Índice

Comandos Básicos de Unix

A continuación se presentan los comandos básicos utilizados y las explicaciones correspondientes:

(Todas las imágenes a las que se hacen referencia se encuentran en el cd entregado y no en el informe por cuestiones de impresión)

- pwd: El comando pwd es una abreviatura de Print Working Directory. Como su nombre lo indica, su función es imprimir por pantalla el directorio actual.

  Si se ejecuta el comando cd /usr/bin y luego se utiliza el comando pwd, justamente se imprime por pantalla /usr/bin ya que la función que realiza el comando cd es cambiar el directorio posicionandonos sobre el especificado, en este caso /usr/bin. Luego, el comando pwd nos indica la realización de este cambio de directorio.

  Si ejecuta el comando cd sin especificar directorio al cual se desea cambiar y luego se ejecuta el comando pwd, entonces el mismo imprime por pantalla /home/grupo5. Esto sucede de esta manera, ya que si al comando cd no se le pasan parámetros, el mismo cambia el directorio al home. En nuestro caso, /home/grupo5.

- cat: El comando cat se utiliza, en este caso, para imprimir por pantalla el contenido del archivo especificado. De este manera, se puede ver el contenido del archivo .profile. Cabe destacar que el comando cat no solo se utiliza para este fin, el mismo también tiene otros usos como concatenar archivos en un archivo nuevo. Ademas, existen varios flags para modificar el comportamiento del comando, tal como ademas de imprimir el archivo, imprimir el número de linea.

- find: Este comando se utiliza para buscar archivos dentro de directorios. Para ejecutar lo pedido se utilizo la sintaxis: sudo find -name vmlinuz*. El mismo indica que se deben buscar los archivos que coincidan en nombre con vmlinuz y luego el asterisco indica que puede seguir de diferentes maneras. Cabe destacar que se necesita permisos de superusuario para poder explorar ciertas carpetas que tiene restricción.

- mkdir: El comando mkdir es una abreviatura de Make Directory, el mismo sirve justamente para crear directorios especificando el nombre del directorio a crear. Este comando se puede utilizar para crear varios directorios a la vez, y también se puede hacer uso de distintos flags por ejemplo para crear directorios de solo lectura. Al utilizar el comando

mkdir tp estando en /home/grupo5, se crea un directorio llamado tp dentro de donde se esta posicionado.

- cp: El comando cp proviene de la palabra copy, justamente la utilidad del mismo es poder copiar archivos. En el caso de ejecutar cp /etc/passwd /home/grupo5/tp lo que se esta haciendo es copiar el archivo fuente (/etc/passwd) al directorio destino, que en este caso es el directorio tp creado recientemente.

- chgrp, chown y chmod: Los comandos chgrp, chown y chmod provienen de Change Group, Change Owner y Change Mode respectivamente. Los mismos se utilizan justamente para cambiar el grupo de un archivo, el owner y los permisos de accesos del mismo.

  Para utilizar los dos primeros comandos se debe especificar el nombre del grupo u owner nuevo y el archivo que se quiere modificar. En los casos pedidos la sintaxis sería, chgrp grupo5 /home/grupo5/tp/passwd y chown grupo5 /home/grupo5/tp/passwd. Luego de ejecutar estos dos comandos, el grupo y el owner de passwd pasan a ser grupo5. Esto se puede ver utilizando el comando ls -l.

  Luego, para utilizar el comando chmod, se debe especificar cuales son los permisos a cambiar y a quien se quiere asignar o desasignar estos permisos. Para lograr lo pedido en este trabajo práctico las sentencias a utilizar serían: chmod u+rwx passwd, chmod g+rx passwd, chmod o+x passwd. El primer comando es para cambiar el user, y le asigna(+) Read, Write y eXecute. El segundo comando le asigna Read y eXecute al Grupo. Por último, el tercer comando asigna eXecute a los demás (Others). Cabe destacar que como el archivo ya traia permisos de Read para los Others, se quitaron los mismos utilizando chmod o-r passwd.

  Luego de la utilización de todos estos comandos, se miran los atributos del archivo mediante el comando ls -l, para ver como van cambiando el grupo, el owner y los permisos a medida que se van utilizando los comandos explicados.

  Cabe destacar que el comando chmod se utilizó varias veces para ir cambiando de a poco los permisos según sean para el usuario, para el grupo o para los otros. Sin embargo, esto se puede realizar utilizando una sola vez el comando chmod. La sintaxis para el mismo sería chmod 751 passwd. El numero 751 esta es una manera de codificar los permisos, el 7(111) corresponde al user y esta poniendo en 1 los 3 permisos; el 5(101) es para el grupo, dado un orden predispuesto, lo que hace esto es setear en 1 los permisos de Read y eXecute, que son el primero y el

tercero respectivamente, y setea en 0 el segundo permiso(Write). Por último, el 1(001) corresponde a setear el permiso de eXecute para los Others.

- grep: El comando grep es para buscar expresiones regulares dentro de un archivo. En el caso pedido lo que se quiere es buscar la expresión localhost dentro del archivo /etc/hosts. Para esto se utiliza el comando de la siguiente manera: grep localhosts /etc/hosts. De esta manera, se puede ver como el comando grep imprime las lineas del archivo hosts en donde aparece la expresión localhosts.

  Luego, se pide realizar la misma acción pero sobre todos los archivos del directorio /etc. Para esto se utiliza la siguiente sintaxis: grep -r POSIX — grep -v Binary. Lo que hace este comando es primero buscar recursivamente (-r) los archivos que concuerden, luego con estos archivos encontrados se hace otra busqueda con -v que indica que hay que evitar los que coincidan con Binary.

- passwd: El comando passwd se utiliza para manejar contraseñas. El mismo posee varias flags para distintas opciones. Sin embargo, si se utiliza sin ninguna opción, su función por default es cambiar el password del user. Primero pide escribir el viejo password y luego pide asginar un nuevo password y confirmarlo (grupo5).

- rm: El comando rm proviene de la palabra remove. El mismo se puede utilizar con diferentes opciones si se quieren eliminar directorios recursivamente por ejemplo. En el caso pedido, solo hay que eleminar el archivo passwd, por lo que la sintaxis es rm passwd.

- ln: El comando ln se utiliza para crear links entre archivos. Para utilizar el comando ln se debe especificar el archivo fuente y el archivo destino. Cabe destacar que el uso de este comando sin ningún flag creara un hard link. Para realizar lo pedido en el trabajo práctico la sintaxis utilizada fue la siguiente: ln /etc/passwd /tmp/contra1 y ln /etc/passwd /tmp/contra2. Al realizar esto, se puede corroborar haciendo ls -l que hay 3 links al archivo /etc/passwd.

  Para realizar un soft link o link simbolico se debe agregar la opción -s al utilizar el comando. Se puede ver como la utilización del comando ln -s /etc/passwd /tmp/contra3 no cambia la cantidad de hard links usando el comando ls -l.

- mount: Para realizar lo pedido, debemos utilizar el comando mount de la siguiente manera: mount /dev/cdrom de esta manera, estamos

montando el cdrom. Luego para poder ver los filesystems que estan montados debemos ver el archivo /etc/mtab. Se puede ver como se monta la unidad de cd y luego haciendo cat /etc/mtab se muestran los filesystems montados.

- df y ps: Los comandos df y ps sirven para mostrar el espacio libre de los filesystems y los procesos que se tienen ejecutando. El comando ps, se utiliza con la opción -A para listar todos los procesos y ahi se pueden notar cuales son del sistema y cuales son de los usuarios.

- umount: El comando umount sirve para desmontar el disposito de cdrom utilizando la sintaxis umount /dev/cdrom. Se puede ver que al realizar esto y luego al utilizar el comando df, se ve como el filesystem del cd ya no se muestra.

- uptime y uname: Estos comandos se utilizan para saber cuanto tiempo lleva el sistema corriendo y para saber que version del kernel se esta usando.

Comandos Extendidos de Unix

Ejecución automática

- Para poder imprimir un hola por pantalla cada vez que un usuario se loguea, se debe modificar el archivo pertinente. En este caso, para que el hola aparezca cuando se loguea cualquier usuario se debe modificar el archivo /etc/.profile. En dicho archivo, se debe agregar una linea que especifique la acción requerida, la sintaxis es simplemente echo Hola. Luego, cada vez que un usuario se loguee, el archivo .profile se correra y el mismo indicará que se imprima Hola por pantalla.

- Para escribir Buenos Dias al prender la máquina se hizo un script que solo tuviese un echo Buenos Dias, el mismo se linkeo con el archivo /etc/rc.local que es el que se corre al prender la máquina.

- Para poder imprimir un Adios por pantalla cada vez que un usuario se desloguea el proceso es diferente al realizado en el primer item de este punto. La manera debe ser diferente debido a que, a diferencia del fichero profile, no se encontro un fichero que se corra automaticamente al hacer logout.

  Para poder indicarle al sistema que se quiere correr un archivo al hacer logout lo que se realizó fue modificar nuevamente el fichero /etc/profile

con la siguiente linea de comando: trap '. /etc/.logout;exit' 0. Lo que hace el comando trap es darle una directiva al sistema operativo, en este caso lo que hace es darle la directiva de que cuando se reciba una señal de exit(logout) se corra el fichero .logout.

Luego, se creo un archivo .logout con los scripts que se quisiesen correr a la hora del logout, en nuestro caso el archivo consta solamente de una linea que indica la impresion de una linea de Adios. La sintaxis dentro del archivo es simplemente echo Adios.

Por último lo que se realizó fue hacer chmod 755 .logut, esto se hizo para que el archivo tuviese permisos de ejecución.

Cabe destacar que luego de realizar esto, se encontró el archivo que se corre al hacer logout y directamente se escribo echo Adios en este archivo al igual que se hizo con el Hola.

- Para escribir Hasta la vista baby cada vez que se apaga la maquina lo que se hizo en primer lugar fue crear un script que nada más tuviese un echo Hasta la vista baby. Luego, tuvimos que buscar como hacer que se ejecutase al apagarse la maquina y se encontró que al reiniciarse se corre el rc0 y cuando se apaga totalmente se corre el rc6, por lo que se procedio a hacer un link entre el script hecho y los rc0 y rc6.

Dispositivos Especiales

Para montar estos dos dispositivos se utiló la siguiente sintaxis:
sudo mount /dev/fd0 directorioElegida
sudo mount -o loop imagenIso directorioElegido

Cambio de Hora

Para esta sección lo primero que hay que hacer es crear un alias en el fichero deseado. Para eso se debe editar este archivo y poner algun alias como puede ser alias bla = ls. Luego hay que mantenerle el timestamp original, para esto es necesaria haberse guardado el timestamp antes de modificar el archivo, para conseguir el timestamp se utiliza el comando stat. Luego de modificarlo, se utiliza el comando touch para modificar el timestamp y poner el original, el mismo se usa con el flag -m que justamente indica que vamos a tocar el timestamp de modified.

Luego, para cambiar el horario del sistema linux se utiliza el comando date –set= y la fecha que se quiere poner, si el cambio es hacia atras no se nota ningún problema porque el sistema operativo reacomoda su hora con

la hora del hardware cada cierto intervalo de tiempo. Si se hace un date rapidamente despues del cambio, se vera la fecha que fue modificada, pero si se deja un lapso de tiempo, el sistema se actualiza automaticamente. Lo que se podría hacer es utilizar el comando hwclock para setear el horario del hardware.

Lapsos y Tiempos

Para conocer toda la información que se pide en esta sección hay varios comandos. En primer lugar se puede utilizar el comando uptime de la misma manera que se utilizó en la primer sección de este trabajo, para tener información de cuando se prendió la maquina y cuantos usuarios hay logueados.

Sin embargo, el comando que más información pertinente nos da, es el comando last. El mismo lista todas los últimos movimiento del sistema, mostrando quien se logueo, en que terminal y a que hora. Así como otros datos pertinentes.

Salida Estándar y Pipes

STDOUT:

En primer lugar se utilizó el siguiente comando para volcar la información pedida: ls -R >> /home/grupo5/tp/config. El mismo se utilizó estando en el directorio /etc, de esta manera el comando ls -R lo que hizo fue lista recusirvamente todos el contenido del directorio y su subdirectorios. Mientras que >> lo que hizo fue volcar esta información al archivo especificado.

Para obtener la información sobre la cantidad de lineas, palabras y caracteres se utilizó el comando wc con las opciones necesarios en cada caso (-l, -w, -m). El resultado fue: 665 lineas, 593 palabras y 7305 caracteres.

En tercer lugar, se buscar ordenar el contenido de /etc/passwd y ponerlo al final del archivo config anteriormente nombrado. Para ordenar se utilizó el comando sort, mientras que para volcar en el archivo nuevamente se utilizo >>. De esta manera la sintaxis utilizada fue: sort /etc/passwd >> /home/grupo5/tp/config

Por último se procedió a contar las lineas, palabras y caracteres del archivo de la misma forma que se habia realizado anteriormete, el resultado fue: 688 lineas, 621 palabras y 8238 caracteres.

Pipes: Para realizar lo pedido, sin utilizar archivos temporales, se ejecutó la siguiente linea de comando que hace lo que el ejercicio pide, en el orden requerido. La sintaxis es: ls -al /usr/bin/a* — grep apt — wc. La

misma es una combinación de los comandos utilizados en puntos anteriores, para lograr todos los efectos requeridos.

Scripting

En este punto se debe hacer un script que cada 5 minutos diga HOLA por pantalla. Para esto, se utilizaron dos resoluciones diferentes, una que es totalmente automática y la otra semiautomática.

Por un lado, se realizó un script de shell que, una vez ejecutado, corriese un ciclo infinito; dentro de este ciclo, el script espera 300 segundos mediante el comando sleep 300 y luego hace un echo HOLA para imprimir por pantalla la palabra requerida. Esta solución requiere que el usuario ejecute el script. Dicho script se encuentra en la carpeta /home/grupo5/tp.

Por otro lado, se encuentra otra solución totalmente automática, es decir que no requiere que el usuario ejecute el script. La misma consiste en agregar una entrada a la crontable que indique que se debe imprimir HOLA cada 5 minutos. Cabe destacar que la distribución de ubuntu utilizada para este trabajo, no cuenta con el cron por lo que fue necesario instalarlo. Una vez instalado, se agrega en la tabla la entrada correspondiente. Dicha entrada tiene la siguiente sintaxis: */5 * * * * root echo HOLA >> /dev/console. Esta sintaxis dice que, cada 5 minutos, a cualquier hora, en cualquier dia, en cualquier mes y en cualquier dia de la semana; el usuario root corra el comando echo HOLA y lo imprima en la consola.

Para hacer lo mismo pero a una hora determinada, se debe explicitar la hora requerida en la entrada de la crontable. Por ejemplo, en la imagen se encuentra la entrada 25 7 * * * root echo HOLA >> /dev/console. Esta entrada tiene la misma acción que la anterior, pero se ejecuta todos los días a las 7 y 25.

Por otro lado, se debe implementar un script para actualizar contraseñas. Este script se encuentra en la imagen del disco, en la carpeta /home/grupo5/tp/scripts. En el mismo, se puede detallar como se va realizando lo pedido por el enunciado mediante la modificación de los archivos pertinentes a partir de saber si se esta modificando un usuario o un grupo.

Ejecución de Procesos en Background

En primer lugar, se procedió a transcribir el programa loop, se guardo el mismo en el directorio /home/grupo5/tp. Una vez transcripto, se procedio a compilar el mismo utilizando el gcc.

En primer lugar el proceso se corrió en foreground. Esto quiere decir que el proceso toma el control de la terminal que se esta utilizando, es decir que se corre en un primer plano. El problema con este proceso es que el mismo contiene un ciclo infinito que escribe en la pantalla. Como el proceso esta corriendo en foreground, y el ciclo es infinito, el mismo nunca devuelve el control de la terminal y sigue ensuciando la consola sin poder realizar otra acción. Luego, se mató el proceso mediante Ctrl+C para poder interrumpirlo y así volver a tomar el control de la terminal.

En segundo lugar, el proceso se corrió en background. Esto quier decir que el proceso queda corriendo en un segundo plano, mientras que el usuario sigue teniendo control de la terminal. Para solucionar el hecho de que el proceso ensucia la terminal, el mismo se corrió mandando su salida a /dev/null. Al realizar esta acción, el proceso queda corriendo, pero se puede seguir haciendo uso de la terminal.

Como este proceso no esta corriendo en foreground, Ctrl+C para matarlo no servirá. Es por esto que se debe matar el proceso mediante el comando kill, el mismo necesita del ID del proceso que se quiere matar. Este ID se obtiene al correr el programa, ya que al correr un proceso en background, por consola se devuelve el número de proceso asignado para, por ejemplo, saber que número tiene al querer matarlo.

IPC y Sincronización

Pipes: El problema de pipes se encuentra en la imagen del disco en la carpeta /home/grupo5/tp/pipes.

Dicho ejercicio implementa exclusión mutua entre dos procesos mediante un pipe de manera análoga a lo explicado en clase.

- Se crea un proceso y se abre un pipe.
- Con fork() se crea un proceso hijo que hereda los descriptores del pipe del padre.
- Por último, se crea la exclusión mutua haciendo que el hijo solo pueda leer del pipe y el padre solo pueda escribir en el pipe.

Threads: Para este problema, el código se encuentra en la imagen en la carpeta /home/grupo5/tp/threads.

Cabe destacar que el codígo implementado se hizo utilizando las explicaciones de las clases prácticas y utilizando referencia bibliográfica. En particular se utilizó la guia de programación multithread de Sun Microsystems que se

puede obtener de http://www.cs.bu.edu/fac/richwest/cs552_fall_2009/notes/threads_sun.pdf. A partir de la página 103, se encuentra la información pertinente para este enunciado.

El Kernel Linux

Adminsitración del procesador

En cuanto a la administración del procesador de linux refiere, el sistema operativo maneja los procesos o tareas. Los procesos o tareas, son partes de un programa, o un programa entero, que cumplen alguna función específica mediante el manipulamiento de ciertas variables y funciones.

La administración del procesador justamente se basa en ver como manejar estos procesos para lograr que todos funcionen correctamente en tiempo y forma.

Los procesos en un sistema operativo de tipo linux tienen varios estados:

- Running: Es cuando el proceso se encuentra ejecutando, tiene todos los recursos que necesita para poder funcionar correctamente.
- Sleeping: Estos procesos se encuentran dormidos, puede ser porque son procesos interactivos que no se utilizan hace cierta cantidad de tiempo por ejemplo; o bien porque es un proceso que esta esperando algún suceso para continuar.
- Zombie: Un proceso zombie es aquel que ya no esta en funcionamiento pero que tiene una entrada en la tabla de procesos. Esto puede suceder cuando un proceso hijo termina, pero el padre todavia es capaz de ver su estado de salida por algún motivo en particular.
- Stopped: Son procesos detenidos totalmente, pero que pueden ser reiniciados para continuar con su ejecución.
- Uninterruptible sleep: Son procesos que se encuentran durmiendo pero que no pueden ser interrumpidos, generalmente tienen que ver con entrada/salida del sistema operativo mismo.
- Dead: proceso terminado que puede seguir apareciendo en el listado de procesos.

Si se esta trabajando en un sistema operativo con interfaz gráfica como ubuntu, se puede utilizar alguna herramiento como el gnome-system-monitor

para ver todos estos procesos. A su vez, a un costado del mismo, aparecerá el estado del proceso, que será alguno de los anteriormente explicados.

Por otro lado, en un sistema sin interfaz gráfica como el utilizado para este trabajo práctico se puede utilizar el comando ps con algún flag que nos de toda la información posible. Usualmente en estos casos aparece una letra que indica el estado del proceso: R de running, S de sleeping, Z de zombie, T de sTopped, U de Uninterruptible, X de dead.

Una vez explicados como son los procesos en linux, es importante ver como se organizan estos para su ejecución. Linux se basa en lista de prioridades para poder manejar sus procesos. Para esto, linux divide sus tareas en dos grupos:

- Procesos del sistema
- Procesos de usuarios

Procesos del sistema: Para los procesos del sistema linux posee 100 listas de prioridad, donde a menor número, mayor prioridad tiene el proceso. Para este tipo de procesos linux posee dos pólticias diferentes para manejarlos:

- FIFO: First in, First out. Es simplemente una cola, donde el primer proceso que entra, es el primer proceso que sale de la cola de prioridad.
- Round Robin: Es un sistema que se basa en darle un tiempo de ejecución a cada uno de los procesos. Es decir, se entrega un quantum a cada proceso, si el proceso finaliza o el quantum se agota, el proceso es desalojado.

Procesos de usuarios: Para estos procesos linux posee 40 listas de prioridad, numeradas desde -20 hasta 19. Para estos procesos el sistema operativo se encarga de alternarlos de manera que no haya ninguno que se encuentre mucho tiempo inactivo. Para esto el sistema se encarga de penalizar los procesos que tienen mucho tiempo de cpu, y trata de darle más prioridad a los procesos que tienen más entrada/salida, que se supone que estuvieron bloqueados una mayor cantidad de tiempo.

Por defecto, el proceso cuando se ejecuta tiene nivel de prioridad cero. Si se quiere que un proceso tenga una prioridad diferente, se puede utilizar el comando nice para realizar un cambio de prioridad. Si la prioridad a asignar es positiva, es decir una prioridad menor, se puede hacer directamente. Si, por el contrario, se quiere asignar una prioridad mayor, es decir negativa, se necesitan permisos de superusuario.

En el caso que un proceso ya se encuentre en ejecución, se puede cambiar la prioridad del mismo sin necesidad de matarlo. Esto se puede hacer con el comando renice, que reasigna prioridad en tiempo de ejecución.

Administración de la memoria

La administración de la memoria en linux, al igual que en la mayoría de los sistemas operativos modernos, es página por demanda debido a que resulta efectivo poder simular mayor cantidad de memoria ram utilizando otro dispositivo como el disco duro. Linux posee páginas de 4kb o 4mb dependiendo la arquitectura del procesador y contiene un espacio especial de memoria en el disco denominado memoria swap donde justamente hace el intercambio entre las paginas que se encuentran en memoria principal y las que se encuentran en memoria secundaria según las necesidades requeridas. Esta memoria se puede asignar manualmente al instalar el sistema operativo y tiene como consecuencia una mejor o peor eficiencia del manejo de la memoria virtual.

Cabe destacar, que al momento de realizar los intercambios, el sistema operativo linux toma una política LRU, last recently used, es decir que la página elegida para ser removida de memoria principal es la que más tiempo lleva sin ser referenciada.

Sistema de archivos

El sistema de archivos de linux es, basicamente, el heredado del sistema UNIX. El mismo es un sistema de archivo jerárquico, donde todo se toma o como un directorio o como un archivo. Este sistema de archivo es jerárquicos, ya que los directorios con sus subdirectorios correspondientes tiene una topología de árbol donde se encuentra el directorio root y luego todos los demás, colgando de este primero.

Es importante destacar que para el filesystem de linux, todo es un archivo o un directorio, es por esto que se puede acceder por ejemplo a los dispositivos de la misma manera que a cualquier otro directorio. Por ejemplo, si se escribe información en el directorio /dev/dsp, se podra escuchar esta información por los parlantes dado que este directorio corresponde al dispositivo de sonido.

Al heredar este filesystem de UNIX, el mismo posee las mismas propiedas a bajo nivel, es decir que los medios de representación son los i-nodos. Además, el sistema posee un Virtual file system que se encarga de ver los filesystem que se pueden ir montando sobre el original.

Módulos del sistema operativo

En primer lugar se encuentra el módulo de hello.c otorgado por la cátera, este módulo se encuentra en la imagen del sistema operativo y fue probado con exito sin ninguna otra observación particular.

Por otro lado, se encuentra el módulo para manejar las luces del teclado:

- En primer lugar se compilo el programa entregado por la cátedra para poder ver el estado de las luces.
- Se procedio a realizar el módulo correspondiente, el mismo se encarga de crear un archivo en el directorio /proc que sirve para manejar los leds. Cuando se escribe en el mismo, cambia el estado de los leds, según lo que se haya escrito en el archivo creado. Una vez instalado el modulo, se puede ver como haciendo echo Q >> /proc/archivo, se prende la luz correspondiente, y dicha acción se puede monitorear con el programa anteriormente mencionado.

  Cabe destacar que dicho módulo se inserto con éxito en el sistema operativo de la maquina host, donde si se poseen leds; y se pudo ver como escribiendo en el archivo correspondiente se pueden controlar los leds sin necesidad de presionar el botón correspondiente.

Temas del Sistema Operativo

Comunicación

Para poder compartir una carpeta entre el host y el guest se deben seguir los siguientes pasos:

- En primer lugar hay que dirigirse dentro de la maquina virtual a Devices y luego hacer click en la opción Guest Additions. Montando en el cd esta opción.
- Luego se ejecuta el archivo de instalación dependiendo de la versión utilizada. En nuestro caso corresponde a linux86
- Luego tenemos que agregar la carpeta del host que queremos compartir. Esta carpeta fue previamente creada en el host y luego se debe agregar la carpeta yendo a devices y ahi dentro a la opción shared folders.
- Luego debemos crear la carpeta dentro del sistema operativo en la maquina virtual.
- Por último hay que sincronizar estas dos carpetas. Esto se hace mediante el comando sudo mount -t vboxsf nombreCarpetaHost nombreCarpetaGuest
- Ya se pueden compartir archivos poniendolos directamente en dicha carpeta, tanto de host a guest como al reves.

File System

Como se explico en puntos anteriores, linux hereda el file system de UNIX. Es por esto que un hardlink apunta a la misma estructura que se encontraba presente en el file system de UNIX, es decir a los i-nodos. El i-nodo es un bloque que guarda información sobre el archivo o directorio al cual hace referencia; guarda toda la información necesaria no solo para poder referenciar el espacio físico del mismo, sino para tener la información pertinente.

El i-nodo guarda mucha información como el id del usuario owner, el id del grupo, que tan grande es el archivo y los punteros a los bloques físicos donde se encuentra el archivo. Pero, además, posee un campo importante que es el contador de hardlinks, esto indica cuantos hardlinks estan haciendo referencia a este mismo i-nodo.

Un hardlink es una referencia directa al i-nodo y al archivo o directorio en cuestion. Si se tienen dos hardlinks al mismo archivo en diferentes directorios, es como tener dos copias del mismo archivo, pero que en realidad hacen referencia a una sola y por lo tanto no hay desperdicio de bytes. Cabe destacar que si una copia es alterada, entonces todas las copias apuntadas por otros hardlinks son alteradas, ya que el archivo en cuestión es solamente uno.

A diferencia de estos, los softlinks nada mas son un acceso directo al archivo, si el archivo original se borra el softlinks queda huérfano.

La idea de que un i-nodo tenga contandores de hardlinks es para saber cuantas falsas copias del archivo hay dando vueltas. Entonces, se puede borrar un hardlink sin que el archivo sufra ningún tipo de cambio, ya que lo único que sucede es que el contador de hardlinks baja en uno, y los demás siguen apuntando al archivo como lo hacían antes. En el caso de que se borren todos los hardlinks, ya no quedan referencias al archivo, por lo que se puede borrar el mismo.

Prioridades

Para ejecutar los procesos en background se hace lo mismo que se explico en secciones anteriores. Es decir, se utiliza el ampersand para que se ejecute en un segundo plano y se manda el flujo de salida a /dev/null para que no se ensucie la consola. Luego, lo importante del ejercicio es ver como el loop3 se corre con mayor prioridad, para esto se utiliza lo explicado en administración del procesador, utilizar el comando nice para cambiarle la prioridad y poder así ver como toma más porcentaje del cpu.

La sintaxis utilizada entonces es: $./loop1 >> /dev/null \ \&$
$./loop2 >> /dev/null \ \&$
nice -n -18 $/home/grupo5/tp/loop3 >> /dev/null \ \&$

Luego, utilizando el comando top se puede ver cual es el tiempo de cpu que requiere cada proceso. Cabe destacar que al utilizar el comando top solo se observa el id del proceso, los ids de dichos procesos se obtienen al ejecutarlos en background. Luego de ejecutarlos en background, se imprime por consola el id del mismo.

Parámetros del Kernel

En primer lugar, vamos a utilizar un comando que ya se utilizó anteriormente en este trabajo. Dicho comando es el free, el mismo nos indica cuanta memoria se posee y cuanta se esta utilizando.

Luego, tenemos que lograr que el sistema use menos memoria, si bien se podría limitar que procesos se utilizan, la idea final de este punto es poder imponer una restricción mayor a una solucion ad hoc que sería ir deshaciendo de procesos a medida que la memoria ram sube. La idea entonces es restringir de entrada al sistema para que no puede usar más memoria ram que una cierta cantidad predispuesta.

Para esto, lo que se hizo, fue agregar una entrada a la lista del grub, para poder bootear la maquina en un modo donde la memoria ram se encuentre limitada por un parámetro puesto por el grupo.

Como crear una entrada al grub puede ser complicado debido a que desconocemos los parámetros que se utilizan, lo que se hizo fue copiar la entrada default, es decir, la entrada que se estuvo ejecutando todas las veces que iniciamos la máquina.

Luego, lo que se le hizo a esta entrada fue agregarle el parámetro mem=200M. La lista que se debe modificar se encuentra en De esta manera, la entrada, y por ende todos los atributos del sistema, son iguales a lo que teníamos anteriormente, pero con la diferencia de que tiene la memoria ram restringida. De esta manera, al reiniciar el sistema, se puede ver como el grub ahora nos da la opción que teniamos anteriormente, y una segunda opción que es el mismo sistema pero con la memoria ram restringida.

Una vez iniciada la máquina en esta nueva modalidad, se puede volver a utilizar el comando free para ver cómo cambió el uso de memoria ram.

Administración de la Memoria

En primer lugar, debemos determinar la memoria swap que se esta utilizando. Para esto volvemos a utilizar comando free, aunque hay varios comandos informativos que indican lo requerido. El comando free, nos indica cuanta swap se posee, y cuanta se esta utilizando en el momento.

Luego, debemos crear un nuevo archivo que lo vamos utilizar como swap, para esto utilizamos el comando dd. La sintaxis es la siguiente: dd if=/dev/zero of=/mnt/swapgrupo5 bs=4M count=10

Con esto lo que hacemos es crear un archivo que tiene 10 bloques de 4 megas.

Luego, tenemos que darle formato de memoria swap, para esto se utiliza el comando mkswap. Haciendo mkswap /mnt/swapgrupo5

Por último hay que agregarlo a la fstab para que el sistema lo reconosca como memoria swap que puede utilizar. Aquí se edita, mediante nano o por consola, el archivo fstab y se agrega la linea /mnt/swapgrupo5 none swap sw 0 0 al archivo /etc/fstab.

Luego, al reiniciar el sistema y utilizar el comando free, se puede ver que ahora se posee más memoria swap que lo que se poseía anteriormente.

Promoción

Módulo Mkdir

Para hacer este módulo el truco estaba en interceptar un systemcall, de esta manera se puede interceptar el mkdir para que no se pueda crear un directorio si el módulo esta cargado. La idea del módulo sería entonces crear las dos funciones más importantes, init y exit, para que ambas tengan el comportamiento deseado.

El problema para crear este modulo es que para poder interceptar una system call se debe tener un puntero a la tabla de system calls y esto no se puede hacer para versiones del kernel a partir de la 2.6 y justamente en este trabajo se esta utilizando una de estas versiones. El problema es que por cuestiones de seguridades, según lo que se encontro en la bibliografía, la tabla de la systema calls ya no se exporta, por lo que no se puede utilizar el puntero requerido y, por lo tanto, no se puede interceptar una system call de esta manera.

Se pueden encontrar varias soluciones a este modulo, pero todas explican que tiene que ser una versión menor a la 2.6 para poder realizarse, ya que todas las soluciones encontradas utilizan el puntero a la tabla de las system calls. En la cita bibliográfica número 1, se puede ver un ejemplo de solución, donde el autor explica que no se puede utilizar con estas nuevas versiones del kernel.

Módulo de probabilidad

Para realizar este módulo se hizo de manera análoga a los anteriores, el mismo se puede encontrar en la imagen en la carpeta de modulos.

A continuación explicamos las funciones básicas:

- En primer lugar se encuentra la función de inicialización del modúlo. Esta función abre un archivo en /proc y mapea el dispositivo /dev/probabilidad que fue creado anteriormente con el comando mknod.
- La función de exit del módulo borra el archivo.
- La función device read lee el archivo pertinente y devuelve el caracter aleatorio. Lo que se puede ver haciendo cat /dev/probabilidad.
- La función device write no tiene utilidad ya que solamente se lee el caracter del dispositivo.
- Luego se encuentran las funciones device open y device realease que abren y liberan el dispositivo respectivamente.
- La función procfile write no la usamos para grabar en el archivo ya que no usamos ninguna semilla. Si se quisiese ir cambiando la semilla para la función aleatoria, se debería utilizar esta función para escribir el archivo.
- Por último, la función procfile read se llama cuando leemos del archivo en proc.

Cabe destacar que el módulo fue insertado exitosamente, pudiendose constatar como se leen los caractéres aleatorios por el dispositivo creado.

Sistema operativo Windows

Administración de la memoria:

Los sistemas operativos windows usan una administración de la memoria paginada por demanda. Esto quiere decir que además de poseer la memoria paginada, posee la utilidad de tener memoria virtual para poder utilizar más memoria ram que la que realmente el procesador posee(Cabe destacar que esta utilidad comenzo a partir de Windows 3.0). Para realizar esto, el sistema operativo posee una memoria swap que la utiliza para hacer los intercambios entre las paginas presentes en la memoria principal y las memorias presentes en memoria secundaria. Cuando se esta ejecutando un proceso que hace una

referencia a una pagina que no esta en memoria principal, el memory manager se encarga de traer esta pagina de memoria secundaria y swapearla con alguna de la memoria principal. Puede suceder en estos casos que el memory manager no encuentre la pagina ausente que se esta queriendo traer a memoria principal, produciendo una de las conocidas pantallas azules de windows.

El memory manager también es el encargado de mapear la memoria virtual a la física, para este propósito Windows utiliza un sistema de paginación de varios niveles. En primer lugar existen directorios de tablas de paginas, los cuales poseen varias entradas, en una entrada de directorio de tabla de pagina podemos encontrar un puntero que nos lleva a la base una tabla de paginas. Luego, esta tabla de paginas posee varias entradas, cada entrada de la tabla de paginas posee un puntero hacia la página física propiamente dicha.

Por ejemplo, es un procesador intel de 32 bits el proceso es el siguiente:

- Se tiene una dirección de 32 bits, la cual se descompone en los 10 bits más significativos. Los 10 bits siguientes, y los últimos 12 bits.
- Con el registro del procesador CR3, se identifica la base del directorio de tabla de paginas y se utilizan los 10 primeros bits de la dirección original como indice dentro de este directorio.
- Una vez que estamos en la entrada correspondiente, podemos obtener la base de la tabla de paginas a la que hay que moverse. A esta base se le suma el índice que son los segundos 10 bits de la dirección original.
- En esta entrada de la tabla de páginas obtenemos la base de la página a la que se quiere referenciar.
- Luego de obtener la base de la página, nos deplazamos dentro de la página tanto como los 12 bits menos significativos de la dirección original lo indiquen.
- Luego de estos pasos, nos encontraremos en la dirección física correspondiente a la dirección virtual original.

Hay otras dos funciones importantes del memory manager de Windows. En primer lugar, tiene la función de proteger el espacio de direcciones de todos los procesos a partir de la validación de permisos para acceder a cierta parte de la memoria.

Por otro lado, a partir de Windows Vista en adelante, el memory manager esta encargado de ubicar dinámicamente la memoria correspondiente al kernel. En estas últimas versiones en el kernel es ubicado dinámicamente para

satisfacer mejor las necesidades del sistema, ya que una ubicación dinámica le provee una mayor flexibilidad a la hora de asignar las cantidades de memoria para cada uno de los procesos del sistema.

Scheduler de Windows: Al igual que en paginación, lo que explicaremos aquí es a partir de windows 3.0 ya que es necesario que un sistema sea multitasking para que tenga sentido hablar de un scheduler.

Los sistemas Windows actuales también están basados en prioridades como explicamos anteriormente para linux. Windows posee 32 niveles diferentes de prioridad, numerados del 0 al 31.

Las prioridades del 0 al 15 son las utilizadas para los procesos normales de los usuarios y del sistema. Mientras que las prioridades del 16 al 31 son para los procesos de tiempo real. En ambos niveles de prioridades, el número más bajo es el más prioritario, siendo el nivel 0 el nivel para los procesos del sistema.

Por un lado, para los procesos de tiempo real, el scheduler de Windows utiliza, al igual que el de linux, la técnica de Round-Robin. Esto quiere decir que les asigna un quatum a cada proceso y luego un proceso es desalojado si se le acaba su quatum en ejecución. Vemos que esta técnica es la mayormente utilizida por los sistemas operativos en la actualidad, ya que aseguran que todos los procesos de tiempo real puedan ir ejecutandose paralelamente.

Por otro lado, se encuentran los procesos que no son de tiempo real. En Windows, existen 16 niveles de prioridad diferentes, y cada nivel de prioridad se implementa mediante una cola de prioridad que va ordenando la ejecución de los procesos. Cabe destacar que, al igual que en linux como explicamos anteriormente, el scheduler de Windows penaliza a los procesos que son CPU-bound y beneficia a los procesos que son I/O-bound ya que se supone que al estar bloqueados por entrada/salidad, pierden mucho tiempo en esto y se les da mayor prioridad para que vuelvan a ejecutar pronto.

Bibliografía

1. http://tldp.org/LDP/lkmpg/2.6/html/x978.html
2. Understanding the Linux Kernel, Editorial O'reilly.
3. http://www.wikipedia.org
4. http://www.linux-es.org/node/127
5. http://www.oreillynet.com/linux/cmd

6. http://www.computerhope.com/unix

7. http://linuxcommand.gds.tuwien.ac.at

8. http://www.bulma.net/body.phtml?nIdNoticia=50

9. http://bloggerdigest.blogspot.com/2006/10/enable-linux-login-logout-scripts.html

10. http://www.linuxforums.org/forum/linux-programming-scripting/12551-simple-shutdown-script.html

11. http://www.laps3.com/foro/19_guias_tutoriales/23114-manual_de_shell_linux_comandos_bas

12. http://www.unix.com/es/shell-programming-scripting/111483-script-run-every-5-minutes.html

13. http://www.hedeshian.com/node/12

14. http://www.linuxtotal.com.mx/index.php?cont=info_admon_006

15. http://www.ace.ual.es/ vruiz/docencia/cursos/linux/html/node48.html

16. http://tldp.org/LDP/lkmpg/2.6/html/x1194.html