HISTORIA DE DEBIAN

Que es Debian?

Debian es un sistema operativo libre, dicho sistema fue desarrollado por mas de mil personas voluntarias alrededor del mundo

El proyecto Debian fue creado en agosto del año de 1993 por Ian Murdock Debian fue patrocinado por el proyecto GNU fundado en Septiembre y creado por Richard Stallman con la finalidad de crear un sistema operativo completo de software libre. El nombre de "Debian" fue basado en una combinación de su novia (que actualmente es su ex esposa) Deborah con su propio nombre Ian así creando Debían.

La primer versión de debían fue la 1.x lanzada en 1996 mismo año en el que Bruce Perens sustituyo a Ian Murdock en el proyecto (desde abril de 1996)en su transcurso de líder Bruce Perens desarrollo BusyBox además de ser el autor de la definición del concepto de Código abierto.

También fue el fundador y primer líder del proyecto Linux Estándar Baseademás de eso Bruce es experto en gráficos y animación por computadora ya que trabajo por más de veinte años en pixar es por eso que los nombres de las versiones de Debian han sido tomadas de la película de Toy Story; Bruce Perens se retiró en diciembre de 1997, poco antes de la primera versión de Debían basada en glibc que es la biblioteca estándar de lenguaje C de GNU.

Debian ha tenido a varios líderes de proyecto tales como :

Ian Murdock que fue el fundador del proyecto debian en agosto de 1993 hasta marzo de 1996.

Bruce Perens condujo debian desde abril de 1996 hasta diciembre de 1997

Ian Jackson lo condujo dersde enero de 1998 hasta diciembre del mismo año

Wichert Akkerman condujo Debian desde enero de 1999 hasta marzo del 2001

Ben Collins condujo debian desde abril de 2001 hasta abril del 2002

Y actualmente martinmichlmayr fue elegido en marzo del 2003 y es el líder del proyecto en la actualidad.

ANTECEDENTES DE DEBIAN

Las siguientes versiones de Debian se llaman:

- Debian 6.0 (squeeze) — versión estable actual.
- Debian GNU/Linux 5.0 (lenny)
- Debian GNU/Linux 4.0 (etch)
- Debian GNU/Linux 3.1 (sarge)
- Debian GNU/Linux 3.0 (woody)
- Debian GNU/Linux 2.2 (potato)
- Debian GNU/Linux 2.1 (slink)
- Debian GNU/Linux 2.0 (hamm)

CARACTERISTICAS DE DEBIAN

Debian se caracteriza por:

- Debian es disponible para distintas plataformas la versión 3.1a es compatible con 11 plataformas.
- Una amplia colección de software disponible. La versión 3.1a viene con unos 15490 paquetes de software
- Un grupo de herramientas para facilitar el proceso de instalación y actualización del software.
- Su compromiso con los principios y valores involucrados en el movimiento del Software Libre.
- No tiene marcado ningún entorno gráfico en especial, ya sea GNOME, KDE u otro.
- Debian viene con más de 29000 elementos de software diferentes.
- Debian nace como una apuesta porseparar en sus versiones el software libre del software no libre.

Qué es Ubuntu?

Ubuntu es un sistema operativo basado en un núcleo de Linux y su origen está basada en debían. También es totalmente libre (no tiene costo) lo puedes bajar de internet, lo puedes usar y compartir con las personas que desees ya que puedes copiar o modificar sus versiones a terceros. Tiene como objetivo hacer que Linux sea un sistema operativo más accesible y fácil de usar. Cada seis meses se publica una nueva versión de Ubuntu la cual recibe soporte por parte de Canonical, durante 18 meses, por medio de actualizaciones de seguridad.

Sus antecedentes:

Ubuntu es una bifurcación del código base del proyecto debían. Su objetivo era hacer que debían fuera más fácil de usar y entender para los usuarios finales, corrigiendo varios errores de este y haciendo más sencillas algunas tareas como la gestión de programas. Su primer lanzamiento fue el 20 de octubre del 2004.

Antes de cada lanzamiento, se lleva a cabo una importación de paquetes, desde debían, aplicando las modificaciones especificas de Ubuntu. Un mes antes del lanzamiento, comienza un proceso de congelación de importaciones, ayudando a que los desarrolladores puedan asegurarse que el software sea suficientemente estable.

Desde el inicio del proyecto, Shuttleworth proporciono el soporte económico gracias a los beneficios obtenidos después de vender su empresa Thawte a VeriSign, por unos 575 millones de dólares estadounidenses.

El 8 de julio del 2005, Mark Shuttleworth y su empresa Canonical Ltd. anunciaron la creación de la fundación Ubuntu y aportaron 10 millones de dólares como presupuesto inicial.

En octubre del 2009 Canonical lanza fácilmente el centro el centro de software de Ubuntu, permite buscar, instalar, desinstalar aplicaciones y además permite agregar repositorios de terceros.

La más reciente versión estable de Ubuntu 11.10 (Oneiric Ocelot) 13 de octubre de 2011; hace 6 meses. Ultima versión en pruebas Ubuntu12.04 (Precise Pangolín) 22 de marzo de 2012; hace 26 días. El 31 de octubre de 2011, durante la presentación del

Ubuntu Developer Summit, Mark Shuttleworth anuncia la integración de Ubuntu en varios otros dispositivos, tales como tablets Smart TVs y Smartphones. Toda esta integración llegará en la versión 14.04, en abril de 2014.En enero de 2012, durante la feria tecnológica CES 2012, Canonical revela la interfaz de Ubuntu TV, el cual ofrece una manera simple e intuitiva de organizar contenidos y servicios para TV.

En febrero de 2012, Canonical anuncia 'Ubuntu for Android', el cual permite ejecutar el escritorio de Ubuntu directo desde un smartphones android al conectarse en un monitor. Cosas como la sincronización de contactos, sincronización de redes sociales, y vista de aplicaciones Android son posibles. Ubuntu para Android tiene compatibilidad con smartphones con múltiples núcleos ARM y la ventaja de compartir el mismo kernel con Android.

Estructura de archivos que usa Linux:

Esto es importante para quien le interese el mundo de Ubuntu.

Existe un estándar, el "*estándar de jerarquía de ficheros*" FHS – Filesystem Hierarchy Standard) que intenta definir unas bases, para lo que necesitan los usuarios. De un modo general podríamos decir que existen dos tipos de distinciones cuando hablamos del tipo de contenido de un directorio: Estáticos/dinámicos y compartibles/ no compartibles.

- Estáticos: contiene binarios, bibliotecas, documentación y otros ficheros que no cambian sin intervención del administrador. Pueden estar en dispositivos de solo lectura (read – only) y no necesitan que se hagan copias de seguridad tan a menudo como con ficheros dinámicos

- Dinámicos: contiene ficheros que no son estáticos. Deben de encontrarse en dispositivos de lectura – escritura (read-write). Necesitan que se hagan copias de seguridad a menudo.

- Compartibles: contiene ficheros que se pueden encontraren un ordenador y utilizarse en otro.

- No compartibles: contiene ficheros que no son compartibles.

A continuación se listaran algunos ejemplos:

Estáticos: /bin,/sbin,/boot,/usr/bin

Dinámicos: /var,/mail,/var,/run,/var,/lock,/home

Compartibles:/usr,/bin,/opt

No compartobles:/etc,/boot,/var,/run,/var,/loca

Todos los ficheros y directorios aparecen debajo del directorio raíz << / >> aunque se encuentren en discos /dispositivos distintos. En Linux/Unix no existen letras de discos (C:, D:, etc) los dispositivos empiezan a formar parte del directorios del sistema.

A continuación se tiene una lista con los directorios más importantes del sistema y para que se usan. Para acceder a los mismos se puede usar el comando cd “nombre del directorio”. Para ver el contenido de los mismos pueden usar el comando 1s -1 “nombre del directorio”.

/bin/comandos/programas binarios esenciales (cp, mv, ls, rm, etc)

/boot/ficheros utilizados durante el arranque del sistema (nucleo y discos RAM)

/dev/ dispositivos esenciales, discos duros, terminales, sonido, vídeo, lectores dvd/cd, etc

/etc/ ficheros de configuración utilizados en todo el sistema y que son específicos del ordenador

/etc/opt/ ficheros de configuración utilizados por programas alojados dentro de /opt/

/etc/X11/ ficheros de configuración para el sistema X Windows

/etc/sgml/ficheros de configuración para AGML (opcional)

/etc/xml/ ficheros de configuración para XML (opcional)

/home/ directorio de inicios de los usuarios (opcional)

/lib/ bibliotecas compartidas esenciales para los binarios de /bin/,/sbin/ y el nucleo del sistema.

/mnt/ sistemas de ficheros montados temporalmente.

/media/ puntos de montaje para dispositivos de medios como unidades lectoras de discos compactos. Nota: Ubuntu monta en este directorio las particiones Windows caso de existir.

/opt/ paquetes de aplicaciones estaticas.

/proc/ sistema de ficheros virtual que documenta sucesos u estados de nucleo. Contiene principalmente ficheros de texto.

/root/ directorio de inicio del usuario root

/sbin/ comandos/programas binarios de administración del sistema.

/tmp/ ficheros temporales

/srv/ datos específicos de sitio servidos por el sistema.

/usr/ jerarquía secundaria para datos compartidos de solo lectura (Unix system resources). Este directorio puede ser compartido por multiples ordenadores y no debe contener datos específicos del ordenador que los comparte.

Software libre:

el software a tenido un gran impacto de la competividad empresarial las cuales pueden medirse con números y graficas.

El software te ofrece cuatro libertades las cuales son:

* Libertad de utilizar el programa con el objetivo que desees.

* Libertad de aprender del programa y como usarlo de acuerdo a nuestras necesidades.

* Libertad de distribuir copias del programa para las personas que estén interesadas.

* Libertad para modificar el programa y distribuirlo para el beneficio de la comunidad.

El software libre ha tenido 5 éxitos:

* GNU/Linux

* Mozilla Firefox

* Apache

* Creative Commons

Aprovechamiento más adecuado de los recursos del software libre son aplicaciones las cuales son también utilizadas por otros sectores de la sociedad.

* Fomento de la industria local una de las ventajas es la posibilidad de desarrollar la industria local del software.

* Independencia del proveedor, depende de un mercado de régimen de competencia.
*adaptación a las necesidad, exactas ya que la adaptación puede hacerse con mucha mayor facilidad.

*disponibilidad a largo plazo, son muchos datos que estarán disponibles dentro de decenas de años.

Comandos de linux

Apt-get Hay diferentes Apt-get como son los siguientes y en donde explico de que trata cada uno de ellos:

apt y utilización basica:

Como ya explique, **apt**, es el gestor de paquetes de **Debian**. Su uso es desde el usuario administrador (**root**). Lo más basico que hay que saber:

apt-get install *nombre_del_paquete*: Con este comando, descarga e instala un paquete.

apt-get remove *nombre_del_paquete*: Para eliminar un paquete del sistema.

apt-cache search *descripción_o_nombre_del_paquete*: Realiza una busqueda en la base de datos. Devuelve nombre del paquete - descripción. Muy útil para encontrar algún software.

Mantenimiento del servidor

apt nos facilita la tarea de mantenimiento, reduciendola a dos comandos (todos los comandos a continuación deben ser corridos como **root**):

apt-get update, que actualiza las listas de los repositorios (sitios de donde se descargan los paquetes), busca en internet actualizaciones de la lista de paquetes y las descarga.

apt-get upgrade, luego de apt-get update, con este se actualizan los paquetes a la ultima versión.

updatedb, para utilizar el comando **locate *nombre_de_archivo***, genera una base de datos en base a los ficheros en el disco.

Cd - Nos cambia de directorio ejemplo:

cd: Cambio de directorio

Con `cd` cambiamos el directorio donde estamos trabajando.

Ejemplos: Cambio absoluto de directorio:

```
cd /usr/local/bin
```

Cambia al directorio citado,

```
cd subdir
```

Cambia al directorio subdir del directorio actual; si este no existe hay error.

```
cd
```

Cambia al "directorio base" o "*home directory*" designado por el super-usuario; en algunos sistemas esto lo puede cambiar el usuario. El directorio base esta grabado en la variable de shell `$HOME`.

Chown- En este tip tengo la intención de explicar como funciona el **comando chown** en Linux y hacer unos pequeños ejemplos.

El **comando chown** te permite modificar a los usuarios o grupos dueños de un archivo o carpeta en el sistema de archivos.

Algunas de las formas de utilizar el **comando chown** son las siguientes:

- chown usuario archivo o carpeta.
- chown -R usuario archivo o carpeta.
- chown usuario *

El primer ejemplo nos muestra que el nombre que coloquemos en donde dice **"usuario"** será el nuevo propietario del archivo o carpeta especificada.

El segundo ejemplo hace lo mismo que el primero a diferencia que afecta a todos los archivos y carpetas dentro de la carpeta indicada.

El tercer ejemplo cambia el dueño de todos los archivos y carpetas en donde se ejecute el comando.

Ejemplo:
chown -R psycho fotos/

chown: Comando Linux para modificar a los usuarios y grupos dueños de alguna carpeta o archivo.
-R: indica que los cambios se le deben de aplicar tanto a la carpeta especificada como a los archivos y subcarpetas.
psycho: nuevo propietario del archivo o carpeta.
fotos/: carpeta a la cual se le van a aplicar los cambios

También se pueden modificar los grupos asociados a dichos archivos o carpetas de la siguiente manera:

Ejemplo:
chown -R psycho.familia fotos/

Como ven es igual al primer ejemplo a diferencia que aparece "**.familia**" luego

del nombre de usuario, esto indica que ahora el grupo asociado al archivo o carpeta es familia.

Chmod- Para cambiar los permisos se utiliza el comando: `chmod permisos fichero`
Los permisos se pueden especificar de diferentes maneras, una serie de ejemplos, es lo mejor para comprenderlo:
chmod ugo+rwx test (da permisos rwx a todos, user,group,others)
chmod ugo-x test (quita permiso x (ejecucion) a todos, user,group,others)
chmod o-rwx test (quita permisos rwx a others)
chmod u=rwx,g=rx test (da permisos rwx a user, rx a group y ninguno a others)
Asi podriamos continuar con todas las posibles combinaciones de letras, es cuestión de usar la imaginación ;-)

Du - Descripción: =disk use. uso de disco. Muestra el espacio que esta ocupado en disco.
Ejemplos: du *, du -sH /*, du -sH /etc

Df - Descripción: = disk free. espacio en disco disponible. Muy util.
Ejemplos: df, df –h

Fdisk - Antes de instalar el sistema operativo en un disco duro, una partición de algún tipo se crea en el disco. Esto se hace habitualmente cuando se ejecuta la rutina de instalación, pero también puede hacerse utilizando el ksidf o los odnamocs parted.

Después de haber sido creada, la partición de se le asigna un tipo de sistema de archivos y, a continuación .. formato

Después de ser formateado, el sistema operativo y los otros componentes de software, tales como los odnamocs de GNU, servidores, un ordenador de sobremesa y otros componentes se instalan en él.

Puede hacer que las rutinas de instalación de la mayoría de las distribuciones de que hacen todo esto para usted.

Sin embargo, como administrador de sistemas tendrá que modificar las de después del sistema operativo ha sido instalado y para ello, es necesario ejecutar ksidf (para la mayoría de las tareas) o separado (para las tareas más complejas).

Grep - Encontrar un patron en una lista de archivos

El nombre `grep` es criptico: significa "*global regular expression and print*", pero su funcion es sencilla: encontrar un patron en una lista de archivos (por ejemplo, todos los archivos que contengan la palabra "CALL").

Para verdaderamente usar `grep` y muchos otros comandos de Linux hay que aprender el sistema de expresar patrones, llamado **expresiones regulares**, lo cual haremos en otra leccion (ver **). Por los momentos nos contentaremos con patrones simples. Grep tiene muchas opciones (ver manual) para buscar con o sin mayusculas, buscar archivos que **no** contienen al patron (especie de anti-grep), etc.

```
grep CALL *.f
```

Encuentra todas las lineas de todos los archivos que contienen la palabra CALL en todos los archivos que terminan en .f. Esto se podria usar como base a una tabla de referencias de un programa fortran.

```
grep juanr /etc/passwd
```

Busca la linea(s) de /etc/passwd que contiene(n) "juanr".

Mucho cuidado con patrones que contienen caracteres especiales del shell, como * y ?. Estos deben ser escapados, o usar comillas

Gcc - Para verificar que lo tiene, ingrese
`which gcc`

Si obtiene algo como
`/usr/bin/gcc`

, tiene bien el gcc
Si no, si siguió lo que se hizo líneas arriba, y de acuerdo a su distribución, puede utilizar
`urpmi gcc`

o
`apt-get install gcc`

o incluso
`yum gcc`

More - Ver archivos con control de pantalla

Sirve para examinar un archivo. Es preferible al uso de `cat`, que se usa para el mismo proposito, porque `more` permite retroceder, avanzar, o hacer busquedas. Por ejemplo

```
more xyz
```

Nos permite ver el archivo `xyz` en pantalla. Una vez entrado a `more`, se controla con los siguientes comandos, o mejor dicho "subcomandos:

- `espacio`-- adelanta una pantalla

- `b` (back) retrocede una pantalla
- `enter` avanza una linea
- /`patron` busca "patron" en el archivo
- `n` busca la proxima ocurrencia de el patron anterior
- `q` abandona el programa (salida).

En algunos sistemas de sabor a SysV, este mismo comando se llama `pg`.

Rm - Borrar archivos

`rm` borra archivos y con ciertas opciones, hasta directorios. **Advertencia**: este comando es **irreversible**.

Ejemplos:

```
rm xyz *.o
```

Este comando borra el archivo xyz y todos los archivos que terminan en .o. **PELIGRO**: La especificacion *.o, y todas las expresiones que contienen el asterisco, son sumamente peligrosas. Por ejemplo supongamos que por error se deja un espacio en blanco en el ejemplo anterior:

```
rm xyz * .o
```

Esto es desastroso: el asterisco borra todos los archivos, silenciosamente, y despues se queja que "no puedo encontrar .o". La mejor manera de evitar este accidente es con cuidado. Otra posibilidad es de usar la opcion interactiva:

```
rm -i xxx xyz
```

Esto pregunta antes de efectuar la remocion.

Con la opcion recursiva, se puede borrar un directorio y todos los archivos dentro de ese directorio. Esto es equivalente a hacer cd a ese directorio, borrar todos los archivos, subir, y hacer `rmdir`. Use esta opcion con sumo **cuidado**.

```
rm -r direc
```

Esto borra el directorio `direc` y todo lo que pueda estar por debajo de direc.

Sudo - Los usuarios deben confirmar su identidad al ejecutar *sudo* dando su propia contraseña antes de ejecutar el programa requerido. Una vez se ha autenticado el usuario, y si el archivo de configuración `/etc/sudoers` permite dar al usuario acceso al comando requerido, el sistema lo ejecuta y lo registra.

Si en un inicio y para evitar el constante ingreso del comando sudo en cada comando desde un inicio podemos aplicar los siguiente:

sudo su -

A continuación pedirá el password del usuario que ejecuto el comando y si esta autorizado en el archivo sudoers, se iniciara sesión con el usuario root.

El archivo de configuración `/etc/sudoers` especifica qué usuarios pueden ejecutar qué comandos en nombre de qué otros usuarios. Como *sudo* es muy estricto con el formato de este archivo, y cualquier error podría causar problemas serios, existe la utilidad *visudo*; ésta permite al usuario *root* editar el archivo y luego revisar su corrección antes de guardarlo. Un aspecto muy importante, si se desea editar el fichero *sudoers*, por ejemplo para modificar el timeout de sudo, es que bajo ningún concepto se le deben cambiar los permisos (los permisos que tiene asignados son 0440 y no deben modificarse), puesto que hacerlo inutiliza la utilidad *sudo* y solamente podría subsanarse el problema desde un Live-CD.

Tar -

Comprimir y descomprimir archivos es una de las tareas mas comunes que vamos a tener que realizar en Linux por línea de comandos. Podemos hacerlo mediante diversas herramientas, pero existen dos que van a estar disponibles en casi cualquier distribución.

Son dos herramientas que se tienen que utilizar de manera complementaria. Se trata de tar, que simplemente empaqueta varios archivos en un único fichero, y gzip, que hace la compresión propiamente dicha.

El comando tar tiene la siguiente sintaxis:

tar [opciones] [origen] [destino]

Donde opciones son, por ejemplo, las siguientes:

- c: crear un archivo
- x: extraer de un archivo
- t: listar los contenidos de un archivo
- v: ver un reporte de las acciones a medida que se van realizando
- f: empaquetar contenidos de archivos
- z: para comprimir a la vez que se empaqueta

Como se puede ver, con la opción z se puede comprimir en el mismo paso en el que se empaqueta, lo que puede hacer las cosas más rápidas y cómodas. De todos modos, tar simplemente hace el empaquetado y es gzip el que realiza la compresión. Simplemente que nosotros no tenemos que llamar a gzip, sino que ya lo hace directa e internamente tar.

Ejemplos

1) Si queremos empaquetar un directorio llamado "html" y guardar los datos en "html-paq.tar", lo haríamos con la sentencia:

tar cvf html-paq.tar html

2) Si queremos comprimir un directorio llamado "archivos" y guardarlo en un fichero llamado "archivos-comp.tgz", `podríamos hacer algo como esto:

tar czvf archivos-comp.tgz archivos

Si nos fijamos, en este caso el nombre del archivo comprimido le hemos puesto extensión .tgz, que indica que está empaquetado y comprimido.

3) Si queremos desempaquetar un archivo llamado xxx.tar podemos utilizar un comando como este:

tar xvf xxx.tar

Eso supone que tenemos el archivo xxx.tar en el mismo directorio desde donde lanzamos el comando. Los contenidos se desempaquetarán en el mismo directorio donde estamos situados.

4) Si el archivo que queremos desempaquetar está comprimido. Es decir, si queremos descomprimir y desempaquetar en un solo paso, tenemos que hacerlo con la opción z del comando tar. Algo como esto:

tar xzvf xxx.tgz

Tail - Ver las ultimas n lineas de un archivos

tail da las ultimas 10 lineas de un archivo, u opcionalmente las ultimas n lineas del archivo. Por ejemplo

```
tail xxx
```

Da las ultimas 10 lineas del archivo xxx, mientras que

```
tail -1 xxx
```

Da la ultima.

Este comando es util para seguir el progreso de un programa de larga corrida.

Vi - **vi** es un clásico editor en ambientes tipo UNIX cuya prioridad es la eficiencia. Por lo mismo no requiere el uso de ratón sino de breves comandos que a medida que se memoricen y usen harán más rápida la edición de textos.

Para editar un archivo (digamos `ma.txt`) con el editor **vi**, teclee desde un intérprete de comandos:

```
$ vi ma.txt
```

Cuando **vi** cargue, se observará en la pantalla una parte del archivo de texto que va a editar. Al arrancar **vi**, está en un modo especial llamado `modo comando` ó `modo normal` . Esto significa, por ejemplo, que si presiona

Wc - **wc que te cuenta líneas, palabras y caracteres.**

La sintaxis es la siguiente:

wc -parametro fichero

Tiene tres parámetros posibles:

-l que cuenta sólo las líneas de un fichero

-w cuenta las palabras de un fichero

-c cuenta los caracteres de un fichero

Find - El comando `find` se usa para encontrar archivos en el arbol de directorios de Linux. La estructura de directorios puede ser arbitraria. `find` requiere un punto de partida y las caracteristicas del archivo a encontrarse. Despues, `find` revisa ese directorio y todos los directorios subordinados, buscando los archivos que cumplan la condicion(es) citada(s).

Lo examinaremos en mas detalle en otra leccion, (ver **) pero algunas formas comunes son:

```
find . -name perdido -print
```

Esto busca en el directorio actual (.) todos los archivos o directorios de nombre `perdido` y pone el resulotado a pantalla (-print). En algunas versiones modernas de Linux, la opcion -`print` no es necesaria, pero en otras si (de lo contrario no pasa nada!)

```
find /usr/people -name '*.f' -print
```

Busca, a partir del directorio /usr/people, todos los archivos que terminen en `.f`. El uso de las comillas es **indispensable** porque de lo contrario, el shell sustituye por el asterisco los nombres de todos los archivos en el directorio de partida.

Dpkg - Utilizar **dpkg** nos puede ser muy útil y **acelerará el proceso de instalación de paquetes** .deb que obtengamos nosotros **sin utilizar apt o synaptic**. A mi personalmente, **me parece muy incómoda y lenta la interfaz de Gdebi** (el instalador gráfico de paquetes .deb), ya que este a veces **tarda mucho en cargar** y nos hace perder tiempo, pudiendo **acelerar el proceso con dpkg**.

Veamos como utilizar dpkg para las opciones más comunes:

Para obtener una lista completa de los paquetes instalados en el sistema, ejecutamos lo siguiente:

dpkg -l

Si queremos información detallada de un determinado paquete de la lista anterior, ejecutamos:

dpkg -s paquete

Para obtener la información completa (dependencias, descripción, versión…):

dpkg -p paquete

Si queremos ver un listado de los componentes que proporciona un paquete instalado:

dpkg -L paquete

En cambio, si el paquete no esta instalado y queremos ver los componentes que provee, ejecutamos:

dpkg -c nombredelpaquete.deb

Para saber a que paquete pertenece un fichero o otro componente, ejecutamos:

dpkg -S /directorio/fichero

Para instalar un paquete .deb, ejecutamos:

dpkg -i nombredelpaquete.deb

Para instalar todos los paquetes del directorio actual, ejecutamos:

dpkg -i *.deb

Si queremos instalar o actualizar el paquete .deb, ejecutamos:

dpkg -Gi nombredelpaquete.deb

Para desinstalar un paquete ejecutamos:

dpkg -r paquete

Para desinstalarlo y borrar los archivos de configuración, ejecutamos:

dpkg -P paquete

Para desinstalarlo y borrar los archivos de configuración, ejecutamos:

dpkg -P paquete

Como vez, **el uso de dpkg es muy simple**. Su sintaxis es muy sencilla. Recuerda que para realizar acciones con determinados paquetes .deb no instalados debes situaros en el directorio en el que se encuentre usando cd /directorio/carpetadondeseencuentraelpaquete.

Usando dpkg ahorraremos tiempo instalando paquetes y nos no tendremos que utilizar la interfaz gráfica.

Update-grub Lo primero de todo es generar el menu.lst, ejecutando:

```
# update-grub
```

NOTA: Edita manualmente el archivo /boot/grub/menu.lst si fuera necesario para arrancar tu sistema Debian (ejemplo, si necesitas desactivar desde GRUB el soporte para USB, PCI, ACPI, etc.).

Nano -

Es un editor de textos en modo consola muy sencillo e intuitivo el cual es rapido cuando deseamos editar algun archivo de texto o de configuracion en linux, Es de gran ayuda cuando perdemos por alguna razon conocida el entorno gráfico.

Para ejecutar el editor primero. se debe abrir una consola Aplicaciones—->Accesorios——>Terminal

Escribimos en la consola el nombre del programa en este caso “nano” seguido de un espacio luego el nombre del archivo a crear “nombredelarchivo.txt

Ls-l Probablemente el comando mas usado en Linux, `ls` nos permite ver el contenido de un directorio y opcionalmente sus subdirectorios. Este comando tiene muchas opciones. La forma mas corriente es simplemente

```
ls
```

Que lista en varias columnas los nombres de los archivos en el directorio actual. Otra variante comun es

```
ls -l
```

lo cual da el listado largo (permisos, tama#os, due#o, etc.). Una opcion util es

```
ls -FC
```

que le pone a los ejecutables un asterisco, a los directorios la barra /, y a los archivos comunes nada (pruebe esto en us sistema). `ls` tambien acepta especificaciones:

```
ls -l *.f
```

Esto da todos los archivos que terminan en .f, con listado largo. Tambien podemos ordenar la lista de varias maneras, por ejemplo por edad:

```
ls -lt *.f
```

Esto nos lista en edad descendiente (mas viejo de ultimo) los archivos en que terminan en .f, con listado largo.

Aptitude- aptitud, habilidad. En el fondo juega con las siglas de apt para crear aptitude. Es una versión mejorada de apt

- Para abrir el interfaz gráfico de aptitude, tan sólo hay que teclearlo:

$ aptitude

Sin embargo, también se puede usar exactamente igual que apt, pero con las características que he comentado de aptitude:

$ aptitude search nombre_paquete
$ aptitude install nombre_paquete
$ aptitude remove nombre_paquete
$ aptitude purge nombre_paquete
$ aptitude update
$ aptitude upgrade

Head - Se usa para ver las primeras lineas (cabeza) de un archivo. Por defecto, se ven 10 lineas, pero esto sae puede cambiar. Por ejemplo,

```
head /etc/passwd
```

Pone en pantalla las primeras 10 lineas de /etc/passwd, mientras que

```
head -2  /etc/passwd
```

pone en patalla las primeras dos.