Introducción a LINUX

Linux es un sistema operativo, es decir, una interfaz entre el *hardware* de la computadora y los programas que el usuario corre en ella. Deriva de un *kernel* creado por Linus Torvalds, cuando aún era estudiante en la Universidad de Helsinki. Él estaba utilizando una versión de UNIX llamada "Minix", y le envió sugerencias a su creador, Andrew Tanenbaum, para modificar el sistema operativo. Sin embargo, Andrew no las creyó necesarias. Entonces, Linus se propuso crear su propio sistema operativo, bajo la ideología de que cualquier usuario podría sugerir ideas de mejoras.

Esta filosofía de recibir sugerencias de los usuarios para generar "*software* libre" ya existía desde los años '70; Richard Stallman, pionero en este espíritu, puntualizaba que "libre" significa "libertad", y no "de costo cero". Él trabajaba en el Massachusetts Institute of Technology (MIT), y encontró que era muy difícil continuar trabajando allí bajo esta filosofía; entonces, abandonó el MIT y fundó GNU, que desarrolla *software* de uso, distribución y modificación libre.

El punto central de todo sistema operativo es el *kernel*. Sin entrar en detalles, el *kernel* es quien le comunica al chip que controla la computadora qué hacer para que el programa que estamos usando haga lo que queremos que haga. Es decir, que es un puente entre las aplicaciones y el procesador, comunicando *software* y *hardware*, como se muestra en la Figura 1. Sin un *kernel*, un sistema operativo no existe. Sin programas, el *kernel* es inútil. Linus desarrolló el *kernel*, y GNU aportó los programas.

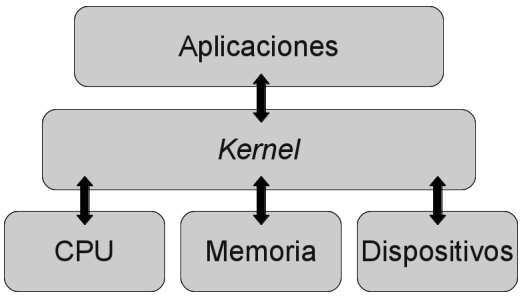

Figura 1: Comunicación entre *software* y *hardware*.

Hoy en día, Linux es accesible para todos los usuarios, y no sólo para expertos en el área, como era al principio. El sistema operativo ha demostrado ser súmamente estable y versatil, particularmente como servidor de redes. Linux, además, puede instalarse en cualquier PC sin costo alguno. Por lo tanto, ofrece mayor confiabilidad, al menor costo. Por otro lado, una de sus mayores virtudes es el hecho de que fue pensado como un sistema multi-usuario, es decir, varias personas pueden trabajar dentro del mismo sistema al mismo tiempo, y compartir programas y archivos.

1. Moviéndonos en los directorios y manipulando archivos

Cuando estamos en Linux y comenzamos a escribir en una pantalla negra, estamos utilizando una *shell*. Todos los sistemas operativos utilizan las *shells* para introducir comandos desde el teclado. La *shell* más popular utilizada en Linux es la *shell* bash (Bourne Again Shell), la cual utiliza juegos de palabras en los comandos, como vamos a ver a continuación.

1.1. Entrando y saliendo de los directorios: cd

Este comando sirve para entrar y salir de los directorios (cd = change directory).

- **cd**: nos traslada al directorio raíz
- **cd ..**: nos traslada al directorio directamente superior
- **cd /path**: nos traslada al directorio especificado en el camino *path*

1.2. Obteniendo información

La mayor parte de los comandos en UNIX admiten el uso de parámetros que modifican y/o amplían la acción que éstos ejecutan.

El comando **man** muestra el manual detallado de un comando o programa. El manual es un archivo que muestra cómo utilizar el comando y una lista de las diferentes opciones para el mismo.

Por otro lado, el comando **info** proporciona información sobre el comando o programa, generalmente más simple y de lectura más fácil.

- **man comando**: nos muestra el manual del comando "comando"
- **info comando**: nos muestra información acerca del comando "comando"

1.3. Viendo el contenido de un archivo de texto

Para ver en la consola el contenido total o parcial de algún archivo de texto (pero sin modificarlo), podemos hacer uso de alguno de los siguientes comandos:

- **less**: muestra en la consola el contenido total del arhivo. Nos podemos desplazar hacia arriba y hacia abajo para ver la totalidad del mismo

- **tail**: muestra las últimas líneas del archivo
- **head**: muestra las primeras líneas del archivo

1.4. Copiando y duplicando archivos

El comando **cp** (copy) nos permite copiar archivos desde un lugar hacia otro, o crear duplicados en el mismo directorio de trabajo (con otro nombre, claro).

- **cp archivo directorio**: copia el archivo "archivo" desde el lugar de trabajo hacia el directorio "directorio"
- **cp archivo copia**: genera un duplicado de "archivo" llamado "copia"
- **cp /path/archivo .**: copia el archivo ubicado en el camino *path* al directorio de trabajo
- **cp /path/archivo /path2**: copia el archivo ubicado en el camino *path* en el directorio ubicado en el camino *path2*
- **cp -r directorio directorio2**: copia el directorio "directorio" en el directorio "directorio2"

1.5. Algunos comandos importantes

- **ls**: nos muestra los archivos contenidos en el presente directorio
- **top**: nos muestra los procesos que están corriendo actualmente en la computadora, y de qué modo está siendo utilizada la CPU
- **ps**: muestra los procesos que estamos corriendo como usuarios
- **kill PID**: termina el proceso de PID (process ID) especificado
- **mkdir nombre**: crea un nuevo directorio llamado "nombre" (make directory)
- **rm nombre**: elimina el archivo "nombre" (remove). Para eliminar directorios, agregamos **-r** entre el comando y la carpeta que vamos a eliminar. ¡Cuidado! los archivos/directorios son eliminados para siempre...
- **mv**: sirve para mover archivos o directorios desde un directorio a otro, o para renombrarlos (move)
- **pwd**: nos muestra todo el *path* hasta nuestro directorio de trabajo

1.6. Comprimiendo y descomprimiendo archivos

El programa **tar** ensambla varios archivos en un sólo paquete, generalmente llamado "tarball". Por ejemplo, si tenemos varios archivos:

- datos1.dat
- datos2.dat
- datos3.dat
- datos4.dat

en el directorio **/metodos**. Para hacer un *backup* o para crear un sólo archivo que contenga todos los datos, hacemos:

tar -cvf notas.tar datos*.dat

Como se dijo anteriormente, **tar** sólo agrupa los archivos. No hay reducción en el tamaño de los mismos (¡incluso el tarball puede llegar a ser más grande!). Para comprimir los archivos, debemos utilizar la herramienta **gzip**. Para reducir el tamaño de un archivo .tar creado, hacemos:

gzip notas.tar

El resultado será el archivo **notas.tar.gz**.

Sin embargo, podemos crear directamente un archivo único que contenga todos los que queremos agregar, y además sea de tamaño reducido. Para ésto, sólo debemos agregar la *z* en la lista de opciones que agregamos al comando **tar**, y cambiar la extensión del nombre del archivo que queremos crear:

tar -cvzf notas.tar.gz datos*.dat

Es importante que la opción *f* se encuentre siempre al final (f:file (crear un archivo)).

Para descomprimir o desarmar el archivo .tar ó .tar.gz, hacemos:

tar -xvzf notas.tar.gz

Los archivos son desagrupados, y el archivo .tar.gz se conserva en el mismo directorio.

Las opciones más utilizadas del comando son:

- -z - *unzip* el archivo primero
- -x - extraer los archivos del *tarball*
- -v - (verbose) mostrar los archivos que se están agrupando/desagrupando
- -p - preservar las fechas y permisos de los archivos originales
- -f - utilizar el archivo en cuestión

1.7. Algunos programas que vamos a utilizar

- **nedit**: editor de textos
- **gnuplot**: graficador de datos en 2D y 3D
- **xmgrace**: graficador de datos en 2D
- **g77**: compilador de FORTRAN 77 (**g77** -o ejecutable programa.f)

Para más información, sugerimos visitar el sitio **www.linux.org**