AMPLIACIÓN DE SISTEMAS OPERATIVOS (71023016) Septiembre 2022

Material permitido: **Ninguno** Tiempo: **120 minutos** **N**	**Aviso 1**: Todas las respuestas deben estar debidamente razonadas. **Aviso 2**: Escriba con buena letra y evite los tachones. **Aviso 3**: Solución del examen y fecha de revisión en http://www.uned.es/71023016/

1. Conteste **razonadamente** a los siguientes apartados:

a) *(1 p)* En los SOBUNIX, ¿Cuáles son los identificadores efectivos de un usuario?¿Cuándo son consultados por el núcleo?

b) *(1 p)* En los SOBUNIX, ¿Qué es una página anónima?¿Cuándo se crea?¿Qué se entiende por memoria anónima de un proceso?

c) *(1 p)* ¿Cuál es la utilidad de la capa nodo-v/SAV en los SOBUNIX?

d) *(1 p)* Describir la implementación del subsistema de E/S en Windows.

2. *(2 p)* Explicar **razonadamente** cómo se crean y las principales características de las tuberías sin nombre y de los archivos FIFOs en los SOBUNIX.

3. *(2 p)* Explique **razonadamente** cuál sería el resultado de la ejecución de las siguientes órdenes:

a) `PATH=$PATH:.`

b) `ps -Al`

c) `ls *[ot]`

d) `echo'DIR=$(pwd)'`

4. *(2 p)* Explique **razonadamente** el significado de las siguientes llamadas al sistema de los SOBUNIX:

a) `execv("data",0);`

b) `kill(513,SIGUSR1);`

c) `signal(SIGUSR2,SIG_IGN);`

d) `d=semget(a,b,c);`

AMPLIACIÓN DE SISTEMAS OPERATIVOS (Cód. 71023016)
Solución Examen Septiembre 2022

Solución Ejercicio 1

a) En los SOBUNIX cada usuario, aparte de los identificadores reales, también tiene asignados un *identificador de usuario efectivo* (Effective User IDentifier, EUID) y un *identificador de grupo efectivo* (Effective Group IDentifier, EGID). Estos *identificadores efectivos* son consultados por el núcleo en diferentes circunstancias, como por ejemplo cuando un usuario intenta crear o abrir un archivo, o si un proceso desea enviar una señal a otro proceso.

Los identificadores efectivos se inicializan con el valor de sus identificadores reales cuando un usuario inicia una sesión de trabajo en el sistema. Por otra parte, si un usuario intenta ejecutar un archivo que tiene su bit `S_ISUID` activado entonces el EUID se configura con el valor del UID del propietario del archivo. Lo mismo ocurre a nivel de grupo si se activa el bit `S_ISGID`. Esta propiedad resulta muy útil para permitir a un usuario ejecutar programas que son propiedad de usuarios más privilegiados, como el superusuario.

b) Una *página anónima* es una página que antes de su creación no dispone de un almacenamiento persistente de respaldo en memoria secundaria. Las páginas anónimas son utilizadas por ejemplo, en la región de datos no inicializados, el montículo y la región de pila. También son utilizadas en las regiones asociadas a archivos mapeados en memoria con el indicador `MAP_PRIVATE`, este el caso por ejemplo de la región de datos inicializados.

Una página anónima solo se crea cuando se accede por primera vez a ella durante una operación de escritura. Es durante el tratamiento del fallo de página correspondiente cuando se le asigna un marco de memoria principal, se inicializa el marco con ceros o con la copia del contenido de otra página, y se le asigna espacio en el área de intercambio.

La *memoria anónima de un proceso* es el conjunto de páginas anónimas utilizadas por dicho proceso.

c) La capa nodo-v/SAV permite aislar los detalles de la implementación de cada tipo de sistema de archivos del resto del núcleo, lo que simplifica su diseño. Cualquier operación sobre un archivo o sobre un sistema de archivos completo invocada por el usuario a través de una llamada al sistema o internamente por algún subsistema del núcleo es redireccionada por la capa nodo-v/SAV hacía la rutina dependiente del sistema de archivos al que pertenezca el archivo que da soporte a dicha operación. Obviamente, el núcleo de cada SOBUNIX debe disponer, por cada tipo de sistema de archivos que soporte, de las rutinas que implementan las operaciones sobre dicho sistema de archivos.

En definitiva la capa nodo-v/SAV permite descomponer las operaciones sobre un archivo en dos partes: una independiente del tipo de sistema de archivos y otra dependiente del tipo de sistema de archivos. La capa nodo-v/SAV es la interfaz que permite el paso entre ambas partes (ver Figura 5.5).

d) El *subsistema de E/S* de Windows se implementa mediante los siguientes componentes del ejecutivo:

   – *Administrador de* E/S. Es el componente central del subsistema de E/S. Se encarga de atender las peticiones de E/S realizadas por las aplicaciones de los usuarios y el resto de componentes del ejecutivo. Además define y soporta la interfaz de trabajo con los drivers.
   – *Administrador de* PnP. Se encarga de detectar la conexión o desconexión de dispositivos de E/S en el computador. Cuando el administrador de PnP detecta que se ha conectado un dispositivo entonces se encarga de cargar el driver apropiado.
   – *Administrador de potencia*. Se encarga de gestionar el consumo de energía de los componentes hardware del computador.

   Estos componentes del ejecutivo de Windows para realizar sus tareas trabajan estrechamente con los drivers de los dispositivos. En el registro de Windows se almacena una descripción de los dispositivos de E/S existentes en el computador así como la información necesaria para la inicialización y configuración de los drivers de dichos dispositivos.

Solución Ejercicio 2

Existen dos posibles tipos de tuberías:

- *Tuberías sin nombre*. Se crean haciendo uso de la llamada al sistema `r=pipe(fildes)`. Esta llamada necesita como argumento de entrada la dirección de un array entero `fildes` de dos elementos para almacenar los descriptores de archivos D_0 y D_1. Si la llamada al sistema se ejecuta con éxito devuelve 0 en caso contrario devuelve -1.

  Los intérpretes de comandos utilizan estas tuberías para hacer que la salida de un programa sea la entrada de otro programa. Es el propio intérprete el que se encarga de manipular los descriptores de archivos de la tubería devueltos por la llamada al sistema `pipe` para hacer que un programa sea el emisor y otro el receptor.

  Las tuberías sin nombre se caracterizan porque solo pueden ser utilizadas por procesos relacionados genealógicamente. Además no son persistentes, es decir, son eliminadas al finalizar los procesos que las utilizan.

- *Tuberías con nombre*, también denominadas archivos FIFOs o simplemente FIFOs. Se crean mediante la llamadas al sistema `mknod` o `mkfifo`, o desde un intérprete de comandos usando los comandos homónimos.

  Las tuberías con nombre se caracterizan porque se les puede asignar una ruta dentro del sistema de archivos. Cualquier proceso aunque no sea hijo del proceso creador de la tubería puede acceder a la tubería si conoce la ruta de la tubería y dispone de los permisos adecuados. Asimismo, los tuberías con nombre son persistentes, es decir, continúan existiendo aunque todos los procesos que lo estaban utilizando ya hayan terminado. Para eliminarlas se debe usar la llamada al sistema `unlink` o el comando homónimo.

Solución Ejercicio 3

a) Esta orden añade el directorio de trabajo actual a la variable `PATH`, la cual contiene los nombres de ruta de los directorios donde el intérprete de comandos busca los comandos externos.

b) Esta orden muestra un listado con información sobre los procesos existentes en el sistema, la opción `-A` hace que se muestren todos los procesos existentes en el sistema. Mientras que la opción `-l` hace que se muestre un listado con más información, es decir, un mayor número de columnas.

c) Esta orden muestra en la salida estándar un listado de todos los archivos o subdirectorios del directorio de trabajo actual que terminen en `o` o en `t`.

d) Esta orden muestra en la salida estándar el mensaje

```
DIR=$(pwd)
```

es decir, el argumento del comando `echo`. Esto es así porque dicho argumento está entrecomillado con comillas simples lo que indica al intérprete que no debe analizarlo.

Solución Ejercicio 4

a) Esta llamada al sistema reemplaza las regiones del espacio de direcciones del proceso que invoca la llamada por las regiones del programa `data` pasado como argumento. Cuando finaliza la llamada al sistema `exec` el proceso invocador pasa a ejecutar el código del programa invocado.

b) Esta llamada al sistema envía la señal `SIGUSR1` al proceso con identificador PID igual 513.

c) Esta llamada al sistema especifica al núcleo la acción que debe realizar cuando el proceso que invoca la llamada reciba una señal tipo `SIGUSR2`. En este caso con `SIG_IGN` se indica que la acción que debe realizarse es ignorar la recepción de este tipo de señales.

d) Esta llamada al sistema permite crear un conjunto de semáforos u obtener el uso de un conjunto ya creado. Requiere como argumentos de entrada la llave numérica `a` que se quiere asociar al conjunto de semáforos o que ya tiene asociada (si ya está creado), el número `b` de semáforos que tendrá el conjunto y una serie de indicadores `c` que permiten especificar, entre otras cosas, los permisos de acceso al conjunto de semáforos. Un indicador utilizado frecuentemente es `IPC_CREAT` que fuerza a crear un conjunto nuevo si no existe uno ya creado. Si la llamada al sistema se ejecuta con éxito entonces guarda en `d` el identificador del conjunto de semáforos. En caso contrario almacena en `d` el valor -1.