1. INTRODUCCION

Linux es un sistema operativo de 32 bits diseñado para su uso en PC´s , basados en Intel 80386 ó superior. Técnicamente, Linux funciona de forma similar a UNIX, lo que supone que responde a los comandos estándar de UNIX y ejecuta sus programas, además Linux se adhiere a las especificaciones POSIX, con algunas extensiones BSD y System V.

Linux nació como proyecto de un solo hombre, Linus Torvalds, que en el momento de la creación de Linux estudiaba en la Universidad de Finlandia, en Helsinki. Linux Torvalds necesitaba una alternativa propia a otra de las alternativas de UNIX, en particular, el sistema operativo Minix, un sistema comercial parecido a UNIX diseñado para trabajar con PC´s. Torvalds diseñó Linux de forma similar a Minix, de hecho el sistema de archivos original de Minix fue incorporado a Linux, pero haciéndolo más estable y libremente accesible. Torvalds posee aún los derechos del núcleo de Linux, pero permite el libre acceso bajo las condiciones del GNU General Public License. Durante largo tiempo Linux fue un sistema operativo en desarrollo, como otras muchas versiones de Linux que circulaban a través del mundo informático, en su mayoría distribuidas por la red de internet. La versión 0.2 se lanzó al mundo a mediados de 1991; en 1993, la versión 1.0 fue finalmente lanzada. Un grupo de usuarios se prestaron voluntarios para ayudar a Torvald a terminar Linux y contribuyeron, además, a crear el software adicional que ayudó a hacer de Linux un sistema operativo popular. Después se fueron creando diversas distribuciones de Linux, en las que hay que destacar Red Hat Linux, Debian Linux, Slackware Linux, Suse Linux, Corel Linux y en la que basaremos este trabajo Esware Linux.

Ante la gran variedad de distribuciones de Linux uno se hace la pregunta, ¿Por qué existen múltiples distribuciones de un mismo software?, la respuesta es sencilla, no todas las distribuciones de Linux son iguales. Todas están basadas en el núcleo Linux, que se puso a libre disposición del mundo bajo las condiciones del GNU General Public License. Este acuerdo sanciona la libertad de venta o libre disposición de Linux, siempre y cuando se incluya el código fuente y, en el caso de que se realicen modificaciones al paquete, se proporcionen los códigos fuentes de tales modificaciones. Quienes lo utilicen y distribuyan quedan bajo el acuerdo GPL y se dice que posee un "copyleft", que es lo contrario de un "copyright". Las distintas distribuciones son muy parecidas en muchas formas, pero incluyen distintas operaciones de instalación y algunas diferentes contribuciones de terceros.

Linux, en cuanto a sistema operativo, es bastante elegante y adecuado. El núcleo básico de Linux puede instalarse mediante un conjunto de tres disquetes, pero este núcleo básico se limita a poco más de una línea de comando y responde a un número limitado de éstos, las distribuciones de Linux suelen venir en CD−ROM, ya que tienen una gran multitud de aplicaciones además del núcleo del sistema operativo. En este trabajo veremos las características y ventajas de Linux, empezando por su modo multiusuario, siguiendo por sus dispositivos de seguridad ante otros usuarios, tanto de su propio ordenador, como de los conectados a la red. Por último veremos la gran característica de Linux, Internet.

2.¿QUÉ HACE ESPECIAL A LINUX?

Estas son las principales características que hacen especial a Linux en el mundo de los sistemas operativos:

Linux es una alternativa a los sistemas operativos comerciales. Linux es el resultado de muchas horas de trabajo de voluntarios, que creen que una aproximación a nivel básico que las hinchadas ofertas comerciales. Que se compartan estos valores es una opción de cada cual, pero no es posible evitar quedar impresionado por un sistema operativo excepcional, rico en complejidad y características.

Linux se ha creado para internet y gestión de redes. La red de Internet es una parte importante del mundo

de la informática. Probablemente, Linux no habría al lugar que ocupa hoy sin la red de Internet; cientos de voluntarios participan a través de ella, enviando códigos fuente y archivos de programa de ordenador en ordenador. Linux ofrece la posibilidad de usarlo como servidor de Internet, por sí sólo, algo que otros sistemas operativos no pueden conseguir.

Linux es completamente abierto. Además de ser Linux un producto de libre distribución, nos ofrece la posibilidad de llevar a cabo cambios en nuestro sistema operativo, o conseguir *drivers* para los nuevos periféricos que vamos incorporando, todo esto, mediante la página Linux de internet.

Linux es un sistema operativo multitarea. Linux es capaz de hacer más de una tarea simultáneamente y de asignar verdaderas preferencias a las tareas cundo éstas necesiten prioridades distintas.

Linux es un sistema operativo multiusuario. Podemos instalar Linux en un servidor y conectar otros usuarios al mismo, de manera que trabajen bajo el mismo sistema operativo, con la ventaja de poder tratar unos los ficheros de otros, bajo el mismo entorno.

Linux es un sistema operativo eficiente. Linux se construyó para procesadores Intel 80836, el ancestro tecnológico de i486 y procesadores Pentium y se aprovecha de las características de la familia de procesadores Intel. Linux suele disponer de una memoria de protección entre procesos, éste impide que un programa en mal estado colapse por completo el sistema operativo, algo muy frecuente en otros sistemas operativos. También dispone de páginas copy−on−write entre los ejecutables, que disminuyen los usos de memoria e incrementa su velocidad de ejecución. Por último presenta una memoria virtual, swap, que intercambia secciones de memoria, no procesos completos, con el disco cuando la memoria escasea.

Acceso a la información de un activo grupo de usuarios. Hay en número estimado de usuarios de Linux de unos 7.5 millones y buena parte de ellos participan en discusiones en el usenet acerca de la resolución de problemas en Linux, este soporte técnico improvisado es totalmente fiable, incluso en 1997 la revista InfoWorld otorgó el premio al "Mejor Soporte Técnico" a la comunidad de usuarios de Linux, por encima de soportes técnicos profesionalizados como Microsoft, Apple Computer, Oracle.

3.SISTEMA OPERATIVO LINUX

Linux es un sistema operativo de 32 bits diseñado para su uso en PC´s basados en Intel 80386 o superior, técnicamente Linux funciona de forma similar a UNIX. Linux es un sistema operativo multitarea y multiusuario, lo que quiere decir que puede haber más de una persona utilizando un ordenador a la vez, cada uno ejecutando a la vez diferentes aplicaciones, para la distinción de los distintos usuarios de un sistema se utiliza el *login*, que es una palabra clave elegida por el usuario que a su vez va protegido por un password para que así nadie pueda presentarse en el sistema con un nombre de usuario que no le corresponda. Además de estos marcadores identificadores cada sistema Linux posee un nombre del sistema, el *hostname*, que es otra palabra clave predeterminada, el hostname se utiliza para diferenciar sistemas conectados en red, con todo esto el prompt de Esware Linux está compuesto por: el nombre del usuario +@ el nombre de la máquina + el directorio actual + símbolo $ (en el caso de que estuviésemos en el usuario universal root el símbolo final sería #).

2.1 Ficheros y directorios linux

Antes de meternos en materia definiremos fichero como un conjunto de información al que se le ha asignado un nombre común y directorio como una colección de ficheros almacenados todos en un mismo lugar. Los directorios pueden contener a su vez otros directorios, recibiendo el nombre de *directorio padre* el que contiene al subdirectorio. No existe un formato estándar para los nombres de los ficheros, como ocurre en otros sistemas operativos, el único requisito impuesto por Linux es el de no contener el caracter " / " y tener una longitud de caracteres menor de 256 además de todo esto hay que tener en cuenta que Linux diferencia las

letras mayúsculas de las minúsculas. Cada fichero está definido por su nombre y su ruta de acceso, de tal manera que puede haber dos ficheros de igual nombre siempre y cuando estén situados en diferentes directorios, además cada fichero posee su inodo, y aunque pueden existir dos ficheros con el mismo inodo aunque para Linux serían el mismo. La ruta de acceso de un fichero está compuesta por su nombre antecedido del directorio que le comprende mediante el carácter "/", en el caso de que este directorio estuviese comprendido dentro de otro directorio padre, primero se escribiría el nombre del directorio padre seguido del carácter " / " (d*irectorio padre/directorio/nombre del fichero).*

Para la fácil localización de los ficheros Linux Esware está compuesto por una organización de directorios conocida como árbol de directorios. Está organización consiste en la existencia de un directorio padre que comprende a cualquier otro directorio, conocido como directorio raíz y representado con el carácter " / ", este directorio raíz comprende a otros directorios como /bin, /usr etc. Otro de los subdirectorios contenidos en el directorio raíz es el directorio /home, propio de cada usuario y será donde el usuario guardará sus ficheros personales.

Ahora veremos con más detalle el sistema de ficheros que es conocido como la colección de ficheros y la jerarquía de directorios de su sistema, este sistema de ficheros está compuesto por los siguientes directorios:

- **/bin:** Este directorio contiene la mayoría de los programas esenciales del sistema, la gran mayoría de los ficheros contenidos en este directorio son ficheros ejecutables, estos ficheros ejecutables llevan un asterisco (*) añadido al final de sus nombres, cuyo símbolo es el utilizado por Linux para distinguirlos de los demás tipos de ficheros.
- **/dev:** El directorio /dev contiene los ficheros conocidos como controladores de dispositivo o driver, estos ficheros son utilizados para a los dispositivos y recursos del sistema como discos duros, modems etc.. Los ficheros que comienzan su nombre con *fd* son controladores de disqueteras, por ejemplo fd0, es el controlador de la primera disquetera. Los ficheros que comienzan por *hd* acceden a discos duros o particiones de ellos, por ejemplo el dispositivo /dev/hda1, hace referencia a la primera partición del disco duro, mientras que el dispositivo /dev/hda haría referencia a la totalidad del primer disco duro. Los dispositivos que comienzan por *sd* son dispositivos para acceder a discos duros SCSI. Los ficheros que comienzan por *lp* acceden a los puertos paralelo. Además de estos ficheros y otros no mencionados que recoge el directorio /dev se encuentra dentro de éste el dispositivo /dev/null que es utilizada para destruir datos enviándolos a este dispositivo.
- **/etc:** Contiene los ficheros de configuración del sistema.
- **/sbin:** Es un directorio en donde se almacenan programas del sistema que son usados por el administrador.
- **/home:** Es el directorio personal de cada usuario.
- **/lib:** Contiene códigos usados por varios programas, reduciendo así el espacio utilizado en el disco duro mediante la omisión de estos códigos en los programas ejecutables.
- **/proc:** Sus ficheros residen en la memoria y contienen información de los programas y procesos ejecutados en ese momento.
- **/tmp:** Formado por ficheros temporales que contienen información temporal de los programas en ejecución.

Todos estos directorios son imprescindibles para que Linux funcione, a continuación veremos el directorio causante de hacer de Linux un sistema operativo competente y con multitud de prestaciones y aplicaciones el directorio /**usr:**

- **/usr/X11R6:** Contiene toda la información de X Window, el entorno gráfico de Linux, tanto ficheros de ejecución, como de configuración o de soporte.
- **/usr/bin:** En este directorio residen el resto de programas ejecutables de Linux que no se encuentran en /bin, podríamos decir que aquí se encuentran los archivos ejecutables de los programas añadidos por el directorio /usr.
- **/usr/etc:** Al igual que el anterior directorio contiene los ficheros de configuración y programas del sistema de los nuevos programas integrados por el directorio /usr.

- **/usr/include:** Contiene los ficheros de cabecera para el compilador C, haciendo una programación menos complicada. Al igual que este directorio el directorio /usr/g++include contiene los ficheros de cabecera del compilador C++.
- **/usr/lib:** Tiene la misma función que "su hermano mayor" /lib, aunque limitada únicamente a programas contenidos en /usr.
- **/usr/local:** Añade más ficheros y programas a los ya añadidos en el directorio /usr, este directorio es la principal diferencia entre unas distribuciones Linux y otras ya que se incluyen los programas característicos de cada fabricante Linux.
- **/usr/man:** Este es el directorio de ayuda de Linux, en donde incluye sus manuales sobre los distintos programas.
- **/usr/src:** Contiene los códigos fuentes de todos los programas incluidos, entre ellos el del núcleo de Linux.

Además de estos directorios Linux posee el directorio /var que contiene ficheros que van a ser pasados a otros programas y por lo tanto éstos cambian de tamaño por lo tanto, aunque antes este directorio estaba incluido dentro de /usr es mejor tenerlo aparte para una mejor accesibilidad y para que estos continuos cambios no afecten a otros ficheros o programas.

2.2 COMANDOS

Linux posee una enorme cantidad de comandos, aunque no veremos todos, sí veremos los comandos más importantes para poder moverse con soltura por con soltura dentro de un entorno Linux, además Linux nos ofrece una gran gama de distintas asociaciones de comandos a las que también haremos referencia, pero primero veamos como trabaja el intérprete de comandos, que es el que se encarga de hacer entender a la máquina lo que hemos puesto en la pantalla.

2.2.1 El interprete de comandos

El intérprete de comandos es una de las interfaces que el usuario tiene con Linux, el intérprete de comandos es un programa que lee las entradas impuestas por el usuario que posteriormente son traducidas a un lenguaje que la máquina es capaz de entender y utilizar.

Para que el intérprete de comandos interprete una orden ésta ha de tener una sintaxis concreta, la sintaxis correcta que es interpretada por el intérprete es " orden −opciones argumentos" por ejemplo por ejemplo "*ls −nl datos facturas"* mostraría información en formato largo de los archivos de datos y de facturas. En el ejemplo anterior la orden es "ls", las opciones "n y l" y los argumentos serían "datos y facturas", puede ocurrir que algunas órdenes no les sea necesario añadir opciones o argumentos para poder ser ejecutadas. En el PATH se encuentran las direcciones de las órdenes, para poder ser encontradas y ejecutadas por el intérprete de comandos. Por lo tanto cuando escribimos una orden el intérprete de comandos busca el programa en el PATH y lo ejecuta.

Las órdenes básicas de Linux son:

- cd: con la orden cd podemos desplazarnos a un subdirectorio del directorio en el que actualmente estamos trabajando con solo teclear su nombre a continuación de cd. Aunque si no ponemos ningún directorio, el intérprete de comandos, por defecto nos enviará al directorio de origen, exclusivo de cada usuario.
- cd.. : Mediante esta orden el intérprete de comandos nos devuelve al directorio padre con respecto al directorio en donde estamos trabajando.
- ls: Este comando nos ofrece una lista de ficheros y subdirectorios del directorio elegido, si no se fija ningún directorio entonces por defecto se ofrecerá la lista del directorio actual de trabajo. En esta lista no se nos ofrece si lo que estamos viendo su ficheros o son subdirectorios, para lo cual tenemos que usar la orden ls −F. Con

ls −l se nos muestra los permisos de acceso para un usuario a un fichero, según sea grado de enlace con el creador del fichero. La ls −i nos muestra un el número de inodo del fichero elegido.

- more: Se utiliza para ver el contenido de los ficheros.
- mkdir: Nos permite crear un nuevo directorio con el nombre que queramos, respetando el régimen de caracteres que Linux sigue.
- mv: En cambio esta orden mueve los ficheros en lugar de copiarlos y no solo experimenta cambios en los ficheros del directorio del que son copiados, sino que además, si en el directorio que acoge al fichero se encuentra uno con su mismo nombre es automáticamente sobrescrito.
- cp: Esta orden copia ficheros trasladándolos de un directorio a otro sin moverlos de su directorio actual, por lo que el directorio copiado no experimenta cambio alguno. Esta orden también sobrescribe el fichero que tenga el mismo nombre que alguno de los ficheros que se están aposentando en el directorio.
- rm: Esta orden es utilizada para borrar un fichero.
- rmdir: Para borrar directorios éste será el comando elegido, pero sólo podran ser borrados si se encuentran vacíos, para así poder evitar que se borren ficheros o subdirectorios accidentalmente.
- man man: Activando este comando la pantalla nos muestra un manual del programa en curso. Incluso si queremos echar un vistazo a la lista de manuales nos bastaría con teclear la orden "*man* " y a continuación el comando o programa sobre el que han surgido dudas, por ejemplo "*man ls*" muestra el manual de la orden ls.
- clear: tecleando esta orden la pantalla se borra totalmente mostrando únicamente el prompt en la parte superior izquierda.
- passwd: Cambiará la contraseña actual del usuario, para lo cual ha de introducir la vieja contraseña.
- adduser: Mediante la ejecución de esta orden el intérprete crea una nueva cuenta.
- startx: Este es el comando utilizado para ejecutar X Windows, el entorno gráfico de Linux, en este entorno puedes ejecutar órdenes, comandos, o programas con el simple uso del ratón y del teclado, lo que hace más fácil de entender este sistema operativo.
- chmod: Comando utilizado para establecer o cambiar los permisos de un fichero, este comando sólo puede ser utilizable por el creador del fichero, o en su defecto por el administrador del sistema.
- ln: Mediante este comando se crean enlaces duros entre ficheros.
- ln −s: Comando utilizado para crear enlaces simbólicos entre ficheros, tanto la utilización de enlaces duros, como simbólicos se verá posteriormente.
- mount: Se utiliza para montar los sistemas de archivos. Por ejemplo montar el sistema de archivos de un disquete en un directorio.
- unmount: Esta es la orden opuesta a mount, luego se utiliza para desmontar un sistema de archivos.
- eject: Mediante la utilización de este comando se conseguirá la expulsión de la unidad de CD, siempre y cuando esta no esté en uso.
- tar: Este comando activa el empaquetador tar.
- gzip: Este comando activa el programa compresor gzip.
- cat: Es un comando utilizado principalmente para ver el contenido de los ficheros, aunque también se puede usar para copiar ficheros cambiándolos de nombre y multitud de funciones más.
- head: Comando utilizado para examinar las primeras líneas de un fichero, esto es muy útil ya que si tan solo queremos echar un vistazo al fichero esta forma de verlo es más rápida y usa menos memoria.
- tail: Este comando es utilizado para examinar las últimas líneas de un fichero.
- find: La función de este comando es buscar el fichero deseado en todos los directorios de Linux.
- date: Este comando muestra en pantalla, la fecha y la hora en la que el sistema se encuentra en esos momentos.
- cal: Muestra el calendario de un mes y de un año determinado, sino se marca ningún mes en concreto, en la pantalla aparece por defecto el mes actual en el que se encuentra el sistema.
- lpr: Si queremos imprimir un fichero, debemos teclear este comando seguido del nombre del fichero a imprimir.
- file: comando que permite determinar el tipo de datos que contiene un determinado tipo de archivo.
- exit: Es el comando utilizado para salir del sistema operativo Linux.

2.2.2 Agrupación de comandos

Ante la gran cantidad de órdenes y comandos que Linux nos ofrece la posibilidad de poder agruparlas y hacer que varias funcionen a la vez, nos permite que las prestaciones que nos ofrece Linux sean cada vez más importantes, mediante estas agrupaciones de órdenes poder conseguir complicados procesos capaces de ser entendidos por la máquina, a continuación redactaremos algunos de los procesos de agrupación de órdenes más usados:

- **Metacaracteres:** La utilización de estos metacaracteres o caracteres comodines dan al usuario la posibilidad de dirigirse a más de un archivo. Mediante el carácter "*" podemos referirnos a todos los ficheros que tengan algún tipo de coincidencia en sus caracteres. El metacaracter "?" es utilizado para sustituir a un único carácter, que puede ser cualquier valor. El último metacaracter es [ ], este metacaracter sustituye cualquier valor incluido entre los corchetes, por ejemplo si en un mismo directorio se encuentran dos o más ficheros que difieren en un único carácter basta con escribir los caracteres comunes y meter entre los corchetes los no comunes para poder actuar sobre los dos ficheros a la vez.
- **Agrupación de órdenes:** Para que dos o más órdenes se ejecuten sucesivamente basta con escribirlas, siempre y cuando estén separadas por el carácter ";"
- **Función AND:** La sintaxis de la función "and" es "orden1 && orden2", esta sintaxis nos viene ha decir que la orden 2 se ejecutará sólo si la orden 1 ha sido ejecutada con éxito.
- **Función OR:** La sintaxis de esta función es "orden1 | | orden2", esta función traduce a la máquina la expresión " sólo se ejecutará la orden 2 si la orden 1 no ha sido ejecutada con éxito.
- **Scripts:** Los scripts son ficheros que contienen una serie de comandos para su ejecución, cada comando es leído y ejecutado, uno detrás de otro, por el intérprete de comandos. Los scripts deben ser creados por algún editor de textos de Linux. Un usuario puede usar los scripts creados por otro usuario, siempre y cuando los permisos del scripts le dejen acceder a él, al igual, un script creado por el usuario

puede ser usado por otros usuarios, siempre y cuando el autor autorice a éstos a la ejecución del archivo. La sintaxis de los scripts es muy sencilla, ya que cada orden está escrita en una línea, además los scripts de Linux permiten que algunas líneas escritas por el creador del script sean ignoradas por el intérprete de comandos, mediante la sencilla tarea de comenzar la línea con el carácter "#", esta propiedad hace que el autor pueda escribir comentarios acerca de cómo se creo el script o a que se refiere exactamente alguna de las líneas. Una vez realizado el script para que pueda ser ejecutado por el intérprete de comandos hay que dar al fichero permiso de ejecución y colocarlo en un directorio que se encuentre dentro del PATH, de esta manera el intérprete de comandos podrá encontrar y ejecutar el script sin ningún problema. Dentro de un script podemos usar unos programas llamados bucles, mediante sentencias de repetición, para lo cual es necesario evaluar y comparar los archivos u ordenes que se van a ejecutar y a partir de ahí utilizar las diferentes sentencias que el intérprete de comandos reconoce, estas sentencias son las siguientes:

- Sentecia for: Repite la operación marcada tantas veces como se le ha sido impuesta, ya sea por una variable o por una seie de números. Por ejemplo:

for contador in 1 2 3

do

hecho hola $c

done

En este bucle el ordenador entiende que debe saludarnos tres veces.

- Sentencia while: Esta sentencia deberá ejecutarse mientras que suceda algo que hemos especificado, cuando

esto deje de cumplirse el bucle se dará por finalizado.

- Sentencia until: Esta sentencia ejecutará las órdenes hasta que se cumpla una condición dada, cuando deje de ser cierto el bucle finalizará.
- Sentencia if: Esta sentencia es utilizada para traducir a la máquina la expresión "si algo se cumple haz esto, pero si ese algo no se cumple haz esto otro".
- Sentencia case: Se utiliza para ejecutar distintas sentencias en función de los valores que coincidan con la variable especificada. Además de todo esto un script puede ser ejecutado cuando se desee aunque no se encuentre nadie para dar la orden de ejecución mediante las órdenes **at**(te permite fijar el día y la hora de lanzamiento del script), **batch**(ejecuta el script cuando el nivel de carga del equipo es bajo evitando así que el sistema se bloquee por una sobrecarga de procesos a realizar), **nohup** (es necesario que sea ejecutada por el usuario pero una vez ejecutada se seguirá ejecutando aunque el usuario salga del sistema), **nice** (este comando permite al usuario dar prioridad a la ejecución de sus scripts),

Por lo tanto mediante los scripts podemos realizar cualquier operación que queramos hacer con nuestra sistema. Con la explicación de estos scripts acabamos el apartado de los comandos puesto que ya sabemos hacer todo lo que nos propondramos en nuestro intérprete de comandos.

2.3.SEGURIDAD LINUX

Como ya hemos mencionado anteriormente Linux es un sistema multiusuario y por lo tanto un puede darse el caso de que algún usuario no quiera que algunos de sus ficheros puedan ser ejecutados o examinados por los demás usuarios, aquí es donde aparece el término de permiso sobre los ficheros, los permisos sobre los ficheros son diferentes según sea el usuario que intente usar ese fichero. Cada usuario debe tener su propia cuenta para que el sistema pueda reconocerle, en esta cuenta debe aparecer el nombre del usuario, su identificación, su identificación de grupo, su contraseña, su verdadero nombre, su directorio personal(/home) y su intérprete de comandos. Cada usuario al ser registrado pertenece, por defecto, al menos a un grupo y mediante al acceso del administrador del sistema puede tener acceso a más de uno. Los permisos sobre los ficheros pueden ser fijados para tres clases de usuarios: el propietario del fichero, el grupo al que pertenece el propietario del fichero, y el resto de usuarios que están excluidos del grupo. Así vez existen tres tipos de permisos: lectura, escritura y ejecución.

El permiso de lectura (**r**) permite al usuario leer el contenido del fichero mediante la orden " more " o del directorio mediante " ls ". El permiso de lectura en ningún caso puede utilizarse para escribir o ejecutar el archivo.

El permiso de escritura (**w**) permite a un usuario leer, escribir y modificar el texto, incluso este permiso en un directorio permite crea y borrar ficheros que ya existían antes de su entrada en el directorio.

El permiso de ejecución (**x**) permite a un usuario ejecutar un fichero, siempre que éste sea un fichero ejecutable. En el caso de que este permiso afecte a un directorio implicaría que el usuario tendría acceso a él mediante el comando " cd".

Para ver los permisos de ficheros usaremos el comando ls−l y el nombre del fichero o directorio del cual queremos ver nuestros derechos sobre él, al teclear el comando nos aparecerá en pantalla cinco campos distintos, en el que el primer campo se refiere a los permisos que el propietario del fichero o directorio ha dado a los demás usuarios del sistema, el segundo campo muestra el número de enlaces que hay apuntando hacia el fichero, el tercer campo muestra quien es el propietario del fichero o directorio, el cuarto nos da la información sobre el grupo al que pertenece el fichero o directorio y el último campo nos muestra información sobre el nombre y la fecha de creación del fichero o directorio.

Ahora veremos con más detenimiento el campo sobre el permiso de ficheros, el campo del permiso de ficheros está compuesto por diez caracteres, en el que el primero de ellos nos muestra si es un fichero o es un

directorio, mediante los caracteres "−" y "d" respectivamente, los tres siguientes nos muestran los permisos que el propietario del fichero tiene sobre él. Los tres siguientes nos da la información de permisos que el grupo al que pertenece el fichero tiene sobre él, y los tres últimos representan los permisos para cualquier otro usuario del sistema, a continuación veremos un ejemplo para aclararnos.

Al teclear *ls−l facturas* nos aparecerá en la línea de debajo lo siguiente:

−rwxrw−r−− 1 Pedro contabilidad 505 Mar 13 19:05 facturas

En este ejemplo estamos viendo los permisos de ejecución del fichero facturas, ya

que el primer carácter es " − " luego observamos los permisos de cada tipo de usuario sobre el fichero, en el que nos dice que el propietario puede leer, sobreescribir y ejecutar el archivo, además nos dice que los usuarios que pertenezcan al grupo contabilidad, grupo al que pertenece el fichero, tienen permiso de lectura y escritura sobre el archivo y por último los demás usuarios tan sólo tienen permiso de lectura. Además de todo esto en el ejemplo nos dice que el fichero se llama facturas, que fue creado el o modificado el 13 de marzo a las 19:05 horas por el usuario Pedro, perteneciente al grupo de contabilidad, además nos dice que este fichero tiene un enlace con cualquier otro fichero, que para saberlo habría que buscar un fichero con su mismo número de inodo. Pero no sólo con esto sabremos los permisos que tenemos sobre el fichero, ya que si el directorio no nos da alguno de los permisos otorgados por el fichero, careceremos de ese permiso sobre el fichero.

Pero Linux no sólo ofrece estas posibilidades sobre el uso de los permisos de ficheros, además el propietario del fichero tiene derecho a cambiar los permisos hacia cada usuario cuando lo crea conveniente mediante el comando " chmod ", el comando chmod, tiene la siguiente sintaxis:

chmod {a, u, g, o} {+, −} {r, w, x} <nombredelfichero>

El primer campo nos indica si el cambio de órdenes afecta a todos los usuarios del sistema (a), al propietario del fichero (u), al grupo propietario del fichero (g) o al resto de los usuarios del sistema (o). El segundo campo nos indica si otoramos permisos (+) o se los quitamos (−) a los usuarios referidos. El tercer campo se refiere a los permisos que otorgamos o quitamos a los distintos usuarios del sistema y por el último el cuarto campo indica el fichero sobre el cual vamos a cambiar los permisos de fichero. Por ejemplo la orden " *chmod og−r facturas* " nos dice que todos los usuarios del sistema, excepto el propietario son excluidos de su permiso de lectura sobre el fichero facturas.

Además de este sistema de seguridad ante el resto de usuarios, Linux nos ofrece la posibilidad de proteger nuestros ficheros ante la posibilidad de un olvido al guardarlo, o borrarlo por error. Estas características se consiguen mediante los enlaces, hay dos tipos de enlaces:

- Enlaces duros: La orden usada para crear enlaces duros es " ln ", esta orden viene acompañada por los dos ficheros a enlazar. Cada fichero está determinado por su número de inodo, el cual es utilizado para poder ser identificado por el sistema de archivos, haciendo uso de la orden " ls −i ", podremos ver el número de inodo del fichero requerido. Los enlaces duros enlazan dos ficheros directamente por el inodo, haciendo que el sistema los trate como el mismo fichero, por lo tanto al modificar uno de los ficheros enlazados, el otro también modificará su contenido, en cambio Linux ofrece la posibilidad de que cuando borremos uno de los archivos enlazados con enlace duro, el otro fichero enlazado no quede borrado. La única restricción de estos enlaces es que los dos ficheros enlazados han de pertenecer al mismo sistema de ficheros, pero esta restricción se acaba con la utilización de enlaces simbólicos.
- Enlaces simbólicos: Los enlaces simbólicos se crean mediante la orden " ln −s ", seguida de los nombres de los ficheros a enlazar. Un enlace simbólico permite dar a un fichero el nombre de otro, pero no enlaza el fichero con un inodo, los dos ficheros enlazados tienen un número de inodo diferente. Cuando creamos un

enlace simbólico creamos un fichero que depende totalmente del fichero al que está apuntando, los cambios realizados en el fichero apuntado son producidos también en el apuntador, pero no viceversa, por otra parte los permisos sobre el fichero del enlace simbólico son siempre los mismos que los del fichero apuntado por el enlace. Para poder ver cual de los archivos es el apuntado y cual el apuntador usaremos la orden ls −l.

Estas ventajas sobre la seguridad del entorno del usuario sobre Linux son reforzadas por la existencia de una contraseña personal que es requerida cada vez que un usuario desee entrar en su cuenta, además Linux ofrece la posibilidad de poder cambiar la contraseña, en el caso de que ésta hubiese sido encontrada y malutilizada por otro usuario, para lo cual le pide la anterior contraseña, para así poder evitar que cualquier otro usuario pueda tener control sobre la contraseña. En conclusión podemos decir que Linux, tapa toda duda sobre el que no le guste trabajar en entorno multiusuario, haciendo que de este entorno sólo queden ventajas y desaparezcan las desventajas.

2.4 HERRAMIENTAS DE COMPRESIÓN

La instalación de nuevos y novedosos programas de software para Linux, servirán para que nuestro sistema operativo no se quede viejo y obsoleto. La gran mayoría de estos productos vendrán empaquetados o comprimidos, para lo cual veremos las tres principales versiones de Linux para empaquetar, comprimir, desempaquetar y descomprimir programas:

- El empaquetador **tar:** Utilizando este comando conseguiremos guardar varios ficheros o directorios bajo el nombre de un único fichero, que será ejecutable. La principal ventaja que este empaquetador nos ofrece es que al desempaquetar el fichero, permanecerá la estructura de directorio. Con este comando se pueden empaquetar ficheros y directorios, además de hacer copias de seguridad. La desventaja de este empaquetador, es la cantidad de espacio de disco duro que ocupa.
- El programa de compresión **gzip**: El programa gzip reduce el tamaño de los ficheros dados mediante el algoritmo de compresión de Lempel−Ziv (LZ77). Mediante este proceso el fichero que es comprimido se reemplaza por otro de extensión .gz, que mantiene los mismos permisos de fichero, pero ocupa un espacio menor. Existen nueve niveles de compresión gzip, etos nueve niveles pueden ser elegidos por nosotros para cuando se vaya a hacer una compresión, el único inconveniente que los niveles de compresión implican, es que a mayor nivel, mayor tiempo de ejecución, pero sin embargo la calidad será la misma. Además con gzip podemos manejar los archivos de compresión hechos con *compress*., el compresor antes utilizado por la mayor parte de usuarios de Linux.
- El empaquetador **rpm**: rpm es un nuevo sistema de empaquetamiento que poco a poco, dadas sus grandes prestaciones se está abriendo un hueco entre los programas de compresión de Linux. rpm posee una base de datos de los paquetes instalados y de sus archivos, lo que nos permite realizar consultas y verificaciones del sistema, además al realizar una actualización de software, el empaquetador rpm, nos mantiene los archivos de configuración, de manera que el usuario se despreocupa de volver a realizar los ajustes específicos de configuración del programa. Además rpm nos da la posibilidad de saber ocurrirá, mediante una prueba de instalación, mediante la opción "test". La enésima ventaja de rpm, es que a la hora de desinstalar un paquete ya instalado, rpm buscará todos los ficheros del paquete, aunque no se encuentren en el directorio principal del paquete, y los eliminará o modificará, según proceda. Por último la última ventaja a la que haremos referencia es a la posibilidad de verificación que rpm nos ofrece, mediante esta verificación podremos saber si un paquete instalado anteriormente ha sido modificado en sus partes más importantes de configuración y en caso de haber sido modificado, rpm nos da la posibilidad de ver cuales han sido sus modificaciones, esta prestación puede ser útil para descubrir porque nuestro sistema no funciona correctamente.

2.5. X−WINDOW

Además de todo lo visto anteriormente, Linux ofrece un entorno gráfico de escritorio, en el que aunque no se puedan realizar todas las aplicaciones ya señaladas, si facilitará las que a cada usuario le parezcan más complicadas. En este punto analizaremos el entorno de escritorio KDE, que es la versión X−Window de la

distribución Linux de Esware.

Para lanzar la aplicación KDE, basta con teclear a nuestro intérprete de comandos la orden "starx", aunque también tenemos la posibilidad de hacer que la máquina arranque directamente desde el entorno gráfico. A continuación, al igual que en el principio de la sesión Linux le pedirá que se identifique, reclamándole su nombre de usuario y su contraseña. A continuación veremos la apariencia del entorno gráfico de KDE:

AQUÍ IRÍA LA FIGURA 15.1 DE LA PAG 328.

En la anterior figura vemos la apariencia del escritorio KDE, con las opciones de instalación por defecto, en esta imagen podemos apreciar 2 zonas:

1. En la parte inferior se encuentra el panel que nos servirá para lanzar aplicaciones y cambiar entre escritorios. Por otra parte en la zona izquierda del panel encontramos un icono que nos da acceso a una multitud de menús en donde puedo lanzar más aplicaciones o configurar mi sistema. (Este icono es similar al menú inicio de Windows).

2. El resto de la superficie está ocupado por el Escritorio, que es el área de trabajo donde aparecerán las aplicaciones que se vayan ejecutando, si no se está ejecutando ninguna aplicación, en el escritorio aparecerán unos iconos que representan las aplicaciones y procedimientos más utilizados, pudiendo así lanzarlos simplemente haciendo clic, con el ratón encima de sus iconos.

Para movernos por el entorno gráfico utilizaremos, normalmente, el ratón. El ratón está formado por dos botones, el derecho, nos mostrará una lista de propiedades del icono sobre el que hemos puestos el puntero del ratón, si pulsamos el botón izquierdo ejecutamos la acción asociada al icono sobre el que tenemos el puntero. Además mediante la presión continúa sobre el botón izquierdo nos permite arrastrar iconos, pudiendo moverlos hacia otros directorios, hacia alguna aplicación o eliminarlos.

El sexto icono en el panel nos representa la aplicación kfm, que contiene nuestro directorio de ficheros personales, el directorio *home*, aunque ahora desde esta ventana lanzada por un clic en el icono también podemos ver los ficheros que contiene la unidad de cd−rom o la disquetera, esta aplicación nos permite mover ficheros por nuestros directorios, de uno en uno o un grupo de ellos mediante la selección de sus correspondientes iconos, con la tecla *CTRL+botón izquierdo del ratón.*

Al hacer clic sobre el icono de la concha lanzamos el programa konsole nos proporciona una sesión de línea de comandos idéntica a la que se realiza en modo de texto, lo que nos permite volver a la sesión sin necesidad de reiniciar el sistema.

Otra de las prestaciones de este entorno KDE es la posibilidad de tener las ventanas que el usuario crea oportuno abiertas sobre el escritorio, cada ventana tiene tres iconos en su parte superior derecha, un punto, un cuadrado y un aspa. Al hacer clic sobre el punto la ventana de la aplicación abierta se minimiza y desaparece del escritorio, para poder recuperar esta ventana habrá que buscarla en el icono "K", el cual tiene un apartado para recuperar ventanas tal y como estaban antes de ser minimizadas. Al hacer clic sobre el cuadrado la ventana se maximizará ocupando la totalidad del escritorio, al hacer clic otra vez sobre él la ventana volverá a su tamaño natural. Por último si hacemos clic sobre el aspa la aplicación quedará totalmente cerrada. Además de todo esto las ventanas pueden ser movidas y modificadas de tamaño según sean las exigencias del usuario, de esta manera el usuario podrá tener tantas ventanas visibles como quiera. Por último el entorno KDE nos ofrece la posibilidad de tener cuatro escritorios distintos para así tener una mejor distribución de las ventanas abiertas, esto lo consigue mediante la utilización de los cuatro botones situados en el panel. Además se puede tener una ventana abierta en todos los escritorios virtuales mediante la opción "pegar" del menú desplegable que se encuentra en la parte superior izquierda de cada ventana.

Como anteriormente hemos explicado para la lectura de un disquete o CD−ROM era necesario desmontarlo mediante el comando mount, sin embargo en el entorno KDE esto no es necesario ya que al ejecutar la lectura de alguno de los dispositivos, éstos son desmontados automáticamente. Además también sabemos que antes de extraer el medio de la unidad hay que desmontar el dispositivo, y aunque esto no se hace automáticamente basta con hacer clic derecho y elegir la opción "desmontar".

El entorno gráfico KDE nos da también la posibilidad de al lanzar un programa éste sea ejecutada directamente desde su aplicación, esto lo conseguimos mediante la colocación de tipos MIME, en las aplicaciones de destino, aunque esta opción suele ser procesada directamente por el sistema.

Tanto la instalación como la ejecución de internet desde nuestro entorno gráfico KDE está guiado por un asistente en entorno gráfico, lo que hace accesible a cualquier persona el poder "navegar" por internet.

La sesión de KDE se finaliza mediante un clic sobre el aspa que se encuentra en el panel. Pero hemos de saber que el cerrar KDE imprudentemente puede hacernos perder la información sobre lo que estábamos trabajando para lo cual antes de cerrar la sesión con KDE deberemos guardar toda la información trabajada.

Como hemos visto trabajar con KDE es muy similar que trabajar en un entorno Windows de Microsoft, por lo que aunque no están contadas todas las posibilidades del entorno KDE, éstas son muy fáciles de explorar y utilizar.