL GRUPO
- en el código fuente de la práctica debe figurar como comentario el nombre de los autores **exactamente** en el siguiente formato

```
/*
AUTOR:apellido11 apellido12, nombre1:login_en_el_que_se_entrega
AUTOR:apellido21 apellido22, nombre2:login_en_el_que_se_entrega
*/
```

  donde:

  1. La palabra autor aparece en mayúsculas.
  2. Los apellidos y el nombre de los autores están totalmente en minúsculas.
  3. apellidoij representa el apellidoj del componente i del grupo de prácticas.
  4. No hay espacios antes y despues de los dos puntos.
  5. El login que aparece es el del que entrega la práctica (aparece el mismo login en las dos líneas).
  6. Los símbolos de comentarios están en líneas distintas.
  7. No debe incluirse la letra ñ ni vocales acentuadas en los nombres

FECHA DE ENTREGA Viernes 20 Noviembre de 2009