DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico

10/12/2009 Sistemas Operativos

Nro. 10

Integrante	LU	Correo electrónico
Dinota, Matías	076/07	matiasgd@gmail.com
Leveroni, Luciano	360/07	lucianolev@gmail.com
Mosteiro, Agustín	125/07	agustinmosteiro@gmail.com

Facultad de Ciencias Exactas y Naturales

Universidad de Buenos Aires

Ciudad Universitaria - (Pabellón I/Planta Baja)

Intendente Güiraldes 2160 - C1428EGA

Ciudad Autónoma de Buenos Aires - Rep. Argentina

Tel/Fax: (54 11) 4576-3359

http://www.fcen.uba.ar

Aclaraciones generales

Antes de comenzar el análisis de cada ejercicio, cabe mencionar lo siguiente:

1. Comandos básicos de Unix

1. *a*) El directorio actual pasa a ser **/usr/bin**.
   b) El directorio actual pasa a ser **/home/tpsisop**.
   c) Al no ingresar ningún parámetro, el comando *cd* cambia el directorio actual al directorio personal del usuario actual.

2. El contenido del archivo es:

```
# ~/.profile: executed by the command interpreter for login shells.
# This file is not read by bash(1), if ~/.bash_profile or ~/.bash_login
# exists.
# see /usr/share/doc/bash/examples/startup-files for examples.
# the files are located in the bash-doc package.

# the default umask is set in /etc/profile
#umask 022

# if running bash
if [ -n "$BASH_VERSION" ]; then
    # include .bashrc if it exists
    if [ -f "$HOME/.bashrc" ]; then
. "$HOME/.bashrc"
    fi
fi

# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/bin" ] ; then
    PATH="$HOME/bin:$PATH"
fi
```

3. El único archivo encontrado es **/vmlinuz**.

4. Se generó el directorio utilizando el comando **mkdir /home/tpsisop/tp**.

5. Se copió el archivo utilizando el comando **cp /etc/passwd /home/tpsisop/tp/**.

6. Se cambió el grupo utilizando el comando **chgrp tpsisop /home/tpsisop/tp/passwd**.

7. Se cambió el usuario utilizando el comando **chown tpsisop /home/tpsisop/tp/passwd**.

8. Se cambiaron los permiso del archivo **/home/tpsisop/tp/passwd**.

   - Para que el propietario tenga permisos de lectura, escritura y ejecución: **chmod u+rwx /home/tpsisop/tp/passwd**.

- Para que el grupo tenga sólo permisos de lectura y ejecución: **chmod g−w+rx /home/tpsisop/tp/passwd**.
- Para que el resto tenga sólo permisos de ejecución: **chmod o−rw+x /home/tpsisop/tp/passwd**

9.
   -
     ```
     127.0.0.1 localhost
     ::1     ip6-localhost ip6-loopback
     ```
   -

10. La nueva password es *guiguigui* con el comando **passwd**.

11. Se borró el archivo con el comando **rm /home/tpsisop/tp/passwd**.

12. Se enlazaron los archivos con los comandos **ln /etc/passwd /tmp/contra1**, **ln /etc/passwd /tmp/contra2** y **ln -s /etc/passwd /tmp/contra3** respectivamente.

13. Se montó el CD-ROM de instalacion de Ubuntu JeOS con el comando **mount /dev/sr0/ /home/tpsisop/tp/**. El contenido del directorio es:

    ```
    -r-xr-xr-x 1 root root   942 2008-04-22 03:07 cdromupgrade
    dr-xr-xr-x 3 root root  2048 2009-07-14 13:23 dists
    dr-xr-xr-x 3 root root  2048 2009-07-14 13:23 doc
    dr-xr-xr-x 3 root root  2048 2009-07-14 13:24 install
    dr-xr-xr-x 2 root root 12288 2009-07-14 13:24 isolinux
    -r--r--r-- 1 root root 47110 2009-07-14 13:24 md5sum.txt
    dr-xr-xr-x 2 root root  2048 2009-07-14 13:23 pics
    dr-xr-xr-x 4 root root  2048 2009-07-14 13:23 pool
    dr-xr-xr-x 2 root root  2048 2009-07-14 13:23 preseed
    -r--r--r-- 1 root root   228 2009-07-14 13:23 README.diskdefines
    lr-xr-xr-x 1 root root     1 2009-07-14 13:23 ubuntu -> .
    ```

    Se utilizó el comando **mount** para mostrar los siguientes *filesystems* montados:

    ```
    /dev/sda1 on / type ext3 (rw,relatime,errors=remount-ro)
    proc on /proc type proc (rw,noexec,nosuid,nodev)
    /sys on /sys type sysfs (rw,noexec,nosuid,nodev)
    varrun on /var/run type tmpfs (rw,noexec,nosuid,nodev,mode=0755)
    varlock on /var/lock type tmpfs (rw,noexec,nosuid,nodev,mode=1777)
    udev on /dev type tmpfs (rw,mode=0755)
    devshm on /dev/shm type tmpfs (rw)
    devpts on /dev/pts type devpts (rw,gid=5,mode=620)
    /dev/scd0 on /home/tpsisop/tp type iso9660 (ro)
    ```

14. Se utilizó el comando **df** para mostrar el espacio libre de los *filesystems* montados:

    ```
    Filesystem            Size  Used Avail Use% Mounted on
    /dev/sda1             494M  423M   46M  91% /
    varrun                252M   32K  252M   1% /var/run
    varlock               252M     0  252M   0% /var/lock
    ```

```
udev                  252M   40K  252M   1% /dev
devshm                252M     0  252M   0% /dev/shm
/dev/scd0             101M  101M     0 100% /home/tpsisop/tp
```

15. TODO

16. Se desmontó el CD-ROM de instalación de Ubuntu JeOS con el comando **unmount /dev/scd0**.

17. La maquina virtual lleva 58 minutos de ejecución. Esto se corroboró con el comando **uptime**.

18. La versión de kernel utilizada es la 2.6.24-24-virtual. Esta información se obtuvo utilizando el comando **uname -a**.

2. Comandos Extendidos de Unix

1. Para escribir HOLA en la pantalla cada vez que se loguee un usuario se agregó la siguiente línea al archivo /etc/bash.bashrc: **echo "HOLA"**

   Para escribir BUENOS DIAS en la pantalla cada vez que se encienda la máquina se agregó la siguiente línea al archivo /etc/rc.local: **echo "BUENOS DIAS"**

   Para escribir ADIOS en la pantalla cada vez que se desloguee un usuario se creó un archivo llamado logout en el directorio /etc. El contenido del archivo es el siguiente: **echo .ªDIOS"** Luego, se agregó en el archivo /etc/rc.local la siguiente línea: **trap '/etc/logout;exit' 0**

   Para escribir HASTA LA VISTA BABY en la pantalla cada vez que se apague la máquina se agregó la siguiente línea al archivo /etc/rc0.d: **echo "HASTA LA VISTA BABY"**

2. Para montar una imagen de floppy se utilizó el siguiente comando: **mount -o loop imagen.img /media/floppy1/**

   Para montar una imagen iso se utilizó el siguiente comando: **mount -o loop imagen.iso /media/iso/**

3. Se agregó un alias en el archivo .bashrc sin modificar su fecha (timestamp). Para esto se utilizó el comando **touch -t timestamp**. El timestamp del archivo previo a la modificación se obtuvo con el comando **ls -l**. El día y la hora del sistema se modificó con el comando **date timestamp**.

4. TODO

5. *a*) Se guardó la información en el archivo *config* utilizando el comando **ls -Rl /etc ¿/home/tpsisop/tp/config**.

   b) El archivo *config* posee 789 líneas, 5061 palabras y 39070 caracteres. Se obtuvo la información mediante el comando **wc config**.

   c) Se agrego el contenido ordenado del archivo */etc/passwd* al final del archivo *config* con el comando **sort /etc/passwd >> /home/tpsisop/tp/config**.

d) El archivo *config* posee 813 líneas, 5090 palabras y 40051 caracteres. Se obtuvo la información mediante el comando **wc config**.

e) Se realizó lo pedido ejecutando el siguiente comando: **ls -l /usr/bin/a* — grep apt — wc**.

El kernel de Linux

Funcionamiento del kernel linux

1. Ej 1

2. **Administración de la memoria**

   Para administrar la memoria linux utiliza un esquema de paginación por demanda de tres niveles con paginas fijas de 4KB. Dentro de este esquema cada proceso de usuario tiene su propio espacio de direcciones virtual. Mediante el uso de memoria swap se tiene un espacio virtual de 4GB donde 3 son para procesos de usuario y el restante para uso del kernel.

   Para administrar los tres niveles de paginación se utilizan las siguientes tablas:

   a) Directorio global: cada proceso tiene solo una que ha de estar en memoria y su tamaño es de una página.

   b) Directorio intermedio de páginas: puede ocupar varias páginas. Cada entrada señala a una página de la tabla de páginas final

   c) Tabla de páginas: cada una de sus entradas hace referencia a la página virtual requerida.

   En cuanto a la administración de la memoria física, una parte es utilizada por el kernel. Esta partición es permanente, es decir, ninguna de sus partes se pagina a disco. El resto de la memoria esta disponible para:

   a) Páginas de usuario

   b) El cache de buffer: Contiene bloques de disco que se han leido recientemente, su tamaño es dinámico y compite por la misma reserva de páginas que las páginas de usuario.

   c) El cache de paginación: formado por un conjunto de páginas de usuario que ya no se necesitan y estan esperando que se les pagine a disco.

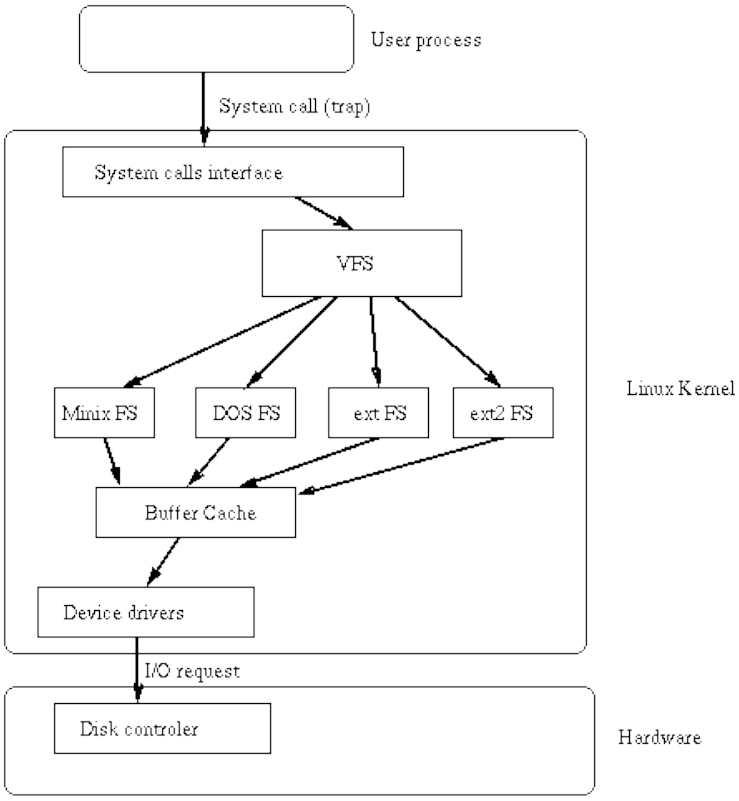
3. *a*) **Sistema de archivos**

   El sistema de archivos utilizado en UbuntuJeOS es extended 3 (ext3). Este sistema de archivos tiene un tipo de tabla de tamaño fijo donde se almacenan los i-nodos. Estos almacenan información del archivo como por ejemplo la ruta o path, tamaño, permisos, ubicación física, etc (la descripción completa de un i-nodo se presentará en posteriores secciones). En cuanto a la ubicación física, los i-nodos contienen referencias a bloques ubicados en el disco. Estos bloques son de tamaño especificable cuando se crea el sistema de archivos, desde los 512 bytes hasta los 4 KB, lo cual asegura un buen aprovechamiento del espacio libre con archivos pequeños.

El espacio en ext3 está dividido en bloques, y los bloques organizados en grupos. Esto se hace para reducir la fragmentación externa y reducir al mínimo el número de búsquedas de disco cuando se lee una gran cantidad de datos consecutivos. Cada bloque contiene un superbloque grupo, el grupo de bloques de mapa de bits, mapa de bits i-nodo, seguidos por los bloques de datos reales. El superbloque contiene información importante que es crucial para el arranque del sistema operativo, con lo que las copias se realizan en cada bloque de grupo de cada bloque en el sistema de archivos. Sin embargo, sólo la primera copia de la misma, que se encuentra en el primer bloque del sistema de archivos, se utiliza en el arranque. El grupo descriptor almacena el valor del bloque de mapa de bits, mapa de bits inodo y el comienzo de la tabla de i-nodos por cada bloque de grupo y estos, a su vez, se almacena en un grupo descriptor tabla.

A diferencia de su antecesor ext3 implementa un sistema de registro por diario (journaling) para hacer posible el uso de transacciones. Este sistema se basa en mantener un registro en el que se almacena la información necesaria para restablecer los datos afectados por la transacción en caso de que esta falle.

b) El kernel de Linux contiene una capa llamada Virtual File System (VFS) que se utiliza en las system call que actúan sobre archivos. El VFS es una capa de indirección encargada de recibir las llamadas al sistema relacionadas al manejo de archivos y llamar a las funciones necesarias en el sistema de archivos físico. Cuando un proceso hace una llamada al sistema para acceder a un archivo, el kernel llama a una función en el VFS. Esta función hace las operaciones que son independientes a la estructura del archivo y llama a la correspondiente función en el sistema de archivos físico para realizar las operaciones que dependen de la estructura. A continuación se presenta una figura que presenta el esquema mencionado.



Temas del Sistema Operativo

1. **Comunicación** Para realizar transferencia de archivos entre la máquina virtual y el sistema operativo anfitrión se optó por utilizar carpetas compartidas. Esto se logró mediante los siguientes pasos:

   - Dentro de la máquina virtual se accede a *Directorios compartidos* dentro del menú *Dispositivos*.
   - Se selecciona la carpeta que se desea compartir en la máquina anfitrion y se le asigna un nombre.
   - Se ejecuta el comando **mount -t vboxsf 'nombre-carpeta' /mnt/** para montar la carpeta compartida en la máquina virtual.

2. **File System** Los *hardlinks* apuntan a una estructura llamada i-nodo que contiene información sobre el archivo al que hace referencia el link y punteros a los bloques de memoria física en donde está alojado dicho archivo. Cada i-nodo almacena en uno de sus campos la cantidad de links que existen al archivo al que referencia. De este modo, sólo se elimina un i-nodo, y su correspondiente archivo, cuando la cantidad de links al archivo es 0, es decir, si se elimina un *hardlink*, pero siguen existiendo links al archivo, este no será eliminado. Un i-nodo contiene la siguiente información:

   - Modo (tipo de archivo y permisos)
   - Cantidad de links

- UID del owner
- GID del owner
- Tamaño del archivo (en bytes)
- Fecha en la que el archivo fue accedido por última vez
- Fecha en la que el archivo fue modificado por última vez
- Fecha en la que el i-nodo fue modificado por última vez
- 12 punteros a bloques
- 1 puntero indirecto a bloques
- 1 puntero doble indirecto a bloques
- 1 puntero triple indirecto a bloques
- Estado del i-nodo (flags)
- Cantidad de bloques que ocupa el archivo
- Campos extra o reservados

3. **Prioridades**

   Para lograr ejecutar procesos con distintas prioridades utilizamos el comando **nice**. En este caso se quieren ejecutar tres procesos (*loop1*, *loop2* y *loop3*) y darle mayor prioridad a uno de ellos. Con este objetivo se ejecutaron los procesos *loop1* y *loop2* asignandoles una muy baja prioridad (**nice -15 ./loop1** y **nice -15 ./loop2**). Luego se ejecutó el proceso *loop3* normalmente logrando asi que tenga más prioridad que los otros procesos mencionados.

   Para verificar el uso del CPU por parte de cada uno de los procesos se utilizó el comando **top**. De esta manera se obtuvieron los siguientes resultados:

   - *loop1*: 3.3 %
   - *loop2*: 3.3 %
   - *loop3*: 93.1 %