FUNDAMENTOS

DE SISTEMAS

GNU/LINUX

GUIA DE ESTUDIO HACIA UNA CAPACITACION SEGURA

FUNDACION

Código Libre Dominicano

Antonio Perpinan

FUNDAMENTOS
DE SISTEMAS
GNU/LINUX

GUIA DE ESTUDIO HACIA UNA CAPACITACION SEGURA

FUNDACION
Código Libre Dominicano

FUNDAMENTOS DE SISTEMA GNU/LINUX

GUIA DE AUTO ESTUDIO HACIA UNA CAPACITACION SEGURA

LOS PROPÓSITOS DEL CURSO

Los profesionales de la tecnología de la información (TI) son críticos hoy día para el ambiente de negocio. Adquirir las herramientas y conocimiento disponible en la tecnología de hoy es vital. GNU/Linux y el Código Libre y Abierto han colocado un nuevo estándar en lo que es desarrollo e implementación de aplicaciones nuevas y personalizables. GNU/Linux continúa ganando espacio de reconocimiento entre los profesionales y administradores del TI debido a su flexibilidad, estabilidad, y su poderosa funcionalidad. A medida que más empresas utilizan GNU/Linux, crece la necesidad de soporte y planificación sobre la integración de GNU/Linux en infraestructuras nuevas y/o existentes. El rol del administrador es guiar la implementación y desarrollo de soluciones basadas en GNU/Linux. Su éxito o derrota dependerán de su conocimiento y experiencia de esta fantástica arquitectura.

Este curso es un repaso comprensivo de las características y funcionalidad de GNU/Linux, orientada a preparar al estudiante con las herramientas necesaria para la certificación. Explicación detallada se provee de los conceptos claves, muchos conceptos y utilidades de GNU/Linux son idénticos sin importar la distribución específica siendo utilizada. Algunas características están disponibles en algunas distribuciones, y otras son añadidas durante la instalación. La naturaleza de GNU/Linux y el Software Open Source, es tal, que cambios al fuente y cambio a funcionalidad de cualquier componente debe ser incluido en la distribución específica . Los conceptos sublimes de las capacidades de GNU/Linux se mantienen consistentes a través de cada distribución, kernel y cambio de Software.

Estos libros han sido desarrollados de acuerdo con los estándares de la industria de la certificación de GNU/Linux. Los objetivos de la certificación GNU han sido elementos claves en el desarrollo de este material. La secuencia de los exámenes de certificación GNU/Linux provee la gama más amplia de los conceptos necesarios para dominar GNU/Linux. Los objetivos de las certificaciones LPI y RHCE también son incluidos. El CD interactivo y la página Web con el curso contiene videos digitales y pequeñas prácticas de selección múltiple igual a los del exámen. En el libro LA GUIA DEL ESTUDIANTE se provee una guía específica para la preparación de la certificación.

Este libro provee los conceptos y principios fundamentales necesarios para administrar un sistema GNU/Linux. Los conceptos y las tareas de administración pueden ser un poco amplios. Se le dará una explicación del rol del administrador, estructura y función detallada del kernel, y cubriremos tópicos administrativos claves del manejo de paquetes, procesos, espacio de disco, Backups y los usuarios así como las tareas programáticas, y los Logs/Registros del sistema. Este conjunto de herramientas te permitirán apropiadamente administrar un sistema GNU/Linux sea este de unos cuantos hasta miles de usuarios. Estos capítulos también te proveerán la información que necesitas para Certificarte.

Fundamentos de GNU/Linux proporciona una introducción a profundidad de los conceptos y de los principios que son necesarios para instalar un sistema GNU/Linux y desenvolverse en los ambientes de ventana del X y de la línea de comandos. Este manual da la dirección paso a paso para las distribuciones importantes de GNU/Linux y su instalación, incluyendo RedHat, Debian, Mandrake y Slackware. Se enfatizan los conceptos de instalación, las utilidades, y la funcionalidad de GNU/Linux común a todas las distribuciones y estas se explican en detalle adicional. Un principiante o un experto pueden aprender o repasar los conceptos de particionar discos y localizar los archivos de configuración, usando el shell y las consolas, crear los scripts, y editar archivos de texto que permanecen dominantes, sin importar la nuevas herramientas gráficas, para los ajustes de configuración. Este conjunto de tópicos permitirá que usted instale y configure correcta-

mente un sistema GNU/Linux. En estos capítulos también se le provee la información necesaria para certificar sus habilidades en GNU/Linux

METAS DEL CURSO

Este curso le proveerá la información necesaria para completar los siguientes tópicos:

- Describir los componentes estructurales y distinguir entre una distribución de GNU/Linux y otra.
- Describir software de fuente abierta (Software Open Source) y diferenciar entre GNU/GPL.
- Crear los disquetes de arranque de instalación.
- Instalar las principales distribuciones de GNU/Linux: RedHat (RPM), Debian (DPKG) y Slackware (tar.gz).
- Utilizar los ambientes de escritorio KDE y GNOME.
- Instalar y configurar XFree86.
- Localizar y utilizar la ayuda en línea.
- Configurar el hardware del sistema.
- El uso de fdisk o el cfdisk para crear, corregir, y suprimir particiones del disco.
- Utilizar el LILO/GRUB para manejar opciones para cargar el sistema.
- Arrancar el sistema, cambiar los runlevels, y cerrar o re-iniciar el sistema.
- Utilizar los disquetes de rescate para iniciar un sistema que se ha dañado.
- Describir el sistema de archivos jerárquico de GNU/Linux y el papel de los directorios y archivos claves en la organización del sistema. Trabajar con eficacia en la línea de comando de Linux usando comandos comunes del shell, streams, tuberías, filtros, y cambio de dirección.
- Usar scripts del shell para realizar tareas repetitivas rápidamente.
- Abrir, corregir, y almacenar documentos de texto usando el editor 'vi'.
- Manejar los sistemas de impresión locales.
- Describir algunas aplicaciones comunes disponibles al usuario para sus tareas, tales como: navegar en Internet y acceso a E-mail, procesamiento de textos, presentaciones, hojas de cálculo, y manejo de gráficos.

EJERCICIOS

Los ejercicios en este manual son diseñados para dar practicas reales en los ambientes de redes y aislados (stand-alone o networking) al usuario. Es altamente recomendado que usted complete todos los ejercicios en cada capítulo antes de continuar al próximo. Entendemos que en raros casos tal vez esto no sea conveniente cuando estudia fuera del taller. Si por alguna razón no puedes completar un ejercicio por circunstancias ajenas, debes planificar completarlo tan pronto sea posible.

Existirán ejercicios que no podrás completar por el limitante de equipo ó software. No permita que esto le impida completar los otros ejercicios que resten en el capítulo ó modulo.

TOME NOTA

Los ejercicios en este libro fueron diseñados para ser ejecutados en un equipo de prueba y nunca deben ser llevados a cabo en uno trabajando y donde se ejecuten aplicaciones importantes. Instalar GNU/Linux, reparticionar para instalar GNU/Linux, o practicando los ejercicios en una LAN u ordenador de trabajo puede causar problemas de configuración, lo cual puede conllevar a perdidas irreparable de data y dispositivos periféricos. Por favor siempre recuerde esta advertencia. Es preferible que dediques una estación de trabajo para practicar estos ejercicios. Instalar GNU/Linux en una situación dual-boot es una alternativa razonable, pero aún así conlleva ciertos riesgos.

WEB y CD

Una parte muy clave de esta serie de auto-aprendizaje es el portal de soporte. Las lecciones que le indiquen visitar la página web o el CD-ROM que le acompaña, a menudo, es para ayuda con los conceptos que son mejor entendidos después de una descripción visual. Los segmentos video Digital proporcionan una ilustración gráfica acompañada por una narración de los instructores. Estas lecciones son ideales, ambos como introducciones para afinar conceptos y para ayudar el refuerzo.

RECUERDE

Como herramienta de soporte les ofrecemos el CD interactivo, incluido en este libro, y nuestra página web http://www.abiertos.org y allí accesar hacia la sección Linux-Certificación, estos contienen exámenes de prueba de las diferentes certificaciones. Recreamos el escenario de las preguntas de selección múltiples, multi-selección y falso verdadero. Es muy importante que tome muchas horas de practicas antes de intentar pasar el exámen de certificación que le corresponda ya sea LPI ó RHCE.

CONTENIIDO

CAPITULO 1

ADMINISTRACIÓN DEL SISTEMA OPERATIVO GNU/LINUX

¿QUÉ ES GNU/LINUX?

OBJETIVOS

Al completar este capítulo, usted podrá:

- Describir la estructura de los componentes de GNU/Linux.
- Describir la genealogía de Linux y UNiX
- La Historia de UNiX y Linux
- Software de Fuente Abierta y Free Software (Open y Libre)
- La Licencia GPL (General Public License) del GNU
- Describir que es una Distribución GNU/Linux
- Detallar los puntos Fuertes y Débiles de GNU/Linux
- Describir Multitasking (Multi-Tarea)
- Contrastar Línea de comandos y las Interfaces Graficas de Usuarios (GUIs)
- Detallar como buscar ayuda en línea.

Preguntas Pre-Exámen

1. ¿Qué es una distribución?

2. ¿Liste 4 distribuciones populares de GNU/Linux?

3. ¿Cómo es GNU/Linux Distribuido?

4. ¿Cómo es GNU/Linux Licenciado?

5. ¿Qué es Software Open Source?

6. ¿Qué es Software Libre?

INTRODUCCION

En éste capítulo analizamos el background del Sistema de GNU/Linux: Como ha evolucionado, su estructura, el ambiente en el cual se desarrolla, y contrastaremos sus ventajas y desventajas. Aunque en el manual nos concentraremos en Slackware, Debian y RedHat siempre compararemos con otras distribuciones muy utilizadas en el Mercado de hoy día como lo son: SuSE, Mandrake, Turbo Linux, etc...

GNU/LINUX, SISTEMA OPERATIVO DE REDES

PGNU/Linux es un sistema operativo Libre (free), de fuente abierta (Open Source), Parecido-a-UNiX (UNiX-Like), interactivo, multiusuario, multítarea, de redes (network).

Demos un vistazo a esta descripción parte por parte:

LIBRE (FREE)

El Software Libre proporciona la libertad de:

1.- Ejecutar el programa, para cualquier propósito;
2.- Estudiar el funcionamiento del programa, y adaptarlo a sus necesidades;
3.- Redistribuir copias;
4.- Mejorar el programa y poner sus mejoras a disposición del público, para beneficio de toda la comunidad.

Como consecuencia de estas 4 libertades, el Software Libre ofrece la libertad de aprender, libertad de enseñar, libertad de competir, libertad de expresión y libertad de elección.

OpenSource	El término “Open Source” se refiere a tener acceso al código fuente. Pero el acceso al código fuente es apenas un pre-requisito para dos de las cuatro libertades que definen al Software Libre. Muchas personas no entienden que el acceso al código fuente no es suficiente. “Software Libre” evita caer en esa confusión.
UNiX-Like	UNiX es un sistema operativo desarrollado por Bell Labs de AT&T en el 1969. Aunque el termino UNiX se utiliza liberalmente al discutir éstos sistemas operativos, y GNU/Linux es uno de ellos, no todos los sistemas operativos parecido a unix son considerados UNiXLike. UNiX es una marca registrada del Open Group, y sólo los sistemas operativos que pasan completamente su prueba pueden ser etiquetados y certificados UNiX (Solaris de Sun Microsystems, es UNiX). Linux es UNiX-Like en su funcionamiento y en su estructura, pero no contiene código del AT&T UNiX.
Network	Una red es un conjunto de ordenadores, conectados entre sí, que pueden comunicarse compartiendo datos y recursos. Linux se hizo con para trabajar en redes desde sus inicios, y todas las distribuciones incluyen los programas y utilidades necesarias para que el computador pueda ser incluida en una red y funcionar adecuadamente.
Sist. Operativo	Un Sistema Operativo (SO) es el conjunto de programas básicos y utilidades que hacen que una máquina funcione y resulte útil a los usuarios. El sistema operativo comienza a trabajar cuando encendemos el computador, y administra los recursos de hardware de la máquina en los niveles más básicos.
Interactivo	GNU/Linux permite que los usuarios interactúen con el equipo, digitando comandos que se ejecutan inmediatamente (por ejemplo no es así en los main frames donde los comando se almacén por el sistema operativo para luego ser ejecutados en grupos…batch).
Multiusuario	GNU/Linux es un Sistema Operativo multiusuario que permite a más de un usuario accesar una computadora. Claro que, para llevarse esto a cabo, el Sistema Operativo también debe ser capaz de efectuar multitareas. Debe diferenciar entre los diferentes procesos y los diferentes usuarios.
Multitarea	Linux es capaz de manejar más de una tarea a la vez.

Para más información, refiérase a:
1. http://www.gnu.org
2. http://www.linux.org,
3. http://www.fsf.org
4. http://www.opensource.org

Los siguientes tópicos son discutidos en esta sección:
• Estructura de GNU/Linux
• ¿Quien usa GNU/Linux?
• Ambiente de Sistemas
• Linux y la Genealogía UNiX
• La Naturaleza del Desarrollo de Software OpenSource y Libre
• La Rentabilidad de negociar en el Free Software
• Importante Implementaciones de GNU/Linux
• GNU/Linux sus Ventajas
• GNU/Linux sus Desventajas
• Múltiple Usuarios y Multitareas

Estructura de GNU/Linux

GNU/Linux esta estructurado con un kernel pequeño y un número de programas utilitarios montados encima del kernel. El núcleo maneja los recursos de la computadora, tal como el procesador y la memoria, y en esto debe asegurarse de que cada quien que trata de utilizar éstos recursos es dado una oportunidad apropiada de tiempo de acceso.

El kernel se carga en memoria cuando Linux se inicia y permanece en memoria hasta que el sistema se descarga por completo. Se diseña para ser lo más pequeño que sea posible, permitiendo así que la memoria restante sea compartida entre todos los programas que se ejecutan en el sistema.

Los programas utilitarios proporcionan manejo de archivo, supervisión del sistema, desarrollo de aplicaciones, manejo de usuario, y comunicación de red. Puede haber más de 2.000 utilidades en un sistema de GNU/Linux.

La filosofía de GNU/Linux, como la de la mayoría del UNiX y los sistemas operativos UNiX-Like, ha sido mantener el kernel lo más pequeño posible, moviendo todas las actividades que no tienen que ser realizados absolutamente por el kernel en programas utilitarios a nivel del usuario. El ejemplo más obvio de esto es el intérprete interactivo de comando. Debajo de Linux, éste intérprete de comando, mejor conocido como el shell, es un programa normal ejecutado cuando un usuario entra que sirve como base para que el usuario pueda ejecutar comandos. El shell no es parte del kernel.

¿Quien Usa GNU/Linux?

GNU/Linux se utiliza en una amplia gama de instituciones y de organizaciones. Cada día más y más paises y compañías se alínean al uso y filosofía del SoftWare Libre.

• Los Proveedores de Internet (ISPs) lo utilizan para los servidores de red, tales como servidores WEB.
• Las universidades y los centros de investigación lo utilizan para las matemáticas que procesan, desarrollo de aplicaciones, y Correo Electrónico.
• Las grandes organizaciones comerciales, como los bancos, lo utilizan para sus servidores de base de datos.
• Las industrias de servicio, tales como hoteles y líneas aéreas, lo utilizan para las reservaciones.
• Muchas industrias lo emplean para usarla en estaciones de trabajos gráficas.

- GNU/Linux se utiliza en sistemas médicos, scanners y sistemas de imagen.
- También se utiliza en la fabricación, la tecnología, CAD/CAM, investigación y desarrollo aplicaciones.
- Puede ser utilizado en sistemas de energía y grande simulaciones de sistemas.
- Puede ser utilizado en el gobierno y las ramas militares, simuladoras de aviones y aeroespacio, y predicción del tiempo.

Ambientes de Sistema

GNU/Linux se puede utilizar en una amplia gama de ambientes relacionados con la informática. Este sistema operativo fué diseñado para ser robusto, estable y escalable.

- GNU/Linux se puede emplear como estación de trabajo de escritorio de mono-usuario.
- Las máquinas de GNU/LinuxLinux pueden tener acceso y servir recursos a otros sistemas operativos.
- GNU/Linux se presta para la construcción de sistemas distribuidos grande clusters.

La disponibilidad de GNU/Linux para una amplia gama de máquinas y su gran capacidad de red, hace a menudo pues que GNU/Linux sea utilizado en parte de grande redes heterogéneas (redes utilizando una mezcla de diversos sistemas operativos). Con GNU/Linux, se puede proporcionar acceso a una variedad grande de sistemas operativos, incluyendo UNiX y sistemas UNiX-Like, Novell, Macintosh, OS/2 y otros sistemas operativos todo utilizando sus métodos nativos de comunicación.

Sistemas distribuidos o clusters de gran capacidad pueden también ser construido utilizando Linux, esta siendo una área importantísima en éstos momentos. Los sistemas clusters o Super Computadores de Linux están demostrando ser considerablemente más rentable que sus gigantes competidores para las tareas grandes, tales como simulaciones de medio ambiente.

Linux y la Genealogía UNiX

La genealogía del sistema operativo UNiX y todos sus derivados (incluyendo Linux) es complicada y no se puede representar en un sólo diagrama. Este gráfico demuestra una vista simplificada de los principales variantes importantes en el desarrollo de Linux con respecto a UNiX y algo de sus variantes. El diagrama ilustra, que Linux es relativamente nuevo en el mundo de UNiX.

La Historia de Linux

En 1991, Linus Benedict Torvalds, estudiante de la Universidad Helsinki, estrenó la primera versión pública de su sistema operativo Linux la 0.02. Desde entonces, millones de usuarios de todo el mundo poseen éste sistema gratuito y miles de ellos contribuyen a su contínuo desarrollo aportando ideas, programas, información sobre fallos del sistema ya sea en hardware/software (bugs), ayuda, tutoriales, etc.

Linux nació de la idea de crear un sistema clon de UNiX basado en GNU (General Public License, Licencia General Pública) y el código fuente disponible gratuitamente. Esta idea nació en 1991 cuando Linus Torvalds estudiaba la carrera de Ciencias Informáticas. Torvalds se encontraba especialmente interesado en Minix, el único sistema UNiX disponible en aquél entonces de fácil acceso para los estudiantes y profesores. Este sistema gratuito fué creado por Andrew Tanenbaum con el porpósito de facilitar a los alumnos de la universidad el estudio y diseño de sistemas operativos. Minix era un UNiX más, tanto en apariencia como en el kernel (núcleo del sistema operativo), pero distaba mucho de ser comparable a uno de los grandes. Es a partir de aquel momento que Torvalds decidió crear un sistema que excediera los éstandares de Minix, poniendo en marcha el proyecto personal Linux.

Torvalds tomó sus primeras clases de C y UNiX en 1990 y en poco tiempo empezó a utilizar el sistema

operativo Minix en su nuevo 386. Linux evolucionó desde el simple programa “Hola, Mundo” a una terminal. Durante mucho tiempo Torvalds trabajó en la soledad de sus ideas, hasta la mañana del 3 de julio de 1991 cuando pidió ayuda a través del Internet. Al principio fueron unos pocos los que le apoyaron, pero al poco tiempo muchos otros cibernautas se unieron al proyecto. En uno de los primeros emails enviados por Torvalds a la comunidad del ciberespacio respecto a Linux, informaba sobre su proyecto como si fuera un hobby, nada tan grande ni comparable con GNU.

Durante el desarrollo Torvalds se encontró con muchos problemas a lo largo de la programación del kernel. Pero Linux empezó a disponer de controladores para los dispositivos internos de la PC y un funcionamiento correcto del disco aproximadamente el 3 de julio, unas horas después de enviar su primer email informado sobre su proyecto. Dos meses más tarde Linux empezaba a funcionar y el código fuente de la primera versión 0.01 ya estaba disponible. La version 0.01 incluía un bash shell 1.08 y el compilador gcc 1.40.

Muy pronto Linux se convirtió en un sistema mucho más fácil de instalar y configurar, y empezó a coger fama en todo el mundo. Al tener en muy poco tiempo miles de usaurios, las nuevas versiones de Linux salían casi semanalmente. En el presente hay millones de usuarios y gracias a ellos y a sus aportes, Linux crece sin respito alguno. La última version del kernel estable es Linux 2.2.13 del 20 de Octubre de 1999 y la próxima versión en desarrollo y lanzado en fase beta es la 2.3.30 Release 6.

Como todos los sistemas operativos, Linux también dispone de un logotipo. Torvalds decidió que la imagen que representaría a Linux sería la de un pingüino, de nombre TUX. En casi todas las páginas web relacionadas con Linux se puede hallar el logotipo. En la imagen que mostramos pintamos el pecho del logotipo con los colores de nuestra bandera, símbolo que identificará éste site de ahora en adelante.

Linux había nacido para ser un sistema operativo del tipo POSIX (sistema variante de UNiX), totalmente gratuito para el usuario y con libre acceso al código fuente. Estas tres ideas fueron las que lo han convertido en el sistema con mejor rendimiento, más fiable, veloz y con más desarrolladores del mundo. Pronto se ha colocado cerca de los grandes sistemas operativos como UNiX en el ámbito de servidores de comunicaciones, especialmente utilizado en empresas proveedoreas de acceso a Internet.

Las versiones más recientes de Linux ofrecen la posiblidad de convertir nuestro ordenador personal en una potente estación de trabajo. Puede funcionar como estación de trabajo personal dándoneos la posiblidad de acceder a las prestaciones que ofrece UNiX y cualquier otro sistema operativo. Además, gracias al aporte de muchas empresas hoy en día cuenta con potentes entornos gráficos que ayudan significativamente a elegir Linux. Puede adempas configurar para funcionar como estación de desarrollo y/o aprendizaje, proveer acceso a Intranets e Internet y muchas otras opciones.

GNU/Linux como estación de desarrollo y/o aprendizaje es uno de los mejores sistemas ya que dispone de muchos lenguajes de programación gratuítos como: GNU C, GNU C++, GNU Fortran 77, ADA, Pascal, TCL/Tk, etc. y muy pronto tal véz las versiones conocidas de Delphi para Linux de Borland Inc. las cuales esperemos que también sean de facíl acceso por los usuarios o en todo caso a un costo razonable que permita contar con esta valiosa herramienta de programación. La mayoría de éstos lenguajes vienen con extensas librerías de código fuente.

GNU/Linux como sistema operativo gratuíto posee características que le hacen único. Las más importantes son: multitarea, memoría virtual, los drivers (controladores de dispositivos) TCP/IP más rápidos del mundo, librerías compartidas, multiusuario, modo de funcionamiento protegido (al contrario de otros Sistema Operativos) y la más fundamental soporta multitarea de 32 y 64 bits.

Posee además capacidades avanzadas para la interconexión de redes de PC´s ya que para desarrollar

Linux hubo que utilizar Internet. El desarrollo del software y las características de interconexión de redes se empezaron a desarrollar desde las primeras versiones de GNU/Linux y desde entonces ha ido evolucionando a gran velocidad y más aún con la gran aceptación de la red; en especial de Internet.

Y para concluir:

Hoy en día GNU/Linux es utilizado por millones de usuarios y miles de empresas. No hay duda pués que Linux es uno de los sitemas operativos con más posiblidades y es el único que se actualiza día a día.

El Movimiento del Software Libre

Mantenemos esta definición de software libre para mostrar claramente qué debe cumplir un programa de software concreto para que se le considere software libre. El “Software Libre” es un asunto de libertad, no de precio. Para entender el concepto, debes pensar en “libre” como en “libertad de expresión", no como en “barra libre” [En inglés una misma palabra (free) significa tanto libre como gratis, lo que ha dado lugar a cierta confusión].

“Software Libre” se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. De modo más preciso, se refiere a cuatro libertades de los usuarios del software:

- La libertad de usar el programa, con cualquier propósito (libertad 0).
- La libertad de estudiar cómo funciona el programa, y adaptarlo a tus necesidades (libertad 1). El acceso al código fuente es una condición previa para esto.
- La libertad de distribuir copias, con lo que puedes ayudar a tu vecino (libertad 2).
- La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad

se beneficie. (libertad 3). El acceso al código fuente es un requisito previo para esto.

Un programa es software libre si los usuarios tienen todas estas libertades. Así pues, deberías tener la libertad de distribuir copias, sea con o sin modificaciones, sea gratis o cobrando una cantidad por la distribución, a cualquiera y a cualquier lugar. El ser libre de hacer esto significa (entre otras cosas) que no tienes que pedir o pagar permisos.

También deberías tener la libertad de hacer modificaciones y utilizarlas de manera privada en tu trabajo u ocio, sin ni siquiera tener que anunciar que dichas modificaciones existen. Si publicas tus cambios, no tienes por qué avisar a nadie en particular, ni de ninguna manera en particular.

La libertad para usar un programa significa la libertad para cualquier persona u organización de usarlo en cualquier tipo de sistema informático, para cualquier clase de trabajo, y sin tener obligación de comunicárselo al desarrollador o a alguna otra entidad específica.

La libertad de distribuir copias debe incluir tanto las formas binarias o ejecutables del programa como su código fuente, sean versiones modificadas o sin modificar (distribuir programas de modo ejecutable es necesario para que los sistemas operativos libres sean fáciles de instalar). Está bien si no hay manera de producir un binario o ejecutable de un programa concreto (ya que algunos lenguajes no tienen esta capacidad), pero debes tener la libertad de distribuir éstos formatos si encontraras o desarrollaras la manera de crearlos.

Para que las libertades de hacer modificaciones y de publicar versiones mejoradas tengan sentido, debes tener acceso al código fuente del programa. Por lo tanto, la posibilidad de acceder al código fuente es una condición necesaria para el software libre.

Para que estas libertades sean reales, deben ser irrevocables mientras no hagas nada incorrecto; si el de

arrollador del software tiene el poder de revocar la licencia aunque no le hayas dado motivos, el software no es libre.

Son aceptables, sin embargo, ciertos tipos de reglas sobre la manera de distribuir software libre, mientras no entren en conflicto con las libertades centrales. Por ejemplo, copyleft [``izquierdo de copia"] (expresado muy simplemente) es la regla que implica que, cuando se redistribuya el programa, no se pueden agregar re tricciones para denegar a otras personas las libertades centrales. Esta regla no entra en conflicto con las libertades centrales, sino que más bien las protege.

Así pues, quizás hayas pagado para obtener copias de software GNU, o tal vez las hayas obtenido sin ningún coste. Pero independientemente de cómo hayas conseguido tus copias, siempre tienes la libertad de copiar y modificar el software, e incluso de vender copias.

“Software Libre” no significa “no comercial”. Un programa libre debe estar disponible para uso comercial, desarrollo comercial y distribución comercial. El desarrollo comercial del software libre ha dejado de ser inusual; el software comercial libre es muy importante.

Es aceptable que haya reglas acerca de cómo empaquetar una versión modificada, siempre que no bloqueen a consecuencia de ello tu libertad de publicar versiones modificadas. Reglas como “Si haces disponible el programa de esta manera, debes hacerlo disponible también de esta otra” pueden ser igualmente aceptables, bajo la misma condición. (Observa que una regla así todavía te deja decidir si publicar o no el programa). También es aceptable que la licencia requiera que, si has distribuido una version modificada y el desarrollador anterior te pide una copia de ella, debas enviársela.

En el proyecto GNU, utilizamos “copyleft” para proteger de modo legal estas libertades para todos. Pero el software libre sin “copyleft” también existe. Creemos que hay razones importantes por las que el software debe ser copyleft (http://www.gnu.org/philosophy/pragmatic.es.html), pero si tus programas son software libre sin ser copyleft, los podemos utilizar de todos modos, ya que aprendemos de ellos.

Visita la página http://www.gnu.org/philosophy/categories.es.html para ver una descripción de las diferencias que hay entre el “software libre", “software con copyleft (`izquierdo' de copia) “y otras categorías de software se relacionan unas con otras.

A veces las normas de control de exportación del gobierno y las sanciones mercantiles pueden restringir tu libertad de distribuir copias de los programas a nivel internacional. Los desarrolladores de software no tienen el poder de eliminar o sobrepasar estas restricciones, pero lo que pueden y deben hacer es rehusar el imponerlas como condiciones de uso del programa. De esta manera, las restricciones no afectarán a actividades y gente fuera de las jurisdicciones de éstos gobiernos.

Cuando se habla de software libre, es mejor evitar términos como: “regalar” o “gratis", porque esos téminos implican que lo importante es el precio, y no la libertad. Algunos términos comunes tales como “piratería” conllevan opiniones que esperamos no apoyes.

Por último, fíjate en que los criterios establecidos en esta definición de software libre requieren pensarse cuidadosamente para interpretarlos. Para decidir si una licencia de software concreta es una licencia de software libre, lo juzgamos basándonos en éstos criterios para determinar si tanto su espíritu como su letra en particular los cumplen. Si una licencia incluye restricciones contrarias a nuestra ética, la rechazamos, aún cuando no hubiéramos previsto el problema en éstos criterios. A veces un requisito de una licencia plantea una situación que necesita de una reflexión minuciosa, e incluso conversaciones con un abogado, antes de que podamos decidir si la exigencia es aceptable. Cuando llegamos a una conclusión, a veces actualizamos

éstos criterios para que sea más fácil ver por qué ciertas licencias se pueden calificar o no como de software libre.

Licencias GNU/GLP (GNU GENERAL PUBLIC LICENSE)

GNU/GLP - Es una licencia que proporciona la libertad de cambiar y compartir el software, esta licencia se aplica a la mayoría del software de la fundación del software libre y a cualquier otro programa cuyos autores se comprometan a hacer uso de ella.

El término “free software” [En inglés free = libre o gratis] se malinterpreta a veces; no tiene nada que ver con el precio. La connotación adecuada es libertad. Aquí, por tanto, está la definición de software libre. Un programa es software libre, para un usuario en particular, si:

1. Tiene libertad para ejecutar el programa, con cualquier propósito.
2. Tiene la libertad para modificar el programa para adaptarlo a sus necesidades. (Para que esta libertad sea efectiva en la práctica, debe tener acceso al código fuente, porque modificar un programa sin disponer del código fuente es extraordinariamente dificultoso).
3. Tiene la libertad para redistribuir copias, tanto gratis como por un costo.
4. Tiene la libertad para distribuir versiones modificadas del programa, de tal manera que la comunidad pueda beneficiarse con sus mejoras.

Como free (libre) se refiere a libertad y no a precio, no existe contradicción entre la venta de copias y el software libre. De hecho, la libertad para vender copias es crucial: las colecciones de software libre que se venden en CD-ROM son importantes para la comunidad, y la venta de las mismas es una manera importante de obtener fondos para el desarrollo de software libre. Por tanto, si la gente no puede incluir un programa en dichas colecciones, dicho programa no es software libre.

Este concepto se originó en base a un movimiento enfocado a crear un sistema operativo “libre” (sin restricciones de uso y licencias), es el proyecto GNU, bajo el cual se desarrollan miles de aplicaciones y utilidades. El sistema Linux fué incluido en dicho proyecto y, por tanto, actualmente se habla del sistema “GNU/Linux” al referirnos al sistema completo (sistema y aplicaciones que lo acompañan).

La colaboración de un número cada vez mayor de programadores, aficionados y expertos en Linux, fué fundamental para llevar a cabo el rápido desarrollo que ha experimentado. Y desde todo el mundo han surgido las aportaciones que, constantemente, han ido y van mejorando y ampliando las prestaciones de su kernel (núcleo).

Linus Torvalds terminó la que llamó versión 1.0 en el primer tercio de 1994. Hasta entonces había desarrollado varias versiones iniciales a las que fué aportando la funcionalidad básica.

En la actualidad Linux se conoce como un clon de UNiX que varios millones de personas utilizan en todo el mundo; movimiento al que, cada vez, mayor número de grandes compañías (Sun Microsystems, IBM, etc.). Se están uniendo, aportando soluciones tanto comerciales como bajo licencia GPL.

La Naturaleza del Desarrollo de SoftWare Libre y OpenSource

A los defensores del software propietario les gusta decir, “El software libre es un bonito sueño, pero todos sabemos que sólo el sistema propietario puede producir productos confiables. Un puñado de 'hackers' simplemente no puede hacer esto.”

La evidencia empírica disiente, sin embargo; pruebas científicas, descritas más adelante, han comproba-

do que el software GNU es más confiable que el software propietario comparable.

Esto no debiera ser una sorpresa; existen buenas razones para la alta confiabilidad del software GNU, buenas razones para esperar que el software libre tendrá a menudo (aunque no siempre) una alta fiabilidad.

¡Utilidades GNU Más Seguras!

Barton P. Miller y sus colegas probaron la fiabilidad de programas de utilidades de UNiX en 1990 y 1995. En ambas ocasiones, las utilidades GNU se destacaron considerablemente. Probaron siete sistemas UNiX comerciales así como GNU. Sometiéndolos a un flujo de entrada aleatorio, pudieron "abortar (con volcado de memoria) o colgar (bucle infinito) más del 40% (en el peor caso) de las utilidades básicas..."

Estos investigadores comprobaron que los sistemas UNiX comerciales tenían una tasa de fallos que iba desde el 15% al 43%. En contraste, la tasa de fallos de GNU fué sólo del 7%.

Miller también dijo que, "los tres sistemas comerciales que comparamos tanto en 1990 como en 1995 mejoraron considerablemente en fiabilidad, pero aún tenían tasas de fallo significativas (las utilidades básicas de GNU/Linux todavía eran considerablemente mejores que las de los sistemas comerciales)."

Para más detalles, vea su artículo: Fuzz Revisited: A Re-examination of the Reliability of UNiX Utilities and Services (http://www.suffritti.it/informatica/tco/fuzz-revisited.pdf) por Barton P. Miller bart@cs.wisc.edu, David Koski, Cjin Pheow Lee, Vivekananda Maganty, Ravi Murthy, Ajitkumar Natarajan y Jeff Steidl.

Haciendo que el Modelo del Software Libre Trabaje en el Mundo de los Negocios

Muchas personas creen que el espíritu del proyecto GNU es que no se debería cobrar dinero por distribuir copias de software o que se debe cobrar lo mínimo, sólo lo suficiente para cubrir el costo. En realidad, recomendamos a la gente que distribuye software libre que cobre tanto como desee o pueda. Si esto le sorprende, por favor siga leyendo.

Aquí, Stallman escribe, "La palabra inglesa 'free' tiene dos significados generales legítimos; puede referirse a libertad como 'libre' o puede referirse a precio como "gratis". Cuando hablamos de "software libre", estamos hablando de libertad, no de ser gratis. (Piense en "expresión libre", no en "barra libre"). Específicamente, significa que el usuario es libre de ejecutar el programa, cambiarlo, y redistribuirlo con o sin cambios.

A veces los programas libres son distribuidos gratis. Otras veces, se cobra mucho por ellos. A menudo, se puede conseguir un programa libre gratis o a un precio alto en lugares distintos. Pero el programa es libre a pesar del precio porque los usuarios tienen la libertad de usarlo a su gusto.

Normalmente, se venden a un precio alto los programas que no son libres, aunque algunas veces una tienda le dará una copia sin cobrar. Con precio o sin él, el programa no es libre si los usuarios no tienen libertad en su uso.

Como el software libre no tiene nada que ver con el precio, un precio bajo no indica que el programa sea más libre o esté más cerca de serlo. Por eso, si usted está redistribuyendo copias de software libre, puede poner un precio alto y ganar dinero. Redistribuir software libre es una actividad buena y legítima. Si usted lo hace, está bien beneficiarse de ella.

El software libre es un proyecto comunitario, y todo el que depende de él debe buscar medios para con-

tribuir a aumentar esta comunidad. Para el distribuidor, el modo de hacerlo es donar una parte de sus ganancias a la Fundación para el Software Libre u otro proyecto para desarrollar software libre. Al financiar el desarrollo, usted ayuda al avance del mundo del software libre.

Distribuir SL es una Oportunidad para Obtener Fondos para Desarrollar. ¡No la desperdicie!

Para poder contribuir con fondos, usted necesita obtener algo extra. Si cobra una tarifa muy baja, no le quedará nada para apoyar el desarrollo.

¿Perjudicará a los Usuarios un Precio de Distribución más Alto?

A veces a la gente le preocupa que un precio de distribución alto ponga al software libre fuera del alcance de los usuarios pobres. Esto es exactamente lo que ocurre con el software propietario, pero el software libre es diferente. La diferencia es que el software libre se difunde de forma natural, y hay muchos medios para obtenerlo.

Los acaparadores de software hacen su esfuerzo más vil para impedir la ejecución de un programa propietario sin que se haya pagado el precio estándar. Si el precio es alto, resulta difícil el que algunos usuarios puedan usarlo.

Con software libre los usuarios no tienen que pagar la tarifa de distribuición para usar el software. Pueden hacer copias del programa de amigos o con la ayuda de algún amigo que tenga acceso a Internet. O varios usuarios se pueden juntar, repartir el coste de un CD-ROM e instalar el software por turnos. Un precio alto de CD-ROM no es gran obstáculo cuando el software es libre.

¿Desincentivará el Uso del Software Libre un Precio de Distribución más Alto?

Otra preocupación común es por la popularidad del software libre. La gente cree que una tarifa alta de distribución podría reducir el número de usuarios, o que un precio bajo probablemente aumentará su número.

Esto es cierto para el software propietario - pero el software libre es diferente. Con tantas formas de obtener copias, el precio de distribución afecta menos a la popularidad.

A la larga, el número de usuarios de software libre está determinado, fundamentalmente, por cuánto puede hacer el software y si es fácil usarlo o no. Muchos de los usuarios van a seguir usando software proprietario si el software libre no puede hacer el trabajo que desean. Así, si queremos aumentar el número de usuarios a largo plazo, debemos sobre todo desarrollar más software libre.

El modo más directo de hacer esto es que usted mismo escriba el software libre o los manuales técnicos que necesitamos. Pero, si usted distribuye el software en vez de escribirlo, la mejor forma de ayudar sería conseguir dinero para que otros escriban los programas.

El Término “Vender Software” Puede Ser Confuso

Hablando con exactitud, “vender” significa cambiar mercancía por dinero. Vender una copia de un programa libre es legítimo, y lo apoyamos. Sin embargo, cuando la gente piensa en “vender software”, usualmente se imaginan que se trata de venderlo como la mayoría de compañías lo hacen: haciendo elsoftware propietario en vez de libre.

Por eso, a menos que usted vaya a hacer distinciones finas, como hace éste artículo, le sugerimos que evite la frase “vender software” y escoja otra. Por ejemplo, usted podría decir “distribuir software libre por

un precio". Esto no es ambiguo.

Precios altos o bajos y la Licencia Pública General de GNU

Salvo en un caso especial, la Licencia Pública General de GNU (GPL de la GNU) no establece restricciones respecto a cuánto puede cobrar usted por distribuir una copia de software libre. Usted puede cobrar nada, un centavo, un dólar, o un billón de dólares. Sólo depende de usted y del mercado. Y no nos reclame si nadie quiere pagar un billón de dólares por una copia.

La única excepción es el caso en el que los ficheros binarios son distribuidos sin el código fuente completo correspondiente. La GPL de la GNU requiere a quienes hacen esto que también provean el código fuente completo en el caso de una solicitud posterior. Si no hubiese un límite para el precio del código fuente, el distribuidor podría cobrar tanto que nadie pudiera pagarlo --por ejemplo, un billón de dólares-- y así decir que facilita el código mientras que en realidad lo estaría reteniendo. Por eso en éste caso tenemos que limitar el pago por el código fuente para proteger la libertad del usuario. Pero en situaciones ordinarias, no hay justificación para limitar los precios de distribución. Por tanto, no las limitamos.

De vez en cuando, compañías cuyas actividades exceden los límites establecidos por la GPL de GNU piden permiso, diciendo que “no van a cobrar dinero por el software GNU” o algo por estilo. Así no van a lograr nada. El software libre trata de libertad, y aplicar y defender la GPL es defender la libertad. Cuando defendemos la libertad de los usuarios no nos distraemos por cuestiones secundarias como cuál es el precio de distribución. Libertad es el asunto, todo el asunto y el único asunto.

Como lo ve RedHat

El distribuidor de GNU/Linux Red Hat tiene un negocio que incluye distribuir gratuitamente CDs por Internet; pero, ellos venden miles de CDs y se han convertido en una compañía billonaría comercializando Linux a alrededor de US$60 los mismos CDs. Lo que Red Hat ha logrado es cambiar la perspectiva de lo que vende de los CDs ha una configuración especifica y servicios. Lo que ellos venden es que atraves de sus CDs es más fácil que con los CDs que descargas desde el Internet.

Pasos Para vender Software Libre

Tomemos como ejemplo una empresa pequeña, imaginate un cliente FTPque le permite al usuario dar los comandos siempre igual independiente del sistema operativo en el cual el se encuentre. Es decir si el FTPesta bajo GNU/Linux o cualquier otro sistema operativo, el usuario todavía podría ejecutar los comandos en el servidor igual que si fuese bajo de GNU/Linux.

Además, permitir la compañía generar ingresos y beneficio mientras que continúa publicando el software bajo licencia de fuente abierta (OpenSource), un negocio pudo tomar las medidas siguientes:

- El software, creado con comentarios en el código fuente con sus páginas de manual (man) al estilo UNiX, es librada bajo la licencia GPL y colocada en el Internet para su libre descarga.
- Se crean Forums y grupos de noticias para invitar a otros que prueben el software y ofrezcan sus consejos sobre cómo mejorarlo.
- Si se asume que el software es estable y la compañía está satisfecha con su funcionalidad, un CD es creado para hacer el programa fácil de manejar para los usuarios inexpertos.
- Una vez la compañía éste satisfecha con el software, crearía un “Web site y ofrecería el CD para la venta.
- De acuerdo con la documentación, los tutoriales, o los manuales de usuario detallados que otros autores GLP han vendido, la compañía pueden desear ofrecer una documentación más detallada para distribuirla con el CD o como un libro separado.

¿Puede Software GNU GPLed Ser Mejor que el Software Proprietaria?

No es casualidad que las utilidades GNU sean más confiables. Hay buenas razones por las cuales el software libre tiende a ser de alta calidad. Una razón es que el software libre consigue involucrar a toda la comunidad para que trabaje unida para arreglar problemas. Los usuarios no sólo informan de errores, incluso los arreglan y envían los arreglos. Los usuarios trabajan juntos, conversando por correo electrónico, para alcanzar el fondo del problema y hacer que el software trabaje sin problemas.

Otra es que los desarrolladores se preocupan realmente de la fiabilidad. Los paquetes de software libre no siempre compiten comercialmente, pero sí compiten por una buena reputación y un programa que sea insatisfactorio no alcanzará la popularidad que los desarrolladores esperan. Lo que es más, un autor que pone el código fuente al alcance de la vista de todos arriesga su reputación, y le conviene hacer el software limpio y claro, bajo pena de la desaprobación de la comunidad.

Puestas en Práctica Importantes De Linux

La puesta en práctica de GNU/Linux puede ser describida de diversas maneras. Estas pueden incluir varios tipos de arquitecturas y distribuciones que obedecen un estándar pero que mantienen sus diferencias.

- Intel 80x86, Sun SPARC, IBM PPC, DEC Alpha, Motorola
- Red Hat, TurboLinux, SuSE, Debian/GNU. Slackware.

Arquitecturas

La Arquitectura se refiere al diseño específico y la construcción de una computadora, su CPU (procesador), y el tamaño del Conjunto de bytes que puede procesar (Ej. 8-bit, 16-bit, 32-bit o 64-bit). Como la mayoría de ustedes sabe, Linux sólo es un núcleo (kernel). Durante mucho tiempo el núcleo Linux sólo corría en la serie de máquinas x86 de Intel, para la cual el fué diseñado, desde el 386 en adelante..

Sin embargo, hoy día esto ya no es cierto. El núcleo Linux ha sido adaptado a una larga y creciente lista de arquitecturas. En general éste proceso tiene un comienzo difícil (hay que conseguir que la libc y el enlazador dinámico funcionen sin trabas), para seguir luego el proceso rutinario, y largo, de conseguir recompilar todos los paquetes bajo las nuevas arquitecturas.

Distribuciones

Dado que GNU/Linux es un sistema operativo libre (gratuito), cualquiera puede agarrar un montón de programas (GNU principalmente), colocarlos en un CD y distribuirlos como un Linux; es decir, no hay un GNU/Linux, hay muchos (alrededor de unos 1,600), tantos como distribuciones. Cada distribución elige qué programas va a incluir y cuales no, tiene sus propios programas de instalación, sus propios interfaces gráficos, eligen versiones determinadas de programas, y finalmente, benefician a una empresa u otra. Algunas distribuciones populares son la RedHat, la SuSE, TurboLinux, Debian, esWare. La mayoría se pueden descargar desde http://linuxiso.org.

Los Estándares

Aunque existen muchas distribuciones de GNU/Linux con métodos variados de lograr tareas (como la de instalación) y al usuario nuevo esto le parezca algo desorganizado es todo lo contrario. Existen entidades que se dedican a la estandarización de GNU/Linux. Esto es para garantizar y preservar compatibilidad entre todas las distribuciones y los sistemas operativos Tipo-Unix y los que no son. La Linux Standard Base (LSB), por ejemplo, es una entidad que define el estándar de los paquetes de software ha ser incluidos

en las distribuciones. La Filesystem Hierarchy Standard (FHS), la cual dicta el estándar de Archivos GNU/Linux y su debida estructura.

Como existe una gran variedad de sistemas operativos de tipo de UNiX y UNiX-Like, también existe un comité de Estándares que los gobierna, llamado el Portable Operating System Interfaz for UNiX (POSIX). POSIX es una definición de interfaz a la cual los sistemas deben conformar, la gran mayoría de los sistemas modernos de GNU/Linux tratan de adherirse estrictamente a éstos estándares de cumplimientos POSIX. Desarrolladores de GNU/Linux se ajustan a los Estándares que se concentran en la funcionalidad del kernel y la Interfaz de Programación de Aplicaciones (Application Programming Interfaz, API).

Linux: Positivos

GNU/Linux tiene un número de cualidades positivas:

- Opera en una amplia gama de hardware
- Conjunto de comandos Poderosos y flexible
- Ambiente de Desarrollo de Software estable
- Capacidades robustas intrínsecas de redes/networking
- Bajo (casi siempre hasta gratís) costo de compra/adquisición y alta disponibilidad de soporte

La popularidad de GNU/Linux puede ser atribuida a un número de factores, de los cuales se pueden resaltar algunos:

- Las utilidades estándares de GNU/Linux son muy versátiles y diversas en sus funciones y pueden ser extendidas por desarrolladores a través de modificación de su código fuente.
- GNU/Linux fué desarrollado con el propósito de facilitar el continuo desarrollo de el mismo, y como resu tado, el a madurado en un ambiente de desarrollo robusto y amistoso.
- Esta disponible en una amplia gama de plataformas de hardware y en cada una de ella presenta un interfaz my similar por no decir la misma independiente de ellas, o sea que es muy fácil llevar los programas que haces para i386 a PPC de Mac/IBM/Morola.
- Es relativamente fácil migrar un sistema GNU/Linux a otro a partir de radicalmente diferente plataformas de hardware. Como lo es también, generalmente fácil, migrar de un GNU/LINUX a UNiX o a otro GNU/Linux.
- Existe una gran cantidad de métodos para intercomunicar a GNU/Linux con otros sistemas operativos. El Kernel de GNU/Linux contiene los elementos básico y utilidades de terceros que complementan la interconectividad con otros sistemas operativos como los son sistemas UNiX o UNiX-Like, Novell, Macintosh, OS/2, y otros no basados en UNiX.

Ventajas de GNU/Linux

GNU/Linux a diferencia de otros sistemas operativos, es multítarea real, y Multiusuario; posee un esquema de seguridad basado en usuarios y permisos de lectura, escritura y ejecución establecidos a los archivos y directorios. Esto significa que cada usuario es propietario de sus archivos, y otro usuario no puede acceder a éstos archivos. Esta propiedad no permite el contagio de virus entre archivos de diferentes usuarios.

Además, GNU/Linux posee un entorno gráfico (X-Window) que le aporta al Sistema Operativo vistosidad por un lado y facilidad de manejo por otro. Al igual que los entornos gráficos de otros sistemas (Solaris, Apple Mac) X-Windows ofrece un entorno multiventana, pero a diferencia de aquellos, X-Windows supone el núcleo sobre el cual se pueden ejecutar distintos gestores de ventanas.

Una diferencia, quizás la más importante de todas, con respecto a cualquier sistema operativo comercial, es el hecho de que éste es software libre, ¿qué quiere decir esto? que junto con el sistema, se puede obtener

el código fuente de cualquier parte del mismo y modificarlo a gusto. Esto da varias ventajas, por ejemplo:

- "La seguridad de saber qué hace un programa tan sólo viendo el código fuente, o en su defecto, tener la seguridad que al estar el código disponible, nadie va a agregar "características ocultas" en los programas que distribuye"
- "La libertad que provee la licencia GPLpermite a cualquier programador modificar y mejorar cualquier parte del sistema, esto da como resultado que la calidad del software incluido en GNU/Linux sea muy buena"
- "El hecho de que el sistema sea mantenido por una gran comunidad de programadores y usuarios alrededor del mundo, provee una gran velocidad de respuesta ante errores de programas que se van descubriendo, que ninguna compañía comercial de software puede igualar"

Debilidades de GNU/Linux

GNU/Linux tiene un número de puntos débiles que se pudiesen enumerar así:

- No es tan fácil administrar para los principiantes o esos acostumbrados a GUIs, tal como sistemas operativos más concentrados en apariencias y menos en seguridad.
- Debido a que todavia algunos fabricantes se muestran cerrados a entregar los APIS de sus dispositivos, el soporte a nuevos dispositivos es un poco más lento que el de otros sistemas operativos.
- Los programas son estructurados en forma monolítica, y las dependencia de uno al otro causa interdepe dencias a veces difíciles de llenar lo requerido.

El interfaz natural de GNU/Linux, es la línea de comando, fué diseñado para ser un ambiente de gran alcance desde el cual el sistema entero podría ser controlado. Para los usuarios principiantes puede ser un poco intimidante, y conduce a críticas del sistema en su totalidad por personas que creen que éste punto de vista es anticuado. Su gran número de comandos tienden a ser cortos, y uniformemente crípticos. GUIs tal como el sistema de Ventana X (XWindow) facilitan un poco la introducción al nuevo usuario.

Aunque el kernel Linux soporta una amplia gama de arquitecturas y dispositivos de hardware, sólo recientemente importantes fabricantes han comenzados a desarrollar sus propios módulos (drivers). Los dueños del hardware sin apoyo alguno desarrollaban la mayoría de los drivers de dispositivo del Kernel Linux, ellos entraban en contacto con los fabricantes y recibían la cooperación de ellos, obteniendo las especificaciones necesarias para escribir los drivers. Estos pasos tienden a tomar un largo tiempo y dan lugar a que a veces los usuarios de GNU/Linux tuviesen que esperar por drivers para Linux.

GNU/Linux ha ganado gran notoriedad, por su ayuda y continuo crecimiento del soporte de nuevos hardware, a través de dueños individuales construyendo los drivers de éstos dispositivos y ahora, con más frecuencia, los fabricantes de hardware los construyen ellos mismos.

GNU/Linux provee simple herramientas de sistema para print-spooler y utilidades para implementar backup. Amedida de los sistemas de GNU/Linux se han ido incorporando en los ambientes corporativos los administradores han sobre pasado estas herramientas simples y han nacidos sistemas completos, tanto comerciales como OpenSource. Mucha de estas ya viene como parte de las mayorías de distribuciones comerciales de GNU/Linux.

Múltiple Usuarios y Multitareas

Sistemas Multiusuario

- Permiten que varios usuarios usen el sistema de forma simultánea.
- Sistema de pertenencias y permisos sobre archivos y procesos.

Sistemas operativos Multiusuarios son aquellos con la habilidad para soportar dos o más usuarios y que transparentemente compartan tiempo y recursos. En un sistema multiusuario, un usuario puede ejecutar la misma cosa que otro usuario en el sistema. GNU/Linux permite que muchos usuarios accesen una misma computadora a la vez, y el sistema puede diferenciar entre un usuario y el otro, al requerir un procedimiento de login en el cual cada usuario debe dar un nombre y una contraseña. Cierto tipo de redes permiten que muchos usuarios accesen a un sistema GNU/Linux desde diferente terminales a la misma vez. En sistemas de único-usuario (mono-usuario), sólo puede haber acceso a recursos por éste único usuario..

Sistemas Multitarea (Multitasking)

- El sistema puede correr varios procesos a la vez;
- Los recursos del sistema de cada proceso están protegidos;
- Existe un mecanismo de control de procesos;
- Se puede hacer una distinción dependiendo de la cantidad de control que el sistema cede a los procesos.
    - Cooperative multitasking: Los procesos toman control del sistema y deben devolverlo al SO.
    - Preemptive multitasking: Los procesos son manejados completamente por el SO.

Multítarea es la habilidad de hacer más de una acción al mismo tiempo. El CPU de la computadora asigna tiempo de proceso a las aplicaciones a medida que ellas lo van requiriendo, así permitiendose un rápido y efectivo desenvolvimiento. GNU/Linux permite que muchas tareas se ejecuten concurrentes. Tareas que consumen mucho tiempo pueden ser enviadas a ejecutarse en segundo plano y de esta manera no necesitar intervención del usuario. Además, tareas pueden programarce para ser ejecutadas a cierto tiempo del día. Aplicaciones corriendo en el sistema XWindow pueden ejecutarse en su propia ventana sin afectar otras aplicaciones.

Sistemas que ofrecen Multítareas Cooperativas (cooperative multitasking como MacOS 8 y 9, entre otros) no son de verdad con derecho preferente, y una aplicación en particular puede no cooperar y tomar a la fuerza el control de todo el ordenador (esto causaría los famosos congelamiento/freeze del sistema completo o por lo menos de la aplicación). En GNU/Linux Aplicaciones no poseen esta habilidad, de arrastrar el sistema completo. Esto sólo puede ocurrir a través de un acceso indebido a hardware de una aplicación (por ejemplo un Servidor de Sonido o el Servidor de X Window).

INTERFACES DE USUARIOS

Generalmente el usuario tiene tres maneras de interactuar con sistemas GNU/Linux: ejecutar comandos en la línea de comandos, responder a un prompt del shell, o interactuar con un escritorio Gráfico lleno de iconos y menus utilizando una combinación de clicks de cursores de mouse y teclado.

En esta sección, cubriremos los siguientes temas:

- Interfáz de Linea de comandos (CLI, Comand Line Interface)
- Interfáz Grafica de Usuarios (GUI, Graphical User Interface)
- Combinando Shells y GUIs

Interfaces de Línea de Comandos (CLI)

Como todo en UNiX existe una gran variedad de shells o interpretes de línea de comandos, éstos shells actúan como la interfaz entre el usuario y el sistema GNU/Linux. Esta interfaz de usuario, o shell, es el programa que responde a los comandos digitados por el usuario, sirviendo de interlocutor entre el sistema

y el usuario. El primer shell fué el Bourne Shell, mejor conocido como 'sh'. En los sistema GNU/Linux, el que se utiliza por defecto es el Bourne-Again Shell, o el bash, cual es una variante del sh. Otros ejemplos de shells son ash, csh, pdksh, tcsh, y zsh, la mayoría tienen características en lo interno de ellas, las cuales pudiesen en un dado momento resultar que su uso sea ventajoso sobre otra shell. Algunas de estas características son: historia reusable de comandos ejecutados, la disponibilidad de editar comandos recordados, y la disponibilidad de poder ejecutar comandos en segundo plano.

Una característica importante del shell es que le permite a los usuarios dirigir la salida de un comando como la entrada de otro comando, así combinando su uso y convirtiendolo en un sólo. Esto es conocido como 'tuberías'(pipes) y es logrado con el símbolo de '|'entre los comandos.

Podemos escribir comandos en archivos de textos que pueden ser ejecutados cuando sean necesarios. El shell interpreta éstos archivos línea por línea y ejecuta los comandos como si fueran digitados por el usuario mismo.

Hay varios interfaces de consolas gráficas, tales como el Midnight Commander (mc), que puede ser invocado para proporcionar ayuda en la línea de comando sin tener que accesar un GUI totalmente gráfico. Estas utilidades son manejadas por eventos y se han estructurado para hacer ciertas tareas y mostrar una representacióno un listado de qué funciones ellas son capaces de gestionar. Esto ayuda al usuario por que así no tiene que recordar una gran cantidad de comandos o variables para modificarlas.

Interfáz gráfico de Usuario (GUI)

El GUI dominante de GNU/Linux es el servidor XFree86 XWindow, el cual esta basado en el protocolo X desarrollado por el MITa mediado de los años 80s. Otros sistemas operativos como el IBM Presentation Manager sólo despliegan GUI en las estaciones de trabajos (workstation) en la que está ejecutando; el protocolo X perite sistemas XWindow ser utilizados vía la red (Networked).

El protocolo X define una relación verdaderamente cliente/servidor. El denominado "look and feel" de las aplicaciones permanecerá idéntico en una PC local o en una remota si existe un verdadero ambiente de redes (networked environment). El proceso de despliegue es completamente divorciado de la aplicación, y no existe ninguna pérdida de funcionamiento al desplegar una aplicación local comparada con una remota (si existe algo de tardanza entonces ya es un problema de ancho de banda, lo cual entonces es relacionado a la red y no la aplicación).

La apariencia y manejo del ambiente del XWindow no es gobernada por el propio servidor. Una aplicación, llamada el manejador de ventana (window manager), se ejecuta sobre él y es cual nos presenta el inte faz. Esto es idéntico en la manera que un shell se coloca encima de un sistema operativo, permitiendo así al usuario introducir comandos. Hay diversos manejadores de ventana disponibles. Uno podría configurar ambientes radicalmente diferentes que ejecutan las mismas aplicaciones. El administrador pudiese configurar el entorno que fuese muy parecido a MacOS u otros para que así usuarios menos expertos pudiesen ser introducidos a GNU/Linux con menos trauma.

Ya que el sistema X Window no proporciona ningún escritorio verdadero (por ejemplo, para almacenar aplicaciones y hipervínculos sobre el escritorio como lo hace MacOS), hasta hace poco tiempo, las aplicaciones tenían que ser lanzadas desde el menú de inicio del manejador de ventana. Proyectos tales como GNOME y KDE han cambiado esto, permitiendo a usuarios del XWindow disfrutar de las ventajas de un ambiente de escritorio verdadero.

Combinando los Shells y GUIs

El poder de cada interfaz es diferente, pero si pueden ser complementario. En lo específico, mientra utiliza el GUI, el usuario de GNU/Linux puede ejecutar comandos desde un xterm, así permitiendole al usuario trabajar desde la interfaz que le produzca mayores ventajas y facilidad de uso.

Usuarios avanzados que pueden ejecutan desde la línea de comandos, a menudo se quejan que los GUIs son poco flexibles. Tareas como la de borrar o mover archivos toman más tiempo desde el X que desde el terminal. Así como casi siempre una tarea requiere varios pasos de visualización, ejecución con el mouse.

Un administrador GNU/Linux debe poder efectuar sus tareas desde ambos ambientes o interfaces. Existen tareas que requerirán la intervención del administrador desde ambas interfaces y conocerlas será absolutamente ventajoso para el administrador. Determinar que combinación es apropiada para una tarea es decisión del administrador, para elegir entre una combinación que le rinda velocidad, fácil manejo, y sobre todo poder.

Sesiones de GNU/Linux

Antes de poder instalar y configurar GNU/Linux en su sistema, hay unos conceptos básicos que debes manejar y estar familiarizado:

- Usuarios GNU/Linux
- Procedimiento de Ingreso (Login)
- Estructura del Shell y la Línea de Comandos
- Teclas Especiales
- Comandos Simple
- Páginas Man
- Tipos de Sesiones de Línea de Comandos

Usuarios GNU/Linux

Para poder comprender el proceso de Login, necesitarás entender más de los Usuarios y como es que el sistema los autentifica..

Que es un Usuario

GNU/Linux identifica los usuarios individuales a través de su autentificación. Un prompt es presentado a los usuarios Para que se le identifiquen al sistema vía éste Login de su nombre de usuario más su contraseña (password). Este es el primer paso (y de echo el más importante) para mantener un sistema libre de uso sin autorización de los recursos.

Autentificación del Usuario

Los usuarios son autentificados por el sistema de la siguiente manera:

1. El sistema le presenta un prompt al usuario con la palabra Login:
2. El usuario digita su nombre asignado por el administrador, por ejemplo "cperez".
3. El sistema le presenta en el prompt la palabra Password:
4. El usuario escribe su password, por ejemplo "lZz02uyt".
5. El sistema valida esta información suplida por el usuario y subsecuentemente si todo marcha bien, y el usuario es legitimo, el sistema le asigna un User Identification Number (UID, Número de Identificación del Usuario), por ejemplo el 735. Este número es utilizado por el sistema para saber que procesos fueron iniciados por cual usuario.

Tipos de Usuarios

Existen diferentes tipos de identidades de usuarios. Tenemos el superusuario, su nombre es root, quien es el administrador del sistema, usuarios creados para ejecutar los procesos del sistema, y usuarios normales con privilegios limitados. Los usuarios asociados con los procesos del sistema, son para asociar archivos privilegiados necesarios para el buen funcionamiento del sistema. Todo el tiempo entrarás al sistema como un usuario con sólo cierto privilegios, sólo utilizando la cuenta de root cuando va a ejecutar tareas administrativas. Es de suma importancia mantener la disciplina de sólo utilizar la cuenta de root cuando es absolutamente necesario, ya que root puede ejecutar comandos que pueden deshabilitar el sistema operativo por completo.

Acceso Para los tres tipos de usuarios:

- Superusuario
  - o Sin restricción el sistema completo
  - o Inherente a todos los sistemas UNiX y los tipos UNiX-Like
- Usuarios de Procesos
  - o Acceso restringidos a los recursos requeridos por el proceso que ejecuta
  - o Inherente a cada y todos los sistemas operativos UNiX y los tipos UNiX-Like
- Usuarios Sin Privilegios
- Restringidos a áreas y actividades a las cuales se les ha dado permiso
- Creados por el administrador del sistema a medida que sean necesarios

Procedimiento de Login

El proceso de login es su ingreso al sistema GNU/Linux. Es una manera simple de asegurarse que sólo personas autorizadas ingresen al sistema y así manteniendo el uso anónimo imposible. La contraseña es la llave a la puerta de entrada al sistema. Sin esta llave la puerta simplemente no abre, si la llave se pierde sólo el root puede asignar otra llave.

- El Login: muestra al usuario que el sistema esta disponible.
- El usuario ingresa utilizando su nombre y contraseña (username and password).
- Un mensaje de bienvenida puede ser desplegado si así lo desea el administrador (root).
- La interfáz shell se inicia y un prompt se desplega.

```
Red Hat Linux release 7.3 (Valhalla)
Kernel 2.4.8 on an 1386
login: abiertos
password:
Recuerde que los Backups empiezan a las 11PM hoy
```

Estructura del Shell y la Línea de Comandos

El shell es la interfaz que le permite al usuario escribir las ordenes o comandos. El shell más utilizado en GNU/Linux es el bash. Otras disponibles son csh, tcsh, ksh, zsh, sh y claro existen más..

Escribiendo Comandos

Unas cuantas cosas que debes saber para escribir ordenes en la línea de comandos:

- Los comandos se escriben el el prompt del shell, casi siempre éste prompt es representado por un "$" o "#" (dependiendo de la cuenta de usuario que estas utilizando en el momento –el superusuario tiene el prompt # por defecto, aunque esto es configurable desde la variable de ambiente PS1).

- Los comandos son similares en todos los sistemas GNU/Linux. Los nombres de los comandos pueden que varíen de una distribución a otra.
- GNU/Linux distingue entre mayúscula y minúsculas.
- El comando empieza la sentencia el resto son sus argumentos. Los argumentos son separados del comando por un espacio en blanco.
- Al shell se le dice que ejecute un comando simplemente pulsando la tecla ENTER

Un ejemplo de un comando a ejecutarse:

$ ls -l -a /home/abiertos

Prompt del Shell	Comando	Opciones	Objectos
$	ls	-l -a	/home/abiertos

- Las Opciones modifican el comportamiento de un comando:
- Usualmente precedido por un (-), después del cual se puede combinar multiples opciones
- Las Opciones mismas pueden que tengan argumentos
- Los Objetos casi siempre son nombres de archivos (o directorio).
- Las Opciones y los Objetos son ambas referenciados como los argumentos de los comandos.
- Algunos comandos no requieren argumentos para ejecutarse.

Teclas Especiales

Estas son caracteres que no se imprimen pero que tienen un efecto predefinido al escribirse,

- Borrar el caracter previo (CONTROL+h) -Usualmente el mismo efecto que presionar la tecla BACKSPACE puede ser la tecla DELETE key en algunos sistemas
- Borrar toda la línea actual (CONTROL+u)
  - o No se despliega el prompt.
  - o Efectivamente se lleva todos los caracteres en la línea actual.
- Interrumpir (abortar) el comando actual (CONTROL+c)
- El prompt se despliega de nuevo.
  Puede ser la tecla DELETE en algunos sistemas.
- Final de entrada (CONTROL+d) -Si se ejecuta en el shell, esta combinación de teclas le hará un log out del sistema.

Las teclas especiales se ejecutan de la siguiente forma por ejemplo, para enviar la señal CONTROL + c señal, presione fijo la tecla CONTROL y luego presiona c.

Las teclas para enviar las señales de suspender y reiniciar la salida a pantalla son (CONTROL+s y CONTROL+ q) y son funciones Estándares ASCII disponible en las mayorías de distribuciones de GNU/Linux. Ellas son utilizadas para iniciar y detener la salida a la pantalla cuando se despliega demasiada información. GNU/Linux tiene un mecanismo Para paginar la salida a pantalla la cual describiremos más adelante.

Comandos Simples

Estos simples ejemplos nos pueden dar a demostrar como funcionan los comandos. Estos pequeños ejemplos nos demostraran La sintaxis básico de los comandos. Entre los paréntesis cuadrados incluiremos algunas de las opciones disponibles.:

date [+format]	despliega hora y fecha actual
cal [[mes] año]	despliega el calendario del mes/año
who	Lista los usuarios actualmente en el sistema

Aquí un ejemplo utilizando unos comandos:

```
$ echo Hola Mundo > /home/nombre_usuario/saludo
$ cat /home/nombre_usuario/saludo
```

Esto escribe y luego despliega el texto "Hola Mundo", primero lo escribe a un archivo de texto llamado saludo y luego lo despliega a pantalla.

Páginas man

Todas las distribuciones de GNU/Linux vienen con alguna versión de las páginas man ya instalada. Las páginas man son la referencia sobre como funciona un comando debido a que normalmente fueron escritas por el mismo programador que creó el comando. Las páginas man, sin embargo, a menudo no contendrán los conceptos subyacentes necesarios para comprender el contexto en el cual un comando se usa. Por consiguiente, no es posible para una persona aprender sobre UNiX meramente desde las páginas man. Sin embargo, una vez que haya adquirido la experiencia necesaria sobre un comando, entonces su página man se vuelve una fuente indispensable de información y puede deshacerse de otro material de iniciación.

Ahora, las páginas man se dividen en secciones, numeradas del 1 al 9. La Sección 1 contiene todas las páginas man para los comandos del sistema como los que ha estado usando anteriormente. Las secciones de la 2 a la 7 contienen información para programadores y para el que le guste la programación, que probablemente aún no tendrás que consultar. La sección 8 contiene páginas concretamente para los comandos de administración del sistema. Las páginas man son referenciadas con notación como cp (1), para el comando cp en la sección 1, que puede ser leída con man 1 cp.

A continuación aparecen las secciones más importantes:

1. Programas de usuario
2. Llamadas al sistema
3. Llamadas a bibliotecas
4. Archivos especiales
5. Formatos de archivo
6. Juegos
7. Variado
8. Administración del Sistema
9. Documentación sobre el kernel

Las páginas info

Las páginas info contienen excelente información de referencia y tutorial en formato de hipertexto. Escribe info en la línea de comandos para ir al menú del nivel superior de toda la jerarquía info. Pinfo es un programa interactivo, al igual que info, con teclas para navegar y buscar documentación.

Hay otros programas para visualizar y navegar la documentación que ya vimos. Uno de ellos es konqueror, el navegador multipropósito del proyecto KDE y el otro es la aplicación tkinfo.

Tipos de Sesiones

Sesión de Línea de Comandos

Como hemos ya mencionado, una de las ventajas de GNU/Linux es su capacidad de utilizar la Línea de Comandos (Interfaz de Línea de Comandos, CLI) para controlar el sistema. La CLI puede ser accesada de manera diferente:

- Terminal desde el XWindow

- Consolas Virtual
- Sesiones de Telnet
- Sesiones de Dial-In

Terminal desde el XWindow

Al ejecutar una sesión del Sistema XWindow, iniciar una sesión de terminal es extremadamente fácil. Existe un sinnúmero de programas de terminales disponibles que se ejecutan bajo el ambiente XWindow. Ya que estas ejecutando una sesión en el X, al iniciar una en el terminal se encontrará ya ingresado al sistema con el mismo usuario.

Consolas Virtuales

Las consolas virtuales te permiten tener sesiones simultáneas en la misma máquina sin necesidad de tener montajes complicados como una red o ejecución de X. Cuando el sistema arranca, mostrará el prompt de login en el monitor una vez finalizado el mismo. Puedes entonces teclear tu login y password y empezar a trabajar, en la primera consola virtual.

En algún momento, probablemente querrás iniciar otra sesión, por ejemplo, para mirar la documentación de un programa que estás ejecutando, o para leer el correo mientras esperas que termine una sesión ftp que tienes establecida. Sólo presiona la combinación de teclas Alt+F2 y encontrarás un prompt un prompt de login esperandote en la segunda "consola virtual". Cuando quieras volver a la sesión original, sólo tienes que pulsar Alt+F1.

La instalación por defecto de GNU/Linux tiene siete consolas virtuales activadas, y Alt+F1, Alt+F2, Alt+F3 hasta Alt-F6 y el Alt+F7 que es la que contiene la sesión de X gráfica, puedes cambiar entre ellas, como sea necesario.

Ejercicio 1-1: Navegar en las Consolas Virtuales

Este ejercicio cubre los conceptos de navegar entre consolas virtuales. Soluciones a éste ejercicio se proveen en el Apéndice A.

1. Desde el prompt de la primera consola virtual, ejecute el comando ps:

   $ ps

2. Repita éste comando en tres otras terminales.

3. Ahora con la tecla ALT+Fn (Fn las diferente consolas virtuales que inicio arriba el comando ps), presione, en cualquier orden, cada tecla correspondiente a un terminal ejecutando su comando ps (F1...F12). Observe los cambios en la pantalla. ¿Cuál es la relación entre las teclas presionadas y el resultado del comando ps?

Sesiones de Telnet

Si el servicio de telnet esta disponible entonces usuarios de la red pueden accesar el sistema desde un cliente telnet (clientes telnet se encuentran preinstalados en la gran mayoría de sistemas UNiX, UNiXLike, y otros sistemas operativos). Una sesión telnet es virtualmente idéntica a una sesión de ingreso al sistema, pero quizás encuentres que el cliente telnet requiere un poco más de configuración para ponerse ne marcha. Cuando incurra en sesiones vía telnet también descubrirá que no puedes accesar las consolas virtuales desde su sesión de telnet.

Sesiones de Dial-In

Si tienes un modem que esta operando correctamente conectado al sistema, usuarios entonces se podrían conectar a través de la línea de teléfono desde localidades remotas utilizando sus modems. Aunque esto requiere configuraciones avanzadas, sus principios son relacionados a los otros métodos de sessión que hemos estado discutiendo. La aplicación que se ejecuta durante el proceso de iniciación que habilita y nos permite ingresar (login) en las consolas virtuales, se llama getty; una aplicación similar llamada mgetty, escucha sobre las líneas seriales para usuarios deseando ingresar al sistema a través de conexiones vía telefónicas o mejor conocidas como dialing in.

Precauciones

Si permites conexiones desde terminales adicionales a las del sistema, sin importar si desde la red interna o a través de un modem, asegúrese de no utilizar contraseñas débiles en todas las cuentas con permiso para acceder el sistema.

Resumen

En éste capítulo usted fué introducido a GNU/Linux, su historia, y sus pros y contras. Esto incluyó:

- GNU/Linux es un sistema operativo multiusuario, multítareas.
- Originalmente creado por Linus Torvalds como un clone gratuito de UNiX para programadores y entusiasta del computador.
- GNU/Linux esta disponible para una gran variedad de equipos.
- GNU/Linux trabaja excelente en ambientes heterogéneos de redes.
- GNU/Linux es poderoso y flexible.
- El shell es la base donde los comandos son ejecutados.
- Existen alternativas gráficas al uso de la línea de comandos.
- GNU/Linux provee autentificación de usuario.
- Páginas del man proveén una ayuda extensiva en línea en todo momento.
- Hay muchas maneras de establecer una sessión en un sistema de GNU/Linux.

Preguntas Post-Exámen

Las respuestas a estas preguntas están en el Apéndice A.

1. ¿Qué define el GUI de GNU/Linux?

2. ¿Cuántas distribuciones existen de GNU/Linux?

3. ¿Puedo usar una copia de una Distro de GNU/Linux para instalar en multiples equipos?

4. ¿Cómo puede el usuario interactúar (los interfaces) con GNU/Linux?

5. Liste 4 implementaciones de GNU/Linux en diferentes arquitecturas.

6. ¿Por qué utilizaría GNU/Linux en vez de un sistema operativo propietario?

7. ¿Por qué utilizaría un sistema operativo propietario en vez de GNU/Linux?

CAPITULO 2

ADMINISTRACIÓN DEL SISTEMA OPERATIVO GNU/LINUX

INSTALAR GNU/LINUX

OBJETIVOS

Al completar este capítulo, usted podrá:

- Describir los pasos comunes a todas las instalaciones de GNU/
- Describir que es un paquete y como usarlo
- Configurar hardware fundamentales del sistema
- Detallar configuraciones de de la arquitectura del PC
- Contrastar adaptadores de video versus capacidad de los monitores
- Detallar ventajas y desventajas de los diferentes medios de instalación
- Detallar estrategias de particionar discos duros.
- Listar los parámetros de configuración de networking (red)

PREGUNTAS PRE-EXAMEN

Las repuestas se encuentran en el Apéndice A.

1.- Listar los pasos comunes a la instalación de todas las distribuciones

2.- Listar la información básica del hardware que debes tener antes de comenzar a instalar

3.- ¿En cuál partición se coloca el Kernel de GNU/Linux y archivos esenciales?

4.- ¿Puedes tener además de GNU/Linux, otro sistema Operativo en un mismo computador?

5.- Liste las diferente media de la cual puedes instalar GNU/Linux

INTRODUCCION

En éste capítulo analizaremos la instalación de GNU/Linux y los pasos preparatorios para las principales distribuciones de GNU/Linux. Antes de comenzar una instalación, usted necesita estar enterado de tales cosas como espacio disponible en disco, opciones de configuración, y paquetes a instalar.

La instalación de GNU/Linux es diferente a la de otros sistemas operativos, y los nuevos usuarios al sistema operativo pueden encontrar la instalación uno de los obstáculos más grande a superar. Sin embargo, como las distribuciones de GNU/Linux se encuentran constantemente en competencia para lograr que la instalación sea cada día menos traumática, la mayoría de las importantes distribuciones emplean herramientas (tales como GUIs y los denominados Live-CD) para hacer el proceso más simple..

Pasos Comunes

Cada distribución de GNU/Linux nos presenta diferente métodos para lograr su instalación, pero existen un sin número de pasos que son básicamente los mismos en todas e inclusive hasta en el mismo orden:

Configuración Principal	Particiones y Manejadores de arranque
	Selección de media para usar durante la instalación
Pre-configuración	Selección de paquetes o grupos de paquetes
Instalación	Del kernel, Utilidades básicas, y paquetes seleccionados
Post-configuración	Definición de la contraseña del superusuario
	Crear cuentas y grupos de usuarios adicionales y opcional
	Configuración de información básica, Ej.: zonas de tiempo

Mejoramiento

En los últimos años los instaladores de GNU/Linux ha mejorado sustancialmente. La gran mayoría han migrado a instalaciones guiadas por menús y interfaces gráficos. Algunas de las características comunes a las distribuciones modernas incluyen:

- El uso de Interfaces Gráficas Asistidas al Usuario (AGUI)
- Preguntas simples de la configuración con respuestas comunes ya por defecto
- Auto detección de hardware del sistema
- Las varias distribuciones de GNU/Linux adoptan mecanismos diversos para instalar software, pero todos los mecanismos siguen el mismo procedimiento básico.
- Algunos sistemas de GNU/Linux requieren un alto nivel del conocimiento para instalar el sistema; afortunadamente, éstos sistemas son pocos y han ido mejorando con tiempo.

Conocimiento

Usted necesita cierto nivel de conocimiento básico antes de que pueda hacer una instalación. Una comprensión básica del hardware de su sistema es importante. La mayoría de las preguntas se pueden responder siguiendo la guía de la instalación de las distribuciones. Los siguientes asuntos serán discutidos en esta sección:

- Conocimiento de Informática General
- Conocimiento un poco Más avanzado

Conocimiento De Informática General

Generalmente se asume que usted posee cierto nivel básico del hardware, entrar y salir de un sistema GNU/Linux, y la jerarquía y estructura del sistema de archivos GNU/Linux. La mayoría de los

paquetes de instalación de las distribuciones de GNU/Linux intentan hacerlo fácil haciendo algunas preguntas y ejecutándole la mayoría del trabajo.

Conocimiento Básico del Hardware

Algunos pasos de la instalación requieren conocer el hardware específico de su sistema.

- Arquitectura del Sistema
- Capacidad de Almacenamiento (IDE, SCSI, USB, y sus parámetros)
- Fabricantes y modelos de sus periféricos (video, sonido, etc.)
- Adaptador de Red (Network, NIC) y/o configuración de su modem

Contabilidad de Sistema Multiusuarios

Hay varias características de los sistemas multiusuarios.

- Los sistemas multiusuarios identifican a los usuarios por el proceso de Login/Password.
- La cuenta del usuario root o superusuario tiene acceso completo a todo el sistema.
- Las áreas de trabajo individuales son protegidas por cuentas de usuarios con acceso limitado.

Jerarquía y Estructura del Sistema de Archivos

El proceso de instalación le permitirá elegir opciones con respecto a la distribución del sistema de archivos.

- Información básica de particionar
- Disposición del sistema de Archivos de GNU/Linux
- Referenciar Dispositivos del /dev

Cubriremos la estructura, navegación, y la manipulación del sistema de archivos más adelante.

Conocimiento un poco más Avanzado

Si su instalación se le dificulta, tal vez tengas que abandonar los menús y la interfaz gráfica y trabajar desde la Línea de Comandos (CLI) para instalar su sistema. Tendrás que saber como ingresar al sistema, como salir, navegar, ejecutar comandos, encontrar ayuda (man, info, etc.), y editar archivos para así poder asistir al instalador y terminar la instalación.

Cualquier fase de la localización de averías será menos agotadora si usted domina los comandos básicos necesarios para la manipulación de archivos de texto. Sólo si enfrentas problemas durante la instalación tendrá usted que ocuparse de estas tareas. Con una instalación sin problemas, usted no necesitará ningún conocimiento en las siguientes áreas (que se cubren más adelante):

Operaciones y Manipulación Básica de Archivos

El proceso de ejecutar comandos se hace desde el CLI utilizando palabras reservadas del sistema operativo GNU/Linux. Los comandos se pueden clasificar en varias categorías y entre las palabras se incluyen en estas:

- Básicos: cp, mv, ls, more, less, cd, pwd, tar, find, etc.
- Filtros: cat, grep, wc, tail, head, sort, etc.
- Comodines: *,?,[], etc.

Edición Básica de Archivos

- Uso básico de editores de texto
- Abrir, cerrar, escribir, y abandonar editar archivos
- Edición básica de texto

Preparación

Antes de comenzar la instalación, debes leer todas las instrucciones. La primera vez un poco rápido y las veces subsecuentes tratar de adquirir un entendimiento de que puede ser imprescindible para su caso en particular. No apresurarse a la instalación de un nuevo sistema si usted nunca lo ha hecho antes. Tomarse su tiempo, leer las instrucciones de los manuales y de instalación y analizar lo que ha aprendido. Ahora los pasos para la asignación de disco en esta sección.

Pasos para la Asignación de Espacio en Disco

Debes tomar en cuenta para que éste sistema GNU/Linux se utilizará y quien se desempeñará en el. Hay muchos pasos necesarios antes de que usted comience su instalación de GNU/Linux.

1.- Seleccione la Distribución Apropiada de GNU/Linux

Desearas que la distribución de GNU/Linux que elija llene sus necesidades. Para lograr esto, tendrá que primero considerar las distribuciones de alto volumen. Todas son lo mismo en el núcleo, todas utilizan el kernel de Linux, pero si hay variaciones substanciales entre ellas con respecto a otros aspectos de configuración y estrategica manera de hacer las cosas.

El manejador de Paquetes es a menudo en lo que una distribución se diferencia de otra. Estas herramientas asisten a instalar, actualizar, o remover paquetes de software. El manejador de paquetes de RedHat (RPM) es el más comúnmente utilizado entre las distribuciones de GNU/Linux. Debian utiliza su propio manejador llamado dpkg y la utilidad de instalación y actualización llamada apt.

Existen más de 1600 distribuciones de GNU/Linux diferente. Cada distribución tiene sus rutinas de instalación, manejador de paquetes, documentación específica, y una comunidad de soporte. Donde ellas difieren esencialmente es en su visión y características particulares para lograr tareas específicas. Las distribuciones principales incluyen:

Debian	Desarrollada por voluntarios y enfocada en la utilización de software verdaderamente OpenSource, fácil mantenimiento de los paquetes y actualización del sistema.
RedHat	La distribución más conocida y utilizada de GNU/Linux; su venta es manejada por una entidad comercial focalizada en soporte y compatibilidad.
Slackware	Una de las primeras distribuciones; conocida por el poco espacio que utiliza del disco duro, es posible hasta instalaciones en Iomega Zip drives. No es basada ni en RPM o DPKG si no los tarballs tar.gz
Gentoo	Una distribución, tal vez parezca incoherente, poco conocida pero muy usada conocida por ser una distribución que es totalmente compilada paquete a paquete. No es basada ni en RPM, DPKG ni tarballs tar.gz, famosa por método de instalación de emerge....XXXXXX
SuSE	Distribución Alemana-Europea basada en RPM con orientación comercial y muy buen soporte, y versiones para una gran variedad de arquitectura.
Turbolinux	Distribución del Pacifico basada en RPM, acompañada de soluciones propietarias de high-end clustering (Nodos de Super Computadores) además de distribuir versiones de Servers y Estación de Trabajo (Workstation).

2.- Determinar Tipo de Funcionalidad

Es importante considerar en que funciones se desempeñará su sistema GNU/Linux. Esto dictará que paquetes serán instalados y afectará el espacio en disco requerido para la instalación. Podemos clasificar los tipos de instalación así:

- Estación de Trabajo con Network (Network Workstation)
- Estación de Desarrollador (Development Workstation)
- Servidor de Internet (Internet Server)

3.- Inventario del Hardware

Es importante que conozcas el hardware con el cual estas trabajando porque el sistema requiere instrucciones específicas para poder interactuar con los diferentes modelos y marcas de periféricos.

Como Descubrir su Hardware

Las mayorías de las instalaciones proveen guías del tipo lista o hoja de tarea para así poder medir el progreso de la instalación. Puedes accesar esta información:

- Inspección Interna
- Utilizar otro OS para realizar el inventario
- Ejecutar software que tratan de identificar hardware durante la instalación.

Debes mantener esta información cerca de la maquina porque pueda ser que la necesites más adelante si surgen problemas, si deseas actualizar la base de tu sistema, o necesitas re-instalar o reconstruir el kernel de GNU/Linux para incluirle la capacidad que no tienes ahora incluido en el actual.

3.- Documentar la Instalación

Un diario es una herramienta indispensable para un administrador. Este diario puede ser utilizado para asentar las notas de las configuraciones de los dispositivos y las aplicaciones; estas notas conforman un inventario y ruta a seguir para cualquier otra instalación. Este diario almacena todos los eventos importantes de cada instalación y reconfiguración, trivial o importante, en la cual se involucro durante la instalación o reconfiguración. Este diario es un buen sitio donde almacenar cosas importantes como donde compro el ítem, números de modelo, licencias, y todas las cosas importantes referente al mismo.

4.- Planificar como Particionar el Disco

Había una vez que para instalar GNU/Linux había que planificar como se particionaría el disco ya que el sistema operativo requería ciertas particiones para su funcionamiento; ya esto no es así. Esta planificación es sólo necesaria para los administradores de sistemas y los usuarios avanzados. Las mayorías de las distribuciones modernas particionan ellas mismas con algoritmos lógicos del mejor aprovechamiento del espacio disponible. Si le interesa algún tipo de arranque dual de dos o más sistemas operativos desde un sólo equipo entonces debes planificar para no ir a cometer un error y destruir la partición del otro(s) sistema operativo. Si no estas seguro de cómo particionar entonces el mejor consejo es el default de su distribución.

5.- Determinar si su Hardware es Soportada

Aunque la vasta mayoría del hardware moderno es soportada por las distribuciones modernas, aún así debes documentarte lo más posible, debes revisar la documentación de su distribución y la Lista de Compatibilidad de Hardware de Linux (HCL) para estar más seguro de todo.

6. Determinar Conectividad

Necesitará cierta información básica de su red si es que deseas tener cierto tipo de comunicación entre los equipos, ya sea de intercambio de archivos, datos, o recursos; éste paso, claro es obligatorio si su instalación es vía los protocolos de redes.

• Dirección IP (IP Address)

Es un número único e irrepetible con el cual se identifica en que red se encuentra un host. La IPes la dirección numérica de las máquinas en Internet. Este número es asignado por el administrador del sistema, para facilitarle el trabajo al administrador cuando la red es grande se utiliza el DHCP, y para no tener que recordar los números de IPse usan los DNS. Tienen el formato de la forma siguiente a.b.c.d, donde cada letra representa un número decimal en el rango entre el 1 al 254, y los puntos actúan sólo como separadores entre los cuatro números decimales. Note que las direcciones de host 0 y 255 no son permitidas.

• Mascara de Red (Network Mask)

La mascara tiene el mismo formato que las direcciones de IPexcepto que los Bits Menos Significativos son ceros (Least Significant Bits, LSBs) y representan el tamaño de la subred a la cual el host pertenece o que esta conectado. Por ejemplo, a.0.0.0 es una subred con capacidad de alojar unas 16,777,214 computadoras.

El patrón a.b.0.0 representa una subred con capacidad de alojar unas 65,534 computadoras. Finalmente, la subred a.b.c.0 representa una subred con capacidad de alojar unas 254 computadoras. Otras posibilidades existen cuando se crean subnets. Redes Internas (LANs) y proveedores de Internet con Digital Subscribe Line (DSL) son otros ejemplos donde usted podría encontrar subnets del tipo a.b.c.240.

• Pasarelas (Gateway)

Una pasarela (gateway, router, bridge, o firewall) es la dirección IP del host que acepta nuestros mensajes dirigidos al internet o redes fuera de nuestra subnet.

• Servidor de Nombres de Dominio (Domain Name System, DNS)

DNS se refiere a la dirección IP del host que acepta un nombre de dominio (una cadena ASCII) y retorna su dirección IP o vice-versa.

7. Decidir el Método de Instalación

Revise la documentación de su distribución para ver que tipos de media ella soporta para ejecutar su instalación. Debe ser una de las siguientes:

- CD-ROM
- File Transfer Protocol (FTP) o Hypertext Transfer Protocol (HTTP) (vía network o dial-up)
- Network File System (NFS)
- PCMCIA (PC Card) Ethernet , PCMCIA (PC Card) CD-ROM
- Floppy disk
- Tape (Cintas)
- UMSDOS desde una partición MS-DOS

Asignación del Espacio en Disco

En la mayoría de los casos usted podrá aceptar los parámetros por defecto de configuración de su distribución a instalar, esto incluye la planificación del uso del disco duro. Para controlar el uso del sistema, tendrás que asignar espacio en su disco duro. tendrás que determinar la configuración del sistema basada en la siguiente:

- Cantidad de Memoria (afecta su memoria swap)
- Cantidad de espacio en disco duro disponible
- Decidir que opciones de software son requeridas por determinación del espacio requerido para las aplicaciones y los paquetes que deseas instalar.
- Asignar espacio adecuado para las particiones root y swap durante la instalación es de suma importancia ya que requerirá aplicaciones de terceros para poder reparticionar una vez haya instalado.

El Directorio Boot

Algunas distribuciones GNU/Linux utilizan o permiten la designación de un directorio /boot, el cual en disco grandes es a menudo colocado en una partición separada cerca o al principio del disco. Este directorio /boot incluye los siguiente:

- Kernel
- System map
- Archivos de Arranque

Versiones anteriores de los cargadores de GNU/Linux no podían accesar data más allá del cilindro 1,024th, y por esto requerían que el directorio /boot tuviese colocado en la partición por debajo del cilindro 1,024th en los discos grandes. La mayoría de las distribuciones de GNU/Linux utilizan los cargadores actualizados, así es que colocar la partición /boot por separado ya no es necesario.

Métodos de Instalación

La gran mayoría de las distribuciones de GNU/Linux han hecho que la instalación sea un proceso muy flexible, permitiendo que se inicie el proceso de varias formas y de medios de instalación diferentes. En esta sección, analizaremos los métodos de instalación, arrancando desde un CD, disquetes, tarjeta de red y copiando el software a un disco duro. Estos tópicos incluyen:

- Métodos de Arranque
- Medios de Instalación
- Manejadores de Arranque
- Arranque Multiples (Multiboot)

Métodos de Arranque

Existen varios métodos de iniciar una instalación de GNU/Linux. Usted puede arrancar desde un disquetes, desde un CD-ROM, ejecutar un programa de instalación desde un sistema operativo diferente, etc. Analizaremos cada uno de éstos métodos. También es posible iniciar desde un disco Zip y desde una tarjeta de red, pero esos métodos son menos común y no lo tocaremos con gran detalle.

Arrancando la instalación de GNU/Linux desde un disquetes funciona igual que iniciando de cualquier disquetes de arranque. El BIOS lee el sector de arranque desde el disquetes y ejecuta las instrucciones allí localizadas. El sector de arranque carga el kernel de GNU/Linux con un pequeño sistema de archivos que contienen el programa de instalación y inicia el proceso de instalación.

Las PCs de hoy (desde aproximadamente 1997) permiten que iniciemos desde el CD-ROM. El CD-ROM debe contener la imagen apropiada para poder iniciar el PC, así parecida a la del disquetes. Tal vez tengas que habilitar esta opción de arranque desde el CD-ROM en el BIOS. Los BIOS modernos soportan el arranque desde varios dispositivos, los cuales incluyen CD-ROM, Disco duros, disquetes, discos USB, etc. Una nota importante es comentar en la manera que un CD-ROM arranca un PC es que pretende que una parte del CD es un disquetes. Esto nos dice que aunque el sistema inicio desde el CD-ROM, el sistema tendrá que ins-

talar un manejador para accesar el CD-ROM. Ya que la gran mayoría de los CDs de GNU/Linux de instalación son auto arrancables, deberás removerlo desde las bahías del cdrom para evitar iniciar con ellos al inicio del sistema.

Algunas distribuciones permitian iniciar la instalación desde otro sistema operativo, casi siempre desde el cual se ejecuta el programa de instalación.

Medios de Instalación

Una vez el programa de instalación se ha cargado, necesitaras señalarle donde se encuentran los archivos de instalación, se le presentaran varias opciones desde la cual es posible instalar GNU/Linux. Están disponible varios tipos de medios locales y protocolos de redes también. Note que de donde se instalara no necesariamente tiene que ser de donde se inicio la instalación o sea de donde se arranco el sistema. A menudo se inicia desde un disquete pero se instala desde CDs o desde un FTPen la red.

Media Local

Los medios locales de instalaciones más comunes son desde un CD-ROM, floppy disks, o archivos copiados a un disco duro. Aunque la instalación de GNU/Linux fué muy popular una vez, las distribuciones de GNU/Linux de hoy son muy grandes para esto ser factible. Existen aún unas cuantas distros pequeñas de uso específicos que permiten aún su instalación desde disquetes, la última distribución en soportar esta opción es Slackware. Una de las distribuciones más populares que funcionan desde un disquetes son las GNU/Linux Router y las de FireWalls, las cuales aprovechan para convertir PCs ya caducadas y las convierten en un router.

La gran mayoría instalan GNU/Linux desde un CD-ROM. Aunque puedes iniciar desde un disquetes o desde algunos sistemas operativos, lo común es desde el CD-ROM. Todos éstos métodos son factibles, una vez ya iniciado el proceso en realidad no hay ninguna diferencia en lo que es el resto de la instalación.

Es posible también instalar GNU/Linux desde archivos copiados a un disco duro local. Normalmente éste método es utilizado cuando su PC no tiene un CD-ROM o no es reconocido por el kernel, lo cual ya no es muy común. El proceso involucra copiar los archivos desde otro sistema operativo al disco local, entonces iniciar desde un disquetes y decirle al instalador donde encontrará los archivos de instalación.

Instalación Vía la Red (Network)

En grandes empresas instalar GNU/Linux puede que sea el mejor de los casos. Puedes crear un disquetes de arranque e ir PC por PC iniciando el proceso de instalación. Casi todas la instalaciones vía Network son iniciadas desde un disquetes. Es posible iniciar desde un CD-ROM.

Existen varios mecanismos para extraer los archivos de instalación a través de la red. Los protocolos más utilizados son NFS, HTTP, y FTP. Algunas distribuciones soportar el protocolo Samba de redes, Red Hat recientemente retiro soporte vía éste protocolo, lo cual casi siempre es una indicación de que éste protocolo no seguirá siendo soportado. Cualquier método a utilizar necesitarás un servidor que le sirva los archivos necesarios para la instalación.

El programa de instalación necesita configurar su capacidad de comunicarse vía redes para poder iniciar y culminar el proceso de instalación. Necesitas toda la información necesaria de la red para poder lograr el objetivo de instalar vía la red. Puedes adquirir la mayoría de esta información si existe un servidor de DHCP (Dynamic Host Configuration Protocol) en la red, la mayoría de instalaciones soportar DHCPy el servidor es muy común en redes corporativas. Otras variantes que debes estar informado es sobre políticas de firewalls

y proxies, por ejemplo Debian y Red Hat son capaces de cargar archivos de instalación utilizando protocolos HTTPy FTPdesde el internet, aunque se encuentre detrás de un servidor a proxy. NFS no pude ser accesada si estas detrás de un proxy, y por esto servidores NFS deberán estar en el mismo lado del firewall que el cliente a instalar.

Instalaciones Automatizadas

Otra herramienta de instalación disponible para instalar desde una red es las instalaciones automatizadas. Estas herramientas en verdad son muy ponderosas y administradores encargados de redes con muchas PC deben estudiarlas a fondo. Este método provee una manera de instalar en varios equipos sin tener que ingresar ninguna o muy poca información en el cliente. El programa Kickstart de RedHat es uno de estas utilidades poderosas. Crearás un disquetes de inicio y editarás las configuraciones que deseas que los clientes resulten tener después del proceso. Puedes dejar cualquier información que deseas ser preguntado al momento de instalación y así podrás cambiar éstos parámetros de computador a computador. El disquetes de Kickstart puede ser utilizado con un CD-ROM o instalación vía el network.

Gestores de Arranque

Durante el proceso de instalación, debió haber configurado el sistema a que inicie GNU/Linux, si tienes más de un sistema operativo instalado aparecerá un menú de elección. Al arranque el BIOS lee el sector de arranque del disco y lo ejecuta. GNU/Linux instala el código al sector de arranque que carga el kernel GNU/Linux y inicia el sistema operativo.

Existen dos gestores de arranque que vienen con las distribuciones, éstos son LILO y GRUB. LILO significa Linux Loader (Cargador de Linux) y GRUB (Grand Unified Boot Loader); se pueden utilizar para iniciar desde disquetes o los discos duros. Sus archivos de configuración son fácil de editar con editores de textos como vi o pico. Estos archivos son /etc/lilo.conf y el de GRUB es /boot/grub/menú.lst. El proceso de instalación de GNU/Linux provee una oportunidad de instalar uno de los gestores de arranque.

Existen otros gestores de arranque disponibles para GNU/Linux. El programa o la utilidad Loadlin se utiliza para arrancar GNU/Linux desde un prompt del DOS. Una ventaja de Loadlin versus LILO o GRUB es que el no sobre escribe el sector de boot del disco duro. El cargador syslinux es comúnmente utilizado en los arranque de disquete. El Grub es el más nuevo de todos y soporta otros sistemas operativos libres, como lo es el GNU/Hurd. Otro que vale la pena mencionar es el llamado (Extended Operating System Loader) XOSL. Aún no es utilizado por ninguna distribución pero si es muy atrevido en sus GUIs y magnifico soporte para todos los sistemas operativos.

Estos gestores de arranque que aquí hemos mencionado hasta ahora son para equipos de arquitecturas basadas Intel y sus compatibles. Otras arquitecturas tienen sus propios manejadores de arranque. El proceso de arranque es diferente de arquitectura a arquitectura. Una vez el kernel es iniciado el proceso de arranque en las diferentes arquitecturas converge. Las estaciones de trabajo SPARC utilizan el gestor de arranque que es un variante de LILO llamado SILO, las Alphas usan uno llamado MILO. PowerMacs usan un programa llamado BootX o Yaboot.

Multi Arranque (Multibooting)

Diferentes sistemas operativos pueden coexistir en distintas particiones de un disco duro. Es posible iniciar cada uno de los sistemas operativos por separado para aprovechar utilizar aplicaciones únicas al sistema operativo particular. Esta capacidad es referida como multibooting. Por ejemplo es posible tener en un sólo disco duro con FreeBSD u otro sistema operativo y GNU/Linux.

Uno de los problemas de crear sistemas de Multi-Arranque es que algunos sistemas operativos asumen que ellos son el único sistema operativo en la PC. Por ejemplo si instalas algunos de éstos sistema operativos incompatibles después de GNU/Linux, sobrescribirán el gestor de arranque de GNU/Linux con el de ellos, que no es capaz de iniciar otro sistema operativo. Cuando esto sucede tendrás que iniciar su sistema GNU/Linux desde disquetes de rescate y/o CD-ROM y reinstalar el gestor de arranque. Esto no es así con los gestores de arranque de GNU/Linux. Lograr que OS/2 u otro de esstos sistemas operativos incompatbles convivan con GNU/Linux utilizando uno de éstos es casi imposible. Sus gestores de arranque son más complejos y difícil de integrar con otros sistemas operativos. Existen un sin número de how-to sobre éste tópico de dual boot.

Particionando

Los discos no son más que superficies que giran cubiertas con una capa magnética. Estos tienen mecanismos que mueven las cabezas magnéticas cerca de la superficie. Variando las corrientes electrónicas, las cabezas pueden magnetizar senales en la superficie giratoria. después las cabezas pueden leer la dirección de las senales magnéticas. La localización de las particiones y del sistema de archivos es abstracto, éstos son los responsables del manejo procesos de software. En esta sección se cubrirán los siguientes tópicos.

- Terminología de Disco
- Redimencionar las particiones existentes.
- Manejando Discos Duros

Terminología de Disco

Un disco es una colección de arreglos de bloques de datos en anillos concéntricos. La cabeza del disco se mueve en forma de un antiguo toca discos sobre la superficie del disco. Una vez la cabeza esta posicionada sobre una pista del disco, esta lee los bloques de datos mientras el disco se mantiene en movimiento. El acceso de velocidad de disco es una función de el tiempo tomado para mover la cabeza de lectura (seek time) y la velocidad de rotación del disco (latency).

Para ayudar en el manejo de los discos, la superficie puede ser dividida en distintas áreas lógicas llamadas particiones. Las particiones también pueden ser referidas como slice, como los de un bizcocho. Normalmente cada partición ocupa un número de cilindros en el disco. Las particiones permiten que los sistemas operativos accesen un disco duro como un número de discos lógicos por separados. El proceso de escribir un sistema de archivos a una partición es llamado dar formato/formatting.

La información de la partición es codificada en el disco y puede ser cambiada a acorde con los requerimientos del sistema. Bajo condiciones normales reparticionar un disco significara perdida de todos los datos contenidos en las particiones afectadas; particiones no alteradas conservarían toda su información intacta.

Sistema de Archivos

El diseño y estructura del sistema de archivos jerárquico utilizado por GNU/Linux tienen un importante impacto en como usted interactúa con los dispositivos de almacenaje, como son los discos duros. Es importante destacar que GNU/Linux cree que todo es un archivo- hasta los dispositivos son accesado a través de el nombre de un archivo en el directorio /dev (dev abreviatura de devices). Aquí algunos subdirectorios del directorio raíz (root, la barra) / que son necesarios al momento de inicialización del sistema:

/etc	Archivos de configuración esenciales
/bin	Archivos ejecutables binarios del usuario
/sbin	Archivos ejecutables binarios del sistema(utilitarios)
/lib	Contiene las librerías requeridas por /bin y /sbin

/dev Archivos de los dispositivos

Más adelante cubriremos el sistema a de archivos.

Tipos de Disco

Subsistemas de discos (IDE y SCSI) son referidos por archivos en el directorio /dev con nombres muy estandarizados:

- /dev/hda Master primario IDE (Integrated Drive Electronics)
- /dev/hdb IDE esclavo primario
- /dev/hdc IDE master secundario
- /dev/hdd IDE esclavo secundario
- /dev/sda Primer dispositivo SCSI (Small Computer Systems Interface)
- /dev/sdb Segundo dispositivo SCSI

Recuerde que IDE, EIDE, y ATAson diferente nombres para misma intefase. El verdadero nombre de esta especificación es ATA(ATAttachment), pero es más comúnmente conocida como IDE. El nombre de EIDE fué usado para especificar una versión mejorada que soporta transferencia de datos más rápidas.

Particiones del Disco

Las particiones de los dispositivos de almacenar son nombradas utilizando la combinación de su dispositivo seguido por un número . Para poder utilizar más de cuatro particiones en un dispositivo, una de las particiones tiene que ser marcada extendida/extended. Luego en esta partición extendida podemos agregar todas las particiones que necesitemos; por ejemplo:

- /dev/hda 1 Primaria
- /dev/hda2 Primaria
- /dev/hda3 Primaria
- /dev/hda4 Extendida
- /dev/hda5 Lógica
- /dev/hda6 Lógica
- /dev/hda7 Lógica

El número límite de las particiones que pueden ser creadas depende del tipo de particiones que se creen. Generalmente GNU/Linux puede acomodar cuatro particiones primarias. De las cuatro particiones primaria una puede ser extendida. Dentro de esta partición extendida, se pueden crear todas las particiones lógicas que se necesiten.

Los sistemas operativos manejan las particiones de distintas maneras. En otros sistema operativos, cada partición se comporta como un disco duro diferente, y establece una jerarquía independiente de la otra. En sistemas GNU/Linux, sólo existe una jerarquía en la estructura del sistema de archivos. Un directorio es asignado como el punto de montar la partición dentro del sistema de archivos. Las particiones individuales pueden ser montadas en puntos de montajes específicos.

Recuerde que arquitecturas de hardware tienen esquemas diferentes de particionar. Aquí sólo describimos los esquemas aplicables a arquitecturas compatibles con la plataforma Intel. En otros sistemas puede ser que difiera.

Redimensionar Particiones ya Existentes

Utilitarios especializados como es Qparted, PartionMagic y FIPS, pueden redimensionar particiones existentes, con muy poco riesgo de perdida de data.

Qparted al igual que Partition Magic son productos comerciales pero tradicionalmente han sido incluido en distros como Mandrake, Knoppix entre otras con licenciamiento de licencia única (single-licensed) de una sola maquina. Tiene una interfáz completamente gráfica de apuntar-y-dar-click. Pueden dinámicamente redimensionar FAT, FAT32, NTFS, GNU/Linux ext2, ext3, y otros sistemas de archivos.

FIPS es de licencia GPL que se ejecuta desde la línea de comandos también capaz de redimensionar su disco. Diferente a Qparted, no poseé una interfáz de usuario gráfica. Utiliza una interfáz de dialogo y respuestas y un sistema de ayuda en línea. Es completamente libre y nos permite redimensionar particiones con sistemas de archivos FAT y FAT32 . FIPS no puede particionar GNU/Linux o NTFS.

Administrar Discos Duro

Crear y eliminar particiones puede ser efectuado con los utilitarios pdisk o cfidisk. Algunas distribuciones integran sus propias herramientas de particionar. Existen varios interfaces de front-end ejecutable bajo línea de comandos como son el YaST de SuSE (Yet Another System Tool).

También hay otros programas que si tienen interfaces gráficas como es el Disk Druid que se encuentra en las distribuciones de Red Hat y Fedora. Esta te permite crear particiones, ajustar sus tamaños, y designar sus puntos de montaje.

Usar el fdisk

El fdisk es un utilitario que se ejecuta desde la línea de comandos. Disponible durante y después de la instalación de un sistema GNU/Linux. Ni muestra ni permite la asignación de puntos de montaje. Mostramos un ejemplo ejecutando el comando fdisk en un sistema instalado y configurado en inglés:

```
[root@abiertos.org /root]# fdisk
Using /dev/hda as default device!

The number of cylinders for this disk is set to 1048.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
software that runs at boot time (e.g.. LILO)
booting and partitioning software from other OSes
(e'.g. . DOS FDISK, OS/2 FDISK)
Command (m for help):
```

Type the letter "m" to show that fdisk has the following commands available for partitioning a hard drive under Linux:

```
toggle a bootable flag
edit bsd disklabel
toggle the dos compatibility flag
delete a partition
list known partition types
print this menu
add a new partition
create a new empty DOS partition table
print the partition table
quit without saving changes
create a new empty Sun disklabel
change a partition's system id
```

change display/entry units
verify the partition table
write table to disk and exit
extra functionality (experts only)
Command (m for help):

By entering “x”, an additional expert-level menu will be produced.

Command (m for help): x

Expert command (in for help): in

Command	**action**
b	move beginning of data in a partition
c	change number of cylinders
d	print the raw data in the partition table
e	list extended partitions
g	create an IRIX partition table
h	change number of heads
m	print this menu
p	print the partition table
q	quit without saving changes
r	return to main menu
s	change number of sectors
v	verify the partition table
w	write table to disk and exit

Expert command (m for help):

Los comandos que usted debe familiarizase incluyen p para escribir a pantalla la información de las particiones, n para crear una partición nueva, t para asignar el tipo (82 y 83 son usados en GNU/Linux), a para activar y desactivar el estatus de arranque/boot, y w para almacenar los cambios que ha efectuado al disco.

Si en cualquier momento detecta que ha cometido un error, simplemente pulse q para salir sin guardar los cambios. Asegúrese de escribir los cambios a pantalla y analizarlos para observar que todos los cambios a efectuar están correctos antes de guardar permanentemente los cambios.

Usar el cfdisk

El utilitario de GNU/Linux cfdisk es una aplicación manejada por menu desde la línea de comandos. Su pantalla principal esta dividida en tres secciones. Cada parte de la pantalla esta descripta en la secciones que continúan:

- Resumen de los Discos
- Particiones del Disco Actual
- Comandos

Resumen de los Discos

La primera sección del comando cfdisk brinda información acerca del disco del sistema seleccionado actualmente. Esta información incluye el nombre del disco seleccionado (ejemplo, hda o scda) y el número de cabezas/heads, sectores por pistas, y el número de cilindros que contiene el disco.

Particiones del Disco Actual

La sección de la pantalla colocada en el medio nos da la información de las particiones actual en el disco seleccionado en su sistema. Use las teclas cursoras para subir y bajar por la lista. Cada línea de esta lista nos da información para la partición del disco indicado y contiene los siguientes campos:

Name	El nombre del disco que contiene la partición, por ejemplo heda1.
Flags	Indica si la partición es arrancable/bootable. Cuando el campo no esta marcable no lo es.
Part	Type Indica si la partición fué creada como primaria o lógica.
FS Type	El tipo de sistema de archivos en la partición, como es Linux Nativo o Swap.
[Label]	Usualmente usado en sistemas DOS.
Size (MB)	Espacio asignado a la partición.

Comandos

Ultima sección de la pantalla del cfdisk contiene el menu usado para lograr las tareas de particionar. Para seleccionar un comando, use las teclas cursos para moverse entre las opciones. Para ejecutar una de las acciones coloquese sobre ella, y presione ENTER.

Copiar Software Después de instalar un sistema base de GNU/Linux, paquetes adicionales pueden ser cargados para agregar funcionalidad. Componentes comunes como son los navegadores de páginas web por lo general no son incluidos en el sistema base de la distribución. Cubriremos los siguientes tópicos:

- Los Paquetes/Packages y Opciones
- Información adicional de Instalación
- Primera Instalación
- Asignando Particiones al Disco
- Archivo Swap
- Administrador de Paquetes/Package Management

Paquetes y Opciones

Muchas distribuciones hoy día incluyen perfiles de instalaciones por defecto, lo que viene a ser un conjunto de paquetes pre-seleccionado a la medida conforme al uso general del sistema. Al seleccionar por ejemplo, Development workstation o Equipo de Desarrollo, nos incluiría la gran mayoría de herramientas de desarrollo y los lenguajes de programación. La ventaja de esta preselección es que nos evita tener que elegir paquetes individuales, y así nos acelera el proceso de instalación. Debe tener mucho cuidado al elegir una de estas opciones y el monto de espacio que estas ocupan. Si su distribución a instalar no soporta éstos perfiles de instalación entonces tendrá que escoger manualmente.

Aunque no podemos detallar cuales paquetes fueran útiles, si es muy recomendable instalar el servidor X necesario para su tarjeta video y sumarle cualquier herramienta de configuración gráfica que éste disponible que le ayude a adicionar paquetes. Estas herramientas pueden hacer que sea más fácil instalar paquetes adicionales que le pueden ser útiles.

Tipos de Paquetes

Los dos paquetes más usados comúnmente son los RPMs y DEBs (un RPM es manipulado por el RPM y los DEBs por el dpkg). Por su características robustas y naturaleza de ser de fuente abiertas, muchas distribuciones implementan ambos formatos de paquetes. Los RPMs son usados por Red Hat, Mandrake, y SuSE y muchas otras distribuciones. Debian, Knoppix, ubuntu, y algunas otras usan debs. Además de éstos dos existe el TGZ de la distribución Slackware que utiliza éste empaquetado binario para hacer el trabajo de instalación un poco más fácil, pero que realmente no tiene es un sistema de comprobación de dependencias entre paquetes.

El formato estándar para las comunidades de UNiX los derivados de UNiX para los archivos comprimidos (similar a los archivos Zip) es el tarball, el cual es manipulado usando dos utilitarios estándares: tar y gzip. Los Tarballs no poseen la funcionalidad de los sistemas de paquetes y simplemente contienen archivos, pero son utilizados por todas las distribuciones para distribuir los códigos fuentes.

Información Adicional para la Instalación

La mayoría de las preguntas al momento de instalación están relacionadas con el equipo/hard, así pues que debes inspeccionar su sistema completo y periféricos (si es posible debe tener la documentación del equipo a mano durante la instalación). La información de la tarjeta de video y el monitor es esencial para el ajuste del XFree86.

Aquí algunas de las cosas que se le preguntarán:

- Teclado y Lenguaje
- Zona del Tiempo/Time Zone
- Tarjeta de Video
  - o Fabricante y Modelo
  - o Total de Memoria y Chipset Usado
- Monitor
  - o Fabricante y Modelo
  - o Tamaño, Resolución, y Capacidad de Refrescar

Primera Instalación

Si usted no ha instalado una distribución antes de esta, experimentela y acepte todos los valores por defecto que el script de instalación le ofrece. Al final lo más seguro es que usted terminará con un sistema instalado que funciona. Puede ser que no tenga acceso a todo su hardware y que el esquema de particiones no sea el perfecto, pero por lo menos funciona y podrá iniciarse.

Cuente con tener que instalar su sistema por los menos dos veces la primera vez que instale GNU/Linux. La primera vez, simplemente acepte todos los valores por defecto de preguntas que no éste seguro de su respuestas. Arranque su sistema recién instalado y pruébelo para observar su comportamiento. Analice y concentrase en lo que ha aprendido de haberlo instalado esta primera vez. Planifique que durante la segunda instalación se tomará más tiempo se concentrará en como quiere que el sistema sea una vez ya ha terminado de instalar.

Asignar Particiones del Disco

Discos individuales y particiones de discos pueden ser utilizadas efectivamente para incrementar las capacidades de su sistema GNU/Linux. Le puede tomar tiempo en encontrar el balance perfecto entre tamaño del sistema de archivo (tamaño de la partición) versus funcionalidad. Particiones muy grandes desperdician espacio; muy pequeñas previenen que el sistema trabaje efectivamente.

Aquí mostramos una distribución típica. Observe que usted puede combinar algunas de estas particiones en una sola.

/	Lo más pequeña posible (incluye /dev, /etc); asegúrese de dejar suficiente espacio para /tmp cuando se encuentre en modo de usuario único/single
/usr	Por separado; suficientemente grande como se requiera; permita crecer
/boot	partición de Arranque/Boot
/opt	SoftWare de terceros; partición no siempre es usada
/home	Directorios home de los usuarios; tamaño varia dependiendo el número de usuarios
/var	Mail/Correo, impresión, archivos de log, etc.

/tmp	Espacio de archivos temporales
swap	Area de intercambio/Swap de la memoria virtual

La solución más simple es asignar todo el espacio a una sola gran partición en todo el disco. Un buen administrador de sistema usaría particiones para limitar el crecimiento de archivos en áreas de crecimiento peligroso (como pueden ser los directorios home de los usuarios y los archivos de spool). Limitando áreas de crecimiento peligroso previene que se nos desborde el sistema de archivos raíz/root. Si se nos llega a llenar el sistema de archivos root, el sistema GNU/Linux no podrá continuar trabajando y lo más probable es que se paralice, así obligando al administrador tener que reiniciar forzado para reiniciar el usuario único y así poder resolver el problema y corregirlo.

La mayoría de los directorios listados previamente son utilizados para almacenar archivos de data creados por los usuarios y los procesos del sistema. Usted debe proteger el sistema (y los usuarios) de la minoría de los usuarios y servicios que generan un gran monto de data, colocando esos directorios en particiones separadas.

La raíz del sistema de debe tener suficiente espacio para archivos temporales creados durante las sesiones de usuario único cuando otros sistemas de archivos no son montados. Es perfectamente normal montar un sistema de archivos grande en /tmp cuando se esta en modo de multiusuario.

RESUMEN

En este capitulo, cubrimos en detalle la tecnologia de encriptacion. Algunos de las publicaciones claves que discutimos fueron:

- La encriptacion puede ser clasificada como Bloque y torrente de cifras.
- Un cifrado es una implementación de un algoritmo de encriptacion.
- Los tipos de algoritmos de encriptacion se incluyen el simetrico, asimetrico y hash.
- El PGP es un programa de encriptacion ampliamente usado que entrega encriptacion de claves publicas para los usuarios y organizaciones.
- La fuerza de la encriptacion es basada en tres factores: lo secreta que puede ser una clave, Longitud de la clave, decadencia debilidades en algoritmos cifrados o implementaciones.
- La Infraestructura de Clave Publica (PKI) requeire administracion de algunas claves que el esquema simetrico de administracion de claves.
- Los certificados digitales proveen un significado de verificacion de identidad.

Archivo Swap

El archivo swap es espacio en disco que actúa como memoria. Páginas de memoría son intercambiadas/ swapped entre la memoria RAM y el archivo swap en disco. El archivo swap es una partición en disco (raw), la cual no es archivo del sistema. El archivo swap es de tamaño fijo.

Es también posible asignar espacio swap temporario con un archivo swap físico en vez de una partición swap. Este procedimiento del uso del swap es más lento pero puede ser preferible a tener que reinstalar o reiniciar para montar otro disco en el sistema. Para especificar un archivo swap temporario, la siguiente secuencia de comandos puede ser ejecutada:

Asignar un archivo de 10-MB de ceros.

dd if=/dev/zero of=/swap bs-1024 count=10240

Definirlo como un archivo swap.

mkswap /swap 10240

Activarlo como un archivo swap. Empezar a darle uso el archivo swap.

swapon /swap

Desactivar el archivo swap.

swapoff /swap

Remover el archivo swap del sistema.

rm -f /swap

Administradores de Paquetes de Software

Años atrás, actualizar un sistema GNU/Linux era en realidad reemplazar casi todos los paquetes. Hoy en día, los diferentes grupos que desarrollan distribuciones de han automatizado en la manera que las aplicaciones son actualizadas/upgraded. Los archivos necesarios ahora son empaquetados en grupos y las herramientas para poder hacer el proceso de actualización fueron desarrolladas.

A los administradores de paquetes, como RPM/DPKG, se le han desarrollado opciones que funcionen desde la línea de comando (CLI), usando opciones para pasarle los parámetros. Luego se desarrollaron, como ha sido el orden tradicional de GNU/Linux las aplicaciones que son manejadas basadas en menues, como por ejemplo, el Red Hat Package Manager (RPM) primero se desarrolló como un paquete para ejecutarse desde la línea de comandos, y luego evolucionó, GLINT para proveer un interfáz gráfico para asistir al usuario.

Luego GLINT fué reemplazado por la herramienta de administración de paquetes GNOME RPM, kpackage, entre otros que sólo se ejecutan dentro de sus propias distribuciones como es el YaST de SuSE, RPMDRAKE de Mandriva, etc.

Algunas distribuciones de GNU/Linux tienen sus propios mecanismo de manejar empaquetados:

- Red Hat-. Formato rpm desarrolló la herramienta RPM y su interfáz gráfica GLINT
- RPM es distribuido libremente; puede ser utilizado en cualquier sistema GNU/Linux
- Red Hat ha usado varios tipos de administradores de paquetes con interfaces gráficos:
    - GLINT- Herramienta gráfica para el manejo de RPMs de Red Hat (ya no esta en uso)
    - GnoRPM y Kpackage- Reemplazo de Red Hat ha adoptado para el reemplazo de GLINT
- Debian- .deb formato dpkg (Debian package manager)
- Slackware-.tgz, formato comprimido tar herramienta de paquetes (pkgtool)
- SuSE- Usa .rpm con YaST o YaST2 como interfáz gráfica (front end)
- Software disponible desde sitios FTP anónimos, diferentes tipos de paquetes.

En la actualidad, el formato RPM es el más utilizado. Las dos distribuciones más usadas en el mundo comercial son basadas en RPM, SuSE y Red Hat, ambas usan RPM. Los paquetes DEB de Debian ofrecen características también muy avanzadas y también han crecido en popularidad.

Concluyendo la Instalación

Hay unas cuantas tareas que deberá completar una vez todo el software éste instalado. Es necesario que ingrese con la cuenta de root para poder efectuar las siguientes tareas administrativas:

- Agregar cuentas de Usuarios.
- Agregar/actualizar discos, impresoras, y modems.
- Monitorear/mantener/mejorar la seguridad del sistema.
- Monitorear uso y medir el rendimiento para así poder optimizarlo.
- Desarrollar y validar un esquema de backup.
- Resolver problemas de configuración,

- Resolver problemas y conflictos de dispositivos/hardware.

Los siguientes tópicos serán discutidos en esta sección:

- Resolver Problemas de Configuración
- Resolver Asuntos de Hardware

Resolver Problemas de Configuración

Troubleshooting es el proceso de diagnosticar cualquier problema de configuración o de hardware y corregirlo, o sea diagnosticar y corregir. Conocimiento y experiencia son las mejores herramientas. Corazonadas trabajan muy a menudo; la única manera de desarrollarse es practicando, practicando y practicando.

Troubleshooting una instalación fallida es muchas veces mejor empezar de nuevo. Encaminarse en todo el proceso de instalación, leer todos los mensajes incrementa su conocimiento y experiencia. Durante el proceso de instalación, puede observar todo lo que sucede con simplemente cambiar a otra terminal virtual (use ALT+F[1-6]). Las primeras terminales virtual son utilizadas por el proceso de instalación; ellas muestran los comandos que se ejecutan y sus resultados. Ingreso en un terminal virtual que no éste en uso por el proceso de instalación, le permite revisar los directorios ya creados con el comando (ls) el espacio disponible y usado en disco con los comandos (du, df), y usar cualquier otro comando que le pueda ayudar en diagnosticar problemas que puedan surgir.

Resolver Asuntos de Hardware

GNU/Linux puede funcionar sobre una grandísima variedad de hardware, y arquitecturas completas. Con tecnologías anteriores, como un dispositivo que requiere de bus tipo Industry Standard Architecture (ISA), GNU/Linux no siempre puede determinar correctamente el ajuste de ciertos dispositivos. También es cierto que algunos dispositivos son extraños en comportamiento, y requieren del usuario hacer un poco de investigación adicional. Esta sección cubre algunos de los problemas más comunes.

Dispositivos no Soportados

Por el hecho que GNU/Linux no usa el firmware del BIOS, es responsable de conocer los detalles del todo el hardware. Por lo general, no existen problemas con interfaces de hardware bien establecidas, con mucho tiempo ya en el mercado. GNU/Linux tiene más dificultades con dispositivos nuevos en el mercado y que los fabricantes sólo proveen la información a ciertos productores de SO en particular. Ejemplo son dispositivos como impresoras, tarjetas de video, y modems.

Impresoras

En cierto tipos de impresoras de inyección de tintas (inkjets o bubble), se necesita protocolo de comunicación especial entre la impresora y el driver del printer. Muchas impresoras pueden manejar el lenguaje PostScript, pero algunas requieren un protocolo de modo Gráfico. Al menos que éste driver éste disponible para una impresora especifica, usted no podrá usarlo desde GNU/Linux.

Adaptadores de Video

Los adaptadores de video de última versión pueden ser problemáticos. Aunque el proyecto XFree86 soporta un gran número de adaptadores de videos, nuevos son lanzados al mercado a diario, y los revendedores tienden a empaquetar los sistemas nuevos con éstos para abaratar costos. El resultado es que usuarios de GNU/Linux deben asesorarse por avanzado de sus adaptadores de videos y si son soportados en GNU/Linux antes de comprar sus sistemas. La distribución de SuSE mantiene una base de datos de tarjetas

de video y sus manejadores en la dirección web http://cdb.SuSE.de/cdb_english.html, y el sitio del XFree86, http://www.xfree86.org/, documenta las tarjetas soportadas en cada revisión. Algunas de estas tarjetas no soportadas directamente si son soportadas indirectamente con drivers proveídos por los fabricantes como es el caso de NVIDIA, éstos drivers son de fuente cerradas y existen siempre proyectos, incluyendo ingeniería reversa de proveer equivalentes de fuente abiertas y libres.

Los Modems

La inovación del diseño de modem sin controladores, los famosos Winmodems. Estos dispositivos emplean un protocolo especial entre el CPU y el adaptador.

Estos inovadores diseños por lo general utilizan recursos del ciclo del CPU que no son utilizados para llevar a cabo tareas que normalmente son funciones del periférico, el manejador/driver al utilizar recursos del sistema lo cargará más así reduciendo el rendimiento del sistema. Soporte y desarrollo para éstos drivers bajo sistemas GNU/Linux se desarrolla continuamente y fabricantes también cooperan pero no con el mismo entusiasmo como lo hacen para otros SO, el sitio web con mayor información sobre éste tema es http://www.linmodems.org

Posible Dificultades de Adaptadores

La dificultad con el bus ISA es que cada adaptador debe seleccionar un valor para su parámetros de I/O de un pequeño grupo de valores, y dos adaptadores no pueden jamas usar el mismo valor. Aún más, los cuatro parámetros deben ser configurados manualmente. Los cuatro parámetros que deben ser configurados son:

- I/O.- address/dirección; Aunque hay muchas posibles direcciones, el hardware del adaptador sólo permite la selección de unas cuantas direcciones que pueden que caigan en conflicto con otros adaptadores.
- (IRQ).- Interrupt Request; Sólo existen alrededor de cinco líneas de IRQ libre, y puede ser que otro adaptador ya éste configurado para usarla.
- Shared Dirección del área de memoria compartida/shared por el adaptador y el CPU.- Los adaptadores por lo general sólo ofrecen un número limitado de direcciones y ya otros adaptadores pueden estar usando éste rango de direcciones de memoria.
- (DMA).- Canal de Direct Memory Access; Sólo están disponible siete canales DMA, y puede ser que ya otro adaptador alla sido asignado el canal de DMA que el usuario intenta asignar en su configuración. Aunque existen cuatro posible parámetros, solamente tres pueden estar en conflicto porque los adaptadores ISA se comunican con el CPU por vía de la memoria compartida o vía el DMA.

Asuntos de Monitor y Tarjetas de Video

Un cuidado especial debe ser prestado para asegurarse que su tarjeta de video ha sido bien configurada para operar en la manera que fué diseñada para operar con las especificaciones de su monitor. Muy a menudo esto resultará en pantallas de garabatos al intentar ejecutar aplicaciones gráficas. Pero, recuerde que un monitor puede ser permanentemente dañado si es forzado a operar más alla de sus capacidad.

RESUMEN

En éste capítulo se le introdujo a temas relacionados con la instalación de distribuciones de GNU/Linux.

- La instalación de GNU/Linux es algo similar para todas las distribuciones aunque no idéntica.
- La instalación de GNU/Linux se efectúa desde la cuenta de root, el superusuario.
- La instalación de GNU/Linux es un proceso mucho más simple que en tiempos atrás.
- Existen pasos básicos comunes a todas las distribuciones GNU/Linux.
- Distribuciones modernas se instalan a través de un script manejado por menues de preguntas/respuestas.
- Algunas distribuciones pueden autodetectar hardware e configurarse con poca interacción del usuario.
- La mayoría del software esta disponible en los paquetes estándares.
- Particionar los discos esta administrado por cfdisk, el comando fdisk o herramientas similares.

Preguntas Post-Exámen

Las respuestas a estas preguntas estan en el Apendice A.

1.- ¿Donde debe ser instalado el directorio /boot en el disco duro?

2.- ¿Qué herramientas están disponibles para examinar el particionado de un disco?

3.- ¿por qué no debe usted trabajar desde la cuenta de root/superuser para todas sus tareas?

4.- ¿Cuál es la función del sistema de administración de paquetes?

5.- ¿Donde es posible encontrar errores durante la instalación?

CAPITULO 3

ADMINISTRACIÓN DEL SISTEMA OPERATIVO GNU/LINUX

INSTALAR GNU/LINUX

Este capítulo damos una breve explicación de las características de cada distro y sus sistemas de instalación, paquetes, interfaces, aplicaciones por defecto, etc... Trataremos de resaltar porque elegir una distro para una tarea y otra para una diferente. También lo dirigiremos a sus fuentes de información para que usted llegue a sus propias conclusiones de cual distro le conviene y porqué.

Distribuciones

GNU/Linux es un nombre genérico que se le da a todos los sistemas operativos que empleen un kernel Linux. Pero de nada nos serviría un kernel si no contamos con un conjunto de aplicaciones que permitan interactúar con el sistema más amigablemente. Las aplicaciones para Linux se han desarrollado a veces parallelo la mayoría de los casos antes del kernel por numerosos programadores interrelacionados a través del Internet y organizado en su mayoría de los casos por el GNU y RMS (Richard Stallman). De acuerdo a esta situación se generan varias distribuciones de Linux, cada una con sus características particulares.

Básicamente una distribución no es más que la agrupación de un kernel (Linux), y un conjunto de aplicaciones, algunas utilidades y un programa de instalación. Existen alrededor de 1600 distribuciones de GNU/Linux. Algunas son conocidas como mayoritarias pues poseen un desarrollo sostenido e independiente, otras son basadas en las anteriores tomando de estas una parte de sus características agradables y modificando otras. También existen micro-distribuciones que pueden almacenarse en uno o dos disquetes. Entre las distribuciones más conocidas y utilizadas pueden citarse a:

- Red Hat Linux
- Mandrake Linux
- Gentoo
- Debian/GNU Linux
- SuSE Linux
- Slackware

Cual es la mejor, depende de sus necesidades. Todas ellas ofrecen sus ventajas y desventajas particulares. Para más información se puede consultar los siguientes enlaces:

http://www.linux.org/dist/
http://www.nombre-distro.org

CONFIGURAR Y DIAGNÓSTICAR EL X

OBJETIVOS

Al completar este capítulo, usted podrá:

- Describir cinco componentes del Sistema X Window.
- Contraste interpretes de línea de comandos (CLI) vs Interfaces Gráficas de Usuario (GUIs) y sus ventajas y desventajas.
- Configurar los dispositivos de video del X server
- Instalar, configurar, y navegar en dos manejadores del X window.
- Explique la diferencias entre los varios ambiente de escritorio y los manejadores de ventanas.
- Identifique y termine aplicaciones del X que no responden.
- Use X remotamente y limite el acceso al servidor de X.
- Contraste a xf86config, XF86Setup, y el Xconfigurator.
- Describa el role de los administradores de ventanas (windows managers) y compare y contraste los siguientes programas: Fvwm95, AfterStep, Window Maker, IceWM, y Sawmill.
- Describa la implementación de las herramientas de despliegue, como lo son XDM, XWin32, y el VNC de ORL (Virtual Network Computing).
- Liste las secciones del archivo de configuración del Servidor del X.

PREGUNTAS PRE-EXAMEN

Las repuestas se encuentran en el Apéndice A.

1.- ¿Cuál es el comando genérico de iniciar el sistema X?

2.- Cuando hay un problema o duda del administrador de ventanas, es importante revisar ¿cuál recurso de información para encontrar una solución?

3.- ¿En caso de emergencia, ¿Qué combinación de teclas puede ser utilizada para eliminar (kill) el servidor del X?

INTRODUCCION

En éste capítulo cubriremos el Sistema X Window, el GUI utilizado por todas las distribuciones de GNU/Linux. Daremos un vistazo a la estructura del X y varios aspectos que usted puede utilizar para mantener el sistema. En éste capítulo, discutiremos a fondo los componentes que conforman el Sistema X Window. Además, cubriremos como configurar el XFree86, es servidor estándar X usado en GNU/Linux, como ejecutar y usar el sever de X y las aplicaciones del X, los ambientes de escritorio, y usando el X remotamente.

Por último, especificaremos recursos para custumizar las aplicaciones. Antes de continuar si aún no esta familiarizado con los entornos de ambiente de escritorios GNOME y KDE se le aconseja tomar un vistazo y regresar una vez ya éste familiarizado

EL SISTEMA X WINDOW

El Sistema X Window es un sistema de ventanas distribuido por redes que permite ventanas de texto, graficos, o ambos ser desplegado en una pantalla de mapa de bits. La ventana a desplegar puede aparecer en el sistema donde se ejecuta la aplicación o en un sistema diferente en la red. La ventana puede caer sobre otra ventana, ponerse más pequeña, o reducirse al tamaño de un icon, o engrandeser hasta llenar toda la pantalla. Las ventanas pueden ser manipulado dinamicamente para su tamaño y colocación. Algunas pueden ser enrolladas como una cortina. El sistema debe ser referido como el Sistema X Window, El X11, o simplemente como el X. Nunca debemos llamarlo el X Windows.

Un concepto clave a recordar del X es que el define el mecanismo para desplegar graficas a pantalla, pero no define politicas. El permite las capas más alta que definan las politicas, como es el “look and feel” del entorno. Politicas son implementadas por el manejador de ventanas, el toolkit de widgets, y el entorno del escritorio. La arquitectura de separar los mecanismos y las politicas tienen también sus ventajas y desventajas. Aunque provee mucha flexibilidad a los programaadores de aplicaciones, esto puede crear una inconsistencia en las apariencias y el comportamiento de las aplicaciones. Ambientes de escritorios estan siempre trabajando para mejorar esta situación.

En esta sección, discutiremos los siguientes tópicos:
- X –vs- Línea de Comandos
- Servidor X
- Protocolo X y el Xlib
- Toolkits y el conjunto Widget
- Administradores de Ventanas
- Display Managers
- Otros Componentes

X -vs- Línea de Comandos

La interfáz natural de GNU/Linux, la Línea de Comando, fué diseñada para ser un ambiente poderoso del cual el sistema completo puede ser controlado. Puede resultar como un reto la introducción del usuario novato, pero, es visto por usuarios de otros sistemas operaativos como algo intimidante. Su números comandos tienden a ser cortos y hasta cripticos. Los GUIs del Sistema X Window de GNU/Linux resultan más cómodos para éstos usuarios. Había una vez que sólo desde el X no se podía controlar todo el sistema, esto ha cambiado y seguirá mejorando. Distribuciones como SuSE, Redhat y Mandrake han hecho grandes aportes.

Hoy en día tienes un sin número de aplicaciones entre las cuales elegir según sus necesidades de admi-

nistrar su sistema GNU/Linux. Un novato puede administrar un sistema completo sin jamás tener que escribir ordenes desde un comand prompt. Mientras esto facilita la introdución del usuario novato a GNU/Linux, a medida que el desarrola las herramientas de lugar el tiende a alejarse del X y una migración paulatina hacia la Línea de Comados.

Servidor X

El componente principal del Sistema X Window es el Servidor X. Su trabajo es aceptar las peticiones desde los programas cliente y desplegar la información a la pantalla. Este coloca y mantiene los recursos asociados con general las imagenes a desplegar. El servidor puede aceptar peticiones desde más de un cliente simultaneos. Cada cliente tiene una ventana que puede usar para dibujar. Estas ventanas pueden tener ventanas hijas dentro de ellas también. Los clientes pueden tratar a estas ventanas hijas como botones, iconos, menus, o cualquier otra cosa. Es importante recordar que el Servidor X se ejecuta en la maquina que esta usted operando. A diferencia de lo común de cualquier otro servidor, en el cual uno siempre se sienta en el cliente y accesa el servidor remotamente. Lo más importante a recordar es que el Servidor X es quien provee el servicio de desplegar la salida de video.

El Servidor X también maneja los eventos. Eventos son cosas que pasan desde fuentes externas. Los eventos más comunes provienen desde el teclado y el mouse. Los eventos también pueden ser generados cuando una nueva ventana es creada, se le cambia el tamaño, se mueve, se reorganiza, o cuando el puntero de mouse entra o sale de la ventana. El servidor procesa éstos eventos y se los envia a los programas clientes. Los programas cliente por eso son denominados como manejados por eventos; en vez de los programas hacer iteraciones a traves de estructuras de bucles de control, los programas esperan que el Servidor X le envie éstos eventos y entonces actuan sobre éstos. A un nivel un poco más fundamental, el cursor del mouse puede ser cualquier tipo de dispositivo con puntero como los es: trackball, trackpad, tablets graficas, etc. Más en lo adelante, cuando examinemos el archivo de configuración, simplemente nos referiremos a el como el dispositivo puntero.

El Servidor X Estándar de GNU/Linux, y muchos otros sistemas operativos LIBRES, se llama el XFree86. Esta basado en la distribución standar del X11R6 pero a la vez contiene muchas mejoras adicionales. El XFree86 no sólo biene con un Servidor X, sino con muchos otros utilitarios. De las que le pueden resultar muy utiles se encuentran el xterm, xclock, xlogo, xsetroot, y xset.

Dependiendo de la version del XFrec86 que estes usando, puede ser que contenga más de un servidor ejecutable. Si usas el XEree86 3, las mayorías de las tarjetas de video usarían el Servidor de X llamado XF86_SVGA, aunque existen otros servidores específicos a tarjetas de videos particulares que así ofrecen mayor rendimiento. También existe un servidor que usa el frame buffer de GNU/Linux. Originalmente fué desarrollado para Macintosh pre-PowerPC (Motorola 680xOs), que lo más seguro es ha ser usado en equipos como de plataformas como SPARCs y PowerPCs que tradicionalmente tienen una pantalla basada en caracteres. Si usas el XFree86 4, sólo existe un servidor ejecutable. Todas las cuestiones especificas a las tarjetas de videos se encuentran en forma de modulos.

Existen Servidores X disponibles para GNU/Linux y para otros sistemas operativos. La mayoría de los Sistemas comerciales UNiX vienen con X preinstalado. Compañías como Metro Link y Xi Graphics producen servidores para GNU/Linux. Existen algunos paquetes X para otros sistemas operativos; el más popular es Exceed, de la compañía de software Hummingbird. Debido a la arquitectura cliente/servidor del X, usted puede libremente diversificar el uso de más de uno de éstos sevidores con los programas clientes que usas en GNU/Linux.

El Protocolo X y Xlib

Los clientes X se comunican con el servidor usando el Protocolo X. Este protocolo es muy a menudo transportado via TCP/IP pero también puede usar otros canales de comunicación. El protocolo X define un mensage por cada petición posible. Por ejemplo, una petición puede ser crear una nueva ventana y otra puede ser dibujar un circulo en una ventana. El servidor acepta la petición, la proceso, y retorna un mensage indicandole si fué exitoso o no.

El protocolo X es de bajo nivel, y es muy extraño que alguien escribiese un programa usandolo directamente. Para que pueda ser más fácil crear programas usando el X, las peticiones al protocolo X estan disponibles en la libreria C en rutinas llamadas Xlib. Las peticiones a las rutinas del Xlib se mapean casi en su totalidad directamente al protocolo X y así liberan al programador de tener que conocer todos sus detalles. El Xlib maneja las respuestas desde el server para el cliente automáticamente. El Xlib también hace el trabajo de conectar el cliente al servidor via algún canal de comunicación.

Toolkits y Conjunto de Widget

Los programas cliente son comunmente escritos usando el toolkit o el conjunto de widgets. El Toolkit provee otra capa de abstracción sobre el Xlib, así simplificando el monto de información que un cliente necesita para mantener. El Xlib se concentra mayormente en las ventanas, y los toolkits mayormente con los widgets. Existen muchos conjunto de widget y toolkits disponible, los tres toolkits más utilizados en GNU/Linux son Xt, gtk y Qt.

En el X, un widget es una parte de la pantalla que el usuario o el programador pueden usar como objetos independiente. Los Widgets incluyen cosas como los botones, iconos, fotos, areas de entrada de texto, etiquetas, checkboxes, menus, barras de scroll, etc. Todos ellas estan construidas desde ventanas pero con un contexto y apariencia diferente.

Xt, Athena, y Motif

El toolkit Xt es denominado como el X Intrinsics Toolkit. Este viene estándar con el X11. Este provee una herarquía orientada a objetos de widgets que pueden ser organizadas. El toolkit no contiene los widgets. Un conjunto de widget se construye en cima del Xt. Los dos conjunto de widgets principales del Xt son Athena (Xaw) y Motif (Xm). Athena se incluye con la distribución del X. Motif incluye algunos programas además del conjunto de widget. Los widgets usados en Athena no son los más preciosos visualmente.

Lo más seguro que el único programa que usted va a usar que usa los widgets de Athena es el xterm. Motif luce un poco mejor, pero la gran mayoría de los usuarios de GNU/Linux no justifican el costo. Existe un proyecto llamado Lesstif que ha creado un widget compatible con Motif y que puede ser usado en lugar de Motif. Los widgets de gtk y Qt han suplentado completamente el uso del conjunto de widgets Xt.

GTK

El gtk es el Toolkit del GNU, fué originalmente diseñado para el programa de dibujo y manipulación de imagen el GIMP, pero se ha extendido mucho más de ahi. El usa sistema orientado a objeto propio en vez de construir sobre el de Xt. Una de las caracteristicas clave del gtk es de la manera que sus contenedores funcionan. Los contenedores determinan dinámicamente el tamaño de las ventanas padres e hijas, empaquetando la ventana en un grupo bien configurado, el gtk usa una capa llamada glib que provee una interfáz a Xlib pero ayuda a hacer el toolkit más transportable. El widget gtk fué elegido por el proyecto

GNOME y siempre es actualizado activamente, gtk ha sido migrado a otros sistemas operativos, y a BeOS y recientemente al MacOS.

Qt

Qt es un un widget toolkit comercial disponible para varios sabores de UNiX como también está disponible en versiones para otros sistemas operativos. Desarrollado por una compañía independiente llamada Trolltech y está libremente disponible para GNU/Linux. El toolkit de Qt fué escrito en C++ y requiere que los programas sean escritos en C++. Así que, Qt usa el lenguaje orientado a objeto de C++ y sobrepasa el uso de Xt. El Ambiente de Desktop del equipo de desarrollo de KDE que eligieron el conjunto de widgets de Qt para desarrollar su desktop. El equipo también ha creado temas de Qt, permitiendo personalizar el entorno visual.

Administradores de Ventanas (Window Managers)

Para mantener la filosofía de separar mecanismo y política, el X usa un programa por separado llamado el window manager para administrar el colocamiento y la manipulación de las ventanas. El window manager tiene varias responsabilidades:

- Decorar las ventanas con borde y barra de titulo
- Colocar las ventanas nuevas
- Mover y redimensionar las ventanas
- Permite que las ventanas sean minimizadas a iconos o escondidas y entonces restauradas
- Mantiene el inventario de la ventana que esta activa (recibe la entrada desde el teclado)
- Colocar una ventana encima de la otra

Basicamente, el administrador de ventanas maneja todas las facetas del usuario interactuando con la ventana activa del cliente. Esto se puede lograr usando botones, menus, atajos del teclado, o clicks del mouse o arrastrar en la decoraciones de la ventana. Mucho de los administradores de ventana también proveen un menu para lanzar los programas y manipular las ventanas ya existentes.

Algunos administradores de ventanas también proveen espacios de trabajo virtuales. Usted pueden cambiar de un espacio de trabajo a otro, y en cada uno de éstos un conjunto diferente de ventanas aparecera. Puede ser que pueda también pasearse por cada espacios de trabajo, haciendo que la vista que vemos sea sólo una pequeña parte de la pantalla entera.

La mayoría de los administradores de ventanas usan un archivo de configuración para especificar la apariencia de las ventanas, menus, y demás. Algunos de ellos utilizan una estructura de archivos para parecerse a los niveles del menus. Uno de los administradores que trabajan de esta manera es el AfterStep. La gran mayoría de los modernos administradores de ventanas tienen un que permiten su fácil configuración. Cuando exista una duda en como configurar su administrador de ventanas del X, lo mejor es revisar su página man (e.j., man afterstep).

Window Managers Disponibles para GNU/Linux

La mayoría de las distribuciones de GNU/Linux vienen con más de un administrador de ventana. Por ejemplo, Red Hat incluye a fvwm, fvwm2, AfterStep, Enlightenment, y Window Maker. Además podemos descargar otros e instalarlos en nuestro sistema.

Con esta gran variedad puede ser un poco confuso en cual elegir. En lo general, todos ellos poseen basicamente la misma funcionabilidad, pero tienen además muchas otras caracteristicas disponibles. La mayoría de las diferencias están en la apariencia de las ventanas y los menus y lo relativamente fácil o dificil que sean de configurar sus apariencias. La mayoría de los administradores de ventanas tienen siempre algunas similitudes a los escritorios tradicionales de MacOS y otros sistemas operativos graficos. Esto debe ayudar a

muchos usuarios a hacer la transición. Existe una gran variedad de administradores de ventanas disponible para GNU/Linux, aqui algunos de ellos:

- **AfterStep**
  - o Altamente configurable a traves del uso de modulos
  - o Utilización baja de memoria
  - o Se aglumeran rápidamente las pantallas
  - o Toma demasiado espacio
- **Fvwm95**
  - o Similar a la interfáz de Ms-Windows 95
  - o Fácil para eso acostumbrados al interfáz de Ms-Windows
  - o Ya no está muy documentada y poco desarrollo
- **Enlightenment**
  - o Una interfáz muy unica y exotica
  - o Altamente configurable
  - o Consume muchos recursos del sistema
  - o Tomará tiempo en acostumbrarse a su interfáz
- **Blackbox**
  - o Super rápido
  - o Económico en uso de memoria
  - o Toma un tiempo en acostumbrarse a su interfáz
  - o Sus temas son memos customisable que los de los otros
- **IceWM**
  - o Muy configurable
  - o Puede ser uado para emular el look de otros GUIs
- **KWM**
  - o Interfáz fácil de usar
  - o Muchos modulos y programas disponible
  - o Pesado en su carga del sistema
  - o Tiende a ser lento en algunos sistemas
- **Sawfish**
  - o Basado en el lenguaje de programaación Lisp
  - o Altamente configurable
  - o Relativamente nuevo en el area de los administradores de ventanas
  - o Anteriormente conocido como sawmill

Para una buena comparasión de las caracteristicas disponibles para varios administradores de ventanas, visita la página Web en http://www.plig.org/xwinman.

Administrador de Pantalla (Display Managers)

El trabajo del display manager es proveer una pantalla gráfica de ingreso al sistema (login) para así permitir el ingreso de los usuarios al sistema. En vez de que los usuarios tengan que logearse en la pantalla de ingresar, podemos gestionar que cuando X inicie, automáticamente acepte el nombre y contraseña de un usuario por defecto. Esto se logra ejecutando el display manager del X desde el proceso de init.

El display manager por defecto es un programa de nombre xdm. Es estándar con el XFree86 pero puede ser reemplazado con un display manager diferente. GNOME y KDE proveen sus propios administradores de pantalla, llamados gdm y kdm sucesivamente. Todos ellos trabajan más o menos igual pero mantienen una apariencia diferente y posiblemente algunas caracteristicas que lo diferencian. El RedHat usa un script de nombre prefdm, cual determina el display manager preferidos y lo ejecuta.

El display manager se ejecuta desde el archivo inittab. En el momento que el sistema cambia al runlevel 5, el programa init da inicio al servidor X y el display manager nos presenta con un prompt de login. Si el usuario ingresa al sistema con exito, el display manager arranca una sesión para el usuario. Puede iniciar una lista de programas especificados por el usuario o recargar los programas que se ejecutaban cuando el usuario salio del sistema la última vez. Cuando el usuario sale de la sessión del X, el display manager reiniciará el Servidor X y coloca el prompt para ingresar al sistema nuevamente.

Otros Componentes

El Sistema X Window tiene otros componentes, algunos de los más comúnes de éstos son el administrador de fuentes, el administrador de sessión, y las extensiones del servidor.

El Servidor X es capáz de implementar extensiones al protocolo X. Los clientes pueden dirigir peticiones al servidor para determinar que extensiones éste soporta. Una de estas extensiones es la extensión SHAPE. Esta permite a las ventanas tener orillas no rectangulares. Para apreciar si su Servidor de X y Administrador de Ventanas soporta ventanas con forma, ejecute el siguiente comando:

```
$ xeyes -shape
```

Otra extensión común es la MIT-SHM. Está es la que da soporte para compartir la memoria entre el cliente y el servidor. Sólo funciona cuando el cliente y el servidor se encuentran ejecutandose en el mismo computador ofreciendo el mecanismo para pasar montos de data sin utilizar el protocol X. Puedes obtener un listado completo de las extensiones suportadas por el Servidor X ejecutando el programa xdpyinfo.

El Administrador de Sessión es una caracteristica que permite que el X mantenga un inventario de los programas que se ejecutan y además que se encontraba haciendo en cada programa en la última sessión. Los Administradores de display gestionan la administración de la sessión, pero también es posible ejecutar un administrador de sessión por separado. Cuando sales de una sessión, el administrador de sessión le ordena a cada programa ejecutándose que guarde la información de su estado actual y que le entregue una llave al administrador para luego poder volver a accesar ese estado. Cuando el administrador de sessión vuelve a cargar el programa, éste le provee la llave, y el programa restaura el estado anterior. La gran mayoría de los programas aún no tienen soporte para los administradores de sessión. Algunos programas no pueden sostener ningún tipo de estado de una sesións a otra. Por ejemplo, si usted termina una sessión de telnet, el programa no puede reconectarse sin hacer la requisición de su contraseña.

La manipulación de los fonts en X se un poco complicada. Existen un alto número de parámetrosque deben ser definidos, incluyendo tamaño, familia, foundry, peso, slant, y codificar. El X poseé su propio esquema de nombrar los tipos para poder codificar esta información en el nombre. El también usa alias para así proveer un nombre más simple, como es por ejemplo 9xl5bold. Las Fuentes pueden o ser cargadas desde un

directorio en el Servidor de X o desde un Servidor de Fuentes por separado. El Servidor estándar es el XFS. Este se puede ejecutar desde el mismo equipo que ejecuta el Servidor X o desde otro cualquiera. Su función es proveer con la información de las fuentes a los clientes iniciando peticiones. Existen Servidores de Fuentes capaces de traducir fuentes de formatos PostScript o TrueType a formatos X.

CONFIGURAR EL XFREE86

Para poder ejecutar un Servidor XFree86, deberá proveerle información de configuración. En el pasado, cada distribución proveía métodos diferente de configuración, y el equipo del XFree86 también proveía dos programas de configuración. Hoy las distribuciones de GNU/Linux en su proceso de instalación son capaces de detectar apropiadamente la gran mayoría de tarjetas de video y proveer los parámetrosde configuración.

La configuración del Servidor X se efectúa en el archivo XF86Config-4 (donde el 4 es la versión del Xfree que usas, podría ser 3 si no esta actualizado aún). Este archivo de texto, el cual puedes editar en un editor como lo es vi, emacs además existen herramientas para lograr éste objetivo. Las herramientas disponibles tradicionales con el XFree86 son de nombre XF86Setup y xf86config. En esta misma sección cubriremos en detalle la configuración de éste archivo con ambas de estas herramientas.

Los nombres pueden ser un poco confusos, primero, el archivo XF86Config es el archivo de configuración mismo. El XF86Setup es la herramienta de configuración gráfica, y el xf86config es la herramienta de configuración desde la línea de comandos.

Muchas distribuciones también traen su propia herramienta de configurar el X. RedHat tiene un programa llamado Xconfigurator. Se ejecuta desde la consola en modo de pantalla completa. Ejecuta pruebas para detectar la tarjeta de video y procede a pedirle los parámetrosde configuración del monitor y su tipo de mouse. La distribución SuSE viene con la herramienta de nombre SaX, y Turbolinux utiliza une programa de nombre turboxconfig.

Los siguientes tópicos se cubrirán en esta sección:

- XF86Config
- XF86Setup
- xf86config
- SaX y Xconfigurator
- XFree86-4

XF86Config

Los Servidores XFree86 son configurados con el archivo XF86Config. Este archivo se coloca en el folder /etc/X11. Todos los programas de configurarlo proveen una interfáz para ayudar a modificar éste archivo de configuración. Es muy importante que aprenda a editar éste archivo manualmente y a usar un GUI para iniciar con una configuración básica.

Lo primero que debe hacer es identificar el tipo de dispositivo del despliegue (display hardware). Esto se puede lograr usando el programa SuperProbe. Este programa busca en su sistema y trata lo mejor posible de identificar los parámetrosde configuración de sus tarjeta de video. Si el programa SuperProbe no puede detectar su tarjeta, entonces existen un buen chance que el XFree86 no soporta su dispositivo de video. Puede tratar entonces de actualizar su version del XFree86 a una más nueva o cambiar a otro Servidor X incluyendo uno de los comerciales o gratuitos. Casi siempre es más fácil reemplazar la tarjeta de video con una que si es soportada. Hoy en día la gran mayoría de tarjetas son soportadas; sólo tendras problema si su tarjeta es más nueva que la version de Xfree86 que esta usando.

El archivo XF86Config tiene varias secciones. Cada sección empieza con la palabra reservada Section y el nombre de la sección y termina con la palabra reservada EndSection. Las secciones que debe detenerse y observar son Pointer, Monitor, Device, y Screen.

Sección Pointer

En esta sección, usted puede definir el protocolo que va a usar para comunicarse con el mouse y el dispositivo que el mouse se encuentra. Los protocolos del mouse incluyen PS/2, IMPS/2, Microsoft, y Logitech. Para todo lo que va desde el puerto PS/2, use el /dev/psaux como el dispositivo. Para los ratones seriales, use /dev/ttySO para el COM1, /dev/ttySl para el COM2, y así sucesivamente. Muchas distribuciones se las ponen fácil con la creación del vínculo simbolico que les permite usar /dev/mouse sin importar que tipo de mouse éste usando. En la sección Pointer, usted puede especificar algunas opciones, como lo es emular el boton del medio haciendo uso del izquierdo y el derecho simultaneamente.

Sección Monitor

La sección Monitor define las propiedades del monitor. Las especificaciones del sync horizontal definen cuanto ancho de banda el monitor puede soportar y es especificado en kilohertz. Esto ayuda a identificar que resolución es su monitor capaz de soportar. El refrescado vertical nos dice cuantas veces por segundo el monitor puede refrescar las imagenes. Estas dos especificaciones pueden ser especificadas en rango de valores que los monitores pueden soportar. Revise las especificaciones en el manual de su monitor o busque las especificaciones usando unas de las herramientas de configuración disponibles.

Deberá tener mucho cuidado con los parámetrosde las frecuencias vertical y horizontal. Si de llevar su monitor a funcionar fuera del rango especificado, podrá causarle daño permanentes. Siempre debe ser conservador al especificar éstos rango de parametros.

Otros parámetrosque encontramos en la sección Monitor son los de los modes. Tenemos dos maneras que podemos especificar los modes: el primero, usar la directriz ModeLine y especificar todos los números en una línea, segundo, usar la subsección Mode, especificando los parámetroscon el uso de marcados (tags). En ambas maneras, éstos parámetrosle comunican al Servidor X que frequencias y posicionamiento usar para cada resolución. Descubrir éstos parámetrosrequiere calculos complejos. Simplemetne use los ejemplos que se proveen y comente los que no desea usar. Si cualquier de los modes listados cae fuera de las capacidades de su monitor, estas serán ignorados.

Sección Device

La Sección Device especifica los parámetrosde la tarjeta de video. En esta sección, usted puede especificar el chipset que se adaptador utliza, monto de RAM de video que tiene, la velocidad que esta puede usar, y cualquier opciones disponible para el driver asociado con éste chipset disponible. En las mayorías de los casos, no necesitará invocar éstos parametros; ya que el servidor debe detectar éstos parametros. Si por alguna razon el servidor no puede detectarlo correctamente, usted podrá ingresar los parámetroscorrectos en esta sección. También debe revisar la documentación del XFree86; puede ser que el servidor le sugiera parámetros probar en caso de que su tarjeta le esta creando dificultades.

Sección Screen

La Sección Screen unifica toda la información necesaria desde las otras secciones. Usted puede tener más de una Sección de Device o Monitor en el archivo, pero sólo los listado en la sección Screen serán los utilizados. Por esto es que cada sección incluye un identificador. Esta sección también se especifica cual modulo usar, la resolución y la intensidad del color.

XF86Setup

El programa XF86Setup es un programa gráfico que se ejecuta desde el X. Si en ese momento no se encuentra en ejecusion el X, éste calgará un Servidor X generico que debe poder ejecutarse virtualmente en cualquier tarjeta de video compatible con VGA. El programa también le permite el uso del teclado para navegar en el interfáz en el cso de que aún no se ha configurado el mouse. La ventana del XF86Setup contiene varios botones arriba, permitiendole seleccionar la porción del Servidor X que necesita configrar.

La primera sección le permite configurar los parámetrosdel mouse. En la mayor de los casos, sólo nesecitará elegir el tipo de mouse y a que puerto está conectado. Si su mouse está conectado al puerto PS/2, deberá seleccionar el dispositivo /dev/psaux. Si el mouse está conectado al puero serial, usted tendrá que elegir /dev/ttySO. Este es el equivalente al COM1 en MS-DOS o MS-Windows.

La próxima sección le permite a usted seleccionar su teclado (layout). Mayormente ya esto no hay que efectuarle ningunos cambios. El tipo por defecto es un teclado generico de 101-teclas del tipo U.S. English. El utilitario de ayuda en el programa de instalación da una pequeña pero útil explicación de cuales excepciones son significativas.

La sección más importante del programa XF86Setup es la de la selección de la tarjeta de la tarjeta de video. Lo primero a revisar es si su tarjeta esta en la lista de las soportadas. Para ver esta liste pulse en el botón de lista de tarjetas soportadas en la esquina derecha abajo. Navege por la lista y de click sobre el nombre de su tarjeta. Si no esta listada, tendrá que seleccionar el botón de (Detailed Setup) Configuración Detallada. En éste modo, necesitará saber el fabricante del chipset de su tarjeta. El comando superprobe debe poder determinar esta informacion para usted. Usted deberá elegir cual servidor X a usar. Al menos que su tarjeta no sea más de digamos cuatro años, éste deverá ser el Servidor SVGA. Si el Servidor X no puede apropiadamente determinar el monto de de video RAM o o cualquier otros parametros, usted puede selecciona éstos parámetrosen vez de permitir que el el Servidor compruebe para auto determinarlo. Las versiones recientes del XFree86 hacen un magnifico trabajoen detectar automáticamente los parámetrosapropiadamente, así es que lo más seguro que no tendrá que cambiar demasiados de los parametros.

La sección Monitor les permite a usted establecer los rangos de frequencias que su monitor soporta. Si su monitor no esta listado, elija un monitor generico que soporte la la misma resolución que la de su monitor. Al menos que su monitor sea viejo, éste debe poder manejar resoluciones de por lo menos 1,024 x 768 alrededor de 70 Hz.

El panel de Selección de Modes le permite elegir la resolución y la profundida de color a usar. Elija la resolución apropriada para su monitor y su tamaño preferido. La profundida del color dependera en el monto total de RAM de video que tenga su equipo. En lo general, elija color de 24-bit si usted tiene RAM de video suficiente. Las mayoría de tarjetas de videos de hoy poseen más de 8 MB, lo cual es suficiente para ejecutar color de 24-bit en cualquier de los modes que soporta su monitor.

Una vez haya terminado de establecer todos los parámetroscorrectos, dará click sobre el botón Done abajo en la pantalla. Y la configuración se salvará en el archivo XF86Config-4. Ahora podrás ejecutar su Servidor X a ver si funciona.

Ejercicios 4-1: El Uso del XF86Setup

En éste ejercicio practicaremos usando la herramienta de configuración gráfica del Servidor X, XF86Setup. Aunque como cada equipo es diferente podemos simular un equipo en particular y camniar de principio a fin por los pasos requeridos. Pero recuerde no salva esta configuración de práctica ya que lo más seguro que no trabajará en su sistema. No se proveen soluciones a éste ejercicio.

1.- Inicie el Sistema X Window, si es que ya no se encuentra ejecutando.

2.- Inicie un shell prompt en un xterm.

3.- Ya en el command prompt, inicie el programa XF86Setup así:
 # XF86Setup

4.- Establesca que el mouse es un Logitech en el COM1 puerto serial Ms-DOS.

5.- Configure la tarjeta de video como una Matrox Millennium II con 16 MB de RAM de video.

6.- Configure el monitor como uno de alta frecuencia tipo SVGA (1,024 x 768 de 70 Hz).

7.- Establezca el Servidor de X se inicie con una resolución de 1,024 x 768 con 24-bit de color.

8.- Salga sin salvar sus cambios.

Nota: Puede salvar sus cambio para practicar con tan sólo salvar el archivo XF86Config-4 antes de empezar el proceso. Si no sale todo bien podrás siempre regresar a éste posteriormente, así no afectando su sistema en caso de fracasos.

xf86config

Si por alguna razón no puede usar la herramienta XF86Setup, existe una herramienta de configuración capáz de ejecutarse desde la línea de comandos de nombre xf86config. Esta herramienta es un poco primitiva; ella simplemetne es del tipo dialogo repuesta, donde esta pregunta y espera por una repuesta de las electivas que presenta. En la mayoría de los casos, la elección se hace por número o Si/No respuestas. Las preguntas por lo general es del formato del archivo de configuración. Por ejemplo, se le preguntará que tipo de mouse es que tiene conectado a su equipo.

Una vez ha respondido todas las preguntas, se le pedirá que nombre el archivo de configuración. Es una buena idea alamcenarlo en una posición temporaria y así poder observalo antes de utilizarlo. Puede ser que desee compararlo con el archivo de ejemplo que se le provee con la instalación del XFree. Cuando éste listo para probarlo, guarde una copia de seguridad y reemplacelo con el nuevo. Entonces proceda a ejecutar el Servidor X para probarlo.

SaX y el Xconfigurator

Además a los utilitarios ya listado, existen otros ofertados por SuSE y Red Hat. Ambas herramientas proveen interfaces gráficos que facilitan la creación del archivo XF86Config para las necesidades de su sistema. Lo que lo diferencia uno del otro es sus extensas librerias de monitores y perfiles de tarjetas de video y claro su fácil manejo. SaX es lo suficientemente intuitivo para hacer la configuración del Servidor X fácil hasta para un novato en GNU/Linux.

XFree86-4

El más reciente Servidor X del Proyecto XFree86 es la version 4. Esta última version tiene muchas nuevas caracteristicas, uncluyendo una nueva arquitectura de sistema. En vez de tener un servidor diferente para cada tarjeta de video, hay un sólo servidor con modulos injectables como plug-ins para cada tarjeta. También tiene integrado soporte para las caracteristicas 3-D, incluyendo el OpenGL. El Servidor aún usa el archivo XF86Config, pero su formato es un poco diferente.

Además de simplificar por el echo de tener sólo un Servidor ejecutable, la diferencia principal en configurar el XFree86-4 es que éste servidor es capáz de auto generar un archivo de configuración inicial. Para lograrlo, simplemente ejecute el siguiente comando:

 # XFree86 -configure

Este comando generará un archivo de configuración de nombre XF86Config.new en su directorio home.

Primero observe el contenido de éste arcivo en un editor como vi, si se ve en orden, ejecute el Servidor X utilizando éste archivo de configuración:

 # XFree86 -xf86config XF86Config.new

El Servidor X debe iniciarse con un background gris, y usted debe poder mover el cursor. Debe haber pocas cosas que necesite cambiar en éste archivo de configuración. Hay veces que el tipo de mouse detectado éste incorrecto. Estos para estan en la Sección Pointer. Mayormente sólo tendrá que cambiar el protocolo a PS/2 o IMPS/2. Si su monitor no fué detectado apropiadamente, tendrá que asignarle sus parámetros correctos. Si las cosas en su pantalla aparentan muy grande, tendrá que cambiar los parámetrosde su monitor. Otra cosa que pueda querer cambiar en el archivo de configuración XFree86-4 es la resolución defecto y profundida del color. Las opciones disponibles dependen del hardware, pero la mayoría de los sistemas modernos deben soportar profundida de color de 24-bit y una resolución de hasta 1,024 x 768.

Para ajustar la profundidad por defecto en 24, agregue la siguiente directiva en la Sección Screen:
DefaultDepth 24

Usar el X

Usar las aplicaciones X no es algo dificil. Si usted esta familiarizado con MacOS u otros sistemas operativos gráficos, la gran mayoría de las operaciones para manipular las aplicaciones graficas deben parecerles muy familiar. La mayoría de los conceptos y componentes son los mismos entre plataformas. La gran diferencia es la terminología. Claro existen unas cosas que trabajan un poco diferente debido a la implementación del Servidor X. Cubriremos suficiente de las cosas básicas para que pueda iniciarse.

El Servidor X se puede iniciar de dos maneras diferentes. Se puede establecer que el sistema inicie en modo texto como en terminal y entonces lanzar el X desde ahí, o usted puede configurar que el sistema inicie directamente al X y le pida ingresar al sistema ya en un login grafico. En ambos casos, la línea de comandos de GNU/Linux y las consolas estarán disponibles. Cubriremos los métodos de arranque del X en esta sección.

Iniciar un programa en el X es idéntico que en cualquier otro sistema operativo. Usted puede iniciar en una aplicación X desde la línea de comandos o desde otro programa. Al usar la línea de comando, usted debe iniciar la aplicación en el background o segundo plano (agregandole una “&” ampersand al final del comando) así podrá continuar digitando más comandos. A menudo iniciará programas desde un menu proveido por su administrador ventanas o el del escritorio. Usted puede iniciar programas desde el administrador de archivos o el panel del escritorio. El método que usted elija usar para iniciar sus programas del X no importará una vez el programa se ha iniciado.

Una vez haya iniciado el programa de X, usted puede manipularlo como en cualquier otro sistema de ventanas. Existen unas cuantas diferencias que cubriremos. La gran mayoría de estas diferencias son causa que el administrador de ventanas es un componente por separado. Esto significa que lo que usted ve en su sistema puede diferir a lo que le presentemos aquí, pero como hemos dicho, los conceptos deben hacerle sentir comodo con el X. Otra cosa que difiere en aplicaciones X es el método de las operaciones de cortar y pegar.

En esta sección, discutiremos los siguientes TÓPICOS:

- Iniciar el X
- Cortar y Pegar
- Cambiar de Administrador de Ventana
- Trabajar con los Administradores de Ventana
- Cerrar

Iniciar el X

Hay varias maneras de iniciar el X. En la mayoría de los sistemas GNU/Linux, el comando runlevel 5 automáticamente cargará el X11. Esto se logra ejecutando el administrador de Pantalla desde el proceso de init. El administrador de pantalla le preguntará por un nombre de usuario y su contraseña. Normalmente, el administrador de pantalla usará el xdm, cual es el esstandar que viene con el X. Pero cada día más y más distribuciones usan los administradores de pantalla KDE y GNOME (kdm y gdm) ya que tienen más caracteristicas y mejores presentaciones graficas. Una vez el administrador de pantalla le ha gestionado el ingreso, éste da inicio a la sessión del X. Dependiendo del administrador de pantalla, esto ejecutará o los programas programados para el arranque o los que se encontraban ejecutandose al finalizar la última sessión.

Si su sistema no inicia en el X automáticamente, usted puede usar el comando para iniciarlo. En algunas distribuciones, usted tendrá que ser root para poder iniciar el Servidor X porque éste requiere de acceso al equipo (hardware). La mayoría de las distribuciones le permiten a cualquier usuario que desde la consolo inicie el X. El comando startx ejecuta varios scripts de shell después que éste a dado inicio al Servidor X. Estos son usados por lo general para iniciar las variables de ambiente del escritorio o cualquier otro programa que desee cargar.

Usted puede cargar el Servidor X por comando. Este método comunmente es usado sólo en modo de prueba de un Servidor de X recien instalado o reconfigurado. Ningún programa es cargado automáticamente al iniciar el X por comando; tendrá que manualmente cargar cualquier cliente de X si es que lo necesite. De no hacerlo así tendra sólo una pantalla gris con un cursor. Para cargar el Servidor X, sólo ejecute el archivo binario. En el XFree86, éste archivo esta en el folder /usr/X11R6/bin y de nombre algo parecido a XF86_SVGA. La gran mayoría de tarjetas de video usan el Servidor SVGA, pero si existen otros servidores para diferente tarjetas de video, como lo son XF86_Mono y XF86_S3. En la Versión 4 del XFree86, el ejecutable para todas las tarjetas de video es llamado el XFree86 almacenado en el mismo directorio. Las mayorías de las distribuciones proveen un vínculo simbolico al servidor ejecutable para su tarjeta de video, así permitiendole que inicie el Servidor de X simplemente invocando el comando X.

Cortar y Pegar

Muchas aplicaciones X le permiten cortar y pegar data ASCII. Usted puede seleccionar el texto dando click con el mouse izquierdo y arrastrando sobre el texto que desea cortar. Usted puede también dar doble click dentro de un campo de texto para seleccionar una palabra, o todo el contenido del campo de texto, dependiendo del contexto. El texto seleccionado será resaltado o mostrado en colores inverso. Solamente un programa en el X puede tener texto seleccionado a la vez. Así pues si usted seleccionó texto en una aplicación, cualquier otro texto que hayas seleccionado en otra anterior sería deseleccionado.

Pegar en el X es aún más simple que en otros sistemas de ventanas. No tendrá que invocar ningún menu especial o presionar una combinación de teclas especial para indicarle que desea copiar el texto. El texto seleccionado siempre esta disponible para ser pegado. Para pegarlo, simplemente mueva el puntero del mouse a donde desea pegar el texto y de click con el boton del medio y esto pegará el texto. Si su mouse sólo tiene dos botones, podrá simular el tercer boton del medio dando click a los dos botones juntos simultaneamente logrando así el mismo efecto anterior y pegando en texto igual que lo anterior.

Cambiar de Administradores de Ventana

Existen varios administradores de ventanas disponibles en GNU/Linux. Puede ser que desee probar unos cuantos de ellos para apreciar cuales caracteristicas proveen y cual usted prefiere. Cambiar de administrador de ventanas no es dificil, pero cada distribución aparentará tener una manera diferente de hacer

esta tarea. Lo primero que deberá hacer es asegurarse que si tiene otros administradores de ventanas ya instalados. Haga esto utilizando el administrador de paquetes del sistema, igual que cualquier otro paquete de software.

La gran mayoría de distribuciones contemporanias de GNU/Linux incluyen más de una manera de cambiar de un administrador de escitorio a otro incluyendo el administrador de ventanas. Si usted inicia en el X, usted puede elegir en el ambiente que quiere iniciar desde el menu de Sessión. En Red Hat GNU/Linux, las opciones son KDE, GNOME, entre otros. El KDE usa KWM por defecto, y muchas distribuciones tienen a GNOME configurado para usar a Enlightenment por defecto. WindowMaker es otro ambiente que puede usar tanto a wmaker como gnome como sus administradores de ventana.

Otra manera de cambiar de escritorios (y administrador de ventanas) en GNU/Linux es invocando programa switchdesk. Este programa provee un menu pop-up que nos ofrece las misma opciones a elegir de ambientes de escritorios que las del administrador de pantalla. Una vez más, podrá elegir entre GNOME, KDE, o WindowMaker dependiendo cuales tenga instalado en su sistema. Turbolinux le proveep una herramienta, turbowmcfg, que permite que los usuarios cambien rapidamente entre los administradores de ventanas.

La mayoría de los administradores de ventanas incluyen una manera de cambiar a un administrador diferente. El Fvwm tiene una opción de menu para salir del Fvwm y iniciar un administrador de ventanas diferente. Para accesar a éste menu, click izquierdo sobre el centro del escritorio (en la parte donde no éste ninguna ventana) y seleccione Exit/Salir. También puede llegar al menu de Salida/Exit desde el menu Start/Inicio si su sessión esta configurada con un menu de Inicio. El menu de Exit de Fvwm tiene un opción de reiniciar otro administrador de ventanas en lugar de Fvwm.

Si usted usa el escritorio GNOME, usted puede cambiar administrador de ventana desde el Centro de Control, accesible desde el pies de GNOME | Preferencias/Settings | GNOME Control Center. Debajo del pestaña Desktop esta el item seleccionar su administrador de ventana. Aqui todos los administradores de ventanas actualmente instalados deben estar aquí listados. Si usted tiene un administrador de ventana que no aparesca listado, use el botón Add/Agregar button para añadirlo. Si el administrador de ventana que usted eligió incluye un programa de configuración gráfica, usted puede iniciar ese programa de configuración usando el botón Run Configuration Tool listado en esta ventana.

Otras distribuciones tienen métodos diferente de lograr cambiar de un administrador de ventanas a otro y ambiente de escritorio. Consulte la documentación para más detalle de como esa distribución lo logra. En las mayoría de los casos, usted debe poder cambiar usando uno de los tres métodos ya aquí expuesto: durante la fase de login si usted utiliza un administrador de pantalla (display manager), usando algún tipo de panel de control, o usando la función de Exit o Reiniciar del administrador de ventanas.

Ejercicio 4-2: Múltiple Administradores de Ventanas

Este ejercicio explicatorio es para familiarizarlo con las diversas maneras que más de un administrador de ventana pueden co-existir en un mismo equipo ejecutando GNU/Linux. Las preguntas no tienen preguntas correctas ni incorrectas. Respuesta a éste ejercicio estan en el Apéndice A.

1.- Sin iniciar el X, intente determinar cual administrador de ventana estan disponible en su sistema. Explore el arbol de directorio debajo de /etc/X11 y tome apunte de lo que encuentre. Haga lo mismo al directorio /usr/share.

2.- Ejecute el X y determine cual administrador de ventana se esta ejecutando.

3.- Si usted ingresa al sistema via un administrador de pantalla (o sea si usted ingresa desde un prompt gráfico, con su nombre y contraseña), determine cual administrador de pantalla se está usando.

4.- Fíjese si la herramienta switchdesk está disponible en su sistema.

5.- Use switchdesk para cambiar su administrrador de ventanas.

Trabajar con los Administradores de Ventanas

Diferente administradores de ventana dan a las ventanas diferente (look and feel) aspectos y comportamientos. La apariencia y las funciones de los botones, los marcos y los puntos de redimensionar, y la barra de scroll todos variaran de un administrador de ventana a otro.

Además de usar el mouse, algunas acciones pueden ser iniciadas desde el teclado. El mapeo del teclado son llamados aceleradores y casi siempre visible en el menu de la ventana. Estas especificaciones de los atajos del teclado son usualmente almacenados en un archivo de recursos.

Los Admiistradores de Ventana también proveen una interfáz al usuario manejada por menu para iniciar aplicaciones y operaciones de tareas administrativas. Estos menus son generalmente muy configurable.

AfterStep

AfterStep diseñado para emular un administrador de ventana más viejo llamado NeXTStep, cual es considerado por muchos ser uno de las interfaces más intuitivas y útiles. AfterStep fué diseñado para agregarle funcionabilidad y caracteristicas que no fueron originalmente incluidas en NeXTStep. Si necesita más información acerca de AfterStep usted puede visitar su dirección en internet visitando a http://www.afterstep.org/.

Para configurar AfterStep, es necesario editar varios archivos, los cuales se almacena en el directorio /usr/share/afterstep. Aunque la configuración de AfterStep puede ser cambiada desde éste directorio, cambiar esta cambiaría la de todos los usuarios del sistema. Para cambiar la configuración para un sólo usuario, los parámetrosde configuración que debe cambiar estan en el directorio GNUstep/Library/AfterStep/. Algunos de los archivos de configuración más importante de AfterStep son:

gnome	Archivo usado para configurar parámetrosde GNOME para AfterStep
animate	Controla los efectos de animación de AfterStep
audio	Controla los efectos de sonido que AfterStep toca al ejecutar los comandos
autoexec	Especifica que programas cuando se inicia AfterStep
base.xxbpp	Da el nombre y la ruta de los modulos que usa AfterStep
database	La lista de los programas que AfterStep reconoce y los atributos que éstos poseen, como sus iconos y posición y orientación de las barras de titulos entre otros atributos
feel.name	Define el (feel) comportamiento de AfterStep
look.name	Determina como se ve (look) AfterStep
wharf	Permite que el wharf sea configurado

Estos archivos son caracteristicas recientes de AfterStep. Sólo después de la versión 1.8 puede existir más de un archivo de configuración. En las versions anteriores, la configuración era controlada todo en un sólo archivo, de nombre .steprc. Así que, al gestionar con versiones anteriores de AfterStep, es necesario editar el archivo .steprc.

La navegación en AfterStep se logra con el uso de dos menus, accesados dando click con los botones izquierdo y derecho del mouse. El menu Root se despliega con el click derecho, y muestra una listas de los programas del X que se ejecutan. El menu de inicio o Start, se despliega con el click izquierdo, se usa para

dar inicio a las aplicaciones y tener acceso a las tareas administrativas.

El directorio GNUstep se encuentra en el directorio home del usuario. En lo que sigue es un arbol del directorio que se relaciona con el menu Start que se mostró previamente.

Los comandos son almacenados como archivos en sus apropiados directorios. Un ejemplo es el directorio Viewers, allí existe un archivo de nombre XDvi con el contenido de xdvi & (el comando para correr el programa xdvi).

Fvwm95

Fvwm95 es un administrador de ventanas para el X11 que fué diseñado para emular el interfáz de Windows 95 sin consumir demasiado recursos del sistema o sin convertir el código fuente demasiado abultado. El fuente del Fvwm95 es derivado de una versión beta de otro administrador de ventanas de nombre Fvwm version 2, más comúnmente conocido como Fvwm2. Más información del desarrollo de Fvwm95 puede ser encontrado en su website localizable en ftp://mitacl1.uia.ac.be/html-test/rvwm95.html.

El archivo de configuración del sistema completo se llama system.fvwm95rc, el cual por lo general se encuentra en el directorio /var/X11R6/lib/fvwm2/. La configuración de Fvwm95 para un sólo usuario es manejada con simplemete compiar el archivo system.fvwm95rc al directorio home del usuario y renombrarlo .fvwm95rc. La configuración general de Fvwm95 es administrada de la misma manera que la de Fvwm2, aunque esta limitada por por el ambiente de ventanas. Este método de configuración es útil para aquellos ya familiarizados con Fvwm2 para poder configurar el admiistrador de ventana fácil. Algunas de las secciones principales de éste archivo son:

- Fonts Fuentes que serán utilizados por parte de los admininstradores de ventana
- Operating mode Especifica las variadas funciones soportadas por Fvwm95
- Desktop size Provee el tamaño del escritorio como un multiplo de la pantalla fisica
- Module path Nombre de los modulos instalados y la ruta a éstos modulos

Existen aún más secciones que las que hemos listado. Ellas controlan funciones como las que emulan el comportamiento de la fucionabilidad de Windows 95, el contenido de varios de los menus, y la funcionabilidad del teclado y el mouse. La mayoría de las secciones del archivo de configuración son un poco complicadas pero en lo general son entendibles después de examinar un poco su sintaxis además de leer la documentación que provee el propio archivo de configuración.

Para esos aún no muy diestros en editar y generar archivos de configuración para los programas, existen varios programas disponibles que automáticamente generan archivos de texto de configuración para Fvwm. Aunque existen muchos programas que generan automáticamente la configuración, wl ms popular es Dotfile Generator. El Dotfile Generator le permite al usuario configurar los programas así proveiendo un interfáz grafico facil de usar para ajustar varias opciones al programa. Entonces éste automáticamente genera el archivo de configuración requerido por el programa.

Otro método de configurar Fvwm95 es con el uso de los temas. Los temas permiten cambiar rapidamente al administrador de ventana con muy poca dificultad. También pueden ser descargdo del Internet con archivo de configuración ya creado. Debe tener mucho cuidado al usar archivos ya preconfigurados. Además del peligro de seguridad, existen otros problemas de como modules usados por el tema y no instalado en su sistema o las rutas a los modulos puede que no sean igual que los que asigna el archivo.

Salir de Una Aplicación X

Hay dos maneras básicas de salir de una aplicación del X. La primera es usar el método que suple el programa. El segundo es tener el administrador de ventana ordenarle al programa que se detenga. También puedes matar (kill) procesos X así como matas cualquier otro proceso de la línea de comandos (aunque puedes usar la combinación CONTROL+C la mayor de las veces no funcionará con los programas del X).

Cerrar una aplicación X con su propio recurso seleccionando el item del menu Exit desde el menu File, muchas veces existe un atajo de teclas para esta opción para no usar el menu, como es la combinación CONTROL+ Q o CONTROL+X. Este método funciona directamente desde la aplicación; no hay necesidad de interacción entre el programa y el administrador de ventanas. Note que el programa es libre de implemetar cualquier método de apagado que determine necesario. Esto puede ser ejecutar un comando desde el prompt, como es el comando exit en un shell de bash en el xterm. Aunque los ambientes de escritorios estan tratando de estandarizar los interfaces y las combinaciones de teclas, no es requerido que estas llamadas obedescan estas directrices.

Otra manera de cerrar una aplicación es utilizar los widgets del administrador de ventanas. Por lo tanto éstos estan posicionados todo lo alrededores de las ventanas. Cada administrador de ventana es diferente, y hasta diferente configuraciones del mismo administrador de venta puede que coloque las cosas en diferente sitio. Hay vecces que la colocación (o precencia) de los widgets es customizado por el usuario. Usualmente encontrarás un botón de Cerrar/Close en el borde. La gran mayoría de los administradores de ventana modernos utilizan una X para marcar éste botón. La mayoría de los administradores de ventana también tienen un menu asociados con cada ventana. Este menu también contenerá un item para cerrar la ventana. La mayoría de los administradores de ventana también dos funciones de cerrar diferente. Una de estas es permitiendo que el programa le pregunte al usuario si desean salvar y permitir al programa limpiar lo necesario. `esta función por lo general se llama Close, Delete, o Quit en el administrador de ventana. La otra función forzan que la aplicación termine sin permitir que ellas puedan limpiar el ambiente. Este item se denomina como Destroy (Destruir), Kill (Matar), o Annihilate (Liquidar).

Existen otras maneras de cerrar una aplicacion del X. Como cualquier otro proceso, usted le puede enviar una señal al programa usando el programa kill. Este trabaja como cualquier otro programa; el programa puede ser que sepa manejar la señal y cierre apropiadamente o simplemente salga de inmediato. Otro utilitario que podemos usar es xkill. Cuando se ejecuta (desde la línea de comandos o desde el menu), el programa le cambia el cursor a una carabela y cruz. El programa asocia con la proxima ventana sobre la cual usted da click y la termina. El método usado aquí es el de desconctar el programa cliente del Servidor X. Es posible que el programa permanesca ejecutándose después de esto, pero la mayoría de los programas simplemente salen con una condición de error.

Salir del Servidor X

Para salir del X, usted debe necesariamente salir de administrador de ventana o del ambiente de escritorio. En la mayoría de los administradores de ventana, simplemente seleccione la opción Exit o Logout desde el menu. En circumstancias extreme, usted puede matar el Servidor de X con esta sequencia de teclas: CONTROL+ALT+BACKSPACE.

Si usted inició el X desde la línea de comandos (con el comando startx), al salir retornará a la línea de comando de donde inició. Si ejecuta el X automáticamente desde el runlevel 5, el Servidor X se reiniciará y le regresará a la pantalla de Login/Ingreso.

Ambientes de Escritorio/Desktops

Un ambiento de escritorio está compuesto por todo el interfáz Gráfico en el cual trabaja desde su escritorio. Los componentes especificos no estan bien definido, o sea listarlo. Pero por lo general, encontrarás un sistema de menu, un administrador de archivos, programas utilitarios, un panel que contiene applets y lanzadores de programas, un sistema de ayuda, y un conjunto de programas. La meta principal de un ambiente de escritorio es que todos los programas tengan una apariencia, comportamiento y modo de opración consistente y uniforme. El ambiente de escritorio no es lo mismo que un administrador de ventana. Aunque la mayoría de los ambientes de escritorio vienen ya con un administrador de ventana por defecto, usted puede intercambiar los administradores de ventana con los diferentes ambientes de escritorio.

No es necesario ejucutar un ambiente de escritorio. Para lo que es la administración de las estaciones de trabajo, un ambiente basado del todo en el Sistema X Window puede ser la solución más somple y preferida. Por lo general, los ambientes de escritorios están dirigidos a usuarios finales.

Los tres ambiente de escritorio más popular para UNiX son el Common Desktop Environment (CDE), KDE, y GNOME. Cada uno tiene sus ventajas y desventajas, en la próxima sección la listamos. Cada uno esta basado en un conjunto de widget, cual es una colección de componentes de interfáz estándar, como lo son las barras de scroll, butones, listas, y demás que son combinadas para formar una aplicación de ventana completa. Las librerias de distribución para un conjunto de widget deben ser instaladas en un computador para así poder ejecutar programas escritos para ese toolkit. Algunos de los widget propietarios an sido cloneados y proveen un reemplazo alternativo libre a versiones comerciales. Al usar un ambiente de escritorio en particular no excluyes automáticamente ejecutar programas diseñados para correr en otros ambientes. Si el programa contiene caracteristicas especiales que requieren un abiente de escritorio en particular, entonces puede ser que esas caracteristicas no esten disponibles. El programa puede también presentar una apariencia visual diferente a la que presenta al ejecutarse en el ambiente que fué diseñado. En todo otro caso, el programa ejecutará perfectamente bien (esto es claro, tomando en cuenta que las librerias correspondientes al toolkit han sido instaladas en el systema). La razón de esto es que todos los ambientes de Systemas X Window comparten la misma base. Existen además muchas aplicaciones del X que no hacen uso directo de ningún recurso particular de un ambiente, y, claro está éstos se ejecutan identicos de ambiente a ambiente.

Ambientes de escritorio permiten el uso de temas para alterar las caracteristicas visuales y de comportamiento. Los temas incluyen items como es la selección de color, fuentes, colocamiento de varios componentes graficos, e imagenes de fondo. Una gran collección de temas para varios ambientes de escritorios, administradores de ventana, y widget toolkits puede ser encontrado en los sitios Web http://www.themes.org, http://www.gnome.themes.org, http://www.art.kde.org. Estos temas van desde el más simple hasta los más fantasticos y son unas de las ventajas particular fantasticas de éstos escritorios de GNU/linux y UNiX.

Los siguientes TÓPICOS serán discutidos en esta sección:

- CDE
- KDE
- GNOME

CDE

CDE es un producto comercial lanzado por el Open Group y fué uno de los primeros y más exitosos ambiente de escritorio para UNiX. Es distribuido con muchos de las versiones de UNiX comercial, como lo son el HP-UX de la Hewlett-Packard, SCO, UnixWare, Solaris de Sun Microsystems, y el AIX de IBM. Es extremadamente estable y provee una interfáz para las aplicaciones X simple y uniforme. Existen

quejas que CDE ya empieza a desgastarse y le faltan algunas de las mejoras sofisticadas en ambientes de escritorio más modernos, en particular, interoperatividad de los estandares como es CORBA. El CDE esta basada en un conjunto de widget de nombre Motif. Hasta hace poco, Motif era software proprietaria y requiere que los desarrolladores de software compren una licencia para poder desarrollar software basada en Motif. El exito de los varieantes de Open Source UNiX como es el caso de GNU/Linux y FreeBSD han causado que el Open Group lance el codigo fuente con una licencia publica, quizas con la esperanza de que se integren a desarrollar más aplicaciones para la plataforma CDE. Si es cierto, que la naturaleza propietaria de Motif ha inhibido a programadores de GNU/Linux a desarrollar en éste ambiente.

Uno de los incentivos positivos de CDE es el patrocinio corporativo de sus consumidores. CDE tiene un amplia presencia en el mercado corporativo donde tradicionalmente se han usado las variantes de UNiX comercial. Así que un programa escrito con las especificaciones de CDE será muy transportable a cualquier otro variante de Unix. Su uso en las variantes GNU/Linux es menos activo principalmente en debido a los dos ambiente de escritorio grandes de GNU/Linux que son altamente sofisticado, completamente libre que le describiremos a continuación.

KDE

Originalmente para que el nombre fuese como el de CDE, hoy día más por el equipo de alemanes programadores del equipo KDE y el reemplazo de la letra C por K en las lenguas germanicas. Así es que pueda ser que crea que KDE se parece mucho a CDE. No es así. En realidad KDE esta diseñado para parecerce y comportarce a cualquier sistema operativo moderno gráfico. Y de echo hace un trabajo excelente de estas tareas. Muchos consideran que el KDE es el mejor ambiente gráfico al día de hoy de GNU/Linux.

De las cosas positivas de KDE es su estabilidad de muy temprano desarrollo de sus primeras versiones. En el lado negativo, KDE no es el más rápido de los ambientes gráficos y puede turnar lento hasta las computadoras más rápidas. Esta asuntos del rendimiento del KDE han sido enfrentado desde la versión 2.0, y su mejoramiento a sido excelente, KDE ya esta en la versión 3.4. La configuración por defecto de la instalación de KDE es muy automatizada, la cual talves la razón que la comunidad de GNU/Linux lo encuentran un poco aburrido. Esta critica no es justificable, ya que atraves a la aplicación de temas KDE puede ser embellecido al punto de ser tan atractivo como usted deseé.

KDE se ha convertido un poco controversial en la comunidad Open Source por el uso de del conjunto librerias del widget proprietario, de tercero, conocido como Qt, desarrollado por la compañía Noruega Trolltech. Aunque las librerias son distribuidas libremente con KDE, Qt como una plataforma de desarrollo completa es totalmente un producto propietaria cuando es usada para el desarrollo de software comercial. Si desea vender sofware proprietaria que puede ser integrada con KDE y, por ende, también esta basada en Qt, deberá comprar una licencia de distribución de Trolltech por un monto nominal. La ventaja que esto ofrece es que los programas escritos con el uso de las librerias Qt serán fácil de migrar a otro ambiente de ventanas.

Qt también tiene versiones para Win32, numerosas versiones de UNiX propietarias, y ya se esta trabajando en migrarla hacia GNU/Linux embedido utilizando el dispositivo frame buffer, cual puede reproducir raster graphics independiente del Sistema X Window y forma la base para dispositivos de computación del tipo appliance-type, un mercado muy creciente. Qt también es un ambiente C++ orientado a objeto muy sofisticado.

Programadores reportan que es mucho más fácil emplear métodos del desarrollo de objeto en programas que se construyen usando Qt que usando librerias tradicionales como lo es el Motif, Las cuales son complicadas por la limitaciones de los lenguajes procedimental como lo es el lenguaje procedural C. Los componentes desarrollado con Qt tienden a ser reusables debido a la naturaleza modular de la libreria. Esto es una

consideración clave para desarrollar cualquier software comercial.

Desde el punto de vista del uso, KDE es diseñado para hacer que el usuario que viene desde otro ambiente de ventanas se sienta más cómodo. La presencia y el comportamiento visual es suficientemente parecido a los de otros que harán que el usuario nuevo se sienta suficientemente comodo de inmediato con la administración de los archivos, lanzar las aplicaciones, configurar el escritorio, navegar en Internet, etc. Lo que si es indiscutible es, en cuestion de belleza, estabilidad, y profesionalismo, KDE es una opción excelente como ambiente de escritorio en GNU/Linux y es garantizado que su uso siga creciendo.

Ejercicio 4-3: Uso de aplicaciones de ventanas en KDE y su Comportamiento

Se asume que KDE ya está instalado en su sistema y que el comando switchdesk está disponible. Es preferible que éste ejecutando una de las distribuciones reciente de GNU/Linux from Red Hat, SuSE, Caldera, or Turbolinux. No se proveen soluciones para este capitulo.

1. Cambie al ambiente de escritorio KDE.
2. Inicie una sessión de KDE.
3. Inicie una aplicación y llevela al escritorio tres.
4. Cambiese al escritorio Three.
5. Redusca al tamaño de iconos la aplicación que movió en el paso 3
6. Restaure la aplicación que llevó a icono en el paso anterior.
7. Enrolle la aplicacion.
8. Desenrrolle la aplicación.
9. Maximize la aplicación.
10. Kill (matar) la aplicación.

GNOME

GNOME es el ambiente de escritorio del GNU. GNOME significa GNU Network Object Model Environment. Los programas de GNOME son escritos en sus mayorías en C, pero existen bindings disponibles en varios otros lenguajes, incluyendo C++, Guile (un dialecto de Lisp- el nombre es en referencias a Scheme, otro dialecto popular de Lisp), Perl, y Python. Como el KDE, GNOME pone gran énfasis en el ambiente de programación, haciendolo así más fácil crear aplicaciones que conformen. El objetivo principal de GNOME es proveer un ambiente completo, orientado a sessión, para las aplicaciones cliente para los sistemas cliente/servidor de UNiX basados en CORBA una especificación para el modelo de objetos distribuidos parecido al Distributed Component Object Model (DCOM). GNOME se propone extender y mejorar éste modelo cliente/servidor basado sólo en aplicaciones y protocolos OpenSource y no en propietarios.

GNOME ha recibido un gran apoyo de la comunidad de GNU/Linux y sus principales distribudores, esto la ha llevado en tan poco tiempo a un nivel de estabilidad, confianza y rápido desarrollo tal vez no esperado. Además de la comunidad GNU/Linux, los sectores comerciales, tradicionales de UNiX también lo han apoyado, tomemos a SUN por ejemplo, SOLARIS 10 viene con GNOME y no CDE como su desktop por defecto. Estabilidad es una de las razones principales por el rápido desarrollo de GNOME. No todo puede ser perfecto, claro esta, los programadores se quejan que escribir una aplicaciones para el modelo de GNOME es una tarea mucho más difícil. Esto debe ser entendido, tal vez, ya que GNOME es el ambiente de escritorio

más ambicioso en términos de funcionalidad, ya esta bien adulto, la versión es la 2.8, los temas, al igual que los de KDE son estupendos y muchos, disponibles en el sitio WEB de gnome http://art.gnome.org.

Hay que notar, que a diferencias del CDE y KDE, GNOME está basado 100% en software Open Source. El conjunto de herramientas (widget set toolkit) usado para desarrollar (y la mayoría de softwares GNU) completamente a GNOME es conocido como GTK. El GTK es un toolkit muy capaz, aunque, como ya mencionábamos este aún no tiene algunas de las ideas sofisticadas que hacen que Qt sea considerado por los programadores como mejor herramienta para hacer la tarea de programar para el programador más fácil. El GTK es, por cierto, fuerte en particular en desplegar colores hermosos y widgets rápidos y estables. Si el programador está dispuesto a trabajar, los mejores resultados se logran utilizando éste toolkit.

X Remota

El Sistema X Window se estructuró para ser capaz de ser distribuido, así es que es simple ejecutar un programa en una computadora y enviar la salida de video a otra. A diferencias de sistemas ejecutando en otros modelos de ventanas donde los programas asumen que toda su salida esta dirigida a la pantalla local. Esta flexibilidad del X hace que sea fácil administrar computadoras remotamente, y administrar computadoras ejecutando otros sistemas operartivos sea más difícil y requieren software especializada.

En esta sección, discutiremos los siguientes tópicos:

- Displays/Pantallas
- xdm
- VNC (Virtual Network Computing de ORL)
- Security/Seguridad

Displays/Pantallas

Los programas cliente gráficos tienen que abrir una conexión al servidor X para poder acceder y desplegar a pantalla sus salidas. Aunque el cliente y el servidor se encuentren en el mismo sistema, esta conexión deberá ser efectuada. De hecho, los clientes de X asumen que el Servido X se está ejecutando en el mismo equipo al menos que no se le indique diferente.

Hay varias maneras de especificarle a un cliente del X donde encontrar el Servidor X. Antes que todo, la gran mayoría de aplicaciones cliente del X aceptan la opción -display. Simplemente agregue esta opción y el nombre del display, y el cliente intentará conectarse al display especificado. Esto es útil si usted necesita iniciar más de un cliente en un sistema. Si usted desea que todas sus aplicaciones X se ejecuten en un mismo display, entonces usted debe establecer la variable de ambiente de nombre DISPLAY. Usted puede lograr esto en cada comando a ejecutar.

O usted puede establecerlo una sola vez así:

```
$ DISPLAY =host-de-salida:0
$ export DISPLAY
```

Usted también podría establecerlo en sus guiones de inicio (login scripts) si usted desea usar el mismo display cada vez que ingrese al sistema.

Ya sea que decida usar la opción -display o la variable de ambiente DISPLAY, usted tendrá que especificar el nombre del display. Los nombres de display tienen el siguiente formato:

<host>:<display_number>.<screen_number>

host	El nombre o la dirección IP del host donde se ejecuta el Servidor X. Si lo deja en blanco, se asume ser en el sistema local.

display_number	El número ID de la combinación display/keyboard. Este número sería por lo regular 0 al menos que usted no éste ejecutando más de un servidor X en el equipo host.
screen_number	El número ID de la pantalla para los equipos donde un sólo teclado controla más de una monitor (opcional)

Sólo los dos puntos (:) y el número de la pantalla son requeridos.

Aquí presentamos unos ejemplos para ilustrar:

:0	Display 0 en el sistema local (el por defecto)
Equipo1:0.0	Screen 0 en el display 0 en el Equipo1.
Equipo2:0	Display 0 en el Equipo2
192.168.2.7:0	Display 0 en el sistema con ip 192.168.2.7

Cuando especifica el nombre del host por nombre o por IP, no se olvide de los dos puntos y el cero.

xdm

El xdm es un programa simple que le permite al usuario ingresar al sistema e iniciar un shell básico. Al ser utilizado con el X, un usuario debe proveer obligatoriamente ambas el nombre de usuario y el password antes de adquirir acceso al sistema X. Así pues que la seguridad de la contraseña es intrínsica para cualquier programa con el xdm. Además de la seguridad presentada por éste método de login/ingreso, el xdm puede sostener múltiple sesiones a la vez. Tomemos un ejemplo, mientras corregimos un problema, otra ventana de una sessión diferente puede ser abierta para poder determinar si la corrección funcionó o para permitir que otro usuario revise su correo.

El xdm ofrece servicios similares a esos ofrecidos por los programas login, getty, e init pero en un sólo paquete. Con estas capacidades, el xdm controla el proceso de login, y subsecuentemente la autentificación, y entonces monitorea la sessión en curso. Una de las prioridades en el diseño del xdm fué su fácil uso ya que es la primera interfáz que el usuario percibe. Para poder proveer prompts y comandos familiares, se creó un el xwidget. El xwidget se le presenta al usuario con un login y prompt de la contraseña similar al de otros programas.

Otro beneficio del xdm yace en lo fácil que es configurarlo. La mayoría de su comportamiento puede ser ajustado a través del uso de shell scripts y archivos de recursos. Los archivos necesarios para dar ajustes pueden ser encontrados de los archivos xdm.config o desde los archivos nombrados por la opción -config. Como xdm se ejecuta como root, uno debe tener muchísimo cuidado al editarlo.

Aunque xdm es un programa estable, uno debe tener cuidado con cualquier falla que éste contenga. El xdm presenta unos cuantos problemas cuando interactúa con otros sistemas de administradores de ventana instalados en el mismo equipo. Algunos problemas han sido reportado, esto claro esta, limita su uso en sistemas con éste tipo de necesidad.

X-Win32

El X-Win32 es un programa de terminal sofisticado. Es producido por la compañía StarNet y permite que usuarios usando Windows 98 o Windows NT se conecten remotamente a equipos ejecutando sistemas operativos UNiX y sus derivados. X-Win32 permite que aplicaciones X ejecutandose en equipos más viejos en una red LAN se conecte a su computador y utilice su capacidad de desplegar en su monitor usando el protocolo gráfico del X11.

El X-Win32 es un sistema del tipo cliente/servidor pero presenta importante diferencias con otros software cliente/servidor, como es correo/mail o Hypertext Transfer Protocol (HTTP). Aplicaciones X (como es un

terminal) son los clientes, y un display X (como es el X-Win32) es un servidor. Cuando los usuarios inician a X-Win32, básicamente ellos han iniciado un Servidor X en su equipo y están simplemente esperando que un cliente se conecte. En éste contexto un cliente es una aplicación X ejecutandose en el host/equipo remoto.

Típicamente el usuario se conecta a un cliente (casi siempre un equipo ejecutando UNiX), cual es conocido como el host remoto. Cada comando o programa que se ejecuta en el host remoto es una aplicación X.

En esta imagen mostramos el programa utilitario X-Win32. Este le permite a los usuarios crear una nueva sessión o modificar una ya existente. Las sesiones del utilitario X-Win32 se utilizan para almacenar rsh, rexec, y la información del X Display Manager Control Protocol (XDMCP). Las sesiones son creadas con el utilitario X-Util32 e invocadas desde el menu de la Sessión X-Win32.

Existen tres maneras de arrancar un cliente X (aplicación) en un equipo remoto: rsh, rexec, y XDMCP. Ambos el rsh y rexec se conectan a un equipo remoto y ejecutan un comando en esa maquina, normalmente una aplicación X. El XDMCP se conecta al proceso en el equipo remoto del xdm o el CDE para iniciar la sessión del login.

La figuraXXXXXXX anterior es un perfecto ejemplo de crear una sessión en una computadora remota de nombre Equipo1 con el domino abiertos.org. Esta simplemete ejecutaría el comando ls -a para desplegar todos los archivos y directorios en el directorio home del usuario admin.

De una manera similar, la siguiente figurXXXXXXa usa el comando rsh para lograr lo mismo. Recuerda que equipos remotos deben tener los servicios rsh y el rexec ejecutandose.

El usuario puede también usar un archivo script con los comandos rsh o rexec. Este archivo de arranque script se almacena en el host remoto en el directorio home del usuario, así permitiendole al usuario cambiar su ambiente del X cada vez que se inicie una sessión del X.

Virtual Network Computing de ORL (VNC)

El VNC fué desarrollado por Olivetti Research Laboratory. ORL fué fundado en el año 1987 y más tarde pasó a ser financiado en conjunto por la compañía de base de datos ORACLE. Los Laboratorios de AT&T en Cambridge fueron creados cuando AT&T adquirió a ORL en enero del 1999.

El VNC es un sistema de vista a distancia remota que permite que un usuario vea un escritorio desde cualquier lugar en Internet. Para mayor diversidad, los dos equipos no tienen que necesariamente estar ejecutando el mismo sistema operativo.

El VNC de AT&T provee algunas ventajas que la mayoría no ofrecen. No se almacena ningún estado en el equipo que observar. Es muy compacto y fácil de usar. Es lo más cerca de ser independiente de plataforma. Puede ser compartido por muchos usuarios simultáneamente. El software es distribuido bajo la licencia del GNU Public Licence (GPL).

El software se divide en dos aplicaciones que trabajan juntas: el servidor y el visor.

El Servidor

Las aplicaciones lo ven normal como un display X. Un visor de VNC lo percibe como un servidor VNC. El Xvnc utiliza la mayoría de las opciones que utiliza un Servidor X estándar. Una lista de opciones esta siempre disponible ejecutando desde la línea de comandos Xvnc -h. Para iniciar un servidor VNC, simplemente

ejecute el comando vncserver. Este es un script echo en Perl, el cual puede ser personalizado como necesites.

El script/guión iniciará el servidor en el primer número de display/pantalla disponible. Si prefiere en algún caso especificar el número de la pantalla a usar, simplemente escriba "vncserver:NN", donde NN es el número del display a usar. Si éste número está disponible, el servidor se iniciará. Si el número no esta disponible, el servidor simplemente saldrá.

En la siguiente tabla le presentamos algunas de las opciones importantes del Xvnc:

-name NOMBRE	Permite especificar un nombre para cada escritorio; el por defecto es X
-geometry WxH	Establece el tamaño del escritorio, ancho por altura; por defecto es 1024x768.
-depth PROFUNDIDAD	Establece la profundidad de los pixeles del escritorio; por defecto es 8.
-pixelformat FORMATO	Selecciona el formato del pixel, entre BGRnnn o RGBnnn.

Visor

El visor es la segunda parte del software. Este es el display que desplegará el ambiente de escritorio del servidor. Para iniciar el visor, ejecute el script/guión del vncviewer. Una vez ya iniciado, el visor/viewer le preguntará por el nombre del servidor a ver. Naturalmente, el servidor debe estar ejecutandose antes de que el visor pueda funcionar. Usted puede especificar el nombre del servidor y el número del display como una opción. El comando vncviewer Equipo1:2 iniciará una conexión del visor/viewer con el computador y el número del display 2. Si el número es omitido, se asume que el display es 0. Si el nombre del computador es omitido, el viewer asumirá que usted se refiere al computador local que esta ejecutando el viewer.

Una vez el viewer se esta ejecutando, accese una ventana de comando tipo pop-up sólo con presionar F8. La ventana tiene botones de control que permiten la manipulación y cambiar de modos entre pantalla completa y pantalla parcial. La pantalla del vncviewer también permite funcionalidades de portapapel o clipboard para copiar desde sistemas remotos o locales. así como con el servidor, el viewer ofrece muchas opciones desde la línea de comando. Estas opciones se pueden mostrar desde la línea de comandos usando las opciones –xrm. Ambas ventanas la del tipo pop-up y la del escritorio son muy personalizables.

Seguridad

Claro esta, una pantalla que éste en uso no debe ser escrita por cualquier usuario arbitrariamente en la red. El X provee ciertas medidas de control de acceso para asegurarse que otros usuarios no pueden ejecutar programas en su pantalla. En muchos sistemas, éstos mecanismos no están habilitados por defecto, así que tal vez tenga que habilitarlos para hacer que el sistema sea más seguro. Las distribuciones empiezan a configurar sus sistemas con medidas de seguridad más estrictas, así es que puede ser que encuentre que su sistema ya ha implementado estas medidas en su sistema.

Existen dos tipos de control de acceso disponible en el X: el basado en host y el basado en token/ficha. La autentificación basada en Host utiliza un programa de nombre xhost, y la basada en token usa un programa llamado xauth. Daremos un vistazo a ambos mecanismos.

Aunque usted puede limitar quien puede tener acceso a su Display de X, el protocolo X es inherentemente poco seguro. La data transmitida entre el cliente y el servidor no es encriptada de ninguna forma. Esto significa que cualquier password que usted escriba en cualquier programa del X que se ejecute en la red puede ser interceptado. Si usted ejecuta X sobre una WAN, deberá usar siempre y cuando sea posible Virtual Private Network (VPN) para así poder asegurar su conexión. Si ejecuta en una red interna, el riesgo es igual que si ejecuta una sessión de telnet sin encriptar. Afortunadamente, la mayoría de las conexiones del X hoy día se

ejecutan localmente, así que no existe el riesgo de comprometer la seguridad vía la red/network.

El programa xhost

El programa xhost permite control de acceso a base de por host. Cualquiera que se encuentra trabajando en ese host que usted le cedió acceso puede conectarse a la pantalla/display. Funciona muy parecido a la manera que trabajan los wrappers de TCP y como trabajan los archivos hosts.allow y hosts.deny.

Para detener totalmente el control de acceso y permitir a cualquiera tener acceso a su pantalla, use el siguiente comando:

$ xhost +

Este comando en efecto detiene todo el control de acceso. Para reiniciar el control basado en los host, use el siguiente comando:

$ xhost -

Para iniciar acceso a un host en particular, sólo coloque el nombre del host justo después del símbolo de suma:

$ xhost +equipo3

Ahora el servidor X aceptará conexiones desde el host de nombre equipo3. También usted puede remover el privilegios de acceso a un host en particular:

$ xhost -equipo3

Si esto no le provee un control de acceso lo suficientemente afinado, usted debe utilizar el programa auth, para asignar acceso a través del uso de permisos basados en token.

El programa xauth

El programa xauth utiliza un método diferente para decidir quien puede accesar el servidor X. Este usa un llaves (token), también conocido en informática como magic cookie, que es almacenado en secreto por esos autorizado a usar el servidor X. Usted puede solamente accesar al servidor si tiene la llave correctas. El administrador de pantallas genera una llave por cada sesión y hace una copia para el usuario que inicio la sesión. Los programas clientes normalmente administran el proceso de autentificación transparentemente.

Para pasar la llave entre maquinas o usuarios, use el programa xauth. Para dar salida a la llave asociada a la actual sessión a un archivo de nombre auth.out, usted debe usar el siguiente comando:

$ xauth extract auth.out $DISPLAY

Usted puede copiar el archivo auth.out a otro computador. En esta computadora usted deberá ejecutar el siguente comando:

$ xauth merge auth.out

Ahora esta segunda computadora será capaz de accesar la display X de la primera máquina. Usted aún tendrá que especificar la variable de ambiente DISPLAY, para lograr esto; El comando auth sólo le da acceso al otro display; el no redirecciona la salida automáticamente.

Las llaves que el programa xauth genera son almacenadas en el archivo .Xauthority en su directorio home. Usted debe asegurarse de que el archivo no sea leíble por todo el mundo o ellos podrán accesar su display. Si su directorio home esta montado vía nfs, usted no podrá usar el programa xauth ya que el archivo .xauth será compartido y los programas clientes ejecutaran el proceso de autentificación directamente.

Recursos

La mayoría de las aplicaciones clientes usan un conjunto de recursos estándares que se pueden utilizar con opciones desde la línea de comando. Valores pueden ser establecidos vía la línea de comando cuando se ejecuta la aplicación desde una línea de comando:

```
$ xterm -geometry 100x100+10-20 -bg blue -fn vtbold &
$ xclock -g 50x50-0-0 -title “Tick-lock” &
```

Busque en las páginas man del X o la de una aplicación en específico para una lista de los argumentos disponibles para la línea de comando.

Los recursos X se utilizan para personalizar las aplicaciones X. Ellos no son usados por aplicaciones GTK ni Qt. Personalización utilizando los recursos del X es un tópico un poco avanzado. En la mayoría de las circunstancias usted no necesitara tener ningún conocimiento de la aplicación de éstos recursos para ejecutar aplicaciones X, aunque la especificación del color y fuente si se usan en otros lugares dentro del X.

Los siguientes recursos serán cubiertos en esta sección:

- Recursos de Geometría
- Recursos de Fonts/Fuentes
- Recursos de Color
- Recursos de Base de datos/Database

Recursos de Geometría

El recurso de Geometría especifica el tamaño y localidad inicial en la pantalla de una aplicación. Tamaño y localidad pueden ser cambiados después utilizando el administrador de ventana. La sintaxis de la especificación de recursos es:

```
width x height + Xoffset + Yoffset
```

Las unidades de Ancho y altura (width/height) se dan en unidades de pixeles/caracteres. Los Offsets son números de pixeles medidos desde las esquinas.

```
$ xterm -g ‘80x24+0+0’ &
$ xterm -geometry ‘80x24’ &
$ xclock -g ‘700x500-0-0’ &
$ xclock -geometry ‘-1-50-50’ &
```

Omitiendo el width y height de los offsets te retornaran el valor por defecto que es específico a la aplicación.

Recursos de Font/Fuentes

Muchas aplicaciones X permiten al usuario especificar los fonts. Usted debe especificar al lanzar un emulador de terminal para así poder hacer que sea más leíble con fonts más grandes o para desplegar más información en la misma á con fonts más pequeños. Especificar los fonts al iniciar un administrador de ventanas cambiara los fonts utilizados en los menues y características similares.

```
$ xedit -fontname courier &
$ xterrn -fn vtbold &
```

El X soporta un gran número de fonts. Usted puede usar xlsfonts para listar las fonts disponibles en su

sistema. Use el comando xfd para una vista detallada de la definición de un font en particular. El administrador de sistema establecerá alias para los fonts más utilizados. Recuerde que los fonts tiene nombres complejos como el siguiente ejemplo:

misc-times-bold-r-normal—15-140-75-7b-c-90-iso8856-l

Mayormente usted requerirá un font del tipo fixed-spaced. Elija un font que contando en su nombre largo, desde el último campo el cuarto campo contenga una V o "m". Un font proporcionalmente espaciado que contiene en el mismo campo una "p" se vera un poco extraña al dar salida a un campo tabulado, por ejemplo, "ls -l". El termino "fixed" en el segundo campo es también una buena indicación de un fonts de tipo fixed-width, aunque algunas fuentes del tipo fixed-width no contienen esto.

Los nombres Alias de los fonts están definidos en el archivos fonts.alias en el directorio que contiene los fonts. Use xset -q para ver cuales son los directorios de la ruta de los fonts. Los campos en la especificaciones completas de los fonts son:

foundry-family-weight-slarit-setwidth—pixeles-points-hdpi-vdpi-avwid-charset

hdpi, vdpi	Puntos por Pulgada Horizontal y vertical
avwid	Average ancho/width

Prueba el siguiente:

courier*medium-r-*100

Recurso de Color

Muchos aspectos del color de una aplicación como es foreground, background, y text color, pueden ser establecidos. El X usa un modelo RGB para especificar colores. Los colores están compuestos de diferentes intensidades de red/rojo, green/verde, y blue/Azul. El cero denota negro/black. A menudo se especifican los colores utilizando una notación hexadecimal. Una base de datos de nombre Colors esta disponible en el siguiente directorio:

```
/usr/X11R6/lib/X11/rgb.txt
0     0     0     black
255   99    71    tomato
199   21    133   MediumVioletRed
```

He aquí un ejemplo de lanzar el X especificando el uso de los colores basados en el archivo rgb.txt:

$ xterm -bg black -fontcolor gold &
$ xclock -background yellow &

Recurso de Base de Datos/Database

El servidor X mantiene un inventario de los valores por defecto de sus recursos. Opciones de la línea de comandos se anteponen a los recursos por defecto. Sus ajustes personalizados de los recursos se almacenan en un archivo llamado .Xdefaults en su directorio home. Otras implementaciones pueden utilizar un nombre diferente para éste archivo como por ejemplo, .Xresources. Algunas implementaciones no utilizan una base de datos de recurso sino un archivo como es, .Xdefaults éste se lee cuando quiera que una aplicación se crea.

Muchas aplicaciones clientes tienen especificaciones por defecto del sistema definidas en el archivo /usr/X/lib11/app-defaults/<cliente>.

Los recursos en la base de datos usan el mismo criterio de precedencia como los recursos especificados

en su archivo de recursos personales. Usted puede usar el comando xrdb para manipular la base de datos de recursos. La base de datos de recursos se mantiene en memoria y es cargada al inicio del X Window. Usted puede listar los parámetros de los recursos usando el siguiente comando:

$ xrdb -query

La base de datos puede ser recargada usando la opción del comando xrdb -load:

$ xrdb -load /home/usuario/.Xdefaults

Usted puede también agregar o adjuntar con la base de datos actual usando la opción merge. Esto puede leer desde un archivo o desde la salida estándar:

```
$ xrdb -merge
xedit*quit*label; Abiertos
^D .
$ xrdb -merge nombre-archivo
```

Formato del archivo .Xdefaults

Como la mayoría de los archivos de configuración, el archivo .Xdefaults contiene un recurso por línea. Como todo los otros archivos de configuración y scripts, las líneas empiezan con el símbolo de # son comentarios. Cada línea tiene el siguiente formato:

recurso: valor

Las entradas diferencian entre mayúsculas y minúsculas, y los espacios en blanco al final del las líneas son tomados en cuenta y son significativos. Los nombres de los recursos son dependiente de las aplicaciones, pero los nombres estandarizados son reconocidos por la mayoría de los programas.

```
*background:        blue
*font:              courier
*textcolor:         black
*customization:     -color
```

Recursos del Objeto

Cada aplicación X es construida por una jerarquía de objetos. Cada Objeto tiene un nombre y un conjunto de recursos que controlan su apariencia y comportamiento. Usted puede especificar éstos en su archivo de la base de datos de su recursos. Los nombres de los recursos tienen un estructura jerárquica, iniciandose con el nombre de una aplicación o una asterisk (*). Para más información sobre la jerarquía de un objeto de una aplicación X en particular, consulte las páginas man de esa aplicación cliente del X.

```
xedit *background:                  red
xedit *edit Window*background:      blue
xterm *scroll Bar;                  true
```

Existen muchas reglas que gobiernan la precedencia de las especificaciones de los recursos, están están fuera del marco de estudio de éste manual, pero estas reglas pueden ser mencionadas de la siguiente forma:

- Puntos/Dots tienen precedencia sobre los asterisks.
- Instancias tienen precedencia sobre las clases.
- Objetos Explicito tienen precedencia sobre objetos implícitos.
- objetos a la izquierda tienen precedencia sobre objetos a la derecha.

La definición de un recurso tienen el siguiente sintaxis:

application.object.object.object.resource:value

La definición de un recurso puede tener cualquier número de objetos. Una aplicación y un ítemes de Objeto pueden ser omitidos si los puntos son reemplazados por asterisks. Un asterisk significa cualquier aplicación y/o cualquier número de objetos. Si usted utiliza puntos, usted deberá establecer la jerarquía totalmente correcta. así que, xedit.paned.editWindow.background es una especificación completa, y ^background es una mucho menos rigurosa especificación, la cual puede cubrir el mismo ítem (una especificación completa tiene una precedencia mucha más alta).

RESUMEN

En éste capítulo, usted fué introducido a los siguientes temas relacionados con el Sistema X Window:

- El Sistema X Window utiliza una arquitectura cliente/servidor y puede ser utilizado sobre una red transparentemente.
- El X define los mecanismos para hacer las cosas y delega las políticas a otros programas.
- El XFree86 es un paquete Open Source que provee un Servidor X y programas clientes para GNU/Linux y otros sistemas operativos.
- El XFree86 usa el archivo de configuración XF86Config (XF86Config-4 hoy día). Existen varios programas de configuración gráficas que pueden ser utilizados para modificar éste archivo de configuración.
- El servidor X se ejecuta en el equipo que usted está usando. Los clientes pueden ejecutarse localmente o en otros sistemas remotos, y el servidor puede restringir acceso.
- El X está compuesto de muchos componentes, esto incluyen:
    - Servidor X
    - Protocolo X
    - Librerías Xlib
    - Toolkits y conjunto de widget
    - Administrador de Ventana/Window manager
    - Administrador de Pantalla/Display manager
    - Ambiente de Escritorio/Desktop environment
- Un administrador de ventana ayuda a definir la apariencia física y el comportamiento de las ventanas en un escritorio y le permite manipular esas ventanas.
- Las aplicaciones X pueden ser configuradas utilizando opciones desde la línea de comandos y los recursos del X.

Preguntas Post-Exámen

Las respuestas de estas preguntas se encuentran en el Apéndice A.

1.- ¿Cuál es la diferencia entre un administrador de ventanas y un ambiente de escritorio?

2.- ¿Cuáles dos programas pueden ser utilizados para permitir, restringir o denegar acceso al Servidor X? ¿Cuál es la diferencia entre ambos?

3.- Su superior le ha requerido efectuar cambios en las preferencias de sus recursos personales. Primero,

a.- ¿En cuál archivo es que se almacenan estas preferencias en el directorio home de su superior?

b.- ¿Cuál comando se usa para listar éstos valores ajustados?

4.- Liste el orden de éstos componentes del X desde el más bajo nivel hacia el más alto:

Motif, protocolo X, Xt, Xlib.

5.- ¿Cuál es el nombre del archivo que el Servidor Xfree86 X lee para determinar su configuración?

6.- Usted desea desplegar la salida del programa xeyes al servidor X de un equipo de nombre Ivelis.abiertos.org, cual se encuentra ejecutando sólo un servidor X con una sola pantalla. Asumiendo que usted ya ha configurado el servidor para permitir acceso remoto. ¿Cómo puede hacer que el programa se despliegue en éste otro equipo?

DOCUMENTACIÓN Y CORRECCIÓN DE FALLAS

OBJETIVOS

Al completar este capítulo, usted podrá:

- Listar la secuencia de arranque, login y secuencia de apagado.
- Describir y usar lilo.
- Explicar el problema y solución cuando lilo presenta LI.
- Definir el disco de rescate y describir 3 razones para usarlo.
- Explicar por que GNU/Linux podría reportar incorrectamente el tiempo y la fecha y explicar como corregir el problema.
- Listar y describir el uso de programas para navegar el sistema, tales como ps, kill,w, etc.
- Explicar como manipular un programa que no responde.
- Listar y describir 7 herramientas que proveen información sobre otras herramientas.
- Instalar drivers de dispositivos en tiempo de ejecución.
- Describir el rol de los logs del sistema y como usarlos para diagnósticar/corregir problemas.
- Describir la herramienta de Configuración Drakconf, linuxconf y YAST.
- Describir el uso y mal uso de la cuenta superusuario.
- Describa el procedimiento para crear una cuenta de usuario.

PREGUNTAS PRE-EXAMEN

Respuestas a estas preguntas se encuentran en el Apéndice A.

1.- ¿Qué programa utilitario de GNU/Linux carga el kernel en un PowerPC?

2.- ¿Cuál es el propósito de tener varios runlevels?

3.- ¿Porqué quieres usar el comando shutdown en vez de sólo presionar el botón de encendido?

4.- Describir la función del comando kill.

INTRODUCCION

En éste capítulo, examinaremos como negociar con un sistema que no carga u opera de manera normal. La discusión incluye técnicas comunes de resoluciones de problemas que ayudará en el rescate de un sistema deshabilitado.

DOCUMENTACIÓN

Existen muchos recursos de información en relación con los sistemas GNU/Linux. información disponible en línea, que reside en el equipo mismo y otras en el internet. Los libros son un recurso invaluable, particularmente en casos donde el material en línea no esta accesible.

Los siguientes tópicos son discutidos en esta sección:

- Libros
- Usar el internet
- Páginas Man
- Páginas Info
- HOWTOs
- Documentando del sistema

Libros

Existen cada día más fuentes de información para referencia en forma de libros sobre GNU/linux. Los materiales de referencia oscilan desde una ayuda rápida de información general como especifica sobre un tópico, como es el caso de un libro acerca del tópico de bind, sistemas de archivos de red, nfs/nis, etc. Todos son excelentes recursos de información cuando buscamos una información especifica para resolver un problema en alguna aplicación.

Existen usualmente dos tipos de libros :

Libros de Referencia

Estos proveen virtualmente la misma información que las páginas man.

Libros de Guia

Encontraras un titulo similar al siguiente “guia de..., éste libro describe como realizar las tareas a lo que se refiere, comúnmente administración, redes, seguridad de su sistema y frecuentemente ofrecen discusiones, practicas, etc.

Usar el Internet

El internet es probablemente la herramienta más usada y disponible para un administrador. El internet te provee la experiencia y experticio de millones de otros administradores a través de salones de chat y , foros y grupos de noticias. Aquí numerosas preguntas pueden ser hechas y respondidas en cuestión de minutos. También, el internet provee la más rápida actualización de documentación disponible. Probablemente la mejor fuente de DOCUMENTACIÓN en internet es Linux Documentation Project (www.linuxdoc.org). Este grupo sin fines de lucro ha estado trabajando para documentar la características y usos de muchas aplicaciones de GNU/Linux. Ellos proveen las explicaciones sobre como instalar y/o configurar vario paquetes GNU/Linux. Estos HOWTOs frecuentemente no son actualizados pero proveerán la información necesaria.

Páginas Man

Estas páginas documentan muchos de los componentes y opciones de GNU/Linux y sus utilitarios. Usualmente es instalado con el sistema y por esto esta siempre disponibles. Desafortunadamente, las páginas man fácilmente se desactualizán con versiones recientes del software. Si las páginas man no son actualizadas, un administrador podría encontrar una opción que ya éste obsoleta o que tengan nuevas funciones.

Una página del manual puede ser accedida escribiendo man <comando>. Esta sentencia busca en los directorios indicados en la variable de entorno MANPATH. Cuando la página es encontrada, man despliega el texto preformateado en la pantalla. Algunos sistemas despliegan todas las páginas encontradas. Agregando un número de sección al comando man la petición solamente mostrara el comando de la sección deseada. por ejemplo, man 6 <comando>.

Las secciones usadas por el comando man son las siguientes :

1. Comandos generales (Herramientas y utilidades)
2.- Llamadas del sistema.
3.- Librería de rutinas de C.
4.- Archivos especiales.
5.- Formatos de archivos.
6.- Archivos especiales y soporte de hardware.
7.- Información y convenciones misceláneas.
8.- Mantenimiento del sistema y Comandos de operación.

El escenario previo trabajará solamente si el administrador ya conoce el nombre de la página del manual relevante. In otros casos, seria necesario escanear la base de datos whatis para más información. La base de datos whatis consiste en descripciones cortas de varios comandos encontrados en el sistema, los comandos apropos, man -k, o whatis buscaran la base de datos y regresaran alguna ocurrencia de la palabra buscada. Esto es muy usado para encontrar la página del manual relevante.

- whatis <comando> Busca en la base de datos whatis por palabras enteras.
- apropos <comando> Comando para seleccionar palabras claves.

Algunos ejemplos del uso de man son:

$ man man
$ man 6 juegos
$ man ls
$ man n ls

Ejercicio 5-1: Uso de las páginas del manual.

Jamás debe subestimar lo útil que son las páginas del man. Aprendan a usarlas. Si el lenguaje y la termin logías de las páginas del manual le parecen temerosas, ejecute la sentencia man a un comando que cono zca, por ejemplo, a echo o ls. Esto le asistirá a reconocer términos comunes y expresiones usadas. Soluciones a éste ejercicio son proporcionados en el Apéndice A.

1.- Use el comando man para desplegar información sobre el comando passwd, etc. Fíjese que éste muestra información sobre el comando passwd, no el archivo /etc/passwd.

Modifique la sentencia anterior para que se despliegue la descripción del archivo /etc/passwd y no la del comando passwd.

2.- Encuentra cuál paginador esta siendo usado por el comando man y modifique su entorno para utilizar otro. (Por ejemplo, si estas usando el comando less, cambie a usar more y viceversa). Podrías necesitar leer las páginas del manual del comando man.

3.- Encuentre cuál comando de la sección 1 tiene que ver con editar archivo de texto. Recuerde que deberá utilizar una combinación de apropos y el comando grep (para así poder filtrar las líneas de la sección 1 solamente).

Páginas Info

Las páginas info están destinada a ser la próxima generación de las páginas del manual. Debe saber que aún no existen muchas páginas info. En los casos donde una página info no es encontrada o sea no esta disponible, la página info simplemente llama la página del manual relevante. Esto nos es para minimizar la importancia de las páginas info en algunos casos, estas contienen la información más reciente del tópico referente.

HOWTOs

Si se requiere información más específica para una tarea en particular, las páginas HOWTOs son otro recurso que deberá considerar. Existen versiones de documentos HOWTO en HTML, SGML, y en texto, éstos pueden ser encontrados en /usr/share/doc/HOWTO. La mayoría de tópicos amplios tiene un HOWTO escrito para dominarlo, pero para tópicos específicos, verifica el índice del MINI-HOWTO normalmente encontrado en el subdirectorio mini, dentro del directorio HOWTO.

Documentación del Sistema

Para el administrador, quien es responsable de multiples servidores, seria dificultoso o casi imposible recordar las especificaciones de cada host individual. Por esta razón, un administrador puede confiar en el kernel y el entorno operativo para reportar alguna información sobre el host. Podría también ser una buena idea mantener un libro con los logs del sistema detallados.

Usar uname y hostname

El programa uname puede dar al administrador algunas informaciones especificas sobre el host (por ejemplo, el tipo de sistema operativo, el nombre en la red, el tipo de hardware, etc). Esta data puede ser usada cuando configuras varios componentes del sistema operativo (ejemplo, recompilando el kernel). El comando hostname puede También ser usado para ayudar a identificar la maquina y configurar el nombre de la misma.

Libro de los logs del sistema

Un libro de logs del sistema es indispensable para el administrador. el log es usado para mantener récord de eventos del sistema. El libro de logs es un buen lugar donde se detalla todo sobre el sistema, tales como los modelos, el software, hardware instalado, números de seriales, etc.

A continuación mostramos una lista de las cuales deberías tomar nota :

- Fallas del sistema
- Mantenimiento
- Problemas de hardware
- Actualizaciones del sistema
- Instalaciones de software

Mantenimiento preventivo y otras tareas pequeñas pueden ser también incluida, así provee de un punto central de información sobre el sistema. Asegúrese de mantener éste libro en un sitio seguro fuera del edificio donde se encuentra el servidor y alguien que quisiese cometer fechoría le fuese de gran fuente de información.

Diagnóstico y Corrección de Fallas

Varias situaciones podrían suspender su sistema GNU/Linux y causarlo entrar en un estado no operativo. Esto pudiese ocurrir durante una instalación o actualización o simplemente mientras el sistema esta en ejecución. El sistema podría estar en uso o encendido pero no en uso activo. Desafortunadamente, uno nunca sabe cuando algo va a pasar. Las mejores recuperaciones son realizadas por personas que siempre están preparadas.

Los siguientes tópicos serán discutidos en esta sección.

- Tomando el control de la situación.
- Configurando cosas correctas para la primera vez.
- Administrando procesos.
- Tipos de señales.
- La llave mágica SYS RQ.
- Disco de rescate.
- Otras herramientas y técnicas para la solución de problemas.

Tomando control de la situación

Antes de iniciar algunos trucos para intentar reparar la falla del sistema, deberías respirar profundo y tratar de mantener su investigación lo más metódica posible. Esto podría ayudarle a eliminar alguna emoción negativa que podría sentir en ese momento. Tranquilisece y piense en la solución. La siguiente sugerencia podría ayudarle cuando se encuentre en esta situación.

• Verificar los archivos del log

Los logs del sistema son un recurso invaluable cuando estas buscando la raíz de la causa del problema. Una aplicación bien escrita depositara importante mensajes de error en algún lugar del directorio /var/log. Busca a travez de los mensajes y archivos del syslog para encontrar pistas. Si esto falla, busque en los archivos o subdirectorios que puedan estar relacionados con el programa que éste funcionando mal.

• No asumas que es la culpa del computadora

Un sin número de cosas podrían estar causando el problema, incluyendo un error que hayas cometido usted. Un simple error tipográfico de su parte, un problema de permiso de acceso a algún directorio o archivo, o una diferencia de mayúsculas o minúsculas son sólo posibles causas para que una aplicación o sistema operativo falle.

• El computador le ofrecerán cierta retroalimentación de información

Su incapacidad para entender un mensaje en la pantalla que sólo parece basura y pero no significa que nadie pueda interpretarlo, éstos mensajes puede que sean útiles a alguien. A menudo los vendedores de hardware y software catalogan los mensajes de errores más comunes. Aunque sólo sea un simple sonido o un pantallazo de código hexadecimal, trate de recordar los detalles. Debe existir alguna señal que luego le ayude a usted o a alguien a resolver el problema.

• Alguien probablemente le ocurrió esto antes

Por más que quiera ilusionarse que es el primero en descubrir un bug en un programa, probablemente no

eres el primero en verlo, ya que la popularidad de GNU/Linux lo ha llevado a millones de usuarios alrededor del mundo, esto casi garantiza que alguien ha encontrado una corrección para el problema que enfrentas.

Configurandolo Bien desde el Principio

Las herramientas de Configuración Drakconf, linuxconf, Webmin y YaST2 suplen la necesidades de los sistemas GNU/Linux, proporcionando un buen sistema de administración y configuración que le ayudará a manejar los recursos disponibles y a tener una manera rápida de configurar su sistema. Estos 4 utilitarios han sido incorporadas por su enorme funcionalidad para ayudar al administrador a configurar fácilmente cada aspecto del sistema a travez de una interfáz gráfica (GUI).

El Drakconf

Drakconf tiene como objetivo proveer una herramienta de administración de sistema que sea escalable, fácil de usar y compatible con herramientas existentes. La administración unificada minimiza la necesidad del usuario de tener que aprender diferentes interfaces, configuraciones y herramientas. La modularidad permite a usuarios incluir las herramientas que mejor sirven a sus necesidades.

El YaST2

Este es una herramienta de instalación gráfica extremadamente fácil de usar. Automáticamente detecta y configura el servidor X usando SaX (SuSEs advanced X configuration). Por defecto, YaST ofrece inicio desde el disquete, el cual evita el riesgo de comprometer un sistema operativo ya existente.

El linuxconf

Esta es otra herramienta de Configuración que También acciona como una herramienta de activación de servicio. Linuxconf le permiten a los usuarios escoger el tipo de Configuración y el entorno de su preferencia, tales como la línea de comandos, X y basados en web. Todos ellos proveen una ayuda.

El Webmin

Webmin es una interfáz basada en el web para los administradores de sistemas Unix. Utilizando cualquier browser que soporte tables y forms (y Java para los módulos del administrador de archivos), usted puede crear cuentas de usuarios, Apache, DNS, compartir archivos y más.

Webmin consiste de un servidor de páginas web simple, y un número de programas CGI los cuales directamente actualizan archivos de sistemas como son por ejemplo /etc/inetd.conf y /etc/passwd. El servidor de páginas web y todos los programas CGI fueron escritos en Perl versión 5, y no utiliza módulos de Perl que no sean estándar.

Administrando procesos

Un administrador podría notar o recibir un reporte sobre un sistema que no está respondiendo, esta muy lento o presentando algún comportamiento anormal. Una causa común para éstos comportamientos son proceso que se salen fuera de control y así causando que los recursos del sistema se agoten. Las siguientes herramientas proveen información sobre procesos para determinar si están comportandose de manera apropiada y también proveen diferentes maneras para detenerlos o matarlos si han dejado de responder.

El Comando ps

Frecuentemente usaras el comando ps para mirar los procesos y sus atributos. La salida por defecto muestra solamente los procesos que están sobre la terminal actual.

```
$ ps
PID TTY TIME CMD
1844 pts/7 00:00:00 bash
1857 pts/7 00:00:00 ps
```

El comando ps poseé un sin número de opciones para ejecutarse desde la línea de comandos. Fíjese que existen dos tipos de opciones en versiones recientes de ps, las del guión son Syst V compatible y las sin el guión son del tipo BSD. Entre las opciones más comunes tenemos :

$ ps j Muestra los procesos de la terminal actual de manera detallada
$ ps ax a todos los procesos, x procesos que no son regidos por un tty (ej: demonios)
$ ps l Formato Largo (PPID, prioridad, memoria usada,etc)
$ ps u Muestra los procesos por usuario (usuario, memoria usada,uso CPU, tiempo inicio)
$ ps -ef -e significa mostrar todos los procesos; -f en formato completo.
$ ps -C bash Muestra solamente los procesos corridos por el comando bash
$ ps -U cris Muestra los procesos corridos por el usuario cris

Existen dos comandos relacionados a ps: w y who. Estos muestran quien esta logueado en cada tty, El tiempo que tienen ingresado al sistema y que programas están ejecutando.

```
root@monstruo:~# w
18:43:50 up 4:40, 4 users, load average: 0.02, 0.06, 0.12
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU  WHAT
antonio  :0       -                14:05    ?xdm?  16:39  0.00s -:0
root     pts/4    pc-100.sede.org  15:25    3:05m  0.14s  0.14s -bash
root     pts/5    pc-100.sede.org  15:56    0.00s  0.12s  0.00s w

cris@crisdebian:~$ who
root     tty1     Nov 29 09:58
cris     :0       Nov 29 07:20
cris     pts/0    Nov 29 07:30 (:0.0)
cris     pts/4    Nov 29 10:05 (:0.0)
cris     pts/8    Nov 29 13:24 (:0.0)
```

El Comando top

El comando top es probablemente uno de las más importantes herramientas de monitoreo disponibles. Lista los procesos ejecutandose ordenados por su uso de recurso y actualiza la pantalla cada 5 segundos. Por defecto, organiza por uso del CPU pero se puede hacer desplegar en orden diferente. La información que top despliega es similar a la información desplegada por el comando top.

```
Processes:  56 total,      2 running,     54 sleeping...   141 threads    22:34:40
Load Avg:  0.39, 0.20,     0.12 CPU usage: 1.7% user,     13.7% sys, 84.6% idle
SharedLibs: num = 119,     resident = 26.1M code,  2.54M data,   5.54M LinkEdit
MemRegions:        num = 7110, resident = 137M + 5.98M private,   86.0M shared
PhysMem:  67.0M wired, 185M active,      115M inactive, 367M used,  144M free
VM: 4.50G + 82.8M 170246(0) pageins, 128195(0) pageouts

PID    COMMAND      %CPU  TIME     #TH  #PRTS  #MREGS  RPRVT  RSHRD  RSIZE  VSIZE
1492   top          13.6%  0:03.60  1    17     26      528K   436K   2.39M  27.1M
1491   sh           0.0%   0:00.02  1    12     15      156K   884K   724K   18.2M
1490   su           0.0%   0:00.03  1    14     40      144K   684K   2.76M   27.2M
1487   bash         0.0%   0:00.03  1    12     15      176K   788K   772K   18.2M
1486   login        0.0%   0:00.04  1    13     37      144K   428K   508K   26.9M
1485   Terminal     0.8%   0:08.44  2    63     131     2.98M   12.8M  11.7M  145M-
1478   lookupd      0.0%   0:00.23  2    35     56      312K    820K   1.07M  28.5M
1213   Preview      0.0%   0:21.25  1    59     103     2.10M   13.7M  11.8M  145M
```

Mientras top esta ejecutandose, usted puede cambiar la información visualizada, así como el orden en la cual esta sorteada. Presione la tecla ? para ver las opciones para organizar la salida de top como desee. Una de las más importantes es F, la cual se usa para elegir cuales campos de información desplegar. Otra combinación de teclas es SHIFT+M, la cual es usada para organizar la salida por uso de memoria en lugar de uso de CPU.

En la cabecera, aparecen algunas estadísticas completa sobre la PC. La primera línea nos dice cuanto tiempo lleva la maquina y la carga en average del sistema local. La carga en average es el número promedio de carga que tiene el CPU. También puede obtener información utilizando el comando uptime. El programax load muestra un movimiento gráfico del nivel de carga.

Usando kill y killall

Al matar un proceso también podría matar todos los hijos de éste. Si están ejecutandose en background (no suspendidos), continuarán.

kill -TERM -1 (no es L es uno) enviará una señal a todos los procesos con su ID de usuario (UID) excepto al que esta enviando la señal. Esto es de mucha ayuda si tienes procesos frizados fuera de control pero matará todo hasta los de otras terminales. Si eres root, enviará la señal a cada proceso menos los del sistema. Es una buena idea detenerlos antes de matarlos, especialmente si desea reiniciarlos.

Tipos de Señales

El comando kill es usado para enviar señales a procesos. Esta señal toma un número opcional o nombre y una lista de números de identificación de procesos (PID) para recibir la señal.

Algunas señales son sobrecargadas. El uso de SIGHUP (hang up/reiniciar) es frecuentemente usado para enviarle una señal a un proceso del tipo daemon/demonio para que se reinicie. Enviale SIGHUP al proceso init le provoca releer el archivo inittab. Similarmente, enviando SIGHUP al superusuario del Internet (inetd) provoca que éste lea de nuevo su archivo de control (/etc/inetd.conf).

Los procesos rogue son aquellos que parecen haber fallado pero no terminan. En el peor de los casos, éstos procesos ocupan recursos del sistema, tales como, tiempo del CPU, memoria y dispositivos de entrada y salida.

Los procesos iniciados desde una terminal pueden ser usualmente matados o desbloqueados presionando la combinación de teclas CTRL+C o DELETE). Si esto falla, lo más probable es que el programador de la aplicación haya colocado una instrucción en ella de capturar éste tipo de señal e instruido ser ignorada. Desafortunadamente, QUIT muchas veces provoca un vaciado de memoria a un archivo core, porque muchos programadores olvidan sobre (o no la conocen) la señal QUIT y falla al manejarla.

Los procesos que no se pueden matar desde el terminal deben ser terminados usando el comando kill ejecutado desde otra terminal. Sólo el superusuario puede matar cualquier proceso; los usuarios normales sólo pueden matar sus propios procesos. Pruebe el simple comando kill (sin ningún número de señal) para terminar un programa con TERM. Si esto no funciona, ejecute kill -9 <nombre-comando o PID> para forzar el proceso a morir con una señal de KILL. Aquí presentamos un ejemplo:

```
cris@crisdebian:~$ ps -fu
Warning: bad syntax, perhaps a bogus '-'? See http://procps.sf.net/faq.html
USER       PID  %CPU %MEM     VSZ   RSS   TTY      STAT START TIME COMMAND
cris     15747   0.6  0.9    3744  2408 pts/19   Ss   18:27   0:00 bash
cris     15780   0.0  0.3    2520   800 pts/19   R+   18:27   0:00 ps -fu
cris@crisdebian:~$ kill -9 15747
```

Aquellos procesos que no mueren después de recibir una señal de KILL están en el proceso de liberar recursos del sistema y terminarán cuando los mismos hayan sido liberados. Los procesos zomby ya están muertos y por eso no pueden ser matados; el proceso padre de éstos procesos debe salir para que desaparezcan del sistema. En situaciones extremas, podría ser necesario reiniciar el sistema para remover los procesos zombis.

Ejercicio 5-2: Señales

Este ejercicio es sólo una práctica y no provee solución para éste ejercicio.

1.- Inicie algunos programas (por ejemplo, cat) y utilice CONTROL+Z, CONTROL+C y CONTROL+\ para ver que pasa. Sobre CONTROL+Z, utilice el comando fg, entonces CONTROL+Z de nuevo y utilice el comando bg. Imprima un listado con ps u después de cada uno. Entonces utilice el comando kill para matarlo. Asegúrese buscar el archivo de core de la memoria después de los CONTROL+\.

La tecla mágica SYS RQ

Una de los procedimiento para la recuperación más frecuentemente obviado es el uso de la tecla SYSRQ. No todos los sistemas GNU/Linux toman ventaja de esta herramienta; es una opción que esta disponible durante el proceso de compilación del kernel. Es opción del distribuidor/vendedor de la distribución de GNU/Linux si desea utilizar esta característica o no. Si estas compilando su propio kernel, podría considerar incluirlo.

La tecla SYS RQ (en algunos teclados aparece como PRINT SCREEN) es usado en conjunto con otras teclas del teclado para producir combinaciones de teclas que serán reconocidas por el kernel de GNU/Linux. Si el kernel de GNU/Linux no responde a la combinación de teclas, podría ser por dos razones:

- El soporte para la tecla SYSRQ no fué compilada en el kernel.
- El sistema ha caído a niveles lo suficiente para prevenir que el kernel responda.

La siguiente tabla describe algunas de las combinaciones de tecla más comunes para realizar varias funciones:

SYSRQ	ALT B	Reinicia el sistema (Ni sincroniza discos ni desmonta sistema de archivos)
SYSRQ	ALT E	Envía un SIGTERM a todos los procesos menos al init.
SYSRQ	ALT I	Envía SIGKILL a todos los procesos menos al init.
SYSRQ	ALT K	Mata todos los programas ejecutandose en la consola actual.
SYSRQ	ALT R	Deshabilita el modo raw del teclado.
SYSRQ	ALT S	Sincroniza todos los discos montados.
SYSRQ	ALT U	Remonta todos los sistemas de archivos en modo de sólo lectura
SYSRQ	ALT 0 al 9	Ajusta el nivel de log de la consola.

Discos de Rescate

Dos métodos son proporcionados en la mayoría de distribuciones GNU/Linux para reiniciar el sistema y realizar el procedimiento de recuperación:

- Utilizando el Medio de Instalación Original
- Creando a disco de rescate personalizado

Usando los Medios de Instalación Muchas medios de instalación, en la gran mayoría de distribuciones, contienen algún tipo de sistemas de archivos esqueléticos para la realización de tareas relacionadas a la instalación. Algunas inician al usuario directamente a un shell donde usted puede realizar comandos regulares

de GNU/Linux antes de empezar la instalación (tal es el caso de slackware), otros inician directamente un programa de instalación (como es el caso de Fedora y Redhat).

Por ejemplo, La instalación en disquete y CD de RedHat contienen un sin número de utilidades básicas del sistema para asistir en caso de tener que reparar el sistema. Este disco de inicio puede ser creado desde el CD de la distribución utilizando el comando dd de GNU/Linux. Después que su sistema haya iniciado, puede hacer intentos de montar el disco duro para examinarlo para así poder determinar las causa(s) del problema.

Arrancar desde el Disquete de Rescate

Para usar las características de rescate del Disquete de instalación de Red Hat, haz lo siguiente:

1.- Asegúrese que el computador esta apagado.
2.-Insertar el disco de inicio en la disquetera (el Disquete creado durante la instalación o el que contiene boot.img) o el CD de inicio de la instalación en el cdrom e inicia el sistema.
3.-En el prompt del sistema escriba “linux rescue” y oprima la tecla enter.

Luego de que el contenido del disco ha cargado, entrarás al prompt de GNU/Linux y desde ahí estarás listo llevar a cabo tareas de recuperación del sistema y corregir errores.

Discos de Inicio de Distribuciones Específicas

Durante la instalación de la gran mayoría de distribuciones de GNU/Linux, se le dará la oportunidad de crear un disco de inicio/boot. Este disco no debe se confundido con un disco de rescate porque usualmente contiene solamente el kernel de GNU/Linux y los módulos necesarios para iniciar el sistema. El beneficio de usar el disco de inicio es que contiene información de la instalación especifica para tu sistema.

Cuando inicias desde el disquete. Linux es cargado desde el disco de inicio. Tu archivos del sistema regular es montado. Por lo tanto, si tu archivo del sistema raíz esta corrupto o inaccesible, el disco de inicio no será de mucho uso. En éste caso, probablemente necesitaras usar el modo de rescate previamente descrito. El disco de inicio de emergencia personalizado es más usado cuando existe un problema de inicio del sistema.

Crear Discos Personalizado de Inicio y/o de Rescate

Tu puedes crear tu propio disco de rescate o de inicio. Una de las opciones cuando el kernel esta compilando es realizar un zdisk, esta copia la imagen del kernel a un disquete. Esto puede ser usado si tu cargador de inicio no funciona apropiadamente.

Si has compilado opciones especificas como modulos del kernel, necesitarás también crear un disco inicial de RAM. Un kernel en un disquete es poco útil si no tiene los drivers para accesar al hardware del sistema. El comando mkinitrd puede ser usado para crear una imagen de disco de RAM inicial, la cual entonces puede ser cargada dentro del disquete.

Descargando un Disquete de Rescate

Especializado Para aquellos que no están interesados en crear sus propios discos de rescate, una alternativa disponible es descargar alguno que alguien más ya ha creado y lo ha puestos disponibles para el uso libre. Un disquete muy sencillo que esta disponible con todas las características de cualquier distribución GNU/Linux, algunos con un sistema de solución de problemas integrado. Uno en particular, bien conocido para solución de problemas es tomsrtbt o Tom’s Root/Boot. Este contiene un gran número de utilitarios para reparar un sistema dañado.

Ejercicio 5-3: Crear un Disquete de Emergencia

No se proveen soluciones para éste ejercicio.

En las mayoría de las computadoras en sus BIOS uno puede indicarle que en caso de encontrar la presencia de un disco de inicio, y si el disquete de inicio existe y es booteable, que el computador se inicia desde éste disquete. De esta forma, puedes recuperar el control de un sistema que se niega iniciar o si has perdido el password de root. Esta es una de la razones por la que hay que asegurar que el computador físico no esta abiertamente disponible para todos, ya que si su computador esta iniciable desde un disquete cualquiera puede tener acceso a sus recursos.

Cuando primero instalaste GNU/Linux, lo más probable que el sistema le ofreció la oportunidad de crear un disco de emergencia de inicio. Si no posees ese disco (porque obvió crearlo o se extravió), puedes crear uno desde un sistema que aún está trabajando. Los procedimientos específicos varián en algunas distribuciones; Usaremos en éste capítulo el proceso para un sistema RedHat.

El procedimiento utilizará el utilitario mkbootdisk. Lea la página del manual para esta utilidad. Las siguientes instrucciones interpretan algunas de las opciones en la página del manual. Conocimientos de otras opciones vendrán con la práctica y el estudio.

1.- Inicie el sistema normalmente. Es una buena idea crear el disco de emergencia mientras esta instalando el sistema ya que si pierde la capacidad de inicio del sistema, deberá iniciar desde el CD o ún Disco de Inicio.

2.- Entra en el sistema como root, login.

3.- Determine la versión del kernel. Multiples kernels puede que estén instalados y LILO se configura para seleccionar entre ellos. Un disco inicio puede ser creado para cada uno de ellos. La versión del kernel es el nombre del directorio /lib/modules.

Ejemplo:
cris@crisdebian:~$ ls /lib/modules/
2.6.7-1-386

4.- Si aparece más de uno, puedes determinar cual esta actualmente cargado con el comando uname (es actualmente la manera más apropiada para determinar el número de versión).

cris@crisdebian:~$ uname -r
2.6.7-1-386

5.- Ejecute el comando mkbootdisk dandóle el nombre del kernel a ser colocado en el disquete como argumento:

cris@crisdebian:~$ mkbootdisk 2.6.7-1-386

Se le indicará que inserte un disquete y presiones ENTER cuando éste listo. Toda la información almacenada en el disquete será reemplazada, cuando retorne el prompt el proceso habrá concluido.

6.- Retire el disquete y coloquele la banda protectora contra escritura. Esto protege la escritura accidental en el disco. Puedes montar el disquete y examinarlo pero asegurate de desmontarlo antes de sacarlo de la disquetera.

7.- Inicia desde el disco de emergencia creado. Para hacer esto, reinicie o mejor aún apague el sistema, inserte el disquete antes que el sistema escoja su modo de inicio.

Ejemplos:

Reinicie el sistema así:

crisdebian:/home/cris# shutdown -r now

Apague el sistema así:
crisdebian:/home/cris# shutdown -h now

8.- Con algunos tipo de instalación, iniciar desde el disquete, es la manera normal de iniciar el sistema.

Otras Herramientas y Técnicas de Diagnóstico y Solución de Problemas

Ya hemos cubierto algunas maneras de recuperarnos de algunas fallas y problemas específicos, y ahora discutiremos otras herramientas disponibles para algunos otros problemas diferentes.

Módulos

El kernel linux soporta módulos como extensiones para la funcionalidad del mismo. Los módulos proveen una manera de soportar hardware adicional y software que talvés no sea requerido para inicializar el sistema pero tiene un propósito funcional en la operación diaria del sistema. Por ejemplo, tarjetas de red y de sonido son comúnmente soportados a través del uso de los módulos.

Algunas veces podría encontrar que un dispositivo no esta trabajando de la manera correcta o peor aún que no está trabajando del todo. Si estas utilizando módulos en el kernel, podemos hacernos estas preguntas:
- ¿Compiló recientemente el kernel pero no ejecutó make modules o make modules_install?
- ¿Agregó usted recientemente algún hardware que su kernel podría no tener el módulo compilado?.
- ¿Intentó ejecutar lsmod para ver si el módulo esta cargado en memoria?

Reloj del Sistema Inexacto La razón que GNU/Linux podría reportar la hora equivocada es porque lee la hora desde el reloj CMOS. El reloj CMOS no es terriblemente exacto y algunas veces corre muy rápido o muy lento, creando dificultad para mantener el tiempo exacto. Existen algunas soluciones para mantener el reloj un poco más exacto. Una manera para corregir el problema es usar el comando ntpdate, usando como conocido un time-server tal como clock.isc.org.
- La hora es leída desde el reloj CMOS del hardware.
- El comando ntpdate clock.isc.org actualizará el reloj.
- El reloj puede ser actualizado en el programa de Configuración de CMOS al momento de arranque.

Reiniciar la Consola

Es muy común, en GNU/Linux pasar mucho del tiempo leyendo archivos de texto de configuración en la consola. Es muy probable que un día por error trate de leer un archivo ejecutable binario con el comando cat por ejemplo. Esta acción causará que en su consola se imprima una serie de caracteres no soportados, la consola tiene un número limitado de caracteres que puede desplegar a pantalla y cuando esto sucede la pantalla pasará a un estado no-leíble. Cuando trate de escribir a pantalla notará que no podrá leer el texto, para corregir esta situación tendrá que proceder escribiendo reset sin preocuparse que no puede leer bien lo que escribe y simplemente presionar ENTER... casi en todos los casos esto deberá funcionar.

Procedimientos de Recuperación

Sabemos que un día pasará todo lo malo, la ley de Murphy se cumplirá. Su misión crítica en el servidor de base de datos dejará de funcionar en el peor momento posible. Peor aún, la solución planificada en caso de esta situación también fallará y no trabajará como estaba planeado. Su servidor está abajo y necesita subirlo lo más pronto posible. Cada hora que pasa son miles de clientes y ventas perdidas.

En esta sección, examinaremos procedimientos de rescate para revivir un sistema que no responde. Cubriremos los siguientes tópicos en esta sección:
- Generalidades sobre Recuperación de Desastres.

- Crear un conjunto de disquete de Rescate.
- Recuperarse de Falla del Disco Duro Raíz.
- Recuperarse después de Fallas de Energía.

Generalidades sobre Recuperación de Desastres

Se entiende que como todo un buen administrador de si temas, usted tiene una copia de su backup y un esquema de restauración realizable una vez que el sistema éste ya operacional; eso es, si todas las particiones están allí y el hardware trabaja, puedes obtener los archivos desde la copia de seguridad. Debe recordar que recuperarse de un desastre es mucho más complejo que simplemente copiar todo sus datos desde una cinta magnética de backup a sus discos duro. Se asume que usted tiene un plan, ya practicado del cual el primer paso es devolver el sistema a un nivel de funcionalidad mínima.

También es importante entender el propósito general de las herramientas de recuperación y saber como y cuando usar la herramienta adecuada para el trabajo correcto. Por ejemplo, una excelente solución completa de copias de seguridad a cintas magnéticas para guardar a salvo toda la información corporativa de la empresa, pero en el caso de rescatar sólo el Servidor de páginas Web, podría ser que resulte dificultoso o casi imposible. Una herramienta que debería ser una parte esencial en su esquemas de recuperación, es tar. Este utilitario es lo suficientemente pequeño para caber en un disco de rescate, pero lo suficientemente poderoso para recrear una gran estructura de directorios o hasta una jerarquía de un sistema de archivos completa.

La recuperación del sistema en GNU/Linux puede resultar un poco compleja. Por razones tales como, que sistemas de esta índole tienden a permanecer dando servicio por prolongado periodo de tiempo, esto tiende a producir el efecto de falta de experiencia por el hecho de que los sistemas no fracasan a menudo. Pero debemos recordar que por excelente que sea los sistemas GNU/Linux y por bueno que sea nuestro hardware, un disco duro o una tarjeta puede fallar y por esto es que debemos tener planes de contingencia.

El esquema ideal de particionamiento de discos fijos aún no ha sido inventado. Pero se asume que, como mínimo, root y /usr se montan desde sistemas de archivos separados. Si su sistema no esta configurado de esta manera, deberías planificar reestructurarlo para que estén por lo menos así.

Pero, si todas sus particiones están en el mismo disco y éste falla totalmente, es seguro que instalaras un nuevo disco, instalarás un GNU/Linux con requerimientos mínimos, debe recrear las particiones y restaurar todo desde su copia de seguridad. Vamos a tratar de recrear un escenario más común que éste, en el cual el disco falla parcialmente, por ejemplo, el disco esta trabajando con sectores dañados, pero el sistema raíz esta tan corrompido que los errores no permiten repararlo sin desmontarlo.

Esta situación en la que un disco que ha desarrollado un gran número de sectores dañados, es inevitable su reemplazo, esta es la primera etapa de una falla total de un disco, así que el problema lo enfrentaremos recuperando la mayor cantidad de data posible del disco dañado para transferirla al disco nuevo.

Crear un Conjunto de Discos de Rescate

Si su sistema puede ser iniciado desde el CD-ROM, que hoy día es el mayor de los casos, usted puede utilizar los CDs de instalación como paso inicial hacia el rescate de un sistema que no responde u hoy día con la gran cantidad de los denominados Live-CDs (Knoppix, SuSE, Gentoo, Slackware en fin casi uno por distro), pero muchas veces la mejor solución es un conjunto de discos de rescate, preparado por usted mismo hechos a la medida de sus sistemas. Este conjunto que aquí preparamos son de propósito general, pero usted podría usar su creatividad y desarrollar conjuntos mucho más complejos. Además en vez de hacer disquetes se puede preparar en CDs, ya sean Live-CDs o auto-arrancables que nos lleven a un simple shell. El

primer disco nos inicia el sistema y el segundo disquete contiene los utilitarios y archivos que necesitamos para rescatar el sistema. El conjunto de discos de rescate de propósito general trabaja con cualquier distribución, y es útil si usted usa más de una distribución de GNU/Linux en su ambiente de trabajo o si se dedica a dar servicios de soporte. Este conjunto de dos discos contiene las herramientas suficiente para iniciar el proceso de recuperación y en las mayoría de los casos hasta poder completar el trabajo. Otra opción disponible es la de crear una imagen del kernel y un grupo de módulos que igualen exactamente a esos de su sistema para los dos disco de rescate. Estos discos hechos a la medida pueden entonces manejar todo el hardware del sistema como es montar discos, CDs, etc.

Crear un disco de inicio es un procedimiento un poco complejo que esta muy bien documentado en el HowTo que aparece en el Internet. Este está en el portal The Linux Documentation Projects del cual su página web es http://www.linuxdoc.org nosotros nos tomamos la libertad y colocamos, el traducido a español, extraído desde el Linux From Scratch de The Linux Bootdisk HOWTO por Graham Chapman y Tom Fawcett, en nuestro portal de CODIGOLIBRE.ORG en la dirección web siguiente en la sección de HowTos/Comos con su link directo de

http://www.codigolibre.org/modules.php?name=Sections&op=viewarticle& artid=297

Lo que sigue a continuación es sólo un resumen de los puntos principales del proceso:

- Construya una estructura de archivo raíz/root para colocar en el disquete de inicio que preparamos en un disco en RAM (RAM disk). Esto incluirá directorios como son /usr, /etc, /dev, y /proc. Recuerde, que estamos construyendo un sistema GNU/Linux que funcione mínimamente.
- Copie un conjunto básico de utilitarios como son sh, ls, y cp mv, a /usr/bin. Fíjese que copiamos el shell más pequeño por asunto de espacio solamente para salvar espacio en disco.
- Copie las entradas apropiadas del directorio /dev de su hardware en particular.
- Copie las librerías apropiadas de tiempo de ejecución de los utilitarios y programas de recuperación en el directorio /lib y /usr/lib.
- Copie su kernel y el grupo de modulo que esta usando actualmente.
- Ejecute la sentencia lilo sobre su imagen de RAM disk para convertirla en auto-arrancable/bootable.
- Vuelque el contenido de la imagen RAM disk a un disquete/floppy.
- Copie administración de cintas/tape y otros utilitarios a un segundo disquete que va ha ser montado luego que el sistema se inicie desde el disco principal de rescate.

Llevar todas estas tareas a cabo es, modestia a parte, tedioso y peor aún aburrido y frustrante en el momento de tener que elegir las interdependencias de la librerías de las herramientas que colocará en el disco y decidir las entradas del directorio /dev que son esenciales, buscar un tamaño pequeño y a la vez útil de las entradas en el directorio /etc, y todo esto sin perder información critica, ajustar las variables de ambiente del RAM disk, y etc, etc.

Afortunadamente, Tom Fawcett, co-autor del Bootdisk HOWTO, ha escrito un utilitario muy bien conocido de nombre Yard (Yet another rescue disk, su nombre es muy modesto comparado con su sofisticación). Este paquete contiene tres scripts escritos en Perl que llevan a cabo todo el análisis necesario para resolver las dificultades de crear un conjunto de discos de rescate y así permitiendole a usted concentrarse en contenido y funcionalidad.

Usted puede descargar éste paquete en su última versión Yard 2.0 desde la siguiente dirección de internet:
http://www.cro ftj.net/-fawcett/yard.

Los Requisitos de esta son de sistemas que tienen:

- Kernel GNU/Linux versión 2.0 o más actual.
- Tener Perl 5 instalado. Para revisar ejecute el comando perl -v.
- Soporte para RAM disk comprimidos, soporte para sistemas de archivos ext2, y soporte de disquete compilado en el kernel (no como modulo).

La estrategia de Yard es como sigue:

- Analiza completa y exhaustivamente su sistema para la disponibilidad de las herramientas, versiones de las librerías y su uso, consistencia de los sistemas de archivos /etc/fstab y /etc/inittab, consistencia en las definiciones de los usuarios y directorios, y el uso de Pluggable Authentication Module (PAM) y NSS.
- Construye una imagen de un RAM disk basada en éste análisis y en inclusión de archivos especificados por usted, el administrador del sistema.
- Revisa la viabilidad de la imagen construida del RAM disk.
- En el momento que todas las inconsistencias y errores han sido removidas, se escribe la imagen comprimida al disquete.

No se deje confundir aunque se le presente aquí con la mucha facilidad que es escribir esta imagen del RAM disk y todos los detalles de la construcción del Disco de Rescate, aún existe una gran participación por usted de escribir y editar a mano archivos y parámetros de configuración antes de que pueda tener en sus manos una copia funcional del Disco de Rescate. Como hemos visto construir éstos disco de rescate hechos a la medida de sus necesidades es un procedimiento complejo. Pero los resultados que se adquiere de ellos en el momento de fallas valen la pena todo el esfuerzo en ellos empleados.

NOTA: Debe poner mucha atención e interés en reducir el tamaño de su kernel y los módulos para ahorrar espacio en el disco de rescate.

Recuperando la Falla del Disco de Root

El peor escenario es que le falle el disco raíz. Esto significaría:

- No podrás iniciar desde un disquete y entonces montar el disco raíz. Esto significa que el disquete creado con mkbootdisk le será de gran utilidad.
- Si el directorio /etc/ era parte del sistema de archivo raíz o era montado aunque de otra partición, todos los archivos de configuración del sistemas almacenados allí no estarán disponibles.

El disco de rescate suplido por su distribución no le será útil en el caso de una catástrofe total o fallo del disco raíz. Estos discos de rescate no contienen información sobre su red, sus usuario, etc. Estos casos es donde los discos de rescate construidos con el utilitarios YaRD adaptado a sus necesidades muestra sus verdaderas bondades. Con ellos puede instalar su nuevo hardware, entonces localiza la copia que contiene su último backup realizado de su partición raíz ya sea desde otro punto en la red o localmente instalado, iniciar desde los discos de rescate, montar el disco de los utilitarios, y estará listo para iniciar el proceso de rescate. Si la partición raíz esta dañada más allá de poderse reparar, sería tan simple como instalar el disco duro nuevo, ejecutar fdisk para crear las particiones incluyendo cualquier partición swap que se encontrara en el disco dañado, y restaurar la partición desde el backup. Esta garantizado que todo sus archivos y sus permisos se encontrarán en perfecto estado, ya que el conjunto de disco de rescate de YaRD recordará todos los detalles de su sistema.

Si /usr y /home están en el mismo disco en particiones separadas, tendrás que determinar si la partición raíz puede ser reparada hasta un punto de confiabilidad o si necesitará por completo un nuevo disco. En sentido general, errores que se propagan son una segura señal de una segura falla del disco, unos cuantos erro-

res aislados, como es un grupo de sectores dañados concentrados en poco espacio pueden potencialmente ser reparados confiablemente. Aunque sectores dañados pueden ser suficiente causa para corromper un sistema de archivos por completo y detener todo un sistema, se puede reparar recreando la partición con la opción de mapear los sectores dañados, y entonces proceder a restaurar el backup. Esto involucra una decisión del administrador, y envuelve un balance de considerar entre presupuesto de hardware a gastar y tiempo fuera de línea versus confiabilidad de reparar. Reparando la partición raíz puede llevarse a cabo en considerablemente menor tiempo que físicamente reemplazando el disco y restaurar todas estas particiones. Si la decisión de reemplazar el disco se presenta necesaria entonces tendrás que hacer las cosas en el siguiente orden:

- Desmonte todos los sistemas de archivos que residen en el mismo disco (/usr, /home, etc).
- Ejecute fsck -n a todas las particiones. Si encuentra errores en estas también, Heche un vistazo a las secciones del man que se refieren a la reparación manual de un sistema de archivos.
- Si no se reportan ningunos errores, efectué un backup inmediatamente a estas particiones.
- Instale el nuevo(s) disco duro.
- Reinicie desde su disco de rescate.
- Restaure todas sus particiones y ejecute el comando lilo sobre las partición raíz (/) recién restaurada.

Recuperando desde Fallas Eléctricas

Aunque estemos protegidos con todo tipo de seguridad en caso de fallas eléctricas tenemos que siempre estar preparados por si alguien equivocadamente tocara el botón de encendido u ocurriese una falla de energía. Al reiniciar el equipo después de una falla eléctrica lo más seguro es que el utilitario fsck detectará una gran número de errores en su sistema de archivos una vez lo reinicie. En la mayoría de los casos éstos errores son fácilmente reparados por fsck.

De vez en cuando el comando fsck encuentra errores en los discos que no pueden ser automáticamente reparados. Cuando éstos errores son encontrados, la ejecución de los scripts de inicio son detenidos y se le pide ingresar el password de root para darle acceso a un shell de modo protegido (o de usuario único). Desde éste punto usted deberá ejecutar el comando fsck -n a los sistemas de archivos que fueron afectados por la falla, para ver si es posible hacer una reparación rápida. El fsck analizará el sistema de archivos y le preguntará por la acción a tomar en base a los problemas que encontrados. La opción -n le dice a fsck que no escriba los cambios al sistema de archivos que simplemente le presente los cambios que se hicieran. Esto puede darle un buen parámetro de que tan serio ha sido el daño causado al sistema de archivos. Si el daño es extenso, puedes intentar reparar manualmente el sistema de archivos antes de considerar restaurar desde el backup.

Algunas cosas que debes tener presente antes de proceder a reparar el disco manualmente son:

- ¿Qué tan critica es la información en éste sistema de archivos? En sistema de archivo como el de /usr, con información estática, quizás sea mejor simplemente restaurar los archivos del sistema desde la copia.
- ¿Cómo pueden mis esfuerzos de reparación causar aún más daño al sistema de archivos? Puede ser que usted tenga éxito en la restauración del sistema de archivos a costo de la destrucción aún más datos. Simplemente restaurar todo puede que resulte una opción menos riesgosa.
- Necesitaras un nivel avanzado de entendimiento del sistema de archivos para satisfactoriamente realizar la reparación manual. Si fsck no puede repararlo automáticamente, generalmente es una indicación de un problema serio del sistema de archivos que requiere un análisis más cuidadoso.

Repasemos las fases del fsck en detalle.

Fase 1 : Blocks y Tamaños

En esta fase los archivos son verificados que tengan el número correcto de bloques asignados y un pri-

merpaso es llevado a cabo en la búsqueda de archivos cross-linked (archivos que usan los mismos bloques de disco). Se le podría pedir limpiar ínodos o cambiarle el tamaño a un archivo. Ambas acciones son destructivas e irrevocables. Asegúrese que tenga una copia. Generalmente, respondiendo no a algunas de las preguntas es lo mismo que decidirse sobre la reparación manual.

Fase 2 : Nombres de Rutas

La estructura de los directorios habrá cambiado sobre la base de las reparaciones completadas en fase 1. En esta fase el árbol de directorio es hecho consistente. Típicamente daños severos al sistema de archivos requieren purgar, y todos los archivos en el directorio purgado son perdidos. Si puedes montar el sistema de archivos dañado en esta fase, existe una oportunidad de salvar la información antes de dar el próximo paso.

Fase 3 : Conectividad

El fsck verifica directorios que no estén referenciados (aquellos sin parientes) y los conecta a /lost+found.

Fase 4 : Contadores de Referencia

Se lleva a cabo una búsqueda de archivos no referenciados (aquellos sin directorio) y los conecta al directorio /lost+found. También, la cuenta de hard-link para los objetos de sistemas de archivos son recalculados y actualizados. En esta etapa, debes permitirle a fsck que altere los contadores de referencia cuando se le pregunte.

Fase 5 : Lista de Bloques Libres

La lista de los bloques del sistema de archivos no usados es revisada por vínculos cruzados (cross links) a archivos ya existentes, bloques ya en uso, y bloques dañados. Siempre recuerde que debes responder si (yes) a todas las preguntas para obtener un sistema de archivo limpio.

Ejecutar manualmente a fsck es una experiencia muy tediosa, pero trabaja las mayoría de las veces, aunque muchas veces a un costo de un sistema de archivos dramáticamente alterado. Evite problemas en el futuro guardando siempre una copia de seguridad.

LILO (El cargador de GNU/Linux)

Lilo es el LInux LOader utilizado para iniciar GNU/Linux y otros sistemas operativos. Lilo no es dependiente del sistema de archivos de la manera que muchos otros boot loaders dependen. Lilo puede configurarse para iniciar 16 imágenes diferentes. Lilo puede ser instalado en dos lugares diferentes del disco:

- El Master Boot Record (MBR)
- Utilizado si LILO es usado como el bootloader primario.
- Utilizado si GNU/Linux es solamente el sistema operativo en el disco.
- El primer sector de la partición de inicio
- Utilizado si otro boot loader es usado como el boot loader primario.

LILO utiliza un archivo de configuración central de nombre /etc/lilo.conf. Este archivo es de texto y puede ser editado con cualquier editor de texto para poder actualizar y agregar opciones al archivo de configuración. Cada vez que un cambio es realizado al archivo de configuración, LILO debe ser reescrito para que los cambios tengan efecto. Podrás realizar la actualización ejecutando el comando lilo desde la línea de comando en su distribución de GNU/Linux.

La gran mayoría de de distribuciones de GNU/Linux pueden ejecutar LILO satisfactoriamente. Si por alguna razón no puedes o no desea ejecutar LILO en sus sistema, existen muchos programas boot loaders disponibles. Por ejemplo, GRUB, el cargador de GNU/Linux que viene con casi la totalidad de distribuciones de GNU/Linux modernas.

Los siguientes tópicos serán discutidos en la siguiente sección:
- El Proceso de Inicio/Boot
- LILO sus Opciones desde la Línea de Comandos
- LILO Archivo de Configuración (/etc/lilo.conf)
- LILO sus Errores
- Como Recuperarse del Uso inapropiado de LILO

El Proceso de Inicio/Boot

Los sistemas GNU/Linux necesitan un interfáz estándar denominada bootstrap. El bootstrap de más bajo nivel de una máquina es dependiente del hardware. Comúnmente el BIOS del sistema ejecuta el cargador del sistema (LILO/GRUB) desde el MBR del dispositivo por defecto de inicio.

El programa del bootstrap debe ser capaz de inicial a Linux a modo de usuario-simple o multi-usuario y diferente configuraciones del sistema operativo. Esta información es suplida por el usuario desde la línea de comandos.

La mayoría de los sistema permiten que la información por defecto del bootstrap sea definida y que permanentemente sea almacenada para así permitir que cada vez que el sistema se encienda, y suba automáticamente el bootstrap. Sistemas GNU/Linux almacena programas iniciables/bootable en el directorio /boot, el cual comúnmente se encuentra en el disco por separado.

Después que el programa del bootstrap a cargado el kernel, en memoria, le rinde el control al sistema. El sistema GNU/Linux procede a inicializar los dispositivos físicos, controladores de memoria virtual, y sus tabla de control interna para los procesos, archivos, etc.

Opciones de la Línea de Comandos de LILO

LILO nos ofrece varias opciones que pueden ser utilizadas para modificar los parámetro de instalación y para la resolución de problemas.

lilo -r	Especifica un directorio diferente a escoger como directorio raíz; muy útil durante una emergencia.
lilo -t	Prueba la configuración sin actualizar el cargador del sistema.
lilo -C	Especifica un archivo de configuración diferente: útil a usar la opción -t.
lilo -v	Establece el modo verbose, devolviendo más detalle sobre las respuesta de LILO.

El programa LILO también puede ser usado para examinar los parámetros de inicio/boot:

lilo -q	Esta lista los archivo mapeado actualmente. LILO mantiene un archivo, por defecto, /boot/map, que contiene el nombre y la localidad del kernel a iniciar/boot. Esta opción lista los nombres contenido en el archivo map.

Archivo de Configuración de LILO (lilo.conf)

El archivo usado para configurar LILO se llama lilo.conf. Este contiene las opciones globales, seguidas por opciones individuales de la imagen. Aquí un ejemplo típico de /etc/lilo.conf:

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
```

```
timeout=50
image=/boot/vmlinuz-2.0.36-0.7
label=linux
root=/dev/hda5
read-only
```

Esta es una explicación de algunas de las opciones más comunes de lilo.conf. La mayoría de estas aparecen en la sección global del archivo de configuración, las otras pueden aparecer tanto en la sección global como en la sección de imagen individual.

boot	Este especifica el nombre de el dispositivo que contiene el sector de arranque
default	Este especifica la imagen de boot por defecto, por lo tanto, la primera imagen accesible es tratada como la por defecto.
línear	Este usa sector de dirección líneal a diferencia de el cilindro estándar/cabeza/dirección de sector.
map	Este especifica el archivo de map a usar en esa localidad.
message	Este especifica un archivo que contiene un mensaje que será mostrado cada vez que el sistema éste arrancando (booting). Este mensaje es desplegado antes de el prompt de LILO y es regularmente usado para proveer instrucciones o una lista de imágenes de arranque (boot).
timeout	Este especifica el el tiempo que dura en désima de segundo que el LILO espera por una entrada de usuario antes de cargar la imagen por defecto.
vga	Este especifica el modo de video que se usara cuando la maquina esta en el proceso de arranque. Tu puedes ingresar un texto valido o seleccionar una de las otras opciones que están disponibles. • ask el prompt del modo texto a usar • normal especifica el modo texto 80 x 25 • extended especifica el modo texto 85 x 50

Cada imagen individual debe de tener algunas de las siguientes opciones

alias	Este especifica otro nombre al que la imagen debe de hacer referencia
label	Este especifica el nombre de una imagen particular de arranque
password	Este especifica el lugar donde será insertado el password antes cargar la imagen particular.

LILO ofrece muchas opciones especiales para los kernels de Linux. El más común será detallado a continuación:

append	Este especifica opciones que pueden ser pasadas al kernel. Este es comúnmente usado para forzar al kernel a usar configuraciones particulares de hardware, tales como un tamaño de memoria o especificar la geometría de un disco duro.
initrd	Este especifica la ruta que será cargada hacia la RAM disk. La RAM disk inicial son comúnmente usados en sistemas con discos duros Small Computer System Interfáz (SCSI)
read-only	Este especifica que archivos del sistema de archivos de root debe ser montado como de sólo lectura (read-only). Este debe ser usado en la mayoría de los casos porque usualmente GNU/Linux remonta el sistema de archivo lectura/escritura después de haber realizado un fsck de inicio.

Errores de LILO

Ocasionalmente su sistema confrontará problemas iniciando a GNU/Linux. Esta experiencia ocurre muy a menudo al instalar una nueva versión del kernel, agregar o remover discos duros, o al instalar otro sistemas operativos a la vez con GNU/linux en los denominados multi-boot. Algunos errores comunes incluyen:

<nada>	LILO no está cargado, no éste en la partición boot o no esté activo el sector de la partición boot
L	La primera etapa del cargador de inicio esta cargada e iniciada pero no puede encontrar la segunda etapa del cargador. Códigos de errores numéricos impresos en la pantalla pueden indicar falla del medio o error de geometría del disco.
LI	La segunda etapa del cargador subió pero no se inicio.
LIL	La segunda etapa del cargador ejecutó pero no puede cargar la tabla descriptora desde el archivo map
LIL?	La segunda etapa del loader esta cargada de manera incorrecta.
LIL-	La tabla descriptora esta corrupta. Puede ser causado o por una discordancia de geometría o mover el directorio /boot/map sin ejecutar el instalador del map.
LILO	Todo esta cargado correctamente.

Después de compilar e instalar un nuevo kernel, el cargador de boot debe ser actualizado para así poder usar el nuevo kernel. Kernels modernos de Linux tienen muchos componentes opcionales para piezas particulares de hardware, y muchas incrementan el tamaño del kernel significativamente. Puede ser que LILO reporte el tamaño del kernel ser muy grande. En éste caso, podrías alternar entre utilizar make zimage y utilizar make bzimage. Si ambos reportan el mismo error, deberás considerar su configuración del kernel y crear un número de los componentes del kernel como módulos y no compilarlos directamente en el kernel.

Recuperarse del Uso Inapropiado de LILO

Quizás compilaste e instalaste un nuevo kernel, y ejecutaste el comando lilo con las opciones equivocadas o con una sintaxis correcta pero con la información errónea en el archivo /etc/lilo.conf. Afortunadamente estas acciones son fáciles de corregir.

- Inicie con el disco de rescate y monte la partición raíz de su sistema en el directorio /mnt/disk o donde desee. Si /etc/ está en una partición separada, montela también, por ejemplo /mnt/disk2.
- Cambie hacia el directorio /etc/ (/mnt/disk2/etc). Fíjese que /etc se refiere al del disco de rescate y no al del sistema que montamos.
- Edite el lilo.conf del sistema a recuperar para corregir los errores del LILO.
- Ejecute lilo con los argumentos correctos para root, archivo map, archivo del sector de inicio y el archivo de configuración. Deberías entender como lilo trabaja con los archivos map, archivos sector de inicio e imágenes del kernel para evitar daños a su disco de rescate.

La Cuenta Root

La cuenta root es la cuenta del administrador y no hay limitaciones para éste usuario. Por razones de seguridad, un administrador raramente debería ingresar como root en el sistema. El administrador debiera ingresar como un usuario normal y entrar como superusuario, así minimizando los riesgos de daños al sistema inadvertidamente. Esto es lo primero y más veces repetido: ¡No use la cuenta root habitualmente!. Sólo se debe emplear para realizar tareas administrativas. Si se comete un error siendo root, el sistema no le detendrá. Si tienes la duda de como quien estas logueado en el sistema puedes escribir el comando “id” el cual regresara su Efective User ID (EUID). Similarmente, el comando whoami regresa el usuario efectivo.

La formula habitual, si se requieren los poderes del root, es utilizar el comando «su». Este comando ejecuta una shell con identificadores de usuario y grupo distintos. Lo que quiere decir que permite a un usuario convertirse temporalmente en otro usuario. Si no se especifica ningún nombre de usuario, por defecto se usa root. El shell a ejecutar se toma de la entrada correspondiente al usuario en el fichero de contraseñas, o /bin/sh si no está especificada en dicho fichero. La ejecución de “su” solicitará la contraseña del usuario, a menos que se ejecute por el root.

Muchas funciones de los sistemas de administración son realizados utilizando la cuenta root. Root no tiene restricciones de acceso a ningún lado. Algunas cuentas adicionales son utilizadas para administrar sub-

sistemas. Debe utilizar estas cuentas para asegurar que las propiedad de los archivos y sus permisos son correctos para los subsistemas.

El administrador será también el encargado de la realización de cambios en el sistema. Cualquier cambio deberá ser cuidadosamente planificado y probado en un entorno controlado antes de implantarlo en producción. Es imposible evaluar todas las posibles variables y condiciones que se generarán al llegar los usuarios al día siguiente de realizar ese cambio tan importante. Pero inténtelo, la responsabilidad es suya.

El administrador debe ser el que provee las soluciones, si su actividad genera problemas, tenga previsto como restaurar una condición estable en el mínimo lapso de tiempo. No sólo debe pensar como implementar el nuevo cambio, además tenga previsto un plan de recuperación de emergencia, por si algo sale mal. En Sistemas de muchos hosts y servidores, la administración podría ser realizada por varias personas. Es imperativo que multiples administradores coordinen sus actividades. Es posible que una persona deshaga o corrompa el trabajo hecho por otro.

También es buena idea plantear gradualmente las mejoras, es más fácil volver atrás si los cambios son menores que cuando se cambian de golpe gran número de cosas. No sabrá que falló ni porque si hizo todos los cambios a la vez. Y no digamos si afectan a gran parte del equipo. Podrá diagnosticar un problema si sólo tocó un par de cosas y comprobó el funcionamiento antes de seguir adelante.

Cada cambio debe ser anunciado con suficiente antelación cuando afecte a los usuarios, de manera que sepan como comportarse ante la novedad. El mejor sistema, funcionando correctamente puede ocasionar una pequeña revuelta si los usuarios ven afectado su trabajo, aunque sólo hablemos de pequeños cambios de forma. Infórmeles con antelación de lo que va a ocurrir y documente, si es necesario, las nociones sobre el manejo (o lo que sea) que necesiten.

Sin importar si las maquinas están en un área de acceso restringida (como una sala de computadores), Nunca puede ingresar en la consola del sistema como root. Algunos administradores inhabilitan el acceso como root en otras terminales para prevenir que multiples usuarios trabajen como root en el sistema. Esto podría ser una buena idea, la consola se bloqueará y no habrá manera de trabajar como root, así seria una buena idea dejar por lo menos una u otra terminal con permisos de acceso restringido de root.

Configuración Del Sistema

Una vez el sistema éste ya instalado e iniciado. Tendrá aún algunas tareas que necesitan ser realizadas, como son administración de usuarios, configuración de runlevels y así sucesivamente. GNU/Linux provee herramientas de líneas de comandos y GUI para realizar de manera fácil estas tareas.

Cubriremos los siguientes tópicos en esta sección:

- Agregando Usuarios
- Utilidades de Configuración

Agregar Usuarios

Los utilitarios useradd y adduser estas disponibles en casi todas las distribuciones GNU/Linux y son recomendados para administración de usuarios. Ellas permiten modificar los archivos de configuración sin tener que editarlos manualmente. Si estas herramientas fallan tendrá que editar el archivo /etc/passwd manualmente para asegurarse que no fueron corrompidos por la falla de la aplicación.

Algunas de las opciones con el comando useradd son mostradas en la siguiente tabla. Los por defecto están especificados en /etc/defaults/useradd y serán utilizados para modificar las cuentas.

-u uid	Especifica un nuevo UID
-g group	Especifica el grupo por defecto
-c comentario	Descripción del usuario (por defecto esta en blanco)
-d dir	Directorio Home
-m	Crea el directorio home (es recomendado por defecto el nombre del usuario
-k directorio skel	Esqueleto del directorio home (/etc/skel por defecto)
-s shell	Especifica el programa de ingreso (/bin/bash por defecto)

En algunos sistemas el archivo login.defs también jugará un rol importante en la configuración de las cuentas de los usuarios. Recuerde que el comando useradd puede asegurar que el directorio home tenga todos los permisos establecidos correctamente; simplemente ejecutará los comandos chown y chgrp. De cualquier otra forma, los permisos tendrán que ser establecidos manualmente.

El directorio (/etc/skel) esqueleto nos permite configurar una estructura molde (template) que podemos aplicar a todos los usuarios que se crearán. Al ejecutarse acompañados con la opción -k, el comando useradd, copiará el contenido del directorio skel al directorio home del usuario. Todo esto y a la vez se asegurará de mantener los propietarios y permisos intacto. Podemos mantener más de un directorio esqueleto.

Utilidades de Configuración de GNU/Linux

Los utilitarios de configuración de los sistemas GNU/Linux son herramientas poderosas para la administración del sistema. Ellas proveén una ubicación centralizada para la completa administración del sistema, muchas de ellas puede correr bajo modo texto o gráfico. Ambos métodos realizan operaciones idénticas. Aquí le diremos algunas de la que están más avanzadas en su desarrollo, pero ninguna de ellas pueden ejecutarse fuera de sus distribuciones. El único intento que se hizo de hacer una herramienta unificada que se ejecute desde cualquier distro fué WEBMIN, pero no viene con ningún distro por defecto.

Los utilitarios del Centro de Control

Mandriva GNU/Linux

El Centro de Control de Mandriva GNU/Linux es la herramienta principal de configuración de Mandriva GNU/Linux. Permite que el administrador del sistema configure el hardware y los servicios utilizados por todos los usuarios. Las herramientas que se acceden por medio del Centro de Control de Mandriva GNU/Linux simplifican muchísimo el uso del sistema, en particular al evitar el uso de la línea de comandos para los administradores novatos.

Encontrará su icono en el panel de su administrador de ventanas. También puede acceder al Centro de Control de Mandriva GNU/Linux eligiendo Sistema+Configuración->Configurar en su computadora en el menú principal.

¿Ejecuta Desde el CLI?

Sí, el Centro de Control de Mandriva GNU/Linux también está disponible desde la línea de comandos en modo texto ejecutando drakconf.

SuSE YAST2

El Centro de Control de SuSE, conocido como el YaST2 es la herramienta principal de instalación y configuración de SuSE. Permite que el administrador del sistema configure el hardware y los servicios utilizados por todos los usuarios y además es el instalador de SuSE GNU/Linux. Las herramientas que se acceden por medio del Centro de Control simplifican muchísimo el uso del sistema, en particular al evitar el uso de la línea de comandos para los menos expertos.

Encontrará su icono en el panel de su administrador de ventanas. También puede acceder al Centro de Control eligiendo Sistema+Configuración->Configurar su computadora en el menú principal.

¿Ejecuta Desde el CLI?

Sí, el Centro de Control YaST2 también está disponible desde la línea de comandos en modo texto y también con las librerías ncurses, ejecutando yast2.

Redhat/Fedora setup

El Centro de Control de RedHat/Fedora, conocido como el setup es la herramienta principal de configuración de Redhat y Fedora. Permite que el administrador del sistema configure el hardware y los servicios utilizados por todos los usuarios. Las herramientas que se acceden por medio del Centro de Control simplifican muchísimo el uso del sistema, en particular al evitar el uso de la línea de comandos para los menos expertos.

Encontrará su icono en el panel de su administrador de ventanas. También puede acceder al Centro de Control eligiendo Sistema+Configuración->Configurar su computadora en el menú principal.

¿Ejecuta Desde el CLI?

Sí, el Centro de Control setup también está disponible desde la línea de comandos en modo texto y también con las librerías ncurses, ejecutando setup.

RESUMEN

En éste capítulo, usted fué introducido a los siguientes temas relacionados con iniciar y apagar su sistema GNU/Linux así como diagnosticar y reparar (troubleshoot) su sistema cuando un problema se presenta en el proceso de arranque. Se cubrió lo siguiente:

- Los recursos de información disponibles son Libros, HOWTOs, internet, páginas del man e info.
- Mantener logs de los cambios efectuados al sistema puede ser de ayuda al resolver problemas posteriores.
- Los comandos kill y killall pueden terminar un programa que no responde.
- Los comandos ps y top nos dan información sobre los procesos en ejecución.
- Si se compiló en el kernel, la tecla SYS RQ puede ser usada para recuperar el sistema de errores que pueden colgarlo.
- Use lilo para iniciar el arranque, y configure el arranque con /etc/inittab.
- lsmod lista los módulos que están cargados en el kernel.
- Discos de Inicio permiten que un sistema sea arrancado cuando hay problema con el sector de arranque de sus discos.
- Debería tenerse mucho cuidado al utilizar la cuenta root.
- Los comandos adduser y useradd agregan usuarios al sistema.

PREGUNTAS POST-EXAMEN

Las respuestas a estas preguntas se proveen en el Apéndice A.

1.- ¿Qué significa la cuenta privilegiada de root?

2.- ¿Cómo puede un usuario cambiar su EUID ? y ¿Cómo afecta esto al usuario?

3.- ¿Porqué no permanecer como root en el sistema todo el tiempo?

4.- ¿Dónde puedo encontrar los archivos log del sistema?

5.- ¿Cuál es una de las razones para crear un disco de emergencia?

CAPITULO 6

ADMINISTRACIÓN DEL SISTEMA OPERATIVO GNU/LINUX

EL SISTEMA DE ARCHIVOS

OBJETIVOS

Al completar este capítulo, usted podrá:

- Contrastar los diferentes recursos del sistemas de archivos
- Usar las herramientas básicas para navegar en el sistema de archivos
- Describir como los permisos del sistema son usados para controlar acceso
- Describir la causa y solución de errores de lectura.

PREGUNTAS PRE-EXAMEN

Las repuestas se encuentran en el Apéndice A.

1. Debe remover una cuenta de un usuario que ya no trabaja en la compañía. ¿Qué comando usaría usted para remover la cuenta, si esta cuenta tiene su directorio home en /home/usuario? ¿Cómo verificaría usted que fué realmente eliminada?

2. ¿Es el directorio /proc un sistema de archivo virtual o un sistema de archivo físico?

3. ¿Qué es un nombre de ruta relativo?

INTRODUCCION

En éste capítulo discutiremos los conceptos básicos del sistema de archivos y como esto aplica a sistemas GNU/Linux. Hay ciertos puntos clave que debe entender de los sistemas de archivos en general:

- Los sistemas de archivos son estructuras de datos colocados dentro de una partición (un segmento de espacio en un dispositivo que esta asignado para ser usado por un sistema de archivos).
- Diferente sistemas operativos a menudo utilizan sistemas de archivos distintos. Existen diferencias entre GNU/Linux y otros sistemas operativos (y otros UNiX y sus clones) y sus sistema de archivos, lo que incluye la ausencia de letras para denotar las unidades de discos y el acceso a dispositivo através de entradas del sistema de archivos.
- Sistemas de archivos de redes casi siempre contienen una habilidad inherente de limitar el acceso a destino específicos, como un usuario o grupo de usuarios. GNU/Linux emplea el método de acceso a los sistemas de archivos con los estándares usados por todos los UNiX y sus clones.

Sistema de Archivos Jerárquico

El Sistema de Archivos de GNU/Linux tiene una estructura jerárquica. Este sistema de archivo es como un único árbol cubriendo toda la data del sistema. Los archivos son organizados utilizando directorios en una estructura jerárquica.

En esta sección cubriremos los siguientes tópicos:
- Nombres de Archivos
- Nombres de Rutas
- Usuarios, Grupos y Archivos.

Nombres de Archivos

Los nombres de archivos obedecen un gran número de convensiones. Estos pueden contener cualquier caracter ASCII excepto la barra (/), la cual es usada para separar directorios dentro de una especificación de nombre de ruta. Normalmente no encontrará muchos archivos con nombres que contienen (:;-). Los nombres de archivos de GNU/Linux no contienen una extensión, aunque puedes añadir una si así lo desea, no como en las versiones de DOS y VMS (Virtual Memory System). Usted puede agregar más de una extensión a un nombre, el ejemplo más común que puede encontrar son los archivos .tar.gz.

A diferencia del DOS, el cual requiere de 1 a 8 caracteres para un nombre, como una opción de 1 a 3 caracteres para la extensión, no existe tal restricción de formato en los nombres de archivos de GNU/Linux.

Los nombres de archivos de GNU/Linux, siguen la convención de separar el tipo de archivo del nombre interpuesto por un punto. Esto le permite a muchas herramientas de GNU/Linux generar un archivo de salida desde un archivo de entrada. El compilador C, por ejemplo, está a la espera de un archivo que termine con la extensión .c y generá similarmente un archivo de nombre a.out, al menos que se use la opción -o para que genere un arcivo del nombre que usted elija.

prog.c compila como: prucj.o

Este sistema es una pura convensión y realmente GNU/Linux no pone restricción alguna en el formato del nombre del archivo. Estos empiezan con un punto como por ejemplo: ***.profile***

Que son archivos ocultos, normalmente no son desplegado al listar el contenido del directorio y típicamente contienen valores personalizado de la aplicaciones.

Aunque cualquier caracter es permitido en el nombre del archivo. En la práctica, existen caracteres que

pueden causar gran dificultad, éstos deben ser evitado siempre y cuando sea posible. Ejemplos son:

- Un signo de menos (-) como el primer caracter de un nombre de archivo
- Caracteres como son ?,*,(,), &, [,],>, <, el espacio y el tab (ya que éstos tienen un significado especial en la línea de comandos)
- Caracteres ASCII no imprimibles.

Nombre de Ruta

Los archivos son localizados incorporando un esquema de nombre de directorio a través de una ruta que lleva hasta el nombre del archivo individual. Esto entonces es llamado un nombre completo calificado de nombre de archivo.

- Una ruta de nombre absoluta describe una ruta del nombre del archivo o directorio empezando desde el directorio raíz (/).

  /usr/bin/tty
- Una ruta de nombre relativa describe una ruta del nombre del archivo o directorio empezando desde directorio actual.

  bin/tty
- El directorio actual será el punto de referencia si el nombre del archivo no empieza con una / .

Una ruta de nombre absoluta describe una ruta del nombre del archivo o directorio empezando desde la raíz (/), del sistema de archivo porque el sistema de archivo es organizado como un árbol con una sola raíz. Todo archivo o directorio tendrá exactamente un nombre de ruta absoluta y por esto puede ser visto de una manera de indentificación única de un archivo o directorio dentro de un sistema de archivo.

Una ruta de nombre absoluta empieza con una barra (/). De ahí en adelante, está compuesto por nombre de directorio separado por barra, por ejemplo /usr/bin/tty, es el nombre de ruta absoluta al archivo almacenado a la ruta /usr/bin/ de nombre tty.

Un nombre de ruta relativo describe la ruta de un archivo o directorio empezando desde el directorio actual. El formato del nombre de una ruta relativa es muy similar al de la ruta absoluta, excepto que el relativo no empieza con una barra.

El ejemplo nos muestra que cuando el directorio es /usr, entonces la ruta relativa fuese bin/tty usted puede ver que si el directorio actual fuese /usr/bin el nombre de ruta relativo, fuese solamente tty.

Vínculos a Directorios

Todo directorio en el sistema de archivo de GNU/Linux tiene dos entradas especiales. Aquí mostramos estas dos entradas:

. **El directorio actual**
.. **El directorio padre(un paso más en la jerarquía)**

Estos nombres pueden ser usados en los nombres de ruta, ya sean relativas o absolutas, son usados mayormente en nombre de ruta relativas, ya que te permiten especificar rutas que navegan a través de la jerarquía de archivos.

Si nuestro directorio actual es /home/usuario, desde ahí la ruta relativa al directorio /home es (..) y la ruta relativa a /home/usuario/2 es ../usuario/2. Similarmente podemos especificar una ruta relativa a un archivo a nuestro directorio actual, por ejemplo un archivo en mi directorio actual “carta.txt”, puede ser referenciado como ./carta.txt. Esto es muy útil cuando el nombre del archivo empieza con el símbolo de menos, porque

los comandos tratan el símbolo de menos como una opción, simplemente anteponemos ./ para eliminar cualquier confusión.

El Directorio Home

Por lo general, todo usuario del sistema tiene un directorio home. He aquí donde el usuario primero ingresa al sistema. El directorio home contiene los directorios de los usuarios y éstos contienen sus archivos. Aquí el usuario puede crear libremente archivos y directorios. El contenido del directorio HOME está protegido por defecto de los usuarios que no sean el usuario dueño o el super-usuario con los permisos de acceso especiales.

Archivos y directorios de configuración son colocados en el directorio home de los usuarios. Aquí se incluyen archivos de personalización de sección como son .profile o .login, además, los archivos de inicio de configuración de las aplicaciones como son .Xdefaults y .mailrc.

Comando de Directorios

Algunos comandos simples de directorios que son útiles, son incluidos en la siguiente tabla:

Comando	Descripción
cd <directorio>	Cambiar directorio
cd	Cambiar directorio home
pwd	Imprimir directorio actual
ls	Listado corto
ls -l	Listado Largo
ls [opciones] [archivos...]	Listar archivos específicos del directorio

Use el comando cd <directorio> para cambiar del directorio actual y use el comando pwd para imprimir la ruta completa del directorio actual. Use el comando cd sin ningún argumento y lo llevará a su directorio home.

El comando ls lista archivos en un directorio. Este puede ser ejecutado con una lista de nombres de archivos y directorios; si no se pasa el nombre de un archivo como argumento, listará el directorio actual.

Una opción muy útil del comando ls es -l. Esta opción muestra, además del nombre de archivo, muchos de los atributos del archivo:

```
$ ls -l
total 101328
-rw-r--r-- 1 ivelis admin 21448 21 Jan 13:19 2787778_2023.jpg
-rw-r--r-- 1 ivelis admin 16816 15 Dec 17:46 2do_lugar.jpg
-rw-r--r-- 1 ivelis admin 16409 15 Dec 17:47 3er_lugar.jpg
-rw-r--r-- 1 ivelis admin 347819 30 Dec 15:06 BreakingWindows-print.png
-rw-r--r-- 1 ivelis admin 884400 9 Dec 10:02 CHARLA.psd
-rw-r--r-- 1 ivelis admin 53885 26 Jan 13:51 CHARLA_AWK.gif
```

He aquí unas cuantas de las opciones que se le pueden pasar al comando ls:

-l	Muestra el listado largo (FULL)
-a	Lista todos los archivos (incluyendo los archivos ocultos que inician con .)
-C/-x	Lista en columnas ordenadas/cruzadas
-F	Marca el tipo de archivo
-R	Listado directorios recursivamente
-t	Ordena por tiempo y no por nombre
-d	Lista el directorio mismo no su contenido

La opción -a del comando ls listará los archivos ocultos así como los archivos normales, otra opción muy útil del comando ls es -d, cuando es utilizada en un directorio, esta muestra los atributos del directorio especificado y no el contenido de éste. En los siguientes ejemplos note la diferencia entre las 2 opciones:

```
$ ls  -l     /tmp
$ ls  -ld    /tmp
```

Otro uso de la opción -d es con comodín (wildcards), por ejemplo:

```
$ ls  -d    mi*
```

Nos devolverá el nombre de cualquier entrada en el directorio actual que comienza con mi. Sin la opción -d estuviese listado el contenido de cualquier directorio que empezará con mi.

La opción -R nos devuelve un listado recursivo del directorio. Un listado recursivo significa que no sólo nos devolverá el directorio, sino que también todos los subdirectorios dentro de éste hasta el fondo de la estructura del albúm.

Listado de Directorio Detallada

Es muy útil poder obtener la información detallada del listado del directorio. Esto nos provee informaciones muy importantes. Use el comando ls -l para obtener esta salida:

```
$ ls -l
total 101328
-rw-r--r-- 1 ivelis admin 21448 21 Jan 13:19 2787778_2023.jpg
-rw-r--r-- 1 ivelis admin 16816 15 Dec 17:46 2do_lugar.jpg
-rw-r--r-- 1 ivelis admin 16409 15 Dec 17:47 3er_lugar.jpg
-rw-r--r-- 1 ivelis admin 347819 30 Dec 15:06 BreakingWindows.png
```

Copiar Archivos

Para copiar un archivo:

- Use el comando cp para copiar archivos a un archivo de destino o a un directorio:
  cp [opciones] origen destino
  cp [opciones] origen... destino
- Algunas opciones útiles son:
  -i revisar interactivamente (confirma sólo si existe el destino)
  -r copia directorios recursivamente
- Si más de un archivo de origen es especificado entonces el destino deberá ser un directorio.

Para copiar archivos se usa el comando cp. Si el archivo destino es un directorio, todos los archivos de origen serán copiados al directorio; de otra forma el archivo singular de origen es copiado al archivo de destino.

Si el destino ya existe y usted no tiene permisos de escritura, éste no será sobre-escrito. Será necesario una de dos cosas, o borrar el archivo o cambiar su permisos antes de sobre-escribirlo.

Todos los sistemas GNU/Linux soportan la opción -r. Si por alguna razón la suya no lo soporta podrá usar los comandos de archivar (tar y cpio).

Renombrar y Mover Archivo

Para renombrar (o mover) un archivo se utiliza el comando mv, comando que mueve un archivo desde una localidad en la jerarquía del sistema de archivos a otra. Si el destino es un directorio, el archivo se mueve dentro del nuevo directorio, sino entonces el archivo es renombrado. El nombre nuevo, claro está

puede incluir una ruta hacia un directorio y entonces tendrá el efecto de mover y renombrar el archivo.

Para cambiar el nombre de un archivo:

- Use el comando mv para renombrar o mover archivos a destino de otro archivo o un directorio.
  mv [opciones] origen destino
  mv [opciones] origen ... directorio

- opción muy útil es:
  -i Revisado Interactivo (pide confirmación si el archivo destino ya existe)

- Si se especifica más de un archivo de origen, entonces el destino deberá ser un directorio. Si no lo es, perderá data. El comando mv le advertiría sobre esto.

- El comando mv es recursivo automáticamente:
  $ mv archivo1 /tmp/ejemplo
  $ mv -i archivo2 /tmp/ejemplo
  mv: replace ‘’/tmp/ejemplo’?

Si el archivo de destino existe, y no es un directorio éste será sobre-escrito por el archivo movido. Para prevenir sobre escribir por accidente un archivo existente use la opción -i. Se le pediría confirmación para ejecutar esta acción.

Archivos de destino pueden ser sobre-escrito aunque los permisos de escritura no estén disponible. Esto es porque sobre escritura he visto como la eliminación de un archivo y la creación de uno nuevo, así pues los permisos del directorio, no los del archivos, son aplicables. Pero, los permisos del archivo son tomados en cuenta, y a un usuario se le pediría confirmación interactivamente si el archivo de destino no tiene permisos de escritura. La sintaxis de los comandos mover y copiar son muy similares.

Eliminar Archivos

Los archivos son eliminados con el uso del comando rm. El comando rm pedirá confirmación antes de eliminar un archivo protegido de escritura, también pedirá confirmación de todos los archivos a eliminar si se utiliza la opción -i.

Use el comando rm para eliminar o borrar archivos:
rm [-ir] file1 [file2...]
La opción recursiva (-r) es utilizada para eliminar un directorio con todo su contenido incluyendo todos sus subdirectorios. Sin la opción de recursión rm no eliminará un directorio aunque esté vacío. Use el comando rmdir que se describirá a continuación para eliminar directorios vacíos.

- Para eliminar archivo(s):
  -i Revisado Interactivo (pide confirmación para eliminar cada archivo)
  -r Desciende Recursivamente por los Directorios
- Los permisos del directorio que contiene los archivos es de crucial importancia.
- Los permisos del directorio son usados para determinar si es permitido borrar o no.
- Los permisos del archivo son utilizados pero sólo para confirmación adicional.
- Los archivos pueden ser removidos si los permisos del directorio así lo permiten.
  $ rm ejemplo
  $ rm -i ej*
  $ rm -r /tmp/ejemplo

Manipular Directorios

Para crear nuevos directorio se utilizar el comando mkdir y para eliminarlo el comando rmdir. Ambos comandos especifican el nombre del directorio, con ruta absolutas o relativas. Para remover un directorio deberá estar vacío. La opción -r del comando rm borrará directorio y todo su contenido. Al usar esta opción tenga mucho cuidado.

Los directorios son vistos como archivos por la mayoría de comandos, así pues el comando mv puede ser usado para renombrar un directorio. Usted puede igualmente mover directorio dentro de otro directorio.

Para crear, borrar, mover y administrar directorio organizados:

- Para crear directorios use:
  mkdir directorio...
- Para eliminar un directorio vacío use:
  rmdir directorio...
- Para eliminar un directorio y todo su contenido recursivamente use:
  rm -r directorio...
- Para copiar un directorio use:
  cp -r directorio_original directorio_destino
- Para renombrar un directorio use:
  mv directorio_original directorio_destino

Desplegar Contenido de los Archivos

El comando cat puede ser usado para desplegar archivos. Su función es concatenar un archivo o archivos y dar salida al resultado, que por defecto es a la pantalla. Si es usado con un archivo como argumento desde la línea de comandos simplemente desplegará su contenido a la pantalla. Para mostrar el contenido de un archivo una pantalla, a la vez deberá utilizar uno de lo comando de paginar de GNU/Linux como son more y less.

El comando wc cuenta palabras, líneas y caracteres de un archivo. Use las opciones -w, -l, y -c para restringir la acción llevada a cabo.

El comando file es usado para que nos indique que tipo de data está almacenada en un archivo. Si usted no está seguro del contenido de un archivo, use éste comando para investigar si contiene algún tipo de texto que puede ser desplegado en la pantalla.

Algunos de los comando que le permitirá ver dentro de los archivos y su sintaxis:

file archivo[archivos..]	Determina el contenido de un archivo (texto, ejecutable, código C, directorio, etc.)
cat archivo [archivo...]	Despliega el contenido de un archivo (concatenar)
more archivo [archivo...]	Despliega el contenido de un archivo una página a la vez
wc [-cwl] archivo [archivo...]	Cuenta caracteres, palabras y líneas de un archivo

Aquí le presentamos ejemplos de su uso:

```
$ file /etc/issue /bin/echo /tmp
/etc/passwd: ASCII text
/bin/echo: ELF 32-bit LSB executable
/tmp: sticky directory

$ wc /etc/issue
18 20 608 /etc/issue
```

Paginar el Contenido de un Archivo

Usted puede listar el contenido de un archivo a la pantalla utilizando los comandos more o less. Estos programas son referidos como paginadores, ya que ellos muestran archivos de texto, una página a la

vez. El comando more antecede al comando less, por esto el comando less tiene más funcionalidad y mejoras como se muestra en la siguiente tabla.

more	less
<space>	Próxima Página i Adelanta una línea
<CR>	Próxima Línea k Retrocede una línea
b	Atrás una página u Retrocede media página
f	Adelanta una página d Adelanta media página
q	Quit/Salir q Quit/Salir
/cadena	Busca cadena especificada /cadena Busca cadena especificada
h	Despliega página de ayuda h Despliega página de ayuda

Links/Vínculos

Un vínculo es una referencia a un archivo o un directorio y típicamente es usado como un atajo a un ítem muy usado. Existen dos tipos de vínculos. Se explicarán en lo que sigue de esta sección. Los dos tipos de vínculos son creados con un mismo comando "ln".

Hard links pueden ser usados sólo para archivos, no directorios, ellos son creados con el comando ln sin pasarle ninguna opción. Ambos, el archivo de origen como el archivo de destino del link deben de estar dentro del mismo sistema de archivos. Ambos archivos se refieren al mismo inodo y la misma data almacenada en el mismo disco.

Symbolic links son creados utilizando el comando ln -s. El archivo destino apunta a la ruta y el nombre del archivo origen no a su data. Vínculos simbólicos son transparente a todos los comandos excepto a ls y rm. El archivo destino recibe un nuevo inodo diferente al del archivo de origen.

He aquí un ejemplo del comando ln:
ln [-snf] archivo destino
Usando la opción -n le indica al comando no sobre escribir nombres de archivo ya existentes.

Un archivo puede tener más de un nombre, un ejemplo obvio es un directorio, el cual tiene por lo menos dos nombres: el nombre del directorio con el cual fué creado y el nombre del directorio "." .

El comando ln es usado para crear nombres alternativos a un archivo. Esto puede ser conveniente para ahorrar espacio en disco o para usar diferentes nombres para referirse a un mismo archivo para cuestiones de versiones de programas.

Algunas de la razones que los usuarios crean vínculos para referirse a los archivos por alias son:
- Nombre de archivos son difíciles de recordar.
- Queremos que un archivo esté en nuestro directorio pero no queremos copiarlo y desperdiciar espacio.

Un ejemplo es, si usted desea usar con mucha frecuencia el archivo /opt/ooffice/bin/ooffice, pudiéramos crear un vínculo en nuestro /home. Al nivel del sistema la mayoría de los vínculos son simbólicos y son utilizados para proveer compatibilidad entre diferentes versiones de sistemas operativos.

En versiones anteriores de GNU/Linux, la gran mayoría de comandos eran almacenados en el directorio /bin/. En los sistemas nuevos esto ha sido reorganizado y éstos comandos ahora se encuentran en los directorios /usr/bin y /sbin/. Pero, para mantener consistencia entre los sistemas encontrarán que muchos de los archivos y directorios anteriores están vinculados simbólicamente a nuevos archivos.

Observando los Vínculos

El número del inodo del archivo vinculado simbólicamente es diferente al del archivo de origen ya que fué creado como una entrada por separado en la tabla de ínodos. Un número de inodo es único solamente dentro de un sistema de archivos en particular, y no de un sistemas de archivos a otro. Vínculos simbólicos permiten referirse a un archivo cruzando las fronteras de un sistema de archivos a otro y no dependen del inodo.

Use el comando ls para ver los vínculos:
-i Incluye el número de inodo en su listado
-lL Para ver los atributos del archivo original vinculados

```
$ touch archivo1
$ ln archivo1 archivo2
$ ln -s archivo1 archivo3
$ ls -il archivo[1-3]
2327379 -rw-r--r-- 2 ivelis admin 0 28 Jan 13:06 archivo1
2327379 -rw-r--r-- 2 ivelis admin 0 28 Jan 13:06 archivo2
2327382 lrwxr-xr-x 1 ivelis admin 8 28 Jan 13:06 archivo3 -> archivo1
$ ls -ilL archivo3
2327379 -rw-r--r-- 2 ivelis admin 0 28 Jan 13:06 archivo3
```

Note como después del número, en el caso del ls -il, que es el inodo, viene una secuencia de caracteres que representan los permisos de acceso del archivo. En esta secuencia vemos que la primera posición de las 10 posible en algunos casos es un símbolo de menos (-) y en otros casos es un (l). El menos significa que es un archivo normal y la l que es un vínculo, y el resto son los permisos de lugar. En el caso de los vínculos los permisos reportados son los del vínculo y no del archivo al cual éste apunta. En el último ejemplo el caso del (ls -ilL) si se ven los permisos del archivo real al cual ellos apuntan.

El comando rm no elimina el archivo; el simplemente elimina el link (hard) al archivo. Si el último hard link es eliminado, entonces el kernel Linux eliminará el archivo ya que éste no tiene ningún vínculo en el sistema de directorio. El usuario real, efectivamente, nunca elimina archivos sólo los vínculos (hard links).

Eliminar vínculos con rm:

- Un archivo es eliminado cuando su último hard link es eliminado.
- Un symbolic link se torna no servible cuando el nombre de ruta a su archivo de origen es eliminado (vínculo colgado).

Hard links	El número del inodo y su tamaño, y los demás atributos del hard links, son iguales. Ellos debieran ser iguales ya que son el mismo archivo referenciado con nombres diferentes que apuntan a el.
Symbolic links	El tamaño de un vínculo simbólico, refleja el tamaño del archivo al cual el hace referencia, no asimismo.

El comando mv no renombra un archivo. El mueve el vínculo a través de la jerarquía de directorio y puede durante el proceso cambia el nombre almacenado en el directorio.

Un vínculo simbólico puede vincular a un archivo en un sistema de archivo diferente. Un vínculo simbólico puede apuntar a un archivo o a un directorio, recuerde que un hard link no puede hacer esto. Un vínculo simbólico apunta al nombre de ruta no al inodo de un archivo, así que eliminar la ruta a la cual el vínculo simbólico apunta resulta en un vínculo colgante. Reemplazar con otro archivo con el mismo nombre de ruta causa que el vínculo simbólico apunte a una nueva data. A diferencia de un hard link que siempre apunta a un mismo inodo, así que renombrar un hard link que apunta a un archivo no tiene efecto en los otros hard link que apuntan a un mismo archivo, ellos aún apuntan a la misma data.

Ejercicios 6-1: El Sistema de Archivos

El propósito de éstos ejercicios es familiarizarse con el sistema de archivos GNU/Linux así como para practicar los comando de manipulación de archivos y directorios. Soluciones para éste ejercicio se pueden encontrar en el Apéndice A.

Aquí introduciremos nuevos comandos como: **touch**, **head**, **tail**, y **grep**. Podrá observar sus efectos durante los ejercicios pero también se asume que usará las páginas man.

1. Este ejercicio le ayudará a navegar el sistema de archivos y listar el contenido de los directorios. La mayoría de las actividades son detalladamente explicadas.
   Identifique su directorio actual con:
   $ pwd

   Liste el contenido del directorio con:

$ ls	Lista los archivos
$ ls -a	Lista todos los archivos incluyendo los ocultos
$ ls - la	Lista todos los archivos incluyendo los ocultos del sistema y sus atributos

   Para listar los archivos del directorio raíz:
   $ ls /
   $ ls -F /

   No es necesario cambiar de directorio para listar el contenido del directorio. Sólo use la ruta absoluta o relativa al directorio.
   Cambiar al directorio /etc, confirme que se encuentra allí, luego liste el contenido del directorio:
   $ cd /etc; pwd
   Es válido especificar varios comandos en una misma línea separados por punto y coma.

   Ahora pruebe :
   $ ls r*
   Lista todos los archivos y directorios que comienzan con la letra “r”.

   El siguiente comando es para evitar que ls liste el contenido de los directorios:
   $ ls -d r*

   Regrese a su directorio home y confirmalo:
   $ cd; pwd

2. Los siguientes pasos le ayudarán a entender la diferencia entre nombres de rutas relativas y absolutas (relativas son nombre de rutas dadas desde una posición dentro de la jerarquía de archivos. Las absolutas son nombre de rutas dados desde el directorio raíz del sistema de archivos; en pocas palabras que empiezan con una /).
   a. Asumiendo que los siguientes directorios existen:
   /home/usuario/trabajo/cartas/nuevas
   /home/usuario/trabajo/cartas/viejas
   /home/usuario/trabajo/oficios
   ¿Puede usted dibujar la estructura de árbol?

   Si su directorio actual es:
   /home/usuario/trabajo/carta/nuevas

   b. Escriba las rutas relativas a éstos directorios:
   /home/usuario/trabajo/cartas/viejas

/home/usuario/trabajo/oficios
/home/usuario/trabajo
¿Cuál es la ruta relativa a su directorio home?

3. Ahora vamos a comparar los dos paginadores de GNU/Linux: less y more.

   Busque el el directorio /etc/X11/ y encuentre el archivo de configuración del X. Despliegue primero con more y luego con less. Una vez éste dentro del paginador utilice la opción h (help) para ver que puede hacer y familiarizarse con los paginadores.

4. Ahora vamos a explorar los atributos de los archivos que copiamos, movemos, etc. También prestaremos un poco de atención a los permisos de acceso a los archivos, aunque no se cubre en éste capítulo. Sólo tome nota que son consultados al usar el comando rm.
   Copie el archivo /etc/passwd a su directorio home, dejándole el mismo nombre de passwd.

   Ejecute un lista detallada de ambos archivos, el original /etc/passwd y la copia que acaba de crear. Observe los atributos: ¿cuáles atributos son retenidos de los del archivo original y cuáles cambiaron?
   Renombre el archivo nuevo que creo, passwd, en su directorio home y llamélo contraseña.

   Ahora deberá crear un un archivo nuevo y conviertalo de sólo lectura:
   $ touch archivo1; chmod -w archivo1
   Copie el archivo y observe los permisos de ambos archivo:
   $ cp archivo1 archivo2
   $ ls -l archivo?
   Note la falta de los permisos de escritura (w). Elimine el archivo con:

   $ rm file2

   Como el archivo retuvo la misma protección contra escritura como la del archivo original, se le pedirá que confirme que desea remover el archivo.

5. Ahora practicaremos crear y eliminar directorios.
   Trasládese a su directorio home para escribir los siguientes comandos. Asegúrese de su ubicación en el sistema de archivos, antes y después de ejecutar cada uno de los siguientes comandos.

```
$ cd ~
$ mkdir trabajos1
$ mkdir trabajos2
$ cd trabajos1
$ mkdir prueba1
$ touch prueba1/archivo1 prueba1/archivo2
$ pwd; ls -l
$ cd ../trabajos2
$ ls -l ../trabajos1/prueba1
$ cd
```

   Ahora, deberá crear otro subdirectorio, prueba2, debajo del directorio trabajo2 en su directorio home. ¿Cómo puede usted especificar desde la línea de comandos esta operación utilizando los nombres de rutas relativas y absolutas?
   Elimine el directorio trabajo1 creado anteriormente. Ejecutando el siguiente comando:
   $ cd; rmdir trabajo1

   ¿Qué pasó? ¿por qué no funcionó? ¿Puede usted eliminar el directorio trabajo1 usando un comando diferente? Hágalo, pero utilice la opción de interactivo para ver el orden en que los archivos son eliminados.

6. Ahora vamos a ver unos cuantos utilitarios para observar los archivos.
   Para ver parte de los archivos podemos utilizar:

   $ head contraseña
   $ tail contraseña
   $ tail -5 contraseña

   Demos un vistazo al tipo de datos contenido en un archivo con el comando:
   $ file * Todos los archivos en su directorio actual.
   $ file /etc/p* Los archivos en el directorio /etc que comienzan con "p'

   ¿Puede usted pensar en una circunstancia en que el comando file le fuera útil?

7. Tomando en cuenta que el archivo /etc/passwd está compuesto por un usuario valido del sistema por línea. ¿Cuántos usuarios validos tiene su sistema?

Ejercicio 6-2: Navegar el Sistema de Archivos

Las Soluciones a éste ejercicio están en el Apéndice A.

1. Vamos a practicar navegar a través del sistema de archivo y explorar una cuantas opciones. Ejecute las siguientes secuencia de comando:

   $ cd **# Asegúrese de estar en el directorio home**
   $ mkdir -p dir1/dir2 **# ¿Qué efecto tiene la opción -p?**
   $ cp /etc/passwd dir1/dir2
   $ rm -r dir1 **# Responda "no" para ver que sucede**
   $ rm -rf dirl **# ¿Qué efecto tiene la opción -f?**

   Repita el proceso completo, pero en vez de copiar el archivo de /etc/passwd, crea uno que le pertenezca:
   $ mkdir -p dirl/dir2
   $ touch dirl/dir2/archivo1
   $ rm -r dir1
   ¿Cuál fué la diferencia esta vez? ¿Por qué no fué la opción -f?

2. Esto le servirá al comando de búsqueda de patrones grep. El comando grep es un utilitario poderoso, capaz de leer archivo de texto. Su función es desplegar aquellas líneas de un archivo que contengan la cadena de caracteres dada como argumentos.
   Busquemos cadenas de caracteres dentro de un archivo de texto:
   $ grep root /etc/passwd
   $ grep /etc/passwd/bash

Errores en Sistema de Archivos

La tecnología de fabricación de disco sigue siendo hoy día un tema complejo. Las máquinas que usamos dependen de tecnología de estado sólido excepto por los disco que aún trabajan de una forma mecánica. Estos dispositivos que poseen partes de movimientos mecánicos, aunque bien diseñadas tarde o temprano fallarán. Muchas veces duran hasta el tiempo de obsolesencia, pero en otros casos fallan y en el peor de los momentos. Por eso hay que estar preparados, para éstos momentos es que existen las cintas de backup, las cuales increíblemente también son mecánicas. Aquí cubriremos algunos de los errores que puede enfrentar en el momento del mal funcionamiento de un disco.

Mensajes de Error de Lectura al Instalar GNU/Linux

Un mensaje común de error de lectura es:
Read_intr: 0x10 error

Si esto ocurre con acceso normal al disco, el disco tiene bloques defectuosos (bad blocks). Tiene dos opciones disponibles; el comando badblocks para buscar bloques dañados en el disco o reinstalar GNU/Linux, pero asegurándose de elegir la opción scan-for-badblocks. Si esto ocurre al ejecutar los comandos mke2fs o mkswap, entonces el problema es que el tamaño real de la partición es más pequeña que la que se dió como argumento. Puede elegir ejecutar el programa individual badblocks para revisar el dispositivo. Si existe un alto número de bloques defectuosos en su dispositivo de almacenaje, es recomendable que sea reemplazado.

Usuarios, Grupos y Archivos

El acceso a archivo está basado en privilegios otorgados individualmente al usuario o por membresías a grupos. Todos usuarios pertenece por lo menos a un grupo, el cual esta establecido en el archivo /etc/passwd. Los usuarios pueden ser asignados a grupos adicionales en el archivo /etc/group.

Usted puede evaluar los valores de usuarios y grupos de cualquier usuario con el comando id:

```
$ id usuario1
uid=501(usuario1) gid=501(usuario1) groups=501(usuario1), 79(appserverusr), 80(admin), 81(appserveradm)
```

Los archivos son creados con los valores actuales del usuario y grupos. Sólo el dueño de un archivo y el superusuario pueden cambiar los permisos de acceso de un archivo.

Los archivos /etc/passwd y /etc/group son mantenidos por el administrador de sistema. El archivo grupo contienen los nombres de los grupos validos del sistema acompañado de su valor numérico. Por norma, los nombres de los grupos reflejan la estructura de la compañía y los proyectos que se realizan.

Propietario del Archivo

Los archivos son propiedad de un usuario que lo creo y provee acceso a los miembros que están en el grupo a quien ha sido asignado el archivo.

```
$ ls -l archivo3
-rw-r--r-- 2 ivelis admin 0 28 Jan 13:06 archivo3
```

En éste ejemplo vemos que el archivo3 es propiedad del usuario “ivelis” y pertenece al grupo “admin”. Cada archivo y directorio es propiedad de un único usuario y grupo. La información de propiedad se utiliza al momento de aplicar la protección al archivo.

La propiedad es asignada al usuario y grupo en el momento de la creación del archivo, reflejando así el nombre del usuario y al grupo que el pertenecía en el momento que crearon el archivo.

Observe que el comando cp crea un nuevo archivo. Si la operación de copiar es válida, esta crea un archivo con la identidad de usuario que ejecutó el comando cp, sin importar quien es el dueño del archivo de origen.

Los comandos mv y ln manipulan un puntero en la tabla de inodo (hard link), por lo cual no afecta la propiedad del archivo. En consecuencia los atributo originales del archivos se mantienen.

Protección de Archivos

Todos los archivos son protegidos de los tres tipos de usuarios: el usuario, el grupo, y el otro.

user	group	other
rwx	rwx	rwx

Existen tres tipos de permisos para cada tipo de usuario: Read (Leer), Write (Escribir), y Execute (Ejecutar). Esto resulta en un total de nueve opciones. Para accesar un archivo o un directorio que, los permisos pertinentes para el tipo de usuario deben ser proveídos.

El superusuario puede cambiar o ignorar los permisos de protección de un archivo o directorio.

-rw-r----x 2 ivelis admin 0 28 Jan 13:06 archivo3

Esto es un ejemplo de las protecciones de un archivo del usuario (ivelis) que tiene permisos de lectura y escritura (rw), el grupo (admin) tiene permisos de sólo lectura (r), mientras los otros sólo tienen permiso de ejecución (x).

Determinar Acceso al Archivo

Para determinar el conjunto de tres opciones de protección del archivo debemos comparar la propiedad del archivo y los atributos del grupo contra el nombre y el grupo del usuario. Sólo un conjunto de opciones de protección se examina con el criterio mostrado.

El superusuario no tiene restricción; la opción del usuario (dueño) son usadas si el usuario es propietario del archivo. La opción del grupo es utilizado si el usuario no es el dueño del archivo pero pertenece al mismo grupo que el del archivo. La opción otros (others) es usado si el usuario no es dueño del archivo o no pertenece al grupo del archivo.

Protección: De Archivos

Varias operaciones de archivos pueden ser ejecutadas si el bit de permiso correcto es activado, como se muestra en la siguiente tabla.

Read	r	Lectura Acceso y copia de contenido
Write	w	Escritura Actualizar contenido
Execute	X	Ejecución Ejecutar programas

Para ejecutar un archivo como un comando, el bit de ejecución del archivo debe estar encendido. Observe los archivos en el directorio /usr/bin y notará que todos los programas utilitarios comunes tienen su bit de ejecución establecido. Ahora si echa un vistazo al directorio de los binarios del sistema /usr/sbin notará que la mayoría de los utilitarios tienen los bit de ejecución del dueño y del grupo establecidos, pero no los del usuario normal o sea los otros.

Protección: De Directorios

El nombre de un archivo no es parte del archivo; es una entrada en el directorio que contiene el archivo, así pues, para borrar o renombrar un archivo, acceso debe tener permiso de escritura al directorio. Mover el archivo desde un directorio a otro requiere permiso de escritura a ambos directorios.

Permisos	**Significado**	**Permite**
r	lista	Lista el contenido del directorio; no de permiso de cd
rx	busca	Debe poder buscar en el directorio para localizar y trabajar en archivos

El bit de ejecución para un directorio no significa ejecutar, significa buscar. Si éste bit no esta establecido, los usuarios no pueden efectuar búsqueda en el directorio. Esto significa que los usuarios no están permitidos cambiar a tal directorio para operar dentro de los archivos, como también ellos no podrán atravesar éste directorio y ningún subdirectorio subyacente.

Aunque los usuarios tengan los permisos apropiados para el archivo y saben que el archivo existe, ellos no podrán accesar al archivo si no tienen los permisos de ejecución del directorio.

El permiso de lectura para un directorio permite que los nombres de archivos en el directorio sean leídos, pero los archivos no pueden ser accesados sin los permisos de ejecución.

Modificando Permisos de Acceso a los Archivos

Las propiedades de los permisos de acceso a los archivos se modifica de la siguiente manera:

- Cambie la propiedad de un archivo con:
  $ chown usuario archivos...
  Esta acción esta restringida al super usuario

- Cambie el grupo dueño de un archivo así:
  $ chgrp group archivos...
- Cambie los permisos de acceso a un archivo con:
  $ chmod mode archivos...
- Los tres comandos aceptan la opción -R para procesar los archivos en un directorio recursivamente
- Cambiar el dueño, el grupo o los permisos puede ser efectuado sólo por el dueño del archivo o el super usuario.

GNU/Linux provee comandos para cambiar los permisos de un archivo, su grupo y el grupo del usuario ingresado al sistema.

Los comandos proveen diferentes conjuntos de opciones para ser utilizadas bajo distintas circunstancias.

Cambiar la propiedad de un archivo no es una operación de un usuario normal. El usuario debe ser el dueño actual del archivo. Una vez que el archivo ha cambiado de dueño, el dueño anterior no tiene permisos a éste archivo. Algunos sistemas sólo permiten que el super usuario cambie la propiedad de los archivos.

El uso de grupo es un método común para permitir que varios usuarios puedan modificar archivos que residen en un directorio común.

Los sistemas GNU/Linux permiten que con el comando chown puedas cambiar ambas propiedades de archivos y grupos con un sólo comando usando la siguiente sintaxis:

$ chown dueño.grupo archivos...

Modificar los Permisos de Archivos

Un método usado para cambiar las opciones de protección es el uso de operadores mnemotécnico. El operador = establece protecciones, el operador + agrega protecciones y el operador - remueve protecciones. Para cambiar las opciones de acceso a un archivo use la siguiente sintaxis:

$ chmod tipo-de-usuario [=+-] opciones archivos...f

- u, g, y o para especificar el tipo de usuario

u	usuario
g	grupo
o	otros
a	todo tipo de usuario (opción colectiva es igual a ugo)

- r, w, x para especificar las opciones de acceso.

r	lectura
w	escritura
x	ejecución

Multiples cambios pueden ser aplicados en un sólo comando separados por comas. Mostramos algunos ejemplo:

chmod +x	Agrega permisos de ejecución a todos

chmod g+rw	Agrega permisos de lectura y escritura para el grupo
chmod ug-w	Elimina permisos de escritura para el usuario y el grupo
chmod og=rx	Establece acceso de lectura y ejecución para los otros y el grupo, eliminando la de escritura
chmod ug+x, o-rwx	Agrega permisos de ejecución para el usuario y el grupo; elimina todos los permisos para los otros

Ejercicio 6-3: Permisos de archivos

Las soluciones para éste ejercicio se encuentran el Apéndice A.

¿Qué hacen los siguientes comandos?

1. $ chmod u+x archivos...
2. $ chmod a+r archivos...
3. $ chmod g+w archivos...
4. $ chmod og-w archivos...
5. $ chmod -R go-wx directorio
6. $ chmod og=rx archivos...
7. $ chmod go= archivos...

Modificar los Permisos de Archivos

La manera clásica de cambiar los permisos de los archivos es usando un esquema de combinación de números octales que forman un patrón de bit o mascara de bit (bitmask) que representan los permisos. Cada uno de las nueve opciones de los permisos es representada por un valor numérico. El valor de 4 está asociado con el permiso de Lectura (Read), el 2 está asociado con el permiso de Escritura (Write) y el 1 es asociado con el permiso de Ejecución o búsqueda en los directorios.

Usuario	**Grupo**	**Otros**	**Tipo de Usuario**
rwx	rwx	rwx	Protección Requerida
4+2+1	4+2+1	4+2+1	Valores octales
7	7	7	Suma de los valores 4+2+1

Agregue los números octales del bit para cada tipo de conjunto de permisos (usuario, grupo y otro).

Usuario	**Grupo**	**Otros**	**Tipo de Usuario**
rw-	r--	--x	Protección requerida
4+2+0	4+0+0	0+0+1	Valor Actual: 641

$ chmod 755 archivos...
Establece permisos de rwxr-xr-x

$ chmod 644 archivos...
Establece permisos de rw-r--r--

$ chmod 600 archivos...
Establece permisos de rw-------

Este método de cambiar los bits de protección de los permisos de los archivos y directorios, quizás es un poco más compleja que el método con los caracteres, que le permite especificar un nuevo modo a través de un número octal. Existen otros bits de modo, además de los de protección y aplicar un número octal equivocado puede provocar efectos no deseados y peligrosos.

Para especificar la protección que se requiere, analice por separado los tres tipos de usuarios: usuario, grupo y otro. Deberá determinar para cada uno el nivel de protección que desea y empezando con 0; agregar 4 para el permiso de Lectura, 2 para el permiso de Escritura y 1 para el permiso de Ejecución. Si hay un permiso que no desea simplemente no le agregue el número.

Permiso	**Número Octal**
rwx	7
rw-	6

r-x	5
r--	4
-wx	3
-w-	2
--x	1
---	0

Eliminar Archivos

Para eliminar o sobre escribir archivos, usted necesita los permisos de Escribir y Ejecutar (buscar) en el directorio. Usted no necesita tener acceso al archivo, ni ser el propietario de el. Los comandos rm, mv y ln piden confirmación cuando usted los ejecuta sobre archivos de los cuales usted no posee los permisos de escritura. La opción -f forza la operación, sin importar de los permisos del archivo. En los siguientes ejemplos, ilustraremos el uso de éstos comandos:

```
$ ls -l carta*
-r--r--r-- ... carta1.txt
-r--r--r-- ... carta2.txt

$ rm carta1.txt
rm: remove “carta1.txt” overriding mode 0444? y

$ rm -f carta2.txt

$ rm /bin/ls
rrn: remove ‘/bin/ls’ overriding mode 0755? y
rm: /bin/ls: Permission denied

$ rm -f /bin/date
rm: /bin/ls: Permission denied
```

Las sentencias rm -f y mv -f a menudo son usadas dentro de scripts del shell, para que en el caso de que los archivos no tengan los permisos de escritura. Esto le asegura que la operación no fallará ya que el usuario no será pedido que confirme la acción durante la ejecución del script.

Al eliminar directorios completos recursivamente, la sentencia rm -rf es normalmente utilizada ya que evita ser cuestionado para la confirmación de cada archivo que usted no tienen los permisos de escritura.

Ejercicio 6-4: Organizar los Archivos

En éstos ejercicios investigaremos el uso de los utilitarios avanzados y la implementación de expresiones regulares. Las soluciones a éstos ejercicio se encuentran en el Apéndice A.

1. Para ilustrar las implicaciones de derecho a acceso de los archivos usaremos comandos que modifican los permisos de los archivos.
   Cambiese a su directorio home, efectúe una copia del archivo /etc/group y observe sus permisos:
   ```
   $ cp /etc/group grupo.prueba
   $ ls -l grupo.prueba
   ```

   Ahora elimine los permisos de Lectura (Read) para los usuarios otros y confirme el cambio:
   ```
   $ chmod o-r grupo.prueba
   $ ls -l grupo.prueba
   ```

   Ahora elimine los permisos de Lectura (Read) del usuario y trate de leer el archivo:
   ```
   $ chmod u-r grupo.prueba
   ```

$ cat grupo.prueba

¿Qué pasó? ¿Por qué no puede usted leer el archivo? ¿Si usted y el archivo pertenecen al mismo grupo y el grupo tiene permisos de Lectura establecidos, porqué no puede usted leerlo?

2. Ahora prestaremos atención a los bits de permisos cuando aplican a los directorios. Deberá dominar el uso de los comandos de manipulación de archivos, como son cp, mv, y rm, y los requerimientos de permisos para poder ejecutar éstos comandos a los archivos.
   Intente eliminar el archivo grupo.prueba:
   $ rm grupo.prueba

   ¿Porqué es que ahora funciona, si ni siquiera podías leerlo con cat anteriormente?

   Ahora pruebe:
   $ rm /etc/group

   Responda "yes" (sí) cuando le pida confirmación. ¿Porqué no pudo borrarlo?

   Un principio similar se nos presenta en esta siguiente sección sólo para reforzar conocimiento y para entender el significado de la opción -f del comando rm.

   Pruebe éstos comandos desde su directorio home y revise las opciones de permisos después de cada comando:
   $ cp /etc/inittab tabla_inicio
   $ chmod ug=r tabla_inicio
   $ chmod o= tabla_inicio

   Ahora intente eliminar el archivo:
   $ rm tabla_inicio

   Cuando le pida confirmación responda "no". ¿Entiende usted la secuencia de comandos que acabamos de ejecutar?
   Y finalmente utiliza la opción de forzar para eliminarlo:
   $ rm -f tabla_inicio

3. Vamos a investigar los permisos de acceso a los directorio, el bit "x". Usted debe aún tener el directorio trabajo que creamos en los ejercicios anteriores. Si por algún razón lo eliminó, deberás crearlo nuevamente en sus directorio home.

   Coloque un par de archivos en el directorio trabajo; usted puede usar los comandos cp, touch, > o cualquier otro metódo que usted prefiera (vi, emacs, pico, nano, joe etc.).

   Asegúrese de estar en su directorio home, remueva el bit de ejecución (x) del directorio trabajo:
   $ cd; chmod u-x trabajo2

   ¿Cuáles son las implicaciones de lo que usted acaba de ejecutar? Escriba los siguientes comandos:
   $ ls trabajos2
   $ ls -l trabajo2

   ¿Puede usted explicar porque recibe un mensaje de error?

   ¿Puede usted ejecutar lo siguiente con éxito?
   $ cd trabajo2

   Ahora restablezca los permisos así:
   $ chmod u+x trabajo2
   Asegúrese de poder efectuar listado largo del directorio trabajo2, hacer cd, etc.

4. Crearemos aquí un shell script. Escriba los siguientes comandos:

```
$ cat >reloj
cal
date
```

Luego escriba CONTROL+D.

Luego esta próxima secuencia de comandos:

```
$ ./reloj
$ chmod u+x reloj
$ ./reloj
```

Usted acaba de crear su primer shell script de bash que puede ser ejecutado como un comando de GNU/Linux normal. Note que usted necesita los permisos de ejecución (x) en el archivo que creó para hacerlo un script del shell; GNU/Linux no tiene extensiones en sus archivos así que no reconoce los archivos por su extensión.

Ejercicio 6-5: Vínculos/Links

Este ejercicio trata de reforzar el entendimiento de los conceptos de los vínculos. No se proveen soluciones a éste ejercicio.

Observe los próximos comandos cuidadosamente y asegúrese de entender que hace cada uno:

```
$ mkdir vínculos # crea un directorio
$ cd vínculos
$ cat > archivo1 # crea dos archivos de data
Este es el archivo1
^D
$ cat > archivo2
Este es el archivo2
^D
$ ln archivo1 archivo3 # Un vínculo de hard link
$ ln -s archivo2 archivo4 # vínculo simbólico
```

Ahora veamos los siguientes archivos:

```
$ ls -il
$ cat archivo1
$ cat archivo2
$ cat archivo3
$ cat archivo4
$ ls -ilL
```

Ya tenemos tres archivos: archivo1 y archivo3 son los mismos (vea su número de inodo) y el archivo4 es un vínculo simbólico al archivo2.

```
$ rm archivo1
$ cat archivo3
```

Haber removido el archivo1 no tuvo ningún efecto con el archivo3. Un archivo es eliminado cuando su último vínculo es eliminado. Ahora pruebe:

```
$ rm archivo2
$ cat archivo4
```

Fíjese el mensaje de error confuso del comando cat, éste nos dice que no puede leer el archivo4. Nosotros sabemos que el archivo4 existe, pero es un vínculo simbólico al archivo2, el cual borramos anteriormente. Los vínculos son transparentes a todos los utilitarios, excepto ls y rm.

Crea el archivo2 de nuevo, esta vez simplemente copie el archivo passwd del sistema dentro del directorio actual, luego despliegue el contenido del archivo4 a pantalla:

```
$ cp /etc/passwd archivo2
$ ls -il
$ cat archivo4
```

El acceso al archivo2 ha sido re-establecido, usted puede de nuevo desplegar su contenido, aunque ya no es el mismo contenido. Recuerde que deberá remover los vínculos si usted remueve el archivo original.

Administración de los Dispositivos

Desde el punto del kernel Linux, los dispositivos son sólo nombres de archivos que se encuentran en el directorio /dev. En esta sección describiremos la administración de los dispositivos. Los tópicos a discutir en esta sección incluyen:

- El Directorio de los Dispositivos /dev
- Crear Nodos de los Dispositivos
- El Sistema de Archivos /proc

El Directorio Device

El directorio /dev contiene un archivo de referencia a la gran mayoría de dispositivos del sistema. El administrador puede utilizar éstos archivos para accesar a los dispositivos. El buen ejemplo es el disquete o floppy. Es muy fácil copiar un archivo desde el disco duro a un floppy.

Sólo debe almacenar archivos de dispositivos en éste directorio. No debe almacenar ningún otro tipo de archivo en éste directorio. Cualquier archivo que no sea de referencia a un dispositivo que se encuentre en éste directorio lo más seguro que fué copiado aquí por error, ejemplo es, equivocarse al copiar directamente a un archivo de dispositivo por el administrador. Muy común es al copiar datos durante un backup, a una cinta y escribir mal el nombre del dispositivo y crear todos los datos al directorio /dev. Recuerde que /dev normalmente esta en la raíz del sistema de archivos, éste tipo de acción puede causar que se llene el sistema de archivos y causarnos graves problemas.

Interfaces de Dispositivo

El kernel Linux usa una interfaz de archivo virtual para accesar sus dispositivos. Esto simplemente es que los dispositivos son, como todo lo demás, archivos en un sistema de archivos y pueden ser accesados utilizando los utilitarios normales de GNU/Linux. Es posible usar el comando cp para copiar datos a un dispositivo, cat para copiar datos desde un dispositivo a la pantalla, y así sucesivamente. En la mayoría de los casos, como son los discos, usar los utilitarios normales puede rendir resultados no esperados, y escribir datos directamente al dispositivo puede alterar (corromper) el sistema de archivos.

Cuando el kernel accesa un archivo de dispositivo, éste reconoce el bloque especial o el atributo de caracter y redirecciona el requerimiento de E/S (I/O) al software manejador del dispositivo (driver) en el kernel. El manejador de dispositivo elegido es seleccionado por el número mayor (los archivos de dispositivos tienen dos números un mayor y un menor) del archivo; el número menor del archivo es pasado a la interfáz del manejador del dispositivo. Los números menores son específicos a cada tipo de dispositivo y concordantemente interpretados.

El nombre actual del dispositivo no es de ninguna importancia, lo importante son éstos números de mayor y menor que definen el dispositivo. Cada sistema define su convención de como nombrar éstos dispositivos.

Los números de los dispositivos son específicos a cada fabricante y cada tipo de sistema GNU/Linux.

Hasta un mismo sistema GNU/Linux en un mismo equipo, pero con diferente dispositivos y un kernel diferente, puede que tengan diferente número de los dispositivos. Sistemas idénticos tendrán números de dispositivos idénticos.

En el siguiente pantalla mostraremos lo siguiente:

- Dispositivos de caracter son marcados con el tipo de archivo c.
- Dispositivos de Bloque son marcados con el tipo de archivo b.
- Números mayor y menor de los dispositivos se listan donde normalmente se lista el tamaños del archivo.

```
$ ls -l /dev/disk0s[78] /dev/console
crw------- 1 root root 4, 0 30 Jan 23:53 /dev/console
br--r----- 1 root root 14, 7 24 Jan 13:37 /dev/disk0s7
br--r----- 1 root root 14, 8 24 Jan 13:37 /dev/disk0s8
```

Dispositivos de Caracter

La transferencia de datos ocurre un caracter a la vez. Los dispositivos que pertenecen a esta categoría incluyen terminales, impresoras, modems, y cintas magnéticas.

Dispositivos de Bloque

La transferencia de datos hacia y desde éstos dispositivos ocurre un bloque a la vez. Bloques lógicos reflejan la habilidad del kernel para buffer y manipular grandes cantidades de data. El tamaño de bloque depende del sistema de archivo.

Dispositivos Utiles

Los dispositivos de GNU/linux utilizan nombres descriptivos de los dispositivos a cuales ellos hacen referencia. Los dispositivos mostrados son razonablemente estándar en la gran mayoría de sistemas GNU/Linux.

/dev/null	Dispositivo Null (Null Device)
/dev/tty	Pseudo dispositivo de terminal para el ingreso de los usuarios
/dev/console	Nombre común para la consola del sistema
/dev/ttynn	Terminales Directamente conectadas
/dev/ttyxnn	Terminales Multíplexadas (x típicamente a, b, etc)
/dev/ptsnnn	Pseudo Terminales del X Window y de network
/dev/fd0	Nombre común de la controladora del disco floppy
/dev/ftape	Vínculo a la controladora de cinta (tape drive)
/dev/hdx	Discos de Integrado Controlador Electrónico (IDE)
/dev/sdx	Discos de Small Computer Systems Interfáz (SCSI)
/dev/cdrom	Normalmente vínculo al controlador del CD-ROM
/dev/lp0	Puerto Parallelo de Impresión

Crear Nodos de Dispositivos

Ocasionalmente, los nodos de los dispositivos son eliminados por accidente o por error al instalar un nuevo dispositivo. Al instalar un nuevo dispositivo necesitamos ocasionalmente crearle un nodo, para esto existe el comando mknod. Los nuevos dispositivos tienen sus números de dispositivo mayor y menor definidos en la documentación que les acompaña. Para los nodos ya existentes, el administrador tendrá que encontrar los valores correctos. Puede buscar otros dispositivos similares, buscar en otros sistemas o llamar al suplidor del dispositivo para los valores requeridos.

Habiendo creado el inodo, establecido el dueño y el grupo para igualar los otros dispositivos; normalmen-

te, el dueño es root y el grupo es el dispositivo o dev, pero existen muchas excepciones.

```
# cd  /dev
# pwd
/dev
# mknod  prueba  b  14  9
# chown  root  prueba
# chgrp  operator  prueba
# chmod  ugo=rwx  prueba
```

El Sistema de Archivos /proc

El sistema de archivos /proc nos provee una fuente de información en muchos aspectos del kernel en ejecución. Desde sus archivos podemos derivar mucha información de los procesos en ejecución en el sistema. En el directorio /proc, existe un conjunto de subdirectorios cuyos nombres son números correspondientes a cada proceso denominados PID (Process Identification Number) o Número de Identificación del Proceso. Existe un vínculo simbólico de nombre self, el cual siempre apunta al directorio del proceso actual. Dentro de éstos directorios enumerados se encuentran archivos y subdirectorios que nos rinden información acerca de cada proceso.

Aquí una lista de sólo unos cuantos de los archivos en éste directorio que nos proveen importantísima información:

/proc/interrupts	Valores establecidos de IRQ (Interrupt Request)
/proc/ioports	Valores de E/S (I/O)
/proc/cpuinfo	Detalles del CPU (Procesador)
/proc/dma	Valores establecidos de DMA (Direct Memory Access)

Aquí unos cuantos subdirectorios en el directorio /proc:

/proc/scsi	Información de los Dispositivos SCSI
/proc/ide	Información de los Dispositivos IDE
/proc/net	Actividades de la Red /Nerwork
/proc/sys	Parámetros de Configuración del kernel

El sistema de archivos /proc es un sistema de archivos virtual y no físico; por esta razón, éste no ocupa espacio en el disco y cualquier intento de escribir en el será fallido. Está compuesto mayormente por entradas que apuntan al kernel ejecutándose en memoria. Estas localidades están identificadas por el archivo apropiado de /boot/System.map de un kernel en particular.

Anteriormente, los procesos eran examinados por el comando ps, proceso que se ejecutaba etiquetado SUID (Establezca el ID del Usuario). El sistema de Archivo /proc elimina éste potencial problema de seguridad. Uno de los efectos secundarios de utilizar /proc es que ahora la expresión “todo es un archivo” también aplica a los procesos.

Ejercicios 6-6: Administración de los Dispositivos

Para que entienda como GNU/Linux accesa los dispositivos a través de los sistemas de archivos, practicaremos agregando un nuevo dispositivo. Crearemos un nuevo nodo para éste (un nombre de archivo que hace referencia al manejador de un dispositivo que se encuentra en la tabla administrada por el kernel).

La solución a éste ejercicio se encuentra en el Apéndice A.

1. Para crear un dispositivo, haremos un nuevo nodo de dispositivo de bloques y lo llamaremos disquete en el directorio /dev, y usaremos los mismos números de mayor y menor del dispositivo /dev/fd0.

   Crea una segunda entrada y llamela raw.disquete, el cual será un dispositivo de caracter basado en el dispositivo existente /dev/fd0

2. Ahora dé formato a un disquete utilizando su nuevo dispositivo. Use el sistema de archivos ext2. ¿Cuál de los dos nuevos nombres de dispositivos creados utilizaría usted para llevar la operación a cabo? ¿Puede usted escribir archivos al disquete recién formateado?

RESUMEN

En éste capítulo se introdujo los conceptos fundamentales de los sistema de archivos, los principales temas fueron:

- Las particiones permiten tener múltiples sistemas de archivos en un sólo dispositivo.
- Cada archivo es identificado por un número llamado inodo.
- Los Inodos son utilizados para localizar e identificar los archivos.
- El kernel Linux accesa los dispositivos a través de archivos almacenados en el directorio /dev.
- Los archivos de dispositivos utilizan números llamados mayor y menor para identificar manejadores y dispositivos individuales.
- Los nombres de dispositivos están ya bien estandarizados.
- Existen dos tipos de nombres de rutas: absoluto y relativo.
- Un nombre de ruta absoluta es aquel que empieza desde la raíz del sistema de archivos.
- Un nombre de ruta relativa empieza desde el directorio actual o cualquier otro menos /.
- El esquema de particionado del disco se mantiene oculto del usuario.
- Los dispositivos son accesados a través del sistema de archivos (del directorio /dev).
- Los archivos que sus nombres empiezan con un punto son especiales:
- No son listados normalmente al desplegar los directorios.
- Contienen archivos de inicio y de configuración.
- Los archivos de los usuarios se almacenan en el directorio home del usuario.
- Varios utilitarios le permiten a usted manipular archivos y directorios.
- GNU/Linux particiona y crea sistemas de archivos a los discos.
- Los archivos tienen un número único dentro de los sistemas de archivos (llamado inodo).
- Los nombres de archivos son entradas en los directorios.
- Los archivos pueden tener más de un nombre.
- El acceso a los archivos es controlado por el bit de usuario, grupos y esta definido en su inodo.
- Sólo el superusuario y el dueño de un archivo pueden cambiar los permisos de acceso.

PREGUNTAS POST - EXAMEN

Las respuestas a estas preguntas están al final de este libro en el Apéndice A

1. Usted copia el archivo1 archivo2 archivo3 en el directorio /home/usuario/trabajos. ¿Cómo puede hacer esto en una sola línea de comando? ¿Cómo puede usted eliminar éste directorio y todos los archivos dentro de el en un sólo comando?

2. Al editar un documento de 4 páginas su editor se cuelga. ¿Cuál de los dos comando more o less resultaría de mejor provecho para poder ver el contenido hacia arriba y hacia abajo?

3. Usted necesita asignarle los permisos de lectura y escritura al archivo llamado archivo.txt (Asuma que el archivo tiene permisos de -rwx------). Explique ¿Cómo lograr esa tarea?

4. Usted se encontraba en el directorio /home/usuario/trabajo. ¿Cuál de éstos directorios es el directorio padre del directorio trabajo/? ¿Cómo puede usted detectar esto rápidamente?

5. El usuario se encuentra en el directorio /etc y desea encontrar un archivo que sólo sabe que su nombre empieza con “ho” ¿Cómo puede usted encontrar éstos archivos sin utilizar el comando ls -l?

CAPITULO 7

ADMINISTRACIÓN DEL SISTEMA OPERATIVO GNU/LINUX

NAVEGAR EL SHELL DE GNU/LINUX

OBJETIVOS

Al completar este Capítulo, usted podrá:

- Trabajar efectivamente en la línea de comandos de UNiX.
- Usar el Shell para ejecutar procesos Multitareas.
- Diseñar un ambiente ergonómico.
- Explicar comandos y utilitarios.
- Ejecutar comandos esenciales de GNU/Linux desde la línea de comandos.
- Usar los siguientes comandos individualmente o en combinación: ls, cd, more, less, cp, mv, mkdir, rm, rmdir, ln, file, grep, sed, y awk
- Describir comandos comunes del shell de editar.

PREGUNTAS PRE-EXAMEN

Las repuestas se encuentran en el Apéndice A.

1. La mayoría de las distribuciones de GNU/Linux vienen configuradas con un Shell por defecto. ¿Cuál es el nombre de ese Shell?

2. Necesita comparar dos archivos y buscar cuales líneas de código fuente tienen el cambio de versión 1 a versión 2. ¿Cuál comando básico puede ser usado para llevar esto a cabo?

3. ¿Por qué es tan importante para un administrador de sistemas GNU/Linux saber utilizar por lo menos un editor de texto? Nombre algunos editores de textos populares de GNU/Linux.

4. ¿Por qué es que algunos comandos son internos del Shell y algunos comandos son simple ejecutables almacenados en algún directorio específico?

5. ¿Cúal es el significado de la variable del Shell PATH?

INTRODUCCION

En éste capítulo, trataremos por que el Shell es tan importante para GNU/Linux. También trataremos como navegar y desplazarse el sistema de archivos usando el shell y como realizar las operaciones básicas de archivos. También hablaremos del poder que la línea de comandos nos ofrece y por que es tan importante dominarla.

El Ambiente del Shell

Ahora vamos a observar el ambiente del shell e introduciremos algunos conceptos de variables, cuales pueden ser usadas para configurar el shell y otros programas. Variables especiales usadas para determinar como se buscan los programa a ejecutar y describir el prompt del shell en detalle. Entre los tópicos que trataremos en esta sección se encuentran:

- Proceso Padres/Hijos
- Multiproceso
- Ambiente de Shell
- Definiendo Variables del Shell
- Ruta de Búsqueda
- Prompts del Shell
- Archivos de Ambiente

El propósito y configuración de los archivos de inicio, tales como $HOME, /.bash_profile y .bashrc (nombrado en $ENV), también son discutidos.

Los Procesos Padres/Hijos

Entendiendo el concepto de la relación de procesos padres/hijos ayuda mucho cuando queremos comprender los otros aspectos de las operaciones que realiza GNU/Linux. Esto nos ayudará a entender el manejo de variables, las cuales serán discutidas más adelante.

Procesos Padres

1. Cuando accesamos el sistema (login), un shell con un único PID (número de ID de proceso) es invocada. Los Shell de GNU/Linux (y otros programas) han sido diseñados para mantener un denominado ambiente. Un ambiente es un conjunto de definiciones, lo que principalmente son variables, pero también incluyen funciones y alias. El shell ofrece el prompt para indicar que esta preparada para aceptar comandos.

2. Cada vez que se ejecuta un comando externo (un programa desde el disco, como vi), el shell crea una copia de ella misma, un hijo. Hay dos cosas importantes sobre esta nueva copia: se le asignará un nuevo y único PID y esta heredará parte del ambiente de su padre.

3. Una vez nacido y funcionando el proceso hijo, el nuevo comando reemplazará el shell hijo y se ejecutará. Mientras se ejecuta el nuevo proceso, normalmente se tiene una pantalla inactiva en blanco; la shell padre, aquella que nos ofrece el prompt, estará esperando que el proceso hijo finalice. Una vez el proceso hijo finaliza, el shell nos devuelve el prompt.

Multitarea

Una de las características más poderosas de GNU/Linux es su habilidad de hacer más de una cosa al mismo tiempo, un proceso que es referido comúnmente como Multítarea. Existen muchas maneras en las que el usuario puede emplear Multítarea en GNU/Linux, y a continuación algunas de estas serán discutidas.

Línea de Comandos

Mientras se está trabajando en línea de comando, el usuario no necesita esperar que un programa termine antes de iniciar otro comando. Los programas pueden ser ejecutados en segundo plano (background) simplemente agregándole al final del comando el caracter ampersand (&). Por ejemplo, el siguiente comando será ejecutado en background:

```
$ find / -name nombre_archivo.txt &
```

Posteriormente de haber iniciado un comando, el usuario nota que el comando tomará más tiempo del estimado, el comando puede ser suspendido presionando las teclas CONTROL+Z y enviado a background con el comando bg. Por ejemplo:

```
$ find /usr/ -name “nombre_archivo.txt” -print
<Ctrl>Z
[1]+ Stopped
$ bg
```

Terminales Virtuales

Otra manera en que más de una tarea puede ser realizada a la vez, es usando las características de terminal virtual de GNU/Linux. Presionando CONTROL+ALT y una tecla de función (F1 hasta F6) accederás diferentes terminales virtuales. Por ejemplo, usted puede ingresar al sistema por el terminal F1 y empezar una tarea y luego cambiarse al terminal F2 para hacer otra tarea mientras la tarea de la terminal F1 está en proceso.

El Ambiente X Window (Environment)

En el ambiente X Window, el usuario puede tener más de un programa abierto (ventanas) al mismo tiempo, como también puede tener múltiples desktop. Las características de múltiples desktop es dependiente del administrador de ventanas (windows manager) que se esté usando. La mayoría de los administradores de ventanas tienen por defecto cuatro desktop que se accesan a través de un applet incrustada en el panel inferior. Por ejemplo, en el administrador de ventanas Enlightenment se puede configurar hasta 32 en los paneles verticales y horizontales y 32 más en los inferiores y superiores.

Ambiente del Shell

El shell es afectado por su ambiente. Cada programa tiene un ambiente, el cual es usado para almacenar sus definiciones. Un programa hereda su ambiente del programa (proceso) padre; en las situaciones más sencillas el proceso padre será el shell.

Igualmente, un programa pasaría su ambiente a cualquier otro programa (proceso hijo) que éste decide llamar. Un programa puede modificar su ambiente, pero las modificaciones que éste realiza no son aplicadas al ambiente del programa padre.

Cuando un proceso hijo es iniciado, éste proceso también es denominado subshell, su ambiente es copiado desde el proceso padre. Modificaciones subsecuentes por el proceso padre a su propio ambiente no modificará el ambiente del proceso hijo.

Las variables definidas dentro de un proceso son locales a ese shell al menos que sean exportadas al ambiente. Las variables son agregadas al ambiente con el comando export. Este comando recibe una lista de nombres de variables para ser exportas. Un error común es incluir el símbolo de dinero ($) en frente de cada nombre de variable, esto le indicaría al shell que substituya el valor de la variable y entonces proceder a exportar la variable de ese nombre en vez de la actual variable.

Definir Variables del Shell

El shell posee un mecanismo interno para el manejo de las variables. Las Variables son utilizadas para almacenar valores de cadenas que luego pueden ser sustituidas en la línea de comando. Estas nombres de variables pueden contener letras, dígitos o simbolos; ellas deben empezar con una letra.

Variables especiales que empiezan con un dígito son definidas por el shell. Estos son parámetros posicionales, los cuales serán descritos en el próximo capítulo.

El shell también define algunas variables especiales que tienen nombres no alfanuméricos. Al sustituir-variables en línea de comando, el shell lee todas las letras y números después del símbolo de $ para conseguir el nombre de la variables. Para separar el nombre de la variable del texto a su alrededor, esta puede ser encerrada entre las llaves '{ }':

```
$ PRUEBA=”prueba ”
$ echo $PRUEBAcadena # El nombre de la variable es PRUEBAcadena, la cual se encuentra vacía
$
$ echo ${PRUEBA}cadena
prueba cadena
```

Cualquier valor puede ser asignado a una variable. Los valores que contengan espacios, tabulados o el caracter de nueva línea deben ser encerradas entre comillas para prevenir que el shell lo reconozca como separadores de argumentos. Múltiple asignación de variable puede ser otorgado en un sólo comando antes del opcional comando export.

Las variables pueden ser utilizadas desde cualquier lugar en la línea de comandos, incluyendo al principio. Codificando un nombre de ruta de directorios extendido puede ser almacenado en una variable y entonces puede usar la variable para invocar el comando en una manera más abreviada:

```
$ EDITOR=/usr/share/local/editor/bin/editor
```

Ahora para invocar el editor es mucho más simple, luego de haber creado esta variable, sólo tendrá que escribir así:

```
$ EDITOR archivo.txt
```

Ejemplo: Manejando Variables

En éste ejemplo ilustramos como preparar el ambiente de su sessión en vi definiendo y exportando dos variables del shell:

```
$ TERM=vt220 # define el tipo de terminal que estamos usando
$ EXINIT=”Set showmode” # estable opciones de vi deseadas
$ export TERM EXINIT # estableces las variables como “globales”
$ vi archivo.txt # vi hereda a TERM y XINIT de los variables del shell actual
```

Substitución de variable toma lugar dentro de las comillas, como le mostramos aquí:

```
$ HOLA=”Saludo”
$ echo “$HOLA Mundo”
Saludo Mundo
$ echo ‘$HOLA Mundo’
$HOLA Mundo
```

Esto significa que mientras el shell esta escaneando la línea de comandos podrá ver el caracter de pesos ($) a través de las comillas dobles y efectuará la substitución de variable. El símbolo de $ es una de las cuatro excepciones que no son protegidas por las doble comillas. Al usar las comillas sencillas el shell no verá los simbolos especiales y las substituciones no tomarán lugar. Para mostrar todos los métodos de éste tipo de situaciones también muestra la barra invertida (\):

```
$ echo \$HOLA Mundo
$HOLA Mundo
```

Variables del Shell Internas y del Sistema

El shell hace uso de variables para personalizar el ambiente de ingreso al sistema de los usuarios. Algunas variables internos son predefinidas por el shell y se le asignan valores por defecto, por ejemplo, el prompt de login . Algunas variables son definidas en el archivo de perfil del sistema; comúnmente la ruta de búsqueda por defecto, el PATH, también es definido de esta misma manera. Para ver todas las definiciones de las variables, escriba "set" en el prompt. Para poder ver la salida página por página hágalo con el filtro more de esta manera; set | more.

Aquí le mostramos algunas de las variables estándar que muy a menudo se encuentran definidas:

HOME	Directorio Home
PWD	Directorio Actual de Trabajo
OLDPWD	Directorio Previo de Trabajo
PATH	Ruta de Búsqueda
PS1, PS2	Cadenas del Prompt
HISTFILE	Nombre del archivo history
HTSTSIZE	Número de líneas del archivo history
ENV	Archivo de ambiente de bash
TERM	Define el tipo de terminal; vi fracasará sin ésta
VISUAL	Define el editor visual (screen)
EDITOR	Editor por defecto (usado por algunos comandos si no esta establecida la variable VISUAL)

Otras variables útiles incluyen a:

LOGNAME	Su nombre de ingreso (login)
SHELL	El shell de ingreso (login)
PAGER	El paginador por defecto del comando man (more o less)

Muchos sistemas utilizan variables que afectan todo el sistema para definir la localidad de directorios especiales o programas. Por ejemplo, el sistema de base de datos Oracle usa una variable llamada ORACLE_HOME para especificar el lugar de su directorio home.

La variable del shell HOME le informa al cd donde se encuentra el directorio home del usuario (si no especificamos argumentos); MAIL informa al shell cual archivo de correo revisar para detectar si arribó correo nuevo. La variable MAIL es exportada al ambiente y es usada por el programa mail para especificar el archivo mail por defecto. Las variables PATH, PS1, y PS2 serán descriptas más adelante.

Las variables VISUAL y EDITOR son usadas por algunos utilitarios, muchas veces las interfaces con programas administrativos. Cuando se espera que un utilitario modifique interactivamente un archivo de configuración, éste archivo se le presentará dentro del editor definido en las variables VISUAL o EDITOR. Estas variables también son usadas por su shell para decidir cual usar para la edición desde la línea de comandos (como alternativa podemos establecerlo así set -o nombre_editor).

Entre las variables no especificas del shell se incluye LOGNAME, la cual es inicializada en el momento que usted ingresa al sistema y es utilizada mayormente como fuente de información. La variable TERM debe ser establecida para definir el tipo de terminal si programas especializados como el editor visual (vi) serán usados.

Ruta/Path de Búsqueda

El shell usa la variable PATH para definir la lista de los directorios a buscar para encontrar los comandos. Si el comando no se encuentra en uno de éstos directorios, se reportará que el comando no fué encontrado. El programa (comando) escrito en la línea de comandos, es buscado en cada uno de los directorios especificado en la lista definida por la variable PATH es buscado por el programa nombrado. El primer programa por el nombre especificado es ejecutado y la búsqueda es abortada inmediatamente.

Los comandos internos del shell se le darán la preferencia sobre los comandos externos. Es posible que si existen dos programas con el mismo nombre o versiones el primero encontrado en la ruta oculte el segundo. Si deseamos ejecutar el segundo tendremos que definirle su ruta completa. Un error común de programadores de C en GNU/Linux es nombrar su archivos de prueba test.c. El programa compilado (llamado test) es normalmente escondido por el programa estándar test del sistema.

La variable PATH normalmente incluye el directorio actual (.), lo que le dice al Shell que también busque el el directorio actual además de los directorios nombrados por la variable de búsqueda. Si omitimos esta entrada evitará que el shell encuentre programas en el directorio actual. El directorio actual es evaluado al momento de un programa ser ejecutado así, que cambiar de directorio actual (usando el comando cd) no afectará el mecanismo de búsqueda, además de especificar el directorio actual como el punto, un directorio de nombre null también puede ser usado. Dos dos puntos de lado y lado (::) definen un directorio null; también, una ruta que empieza o termina con dos puntos definen el directorio actual. El superusuario jamás debe incluir un punto en su ruta de búsqueda, ya que esto es una causa mayor de violación de seguridad.

Un error común es incluir dos puntos en demasía en la ruta de búsqueda. Cada directorio especificado siempre es buscado. En el siguiente ejemplo incluimos cuatro búsquedas diferentes en el directorio actual, lo cual lo torna muy ineficiente:

```
PATH=:/bin:/usr/bin:.:/usr/local/bin::
```

El orden de búsqueda en la variable PATH puede ser alterada , pero no es aconsejable. El comando type y el comando which nos indican donde podemos encontrar un programa, pasado como argumento, en la ruta de búsqueda (en el PATH).

```
$ type who
who is /usr/bin/who
$ which who
/usr/bin/who
```

Ejemplo: Modificar la Ruta/PATH

Cada usuario tiene control de los directorios de búsqueda del shell para los comandos a ejecutar. Normalmente, nos vemos en la necesidad de agregar nuevos directorios a nuestra ruta de búsqueda. Si un directorio es agregado a la ruta de búsqueda y colocado en frente de los directorios en el PATH actual, éste será buscado primero por los comandos del sistema. Ponga a consideración el siguiente PATH:

```
PATH=/home/usuario/bin:/bin:/usr/bin:.
```

El directorio /bin en el home del usuario es el primer directorio buscado. Ahora es posible escribir versiones de los comandos del sistema como los son ls, cat, more, etc., los cuales tendrán comportamiento que difieren del original a los comandos del sistema. Por ejemplo, el comando ls en el directorio /home/usuario/bin puede ser escrito para que invoque el comando real en /bin/ls con la opción -l.

Otra cuestión ha tomar en consideración al establecer los valores del PATH es la eficiencia del sistema. Si su PATH es muy largo o extendido y los comandos más usados se encuentran en el último directorio, el shell desperdiciará mucho tiempo gestionando la búsqueda de sus comandos.

Prompts del Shell

El prompt del shell es una variable. Es aconsejable por razones útil incluir el nombre del equipo, el directorio actual y el nombre del usuario en el prompt. El shell le permite al usuario personalizar el prompt del shell. La cadena primaria del prompt es almacenada en la variable de nombre PS1 y es la que es desplegada normalmente en el prompt del shell. Líneas de continuación utilizan el prompt secundario almacenado en la variable PS2.

Consideremos los siguientes ejemplos:

• PS1 es usado como el prompt del shell ordinario y primario:

```
$ PS1='$PWD> '
/home/usuario>
```

los prompt del bash deben ser verificados entre comillas simples para retrasar la substitución de variable hasta que el prompt sea desplegado.

• PS2 es utilizado como el prompt del shell secundario (la continuación):

```
/home/usuario> PS2=">>"
/home/usuario> echo -n 'Hola
>>Mundo'
Hola
Mundo
/home/usuario>
```

El shell reconoció que la línea no fué terminada (ya que las comillas no fueron cerradas) e imprimió un prompt de continuación.

• PS4 usado para forzar al shell a mostrar el resultado de un escaneó de línea:

```
/home/usuario> echo $PS4
+
/home/usuario> set -x
/home/usuario>
/home/usuario> Is -d /usr/s*
+ Is -d sbin share src
sbin share src
/home/usuario>
```

En el bash, el prompt estándar es el símbolo de $ para el usuario normal y el símbolo de # para el super usuario. El prompt de # actúa como una advertencia para los administradores de sistemas que se encuentran ejecutando con los privilegios del super usuario y debe tomar un cuidado muy especial.

En bash cualquier variable dentro de la cadena PS1 son expandidas antes de que se presente el prompt. Si aparece el caracter "!" en el texto, éste será reemplazado por el número actual en el historial de comandos.

```
$ PS1='($PWD) !? '          # necesita comillas sencillas, sino PWD expanderá
(/u/train4) 47?             # nuevo prompt: cambia cuando cambias de directorios
```

En el C shell, el prompt primario es almacenados en la variable prompt (en minúscula).

Archivos de Ambiente

El ambiente del shell será afectado por varios archivos al invocar el shell, éste inicialmente lee comandos desde dos archivos. El archivo /etc/profile que contiene las variables de todo el sistema y es administrado por el super usuario para establecer variables locales del sistema y requerimientos especiales. Los archivos de inicio del usuario ($HOME/.bash_profile) es mantenido por cada usuario y puede ser configurado para ejecutar cualquier tipo de inicialización especial.

Los archivos profile sólo son leídos bajo ciertas condiciones, lo que significa que la mayoría de los casos tienen permiso de sólo lectura en el momento de ingreso del usuario al sistema. Los archivos profiles no son leídos cuando se ejecutan los scripts del shell o los subshells. Ambos de éstos archivos son opcionales, pero es poco usual encontrar un sistema sin el archivo /etc/profile

Si la variable env es definida y exportada al ambiente, cualquier shell bash leerá en invocará los comandos en el archivo definido por esta variable. Este archivo debe ser utilizado para establecer características de bash que son requeridas por todos los shells de bash, no sólo el shell de login. Típicamente el archivo nombrado en la variable env es $HOME/.bashrc.

Copias de los archivos .bash_profile, .bashrc, y otros archivos comunes de ambiente son copiados en el momento de la creación de un nuevo usuario, esos templates se almacenan en el directorio /etc/skel. Aquí el administrador de sistemas puede editar las configuraciones de inicio o colocar archivos adicionales.

Ejemplo: Archivos de Ambiente

Consideremos lo siguiente:

• Un archivo de sistema profile típico (/etc/profile):

```
PATH=$PATH:/usr/qa/tools/bin
ENV=$HOME/.bashrc
ORACLE_HOME=/dbs/oracle
ORACLE_SID=qadb
export PATH ENV ORACLE_HOME ORACLE_SID
cat /etc/motd
```

• Un archivo de usuario típico (.bash_profile):

```
PATH=$PATH:$HOME/bin
TERM=uvt1224 export TERM
setty intr \^C
cal
mesg n
```

• Un archivo de usuario típico bashrc (.bashrc):

```
alias dir=ls lf='ls -FC'
PS1='$PWD[!] >'
set -o vi
```

Los alias y otras definiciones típicas de bash deben ser definidas en el archivo .bashrc y no en el archivo .bash_profile; de otra manera algunos de los comandos no podrán utilizarlo. Recuerde las diferencias entre los archivos profile y bashrc. Los archivos profiles son leídos una sola vez durante el proceso de ingreso al sistema, pero el archivo bashrc es leído en ambas situaciones durante el login y cada vez que un nuevo shell de bash es invocado.

Explorando el Ambiente

Para visualizar el ambiente de shell, use los comandos set y env. El comando set se utiliza para manipular el ambiente actual. El comando set sin argumentos listara todas las variables actualmente definida en el shell. El comando set es usado para establecer varias opciones del shell. Algunos de los más comunes son listados adelante:

$ set	Sin argumentos presentará todas las variables.
$ set -o	Desplegará las opciones actuales del shell.
$ set -o opción	Prenderá la opción.
$ set +o opción	Apagará la opción.

$ set -o vi	Establece a vi como el editor de línea de comandos.
$ set -o allexport	Automáticamente exporta todas las variables nuevas.
$ set -o ignoreeof	Desautoriza CONTROL+D como teclas especiales de logoff.

El siguiente es un ejemplo:

$ set +o ignoreeof # autoriza CONTROL+D como teclas especiales de logoff

La parte exportada del ambiente puede ser listada usando el comando env sin argumentos. El comando env también es usado para ajustar el ambiente al hacer un llamando a un programa.

El comando unset removerá una variable o una lista de variables desde el shell y su ambiente, al igual que el comando export, unset necesita una lista de nombres de variables, así que recuerde no colocar el símbolo de $ antes del nombre de la variable.

Atajos a Directorios

Existen algunos atajos útiles para trabajar con directorios. El directorio home es normalmente definido por la variable HOME. Usted también puede utilizar el caracter (˜) tilde como sinónimo del directorio home. El directorio previo y actual también posee variables de bash y atajos.

Shortcut	**Versión normal**
cd	$ cd HOME
cd -	$ cd $OLDPWD
cd ~	$ cd $HOME
cd ~usuario	$ cd $HOME para el usuario
cd ~-	$ cd $OLDPWD

El bash define tres variables que corresponden a directorios usados comúnmente:

HOME	Directorio Home
PWD	Directorio Actual de Trabajo
OLDPWD	Directorio Previo de Trabajo

El caracter tilde puede también ser usado para abreviar éstos directorios para conveniencia de su uso desde la línea de comandos:

~	$HOME
~+	$PWD
~-	$OLDPWD

Un tilde seguido inmediatamente por el nombre de un usuarios interpretado como el directorio home del usuario, como es definido el home en el archivo /etc/passwd.

Ejercicio 7-1: El Ambiente del Shell

En éste ejercicio cubrimos los conceptos de un ambiente de proceso donde usamos el shell para configurar el ambiente. El uso de las variables de ambiente para modificar el comportamiento de programas es introducido así como la habilidad del usuario de configurar su ambiente usando los archivos de profile (perfil). La solución a éste ejercicio se encuentra en el Apéndice A.

1. ¿Cuál es el valor de las siguientes variables del shell en su sistema?
   HOME
   TERM
   PATH
   PS1

2. Defina algunas variables personales, recuerde proteger algunos caracteres de su valor de variable del shell.

   Establezca el nombre de la variable name para que tenga el valor: FCAD

   Despliegue su valor:
   $ echo $name

   Establezca la variable para que tenga el siguiente valor: <4 espacios>Padre Pina 102 (la dirección estará indentada 4 espacios en blanco).
   Despliegue su valor:
   $ echo $direccion
   $ echo " $direccion"

   Si estableció correctamente los valores, las salidas de éstos dos comandos debe aparecer un poco diferente.

   ¿Por qué?

3. Para demostrar el manejo de variables embedidas en texto, establezca una variable llamada nombre y dele el siguiente valor:
   FCAD<Tab>

   Ahora Despliegue el valor de la variable nombre seguida inmediatamente por la cadena Fundación Permita que el tabulado contenido en el nombre separe el valor y no tener que hacerle echo a ningún espacio.

4. Para poder entender el propósito de exportar las variables, escriba la siguiente secuencia de comandos:

$ echo $TERM	# Tome apunte del valor arrojado
$ unset TERM	# Remueve el valor asignado a la variable TERM en memoria
$ echo $TERM	# Usted eliminó el valor de $TERM
$ man bash	# No significa nada al usar bash
$ TERM=	# Valor nota anteriormente
$ man bash	# Aún persiste el problema: ¿Por qué?
$ export TERM	
$ man bash	# Ya todo esta como cuando empezamos

   Pregunta: Explique brevemente el proposito de exportar la variable TERM.

5. Pruebe los siguientes comandos. Asegúrese de entender lo que hace cada comando:
   $ cd
   $ cd /etc
   $ cd -
   $ pwd
   $ cd -
   $ pwd
   $ ls ~
   $ ls ~root
   $ cd
   $ ls ~-

6. Personalice su archivo profile. Agregue las siguientes líneas al final de su archivo .bash_profile:
   # cambios locales
   cal
   mesg n

¿Cuál es la diferencia del proceso de ingreso al sistema/login?

7. Para modificar su prompt para que incluya el directorio actual, escriba los siguientes comandos:
   $ cd
   $ PS1=”$PWD[$$] “

   ¿Qué efecto refleja éste comando en el prompt de su pc?

   Cambie al directorio /etc para ver si el prompt refleja el nuevo directorio. ¿Si no, por qué no?

   Escriba el comando correcto para establecer su prompt para que le incluya su directorio actual.

Ejercicio 7-2: Cambiando el Ambiente

Las soluciones a éste ejercicios se encuentran en el Apéndice A.

1. Cambie su ambiente para que las páginas man se visualicen con more y no con less.

2. Personalice su ambiente con definiciones especificas de bash. Habilitaremos permanentemente la edición con vi de la línea de comandos para todas las futuras sesiones de login (en bash) y agregaremos algunos alias muy útiles, que también serán permanentes, como los siguientes:

dir	**ls**
h	**history**
lf	**ls -FC**
la	**ls -la**

   Salga del sistema e ingrese de nuevo para verificar los cambios.

   Ejecute un segundo shell de bash, usando el comando:
   $ bash y asegúrese que su alias aún son reconocidos. Salga del shell bash.

3. Manipule opciones en su shell interactivo. Ajuste su sesión de login para deshabilitar CONTROL+D como combinación de teclas para salir de sesión pero permita que CONTROL+D salga de otros shell pero no de la sesión Salga y entre de sesión para verificar los cambios. Ejecute un segundo comando:
   $ bash

   Asegúrese que CONTROL+D es aún reconocido y sale del shell.

4. No todos los comandos son iguales. Alguno de ellos no existen como un programa por separado en el disco para éstos en el shell no usa la variable PATH para encontrarlos. Salve y restablezca su ruta de búsqueda con :
   $ SAVEPATH=$PATH
   $ echo $SAVEPATH
   $ PATH=

   Ahora compruebe algunos comandos estándares:
   $ pwd
   $ls
   $wc pass1
   $history

   ¿Por que algunos comandos funcionan y otros no?

   Use el comando type para verificar.

   Restaure su ruta de búsqueda con éste comando:
   $ PATH=$SAVEPATH

5- Vamos a practicar configurando y exportando variables. Defina la variable Var1, dandole un valor que

usted escoja y no la exporte.

Revise el estado de las opciones de su shell actual. Identifique y establezca la opción que obliga que todas las variables nuevas sean exportadas. Defina la variable Var2 y dele otro valor y no la exporte.

Inicie un nuevo shell e imprima las 2 variables. Pruebe los comandos set y env en éste nuevo shell.

Reconocen los comandos sus variables?

Inicie un shell más y repita la exploración.

Finalmente antes de cerrar las secciones adicionales, ejecute el comando ps -j y identifique la relación padre/hijo entre los 3 Shells.

¿Quién es el proceso padre de su Shell principal?

El Shell de Bash

En esta sección se provee una lista amplia de la línea de comandos de bash. Las ideas básicas de la generación de nombres de archivos y comandos presentadas en esta sección son comunes a la mayoría de los shell. Discutiremos los siguientes tópicos:

- Leer la línea de comandos
- Expandir Caracteres de Tipo Comodín
- Generación de Nombres de Archivos
- Uso de Comillas
- Establecer Alias
- Comando History
- Repetir Comandos
- Edición de la Línea de Comandos en bash
- Edición de la Línea de Comandos en Modo vi
- Resumen de Comandos vi
- Edición de la Línea de Comandos en Modo emacs

Las características brindadas por el shell bash como los alias, edición de la línea de comandos y el comando history, son únicas de éste shell. El shell bash es el Interprete de Comandos estándar (CLI) en GNU/Linux. Muchos de los comandos que discutimos en esta sección no tendrán una página man propia; información de ellos puede ser encontrada en la propia página man de bash o bash2.

Leer desde la Línea de Comandos

Al escribir un comando el prompt del shell, podemos escribir hasta que queramos y al final presionar la tecla ENTER, enviando el caracter línea nueva al shell. Este caracter es especial del shell, y el shell lo sabe interpretar como el marcador de línea nueva.

Una vez el shell recibe el mensaje de nueva línea (abreviado </n>), no continuará leyendo el resto de la línea. Este proceso involucra la lectura de la línea de izquierda a derecha, un caracter a la vez. Cada caracter que el shell lee es interpretado y la línea es recreada.

Si un caracter es normal, será colocado nuevamente en la línea. Cualquier caracter especial o metacarácter será interpretado y el resultado de esa acción será substituida nuevamente en la línea.

Si escribimos:

```
$ ls   -1  $HOME/trabajos/*
```

Entonces presionamos ENTER, el shell lee cada caracter en esta línea buscando caracteres especiales que responden a cierta acción. El shell encontrará el símbolo de peso ($) y el asterisco (*). Ambos caracteres son especiales y necesitan evaluación. El resultado de esta evaluación será entonces insertada en la línea de comandos. Después de esto la línea será recreada así:

$ ls -l /home/usuario/trabajos/carta.txt /home/usuario/trabajos/oficio.doc /home/usuario/trabajos/..... etc....

Ahora, Después de leer e interpretar la línea de comandos, y sólo ahora, el shell haría el intento de encontrar el comando.

En lo que queda de capítulo y en el resto del libro, estudiaremos el comportamiento de varios metcarácteres y su uso y que acción el shell ejecuta al enfrentar éstos caracteres especiales.

Los conceptos presentados aquí son verdaderos para todos los shells de GNU/Linux. El shell puede ser llevado a completar comandos y nombres de archivos en la línea de comandos utilizando la tecla de TAB. Aquí ponemos un ejemplo, si se dispone a abrir un archivo cuyo nombre es largo y deseamos economizarnos teclearlo en su totalidad podemos hacer lo siguiente:

$ vi nombre-archivo<TAB> # completaría el nombre de archivo al menos que existan otros iguales

Digamos que en éste caso en el directorio actual tiene dos archivos de nombres largo (nombre-archivo-largo. txt1 y nombre-archivo-largo.txt2) si desea que el shell complete simplemente tiene que escribir la primera letra que contienen los dos archivos, en éste caso la “n”, y luego pulsar la tecla TAB:

$ vi n<TAB> # completaría hasta la primera diferencia entre los nombres
$ vi nombre-archivo-largo.txt # si observamos llego hasta tener que elegir la diferencia que es el 1 o el 2

Si escuchas un bip de las bocinas del sistema al presionar la tecla TAB es que encontró varios archivos que igualan. Presione la tecla TAB de nuevo y verá una lista que desplegará las posibilidades de archivos, incluyendo los que desea y lo que no deseas:

$ vi nombre-archivo-largo.txt<TAB+TAB> #observamos tiene que elegir la diferencia que es el 1 o 2
nombre-archivo-largo.txt1 nombre-archivo-largo.txt2
$ vi nombre-archivo-largo.txt<1 o 2> # debe elegir la diferencia que es el 1 o el 2

Después de su elección entre los dos archivos tendrá que escribir la diferencia y si no es el fin del archivo podrá oprimir la tecla TAB de nuevo para continuar completando hasta la próxima diferencia o que se complete:

$ vi nombre-archivo-largo.txt1<ENTER> # después de elegir 1 o 2 sólo tendrá que oprimir ENTER

En éste punto ya su archivo se abrirá en el vi. Revise las páginas man de bash para aprender otros métodos de completar desde la línea de comandos, incluyendo variables, nombres de archivos, nombre de directorios, nombres de host y comandos.

Expandir Caracteres tipo Wildcard

Símbolos especiales como un asterisco (*) puede que reemplaze uno o más caracteres. Mejor conocidos como comodines o wildcards. Son usados para igualar el nombre de múltiple archivos y directorios. Los caracteres tipo wildcard son interpretados por el shell y no el comando. La expansión de los wildcards es llevado a cabo durante la etapa de lectura antes de la ejecusión del comando; antes de que el shell tenga una oportunidad de revisar si el comando existe o no en la ruta de los comandos.

Lo que se escribe	Lo que el Shell pasa al comando
archi*	archivo.txt archivo.doc archive.swi archiv.png
/home/*	/home/miguel /home/ivellisse /home/jazmine /home/desi /home/mike

El shell igualará el patrón del wildcard con los nombres de los archivos en el sitio indicado en el sistema de archivos. En el ejemplo anterior, el shell comparó en el directorio actual por todos los archivos y directorios cuyo nombres empezaban con archi y continuaban con cualquier número de caracteres más. En el segundo ejemplo, el shell revisa contra todos los archivos en el directorio /home y los argumentos se le pasan al comando con sus rutas absolutas, lo cual incluye el componente /home. Todos los comandos sólo verán los nombres de los archivos expandidos y sus rutas absolutas.

Recuerde que a diferencias de otros sistemas operativos, el patrón caracter (*) en UNiX iguala más allá de los puntos que en otros sistemas operativos el punto es parte de un concepto de extensión, y aquí al igual que en UNiX, en GNU/Linux el punto sólo es un caracter válido más, notemos éste ejemplo:

cart* <iguala posibles candidatos> # puede ser: cart carta carta.txt carta.do carta.txt.doc

Generación de Nombre de Archivos

Aquí los caracteres wildcard (comodines) de GNU/Linux:

*	Iguala cualquier número de caracteres incluyendo ninguno.
?	Iguala un caracter único.
[lista]	Iguala cualquier caracter único en la lista especificada entre los []
[!lista]	Iguala cualquier caracter único que no éste en la lista entre los []

Los nombres de archivos que sus nombres empiezan con un punto normalmente no son igualados con el comando ls y son llamados archivos ocultos. Aquí le presentamos algunos ejemplos de como poder desplegar éstos archivos:

$ ls -d *	# Todos los archivos en el directorio actual que sus nombres no empiezan con punto.
$ ls-d .*	# Todos los directorios que sus nombres empiezan con punto en el directorio actual.

Los dos caracteres de wildcard más usados comúnmente son ? y *. El caracter que se incluye entre las llaves [] iguala un único caracter. Así que es lo mismo:

$ ls -d carta?txt <que si escribimos> $ ls -d carta[.]txt -- pero sólo en el caso de carta.txt

El asterisco iguala cualquier número de caracteres. Así que por ejemplo en el caso de carta* vimos como el shell iguala cualquier número de caracteres después de la raíz dada en el argumento que es carta. Los corchetes [] además de poder suplir una lista a igualar podemos suplir un rango de caracteres que simplemente sean separado por un guión (-). Si el primer caracter entre las llaves es un (!), indicaría cualquier caracter que no sea uno de los incluidos en la lista.

Aquí arrojamos algunos ejemplos:

carta[abc]	carta seguido por a, b, o c.
carta[a-z]	carta seguido por cualquier caracter en letras minúscula.
carta[a-eABCDE]	ligado del rango de la a-a la-e y además A, B, C, D o E.
carta[!a-eABCDE]	igual que anterior pero indicando que no iguale lo que esta en los corchetes.
carta[*?]	Los asteriscos y acertijos entre las llaves significan su valor literar no el wildcards.

La generación de nombres de archivos puede ser detenida usando el siguiente comando:

$ set -o noglob

Hablaremos más sobre el comando set más adelante.

Ejercicio 7-3: Generación de Nombre de Archivo

Las soluciones a éste ejercicios se encuentran en el Apéndice A.

¿Qué se lograría con los siguientes comandos?

1. $ ls *.?
2. $ more [A-Z]*
3. $ ls /etc/[!a-m]*
4. $ file /usr/bin/*x*

5. $ ls [a-z]*[0-9]
159
6. $ ls -a .[!.]*
7. $ ls -d /etc/*.d/*

Uso de las Comillas

Las comillas son utilizadas para evitar que el shell reconozca cierto caracteres especiales:

‘’	Comillas simple evitan el reconocimiento de todos los caracteres especiales.
“ ”	Comillas doble protegen la mayoría de los caracteres especiales con algunas excepciones.
\	La barra invertida escapa al próximo caracter sin importar si es especial o no.

Las comillas también son usadas para preservar espacios y tabulados. Aquí le mostramos un ejemplo de esto:

```
$ echo *.txt
uno.txt dos.txt tres.txt
$ echo ‘*.txt’
*.txt
$ echo Carta.txt Carta.doc
Carta.txt Carta.doc
$ echo ‘Carta.txt Carta.doc’
Carta.txt Carta.doc
```

El caracter barra invertida conocido en inglés como backslash (\), causa que el siguiente caracter sea interpretado literalmente y no como caracter especial. Para poder continuar escribiendo comandos en el CLI después de haber agotado hasta el final de la pantalla simplemente escriba una barra invertida antes de presionar la tecla ENTER. La barra invertida evita que el shell reconozca la tecla especial ENTER.

Comillas sencillas son usadas para prevenir que el shell reconozca todos los caracteres especiales. Todo dentro de estas comillas simple es visto como un sólo argumento. Comillas doble son menos estrictas que las sencillas. Ellas protegen el shell de reconocer la mayoría pero no todos los caracteres. Los caracteres que son excepción de las comillas doble son:

“”	Comillas doble misma.
\	Backslash (caracter de comilla única).
‘	Back-quote, tick (Caracter de substitución de comando).
$	Símbolo de Peso/Dollar (variable de substitución de caracter).

Para incluir unas comillas dobles dentro de una cadena con comillas dobles, preceda la doble comilla con el caracter de escape (“\”). Cuando el shell lee la línea de comandos para dividirla en argumentos, cualquier comillas son divididas dejando la cadena dentro de la comilla como un sólo argumento.

El uso más común de mecanismo de comillas es para:

- Deshabilitar la generación de nombres de archivos; cuando queremos pasar un caracter wildcard a un comando.
- Permitir la inclusión de espacios o tabulados en los argumentos de los comandos.

Crear Alias del Bash

El shell de bash nos provee la características de la creación de alias. Alias son nombres que pueden contener cualquier caracter excepto los caracteres especiales del shell, ni el símbolo de igual. Los alias son usando para crear abreviaturas o nombres alternativos a los comandos, he aquí la sintaxis.

```
$ alias dir=ls
$ alias lf=’ls -Fl’ rm=’rm -I’ cp=’cp -I’
```

Usted puede listar los alias actuales utilizando el comando alias sin ningún argumento. Usted puede ejecutar el nuevo comando simplemente escribiendo su nombre. Este trabajará como un comando interno del

shell usted también puede pasarle opciones a éste comando del cual usted creó un alias y ellas serán agregadas a esas predefinidas en dicho alias. Los caracteres especiales como el espacio o los comodines deben residir dentro de comillas al definir el valor de un alias.

La sustitución de alias ocurre antes de cualquier otro reemplazo en la línea de comandos por el shell , es decir, el alias es lo primero interpretado por el shell. Los caracteres espaciales en la definición de alias son reconocidos como si hubiesen sido digitados en la línea de comandos. Por ejemplo:

$ alias lsnum='ls [0-9]*'

Listará todos los nombres de archivos que empiecen con un dígito.

Los alias sólo aplican al shell actual y se pierden en el momento que el shell termina; existe una manera de evitar que esto suceda, agregandole la sentencia al archivo de ingreso al sistema el archivo profile el cual es ejecutado en cada login. En éste archivo usted puede colocar sus definiciones de alias favoritos. El administrador de sistema también puede proveer un conjunto estándar de alias en el archivo profile de ingreso del sistema, en cual es un archivo profile especial que se ejecuta cada vez que alguien ingresa al sistema. Usted puede obviar éstos alias globales con sus propios archivos profile o cancelar en su totalidad el archivo profile de ingreso. En realidad usted tiene 2 archivos profiles, que son .bashrc .bash_profile, los cuales son asociados con ingresos no interactivos e interactivos respectivamente. Hablaremos de éstos más adelante.

El Comando history

El mecanismo del comando history nos permite accesar a comandos previamente escritos en la línea de comandos. Podemos llamar comandos ya escritos y ejecutarlos inmediatamente o editarlos y luego ejecutarlo. El bash almacena sus comandos en un archivos de nombre history. Los comandos pueden ser llamados y ejecutados de nuevo, o llamados para ser editados. El prompt de bash puede ser configurado para incluir el número del comando actual, he aquí algunos ejemplos del uso del comando history.

```
$ history
220 echo jose Paredes
221 echo 'Jose Paredes'
222 echo alias dir=ls
223 echo alias lf="ls -Fl" rm='rm -i'
224 echo cp='cp -i'
225 echo cp /etc/passwd pass1$
```

Comandos internos adicionales son proveídos para soportar la llamada directa de los comandos previos.

Repetir Comandos

El shell bash provee un comando interno único (fc) para dar soporte al mecanismo de comando history. El shell bash también soporta alias de comandos para el history y el !, los cuales también son usados para proveer nombres más significativos para el comando fc. Usted puede repetir comandos por número o por nombre.

Use el comando (!comando) para especificar la primer instancia del comando pasado como un argumento:

!n	Repite último comando de número n.
!pwd	Repite último comando que empieza con pwd.

Aquí le presentamos un ejemplo usando ! para repetir comandos:

```
$ pwd
/home/ivelis
$ ls
mail trabajo
```

$!pwd
/home/ivelis

La variable FCEDIT es usada para definir el editor a usar con el comando fc. Usando el comando fc, un bloque de comando previos pueden ser insertado en un archivo temporal, editado y entonces invocado para proveer un simple script de shell. Más adelante hablaremos del uso de variables y las cadenas del prompt más adelante.

Edición de la Línea de Comandos de Bash

El shell de bash tiene dos modos de operación para su uso de edición de la línea de comandos y un mecanismo para el comando history. Cada modo soporta características similares pero usan combinaciones de teclas diferentes para editar la línea de comando o llamar comandos previos. El modo vi emula los comandos del editor estándar de GNU/Linux vi, y el modo de emacs emula el editor popular de dominio publico emacs. Los dos modos de edición son mutuamente exclusivo y significativamente diferente. El comando sed es usado para establecer el modo de edición requerido.

$ set -o vi o $ set -o emacs

Edición de Línea de Comandos modo vi

El shell de bash usa vi para su edición de la línea de comandos emulando la gran mayoría de los comandos de vi. Una amplia lista de los comando vi reconocido por el shell de bash es incluida en las páginas man de bash. Aquí algunos puntos importantes a recordar cuando uses la edición vi de bash:

- El mecanismo de history normalmente se inicia en el modo de inserción.
    - Los caracteres escritos son insertado en la línea de comandos.
    - Para borrar el último caracter utilice la tecla BACKSPACE.
- Presione ESC para entrar en el modo de comando.
- Edite la línea usando el comando vi.
- Llame comandos anteriores usando las teclas de cursores (flecha arriba, flecha abajo).
- Ejecute búsqueda de comando previos usando /.
- Si no posee teclas cursores deberá usar:
    - h para moverse a la izquierda
    - j para moverse hacia abajo
    - k para moverse hacia arriba
    - l para moverse hacia la izquierda

Resumen de Comando Modo vi

La siguiente tabla mostramos un resumen de los comando más comunes de edición del modo vi. Cada comando puede ser precedido por un valor multiplicador el cual se utiliza de la siguiente forma. Aquí presentamos para ilustrar un ejemplo útil de lo anteriormente dicho: en vi desde el modo de comando escriba el número 7(o cualquier número), luego presione la i para entrar en el modo de inserción y escriba la palabra hola, luego ENTER y finalmente presione ESC. Esto producirá que la palabra hola sea insertado el números de veces digitada durante el ejercicio.

Presione las teclas ESC para entrar en modo comando:

Comando	Acción
h	mueve el cursor un caracter a la izquierda
l	mueve el cursor un caracter a la derecha
b	mueve el cursor al principio de la palabra anterior
w	mueve el cursor a la próxima palabra

0	mueve el cursor al principio de la línea
$	mueve el cursor al final de la línea
i	inserta texto antes del cursor
a	inserta texto después del cursor
I	inserta texto al inicio de la línea
A	inserta al final de la línea
X	borra hacia la izquierda
X	borra debajo del cursor
cb	cambia al inicio de la palabra
cw	cambia al final de la palabra
r	reemplaza caracteres
D	borra al final de la línea
-	línea anterior (también k)
+	próxima línea (como j)
/cadena	busca comandos que contengan cadena
/^cadena	busca comando

Presione ENTER para ejecutar la línea de comando editada.

Modo de Edición de emacs

Editar la línea de comando usando el mode emacs emula algunos de los comando del editor emacs. Las opciones de edición emacs presenta un sistemas alternativo para los usuarios que no les gusta los comando de vi y su modo de inserción. La combinación de teclas más usadas de emacs son similares a CTRL+TECLA. Los comando normales son insertado en la línea de comando en el momento en que se escribe.

La siguiente tabla muestra algunos de los comando emacs más usados.

Comando	**Acción**
^B	atrás un caracter
^F	adelante un caracter
^A	inicio de la línea
^E	fin de la línea
<BS>	borra hacia la izquierda
^D	borra debajo del cursor
ny	pega texto borrado
^K	borra hasta final de la línea
^P	línea anterior
^N	proxima línea
^Rcadena	busca comando que contengan cadena str
^R^cadena	busca comando str

Una característica poco documentada puede ser utilizada para que las teclas de cursores funcione en terminales de estándar ANSI. El teclado de la consola de pc y la mayoría de las estaciones de trabajo de GNU/Linux generán ANSI estándar secuencia de teclas para las teclas cursores.

Una tecla cursora ANSI generá un código de escape seguido por un corchete abierto y una letra usualmente mayúscula. Declarando un alias cuyo nombre es dos veces _ _(underscore underscore) y requiere una letra. Este alias puede ser invocado desde la edición de línea de comando emacs cuando la tecla relevante es presionada.

En la línea de comando, el nombre alias es definido en la manera usual, pero el valor debe ser especificado escribiendo la tecla (\) y seguido por las teclas de edición de emacs correspondientes.

Normalmente, los alias son definidos en un archivos de nombre .bashrc en el directorio home. Para entrar

un código de control a vi cuando se encuentre en el modo de insertar, use la secuencia CTRL+V antes del código de control que usted desea insertar. Por ejemplo, para definir la tecla cursora flecha arriba, escriba literalmente el siguiente comando vi:

$ alias _A=<CTRL - V><CTRL - P><CR><ESC>

Las teclas cursoras son definidas como sigue:

```
alias _ _A=^P       arriba
alias _ _B=^N       abajo
alias _ _C=^F       derecha
alias _ _D=^B       izquierda
```

Ejercicios 7-4: Uso del Shell Bash

En éste ejercicio le introducimos a los conceptos básicos de la línea de comando del shell y algunas características de bash que son utilizadas para simplificar la entrada de los comandos. Las soluciones a éstos ejercicios se encuentra en el Apéndice A.

1. Primero generamos unas cuantas líneas en el comando history (simplemente ejecutando unos cuantos comandos en el CLI); entonces practicaremos los mecanismos del comando history.

   $ w
   $ pwd
   $ ls -1
   $ more abiertos.txt
   $ more linux
   $ wc -l abiertos.txt linux
   $ history

   Ahora escriba:

   $!ls
   $!wc
   $!more #¿Cuál de los comandos fueron ejecutado?
   $!his
   $!n #Reemplace n con cualquier número del archivo history.

2. Ahora utilizaremos el editor vi para visualizar nuestro archivo history, localizaremos un comando y lo editaremos, y lo prepararemos para otra ejecución. Habilite la edición vi con éste comando:

   $ set -o vi

   Una vez iniciado el mecanismo con el comando set -o vi, usted permanecerá en el modo vi. Usted debe recordar presionar la tecla ESC antes de poder usar cualquier comando, lo que incluye las teclas cursoras.

   Use las teclas de edición para viajar hacia abajo a través del comando history hasta llegar al último comando ls ejecutado. No ejecute éste comando.

   Ahora viaje hacia arriba hacia el comando more y cambie el nombre del archivo carta.txt y ejecute el comando de nuevo

   Use el comando de búsqueda de la línea de comandos, llame el último comando ls y edite sus opciones a que sean -al.

3. Aquí practicaremos el uso de caracteres tipo wildcard. Utilizaremos la característica de generación de nombres de archivos (wildcard) del shell para encontrar todos los archivos y directorios debajo de la jerarquía /etc que iguala los nombres con las siguientes criterios. Para poner a prueba sus patrones de wildcard use

el siguiente comando:
$ echo <patron> o $ ls -d <patron>
donde <patrón> significa la cadena de búsqueda que usted escribiría. Por ejemplo:
$ echo m* # nombres de archivos que empiezan con m
Puede ser que usted encuentre la características de llamar y editar comando útil para esto.
Cambiese al directorio /etc :
$ cd /etc
Seleccione los archivos cuyos nombres:
• empiezan con "p".
• Termina con "y".
• Empieza con "m" y termina con "d".
• Empieza o con "e", "g" o "m".
• Contiene una "o" seguida por (pero no necesariamente de inmediato) por una "u".
• Contiene una cadena "conf" en cualquier lugar del nombre del archivo.
• Empieza con una "s" y contiene una "n".
• Contiene exactamente cuatro caracteres.
• No empieza con letras minúsculas.
• Contiene un dígito en algún lugar del nombre del archivo.

4. Vamos a seguir experimentando con los wildcards. Debe traerle a un entedimiento del concepto de quien hace que en los caracteres especiales en la línea de comandos.
Pruebe con los siguiente comandos:
$ cd
$ echo *[a-z]*
$ ls -d *[a-z]*
Luego pruebe:
$ echo *.*
$ ls -d *.*
¿Entiende usted la salida?

5. Vamos a probar definiendo alias. Cree un alias y llámelo h para el comando history el nuevo alias:
$ alias h=history; h
Agregue nuevo alias con los valores mostrados debajo, y entonces pruébelos:

Nombre del alias	Valor
cambdir	**cd**
dir	**ls -d**
minuz	**ls -d [a-z]***

Ejercicio 7-5: Expansión y Wildcards

Aquí tratamos más sobre la expansión de nombres de rutas y caracteres. La solución a éstos ejercicios se encuentran en el apéndice A.

1. ¿Qué cree usted que efectúan los siguientes comandos?
$ ls -d /*/p*wd
$ cd /usr/*term*/v
$ cd /usr/*/term*/[ab]

2. Para probar las cadenas wildcard, use el comando ls o echo. Use solamente rm cuando usted está seguro de los resultados.
- En su directorio home deberá crear un nuevo directorio y llámelo backup
- Cambiese a éste directorio
- Copie a éste directorio backup todos los archivos de /etc que su nombre empiecen con la letra l.
- Liste y luego clasifique el contenido de los archivos en el directorio backup e identifique todos archivos

de data o binarios. Y tome nota de esto.

- Dando uso a un sólo comando y con especificaciones de wildcard elimine todos los archivos de data o binarios anteriormente identificados

Herramientas Básicas

Trabajar en la consola se concentra al rededor de un conjunto de herramientas básicas que le permiten la navegación y manipulación de los archivos y su contenido en el sistema. Una vez dominas la línea de comandos le será posible utilizar al máximo estas herramientas sin pensarlo. Los conocimientos básicos de esta herramienta significa que podrás despachar tareas con gran eficiencia. En esta sección detallamos una gran porción de los utilitarios base que son el pan de cada día de trabajar desde la consola.

Los siguientes tópicos son discutidos sobre esta sección:

ls	Listar directorio.
cd	Cambiar directorio.
more y less	Paginadores.
cp	Copiar.
ln	Vincular directorios o archivos.
mv	Mueve archivos.
mkdir	Crear directorios.
rm y rmdir	Remover directorios o archivos.
head y tail	Vista parcial de archivos.
file	Determinar el tipo de archivo.
df y du	Espacio en libre en disco y usado.
tar	Empaquetador de archivos.
gzip	Compresión de archivos.

Comando ls- Listar Directorios

Talvéz uno de los utilitarios más utilizado en la línea de comandos. Usted encontrará utilizando esta herramienta sin saber por que lo esta haciendo. Ya que es una de las herramientas más usadas. Típicamente es usada con un gran números de opciones que varían su comportamiento para arrojarnos más información sobre los contenidos de los directorios. Por ejemplo el comando ls en el directorio actúal no podrá poveer los siguientes resultados:

```
$ ls -lF
total 24
drwx------ 25 ivelis admin 850 12 Feb 18:44 Desktop/
drwxr-xr-x 32 ivelis admin 1088 29 Jan 16:00 Documents/
-rwxr-xr-x 18 ivelis admin 612 23 Jan 20:40 Incomplete
drwx------ 26 ivelis admin 884 11 Feb 22:06 Library/
drwxr-xr-x 3 ivelis admin 102 4 Nov 18:55 Movies/
drwxr-xr-x 5 ivelis admin 170 12 Nov 14:21 Music/
drwxr-xr-x 3 ivelis admin 102 4 Nov 18:55 Pictures/
drwxr-xr-x 14 ivelis admin 476 23 Jan 20:05 Shared/
```

La salida de comando ls -lF tiene ocho partes básicas:

• Permisos • Vínculos • Dueños • Grupo • Tamaño de archivo
• Fecha de creación • Nombre de archivo • Tipo de Vínculo

Los permisos serán discutidos más adelante. Los vínculos se refiere al número de entradas en directorios que contiene una referencia a los archivos; ellas son descriptas en más detalles cuando hablemos del comando ln. El dueño se refiere al usuario ivelis que es propietario del archivo y el grupo admin que es la cuenta dueña del archivo. Es posible permitir a un grupo sin miembros ser dueño de un archivo. El Tamaño del archi-

vo esta representado el bytes y los archivos pueden tener cero bytes. Los directorios son creados con el comando mkdir y son un múltiple del tamaño de bloque lógico o 1024 bytes. Los nombres que terminan con una / son directorios.

Al escribir el comando ls modificado por la opción -l (largo) nos revela en la mano izquierda los códigos de los permisos. El primer caracter de esta columna nos revela el tipo de archivo. El caracter "-" en esta primera fila indica un archivo de datos regular que contiene flujo de bytes. La d nos indica que es un directorio.

Las próximas 3 columnas de permisos contiene una combinación de caracteres rwx y - donde r significa permisos de lectura (Read), w significa permisos de escritura (Write) y la x significa permisos de ejecución (Execute). El guión significa la negación de los permisos rwx.

Como podemos ver de la salida del comando anterior, existen tres conjuntos de columnas de permisos para un total de nueve columnas, es importante entender que sólo tres de los nueves son importante en cualquier momento en particular. El primer conjunto de tres son aplicables al dueño del archivo, en éste caso es ivelis, y son solamente activo si la cuenta de usuario, ivelis, esta tratando de accesar los archivos. El segundo conjunto de permisos se refieren al grupo dueño, o el grupo del dueño, y éste conjunto sólo esta activo si un miembro del grupo de admin esta tratando de accesar el archivo. El tercer y último conjunto de permisos se refiere a los otros o a cualquier usuario que no es el el dueño o miembro del grupo que trata de accesar el archivo.

Los archivos de data no pueden ser ejecutados, pero, el bit de ejecución puede ser establecido para los archivos sin ningún efecto sobre el éste. Si a un archivo de data se le establece el bit de ejecución y un intento de ejecutarlo es efectuado, entonces el shell lo evaluará y emitirá un mensaje de error indicando que éste no puede ejecutar el archivo binario. Si un archivo de data ASCII ha sido establecido como ejecutable y un usuario se encuentra tratando de ejecutarlo, el shell hará un intento de de interpretar el contenido del archivo como una serie de comandos del shell. Si deseamos algún interprete en particular, entonces el archivo ASCII deberá empezar con el caracter #! y la ruta del interprete en particular, por ejemplo, la siguiente línea activará el interprete de Perl para leer el contenido del archivo: **#!/usr/bin/perl**

Los directorios son muy parecidos a cualquier otro archivo de datos; aunque ellos no fueron diseñados para ser archivos ejecutables, ellos pueden tener su bit de ejecución activada. El permiso de ejecución en un directorio significa que nombres de archivos pueden y no pueden ser visto por el dueño, grupo o los otros. Esto sólo tiene significado si es para verlo; si el permiso de lectura es valido para la cuenta del usuario, entonces el archivo puede ser leído aunque su nombre no puede ser desplegado con el comando ls. Otra cosa es también que si los permisos de Lectura o Escritura no están establecidos en un directorio y el dueño, entonces ninguno de los archivos en el directorio pueden ser ni leído o escrito respectivamente.

Existen otros atributos de permisos, como son Set User ID (SUID), Set Group ID (SGID), y el sticky bit, éstas son discutidas con el comando chmod. Otra opción disponible del comando ls es (-t) que cambia el comportamiento de ls a desplegar basada en el tiempo de creación. Combinada con las opciones -lrt:

```
$ ls -ltr
total 24
drwxr-xr-x  3    ivelis admin    102    4    Nov    18:55 Pictures
drwxr-xr-x  3    ivelis admin    102    4    Nov    18:55 Movies
drwxr-xr-x  18   ivelis admin    612    23   Jan    20:40 Incomplete
drwxr-xr-x  32   ivelis admin    1088   29   Jan    16:00 Documents
-rw-r--r--  1    ivelis admin    9891   2    Feb    20:50 fir.rtf
drwx--- ---  26   ivelis admin    884    11   Feb    22:06 Library
```

Vemos que el orden es diferente al anterior ya que cambiamos el comportamiento de ls con la opción (-t), y nos despliegue basado en la fecha de creación, observe como los meses incrementan.

Comando cd- Cambiar de Directorio

El comando cd no es un programa autónomo como son las mayorías de los comandos. El cd es un comando interno del shell, a éstos comandos se le refiere como built-in y son parte del shell. La mayoría de las veces el comando se usa en su forma más simple que es escribiendo "cd <nombre-directorio>", para el usuario cambiarse a ese directorio. En los siguientes ejemplos, el prompt del shell bash ha sido configurado para desplegar información adicional. El prompt nos muestra el nombre de la cuenta (ivelis), seguido por el nombre del host (maquina1), seguido por el directorio actual. Por ejemplo:

```
ivelis@maquina1:~ > cd /usr
ivelis@maquina1: /usr>
```

Aquí el símbolo de tilde ~ nos indica que nos encontramos en el directorio home de ivelis; /home/ivelis/. El comando cd significa cambiar de directorio actual. El uso del punto sencillo (.) y el doble (..) permite hacer referencia al directorio mismo y al directorio padre que contiene el directorio en el cual nos encontramos. Un ejemplo del uso del (.) es:

```
$ ivelis@maquina1:/usr > cd ./share/man
$ ivelis@maquina1:/usr/share/man >
```

El .. le mueve al directorio padre:

```
$ ivelis@maquina1:/usr > cd ..
$ ivelis@maquina1:/usr /share>
```

Existe una clase especial de archivos llamados vínculos simbólicos (symbolic links). Estos archivos normalmente contienen la ruta al archivo real pero también puede ser a un directorio. Si usamos el comando cd para cambiar a un directorio representado por su vínculo simbólico, el vínculo es seguido automáticamente al archivo cual el vínculo apunta. Por defecto, el comando cd usa la opción (-L) y nos muestra en nombre vinculado simbólicamente. Si usamos la opción (-P), entonces el verdadero nombre del directorio es desplegado (los que son hard link).

Es típico que un usuario cambie de directorio y luego retorne a su directorio home. Como el shell recuerda su directorio previo, en una variable de nombre $OLDPWD, es posible moverse entre dos directorios diferente usando la sentencia cd -. Por ejemplo, si su directorio anterior fué su directorio home, /home/ivelis, y su directorio actual es /usr, entonces el comando "cd -" tendría el siguiente efecto:

```
$ ivelis@maquina1:/usr > cd -
  /home/ivelis
$ ivelis@maquina1:/usr > cd -
  /usr
$ ivelis@maquina1:/usr > cd -
  /home/ivelis
```

Asociados con el comando cd están los comandos del shell dirs, pushd, y popd. El shell de bash mantiene una lista de los directorios que ella recuerda. El comando dirs listará esta lista. El comando dir tiene el siguiente sintaxis:

```
dirs [-clpv] [+N] [-N]
```

Esta lista de directorios funciona como una pila. El comando pushd empuja un directorio a la lista y el comando popd lo quita de la lista. Aquí le proveemos un ejemplo de los comandos pushd y dirs. En éste ejemplo empezamos desde el directorio home (~) y navegamos a través de varios directorios del sistema.

```
$ ivelis@maquina1:/usr > dirs
  ~
$ ivelis@maquina1:/usr > pushd /etc
  /etc ~
$ ivelis@maquina1:/usr > pushd /bin
  /bin /etc ~
$ ivelis@maquina1:/usr > pushd /etc/opt
  /etc/opt /bin /etc ~
$ ivelis@maquina1:/usr > dirs
  /etc/opt /bin /etc ~
```

En éste momento tenemos ya cuatro directorios recordados. La opción (-l) le dice a dirs que no utilice versiones acortadas de desplegar los directorios como es ~, para representar el directorio home.

```
$ ivelis@maquina1:/usr > dirs -l
  /etc/opt /bin /etc /home/ivelis
```

La opción -v causaría que el comando dirs imprima cada directorio en su propia línea y asociada con un número, así:

```
$ ivelis@maquina1:/usr > dirs -v
0 /etc/opt
1 /bin
2 /etc
3 ~
```

La opción (-p) es la misma que la (-v) pero sin las asociación de los números.

```
$ ivelis@maquina1:/usr > dirs -p
/etc/opt
/bin
/etc
~
```

Las opciones +N y -N despliegan la entrada número N empezando desde el cero (0). +N cuenta de izquierda a derecha, y -N cuenta de derecha a izquierda, veamos éste ejemplo:

```
$ ivelis@maquina1:/usr > dirs +1
  /bin
$ ivelis@maquina1:/usr > dirs -1
  /etc
```

La opción (-c) limpia todos los elementos de la pila, pushd y popd le permiten al usuario manipulara la pila del directorio. El comando pushd coloca el nombrado directorio en la cima de la pila, pushd sin ni gún argumento causa que las dos entradas de directorios sean intercambiados igual que lo efectúa el comando “cd -”. Las opciones disponibles al comando cd son:

```
pushd [dir | +N | -N]
```

La opción de pasar un dir (directorio) como argumento agregará el directorio a la cima de la pila y lo convertirá en el directorio actual de trabajo o mejor conocido como el PWD.

```
$ ivelis@maquina1:/usr > pushd ~/compartidos/juegos/
  ~/compartidos/juegos ~
```

Las opciones +N y -N manipularan la pila para que el directorio número N éste encima de la pila y el directorio actual. La opción +N empieza su conteo desde la izquierda de la lista y -N empieza desde la derecha de la lista. Por ejemplo:

```
$ ivelis@maquina1:/usr > dirs
  /home /bin /etc ~/compartidos/juegos ~
$ ivelis@maquina1:/usr > pushd +3
  ~/compartidos/juegos ~ /home /bin /etc
```

El comando popd remueve los directorios desde la pila. Sino se da ningún argumento, éste remueve los directorios desde la cima de la pila y cambia el directorio actual del usuario llevandolo a la cima.

popd [+N | -N]

Las opciones +N y -N son similares a las opciones del comando pushd excepto que ellas remueven la entrada número N de la pila. Por ejemplo:

```
$ ivelis@maquina1:/compartidos/juegos > dirs
  ~/compartidos/juegos /home /bin /etc
$ ivelis@maquina1:/compartidos/juegos > popd +3
  ~/compartidos/juegos /home /bin
$ ivelis@maquina1:/compartidos/juegos > dirs
  ~/compartidos/juegos /home /bin
```

Paginadores more y less

Existen dos comandos para ver y navegar documentos de textos. Estos comandos son more y less. El comando more, el más antiguo de los dos, y con menos características; el comando less será discutidos en esta sección. Con el comando less, las teclas cursoras pueden ser utilizadas para moverse una línea a la vez. Las teclas PAGEUP y PAGEDOWN le trasladan una página a la vez.

Con el comando less, podemos buscar cadenas de caracteres especificadas con la opción -p; el comando less buscara en el archivo y resaltara todas las coincidencias de la cadena. Para buscar un patrón en pa ticular inicie el comando less desde el prompt con un patrón de búsqueda así:

$ less -p gnu/linux abiertos.txt

Este comando buscara la palabra gnu/linux en el archivo de nombre abiertos.txt y resaltara todas las coincidencias. Dentro del paginador podemos buscar ocurrencias de las cadenas así:

<ESC> /gnu/linux <ENTER>

El comando less buscará la palabra gnu/linux dentro del documento. El comando less también puede tomar varios archivos como argumentos y permitirle navegar a través de ellos. Por ejemplo si escribimos:

$ less archivo1.txt archivo2.txt archivo3.txt

Esto cargará los tres archivos a la vez. Mientras el usuario lee uno de los archivos se puede mover hacia el otro usando éstos dos comandos.

:n **para ir al próximo archivo**
:p **para ir al archivo previo**

Mientras, en el comando less, el archivo desplegado actualmente puede ser editado. Escribimos la letra "v" para invocar el editor por defecto mencionado en la variable de ambiente VISUAL o EDITOR. Si no esta definida el editor será el vi. Después del archivo ser editado y hallamos salido retornaremos al comando less.

Números de líneas pueden ser desplegados con el comando less si simplemente iniciamos el comando less con la opción -N. Cada línea en el archivo será precedida por un número. Esto ayuda cuando estamos bu cando por el número de una línea que contiene un error. Por ejemplo:

$ less -N archivo.txt

Comando cp- Copiar

El comando cp efectúa copias de archivos y directorios. Este comando mantendrá las propiedades del archivo que se copia pero no preserva la propiedad de éste. Con la opción -P preservamos todas las propiedades de un archivo.

En el siguiente ejemplo, copiamos el archivo /etc/passwd, el cual es propiedad de root y del grupo root, Después de copiarlo, ahora es del usuario ivelis y del grupo admin, aunque el resto de los permisos pe manecen iguales. también observe que ahora hay dos copias del archivo en el sistema.

```
$ ls   -l  /etc/passwd
  -rw-r--r--    1    root   root   1374   13 Oct 02:54   /etc/passwd
$ cp /etc/passwd   .
$ ls -l  passwd
-rw-r--r--   1  ivelis   admin   1374   13 Feb 21:38   passwd
```

Por defecto el comando cp no copia directorios. La opción -R deberá ser especifícada para poder hacer copias de los directorios. La opción le indica al comando cp que recursivamente copie el contenido de directorios y archivos regulares. Por ejemplo, si el comando:

```
$ cp /etc/* /etc_backup
```

Sólo los archivos en la jerarquía superior del directorio serán copiados. Pero si ejecutamos el mismo comando con la opción -R, entonces todos los archivos y directorios serán copiados. En el siguiente ejemplo, los archivos y directorios de la jerarquía /usr son copiados /home/usuario.

```
# ls
# cp -R /var/log/* .
# ls
```

CDIS.custom	ftp.log.0.gz	ipfw.log	lookupd.log.0.gz	lpr.log.2.gz	monthly.out	ppp.log
OSInstall.custom	ftp.log.1.gz	ipfw.log.0.gz	lookupd.log.1.gz	lpr.log.3.gz	netinfo.log	sa
asterisk	ftp.log.2.gz	ipfw.log.1.gz	lookupd.log.2.gz	mail.log	netinfo.log.0.gz	samba
cups	ftp.log.3.gz	ipfw.log.2.gz	lookupd.log.3.gz	mail.log.0.gz	netinfo.log.1.gz	system.log
daily.out	httpd	ipfw.log.3.gz	lpr.log	mail.log.1.gz	netinfo.log.2.gz	system.log
fax	install.log	lastlog	lpr.log.0.gz	mail.log.2.gz	netinfo.log.3.gz	sys.log.1.gz
ftp.log	install.log.0.gz	lookupd.log	lpr.log.1.gz	mail.log.3.gz	ppp	wtmp

En muchos de los casos, puede ser que éste copiando un archivo que ya existe donde desea copiarlo. Para esta situación cp tiene muy buenas y útiles opciones. La primera es la opción -f, la cual forza el copiado sin querellarse de que ya el archivo existe. La segunda opción útil es la -i, la cual causa que el comando cp se detenga y nos pida confirmación cuando copia un archivo que ya existe en el punto de destino. Por último la opción -u la cual sobrescribe archivos, solamente si el archivo copiado es más nuevo que el archivo que reemplaza.

Comando ln- Vínculo a Directorio o Archivo

El comando ln es usado para crear un vínculo (link) o atajo entre dos archivos o a un directorio. Un link simplemente es un puntero a otro nombre de archivo o directorio. Por defecto, el comando ln crea hard link para vincular dos o más archivos o directorios. Los archivos que tienen un sólo nombre tienen una entrada en un directorio que está compuesta de un nombre de archivo asociado con un número de inodo (inode) que apunta al inode. El inode describe el contenido del archivo. Los hard links son implementados creando otra entrada en el directorio con un nombre nuevo pero con el mismo número de inode. Para contabilizar el número de hard links, el inode también mantiene una cuenta del número de link. Un hard link es una ruta a otro archivo o directorio, de forma que, si el contenido del vínculo es alterado, también el archivo original será cambiado. Por ejemplo:

```
$ cat  > original.txt
   Este es el archivo original.txt
$ ln original nuevo.txt
$ echo Texto que se le agrega al archivo nuevo.txt >> nuevo.txt
$ cat original.txt
  Este es el archivo original.txt
```

Texto que se le agrega al archivo nuevo.txt

```
$ ls -l
total 16
-rw-r--r--   2   ivelis  admin   76   13 Feb 23:04    nuevo.txt
-rw-r--r--   2   ivelis  admin   76   13 Feb 23:04    original.txt
```

En éste ejemplo, el texto agregado al archivo de nombre nuevo.txt, también tuvo el efecto de cambiar el archivo de nombre original.txt. Note como el nuevo hard link y el archivo original.txt, ambos, tienen una longitud de 76 caracteres. Note además que el número entre los permisos y el nombre del usuario dueño es (número 2) el número de hard link del inode de los archivos de nombres original.txt y nuevo.txt. La desventaja de usar un hard link es que los vínculos deben estar dentro del mismo sistema de archivos esto es por causa de que los números de inode son solamente relativos al mismo volumen y no relativos a si temas de archivos.

El comando ln puede también crear vínculos simbólicos (symbolic links) usando la opción -s. Un symbolic link es justamente un puntero a otro archivo, pero éste puede hacer referencia a un archivo en cualquier lugar del sistema de archivos. Los vínculos simbólicos son similares a los hard links, con la excepción de que son más lentos de accesar que los hard links. La razón es que, una vez el administrador de archivos descubre un vínculo simbólico, éste debe reiniciar el proceso de buscar la ruta. Otra manera en que el vínculo simbólico difiere del hard links, es en que el tamaño del archivo simbólico es igual al tamaño del nombre en sí, y, no al del archivo. Fíjese que, en el siguiente ejemplo, el archivo de nombre nuevo.txt tiene una longitud de 12 bytes, mientras que, el archivo de nombre original.txt es de 76:

```
$ ln -s original.txt nuevo.txt
$ ls -l
total 24
lrwxr-xr-x   1   ivelis admin   12  14  Feb  00:22  nuevo.txt -> original.txt
-rw-r--r--   2   ivelis admin   76  13  Feb  23:14  original.txt
```

Comando mv- Mover o Renombrar

El comando mv es usado para mover un archivo, un grupo de archivos, o un directorio, con todos sus subdirectorios, entre dos directorios. Las páginas man describen éste comando para renombrar archivos, pero es más comúnmente referido como un comando para mover. Este comando tiene dos sintaxis útiles:

```
mv [Opción] <fuente> <destino>
mv [opción] <fuente> <directorio>
```

La primera es la sintaxis de uso para renombrar un archivo:

```
$ ls
original.txt
$ mv original.txt nuevo.txt
$ ls
nuevo.txt
```

El archivo de nombre original.txt ha sido renombrado a nuevo.txt. También podemos mover el archivo a otro directorio así:

```
$ ls
nuevo.txt
$ mv nuevo.txt ~/trabajos/
$ ls
$ cd ~/trabajos/
$ ls
nuevo.txt
```

El archivo nuevo.txt en el directorio actual es movido al directorio /home/usuario/trabajos.*

Comando mkdir- Crear Directorios

El administrador de archivos de GNU/Linux no le da el mismo trato a todos los archivos. Este usa el campo de tipo de archivos en el índice de inodes, o inodo, para determinar qué hacer con el archivo. Si el tipo de inode es especial, ésto quiere decir que, el contenido de los archivos es realmente un puntero de acceso para los manejadores de dispositivos del kernel. Si el tipo de archivo es directorio, entonces el administrador de archivos sabe que el contenido de los archivos es usado para ubicar otros archivos. Concedida esta distinción, el comando mkdir es usado para crear una instancia del tipo de archivo directorio.

Comandos rm y rmdir- Remover Archivos o Directorios

Los comandos rm y rmdir eliminan archivos y directorios respectivamente. El servicio del kernel que el mina un archivo se llama unlink, el cual es más descriptivo de cómo es que el proceso es implementado. Los comandos rm y rmdir deshabilitan las entradas a directorios (un directorio nunca se reduce pero si crece para acomodar más archivos), y ellos reducen la cuenta de los links en el inode. Si la cuenta de link se reduce en número a 0, entonces el bloque de data es retornado a la lista de bloques libre.

El comando tiene una opción especial que le permite eliminar directorios. El comando es:

$ rm -r basura/

Eliminará todos los archivos en el directorio basura además de también el directorio mismo. Si existen subdirectorios dentro del directorio basura, ellos también serán eliminados.

Comandos head y tail- Visores Preliminares de Archivos

Los comandos head y tail ayudan a, rápidamente, ver parte de un log o cualquier otro tipo de archivo de estado. El comando head despliega por defecto las primeras diez líneas de un archivo, y, el comando tail, despliega por defecto las últimas diez. Para ver más líneas, use un argumento numérico. Por ejemplo, para ver las últimas 30 líneas de un archivo log, escriba:

 # tail -30 /var/log/messages | less

En éste ejemplo, las últimas 30 líneas del archivo messages son pasadas al paginador less, el cual despliega el contenido de un archivo una página a la vez. Recuerde que sólo el superusuario puede accesar los archivos log.

Comando file- Para Determinar el Tipo de Archivo

La mayoría de los sistemas operativos que son No-UNiX emplean sistemas de estereotipos de archivos. El mecanismo que usan es el de agregarle extensiones, así que un archivo que es un ejecutable tiene una extensión .exe, los de imagénes pueden ser de extensiones .jpg o .tif, etc . Una vez a los archivos les es dado un tipo, entonces los utilitarios del SO pueden usar la data para llevar a cabo sus tareas, por ejemplo, las base de datos están a la expectativa de archivos de extensiones .dat y entonces las aplicaciones y utilitarios relacionados con la base de datos asumen que éstos son los archivos que contienen su data.

Como GNU/Linux y los otros sistemas operativos Tipo-UNiX no emplean tipos de archivos, entonces no hay manera de un usuario saber con certeza cuál es el tipo de un archivo simplemente por su nombre. Para ésta situación existe el comando file, éste comando ayuda a descubrir, basado en su contenido, el tipo de archivo. Por ejemplo, en éste despliegue mostramos los diferentes tipos de archivos basado en su contenido:

$ file *
RESOLUCION 129-04.pdf: PDF document, versión 1.3

```
PORTG~1.PDF:                    PDF document, versión 1.2
380x.pdf:                       PDF document, versión 1.4
DESODORANTE.xls:                Microsoft Office Document
DIAGRAMA:                       data
Dibujo.psd:                     Adobe Photoshop Image
En la mañana.doc:               Microsoft Office Document
Especificaciones para Linux.sxw: Zip archive data, at least v2.0 to extract
FreeTemplates:                  directory
Guia del Usuario_final:         data
Header.jpg:                     JPEG image data, JFIF standard 1.02, aspect ratio, 100 x 100
Header.png:                     PNG image data, 800 x 104, 8-bit/color RGB, non-interlaced
anunciolinuxIRC.png:            PNG image data, 721 x 326, 8-bit/color RGB, non-interlaced
cartas.sxw:                     Zip archive data, at least v2.0 to extract
logo_Rafael_Vargas.psd:         Adobe Photoshop Image
logo_abierto2.FH9:              Macromedia Freehand 9 Document
vargas_logo.tif:                TIFF image data, big-endian
prueba.sxw:                     Zip archive data, at least v2.0 to extract
```

Comandos df y du- Espacio Libre y en Uso de los Discos

Los comandos du y df son usados con mucha frecuencia para revisar el espacio usado en uno o más directorios, o, para revisar el espacio relativo usado en cada volumen respectivamente. Desafortunadamente, por defecto el comando du reporta en tamaño de bloques lógicos del sistema operativo, o en pedazos de 1024 bytes; mientras que, el comando df reporta en bytes. Las versiones GNU de éstos programas permiten una opción para la lectura de éstos tamaños en forma más leíble al humano. Las salidas de éstos comandos son así:

```
$ du -h    /home/usuario/Desktop/
20          KCurriculum.txt
56K         DIRECTORIO
17M         Documentos
7.8M        FORMULARIO DE INSPECCION.xls
7.8M        Fundamento-Actual
16K         HTMLs
13M         INSTALADORES-LIBRO-FUND
80K         Imágenes/LILO
51M         Imágenes
686M        Libro_KIKO
75M         PDFs
21M         Presentación_Linux
36K         Proyecto de Investigación.doc
60K         cpsld
8.0K        prueba.sxw
4.0K        svgalib-1.4.3.pre.20010602/svgalib-1.4.3
1.1M        tcltu

$ df -h
Filesystem  Size    Used    Avail   Capacity        Mounted on
/dev/hda1   56G     46G     9.2G    83%             /
/dev/hda2   20G     14G     6G      70%             /home
```

Comando tar- Archivador de Cintas

El comando tar (Tape Archiver) es el programa de archivar principal de GNU/Linux. El comando tar combina muchos ficheros en un sólo archivo, también preserva información critica del sistema, como es, quién es el dueño del archivo, los permisos, las rutas de directorios, además de los vínculos simbólicos y hard.

Para crear un archivo del contenido del directorio actual, escriba el siguiente comando:

```
$ tar czf  archivo.tar   *
```

Para ver la tabla del contenido del archivo, escriba el siguiente comando:

```
$ tar  tvf   archivo.tar
```

Para archivar un espacio grande de disco y mover los archivos de una parte del sistema de archivos a otra, utilice el siguiente par de comandos tar :

```
$ tar cf - . | (cd /backup; tar xvf -)
```

Este es un gran ejemplo de lo que es multítarea desde la línea de comandos. La primera instancia del comando tar empieza leyendo el directorio actual (.), archivando los archivos y generando el archivo (Causado por la opción c) a la salida estándar (causado por la combinación de los caracteres f -). La salida estándar entonces es redireccionada a otro programa a través del operador tubería (|). El segundo programa, realmente es la combinación de dos comandos ejecutados en secuencia: el comando cd y la segunda instancia del comando tar. El comando cd nos cambia del directorio actual al directorio backup donde se depositarán los archivos; mientras que, el segundo comando tar, extrae (causado por la opción -x). El archivo leído desde la entrada estándar (causado por la combinación f -), y vuelca los archivos individuales en el directorio llamado backup, en el mismo estado en que los encontró en el directorio original.

Comprimir Archivos con el Comando gzip

El comando gzip reduce el tamaño de los archivos pasados como argumentos usando la codificación Lempel-Ziv (LZ77). Siempre y cuando la operación es exitosa, el archivo es reemplazado con un archivo de extensión .gz, manteniendo siempre los modos de permisos de acceso, y fechas de modificación intactas. Si no especificamos archivos o si usamos el caracter "-", la entrada estándar es comprimida y enviada a la salida estándar, el comando gzip sólo efectúa compresión sobre archivos regulares. En particular ignorará los vínculos simbólicos. Si el nombre del archivo a comprimir es muy largo para el sistema de archivos, gzip lo acortará. El gzip trata de sólo truncar las partes con nombres más largo de tres caracteres: si el nombre del archivo consiste solamente en varias partes pequeñas, las partes más largas serán truncadas, por ejemplo, si los nombres de archivo están limitados a 14 caracteres, manual.gnu.linux.htm será comprimido como man.gnu.lin.htm.gz. Si el sistema no tiene un limite en los nombres de los archivos y directorios, entonces gzip no acorta los nombres.

Por defecto, gzip mantiene el nombre original del archivo y sus estampados de fecha. Estas son usadas cuando descomprimimos con la opción -N. Esta opción es muy útil cuando el nombre del archivo comprimido fué acortado o cuando las etiquetas de las fechas de los archivos no se preservarón después de la transferencia de los archivos. Los archivos comprimidos pueden ser restaurados a su estado original usando a gzip -d, gunzip, o zcat. Si el nombre del archivo almacenado no es válido en el nuevo sistema de archivos, un nuevo nombre es construido de el original para crear uno que sí es válido. El comando gunzip toma como argumento una lista de archivos y los reemplaza con un archivo cuyo nombre termina con .gz, -gz, .z, -z, _z, o .Z y además empiezan con el número mágico correcto (magic number) con un archivo descomprimido sin la extensión original. El comando gunzip también reconoce las extensiones especiales de .tgz y .taz que son abreviaturas de .tar.gz y .tar.Z, respectivamente. Al descomprimir, gzip utiliza la extensión .tgz, si es necesario, en vez de acortar o truncar un archivo con la extensión .tar.

El comando zcat es idéntico al comando gunzip -t. En alguno sistemas el comando zcat puede aparecer como gzcat para preservar el vínculo original al comando compress. El zcat descomprime una lista de archivos pasados desde la línea de comandos como argumento o su entrada estándar y escribe la data descomprimida a la salida estándar, zcat descomprimirá los archivos con el número mágico correcto tengan o no la

extensión .gz.

HERRAMIENTAS PODEROSAS

En esta sección se ofrece una breve descripción de las características de las herramientas más poderosas del ambiente de trabajo de GNU/Linux. Tomaremos en consideración los siguientes tópicos:

• **diff**	Inventario de Cambios Efectuados a un Archivo
• **find**	Busca y Manipula Archivos
• **grep**	Busca Cadenas en el Contenido de Archivos de Texto
• **Expresiones Regulares**	Uso de Abreviaturas para Asistir los Comandos y Utilitarios
• **sed**	Edición de Flujo de Texto (Stream)
• **awk**	Edición Avanzada de Flujo de Texto
• **Perl**	Lenguaje de Programación de Extracción y Reporte

Aquí le damos una brevísima introducción al programa Perl, el cual es el lenguaje defacto de los administradores de sistemas y de páginas web.

Comando diff- Inventario de Cambios a los Archivos

El comando diff toma como parámetros el nombre de 2 archivos y da salida a las diferencias entre los 2 archivos. Cuando es ejecutado sobre un archivo ordinario, como se muestra más adelante, la salida es un poco compleja y hace referencia a comandos de edición utilizados por el ed, la cual puede ser ejecutada para sincronizar los 2 archivos . En el ejemplo mostramos la diferencia en el archivo de contraseñas actual y una copia que sirve de backup. En éstos archivos la línea 23 ha sido cambiada y la línea 24 ha sido agregada al archivo /etc/passwd

```
$ diff passwd passwd.bak
23c23,24
< unknown:*:99:99:Unknown User:/var/empty:/usr/bin/false
—-
> DIFERENTE
> LINEA ADICIONAL
```

Por lo general la salida sólo contiene comandos simple del siguiente formato:

n1 a n3,n4	**Las líneas n3 a n4 agregada al archivo 2 después de la línea n1 en el archivo1**
n1,n2 d n3	**Líneas n1 a n2 eliminadas del archivo 1 (estaban en la línea n3 en el archivo 2)**
n1,n2 c n3,n4	**Línea n1 a n2 en el archivo 1 cambiadas a líneas n3 a n4 en el archivo 2**

Después de cada línea de comandos, las líneas afectadas del primer archivo son listadas precedidas por un <, y cualquier línea nueva en el archivo 2 es listada y precedida por el símbolo >.

El comando diff también puede ser utilizado sobre directorios. Aquí la salida del comando sólo muestra mensajes de “only in ...” (solamente en ...) y “common subdirectories ...” (Subdirectorios comunes ...) y no al estilo de instrucciones parecidas al ed.

```
$ diff dir dir2/
Only in dir: archivo1
Only in dir: passwd.bak
```

Comando find - Busca y Manipula Archivos

Find es un comando que busca en el árbol de directorio y ejecuta comandos sobre el resultado. Este buscará en un directorio y todos sus subdirectorios por archivos basados en un criterio de selección y opcionalmente ejecutaría cualquier comando o script del shell sobre cada archivo que iguale el criterio de

búsqueda.

La sintaxis del comando find es:

$ find directorios opción acción

Las opciones son en forma de palabras reservadas, por ejemplo:

-name nombre	**Busca archivos llamados nombre**
-user nombre	**Busca archivos adueñados por el usuario nombre**
-type [fdlcb]	**Busca archivos del tipo dado (por ejemplo, d es un directorio, l es un vínculo)**
-size [+/-]n[ck]	**Busca archivos de un tamaño dado (ejemplo,+10k significar más grande que 10 Kb)**
-inum número	**Busca archivos con el número de inodo dado (hard link)**

Las opciones del comando find facilitan la búsqueda de archivos, de acuerdo a cualquier tipo de atributo de éstos. Una vez el archivo ha sido localizado, éste puede ser sometido, como un argumento, a cualquier comando de GNU/Linux o script del shell disponible.

La Opción -name soporta los mismos caracteres wildcard que esos del shell (*, ?, y []). Recuerde colocar un nombre que contenga éstos caracteres dentro de comillas dobles para evitar que el shell efectue su propia generación de nombres de archivos y sustitución de argumentos.

Muchas otras opciones del comando find soportan criterios de búsqueda muy poderosos y son descritos en sus páginas man. En realidad, el comando find no es usado para buscar archivos, sino para navegar el sistema de archivos buscando los archivos que igualan cierto criterio de búsqueda y luego ejecutar comandos sobre los resultados. Un uso común es, por ejemplo, buscar todos los archivos que su tiempo de acceso es anterior a cierta fecha y luego moverlos a cierta localidad.

Las acciones pueden ser:

-print	**Imprime los nombres de archivos a la salida estándar**
-exec comando {} \;	**Ejecuta el comando dado por cada archivo encontrado**
-ok comando {} \;	**Ejecuta el comando dado pero pide confirmación**

Ejercicio 7-6: Uso de find

En éste ejercicio le presentamos ejemplos del comando find. Las soluciones a éste ejercicio se encuentran en el Apéndice A.

¿Qué efectúan los siguientes comandos find?

1. $ find . -print
2. $ find . -type d -print
3. $ find /home -name .profile -print
4. $ find /home -name .bash_profile -print
5. $ find . /tmp /usr/tpm -name core -exec rm {} \;
6. $ find . -name “*.o” -ok rm {} \;
7. $ find / -type f -size +1k -print >/tmp/grandes 2>/dev/null &

Comando grep- Buscar en Archivos de Texto

El comando grep es usado para buscar en el contenido de archivos de texto. Sus siglas significan (global regular expression print). El texto usado para especificar un patrón a grep es llamado una expresión regular. Estos caracteres pueden ser normales, alfanuméricos o pueden ser caracteres especiales, para así poder igualar varios patrones de texto. Brevemente cubriremos más sobre expresiones regulares. El comando grep imprime todas las líneas que igualan el patrón especificado.

Todo usuario que desea adquirir niveles altos de eficiencia en GNU/Linux debe dominar el comando grep.

No podrá dominar todas sus opciones y características en una sola y única sesión, deberá dedicarle varias horas de práctica para dominar tan importante herramienta. A medida que use el comando grep y estudie sus páginas man y un buen tutorial, empezará a dominarlo y sentir lo poderoso que es realmente. Usted puede usar el comando grep como un filtro especificándole solamente el patrón que usted desea encontrar, o usted puede especificar el patrón y un archivo o un conjunto de archivos, para llevar a cabo la búsqueda. Aquí le mostramos una lista de algunas de las opciones más usadas con el comando grep:

-v Imprime las líneas que no igualan en vez de las que sí lo hacen.
-c Imprime el número de líneas que igualan en vez de las líneas mismas
-i Indiferente a mayúsculas y minúsculas
-n Precede cada línea de salida con su número correspondiente

Aquí le mostramos un ejemplo usando grep conectado con una tuberia:

```
$ who | grep ttyp | cut -d' ' -f1
ivelis
root
```

En éste ejemplo tomamos la salida del comando who y filtramos las líneas que no contienen el texto ttyp, en pocas palabras, imprimimos las líneas que sí contienen la cadena de caracteres ttyp. Entonces emviamos la salida al comando cut para imprimir solamente el primer campo separado por espacio, que es el campo de los nombres de usuario. Note que el orden de los comandos es de extrema importancia; necesitamos efectuar la búsqueda antes de cortar los campos, ya que eliminaríamos el campo que contiene la cadena "ttyp" que grep necesita para elegir los campos correctos.

Las Expresiones Regulares

GNU/Linux define un mecanismo común de reconocimiento usando expresiones regulares. Las librerías del sistema de GNU/Linux incluyen soporte para las expresiones regulares, lo cual anima a los programadores a crear programas que necesitan usar reconocimiento de patrón, a utilizar expresiones regulares.

Desafortunadamente, el sistema de reconocimiento de caracteres especiales del shell no utiliza expresiones regulares, ya que éstas son más difíciles de usar que las formas abreviadas del shell. Ambos, los caracteres especiales del shell y los de las expresiones regulares, son muy similares, pero los usuarios de GNU/Linux deben apreciar la diferencia de utilizar las expresiones regulares apropiadamente.

Como las expresiones regulares utilizan algunos de los caracteres especiales del shell, todas las expresiones regulares deben estar encerradas entre comillas simples. Las expresiones regulares son descritas en las páginas man del editor ed (simplemente escriba "man ed").

Los Caracteres de las Expresiones Regulares

Las expresiones regulares pueden contener cierto tipo de caracteres especiales para igualar patrones: un punto igualará cualquier caracter único, y es equivalente al símbolo de pregunta (?) del shell. Un punto seguido de un asterisco igualará cero o más ocurrencias de cualquier caracter y su equivalente en el shell es el asterisco. El uso de las llaves cuadradas ([]) es idéntico al del shell, excepto, que el primer caracter de ^ y no el de ! iguala cualquier caracter que no esté en la lista.

Metacaracteres que igualan expresiones incluyen:

. Iguala cualquier caracter
[lista] Iguala cualquier caracter ´único en la lista
[rango] Iguala cualquier caracter único en el rango.

[^rango]	Iguala un caracter único que no éste en el rango de la lista

Cuantificadores usados con los metacaracteres anteriores incluyen:

.	Iguala el caracter anterior 0 o más veces
\{n\}	Iguala el caracter anterior n número de veces
\{n,\}	Iguala al caracter anterior por lo menos n líneas
\{n,m\}	Iguala al caracter entre los números de la n a la m

El control de los metacaracteres incluye:

.	Iguala regexp al principio de la línea solamente
$	Iguala regexp al final de la línea
\	Bloqueo de Reconocimiento de caracteres especiales

Los caracteres de expresiones regulares son usados para marcar el inicio y fin de las líneas y para sopo tar contadas ocurrencias de cadenas. Los caracteres no-especiales funcionan aislados y los caracteres especiales pueden ser escapados usando la barra invertida (backslash \). Algunos ejemplos son:

Ayuda	Iguala cualquier línea que contenga la cadena Ayuda
\..s	Iguala cualquier línea con un punto de penúltimo caracter
^.*$	Iguala todas las líneas
^...$	Iguala cualquier línea con exactamente 3 caracteres
^[0-9]\{3\}[^0-9]	Iguala cualquier línea que empiece con 3 dígitos y seguida por un no-dígito
^\([A-Z][A-Z]\)*$	Iguala cualquier línea que sólo contiene un número par de letras mayúsculas
\(-[a-zA-Z] \)*	Iguala un grupo de opciones de línea de comandos, ej. es, 1 guión, 1 letra, entonces 1 espacio, repetido

Buscar Dentro de vi

Los comandos de búsqueda en vi son los caracteres (/ y ?) y usarán expresiones regulares para formular sus criterios de búsqueda, a menos, que la opción set nomagic sea definida. El mismo método de búsqueda también puede ser usado dentro del mecanismo facilitado por el utilitario history. En cualquier momento presione la tecla ESC, y entonces escriba '/cat', y el último comando que utilizo 'cat' será llamado.

Buscar Dentro de more y less

Los paginadores more y less permiten búsquedas internas. El alcance y el poderío de la implementación de las expresiones regulares puede variar de un sistema a otro, pero la capacidad de búsqueda siempre esta presente. El comando more acepta expresiones regulares al igual que si fuese en el vi. El comando more siempre desplegará unas cuantas líneas adicionales por encima después de la que coincidió con el patrón de búsqueda. Podemos incluir patrones de búsqueda que incluyan expresiones regulares desde la línea de comandos, damos un ejemplo:

```
$ more +'/^Jose' estudiantes.txt
Jose Paredes,19750726,Interior,27,Herrera,1975,Sistema,Programacion,Base de datos
```

El Editor sed- Editor de Flujo de Texto

El comando sed (El editor de flujo de textos, stream editor) es similar al editor ed pero fué diseñado como un filtro en una tubería y no como un editor de línea. Los comandos son normalmente especificados en la línea de comandos, pero pueden ser redireccionados a un archivo. Esto puede ser utilizado para editar varios archivos en un sólo comando.

Los comandos consisten en un sintaxis parecido al del ed (Números de líneas o patrones) y comandos de caracteres únicos, la mayoría que son idénticos a los comandos ed. Si no se dirigió a un sitio específico el comando aplica a toda la línea.

El comando sed no modifica los archivos en los cuales el opera, para usted salvar sus cambios, la salida de sed debe de ser redireccionada a un archivo por separado. Usted no puede redireccionar la salida al mismo archivo que usted esta redireccionando ya que no puede leer y escribir al mismo archivo en un mismo tiempo.

Usando sed

Cuando use sed, tome en cuanta las siguientes consideraciones:

- La sintaxis simple de sed desde la línea de comando es:
  $ sed [-n] 'comando' [archivos...]

- Los comandos de edición de sed pueden incluir una dirección de línea
- Las operaciones pueden ser limitadas a sólo ciertas líneas
- La dirección es especificada inmediatamente antes del comando
- La dirección puede ser específica como un número o como una expresión regular, por ejemplo:

n	**Procese la línea número n**
n,m	**Procese las líneas entre los números n y m**
n,$	**PRocese entre la línea número n y la última línea**
/regexp/	**Procese entre las líneas que igualan la expresión regular**

- Ejemplos simples de sed usando la opción q (q es el comando de quit, salida de sed):
  $ sed '21q' estudiantes.txt
  Lista las primeras 21 líneas del archivo estudiantes.txt luego sale

  $ sed '/^Francis/q' estudiantes.txt
  Pasando líneas a la salida estándar; luego sale con la primera línea que empiece con Francis. La opción -e debe ser pasada al comando sed con cada comando de edición si se especifica más de un comando. La opción -n apaga el mecanismo automático de impresión a la pantalla de cada línea de entrada.

Si se requieren varios comando de edición, la línea puede prolongarse y convertirse en inmanejable. En éste caso usted puede elegir colocar los comando de edición en un archivo y someter luego el archivos al comando sed con la opción -f.

Buscar con sed

Algunas aplicaciones comunes del comando sed son:

- Simple búsqueda de un patrón o expresión regular, el comando p:
  $ sed -n '/Cesar/p' estudiantes.txt
- La función de búsqueda y remplazo el comando s:
  n,ms/anterior/nuevo/flags
  Donde "n,m" definen el rango de líneas que el comando de sustituir va a operar

  Ejemplo: Buscar y remplazar globalmente y salvar:
  $ sed '1,10s/windows/linux/g' estudiantes.txt > estudiantes.mejorados.txt
  Busque en las líneas del 1 al 10 y reemplaza cada ocurrencia de windows con linux en el archivo estudiantes.txt y guarda un nuevo archivo de nombre estudiantes.mejorados.txt.

En el primer ejemplo, utilizamos la opción -n para detener a sed de imprimir todas las líneas en la pantalla. usamos el comando p, el cual efectúa lo mismo pero sólo en aquellas líneas que igualan la expresión regu-

lar. en éste ejemplo en particular es un equivalente directo al comando grep.

En el segundo ejemplo mostramos un rango numérico de dirección de líneas. Sólo las líneas del 1 al 10 serán afectadas. La opción g inmediatamente después de la cadena de búsqueda y sustitución, significa llevarlo a cabo globalmente en cada ocurrencia de la cadena de texto buscada.

Podemos tomar de éste uso de expresiones regulares para decir que puede ser usado para generar ambos elementos de rango para especificar direcciones de líneas. Entonces, esta línea tomaría la siguiente forma:

$ sed -n '/regexp/, /regexp/ s /regexp/nuevo/' archivo.txt

Pruebe la siguiente sentencia:

$ sed -n '/^Francis/, /^Jose/s/,.*$//p' estudiantes.txt

Aquí les presentamos algunos ejemplos adicionales usando sed:

$ sed '/^FIN$/q' archivo.txt

Lista todas las líneas incluyendo las que contienen la cadena FIN.

$ sed 's/Windows/Linux/g' estudiantes.txt >archivo.txt

Lista todas las líneas; sustituyendo toda ocurrencia de Windows por Linux y redirecciona la salida a un archivo de nombre archivo.txt.

$ SCR=$(tty | sed 's,^.*/,,')

Asigne la salida de un comando compuesto a la variable SCR. La salida del comando tty, comando que imprime la ruta absoluta y completa del terminal, ésta salida se le pasa a sed; sed entonces busca cualquier número de caracteres seguido por una barra '/' (note que ésta expresión regular busca esta barra), y substituye éste patrón con nada (por eso es el */ todo hasta la barra y ,, es nada). El efecto de esta sentencia es que el comando sed recortará todo hasta, e incluyendo, la última barra.

Note que se utilizó comas para separar a la cadena a encontrar de la cadena a reemplazar, en vez de la tradicional barra (s/^.*///), porque, como es obvio, aquí, al buscar la barra, y tener que separar por barra, se tornaría un poco confuso. Podemos elegir cualquier caracter único en esta situación, no sólo la coma, para un buen ejemplo pruebe con ! o ?.

Si aún tiene duda de como la última barra será localizada, compare las dos condiciones de búsqueda: primero la que nosotros utilizamos; segundo, la que nos asegura que la primera barra será encontrada.

- Cualquier número de caracteres seguido por una barra
- Cualquier número de caracteres que no sea una barra seguido por una barra

Editor awk- Editor Avanzado de Flujo de Texto

El comando awk es un lenguaje de programación interpretado completo que puede ser usado para filtrar y manipular texto, el uso primario de awk es procesar archivos de texto preformateados, y, muy a menudo, es usado en conjunto con el comando sed.

Al igual que sed, el utilitario awk, es una herramienta para procesar flujos de texto, pero, el awk puede también ser visto y usado como un lenguaje de programación scripting, y puede programar funciones que lleven a cabo tareas verdaderamente complejas de manipulación y generación de reportes basados en data contenida en archivos de texto y salida estándar de otros comandos.

La sintaxis de awk es:

$ awk [-F caracter] [-f Archivo | programa] [Archivos...]

Aquí presentamos algunos ejemplos:

- Para cortar el nombre de usuario y el nombre completo del campo nombre y colocarlo en orden inverso del archivo /etc/passwd:
  $ awk -F: '{print $5,$1}' /etc/passwd

- Para reportar la suma del cuarto campo correspondiente a la edad del archivo de texto estudiantes.txt:
  $ awk -F, '/Sistema/{ suma = suma + $2 } > END { print "La Suma de todas las edades de los estudiantes de sistemas es:", suma }' estudiantes.txt

Los comandos pueden ser especificados desde la línea de comandos o desde un archivo (usando la opción -f y el nombre del archivo que contiene los comandos a ejecutar).

Los comandos consisten de un direccionamiento parecido al del editor ed (números de línea y patrones de caracteres) y llaves especificas de comandos. Hay dos patrones especiales que son (BEGIN y END) ejecutados antes y después de los bloques principales de comandos respectivamente. Si no especificamos direccionamiento de líneas, entonces la sentencia será aplicada a todo el archivo.

Note que awk reconoce campos y records. Cuando el símbolo de $ es usado dentro de awk, éste se refiere al número de un campo dentro de la línea o récord siendo procesado. Nunca lo confunda con las variables del shell, que son precedidas con con un símbolo igual de $.

En el siguiente ejemplo, contamos el número de veces que cada palabra completa aparece en un archivo específico de data:

```
awk 'BEGIN{print "Vamos a Contar Palabras"}
{for (i=1;i<=NF;i++) palabras[$i] += 1}
END{for (p in palabras) print p ":" palabras[p]}'
```

En éste ejemplo mostramos algunas de las características de awk como un lenguaje de filtro programable. La página man describe el resto de las característica del comando awk. Primero salvamos éste texto en un archivo y lo llamamos script.awk, luego lo hacemos ejecutable (chmod a+x script.awk) y finalmente pasamos parte del archivo (que es el contenido de éste mismo párrafo) como entrada a nuestro script.

```
$ cat archivo.txt |./script.awk
comando:1
de:5
ejemplo:1
La:1
éste:3
texto:1
...
```

AWK son las siglas de sus tres autores: Aho, Weinberger, y Kernighan, los autores del tiempo inicial de UNiX en la compañía AT&T. El AWK ha sido mejorado desde aquellos tiempos, y la gran mayoría de los sistemas implementan realmente NAWK, que significa el nuevo AWK, y, en GNU/Linux, ejecutamos realmente la implementación del GNU, GAWK. Muy a menudo, los sistemas crean vínculos del tipo hard link a awk, a uno de éstos utilitarios, si algún día descubre, que sus scripts, que antes le funcionaban muy bien, de repente no le funcionan, lo más probable es que usted usaba nawk, y éste sistema actual está ejecutando el viejo awk.

Es esencial cierto nivel de manejo de awk para continuar su carrera como un administrador de sistemas de tipo UNiX, ya que, sistemas que ejecutan UNiX a gran escala, y por mucho tiempo, lo más seguro es que contienen scripts de awk para automatizar sus procesos, scripts que han sido modificados por años. Estos

scripts pueden ser extremadamente complejos y difícil de leer, debido, a que, la estructura de awk es extraña y críptica. La herramienta que reemplazó a awk para estas tareas de automatización de procesos fué Perl, la cual mencionamos brevemente en la próxima sesión.

Scripts en Perl

Perl es un utilitario del Open Source, de implementación gratuita, aunque su licencia no es GPL, es una licencia flexible para lo que es desarrollo de aplicaciones (Licencia Artística). Perl es muy popular en uso, y existe mucha ayuda en todo el internet. Es una herramienta del sistema muy útil para desarrollar y mantener aplicaciones de servidores web, mejor conocidos como CGI scripts. Perl reúne todas las funciones de sed y awk en una sóla herramienta. Perl es una solución completa como lenguaje de programación, ésta posee la velocidad, capacidad, y niveles de seguridad para tareas de procesamiento de alto volumen. La mayoría de los programadores de scripting de UNiX, proclaman que Perl debe ser el reemplazo completo de sed y awk. Esto se torna difícil, puesto que, la naturaleza de UNiX, y, en especial, el Open Source y el Free SoftWare, garantizan el contínuo desarrollo de cada utilitario; sed y awk son buenas herramientas, y siempre están mejorando.

Perl nació para proveer una herramienta que puede combinar las funciones y además expandir en las capacidades, y provee estructuras y capacidades adicionales a las de awk y sed. Perl también es menos críptico y fácil de leer que awk y sed, y, no compromete su capacidad. Perl ha crecido al punto de defnir la industria del CGI y es bien soportado sobre muchas plataformas, no sólo las del mundo UNiX.

Las siglas PERL significan: Lenguaje Práctico de Extracción y Reporte (Practical Extraction and Report Language). A diferencia de sed y awk, no es una herramienta estándar de UNiX, pero es muy difícil encontrar hoy día un sistema UNiX moderno que no lo incluya. Perl tiene una gran comunidad de usuarios, y, es difícil que pueda pensar en una tarea relacionada a la automatización de servidores para la que no exista una solución de un script de Perl que pueda descargar desde el internet y modificarlo para sus necesidades. Perl también es de plataforma cruzada, una gran ventaja a considerar para un ambiente heterogéneo. también, podemos resaltar, que Perl es uno de los lenguajes más cotizados en el momento de búsqueda de empleo en los Estado Unidos. Todo el que usa GNU/Linux y UNiX debe aprender por lo menos un poco de Perl, ya que, ésta aparecerá tarde o temprano en sus tareas administrativas.

Para empezar a aprender diríjase al sitio web de Perl, visite http://www.perl.org. Existen muchos buenos manuales, tutoriales e información gratuita y además libre como es común en todos los productos de nuestra comunidad.

Ejercicios 7-7: El Uso de Expresiones Regulares con grep

Las soluciones a éste ejercicio aparecen en el Apéndice A, Considere las siguientes preguntas.

1. ¿Qué efectúa el siguiente comando?

```
$ ls -l | grep '^d'
$ grep '^user[0-9]' /etc/passwd
$ grep '^[A-Za-z]*[0-9]$' archivo.txt
$ Is -a | grep '^\.[^.]'
$ grep '^.*,[0-9]\{10,\}' archivo.txt
```

2. ¿Qué buscamos en éste ejercicio?

```
$ grep '^.*.[0-9]\{1,\}' estudiantes.txt
Jose Paredes,19750726,Interior,27,Herrera,1975,Sistema,Programacion,Base de datos
Francis Francis,19750727,Interior,27,Rosal,1974,Sistema,Programacion,Base de datos
```

Ejercicio 7-8: Uso de Expresiones Regulares para Buscar en vi

Las soluciones para éste ejercicio están en el Apéndice A.

1. ¿Qué línea puede ser localizada con el siguiente?
   /^{
   /;$
   /^TERM
   ?^p[<tab><espacio>]*TERM=
   /$'\$

Ejercicio 7-9: Uso Avanzado de Expresiones Regulares

En éste ejercicio investigamos el uso más avanzado de utilitarios de GNU/Linux y las expresiones regulares. Las soluciones para éste ejercicio están en el Apéndice A.

1. En esta parte del ejercicio pondremos en práctica varios aspectos de la manipulación de nombres de archivo, el comportamiento del shell al momento de leer las sentencias que le escribimos, utilizar opciones en la línea de comandos del shell para determinar causas de problemas y escoger la herramienta correcta para efectuar la tarea.
   Nuestra tarea es encontrar, en forma recursiva, todos los archivos que se encuentran debajo de la jerarquía de /usr/bin, que su nombre empieza con la letra "p".
   Prueba lo siguiente:
   $ find /usr/bin -name p* -print
   ¿Puede explicar, por qué no parece estar funcionando como esperábamos?
   Use el shell para ayudarle a diagnosticar el problema; establezca la opción de rastrear en su shell actual con el siguiente comando:
   $ set -xtrace
   Repita el comando find anterior. Ahora el shell debe darle una idea de qué ocurrió mal, éste es un problema, lo más seguro, de evaluación de caracteres wildcard.
   Repare su comando find para que le arroje los resultados esperados.
   ¿Puede usted efectuar una sentencia del comando ls que arroje un resultado similar?
   Pruébelo!, entonces compare los resultados del siguiente comando find, claro ahora corregido y su sentencia ls:
   $ find /usr/bin -name "p*" -print
   $ ls -R /usr/bin/p*
   ¿Son las salidas iguales? ¿Por qué no? Estas son preguntas fundamentales de su entendimiento del comportamiento de la sentencia ls -R y el comando find, ¿Cuál de los dos comandos nos ofrece respuestas más precisas a la tarea original?

2. En ésta pregunta de éste ejercicio ponemos a prueba su manejo del comando find. Encuentre y liste todos los archivos en su directorio home.
   Repita el último comando, pero, redireccione la salida a un archivo y redireccione además, cualquier mensaje de error, al dispositivo null.
   Modifique su última sentencia en la línea de comandos para que liste solamente los archivos que han sido modificados hoy y déle salida a un archivo diferente. Claro está, ésto demanda una nueva opción del comando find; use las páginas man para encontrarla.

3. En éste paso experimentaremos con el uso de expresiones regulares.
   Primero debemos preparar el ambiente. Ejecute los siguientes comandos:
   $ cd /etc
   $ ls -a > ~/datos.txt

$ cd

Este conjunto de comandos le almacenará una lista de los archivos en el directorio /etc en un archivo datos.txt en su directorio home.

Para lo que queda de éste ejercicio, necesitará usar el comando en el siguiente formato:

grep ‘ regexp‘ datos.txt

Donde el patrón ‘regexp’ será apropiado para el tipo de búsqueda que le pediremos llevar a cabo.

Escriba expresiones regulares que encontrarían todas las líneas en el archivo de data que:

- Empiezan con la letra “p”.
- Terminan con la letra “y”.
- Empiezan con una “m” y terminan con una “d”.
- Empiezan con una “e”, “g”, o “m”.
- Contienen una “o” seguida (no necesariamente de inmediato) por una “u”.
- Contienen una “o”, entonces cualquier caracter único y entonces “u”.
- Empiezan con una letra minúscula.
- Contienen un dígito.
- Empiezan con una “s” y contienen una “n”.
- Contienen exactamente 4 caracteres.
- Contienen exactamente 4 caracteres, pero, ninguno de ellos es un “.”.

4. Dé solución al siguiente problema usando sed:
   - Use el comando sed para eliminar todo, menos el nombre del usuario, de la salida del comando who.
   - Use el archivo estudiantes.txt como archivo fuente, elimine todo excepto, los nombres de las carreras.
   - Usando el archivo estudiantes.txt como fuente, imprima toda la data de los estudiantes que estudian sistemas.
   - Lo mismo que el anterior, pero, sólo imprima el nombre de los estudiantes. Use sólo un comando sed.
   - Igual que el anterior, pero, agréguele la cadena: “Estudiantes de:” antes de cada línea o estudiante. Puede usar más de un comando sed.

Ejercicios 7-10: Uso Adicional de Herramientas Poderosas

Las soluciones a éstos ejercicios se encuentran en el Apéndice A.

1. Pruebe los siguientes comandos:

   $ REV=$(tput rev)
   $ nrm=$(tput rmso)
   $ echo ${REV}Reverse${NRM}Normal

   Estos comandos usan a tput para interrogar en la base de datos de información del terminal (terminfo) para recibir la secuencia requerida de la pantalla, para cambiarla en vídeo inverso y normal. Las secuencias han sido guardadas en dos variables: REV y NRM, las cuales pueden ser usadas para resaltar texto en las salidas. Usando estas variables, resalte la fecha actual en la salida del comando cal. Necesitará usar la salida del comando date para recibir el día actual del mes.

2. Use sed para traducir múltiple espacios a sólo una coma en la salida del comando ls -1.

3. Busque todos los usuarios del sistema que usan bash o tcsh.

4. Use una combinación de los comandos tty y sed para definir una variable y llamela TTY, la cual contiene sólo el nombre de login de su dispositivo.

RESUMEN

En este capítulo fue introducido algunos conceptos de firewalls y VPNs, incluyendo:

- Un firewall es un componente critico de su política de seguridad, principalmente porque es donde podemos forzar la autentificación de todos los usuarios y monitorear todo tráfico entrante y saliente.
- Muchas implemteciones requeriran múltiples firewalls para poder manejar los diferentes niveles de seguridad de la red.
- Los firewalls pueden funcionar de diferentes maneras, incluyendo filtrado de paquetes, gateways del nivel de circuito y del nivel de aplicación.
- Para poder diseñar un firewall se requiere de conocimiento sólido de TCP, UDP y ICMP y los servicios que utilizan estos protocolos IP.
- Un VPN requiere de tres componentes.
  - Autentificación
  - Tunneling
  - Encriptación
- El Stunnel es un utilitario de tunel seguro muy utilizado en GNU/Linux.

PREGUNTAS POST - EXAMEN

Las respuestas a estas preguntas están al final de este libro en el Apéndice A

1.- ¿Cúal es el aspecto más importante de la colocación del firewall?

2.- ¿Cómo funciona el fortalecimiento del sistema operativo?

3.- ¿Qué es un screening router?

4. ¿Cúales son las dos configuraciones que puede tener un firewall por defecto?

5.- ¿Qué es un filtrado de paquetes?

6.- ¿Cúales son los dos principios básicos son criticos en el diseño de firewalls?

7.- ¿Cúales pasos son importantes en la creación de un plan de contingencia para su sistema de firewall?

8.- Los enrutadores de filtrado de paquetes proveen una buena y económica protección. Pero, si esta es la única seguridad implementada, ¿cúales fueran algunas desventas, si existen?

9.- Defina que es un VPN.

CAPITULO 8

ADMINISTRACIÓN DEL SISTEMA OPERATIVO GNU/LINUX

PROCESOS Y SCRIPTING DEL SHELLS

OBJETIVOS

Al completar este Capítulo, usted podrá:

- Uso de flujos de texto UNiX, tuberías, y redirección.
- Procesar flujos de texto utilizando filtros
- Comparar y contrastar variables de ambiente versus
- Escribir y personalizar scripts simples del shell.

PREGUNTAS PRE-EXAMEN

Las repuestas se encuentran en el Apéndice A.

1. ¿Cuáles son los shells más comunes usados en GNU/Linux?

2. ¿Cómo puede usted lograr que el comando ls escriba su salida a archivo.txt?

3. ¿Cuál es la diferencia entre un script del shell y un comando regular?

4. ¿Qué es un filtro?

INTRODUCCION

Una de las características más importante de UNiX heredada por GNU/Linux, es su flexibilidad en el shell desde la línea de comandos. GNU/Linux continua en la tradición de UNiX, usando los shells y utilitarios GNU. De hecho, los utilitarios y shells del GNU, son por lo general, más fáciles de usar y proveen una mayor funcionalidad que los utilitarios tradicionales de UNiX.

En éste capítulo, cubriremos por completo los shells y su manejo desde la línea de comandos, redireccionado de las salidas y entradas estándar, las tuberías, los filtros y los scripts de shell.

El Shell

A simple vista un shell es un interprete de comandos. Normalmente es ejecutado en un modo interactivo donde el usuario escribe comandos, y el shell provee salidas de texto a esos comandos. La mayoría de los sistemas operativos poseen shells; hasta el MS-DOS provee un shell simple de nombre command.com. En ocasiones el shell también es llamado el Interprete de la Línea de Comandos (CLI).

La terminología shell, se deriva de la cáscara de la nuez. El centro de esta, la semilla, si nos podemos imaginar, es el kernel, y lo que la rodea, la cáscara, es el shell. Continuando con la comparación, el centro de sistema operativo se llama el kernel y en éste caso el centro de GNU/Linux es su kernel, Linux. Este núcleo provee los servicios más básicos de un sistema operativo, acceso a disco, almacenamiento de archivos, estructura de directorios, acceso a redes, etc.. El usuario nunca tiene que interactuar con esta capa, ya que en esta cualquier simple operación conllevaría muchísimos comandos, pero, el usuario, es proveído de muchos utilitarios para usarlos como herramientas. Alrededor de éstos utilitarios se encuentra la capa que se denomina como el shell.

La verdad es que el shell es mucho más que un simple interprete de comandos, que reconoce cada comandos digitado y lo ejecuta. Los Shells entienden un lenguaje de programación especial. Algunos de éstos lenguajes son muy primitivos y aún utilizan la estructura de goto. Otros shells manejan lenguajes de alto nivel que ofrecen estructuras que se comparan con lenguajes tan complejos como las del lenguaje C, Perl, Python, etc..

En esta sección, empezaremos a aprender el manejo de los shells y cubriremos los siguientes tópicos:

- Los Shells que están disponibles en GNU/Linux
- Razones para Usar el Bash

Shells Disponibles en GNU/Linux

En el sistema operativo GNU/Linux, el shell es un simple programa que le líneas desde el terminal de entrada y ejecuta los comandos. Usted puede escribir su propio shell si así usted lo desea; muchas personas han logrado éste objetivo.

El shell original de UNiX, y que aún continúa disponible, es Bourne shell, escrito por Stephen Bourne de los Laboratorios Bell de AT&T. El Bash del GNU (Bourne-Again Shell), disponible través de la FSF (Free Software foundation), es totalmente compatible con los antecedentes de Bourne Shell y es el por defecto en todas las distribuciones GNU/Linux.

Bill Joy de la Universidad de California en Berkeley y de SUN MicroSystems, escribió un shell más complejo con una sintaxis semejante a la del lenguaje C, denominado C-shell. Desafortunadamente, no es compatible con las anteriores versiones del Bourne Shell. De todas formas ha tenido una acogida razonable en la comunidad de Unix y aún se distribuye hoy en día. La versión actual del C Shell se llama TCSH (extended C shell), estadísticas demuestran que posiblemente es el segundo más utilizado en GNU/Linux. Poseé completado de nombres de archivos y comandos, capacidad de edición de comandos en línea, en algunas área

hasta más avanzadas que las de bash. Es usada, en la mayor parte, de manera interactiva, mientras que el Bourne shell (bash), es la preferida para scripting hasta por los usuarios del C-shell.

Otro shell que se derivó del Bourne shell es el shell Korn, el cual fué originalmente distribuido con el UNiX de AT&T. La versión original fué comercial, como otros shells, pero, ahora está disponible en una versión libre: PDKSH. Una de las diferencias del Korn es que no es compatible con la versión original del Bourne shell pero sí es bien popular entre sus usuarios. Como los shells bash y C shell, hace énfasis en el proceso desde la línea de comandos; así, pues, es casi seguro que los usuarios que usan Korn desde la línea de comandos, muy a menudo usarían Bash o Perl para sus tareas de scripting. Casi todas las distribuciones de GNU/Linux de hoy día vienen con bash y Korn, aunque la popularidad de bash es astronómicamente superior a la popularidad del Korn shell.

Razones para Usar el Bash

Hay muchas razones para usar el GNU bash:

- Un gran número de estructuras de lenguaje de alto nivel (if, while, select, etc.)
- Soporte de examen de archivos y directorios
- Soporte de aritmética de enteros
- Desarrollo de programas más rápido que comparado con lenguajes de como C
- Mecanismo de historial interactivo de comandos que emula a uno de los dos editores más usados de GNU/Linux
- Soporte de funciones
- Estructuras avanzadas para igualar patrones y la habilidad de procesar cadenas de caracteres complejas
- Altamente disponible en todas las distros de GNU/Linux, UNiX, y muchos otros sistemas operativos
- Es el shell GNU/Linux por defecto.

El shell interpreta los scripts del shell, así que no existen las fases de compilación ni de link (vincular). Existe la disponibilidad de estructuras y también capacidades de comparativas avanzadas de patrones de caracteres, maneras de examinar archivos y directorios, aritmética de enteros, y soporte de funciones; un mecanismo que permite revisar el historial de comandos para poder ser editado en vi o emacs, los dos editores de texto más populares de GNU/Linux y Unix.

También existe el soporte de funciones. Esto quiere decir que sus códigos, que son frecuentemente usados, pueden ser almacenados en archivos de funciones y desde sus scripts hacer llamados a ellas y así poder efectuar scripts mucho más pequeños, lo que ayuda a confeccionar scripts mucho más fácil de leer y mantener.

Entrada y Salida de los Comandos

En esta sección se introducen las ideas fundamentales de las entradas y salidas producidas por los comandos. Describiremos cómo el shell permite que la E/S sea redireccionada, desde o hacia, un archivo. Daremos varios ejemplos mostrando cómo esto trabaja en práctica. Los siguientes tópicos serán cubiertos en esta sección:

- E/S Estándar (I/O)
- Redirección

Entrada y Salida Estándar

Todos los comandos de GNU/Linux tienen tres flujos de salida y entrada asociado con ellos. Estos son:

stdin	-Entrada Estándar: La entrada por defecto de los comandos.
stdout	-Salida Estándar: La salida por defecto de los comandos.
stderr	-Salida Estándar de Error: Segundo flujo de salida de los comandos. Útil para las salidas de mensajes

de error y diagnostico de los comandos.

Una manera de ver los flujos de texto como si fuesen arreglos de tres elementos: stdin, stdout, y stderr, enumerados respectivamente 0, 1, y 2.

Bajo condiciones normales, los tres flujos de entrada y salida son establecidos para hacer su salida al terminal de esta manera:

- La stdin es tomada como caracteres escritos desde el teclado.
- Salidas enviadas al stdout (salida estándar) aparecerán en la pantalla del terminal.
- Salidas enviadas al stderr también aparecerán en la pantalla del terminal.

Aunque muchos de los comandos GNU/Linux no piden entrada del usuario una vez ya están en ejecución; la mayoría escriben a la salida estándar (stdout). Un perfecto ejemplo es el comando:

$ ls

Este comando causa que el contenido del directorio actual se despliegue en la pantalla del terminal. Adicionalmente cualquier mensaje de error también aparecerá en la pantalla del terminal.

Redirección

A través de la redirección es posible lograr que la salida estándar de un comando sea redireccionada y almacenada en un archivo, y no lo que normalmente se efectúa, de ser enviada a la pantalla del terminal. Esto se efectúa como en el siguiente ejemplo, empleando el caracter especial de (>):

> nombre_archivo

Simplemente añadiendo éste caracter al final del comando y el nombre del archivo a crear. El shell interpreta éste caracter como una instrucción de enviar todos los caracteres del flujo de la salida estándar (stdout), hacia un archivo de nombre nombre_archivo.• ¿Qué es la Detección de Intrusos?

El shell gestiona que los archivos puedan ser abiertos (si es que aún no existen) y vaciados (cualquier data existente en el archivo será eliminada) antes de que el comando sea ejecutado. Al haberse completado el comando, el shell gestiona que los archivos sean completamente cerrados. El comando se ejecuta sin conocimiento de que su salida estándar ha sido redireccionada.

La mayoría de los comandos de GNU/Linux escriben mensajes de error y diagnóstico a la salida estándar. Aunque ambos: stderr y stdout son por defecto conectados a la pantalla del terminal, el usuario puede separar ambos flujos. Cuando el estándar stdout es redireccionado, el estándar stderr permanece conectado a la pantalla del terminal, y así pues los mensajes de error son enviados a la pantalla.

$ls directorio/ > archivo.ls
ls: directorio: No such file or directory
$ more archivo.ls

stdin	**stdout**	**terminal**
teclado	archivo.ls	pantalla

Nota: Como no hubo salida estándar archivo.ls estará vacío, ya que no existía tal directorio.

En el ejercicio anterior quisimos listar el contenido de un directorio no existente. El mensaje de error nos informo del hecho. Note que archivo.ls es abierto y vaciado antes de la ejecución del comando, así que cualquier contenido anterior es eliminado.

El flujo de error, siendo un flujo por separado, puede ser manipulado, o mejor dicho, redireccionado independientemente de los otros. Para poder redireccionar el error estándar (stderr), preceda el caracter (>) con el

número 2, como ilustramos en el siguente ejemplo:

```
2> errores.txt
```

Esto causa que el mensaje de error y diagnóstico, junto con cualquier otra información que el comando escriba a su salida de error estándar sea escrita a un archivo de nombre errores.txt. La salida estándar aún permanecerá conectada a la pantalla, así pues, la salida normal le aparecerá en el terminal.

Podemos redireccionar cada flujo de salida completamente independientemente si utilizamos más de un operador de redireccion en la línea de comandos. Durante la lectura de la línea de comandos el shell relacionará a cada símbolo de redireccion y abrirá y preparará ambos archivos para ser escritos por los respectivos flujos del comando.

Hay veces que deseamos redireccionar ambas salidas (stdout, stderr) a un mismo sitio. Esto se logra utilizando la siguente sintaxis:

```
2>&1
```

La cual le instruye al shell a conectar o direccionar la salida 2 (stderr), al mismo sitio donde se dirige la salida 1 (stdout). El archivo de destino va a contener toda la salida estándar del comando, y los posibles mensajes de error o diagnostico, de igual manera como la hubiésemos visto en la pantalla del terminal. Note que el caracter de redirección de la entrada estándar (stdin) es el símbolo de <:

```
<archivo_nombre
```

Note que las redirecciones se aplican de izquierda a derecha de las siguientes formas:

```
$ ls directorio1 directorio2 2>&1 > archivo.salida
```

Esta sentencia conectaría el error estándar al mismo sitio donde la salida estándar se dirigía antes de ser redireccionada. En el caso por defecto, es hacia la pantalla, cuando es ejecutado desde la línea de comandos, pero, podría ser a cualquier otro sitio, cuando es ejecutado desde un script.

Redireccionando la Entrada

Es posible redireccionar la entrada estándar de un comando (stdin), para que su procedencia sea de un archivo y no desde el teclado. En la ausencia de argumentos de nombres de archivos, muchos comandos y utilitarios de GNU/Linux leen su entrada desde el STDIN; que en la mayoría de los casos es el teclado asignado a la terminal. Note que la gran mayoría de comandos permiten nombres de archivo en la línea de comandos, y por eso redireccionar, aunque está disponible, no es realmente necesario.

La notación para redireccionar el STDIN es la siguiente:

```
< nombre_archivo.txt
```

Redireccionar la entrada estándar desde un archivo nos permite escribir todos los argumentos de entrada previamente, revisar y corregir errores, así como poder reusar el archivo repetidamente.

Agregar Redirección

Es normal redireccionar la salida a un archivo existente, cualquier contenido de éste archivo será totalmente eliminado. A veces es deseado no eliminar el contenido del archivo existente para que los resultados sean acumulados. Esto se logra usando la siguiente notación:

```
>>nombre_archivo.txt
```

para la salida estándar o para el error estándar:

```
2 >> archivo.txt
```

Toda la información contenida en archivo.txt será preservada cuando usa ésta notación.

El Directorio /dev/null

El dispositivo nulo (/dev/null) cuando es usado como entrada, simplemente genera un fin de archivo (EOF) y cuando es usando para salida, desaparece cualquier data que es redireccionada hacia el. un uso típico del dispositivo null es para deshacernos mensajes de error de cualquier comando:

```
$ grep configure /etc/* 2>/dev/null
```

Esta sentencia grep es usada para efectuar una búsqueda de todas las líneas de los archivos debajo del árbol /etc que contienen la cadena configure. Como ejecutamos el comando como un usuario sin privilegios, que no podrá abrir algunos archivos, nos devolverá muchos mensajes de error que dicen así: “permission denied”. Si redireccionamos el flujo de error al dispositivo nulo, sólo veremos en la pantalla las salidas validas.

Archivos Existentes

Cuando escribimos un símbolo de redirección el shell revisará si el archivo de destino ya existe. Si no, el shell creará un archivo vacío con el nombre especificado. Si el archivo existe, el shell lo vaciará de todo contenido y entonces preparará el archivo para escritura de la salida del comando. De esta manera, la operación claramente indica que debe ser hecha con mucha cautela cuando redireccionamos data a un archivo existente. Es muy fácil perder data sin ni siquiera darse cuenta.

Rastreado de Eventos (Tracing)

Un buen IDS es mucho más que una simple herramienta de logging; esta debe poder determinar dónde un evento tomó lugar. Para los administradores de seguridad esta es la razón principal por la cual ellos implementan un IDS. Con la localización de un ataque, usted puede aprender más sobre su atacante. Este conocimiento le ayudará a disennar una solución al problema así como documentar el ataque por si hay que tomar acciones legales.

Podemos configurar la entrada y salida de bash para que no sobre escriba archivo existente si usamos la siguente opción:

```
$ set -o noclober
```

Esta opción sólo aplica a las redirecciones del shell actual; pero es posible que otros pogramas sobrescriban archivos existente la opción de noclober puede ser ignorada usando la siguiente sintaxis:

```
> | archivo
```

Ejercicio 8-1: Entrada y Salida de Comandos

En éste ejercicio cubrimos la redirección de I/O de los comandos del shell. Este ejercicio le ayudara a fortalecer los conceptos de flujo de data, de donde los comandos toman su entrada y que pasan a la data que ellos generan como salida. Las soluciones a éste ejercicio se encuentran en el Apéndice A.

1. Utilice el comando cat para leer el contenido del archivo /etc/passwd y redireccione la salida a un archivo de nombre archivo.1 en su directorio home.

   Lea el contenido de archivo.1 para confirmar que la redirección funcionó.

   Recuerde que el comando cat normalmente abre y lee un archivo existente que es proveído como argumento en la línea de comandos. Si no le suministramos dicho archivo para abrirlo para leerlo, la entrada

estándar del comando cat se revierte a la por defecto, ¿Qué es cuál?

Use el comando cat para crear un nuevo archivo y llámelo archivo.2 y coloque algún texto en éste. Escríbale alguna línea de entrada y observe que pasa. Para terminar escriba (^D) en una línea sola.

2. Ahora usaremos el mismo método anterior para crear varios archivos. Veremos cómo podemos usar el comando cat para la función de concatenar.

   Usando el mismo método, del ejercicio anterior, deberá crear tres archivos. Nómbrelos archivo1, archivo2, y archivo3. Coloque sólo una línea de texto, respectivamente, en cada uno:

   Este es el archivo1
   Este es el archivo2
   Este es el archivo3

   Aquí mostramos uno de los aspectos más útiles del comando cat; su habilidad de combinar o concatenar varios archivos en un sólo, ya sea a pantalla, como flujo de salida, o a un archivo. Pruebe con los siguientes comandos:

   $ cat archivo*
   $ cat archivo* > archivo123
   $ cat archivo123

3. Pruebe con el siguiente ejemplo y considere los principios detrás de cada comando. ¿Quién abre el archivo, cuándo, y con qué consecuencias?

   Los siguientes dos comandos aparentan tener el mismo efecto. ¿Puede usted explicar cuál es la diferencia en el tiempo de ejecución?

   $ cat archivo1
   $ cat < archivo1

   Los dos siguientes son casi idénticos, pero ¿porqué es la salida un poco diferente?

   $ wc archivo1
   $ wc < archivo1

   Puede usted explicar ¿porqué el segundo de los dos siguientes comandos no funciona?

   $ cat archivo*
   $ cat < archivo*

4. Sigamos examinando más acerca de la redirección y secuencia de los eventos. Pruebe con esto:

   $ cat archivo1
   $ cat archivo1 archivo2 > archivo1
   $ cat archivo1

   El segundo comando generaría un mensaje de error. El mensaje le dará una idea de que ocurrió mal.

   ¿Puede usted explicar el contenido de archivo1?

   ¿Cómo puede usted hacer que el contenido de archivo2 pase a archivo3, sin alterar el contenido de archivo3?

5. En éste ejemplo practicaremos redireccionando los mensajes de error.

   Primero fíjese que tipo de salida genera el siguiente comando:

   $ wc /etc/*

   ¿Cómo redireccionaría usted los mensajes de error a un archivo?

   ¿Cómo se deshiciera usted de los mensajes de error?

   ¿Cómo almacenaría usted la salida a un archivo y se deshiciera de los mensajes de error?

Ejercicios 8-2: Más Redirecciones de las E/S de los Comandos

Las soluciones para éste ejercicio se encuentran en el Apéndice A.

1. Escriba los siguientes comandos:

```
$ cat >> archivos
Este es el archivo5
Este es el archivo6
^D
```

(Recuerde que eso es un CONTROL+D al final.) ¿Qué efecto tuvo éste comando?

Y el siguiente:

```
$ ls -l archivo* >> archivos
```

Este es un método muy común de añadir líneas al archivo de configuración, en su directorio home, .profile o .bash_profile sin tener que recurrir al uso de vi. Siempre asegúrese de usar los caracteres dobles (>>), y no el simple (>).

2. Desde su directorio home, Escriba el siguiente comando:

```
$ wc /etc/* 2>&1 > archivos.etc
```

¿Porqué seguimos viendo mensajes de error en pantalla? ¿Qué debió haber escrito para enviar los mensajes de error al mismo archivo que la salida estándar?

¿Qué hace el siguiente comando?:

```
$ wc /etc/* > archivos.etc 2>&1 > archivos2.etc
```

Pruebe el comando a ver si tiene usted razón.

3. Establezca la opción de noclobber para la redirección del I/O del shell y escriba los siguientes comandos:

```
$ set -o noclobber
$ cat archivol > archivo.nuevo
$ cat archivo2 > archivo.nuevo
$ cat archivo2 >> archivo.nuevo
$ cp archivo2 archivo.nuevo
```

¿Cuáles comandos funcionaron y porqué?

¿Qué cree usted que se logra con el próximo comando?

```
$ cat archivo1 > | archivo.nuevo
```

4. ¿Cree usted que el siguiente comando funcione?

```
$ 2 > /dev/null < archivo.nuevo >> /tmp/archivo.log cat
```

¿Puede usted explicar?

Tuberías y Filtros

En esta sección introducimos el uso de tuberías (pipes) para extender la funcionalidad de los comandos. Las tuberías son como la redirección, pero, trabajan un poco diferente a ellas. Las tuberías permiten que los flujos de I/O (entrada/ salida) de una serie de procesos sean conectados, encadenando de esta forma los comandos de una tubería a otra.

Gravitando desde la descripción de una tubería nace la idea de programas filtros; programas que están diseñados para trabajar en tuberías. Muchos de los filtros de estándar de GNU/Linux son descritos, demostrando lo fácil que es construir y más complejos utilitarios basados en bloques básicos.

Los siguientes tópicos serán discutidos en esta sección:

- Tuberías/Pipes
- Filtros/Filters

Las Tuberías (Pipes)

Si queremos procesar la salida estándar de un comando como la entrada estándar de otro comando, podemos guardar la salida estándar vía el comando de redirección a un archivo temporario:

$ Who > archivo.temp

Ejecutamos el comando who para ver quien esta ingresado en el sistema. La salida del comando who es un usuario por línea, que nos rinde información muy útil, la cual almacenamos en archivo.temp.

Luego corremos éste comando:

$ wc -l < archivo.temp

Esta sentencia cuenta el número de líneas en archivo.temp. El resultado final es lo que deseamos saber, cuantos usuarios se encuentran ingresados (logged in) en el sistema.

GNU/Linux provee un método muy útil de acortar éste proceso. Con el uso del caracter de tubería (|),podemos combinar los comandos que ejecutamos previamente y combinarlos en uno sólo:

$ who | wc -l

El carácter de tubería le indica al shell que conecte la salida estándar (stdout) del comando a la izquierda (who), a la entrada estándar (stdin) del comando en la derecha (wc -l). Ahora la salida del comando who es automáticamente pasada al comando wc -l, sin la necesidad de tener que crear archivo.temp, que tuvimos que hacer anteriormente.

El resultado de éste comando, asistido por una tubería, es producir el número de personas que se encuentran ingresados en el sistema. También como resultado de filtrar con la tubería, vemos, que perdimos parte de la información del comando who, pero, ésto no tiene importancia, ya que sólo estamos interesados en el resultado final, que es el número de personas.

Al igual que en el caso de la redirección de I/O, ambos comandos involucrados en una tubería se ejecutan sin conocimiento de que están conectando sus flujos de salida a otro comando. El shell establece la conexión de tubería antes de ejecutar los comandos.

Ambos comandos son conectados por la tubería simultáneamente. Si el lector hace intentos de lectura en el momento en el que la tubería está vacía, el lector esperará en la tubería antes de continuar. Si se llena la tubería (en el caso de que exista un límite al tamaño de la tubería), el escritor espera hasta que alguna data haya sido removida desde la otra punta de la tubería.

Las tuberías sólo operan en los flujos de salida estándar de los comandos. Todo mensaje escrito al error estándar del comando será escrito a la pantalla del terminal, a menos que, éste sea redireccionado también por separado.

Las tuberías pueden ser usadas para conectar dos o más comandos, hacemos la mención en éste punto, ya que sólo hemos efectuado ejemplos con dos, por cuestión de brevedad. Conexiones de múltiples tuberías es cosa muy común en GNU/Linux, todo ésto es gracias a una clase general de utilitarios, mejor conocidos como filtros. Estudiaremos los filtros más detalladamente en lo adelante, pero sólo una breve explicación, por ahora, es que un filtro es un programa que lee los flujos de su entrada estándar (stdin), y escribe sus resulta-

dos a la salida estándar (stdout), y así llevando a cabo un filtrado a través de un sistema de tuberías.

En éste ejemplo mostramos una doble tubería:

```
$ who | grep ttyp | wc -l
```

La salida del comando who es procesada por el comando grep, comando que filtra (elimina todas las líneas

que no incluyen la cadena de caracteres pasada como argumento a grep) las líneas que no contienen la

cadena “ttyp”. La salida es finalmente pasada por la tubería al comando wc, que procede a contar el número

de líneas, que corresponde directamente al número de usuarios conectados en toda la red.

La capacidad de poder conectar comandos de esta manera, para formar utilitarios grandes y poderosos, es

uno de los aspectos de GNU/Linux que lo convierten en una herramienta verdaderamente productiva. Los utilitarios

estándares muy a menudo sólo son pequeñas piezas con las cuales se construyen grandes herramientas

a través del uso de tuberías y redireccionamiento. Sólo con mucha práctica es que esta destreza puede ser

desarrollada.

Los Filtros

Un filtro es un programa que lee data desde su entrada estándar y escribe a su salida estándar.

Normalmente, un filtro tomará su entrada y la manipulará de alguna manera para producir la salida

deseada. Por ejemplo, puede ser que necesitemos remover líneas, agregar otras o reordenar. Cada filtro por

lo regular, lleva a cabo una función simple. Podemos construir poderosos comandos con el uso de tuberías,

simplemente colocando una serie de filtros. Otros sistemas operativos sólo pueden lograr éste nivel de funcionamiento

escribiendo funciones en programas. Para poder dominar a GNU/Linux totalmente, es imperativo

dominar éstos temas de redirección y tuberías para poder forjar sus propios filtros a través del uso de utilitarios

y comandos GNU.

La gran mayoría de programas filtros pueden tomar su entrada desde su entrada estándar, o como argumentos

desde la línea de comandos. Algunos comandos que normalmente están a la espera de nombres de

comandos especificados en la línea de comandos como argumentos aceptarán el nombre de un archivo o un

símbolo de menos (-) para especificar que deben leer la entrada estándar como el archivo de entrada.

Expondremos a algunos de los programas de filtrado más útiles disponibles en GNU/Linux en esta próxima

sección.

Comando tee

El comando tee es muy útil para capturar resultados desde los comandos intermediarios. El comando tee envía la salida de una tubería a dos sitios diferentes, uno va a la salida estándar (comúnmente la pantalla o la entrada de otra tubería si continua la cadena), y el otro, al archivo nombrado como argumento

al comando. El comando tee puede ser usado en el siguiente formato:

$ tee -a nombre_archivo

La opción -a obliga a tee a entrar en modo de agregar (append mode) y no, su por defecto, que es, primero

eliminar data en el archivo dado como argumento.

Comandos head y tail

Los comandos head y tail son filtros bien simples. A diferencia de que sólo envian una porción de la data seleccionada a la salida estándar, son muy parecidos al comando cat. La data seleccionada no es manipulada en ninguna manera; solamente se envía directamente a la salida estándar.

El comando head es típicamente usado para leer los cabezales de archivos, al contrario de tail, que es mayormente usado para ver las últimas entradas de los archivos de diarios (logs).

199

Comando cut

El comando cut es utilizado para cortar secciones de data desde un flujo de texto entrante. Las secciones pueden ser especificadas por posición de columnas (-c) o por números de campos (-f). Los campos, por defectos, son tabulados, pero la opción -d, nos permite usar cualquier carácter único como separador.

Las columnas y los campos son especificados como una lista separada por comas. Cada ítem en la lista es representado por un dígito o un rango de números. Un guión es usado para separar el principio y el fin de

un tango de valores.

-c1-5 Primeros cinco caracteres en la línea

-f1,5 Primer y el quinto campo

-fl-3,7-8 Primeros tres, séptimo y el octavo campo

Podrás cortar por campo o por columnas, pero no ambos.

Comando sort

El comando sort es un utilitario muy poderoso para ordenar. Este puede ordenar utilizando la línea completa o especificandole campos clave. Los caracteres que diferencian campos pueden ser definidos como una opción desde la línea de comandos. Los campos pueden ser vistos como cadena de caracteres o números; las cadenas pueden ser literales o tipo diccionario (ignorando espacios y tabulaciones). Las líneas pueden ser ordenadas en orden ascendente o descendente.

Por defecto, el comando sort se aplica a la línea entera. Podemos sortear por más de una palabra clave si especificamos la opción de principio y fin de campo.Los campos son enumerados desde el cero hacia arriba.

Para ordenar sólo basandose en el segundo campo, use:

$ sort +1 -2

El ordenado por defecto de sort es una comparativa de orden ascendente de una cadena. El uso de opciones modifica el criterio de ordenanza. Para ordenar el tercer campo como un número, y entonces, el segundo campo como cadena de caracteres en orden reverso, use el siguiente comando:

$ sort +2n -3 +0r -2

Otra opción adicional del comando sort permite dar entrada a multiples archivos para convergerlos en un sólo archivo de salida. Sort es un comando amplio y existen muchas otras opciones que permiten un sin número de opciones, para empezar a estudiar éste comando, es recomendable leer su página man y mucha práctica.

Aquí unos ejemplos del comando sort:

• Para ordenar el archivo de contraseñas por el número de ID del usuario (UID):

$ sort -t: +2n -3 /etc/passwd

• Para ordenar el archivo de contraseñas por los shells de login, y luego el nombre de usuario:

$ sort -t: +6 +0f -1 /etc/passwd

Este es un ejemplo de un ordenado por múltiples campos clave.

• Para ordenar el archivo de los grupos en orden reversa:

$ sort -r /etc/group

Comando uniq

El comando uniq compara líneas adyacentes desde un flujo de entrada y normalmente elimina la segunda

y repetidas líneas cuando ocurren repeticiones. Es muy frecuentemente utilizado en conjunto con

el comando sort para remover líneas duplicadas desde data ya sorteada.

200

Opciones pasadas a uniq que permiten salida de:

• Una copia de cada línea repetida o duplicada (-d).

• Líneas que no están repetidas (-u) o únicas.

Si es usado por defecto sin ninguna opción, uniq producirá el equivalente a las opciones previamente

explicadas de (-ud).

Comando tr

El comando tr es usado para traducir caracteres desde la entrada estándar. Por defecto, el comando sólo lee desde la entrada estándar y no lee nombres de archivo, pero pueden ser suministrados desde la línea de comandos.

El comando tr lee cada caracter que procede desde la entrada estándar. Si el caracter existe en la primera lista de caracteres dada como argumento, éste será reemplazado con el primer caracter correspondiente de la segunda lista pasada como argumento desde la línea de comandos. El caracter substituído es entonces enviado a la salida estándar.

Si usamos la opción -d, todas las ocurrencias de los caracteres en la primera lista de caracteres serán eliminados; y, claro, no se requiere una segunda lista como argumento. Si utilizamos la opción -c, todos los caracteres que no igualan a los de la primera lista, se traducen a los de la segunda lista. La opción -s elimina todas las ocurrencias de caracteres repetidos de los caracteres a traducir.

Los caracteres de la cadena son especificados literalmente o como un rango dentro de llaves cuadradas [], separando el caracter inicial del final con un guión (-), por ejemplo:

[A-Z] Todos las letras mayúsculas

Una secuencia de caracter asterisco número (aplicado a la segunda cadena) significa que el caracter anterior el número de veces especificado, por ejemplo:

[A*5] Cinco copias de la letra A; igual que AAAAA

Si el asterisco no es seguido por un número, el caracter es automáticamente repetido para igualar la segunda cadena a la primera, ya que ambas, deben tener el mismo número de caracteres; veá el último ejemplo.

El caracter de escape (\) es usado para prevenir que caracteres especiales sean reconocidos y para especificar caracteres no imprimibles por su código octal ASCII.

En éste ejemplo vamos a introducir el concepto de construir grandes estructuras basadas en tuberías desde pequeños componentes. Por ejemplo:

```
$ ls -l| tail +2 | tr -s `<tab><space>` `<space><space>` | cut -d ``<space>` -f5,9 | sort +0nr
$ ls -l| tail +2 | tr -s “ “ “ “ | cut -d “ “ -f5,9 | sort +0nr
867 archivo.1
867 archivo.2
867 archivo.3
867 archivo.4
867 archivo.5
```

En éste ejemplo listamos el contenido del directorio actual, con el tamaño y luego el nombre de los archivos. Para entender éste ejemplo y cualquier otro ligado con tuberías, deberás entender cada etapa, o sea, de tubería en tubería. No debes continuar a la próxima sin entender la anterior.

Otra manera alternativa de llevar a cabo el ejemplo anterior sin usar el comando tr, es cortar las columnas desde la salida del comando ls.

```
$ ls -l | tail +2 | cut -c30-36,51- |sort +0nr
867 archivo.1
867 archivo.2
867 archivo.3
867 archivo.4
867 archivo.5
```

Aquí presentamos un último ejemplo. Tomamos un archivo de cualquier formato y le revisamos su ortografía contra un archivo de referencia ya existente.

```
cat archivo.txt |tr ‘[A-Z]’ ‘[a-z]’ | tr -cs ‘[a-z]’ ‘[\012*]’|sort |uniq|comm -23 - lista-palabras
```

Este ejemplo, aunque se torna un poco complejo, nos ayuda a ilustrar características muy interesantes. La primera tubería, recibe su data desde el comando cat, la tubería le pasa la data al comando tr ya que tr no acepta nombres de archivo (aunque podemos usar la redirección de tr < nombre_archivo). Luego tr procede a reemplazar todas las mayúsculas con minúsculas, luego en la próxima tubería, tr reemplaza todas las ocurrencias que no sean caracteres alfabéticos, con el caracter de newline, lo que, en escencia, produce una lista de palabras, una debajo de la otra, sin ningún espacio entre ellas. Así que, cada palabra aparece en una línea por sí sóla y todos los caracteres que no son imprimibles desaparecen. Luego, las líneas son ordenadas, entonces, el comando uniq se deshace de las entradas duplicadas y manda a la salida sólo las líneas únicas (ésto funcionó ya que primero ordenamos, sino, hubiese sido diferente). Ya, para finalizar, se usa el comando comm, comando que compara nuestro archivo transformado con el archivo que le damos de referencia en el argumente final, y nos muestra las diferencias, que debemos presumir, son las palabras que deletreamos mal.

Ejercicio 8-3: Tuberías y Filtros

En éste ejercicio practicaremos los conceptos de filtros y tuberías. En los primeros, le guiaremos paso a

paso y luego dejaremos que aporte las respuestas. Las soluciones a éste ejercicio se encuentran en el Apéndice A.

1. Listemos directorios con la asistencia de filtros:
   $ ls -l /usr/bin | more
   $ ls -l /usr/bin | wc -l
   $ ls -l /usr/bin | grep root | more
   $ ls -l /usr/bin | tee lsbin.txt | grep root | more
   $ more lsbin.txt
   $ ls -l /usr/bin | grep root | | tee lsbin.txt | more
   $ more lsbin

2. Usando los comandos aprendidos, escriba un comando asistido por tuberías que nos reporte cuántos usuarios se encuentran ingresados en el sistema (login). No debe listar los nombres, sólo el total.

3. Experimentemos diferentes maneras de ver un mismo archivo:
   $ grep bash /etc/passwd | sort |more
   $ cut -d: -f1 /etc/passwd | sort |more

4. Para ésta práctica deberá crear un archivo en el editor (vi) y llamarlo estudiantes.txt, éste archivo debe tener el siguiente formato de texto y separado por coma, y tener los campos de data que listamos más adelante.

   Este archivo nos servirá para los siguientes ejemplos, así pues que primero, escriba uno con los nombres del ejemplo que aquí se provee:
   • Nombre
   • Matricula
   202
   • Zona
   • Edad
   • Barrio
   • Año de Nacimiento
   • Año de Inscripción
   • Carrera
   • Mención

   Si tenemos un campo el cual no ésta definido, deberá usar un guión (-), para ocupar su lugar. Si un alumno cursa más de una mención, columnas adicionales deberá ser añadidas por cada mención adicional, así pues que, no todas las entradas tienen el mismo número de columnas.

   Aquí le mostramos entradas de ejemplo, para que usted pueda crear su propio archivo de práctica:
   Jose Paredes,19750726,Interior,27,Herrera,1975,Sistema,Programacion,Base de datos

   En éste ejemplo vemos el nombre (Nombre Apellido) seguido por su fecha de nacimiento (añomesdia), su Zona de procedencia (Interior o Capital), etc. Debe crear un buen número de entradas en su archivo que usare para este ejemplo.

   Usando los lectores paginadores less o more, examine el archivo que digitó en vi, para familiarizarse con su contenido.

5. ¿Cuál sentencia de grep nos mostraría todos los alumnos de la Capital? ¿Cuál sentencia nos mostraría los de la carrera de Sistemas?

   Usando grep, muestre todos los estudiantes que no estudian Sistemas.

6. Usando el comando cut muestre las provincias y las edades de los estudiantes.

¿Cuál comando nos mostraría el nombre del estudiante y su Zona? ¿Cuál comando nos mostraría las Zonas, matriculas y carreras?

Liste todos los alumnos , barrios, y las menciones. Liste las primeras tres letras de cada nombre de estudiante.

7. Usando los comandos cut y grep, muestre los estudiantes que estudian Sistemas, despliegue el nombre del estudiante y las todas las menciones.

   Muestre todos los nombres de estudiantes cuya zona fr procedencia es la Capital. Muestre edad, fechas de nacimiento, y carrera para todos los estudiantes de la Capital.

8. ¿Cuál es la sentencia del comando sort para ordenar alfabéticamente por nombre de estudiante?

   ¿Cuál sentencia del comando sort ordenaría por matriculas de menor a mayor? ¿Cuál sentencia del comando sort ordenaría con la mayor matricula primero? ¿Cuál sentencia de sort lo ordenaría por orden de fecha de inscripción?

9. Muestre todos los estudiantes de Sistemas, las edades, y las carreras de todos los estudiantes del interior, ordenados alfabéticamente por el nombre del estudiante.

   Muestre los estudiantes, edades y las carreras, ordenadas por las edades de los estudiantes.

   Muestre los estudiantes del Interior, listando sus fechas de inscripción y ordenadas en orden reversa.

10. Cuente el número de estudiante de la Capital. Cuente cuántos estudiantes son de sistemas y del interior.

11. Muestre todas las diferentes carreras del archivo completo. Muestre todas las diferentes menciones.

Ejercicios 8-4: Tuberías y Filtros Uso Avanzado

En éste ejercicio, seguimos usando el archivo estudiantes.txt anterior. Las soluciones de éste ejercicio se encuentran en el Apéndice A.

1. Muestre todos los nombres de estudiantes y la Zona del Interior ordenada por Zonas.

   Muestre todos los estudiantes, listando sus nombres, zona, y su carrera oficial; ordenado, primero por carrera, y entonces, la zona, y finalmente, por el nombre del estudiante.

   Repita el último filtro pero no incluya esos estudiantes que no tienen menciones.

2. Muestre los 5 primeros estudiantes, por mayoría de edad, listando sólo el nombre y la edad.

   Muestre sólo los nombres de los estudiantes y el barrio, pero, ordénelo por la edad, sin mostrarla.

   Muestre sólo los estudiantes y su zona, pero, ordénelo por la edad del estudiante y sólo despliegue los cinco de mayor edad.

3. Liste el estudiante de mayor edad de cada Zona.

Scripts del Shell

Como todos los antecedentes de UNiX, GNU/Linux tiene su propia manera de agrupar comandos en un archivo ejecutable, conocido en el mundo UNiX como Scripts del Shell. La comparación con otros puede ser peligrosa, ya que el protagonismo de éstos scripts en los otros sistemas operativos no eran de gran importancia. En UNiX, al igual que en GNU/Linux, éstos scripts juegan un rol esencial, por la caracte-

rística de multítareas del shell de GNU/Linux, que puede llevar a cabo muchas cosas simultaneas desde un script; otros eran sistemas mono-tarea y eran limitado por ende a ejecutar una cosa a la vez. Multítarea implica complejidad de operación, y éstos scripts sirven en el rol de guión para orquestar, monitorear y coordinar toda esta complejidad.

En esencia y funcionalidad, un script desempeña el mismo papel que cualquier otro archivo ejecutable. Este puede leer desde la entrada estándar (STDIN), escribe a la salida estándar (STDOUT) y al error estándar (STDERR). También podemos abrir y cerrar archivos. Puede recibir entrada desde el usuario (user input). Puede abrir una base de datos, editarla, y cerrarla. En fin todo lo que se puede llevar a cabo desde la línea de comandos, se puede efectúar desde un script. Los scripts son muy usados en las partes más criticas de los sistemas GNU/Linux. La inicialización del sistema, inmediatamente después del gestor de arranque, es manejada mayormente por scripts. El inicio del Sistema X Window es normalmente manejado por un script. Para poder ser considerado como un administrador de alta eficiencia de sistemas GNU/Linux, usted debe entender los scripts y sus roles en éstos sistemas.

Los Scripts del Shell son, simples archivos de texto, normalmente creados por editores de texto (vi siendo el más popular y el emacs el segundo), pero, pueden ser creados por cualquier flujo de texto que produzca un archivo. Así, que un script mismo, puede generar, y luego ejecutar, otro script en tiempo de ejecución, y luego eliminarlos después que ya no sean necesitados. Esto puede darnos una idea de lo flexible que son los scripts.

En esta sección cubriremos los siguientes tópicos:

- Ejecutar un Script
- Bang-bang y Comentarios
- Argumentos y Parámetros Especiales
- Variables de Ambiente y del Shell
- Sentencias de Control
- Lectura de Entrada del Usuario

Ejecutar los Scripts

Crear archivos scripts desde cero, involucra cuatro pasos:

1. Identificar qué es lo que desea efectuar con el script.

2. Determinar con cual o cuales comandos efectúa la acción deseada.

3. Usar un editor para crear la secuencia de comandos en un archivo de texto.

4. Usando el comando chmod asignarle los bits de permisos de ejecución al archivo.

He aquí un ejemplo sencillo: digamos que tiene su sistema Gráfico muy inestable y que las aplicaciones están abortando inesperadamente y produciéndole archivos de volcado core, y éstos le están poblando todo su espacio en disco; con el siguiente comando podemos buscar todos éstos archivos core, empezando desde el directorio raíz (/):

```
$ find / -name 'core' -print
```

Aunque maneje bien la sintaxis de find, eventualmente se cansará de escribir la sentencia, una y otra vez, para llevar a cabo tan simple acción. Para automatizar ésto crearemos un script que efectúa la acción.

1. Abrimos un archivo y lo llamamos busca.core:

```
$ vi busca.core
```

2. Escriba el siguiente código sin errores ortográficos:

```
#!/bin/bash
# busca.core - Utilitario para efectuar búsquedas de archivos core en el sistema
#
find $1 -name 'core' -print
```

3- Guardamos el archivo de esta manera en el vi:

```
:wq
```

4. Ahora lo hacemos ejecutable por todos los usuarios:

```
$ chmod +x busca.core
```

5. Copiamos el script al directorio /usr/bin, para que así, se coloque en nuestra ruta de ejecución. Para usarlo simplemente:

```
$ busca.core /usr
```

Esta simple sentencia buscará todos los archivos core empezando desde el directorio (/usr) que usted le pase como argumento desde la línea de comandos. Al finalizar éste ejemplo, puede ver lo simple que es crear scripts que le sean muy útiles. Todos los scripts son variaciones a éste simple ejemplo. Puede ser que sean más complejos, pero al final, son muy parecidos.

Shebang y Comentarios

Note los simbolos #!, #, y $1 en éste ejemplo. La sintaxis #! es un mecanismo que permite la integración de un script desarrollado un shell que sea ejecutado en otra. En nuestro ejemplo, éste le informa al shell que se está ejecutando: que debe usar el programa /bin/bash para ejecutar el resto del script. Si la primera línea de un script empieza con #!, mejor conocido como el shebang (# es el símbolo de prompt del shell y ! es, a veces, llamado el caracter bang), la próxima palabra es tomada como el nombre del programa a ejecutar, con su entrada estándar direccionada a leer desde el resto del script. Esta línea siempre debería ser incluida en todos los scripts. Los diferentes shells tienen diferente sintaxis y capacidades. El mecanismo del shebang garantiza que, un usuario, usando un shell de ingreso al sistema diferente, podrá utilizar el script, sabiendo que ejecutará bien desde el shell correcto. Es de suma importancia para los scripts de mantenimiento del sistema, ya que ésto garantiza que, por un simple cambio del shell de ingreso por defecto, digamos, del root, no se arruinarían todos nuestros scripts de mantenimiento.

Los comentarios normales empiezan siempre con un # y deben ser usados dentro de los scripts del shell para describir la funcionalidad del código. Es siempre buena idea empezar cada script con una breve descripción de su función y cualquier tipo de argumento que debe ser suplido desde la línea de comandos.

Argumentos y Parámetros Especiales

Los simbolos $<dígitos> son usados para pasarle argumentos desde la línea de comandos. Cualquier argumento pasado desde la línea de comandos y especificado en el script es almacenado en Parámetros posicionales enumerados desde S1..S9. El número asignado al script mismo es el $0.

Existe un número de parámetros especiales disponibles en el script: el número de argumentos pasados a un comando es almacenado en la variable $#, y la variable $* es un cadena que representa los argumentos de la línea de comandos. Ejemplos de éstos a veces son int argc y también char *argv[] como argumentos a la función main() del lenguaje de programación C. El parámetro $@, es como el $*, pero, los almacena como entidades individuales, y no, como una cadena.

"$*" ===> "$1 $2 $3 ...."
"$@" ===> "$1" "$2" "$3" ....

El parámetro especial $$ almacena el PID del último proceso del shell actual. Esto es muy útil para generar nombres de archivo únicos para archivos temporales:

```
# para almacenar lista de los archivos en un archivo temporal
ls > /tmp/ls.$$
```

Aún tenemos dos Parámetros más que valen la pena mencionar, ellos son: $? parámetro que retorna el estado retornado del comando más recientemente ejecutado desde el foreground, y el parámetro $!, que es el PID del comando más recientementre ejecutado como proceso desde el background. Todas éstas cosas tienen uso de mucha importancia para poder enlazar comandos basados en sus estados. A diferencia del ambiente de usuario único de sistemas como en otros sistemas, los scripts de GNU/Linux pueden lanzar procesos en el background y continuar ejecutandose. Los Parámetros existen para ayudar a monitorear el sistema mientras los scripts y sus subprocesos se están ejecutando.

Variables de Ambiente/Shell

Las Variables del shell proveen un instrumento de colocar en un archivo la información de la línea de comandos en un programa del shell cuando éste es llamado desde la línea de comandos con argumentos o parámetros temporalmente almacena información dentro del programa shell. Variables del shell también pueden tener asignadas a ellas valores de la siguiente manera:

```
$ nombre-variable=valor-variable
```

Cuando asignamos valores, como en éste ejemplo anterior, el símbolo de dinero ($) no debe preceder la variable. Pero, en todos los otros usos, el símbolo de ($) debe preceder el nombre de la variable, como en el siguiente ejemplo:

```
$ echo $nombre-variable
```

Una variable de ambiente es similar a una variable del shell, excepto, en que es un cadena constante. Una variable de ambiente es pasada a un proceso hijo para inicialización de programas.

Otra diferencias entre las variables del shell y las de ambiente, concierne, a su visibilidad, en ambos casos, dentro y fuera de los programas del shell. Las variables de ambiente son variables globales, ya que, cada programa shell tiene acceso a ella, mientras que, las variables de shell, por otro lado, son variables locales; su visibilidad es restringida al shell en particular en las que ellas se encuentran. Una variable del shell que se establece y es usada dentro de un shell, no estará accesible por otro shell, a menos, que haya sido exportada con el comando export. Consideremos éste ejemplo:

```
$ export nombre-variable
```

La variable de nombre nombre-variable aquí exportada se comportará ahora como una variable de ambiente, poseyendo visibilidad global.

Sentencias de Control

Las sentencias de control le permiten hacer, y ejecutar, diferentes partes del script del shell, dependiendo de condiciones variadas y controladas.

Comando if

El comando interno if es utilizado para poner a prueba el éxito o fracazo de un comando. Si el comando fué éxitoso, los comandos después de la palabra clave son ejecutados. El comando if posee com-

ponentes y opciones para especificar otros comandos a ser ejecutados en el caso, de que, el comando en prueba falle (else), y para poder especificar múltiples pruebas también (elif).

Las líneas de comandos pueden ser tan simples o tan complejas como sea necesario; por ejemplo, un comando único, una tubería, o una lista de comandos secuenciales separados por punto y coma. En cada uno de los casos, el último de los comandos ejecutado es el que define el estado del bucle if.

Las palabras clave if, then, elif, else, y fi no tienen porqué ser especificadas en líneas separadas. La regla de sintaxis es que, las palabras clave, deben estar al inicio del comando (puede estar al inicio de una nueva línea o después de un punto y coma o después de un símbolo de tubería). Algunos shells de GNU/Linux imponen aún una restricción más, y es que la palabra clave deberá ser la última en la línea (el resto de la línea es entonces ignorado, y en algunos de los casos, generará un error si se encuentra otra cosa que no sea un comentario).

El comando if puede ser extendido para proveer la sintaxis clásica else-if (elif) y else.

El comando if del shell provee el mismo alcance, poder y características que el if de lenguajes de ambiente de alto nivel.

El flujo de procesado de la estructura es siempre igual: primero la prueba que sigue la palabra clave es ejecutada y llevada a cabo; si ésta retorna un valor de verdadero, entonces el comando es ejecutado, y las pruebas subsecuentes son ignoradas. Si ésta primera prueba fracasa, entonces la prueba elif es llevada a cabo. Si ambas pruebas fracasan, entonces el comando else es ejecutado.

Múltiples pruebas de elif pueden ser usadas, y el proceso de prueba avanzará en cascada a través de los items, secuencialmente, hasta que la prueba devuelva un valor verdadero (true), y en éste momento, los

comandos asociados son ejecutados y la estructura del if es abandonada por completo.

Desde otro punto de vista, sólo un bloque de la estructura es ejecutado, y éste es el bloque que retorna el valor verdadero, que es el que pasa la prueba, sino se ejecuta el bloque de else, si todos las pruebas fallan.

Comando test

El comando test está diseñado especialmente para ser usado con la estructura de control de flujo interno del shell, como es el if-then-else. Su función es aplicar varias condiciones de prueba definidas por sus argumentos, y salir con éxito o un código de error, dependiendo del resultado de la prueba. En la mayoría de los sistemas GNU/Linux, el comando test es estándar del sistema, pero algunas versiones del shell puede que tengan su propio comando interno test (el shell bash tiene su propio comando interno test).

Las strings (cadenas de caracteres), pueden ser probadas (test) para valores de largo, de no-ceros y ceros o pueden ser comparados por igualdad o no igualdad, medidas contra valores establecidos. Al substituir variables de tipo string para comparar, recuerde incluirlas dentro de comillas alrededor de la cadena, para prevenir substituciones de valores de cadenas en blanco.

Debemos tener, también, mucho cuidado, al usar los comodines de expansión de caracteres en variables constantes de strings (*, ?, y []). A menos que éstas estén definidas dentro de comillas, las reglas de comodines de expansión serán aplicadas, y el resultado quizas, no será el esperado.

El shell bash tiene su propia versión extendida del comando test, la cual utiliza llaves cuadradas o corchetes dobles, para delimitar condiciones de prueba. Un número extra de condiciones han sido provistas. Un

gran número de condiciones de archivos también han sido agregadas, incluyendo pruebas de comparación de fecha de modificación de los archivos. Las comparaciones de cadenas han sido extendidas para permitir que la cadena a la derecha pueda incluir caracteres comodines (wildcard) para igualar patrones:

```
if [[ $(tty) = */pts* ]]; then
# Este es el login de la red
fi
if [[ $SHELL = *bash ]]
# Usamos el shell bash
if [[ $LOGNAME = [a-m]* ]]
# Nombre de login alfabéticamente de la a a la m
fi
```

El comando test soporta muchas opciones útiles para llevar a cabo pruebas del estado de los archivos. Las simples fueron mostrados anteriormente. Existen pruebas adicionales; para detalles completos diríjase a las páginas man.

Comando expr

El comando expr tiene soporte mínimo para la comparativa de números enteros (integer). Los argumentos del tipo cadena (cadenas) son revisados de derecha a izquierda por contenido de dígitos decimales y son convertidos a números para la comparación. Un caracter que no es un dígito detiene la conversión, pero no se generá un error. Los números que no empiezan con un dígito se convierten a cero.

El comando expr puede usarse para ejecutar simples operaciones aritméticas eb imprimir los resultados a la salida estándar, por ejemplo:

```
$ expr 4 + 5
9
$
```

En conjunto con el comando expr, las comparaciones numéricas de prueba le dan al shell una capacidad limitada del manejo de números enteros. Las pruebas pueden ser combinadas utilizando los operadores Booleanos y, OR, y NOT (&&, ||, !).

Comando for

El comando for es usado para iterar por una serie de valores. Es muy similar en operación a los bucles tradicionales de los lenguajes de alto nivel, pero utiliza una serie de valores de tipo cadena y no un rango de valores numéricos. Un simple uso del comando for es para aplicar el mismo comando a una serie de valores.

El siguiente ejemplo es una manera de eliminar los archivos de reguardo creados por el editor del GNU emacs:

```
$ for ext in .CKP . BAK
do
rm *.$ext
done
```

Comando exit

El comando interno exit puede ser usado por un script para retornar un estado de exit. Un script, por defecto, sale después que el último comando en él ha sido ejecutado; y sale con el mismo valor del último comando que están ejecutado. El comando exit toma un argumento opcional, el cual es el valor del número entero al salir del script. El comando exit puede ocurrir en cualquier momento del script.

Los shells de GNU/Linux (bash, tcsh, y pdksh) ofrecen característica adicionales, las cuales la convier-

ten en poderosos ambiente de programación. Muchos libros an sido escritos sobre el tema de la programación básica del shell. En conjunto con el poder inherente de comandos individuales como son awk, sed, find, grep, etc., debe estar claro que, la programación shell es un tópico muy amplio. Un buen sitio para aprender a programar el shell es dentro de los directorios de los scripts de inicialización de su sistema GNU/Linux. El acceso a éstos archivos de directorios, como /etc/rc.d, requiere permisos de root, pero aunque no lo sea, sólo necesita copias de éstos para leerlos y aprender de ellos. Al abrir éstos scripts y poder leerlo le servirán como buena fuente de referencia para encaminarse a dominar tan importante tarea como administrador o programador. La única manera de aprender a programar el shell, es igual que cualquier otro lenguaje de programación, práctica, práctica y más práctica.

Leyendo las entradas del Usuario

El comando interno read habilita el shell para poder leer los argumentos interactivamente desde la entrada estándar y les asigna los valores capturados a las variables. La línea de entrada es fragmentada en argumentos individuales (Palabras separadas por espacio y tabulados), y cada argumento es asignado a una variable única. Los argumentos son asignados en orden de izquierda a derecha, a la lista de variables especificadas. Si proporcionamos argumentos en exceso, las variables no usadas son asignadas a una cadena NULL.

Un error común con las lecturas es empezar cada variable con el símbolo de $.

```
read $alfa $beta $gama
```

Esto causaría que tome lugar la sustitución de variables, y que el valor de la variable sea utilizado como nombre y no como la variable misma. El formato correcto es utilizar el nombre de la variable:

```
read alfa beta gama
```

Para otorgar un prompt sin una nueva línea al final de éste, use el comando echo y termine la cadena del prompt “\c”:

```
echo “Introduzca su nombre: \c”
read NOMBRE
```

En los sistemas GNU/Linux echo toma una opción (- n), que suprime las líneas nuevas y no sin utilizar al contrario de usar el caracter \c.

Ejercicios 8-5: Shell Scripts

Este ejercicio le ayuda a desarrollar los conceptos para poder escribir los scripts de shell de GNU/Linux, Buscando características comunes, como son, parámetros posicionales, ejecución de comandos condicionales, entradas de usuarios y bucles. La solución a éste ejercicio se encuentra en el Apéndice A.

1. En éste ejercicio, usted va a crear un script sencillo llamado rev_usuario. El script podrá aceptar un argumento (un nombre valido de usuario) y el mismo imprimirá toda la información sobre el usuario.

   Cree un directorio llamado mis-scripts en su directorio home. Entre al directorio y cree un archivo llamado rev_usuario conteniendo las siguientes líneas:

   ```
   #!/bin/bash
   who | grep $1
   ```

   Déle permisos de ejecución al script, para ser ejecutado:

   ```
   ./rev_usuario nombre-del-usuario
   ```

   Ahora diríjase a su directorio home y ejecute el comando rev_usuario de nuevo. ¿Puede usted ejecutar el comando? y si no puede, ¿por qué no? ¿Qué tiene usted que hacer para agregar el script a su ruta de eje-

cución (PATH)? Asegúrese de que los cambios efectuados a su PATH sean permanentes para que cada vez que ingrese al sistema pueda seguir ejecutando el comando rev_usuario.

Ahora agregaremos manejo de errores a nuestro script. Después de la línea de #!/bin/bash, agregue las siguientes líneas. Estas líneas generan un mensaje de error si el script no se ejecuta con, exactamente, un sólo argumento (el nombre del usuario).

```
if [[ $# !=1 ]]; then
echo "$0: Error, Faltó el Nombre del Usuario"
echo "Uso: $0 Nombre_Usuario"
exit 1
fi
```

Asegúrese de entender el funcionamiento y el objetivo de esta sentencia if-fi. Ejecute el script para probar si el manejo de los errores es correcto.

2. Este ejemplo está orientado a ayudarle a entender los pasos que involucran la creación de un script. El script será muy simple.

   Escriba un script del shell y llámelo mi_lista, éste script ejecutará el comando "ls -FC" en cualquier directorio que usted le pase como parámetro.

   Agregue éste script a su directorio mis-scripts y pruébelo.

   Mejore éste script para permitir que el comando ls tome más de un argumento (de hecho cualquier número de argumentos).

3. En éste ejemplo ponemos varias ideas de la administración de los scripts en uso. Debe mover los scripts de un directorio a otro.

   Copie todos los scripts desde el directorio mis-scripts a un directorio bin en su directorio home. Claro, si no está, creélo.

   Elimine el directorio mis-scripts, después de haber copiado todos los scripts de éste al directorio bin.

   Asegúrese de eliminar cualquier referencia al directorio mis-scripts en sus archivos ocultos de configuración .profile.

   Asegúrese de que el directorio bin en su directorio home está en su PATH de ejecución.

   Efectúe un log-out (salir) del sistema e ingrese de nuevo, para asegurarse de que los cambios en su archivo .profile están funcionando como es debido, y que aún usted puede ejecutar sus scripts.

4. Vamos a diseñar otro script. De nuevo empecemos con algo muy sencillo y entonces lo haremos más flexible agregandole algún tipo de prueba y entrada interactiva.

   Escriba un script para usar el comando cp -i para copiar un archivo usando los dos primeros argumentos como los archivos de origen y destino. Nombre éste script mi_copia en el directorio bin en su directorio home. Asegúrese de poder ejecutar éste comando utilizando el comando:

   ```
   $ mi_copia /usr/share/doc/Readme.txt Leeme.txt
   ```

   Ahora, mejoremos el script para que, si el usuario suple sólo el nombre del archivo de origen, entonces el comando cp dentro del script, o el script mismo, asumirá que su directorio actual es el directorio por defecto para copiar. Asegúrese de que el script trabaje de la siguiente manera:

   ```
   $ mi_copia /usr/share/doc/Readme.txt
   ```

   Mejoremos el script de nuevo, ahora para que pida que debe especificar el nombre del archivo en la línea de comandos. Si falta uno de los dos argumentos, se le pedirá al usuario que lo administre interactivamente. Que le pida ambos, el archivo de origen y el de destino, pero que le permita al usuario dejar en blan-

co el archivo de destino, lo cual causaría que se fuese por el directorio por defecto, en éste caso, el directorio actual. Revise por ocurrencias de errores obvios, como es que, el archivo de origen no exista o que no es un archivo. Al probarlo debe ser algo así:

$ mi_copia
Copia cual archivo? /usr/share/doc/Readme.txt
A donde?

Ahora vamos a mejorar el script rev_usuario que creamos en la primera pregunta para que use un bucle if para probar el éxito de las tuberías y dar como salida un mensaje de "Ingresó al Sistema" o "No Ingresó al Sistema", dependiendo del éxito o el fracaso del bucle.

6. Mejore ahora, de nuevo, el script rev_usuario de la pregunta anterior para que permita múltiples argumentos desde la línea de comandos y con el uso de bucle for pasará por cada argumento, revisando si cada usuario especificado ha ingresado (logged in) al sistema con éxito.

   Pruebe el comando de la siguiente manera:
   $ rev_usuario root nombre_usuario1 nombre_usuario2 ....

7- Mejore su archivo de configuración .profile guardando toda la salida de su comando mailx -H dentro de una variable. Entonces proceda a usar una sentencia de if y una de prueba para ver si la variable contiene información.

   Si la variable contiene información, entonces escriba el siguiente mensaje:
   He aquí los cabezales desde sus mensajes de correo:

   Aquí sigue la información que está almacenada en la variable.

   Si la variable no contiene ninguna información, escribirá el siguiente mensaje:
   Usted no tiene correo hoy, pruebe más tarde!

Ejercicio 8-6; Más Scripts del Shell

Las soluciones a éste ejercicio se encuentran en el Apéndice A.

1. Escriba un script que liste los archivos en el directorio actual, y que le indique si el archivo es un directorio, un archivo ejecutable, o si es un archivo normal. Dé salida a cada archivo en su propia línea, empezando con los siguientes simbolos especiales:

/	**Si es un Directorio**
*****	**Si es un Ejecutable**
-	**Si es un archivo Normal (todos los otros)**

   Use un bucle for y un comodín (wildcard) para pasar por la lista de archivos. Modifique el script para que despliegue el símbolo al final del nombre del archivo como lo hace el comando ls -F.

   Modifique el script para que tome un directorio como un parámetro y que use el directorio actual si ninguno es suplido como argumento en la línea de comando.

2. Escriba un script, llámelo agregar_path, script que agrega un único parámetro a la ruta (PATH) actual, e imprime a la consola el nuevo valor de la variable PATH. Asegúrese de que el script escribe un error a pantalla si, el script, se ejecuta con cualquier otra cosa que no sea un sólo parámetro. Ponga el script a prueba, recuerde que el script se ejecuta en un subshell, así que no modificará su shell actual.

   Crée un alias y llámelo path (minúscula), el cuál, se ejecutará en el ambiente del shell actual, usando el comando (.). Ponga su script a prueba para asegurarse de que la variable PATH ha sido extendida en su ambiente actual de trabajo. Escriba algo como lo que sigue a continuación:

```
$ path /usr/sbin
PATH=/bin:/usr/bin:/home/usuario/bin:.:/usr/sbin
$ echo $PATH
/bin:/usr/bin:/home/usuario/bin:.:/usr/sbin
```

Modifique su script para que acepte múltiples argumentos de directorio y que imprima el valor actual, si no se suple ningún argumento. Modifique su script para aceptar a -d como un parámetro. En éste caso que elimine el último directorio en la variable PATH.

Modifique su script para que acepte el parámetro -i. En éste caso, que imprima a pantalla cada componente de la ruta y que pida al usuario que escriba "d" para eliminar el directorio; y sino, el directorio es retenido. Reconstruya un nuevo PATH basado en las respuestas del usuario.

3. Defina una función del shell que localice un archivo en particular en un directorio elegido. Aunque no se ha cubierto profundamente el tema de las funciones, le damos un ejemplo para que usted sigua su sintaxis.

   Observe los espacios alrededor de la llaves ({}).

   ```
   $fn() { find $1 -name "$2" -print 2>/dev/null; }
   ```

   ¿Puede usted descifrar cómo trabaja esta función?

   El sistema posee un archivo debajo de la estructura de directorio /etc de nombre inittab. Ejecute su función para encontrarlo.

   Para ejecutar una función, escriba su nombre sin los paréntesis (seguido por ...?)

RESUMEN

En éste capítulo cubrimos varios tópicos relacionados con los shells de GNU/Linux:

- El shell bash es el shell por defecto de GNU/Linux:
  - Toma la entrada del usuario desde la línea de comandos.
  - Es un poderoso ambiente de programación.
    - if-then-elif-fi
    - for-in-do
    - test / [[ ]]
- Los scripts pueden simplificar tareas comunes.
  - Flujos Estándares de E/S (I/O):
    - stdin
    - stdout
    - stderr
  - Redirección de caracteres:
    > >> 2> 2>> 2>&1 <
  - Tuberías que redireccionan la salida de un comando a la entrada de otro.
- Los filtros son programas que leen desde la entrada estándar y escriben a la salida estándar:
  - sort
  - tr
  - uniq
  - cut
  - grep
- Se puede elaborar filtros que logren tareas complejas combinando pequeños bloques de comandos juntos.

PREGUNTAS POST - EXAMEN

Las respuestas a estas preguntas están en el Apéndice A

1. ¿Cuál comando usaría usted para traducir de mayúsculas a minúsculas?

2. ¿Cómo puede usted lograr que la salida estándar y la de error, ambas, sean enviadas a un archivo de nombre ls.salida?

3. Defina la primera línea de un script bash.

4. ¿Cómo podemos usar el comando w para imprimir los usuarios ingresados al sistema desde el dominio abc.com?

5. ¿Qué símbolo podemos usar en vez de test en un script de bash?

CAPITULO 9

ADMINISTRACIÓN DEL SISTEMA OPERATIVO GNU/LINUX

EDICIÓN DE ARCHIVOS DE TEXTOS

OBJETIVOS

Al completar este Capítulo, usted podrá realizar las siguientes tareas:

- Abrir, editar y guardar documentos de texto usando el editor Vi.
- Utilizar los siguientes comandos: i, ZZ, :w, :w!, :q!, dd, x, D, J

PREGUNTAS PRE-EXAMEN

Las repuestas se encuentran en el Apéndice A.

1-¿Qué es vi?

2.- ¿Qué es emacs?

3-¿Cómo puede ser grabado en vi, un archivo con un nombre diferente?

4-¿Puede usted verificar faltas ortográficas en Vi?

5-Usando el editor vi, nombre maneras diferentes de entrar en el modo de inserción de texto.

6-¿Cómo usted entra en modo de insertar en el editor pico?

INTRODUCCIÓN

En éste capítulo le introduciremos y exploraremos el editor vi en gran detalle. Cubriremos tópicos de la indole de cortar, copiar y eliminar dentro del editor. Definiremos muchos comandos y lo ilustraremos todo con ejemplos para así mejor demostrar su uso.

EL EDITOR VI

La elección de un editor de texto en el uso de sistemas UNiX es una desición crítica para los nuevos usuarios. El aprendizaje del uso de un editor puede presentar ciertas dificultadas a los nuevos usuarios de GNU/Linux.

El nombre del editor es pronunciado (vi en español y vi-ai en inglés). Este editor está disponible para todas las plataformas de sistemas operativos derivados de UNiX, y por ende, todas las distribuciones de GNU/Linux. Tarde o temprano tendrá que aprender vi en su carrera como administrador de sistemas UNiX. Es el editor preferido cuando el espacio en disco es critico.

En esta sección describimos el editor de pantalla estándar de GNU/Linux y describiremos los siguientes tópicos:

- Conceptos de vi
- Introducción al vi
- Alternativas al vi

Conceptos de vi

El editor vi creció como una mejora del editor de línea llamado ex, usado en los primeros años de UNiX. Aún siendo estudiante de grado, Bil Joy creó un front-end de edición visual para el editor ex que usaba las capidades completas de pantalla de las terminales ADM-3a. Estas terminales tenían cursores direccionales de pantalla. Como editor visual, utiliza toda la pantalla para desplegar texto.

Bill Joy, además, desarrolló la base para los interpretes term-cap para monitores, que permitió que otras terminales avanzadas aprovecharán las ventajas de un cursor direcionable en pantalla. Con esa capacidad, le fué facil perpetuar el esfuezo de haber creado el editor vi. Debido a éste esfuerzo el editor vi no tenía que ser reescrito para cada terminal individual.

Hay unas cuantas cosas que se deben saber del editor vi, antes de continuar.

1- Vi es un editor visual basado en el editor de línea ex.
2- La interfaz, por lo general, puede ser dificultosa para nuevos usuarios.
3- Lo mejor es su rapidez y su poderío.
4- Por lo general, los caracteres son interpretados como comandos (en el modo de comandos)
5- Algunos comandos permiten que texto sea insertado después de presionar ESC (modo insertar).
6- Algunos comandos permiten que se introduzca texto en la base de la pantalla hasta que se presione la tecla ENTER (búsqueda y comandos extendidos).
7- Vi no asume la presencia del cursor, pero las versiones modernas trabajan bien con éstas.
8- Existen muchas variantes de éste editor, pero, todas ellas contienen los comandos básicos de Vi.

El editor vi tiene algunas pequeñas desventajas que podemos citar:

- La tecla CAPSLOCK es su enemigo
- La asistencia del mouse no es universal
- Toma esfuerzo aprender a usar vi:
- Poco intuitivo y confuso de aprender para nuevos usuarios

• No se puede aprender todo en un día.

Una vez el usuario aprende, las ventajas del editor vi, se olvida de las desventaja que el manejo de éste le pueda ocasionar. Como es costumbre con las aplicaciones GNU/Linux, el poder de las aplicaciones supera con creces la complejidad de su aprendizaje:

• Los dedos nunca se retiran del area del teclado.
• No depende del Ratón.
• No depende de teclas de funciones programable.
• Rápido.
• Poderoso
• Completo de herramientas.
• Orientado al contenido
• Combinaciones de teclas aprendidas que son comunes en otras aplicaciones de GNU/Linux.
• Muchas características avanzadas, tales como, TAGS (no las cubrimos aquí).

Caracteres

El editor vi utiliza casi todas las letras del alfabeto en inglés como un comando dentro del editor. Pues un comando en letras minúscula, y en mayúsculas tiene diferente significado. Si acidentalmente, presiona la tecla CAPS LOCK, ésto puede ocasionarle problemas inesperados.

Lo Básico

Usted necesitará aprender en el modo básico la navegación así como insertar, suprimir, corregir, buscar y copiar texto en el editor vi. También necesitará aprender el significado de cada comando, aunque, no todo a la vez. Por suerte no necesita saber mucho para poder funcionar apropiadamente. Con cada comando adicional que aprenda, será recompensado con más capacidad de manejo de los arcivos de texto.

Introducción al vi

La forma más cómoda de trabajar con un texto (modificarlo), es mediante un editor. GNU/Linux nos provee del editor vi, aunque debes dedicarle tiempo en aprender su manejo, permite realizar operaciones que no todos los editores permiten. La forma de invocarlo es:

$ vi archivo

La mayoría de los comandos se dan pulsando las teclas indicadas sin que éstas aparezcan en la pantalla ni tampoco es necesario pulsar <ENTER> al final de ellos, sólo los comandos que comienzan con : , / y ? son mostrados en la última línea de la pantalla y requieren de la pulsación de <ENTER> para finalizar (éstos corresponden a los comandos del editor ex, en el cual se basa vi).

Antes de comenzar a describir los comandos se establecen las normas de la notación:

i texto <ESC>
(a) (b) (c)

(a) Corresponde al comando, en éste caso basta pulsar sólo la letra i
(b) Aquí debe ingresar el texto que desee, puede utilizar más de una línea pulsando <ENTER> al final de cada una
(c) Debe pulsar la tecla <ESC> (Escape)

c :	Un caracter cualquiera.
l :	Una letra del alfabeto inglés.
^X :	Pulsar las teclas <control> y X
caracter :	Un caracter cualquiera.
CARACTER :	Un caracter distinto de espacio.
palabra :	Una secuencia de letras y/o números.
PALABRA :	Una secuencia de caracteres incluyendo los espacios que siguen.

arch :	Algún archivo del disco (Existente o no).
patrón :	Secuencia de caracteres a utilizar en un patrón de búsqueda.
movimiento :	Algún comando de movimiento.

A continuación se describirán la mayoría de los comandos de vi (sólo se excluyen los más complicados), a muchos de éstos se les puede anteponer un número decimal que indica un factor de repetición del comando, es decir, si se escribe 20 antes del comando, éste se repite 20 veces. Los comandos que tienen esta capacidad serán señalados con una letra n en la columna izquierda, al no colocar el valor, el comando se ejecuta sólo una vez.

Los comandos básicos de vi son:

:wq	Salvar y Salir
:q!	Salir sin salvar:
a	Append: Modo insertar en la siguiente posición del cursor
i	Insert: Modo insertar sobre la posición del cursor
x	Borra caracter bajo el cursor
dw	Borra palabra
dd	Borra línea
u	Deshacer (Undo)
Ctrl-F	Pantalla sgte.
Ctrl-B	Pantalla anterior
O	Inserta línea en blanco

Secuencia de comandos básicos

El vi utiliza el editor ex cuando trabaja en modo de comando y siempre es la última línea de la pantalla, con el prompt: por donde se le indican los comandos al ex. Los comandos admiten repetición, que indicará cuántas veces se ejecutará el comando. Por ejemplo, dd borra una línea, pero 4dd borrará 4 líneas. La mayoría de los comandos no producen salida por display de la acción, pero si cuando la acción es finalizada, por ejemplo, cuando ponga 11dd (borrar 11 líneas), el vi le dirá '11 lines deleted'.

Movimientos dentro del archivo

Ud. debe estar en modo comando para moverse dentro del archivo. Generalmente las teclas de cursor funcionan bien (pueden no funcionar si Ud. está conectado a través de un emulador de terminal desde un PC por ejemplo, y no coinciden las variable TERM de UNiX con las que Ud. está, emulando, solución: cambie la variable TERM con -en csh- setenv TERM vt100 -en sh- TERM=vt100 export TERM y en el emulador, seleccione como terminal vt100). Si Ud. está en modo insertar y presiona las teclas de cursor le insertará el caracter que ellas representan. Presionando la tecla RETURN se moverá a la siguiente línea.

La tabla siguiente muestra los movimientos más comunes:

<Return>	Siguiente línea
j	Siguiente línea
k	Línea previa
l	Siguiente caracter
h	Caracter anterior
Ctrl-F	Pantalla siguiente
Ctrl-B	Pantalla anterior
Ctrl-D	Media pantalla siguiente
Ctrl-U	Media pantalla anterior
[[	Inicio documento
]]	Fin documento
nG	Ir a línea n
w	Una palabra a la derecha

b	Una palabra a la izquierda
{	Fin párrafo
}	Fin párrafo anterior
/	string Busca string

Para ir a la línea 10 introduzca 10G (si sólo introduce G se moverá hasta el final del archivo). Para averiguar en que línea está, presione Ctrl-G. Las palabras, sentencias, y párrafos tienen especial significado para el vi, y existen comandos para moverse a través de ellas. Una palabra es cualquier caracter delimitado por blancos o puntuación, y también cada símbolo de puntuación es una palabra. Sin embargo si Ud. utiliza la mayúcula del comando de movimiento se saltará la puntuación, por ejemplo: B es lo mismo que b y mueve el cursor una palabra hacia atrás, pero si hay un punto, b se parará aqui, pero B no.

Una sentecia es un string con un punto final y dos espacios en blanco. Con) y (Ud. se moverá hacia adelante una sentencia, o hacia atrás, respectivamente. Un párrafo es el que termina con dos Return.

Búsqueda de strings

Las búquedas se realizarán en modo comando y para iniciarlas, debe presionar /. Esto causará que el cursor baje a la línea inferior: indique el string a buscar finalizando con un Return. Si desea buscar en orden inverso presione ? en vez de /.

Modo Texto

Antes de ir al modo texto recordar que siempre se sale de él mediante la tecla de <ESC> escape. La tabla siguiente muestra los comando básicos para pasar a modo texto.

a	append, inserta después del caracter sobre el cual estamos
i	insert, antes de caracter sobre el que estamos
A	append, al final de la línea actual
I	insert, antes del 1º caracter diferente de espacio en la línea corriente
o	abre una línea en blanco debajo de la actual
O	ídem anterior, pero sobre la actual

Correcciones

La única forma de corregir un error en modo texto es hacer un backspace y reintroducir los caracteres. Las correcciones más complejas deben ser realizadas desde el modo comandos. La siguiente tabla muestra los comandos que pueden ser utilizados:

x	borra caracter
dw	borra palabra
dd	borra línea
r	reemplaza un caracter sobre el cursor
R	reemplaza un string de caracteres (sobreescribe)
cw	cambia una palabra
s	sustituye un caracter por un string
.	repite el último cambio

Un comando interesante es el punto (.) el cual repite el último cambio hecho en edición. Este puede ser utilizado para cambiar cada ocurrencia de un string con otro, por ejemplo: si introducimos el comando /feo para buscar el string "feo". Entonces entre el comando cw e introduzca "lindo" para reemplazar feo por lindo.

Presione "\" para aceptar el cambio. Para encontrar la siguiente ocurrencia introduzca el comando n, y luego, "." para repetir el último cambio.

Comandos ex

La flexibilidad del vi está en el editor ex. En particular operaciones de búsqueda global y reemplazo están soportadas por el ex. Siempre sabemos cuáles son los comandos ex porque aparecerán los ":" como prompt. Para pasar al modo comandos del ex Ud. debe introducir desde el modo comando del vi (no insertando) los dos puntos (:). En éste modo, el cursor bajará a la última línea del display y aceptará comandos en modo ex. El comando es ejecutado cuando Ud. presiona la tecla Return. Con Ctrl-C el comando es anulado.

La siguiente tabla muestra alguno de los comandos básico del ex:

:w	Escribe el archivo
:q	Sale sin guardar cambios
:e	nombre Edita archivo nombre
:sh	Ejecuta un shell sin salir del editor

Movimientos dentro del ex

Sobre el ex puede ejecutar comandos refiriéndose a las líneas del texto. Por ejemplo el punto (.) representa la línea actual, y el $ la última línea. Por lo tanto, se pueden combinar, tal como ".,$" lo cual significaría desde la línea actual hasta hasta la última línea en el archivo. Combinando éste tipo de especificaciones se pueden crear comandos tales como :.+1,$d para borrar cada línea, desde la actual hasta el fin del archivo. La palabra g puede ser utilizada para realizar el tratamiento como global.

Búsqueda y reemplazo global

El siguiente ejemplo es una búsqueda global con reemplazo para mostrar las capacidades del vi. Se desea buscar la cadena viejo y sustituirla por nuevo, desde el principio al final del texto:

:1,$s/viejo/nuevo/g

Se desea buscar en todo el archivo "feo" y reeemplazarlo por "lindo":

:g/feo/s//lindo/g

Si se desea utilizar caracteres especiales se debe emplear el \ (backslash). Por ejemplo, para reemplazar todos los $ en **:

:g/\$/s///g**

El return también puede ser especificado introduciendo 'ControlV'.

Puntos de Referencia

La siguiente tabla da una referencia rápida con lo que soporta el vi (generalmente el vi tiene los mismos comandos en diferentes sistemas):

Obs: Los comandos comienzan por : y terminan con return, preste atención a la tecla de Mayúsculas, porque los comandos difieren sin está apretada o no.

Para Empezar:

$ vi file	Edita file
$ vi -r file	Edita la última versión salvada de file (recupera)
$ vi +n file	Edita y pone el cursor en la línea n
$ vi + file	ídentico pero pone el cursor en la última línea
$ vi filex ... filey	Edita los archivos filex hasta filey, después de salvar file, presione n para el siguiente
$ vi + /cadena file	Edita y sitúa el cursor en la línea que contiene cadena

Salvar y Salir

ZZ o :wq o :x	Salvar y salir
:w file	Salva en file, si no existe file, salva en el actual
:w!	Salva y no controla la protección de escritura
n,mw file	Salva desde la línea n a la m en file
n,mw>> file	Añade desde n a m al final de file
:q	Sale (si hay cambios no se ejecuta el comando)
:q!	Sale (si hay cambios los descarta)

Q Permite pasar al ex (:vi retorna)

:e!	Reedita el archivo actual, descarta los cambios. Comandos de status
:.=	Imprime línea actual
:=	Imprime nº de líneas en el archivo
Ctrl-g	Imprime status del archivo
:l (ele)	Imprime los caracteres especiales de la línea actual Insertar

Insertar

a	Append
A	Append después del final de la línea
i	Inserta
I	Insert antes del cominezo de la línea
o	Inserta un línea nueva (abajo)
O	Inserta una línea nueva (arriba)
Ctrl-V char	Inserta el char (válido para insertar caracteres de control)
:r file	Lee a file e inserta después de la línea actual
:nr file	Idéntico al anterior pero inserta después de la línea n

Undoing/Deshacer

u	undo/deshace último comando
U	Restaura la línea al estado original
“np	Retrive el n-esimo elimina (hasta 9 max)
N	Repite el último / o ? (búsqueda)
N	Idéntico al anterior pero en orden inverso
,	Idéntico al anterior pero en orden inverso
.	Repite el último cambio de texto

Posicionamiento del cursor

{	Marca las secciones cuando está en la primera columna
[[	Hacia atrás y principo de sección
]]	Hacia adelante y principo de sección

Movimientos del cursor

Derecha	k
arriba	j
abajo	h
izquierda	l (ele) o <Space> derecha
w o W	siguiente palabra (Mayúscula ignora puntuación)
b o B	palabra previa

e o E	fin palabra
0 o | 1º	columna
$	último caracter en la línea
+ o	Return 1º caracter de la línea
- 1º	caracter no blanco de la línea
G	última línea
G$	último caracter
nG	Linea n
(	Comienzo de sentencia
)	Cominezo de sgte sentencia
{	Comienzo de párrafo
}	Comienzo de párrafo sgte

Borrar

<- o Ctrl h	borra caracter en modo insertar
Ctrl w	borra palabra en modo insertar
Crtl x	borra texto insertado en modo insertar
nx	borra n caracteres incluyendo el de la posición del cursor
nX	borra n caracteres previos al cursor incluido
d	borra linea actual
D	borra desde posición actual al fin de línea
ndw	borra n palabras

Búsqueda

%	busca el comienzo de () [] {}
fchar	busca hacia adelante char
Fchar	busca hacia atrás char
tchar	busca hacia adelante linea actual
Tchar	busca hacia atrás linea actual
/str	busca string
?str	busca string hacia atrás
:set ic	ignorar caso
:set noic	no ignorar caso

Copiar

nyy o nY	copia n líneas en el buffer desde la actual
p	imprime las líneas de buffer en la posición del cursor
P	Idéntico, pero antes del cursor

El Editor pico

Un editor de texto es un programa que permite escribir y modificar archivos digitales compuestos, únicamente, por texto sin formato, conocidos comúnmente, como archivos de texto. Se distinguen de los procesadores de texto, en que se usan para escribir sólo texto, sin formato y sin imágenes.

Hay una gran variedad de editores de texto. Algunos son de uso general, mientras, que, otros están diseñados para escribir o programar en un lenguaje. Algunos son muy sencillos, mientras que, otros tienen implementadas gran cantidad de funciones.

Sintaxis: pico [opciones] [fichero]

Algunas opciones:

-w :	deshabilita el picado automático de las líneas.
-m :	habilita la funcionalidad del mouse en entornos gráficos, para posicionarse en el texto.
+<n> :	se establece el comienzo de la edición en la línea n-ésima.

Algunas opciones durante la edición:

Ctrl k :	borra y copia la línea actual en el clipboard.
Ctrl u :	inserta el contenido del clipboard en el texto.
Ctrl o :	salva el texto.
Ctrl x :	sale del editor y pregunta si salvar primero en caso de haberse modificado el texto.

El Editor Emacs

Historia de editor Emacs

En 1974 en el MIT, Richard Stallman modificó TECO, un editor de texto del laboratorio de IA al que se le añadieron diferentes macros hasta 1976, cuando escribió la primera versión de Emacs (Editor Macros).

En 1978 uno de los nuevos editores surgidos de Emacs, MulticsEmacs, fué escrito en MacLisp, una versión del lenguaje de programación Lisp. Gracias a su extensibilidad se ha mantenido en los siguientes editores Emacs hasta hoy.

En 1981 Emacs funciona en los sistemas operativos Unix.

A finales de 1983 una compañía exíge sus derechos sobre parte del código escrito por ellos hacia 1980. Stallman abandona el Emacs original y empieza GNU/Emacs, ahora con la licencia GPL. Es el primer programa del proyecto GNU para crear un sistema operativo libre.

En 1991, surge XEmacs como programa surgido de versiones antiguas de GNU/Emacs e independientes desde entonces de éste.

Introducción a Emacs

¿Qué es Emacs?

Emacs es un poderoso editor de texto. La versión actual, es distribuida por GNU, y es una versión mejorada de la original escrita por Richard Stallman.

¿Qué puedo hacer en Emacs?

Con Emacs se puede hacer casi todo lo referente a un editor de texto tradicional. A la vez permite editar una enorme cantidad de formatos de lenguajes tradicionales tales como “C”, “C++” o lenguaje típicos de PC como Pascal, etc. Emacs a su vez permite interactuar directamente con el Shell (aka bash).

¿Cómo puedo invocar éste editor?

Desde la línea de comandos o shell, simplemente invoque al programa “emacs”, ej:

$ emacs <ENTER>

¿Cuántos formatos o extensiones acepta éste editor?

Como ya decíamos, Emacs permite editar una extensa cantidad de tipos de lenguajes y derivados. A su vez puede actuar como un poderoso editor de Texto combinado con herramientas tales como TeX o sus deri-

vados. Entre las extensiones de Lenguajes conocidas, por listar algunas, mencionaremos las siguientes:

1. C++, C
2. Prolog , Lisp, Perl
3. Pascal Standard, Pascal DOS
4. Emacs Lisp, Common Lisp
5. Cobol, Fortran
6. HTML, TeX

Y así muchos más. Para los "gurus" de Lisp o Emacs-Lisp pueden Uds. realizar sus propias librerías que acepten nuevos formatos.

Manejo con el Editor

Edición de un archivo

Para comenzar a editar un archivo existen dos maneras de hacerlo: La primera es invocarlo directamente desde el Shell como antes lo habíamos explicado. Una vez dentro del editor, presionar las teclas Cx C-f (desde ahora en adelante nos referiremos de esta manera para expresarnos sobre la tecla Control-algo). Esto nos producirá una línea de comandos en el mismo editor, el cual, nos pedirá el archivo que deseamos editar. La segunda manera de editar un archivo será similar a la anterior, la diferencia será, que esta vez en nuestra línea de comandos escribimos lo siguiente:

$ emacs archivo_a_editar <ENTER>

Esto nos ubicará automáticamente en el archivo que deseamos editar.

Observación: Si el archivo a editar no existe, éste se creará automáticamente cuando hagamos el llamado.

Para salvar sin salir sólo presionamos las teclas C-x C-s.

Observación: En algunas terminales al presionar C-s éste se bloquea, por lo tanto, para desbloquearlo sólo presionamos C-q. Si ésto llegara a suceder, entonces para realizar la misma operación, presionamos M-x save-buffer ("M-x" se refiere a la tecla ESC).

Para salvar y salir sólo presionamos las teclas C-x C-c. Esto nos arrojará una sublínea de comandos, la cual, nos pedirá una opción similar a la siguiente:

Save file/home/usuario/text? (y, n, !, ., q, C-r or C-h)

Aquí se presiona la tecla "y" para salvar.

Inserción de un archivo

Para insertar un archivo, nos ubicamos en el texto en la posición donde deseamos insertar y presionamos las teclas C-xi; el mensaje será similar al siguiente: **Insert file: ~/**

Aquí se ingresa el archivo que deseamos insertar.

Operación con bloques de Texto

Mover Copiar

1. Ubicarse al comienzo del párrafo
2. Se presiona M-@
3. Vamos al final del párrafo y se presiona C-w
4. Se ubica el cursor en la posición final
5. Luego se presiona C-y

1. Ubicarse al comienzo del párrafo
2. C-space o C-shift-2 C-y
3. Vamos al final del párrafo y se presiona C-w
4. Se ubica el cursor en la posición final
5. Luego se presiona C-y

Para recuperarlo las veces que deseemos sólo presionamos C-y.

Creación de Ventanas en Emacs

Emacs soporta la creación de diferentes ventanas en su ambiente de trabajo, para crearlas sólo presionamos C-xn, donde "n", es el número de ventanas que deseamos crear. Ahora bien, para movernos a través de las ventanas sólo presionamos C-o.

Comandos de Movimientos

Para moverse dentro del editor simplemente usamos las teclas de movimiento o cursoras. Ahora bien, como existen demasiados, éste método quizás resulte poco eficiente, por lo tanto, he acá una guía útil de sistema de movimiento:

1. **C-v** avanzar una página
2. **M-v** retroceder una página

C-p línea previa
:
C-b caracter atrás **. .** **. . C-f** caracter adelante
:
C-n línea siguiente

3. **M-f** palabra atrás
4. **C-a** comienzo de línea
5. **C-e** final de línea
6. **M-<** comienzo del archivo
7. **M->** final del archivo

RESUMEN

En éste capítulo, usted fué introducido al editor vi, sus fortalezas y debilidades, y a algunas alternativas a éste editor disponibles en sistemas GNU/Linux, y otras alternativas UNiX. Algunas cosas discutidas fueron:

- vi es un editor de pantalla.
- vi usa letras como comandos para moverse dentro del archivo que se está editando.
- Algunos comandos permiten el ingreso de texto hasta el momento que se presione la tecla ESC.
- Podemos editar líneas usando la tecla BACKSPACE pero sólo hasta el principio de la línea.
- Los comandos extendidos (:) proveen acceso al editor de línea ed.
- El editor pico permite edición simple para los usuarios introduciéndose a GNU/Linux.
- emacs permite la edición potente dentro de un ambiente autocontenido.

PREGUNTAS POST - EXAMEN

Las respuestas a estas preguntas están en el Apéndice A

1.- Describa la función del comando view archivo.txt

2.- ¿Cúal secuencia de comandos puede ser usada para desplegar todas las líneas que empiezan con número en el editor vi?

3.- ¿Cómo puede usted ingresar el contenido del archivo /etc/passwd en el archivo actual que esta editando dentro de una sessión de vi?

4.- ¿Cómo puede usted insertar la fecha y hora en la sessión actual de vi?

5.- ¿Cómo encuentra usted como ejecutar un comando dentro del editor pico?

APENDICE A

ADMINISTRACIÓN DEL SISTEMA OPERATIVO GNU/LINUX

CAPÍTULO 1

PREGUNTAS PRE-EXAMEN

1. ¿Qué es una distribución?

   Una colleción de un Kernel de Linux, utilitarios, scripts de instalación, documentación y códigos fuente. Algunos paquetes y utilitarios comerciales y propietarios también pueden ser incluídos en algunas versiones de vendedores oficiales.

2. ¿Liste 4 distribuciones populares de GNU/Linux?

   RedHat, Fedora, Debian, SlackWare, SuSE, Mandriva, TurboLinux, etc.

3. ¿Cómo es GNU/Linux Distribuido?

   Típicamente por el Internet; personas pueden instalar directamente desde el Internet, descargar imagenes ISO para quemar en CD, comprar una preempaquetada, en fin, existen muchos medios por lo cual obtener una copia de su distro favorita de GNU/Linux.

4. ¿Cómo es GNU/Linux Licenciado?

   Esta licenciado bajo GPL (Licencia Pública General).

5. ¿Qué es Software Open Source?

   El Softwaare que pone libremente su código fuente a disposición del usuario, casi siempre permitiendo su modificación.

6. ¿Qué es Software Libre?

   Software que conforma a las cuatras directrices que hacen la ley GPL.

PREGUNTAS POST- EXAMEN

1. ¿Qué define el GUI de GNU/Linux?

   Cualquier servidor ejecutando el protocolo X, a veces referido como el X o el Sistema X Window.

2. ¿Cuántas distribuciones existen de GNU/Linux?

   Existen más de 1,600 distribuciones de GNU/Linux. Un buen sitio para conocer muchas por nombre son las páginas Web de http://linuxiso.org y http://distrowatch.org

3. ¿Puedo usar una copia de una Distro de GNU/Linux para instalar en multiples equipos?

   Si, ya que la gran mayoría de aplicaciones y utilitarios que conforman un distro GNU/Linux están regidos por la licencia GPL; su duplicación y uso estean libre de costo de licencia.

4. ¿Cómo puede el usuario interactúar (los interfaces) con GNU/Linux?

O desde un CLI, un front end basado en Menu o un GUI.

5. Liste 4 implementaciones de GNU/Linux en diferentes arquitecturas.

INTEL, SPARC, PowerPC, ALPHA, IBM RS/6000, Motorolla 68000; estas no son todas...

6. ¿Por qué utilizaría GNU/Linux en vez de un sistema operativo propietario?

Hay muchas razones para elegir GNU/Linux...aqui mencionamos algunas:
- Costo
- Compatibilidad
- Flexibilidad
- Soporte
- Rendimiento
- Manejabilidad

7. ¿Por qué utilizaría un sistema operativo propietario en vez de GNU/Linux?

Nosotros somo de crencia que los administradores que elijen no usar GNU/Linux lo hacen por desconocimiento o la teoría de FUD (Siglas en inglés, Fear, Uncertainty y Doubt) que significan Miedo, Incertidumbre y Duda. Estas contribuyen a:
- Falta de conocimiento
- Curva de Aprendizaje tradicional de UNiX
- Paquetes de software no disponible aún en GNU/Linux
- La idea erronéa que todo lo LIBRE no es bueno

CAPÍTULO 2

PREGUNTAS PRE-EXAMEN

1.- Listar los pasos comunes a la instalación de todas las distribuciones

Asignación del Disco, que máquina por lo menos arranque de un disquete, Particionar los discos duro y copiar todo el sistema GNU/Linux.

2.- Listar la información básica del hardware que debes tener antes de comenzar a instalar

CPU, RAM, Monitor, Tarjeta Vídeo, RAM vídeo, tipo Mouse, teclado (idioma y tipo ej. QWERTY), tipo Disco Duro (IDE, SCSI, etc.), y establecidos en que controladora (Esclavo primario, etc.), puertos COM, NICS de redes, Modems, tarjetas Sonido, etc.

3.- ¿En cuál partición se coloca el Kernel de GNU/Linux y archivos esenciales?

En la Partición "/"

4.- ¿Puedes tener además de GNU/Linux, otro sistema Operativo en un mismo computador?

Si. Si tiene más de una partición en su disco o más de un disco duro, puede instalar más de un sistema operativo.

5.- Liste las diferente media de la cual puedes instalar GNU/Linux

CD-ROM, HTTP, Disco Duro y Red.

PREGUNTAS POST-EXAMEN

1.- ¿Donde debe ser instalado el directorio /boot en el disco duro?

El directorio /boot debe residir debajo del cilindro 1024.

2.- ¿Qué herramientas están disponibles para examinar el particionado de un disco?

Las herramientas son fdisk, cfdisk y Disk Druid. Qparted, Partition Magic y FIPS son para permitir particionar y redimensionar particiones ya existentes sin perdida de data.

3.- ¿Por qué no debe usted trabajar desde la cuenta de root/superuser para todas sus tareas?

Porque el root tienes poderes absolutos a todas las areas del sistema, y puede causar daños irreparables si ejecuta los comandos equivocados.

4.- ¿Cuál es la función del sistema de administración de paquetes?

Los administradores de paquetes hacen del proceso de instalación de software y su mantenimiento una tarea más manejable.

5.- ¿Donde es posible encontrar errores durante la instalación?

Todo los errores del kernel son escritos al stderr, el cual es donde sus errores del terminal y todos los otros errores son almacenados debajo del directorio /var/log/.

CAPÍTULO 3

No se presentan preguntas en este modulo.

CAPÍTULO 4

PREGUNTAS PRE-EXAMEN

1.- ¿Cuál es el comando genérico de iniciar el sistema X?
startx

2.- Cuando hay un problema o duda del administrador de ventanas, es importante revisar ¿cuál recurso de información para encontrar una solución?

Revisar las páginas man.

3.- ¿En caso de emergencia, ¿Qué combinación de teclas puede ser utilizada para eliminar (kill) el servidor del X?
CTRL+ALT+BACKSPACE

Ejercicios 4-1: El Uso del XF86Setup

En éste ejercicio practicaremos usando la herramienta de configuración gráfica del Servidor X, XF86Setup. Aunque como cada equipo es diferente podemos simular un equipo en particular y camniar de principio a fin por los pasos requeridos. Pero recuerde no salva esta configuración de práctica ya que lo más seguro que no trabajará en su sistema. No se proveen soluciones a éste ejercicio.

1.- Inicie el Sistema X Window, si es que ya no se encuentra ejecutando.

2.- Inicie un shell prompt en un xterm.

3.- Ya en el command prompt, inicie el programa XF86Setup así:
 # XF86Setup
4.- Establesca que el mouse es un Logitech en el COM1 puerto serial Ms-DOS.
5.- Configure la tarjeta de video como una Matrox Millennium II con 16 MB de RAM de video.
6.- Configure el monitor como uno de alta frecuencia tipo SVGA (1,024 x 768 de 70 Hz).
7.- Establezca el Servidor de X se inicie con una resolución de 1,024 x 768 con 24-bit de color.
8.- Salga sin salvar sus cambios.

Nota: Puede salvar sus cambio para practicar con tan sólo salvar el archivo XF86Config-4 antes de empezar el proceso. Si no sale todo bien podrás siempre regresar a éste posteriormente, así no afectando su sistema en caso de fracasos.

Ejercicio 4-2: Múltiple Administradores de Ventanas

Este ejercicio explicatorio es para familiarizarlo con las diversas maneras que más de un administrador de ventana pueden co-existir en un mismo equipo ejecutando GNU/Linux. Las preguntas no tienen preguntas correctas ni incorrectas. Respuesta a éste ejercicio estan en el Apéndice A.

1.- Sin iniciar el X, intente determinar cual administrador de ventana estan disponible en su sistema. Explore el arbol de directorio debajo de /etc/X11 y tome apunte de lo que encuentre. Haga lo mismo al directorio /usr/share.

Casi todo lo que pasa en el X se inicia en el directorio /etc/X11. Heche un vistado en el directorio de /etc/X11/xinit y vera dos archivos llamados xinitrc y Xclients. estos archivos controlan el inicio para todos los usuarios en el sistema. basicamen te hacen la misma cosa; le dicen al servidor X que hacer inmediantemente despues que se cargan. el sistema puede ser configurado para copiar estos archivos en su directorio home como archivos ocultos. Estos seran ejecutados a preferencia de los archivos generales en el directorio /etc/X11/xinit.

La mayoria de los sistemas usan esta logica para autoiniciar los programas X:

a. Revise a ver si tiene un archivo oculto en su directorio home llamado xinitrc. Si es asi use el comando exec para transferir a este la ejecucion del incio del X.
b. Revise a ver si tiene un archivo oculto en su directorio home llamado Xclient. Si es asi use el comando exec para transferir a este la ejecucion del incio del X.
c. Revise a ver si tiene un archivo oculto en el directorio /etc/X11/xinit llamado xinitrc. Si es asi use el comando exec para transferir a este la ejecucion del incio del X.
d. Revise a ver si tiene un archivo oculto en el directorio /etc/X11/xinit llamado Xclient. Si es asi use el comando exec para transferir a este la ejecucion del incio del X.

Si ninguno de estos archivos esta presente el X se iniciara en modo por defecto, nada pero la ventana de root con el cursor del mouse y un terminal de texto X. Ningun manejador de ventana estara ejecutando. Asi que, en algunos archivos en los puntos a, b, c o d que da inicio al windows manager. Usted debe dar un vistaso y dar seguimiento a la logica del script que da inicio al windows manager.

Mucho manejadores de ventanas mantienen sus archivos de configuracion en un subdirectorio debajo de /usr/share. Revise sus sitema por los siguientes subdirectorios:

/usr/share/enligment
/usr/share/gnome
/usr/share/themes
/usr/share/apps
/usr/share/apps/switchdesk

2.- Ejecute el X y determine cual administrador de ventana se esta ejecutando.

El X utiliza la consola virtual numero 7. Una vez el X es iniciado usted puede trasladarse a una consola de testo presionando las teclas CTRL+ ALT+Fx, donde la x es 1,2...6 y representan una consola virtual de GNU/linux. Para retornar al X presione CTRL+ ALT+F7.

Desde el X, dirijase a la consola 3 presionando CTRL+ ALT+F3. Ingrese como root y despliegue la lista de los procesos con el siguiente comando:

$ ps eax | less

Ahora analice la lista de los procesos y usted anda buscando uno de los adminsitradores de ventana:

kwm	El adminsitrador de ventana nativo de kde
afterstep	El adminsitrador de ventana nativo de AfterStep
wmaker	El adminsitrador de ventana nativo de Window Maker

De nuevo, regrese al X, presionando CTRL+ ALT+F7.

3.- Si usted ingresa al sistema via un administrador de pantalla (o sea si usted ingresa desde un prompt gráfico, con su nombre y contraseña), determine cual administrador de pantalla se está usando.

Parecido al paso dos, busque en la lista de procesos por xdm (el display manager por defecto del XFree86), kdm (KDE), gdm (GNOME) etc..

4.- Fíjese si la herramienta switchdesk está disponible en su sistema.

$which switchdesk

5.- Use switchdesk para cambiar su administrrador de ventanas.

Para cambiar a kde:

$ switchdesk KDE

Para cambiar a gnome:

$ switchdesk GNOME

Ejercicio 4-3: Uso de aplicaciones de ventanas en KDE y su Comportamiento

Se asume que KDE ya está instalado en su sistema y que el comando switchdesk está disponible. Es preferible que éste ejecutando una de las distribuciones reciente de GNU/Linux from Red Hat, SuSE, or Turbolinux. No se proveen soluciones para este ejercicio.

PREGUNTAS POST-EXAMEN

1.- ¿Cuál es la diferencia entre un administrador de ventanas y un ambiente de escritorio?

El trabajo de un administrador de ventana es el colocamiento, movimiento, decoracion y redemencionar las ventanas en la pantalla. Un ambiente de escritorio es un conjunto integrados de programas que presenta una apariencia y comportamiemto comun. Un ambiente de escritorio puede incluir un admisntrador de ventana.

2.- ¿Cuáles dos programas pueden ser utilizados para permitir, restringir o denegar acceso al Servidor X?

Cuál es la diferencia entre ambos?

xhost	Asinga los derechos de accesos por host
xauth	Asigna los derechos solo a esos que tienen una copia del token que servidor X genero al incio

3.- Su superior le ha requerido efectuar cambios en las preferencias de sus recursos personales. Primero,

a.- ¿En cuál archivo es que se almacenan estas preferencias en el directorio home de su superior?

.xinitrc

b.- ¿Cuál comando se usa para listar éstos valores ajustados?

cat .xinitrc

4.- Liste el orden de éstos componentes del X desde el más bajo nivel hacia el más alto:

Motif, protocolo X, Xt, Xlib.
Protocolo X, Xlib, Xt y Motif

5.- ¿Cuál es el nombre del archivo que el Servidor Xfree86 X lee para determinar su configuración?

XF86Config; normalmente se encuentra en el directorio /etc/X11 o /etc.

6.- Usted desea desplegar la salida del programa xeyes al servidor X de un equipo de nombre Ivelis.abiertos.org, cual se encuentra ejecutando sólo un servidor X con una sola pantalla. Asumiendo que usted ya ha configurado el servidor para permitir acceso remoto. ¿Cómo puede hacer que el programa se despliegue en éste otro equipo?
$ xeyes -display Ivelis.abiertos.org:0.0

CAPÍTULO 5

PREGUNTAS PRE-EXAMEN

1.- ¿Qué programa utilitario de GNU/Linux carga el kernel de una PowerPC?

Yaboot, GRUB, entre otros

2.- ¿Cuál es el propósito de tener varios runlevels?

Existe un runlevel que corresponde a cada uno de los siguientes: reiniciar el sistema, apagar el sistema, iniciar en modo monousuario, iniciar en modo multiusuario con o sin servicios de redes.

3.- ¿Porqué quieres usar el comando shutdown en vez de sólo presionar el botón de encendido?

Usar el comando shutdown le envia un mensaje a los usuarios de que el sistema se va a apagar, deben salvar cualquier archivo que este abierto. Apagar el sistemade manera inapropiada puede corromper la data en los discos.

4.- Describir la función del comando kill.

Enviar señales a los programas que se encuentran en ejecución. La señal enviada por defecto es la TERM que termina los procesos.

Ejercicio 5-1: Uso de las páginas del manual.

Jamás debe subestimar lo útil que son las páginas del man. Aprendan a usarlas. Si el lenguaje y la terminologías de las páginas del manual le parecen temerosas, ejecute la sentencia man a un comando que conozca, por ejemplo, a echo o ls. Esto le asistirá a reconocer términos comunes y expresiones usadas.

1.- Use el comando man para desplegar información sobre el comando passwd, etc. Fíjese que éste muestra información sobre el comando passwd, no el archivo /etc/passwd.
$man passwd

Modifique la sentencia anterior para que se despliegue la descripción del archivo /etc/passwd y no la del comando passwd.
$man 5 passwd

2.- Encuentra cuál paginador esta siendo usado por el comando man y modifique su entorno para utilizar otro. (Por ejemplo, si estas usando el comando less, cambie a usar more y viceversa). Podrías necesitar leer las páginas del manual del comando man.

Las paginas man son encontrada por una variable de ambiente:

- MANPAGER, si esta establecido como el programa para desplegar las paginas man. Si no esta establecido entonces PAGE es usado. Si este no tiene valor entonces se usa /usr/bin/less
- Para cambiar cual paginador esta en uso, modifique la variable de ambiente
  $ export MANPAGER="less" o $ export MANPAGER="more"
- También usted puede colocar estas directrices en el archivo .bash_profile o el .bashrc.

3.- Encuentre cuál comando de la sección 1 tiene que ver con editar archivo de texto. Recuerde que deberá utilizar una combinación de apropos y el comando grep (para así poder filtrar las líneas de la sección 1 solamente).

El siguiente comando es una de las posibles respuestas :
$apropos edit | grep "(1)"

Ejercicio 5-2: Señales

Este ejercicio es sólo una práctica y no provee solución para éste ejercicio.

1.- Inicie algunos programas (por ejemplo, cat) y utilice CONTROL+Z, CONTROL+C y CONTROL+\ para ver que pasa. Sobre CONTROL+Z, utilice el comando fg, entonces CONTROL+Z de nuevo y utilice el comando bg. Imprima un listado con ps u después de cada uno. Entonces utilice el comando kill para matarlo. Asegúrese buscar el archivo de core de la memoria después de los CONTROL+\.

Ejercicio 5-3: Crear un Disquete de Emergencia

No se proveen soluciones para éste ejercicio.

En las mayoría de las computadoras en sus BIOS uno puede indicarle que en caso de encontrar la presencia de un disco de inicio, y si el disquete de inicio existe y es booteable, que el computador se inicia desde éste disquete. De esta forma, puedes recuperar el control de un sistema que se niega iniciar o si has perdido el password de root. Esta es una de la razones por la que hay que asegurar que el computador físico no esta abiertamente disponible para todos, ya que si su computador esta iniciable desde un disquete cualquiera puede tener acceso a sus recursos.

Cuando primero instalaste GNU/Linux, lo más probable que el sistema le ofreció la oportunidad de crear un disco de emergencia de inicio. Si no posees ese disco (porque obvió crearlo o se extravió), puedes crear uno desde un sistema que aún está trabajando. Los procedimientos específicos varián en algunas distribuciones; Usaremos en éste capítulo el proceso para un sistema RedHat.

El procedimiento utilizará el utilitario mkbootdisk. Lea la página del manual para esta utilidad. Las siguientes instrucciones interpretan algunas de las opciones en la página del manual. Conocimientos de otras opciones vendrán con la práctica y el estudio.

1.- Inicie el sistema normalmente. Es una buena idea crear el disco de emergencia mientras esta instalando el sistema ya que si pierde la capacidad de inicio del sistema, deberá iniciar desde el CD o ún Disco de Inicio.

2.- Entra en el sistema como root, login.

3.- Determine la versión del kernel. Multiples kernels puede que estén instalados y LILO se configura para seleccionar entre ellos. Un disco inicio puede ser creado para cada uno de ellos. La versión del kernel es el nombre del directorio /lib/modules.

Ejemplo:
cris@crisdebian:~$ ls /lib/modules/
2.6.7-1-386

4.- Si aparece más de uno, puedes determinar cual esta actualmente cargado con el comando uname (es actualmente la manera más apropiada para determinar el número de versión).
cris@crisdebian:~$ uname -r
2.6.7-1-386

5.- Ejecute el comando mkbootdisk dandóle el nombre del kernel a ser colocado en el disquete como argumento:
cris@crisdebian:~$ mkbootdisk 2.6.7-1-386

Se le indicará que inserte un disquete y presiones ENTER cuando éste listo. Toda la información almacenada en el disquete será reemplazada, cuando retorne el prompt el proceso habrá concluido.

6.- Retire el disquete y coloquele la banda protectora contra escritura. Esto protege la escritura accidental en el disco. Puedes montar el disquete y examinarlo pero asegurate de desmontarlo antes de sacarlo de la disquetera.

7.- Inicia desde el disco de emergencia creado. Para hacer esto, reinicie o mejor aún apague el sistema, inserte el disquete antes que el sistema escoja su modo de inicio.

Ejemplos:

Reinicie el sistema así:
crisdebian:/home/cris# shutdown -r now

Apague el sistema así:
crisdebian:/home/cris# shutdown -h now

8.- Con algunos tipo de instalación, iniciar desde el disquete, es la manera normal de iniciar el sistema.

PREGUNTAS POST-EXAMEN

1.- ¿Qué significa la cuenta privilegiada de root?

La cuenta de root tiene acceso sin restriccion a todas las funciones del sistema, incluyendo la de agregar, cambiar y remover cuentas de usuario. Leer, modificar y eliminar archivos del sistema

2.- ¿Cómo puede un usuario cambiar su EUID ?

Usar el comando su.

¿Cómo afecta esto al usuario?

El usuario adquirira los privilegios del nuevo usuario perdiendo sus privilegios hasta que ejecute el comando exit.

3.- ¿Porqué no permanecer como root en el sistema todo el tiempo?

El uso casual de la cuenta de root es extremadamente peligrosa ya que esta cuenta tiene acceso sin restriccion a las funciones del sistema y hasta un error tipografico puede causar gran destruccion masiva de data del sistema. Mantenerse ingresado en la cuenta de root todo el tiempo es un riesgo mayor de seguridad; cualquier con acceso fisico al sistema puede adquirir acceso al sistema sin restriccion.

4.- ¿Dónde puedo encontrar los archivos log del sistema ?

Los subdirectorios de /var/log

5.- ¿Cuál es una de las razones para crear un disco de emergencia ?

En caso de que su gestor de arranque se dañe o sea removido, si instala un kernel nuevo que no funciona correctamente o si su kernel es removido por error.

CAPÍTULO 6

PREGUNTAS PRE-EXAMEN

1. Debe remover una cuenta de un usuario que ya no trabaja en la compañía. ¿Qué comando usaría usted para remover la cuenta, si esta cuenta tiene su directorio home en /home/usuario? ¿Cómo verificaría usted que fué realmente eliminada?
rm -r /home/usuario

Para verificar esto utilice el comando ls para verificar que ya no existe.

2. ¿Es el directorio /proc un sistema de archivo virtual o un sistema de archivo físico?

El directorio proc es un sistema de archivos virtual el cual no es fisico. Asi que no ocupa ningun espacio en disco. Cualquier intento de escribir a este fracasara.

3. ¿Qué es un nombre de ruta relativo?

Aquel que no empieza por la /

Ejercicios 6-1: El Sistema de Archivos

La mayoria de estas preguntas en esta seccion solamente requieren que usted instrodusca los comandos y observe el resultado, esas preguntas no tienen soluciones formales pero proveen un texto explicatorio.

1. Las siguientes rutas relativas pertenecen a:

La mayoria de respuestas en esta seccion simplemente requieren que usted ingrese los comandos y observer los resultados. No se ofrece una solucion formal a estas preguntas.

2. Los siguientes pasos le ayudarán a entender la diferencia entre nombres de rutas relativas y absolutas (relativas son nombre de rutas dadas desde una posición dentro de la jerarquía de archivos. Las absolutas son nombre de rutas dados desde el directorio raíz del sistema de archivos; en pocas palabras que empiezan con una /).

a. Asumiendo que los siguientes directorios existen:
/home/usuario/trabajo/cartas/nuevas
/home/usuario/trabajo/cartas/viejas
/home/usuario/trabajo/oficios

¿Puede usted dibujar la estructura de árbol?. La respuesta es un dibujo de un arbol la cual no se provee.

Si su directorio actual es:
/home/usuario/trabajo/carta/nuevas

b. Escriba las rutas relativas a éstos directorios: /home/usuario/trabajo/cartas/viejas
/home/usuario/trabajo/cartas/viejas ./

/home/usuario/trabajo/oficios	**../../oficios**
/home/usuario/trabajo	**../../**
¿Cuál es la ruta relativa a su directorio home?	**../../../**

3. Ahora vamos a comparar los dos paginadores de GNU/Linux: less y more.

Busque el el directorio /etc/X11/ y encuentre el archivo de configuración del X. Despliegue primero co more y luego con less. Una vez éste dentro del paginador utilice la opción h (help) para ver que puede hacer y familiarizarse con los paginadores.

No se proveen respuestas para este ejercicio.

4. Ahora vamos a explorar los atributos de los archivos que copiamos, movemos, etc. También prestaremos un poco de atención a los permisos de acceso a los archivos, aunque no se cubre en éste capítulo. Sólo tome nota que son consultados al usar el comando rm.

Copie el archivo /etc/passwd a su directorio home, dejándole el mismo nombre de passwd.
cp /etc/passwd passwd

Ejecute un lista detallada de ambos archivos, el original /etc/passwd y la copia que acaba de crear.

Observe los atributos: ¿cuáles atributos son retenidos de los del archivo original y cuáles cambiaron?
ls -l /etc/passwd passwd

Renombre el archivo nuevo que creo, passwd, en su directorio home y llamélo contraseña.
$ mv passwd pass1

Ahora deberá crear un un archivo nuevo y conviertalo de sólo lectura:
$ touch archivo1; chmod -w archivo1

Copie el archivo y observe los permisos de ambos archivo:
$ cp archivo1 archivo2
$ ls -l archivo?

Note la falta de los permisos de escritura (w). Elimine el archivo con:
$ rm file2

Como el archivo retuvo la misma protección contra escritura como la del archivo original, se le pedirá que confirme que desea remover el archivo.

5. Ahora practicaremos crear y eliminar directorios.

Trasládese a su directorio home para escribir los siguientes comandos. Asegúrese de su ubicación en el sistema de archivos, antes y después de ejecutar cada uno de los siguientes comandos.

```
$ cd ~
$ mkdir trabajos1
$ mkdir trabajos2
$ cd trabajos1
$ mkdir prueba1
$ touch prueba1/archivo1 prueba1/archivo2
$ pwd; ls -l
$ cd ../trabajos2
$ ls -l ../trabajos1/prueba1
$ cd
```

Ahora, deberá crear otro subdirectorio, prueba2, debajo del directorio trabajo2 en su directorio home.

¿Cómo puede usted especificar desde la línea de comandos esta operación utilizando los nombres de rutas-relativas y absolutas?

Elimine el directorio trabajo1 creado anteriormente. Ejecutando el siguiente comando:

$ cd; rmdir trabajo1

¿Qué pasó? ¿por qué no funcionó? ¿Puede usted eliminar el directorio trabajo1 usando un comando diferente? Hágalo, pero utilice la opción de interactivo para ver el orden en que los archivos son eliminados.

6. Ahora vamos a ver unos cuantos utilitarios para observar los archivos.

Para ver parte de los archivos podemos utilizar:

$ head contraseña
$ tail contraseña
$ tail -5 contraseña

Demos un vistazo al tipo de datos contenido en un archivo con el comando:

$ file * **Todos los archivos en su directorio actual.**
$ file /etc/p* **Los archivos en el directorio /etc que comienzan con "p'**

¿Puede usted pensar en una circunstancia en que el comando file le fuera útil?

7. Tomando en cuenta que el archivo /etc/passwd está compuesto por un usuario valido del sistema por línea. ¿Cuántos usuarios validos tiene su sistema?

$ wc -l /etc/passwd

Ejercicio 6-2: Navegar el Sistema de Archivos

1. Vamos a practicar navegar a través del sistema de archivo y explorar una cuantas opciones. Ejecute las siguientes secuencia de comando:

$ cd # Asegúrese de estar en el directorio home
$ mkdir -p dir1/dir2 # ¿Qué efecto tiene la opción -p?
$ cp /etc/passwd dir1/dir2
$ rm -r dir1 # Responda "no" para ver que sucede
$ rm -rf dirl # ¿Qué efecto tiene la opción -f?

Repita el proceso completo, pero en vez de copiar el archivo de /etc/passwd, crea uno que le pertenezca:

$ mkdir -p dirl/dir2
$ touch dirl/dir2/archivo1
$ rm -r dir1

¿Cuál fué la diferencia esta vez? ¿Por qué no fué la opción -f?

Si no tenemos permisos de escribir (w) a un archivo, somo advertido cuando intentamos vorrar con el comando rm. La opción -f del comando rm le informa al sistema que sambemos exactamnete lo que hacemos, asi que no me preguntes si estoy seguro.

2. Esto le servirá al comando de búsqueda de patrones grep. El comando grep es un utilitario poderoso, capaz de leer archivo de texto. Su función es desplegar aquellas líneas de un archivo que contengan la cadena de caracteres dada como argumentos.

Busquemos cadenas de caracteres dentro de un archivo de texto:

$ grep root /etc/passwd

Desplega todas las líneas del archivo /etc/passwd que continen la cadena root.

$ grep bash /etc/passwd

Desplega todas las líneas del archivo /etc/passwd que continen la cadena "bash".

Ejercicio 6-3: Permisos de Archivos

¿Qué hacen los siguientes comandos?

1. $ chmod u+x archivos...

   Agrega permisos de ejecución para el usuario.

2. $ chmod a+r archivos...

   Agrega permisos de lectura para todos.

3. $ chmod g+w archivos...

   Agrega permisos de Lectura y Escritura para el grupo.

4. $ chmod og-w archivos...

   Retira los permisos de Escritura para el grupo y los otros.

5. $ chmod -R go-wx directorio

   Retira los permisos de Lectura y Escritura para el grupo y los otros recursivamente.

6. $ chmod og=rx archivos...

   Establece los permisos de Lectura y Ejecución para el grupo y los otros.

7. $ chmod go= archivos...

   Retira todos los permisos para el grupo y los otros.

Ejercicio 6-4: Organizar los Archivos

1. Para ilustrar las implicaciones de derecho a acceso de los archivos usaremos comandos que modifican los permisos de los archivos.

   Cambiese a su directorio home, efectúe una copia del archivo /etc/group y observe sus permisos:
   $ cp /etc/group grupo.prueba
   $ ls -l grupo.prueba

   Ahora elimine los permisos de Lectura (Read) para los usuarios otros y confirme el cambio:
   $ chmod o-r grupo.prueba
   $ ls -l grupo.prueba

   Ahora elimine los permisos de Lectura (Read) del usuario y trate de leer el archivo:
   $ chmod u-r grupo.prueba
   $ cat grupo.prueba

   ¿Qué pasó? ¿Por qué no puede usted leer el archivo? ¿Si usted y el archivo pertenecen al mismo grupo y el grupo tiene permisos de Lectura establecidos, porqué no puede usted leerlo?

   Después de remover los permisos de Lectura a un archivo para el dueño del archivo, usted no podrá leermo. Fijese que los permisos del grupo y los otros no son consultado si usted es el dueño del archivo.

2. $ rm grupo.prueba

   Usted puede eliminar el archivo porque para borrar simplemente debe tener acceso de W y X al directorio que el archivo reecide. El comando rm es una operación sobre el directorio, no el archivo.

Ahora pruebe:
$ rm /etc/group

Responda "yes" (sí) cuando le pida confirmación. ¿Porqué no pudo borrarlo?

No pudo borrarlo porque usted no puede escribir al directorio /etc, los permisos del archivo entonces no son importante. Observe los permisos de /etc con el comando:
$ ls -ld /etc

Pruebe éstos comandos desde su directorio home y revise las opciones de permisos después de cada comando:
$ cp /etc/inittab tabla_inicio
$ chmod ug=r tabla_inicio
$ chmod o= tabla_inicio

Ahora intente eliminar el archivo:
$ rm tabla_inicio

Cuando le pida confirmación responda "no". ¿Entiende usted la secuencia de comandos que acabamos de ejecutar?

Y finalmente utiliza la opción de forzar para eliminarlo:
$ rm -f tabla_inicio

Los mismos principios aplican en este ejemplo, cuanddo hacemos un archivo no escribible, el comando rm le pedirá confirmación, pero le permitirá eliminarlo si la respuesta es positiva, y los permisos del directorio lo permiten.

La opción -f simplemente suprime el mensaje de advertencia.

3. Vamos a investigar los permisos de acceso a los directorio, el bit "x". Usted debe aún tener el directorio trabajo que creamos en los ejercicios anteriores. Si por algún razón lo eliminó, deberás crearlo nuevamente en sus directorio home.

   Coloque un par de archivos en el directorio trabajo; usted puede usar los comandos cp, touch, > o cualquier otro metódo que usted prefiera (vi, emacs, pico, nano, joe etc.).

   Asegúrese de estar en su directorio home, remueva el bit de ejecución (x) del directorio trabajo:
   $ cd; chmod u-x trabajo2

   ¿Cuáles son las implicaciones de lo que usted acaba de ejecutar?

   El comando chmod elimina el permiso de Ejecución (búsqueda) del directoro trabajo2.

   Escriba los siguientes comandos:
   $ ls trabajos2
   $ ls -l trabajo2

   ¿Puede usted explicar porque recibe un mensaje de error?

   El comando simple ls puede desplegar el nombre de los archivos, pero con la opción ls -l no puede buscar los inodos ni lo atributos de los archivos.

   ¿Puede usted ejecutar lo siguiente con éxito?
   $ cd trabajo2

   Usted no puede cambiar (cd) su pwd a un directorio que usted no tiene el bit de ejcución (x), tampo, puede leer el contenido de nibgún archivo allí dentro.

   Ahora restablezca los permisos así:
   $ chmod u+x trabajo2

Asegúrese de poder efectuar listado largo del directorio trabajo2, hacer cd, etc.

4. Crearemos aquí un shell script. Escriba los siguientes comandos:

```
$ cat >reloj
cal
date
Luego escriba CONTROL+D.
```

Luego esta próxima secuencia de comandos:

```
$ ./reloj
$ chmod u+x reloj
$ ./reloj
```

Usted acaba de crear su primer shell script de bash que puede ser ejecutado como un comando de GNU/Linux normal. Note que usted necesita los permisos de ejecución (x) en el archivo que creó para hacerlo un script del shell; GNU/Linux no tiene extensiones en sus archivos así que no reconoce los archivos por su extensión.

Ejercicio 6-5: Vínculos/Links

No se proveen soluciones a éste ejercicio.

Observe los próximos comandos cuidadosamente y asegúrese de entender que hace cada uno:

```
$ mkdir vínculos # crea un directorio
$ cd vínculos
$ cat > archivo1 # crea dos archivos de data
Este es el archivo1
^D

$ cat > archivo2
Este es el archivo2
^D

$ ln archivo1 archivo3 # Un vínculo de hard link
$ ln -s archivo2 archivo4 # vínculo simbólico
```

Ahora veamos los siguientes archivos:

```
$ ls -il
$ cat archivo1
$ cat archivo2
$ cat archivo3
$ cat archivo4
$ ls -ilL
```

Ya tenemos tres archivos: archivo1 y archivo3 son los mismos (vea su número de inodo) y el archivo4 es un vínculo simbólico al archivo2.

```
$ rm archivo1
$ cat archivo3
```

Haber removido el archivo1 no tuvo ningún efecto con el archivo3. Un archivo es eliminado cuando su último vínculo es eliminado. Ahora pruebe:

```
$ rm archivo2
$ cat archivo4
```

Fíjese el mensaje de error confuso del comando cat, éste nos dice que no puede leer el archivo4. Nosotros sabemos que el archivo4 existe, pero es un vínculo simbólico al archivo2, el cual borramos anteriormente. Los vínculos son transparentes a todos los utilitarios, excepto ls y rm.

Crea el archivo2 de nuevo, esta vez simplemente copie el archivo passwd del sistema dentro del directorio actual, luego despliegue el contenido del archivo4 a pantalla:

$ cp /etc/passwd archivo2
$ ls -il
$ cat archivo4

El acceso al archivo2 ha sido re-establecido, usted puede de nuevo desplegar su contenido, aunque ya no es el mismo contenido. Recuerde que deberá remover los vínculos si usted remueve el archivo original.

Ejercicios 6-6: Administración de los Dispositivos

1. Para crear un dispositivo, haremos un nuevo nodo de dispositivo de bloques y lo llamaremos disquete en el directorio /dev, y usaremos los mismos números de mayor y menor del dispositivo /dev/fd0.
   # mknod /dev/disquete b 1 112

   Crea una segunda entrada y llamela raw.disquete, el cual será un dispositivo de caracter basado en el dispositivo existente /dev/fd0
   # mknod /dev/raw.disquete c 1 112

2. Ahora dé formato a un disquete utilizando su nuevo dispositivo. Use el sistema de archivos ext2.:

   ¿Cuál de los dos nuevos nombres de dispositivos creados utilizaría usted para llevar la operación a cabo?
   # mk2fs /dev/disquete

   ¿Puede usted escribir archivos al disquete recién formateado? Ya que mk2fs crea un sistema de archivos la respuesta si podemos escribir al floppy.

PREGUNTAS POST-EXAMEN

1. Usted copia el archivo1 archivo2 archivo3 en el directorio /home/usuario/trabajos. ¿Cómo puede hacer esto en una sola línea de comando?
   (R) $ cp archivo1 archivo2 archivo3 /home/usuario/trabajos

   ¿Cómo puede usted eliminar éste directorio y todos los archivos dentro de el en un sólo comando?
   (R) $ rm -r /home/usuario/trabajos

2. Al editar un documento de 4 páginas su editor se cuelga. ¿Cuál de los dos comando more o less resultaría de mejor provecho para poder ver el contenido hacia arriba y hacia abajo?
   (R) less porque nos permite subir y bajar en el documento.

3. Usted necesita asignarle los permisos de lectura y escritura al archivo llamado archivo.txt (Asuma que el archivo tiene permisos de -rwx------). Explique ¿Cómo lograr esa tarea?
   (R) chmod 555 archivo.txt
   (R) chmod ugo-w archivo.txt

   Uno de estos comandos le funcionará.

4. Usted se encontraba en el directorio /home/usuario/trabajo. ¿Cuál de éstos directorios es el directorio padre del directorio trabajo/? ¿Cómo puede usted detectar esto rápidamente?
   (R) usuario es el directorio padre
   (R) use el comando cd ..

5. El usuario se encuentra en el directorio /etc y desea encontrar un archivo que sólo sabe que su nombre empieza con “ho” ¿Cómo puede usted encontrar éstos archivos sin utilizar el comando ls -l?

(R) Pruebe con: ls -l ho*

CAPÍTULO 7

PREGUNTAS PRE-EXAMEN

1. La mayoría de las distribuciones de GNU/Linux vienen configuradas con un Shell por defecto. ¿Cuál es el nombre de ese Shell?
   (R) bash

2. Necesita comparar dos archivos y buscar cuales líneas de código fuente tienen el cambio de versión 1 a versión 2. ¿Cuál comando básico puede ser usado para llevar esto a cabo?
   (R) diff

3. ¿Por qué es tan importante para un administrador de sistemas GNU/Linux saber utilizar por lo menos un editor de texto? Nombre algunos editores de textos populares de GNU/Linux.

   (R) Para editar los archivos de configuración, los cuales son todos de texto. Algunos editores de texto populares son vi, emacs, nano, pico, joe, elvis, etc.

4. ¿Por qué es que algunos comandos son internos del Shell y algunos comandos son simple ejecutables almacenados en algún directorio específico?

   (R) Los comandos built-in siempre son encontrados antes que los externos. Es posible que un comando en un directorio anterior en el orden de la ruta o path oculte en segundo.

5. ¿Cúal es el significado de la variable del Shell PATH?

   (R) El shell busca en los directorios de la variable PATH por los programas a ejecutar.

Ejercicio 7-1: El Ambiente del Shell

1. ¿Cuál es el valor de las siguientes variables del shell en su sistema? Debe ejecutar la siguiente sentencias.
   HOME ----->>> $ echo $HOME
   TERM ----->>> $ echo $TERM
   PATH ----->>> $ echo $PATH
   PS1 ----->>> $ echo $PS1

   También puedes ejectar el comando:
   $ set

2. Defina algunas variables personales, recuerde proteger algunos caracteres de su valor de variable del shell. Establezca el nombre de la variable name para que tenga el valor: FCAD

   Despliegue su valor:
   $ echo $nombre

   Establezca la variable para que tenga el siguiente valor: <4 espacios>Padre Pina 102 (la dirección estará indentada 4 espacios en blanco).

   Despliegue su valor:
   $ echo $direccion
   $ echo " $direccion"

   Si estableció correctamente los valores, la salidas de éstos dos comandos debe aparecer un poco diferen-

te.

¿Por qué?

(R) El echo sin comillas, imprime el valor tal cual de la variable dirección.

(R) El echo con comillas, imprime el valor tal cual de la variable dirección, pero icluye los cuatros espacios antes de la dirección.

3. Para demostrar el manejo de variables embedidas en texto, establezca una variable llamada nombre y dele el siguiente valor:
FCAD<Tab>
(R) $ nombre=”FCAD<Tab>”

Debemos usar las comillas o el shell ignora el espacio de tab.

Ahora Despliegue el valor de la variable nombre seguida inmediatamente por la cadena Fundación. Permita que el tabulado contenido en el nombre separe el valor y no tener que hacerle echo a ningún espacio.
(R) $ echo ${nombre}, Fundación Código Abierto Dominicana.

Una manera alternativa, pero no menos efectiva es:
(R) $ echo “$nombre”, Fundación Código Abierto Dominicana.

4. Para poder entender el propósito de exportar las variables, escriba la siguiente secuencia de comandos:

```
$ echo $TERM          # Tome apunte del valor arrojado
$ unset TERM          # Remueve el valor asignado a la variable TERM en memoria
$ echo $TERM          # Usted eliminó el valor de $TERM
$ man bash            # No significa nada al usar bash
$ TERM=               # Valor nota anteriormente
$ man bash            # Aún persiste el problema: ¿Por qué?
$ export TERM
$ man bash            # Ya todo esta como cuando empezamos
```

Pregunta: Explique brevemente el proposito de exportar la variable TERM.

(R) Si la variable TERM no es exportada, entonces los subprocesos, como es el comando man, no pueden accesarlo. En este caso, man no sabe que tipo de terminal usted se encuentra ejecutando comandos y quizás no encuentre como manejarse en ella. En la mayoría de sistemas el primer comando que nos demuestra tener problema con TERM, que falta o que esta mal establecido, es el vi. En GNU/Linux, vi tiene un manejo genérico del manejo de la pantalla y puede funcionar sin la variable TERM.

5. Pruebe los siguientes comandos. Asegúrese de entender lo que hace cada comando:

```
$ cd              # Cambia de directorio a su directorio home
$ cd /etc         # Cambia de directorio al directorio /etc
$ cd -            # Cambia de directorio al directorio anterior, en este ejemplo al directorio home
$ pwd             # Imprime directorio actual, en este caso su directorio home
$ cd -            # Cambia de directorio al directorio anterior, en este ejemplo al directorio /etc
$ pwd             # Imprime directorio actual, en este caso su directorio /etc
$ ls ~            # Lista el contenido de su directorio home
$ ls ~root        # Lista el contenido del directorio home de la cuenta root
$ cd              # Cambia de directorio a su directorio home
$ ls ~-           # Lista el contenido del directorio anterior, /etc
```

6. Personalice su archivo profile. Agregue las siguientes líneas al final de su archivo .bash_profile:

```
# cambios locales
cal
mesg n
```

¿Cuál es la diferencia del proceso de ingreso al sistema/login?

(R) Una vez agregue estas líneas a su archivo .bash_profile, se le mostrará un calendario de los meses y los cabezales de sus mensajes de correo al ingresar al sistema. Esto se le presentará en todo los logins hasta que elimine estas líneas de su archivo de preferencias.

7. Para modificar su prompt para que incluya el directorio actual, escriba los siguientes comandos:

```
$ cd
$ PS1="$PWD[$$] "
```

¿Qué efecto refleja éste comando en el prompt de su pc?

(R) Para incluir el directorio de trabajo actual en el PROMPT, la variable PS1 debe tener en su valor la cadena $PWD. Al usar comillas doble permite al shell expandir el valor de la cadena $PWD en la línea de comando durante su lectura. Esto corrije el prompt permanentemente al valor correcto en el momento de asignación.

Cambie al directorio /etc para ver si el prompt refleja el nuevo directorio. ¿Si no, por qué no?

(R) Para prevenir esto y correjirlo para que el shell sustituya el valor cuando realmente se expide el prompt, deberá usar las comillas simples.

Escriba el comando correcto para establecer su prompt para que le incluya su directorio actual.

(R) $ PS1=' $PWD[$$] '

Ejercicio 7-2: Cambiando el Ambiente

1. Cambie su ambiente para que las páginas man se visualicen con more y no con less.

```
(R) $ PAGER=more
$ export PAGER
o
(R) $ export PAGER=more
```

Deberá agregar una de estas sentencias a su archivo .bash_profile o .bashrc para hacer el cambio permanente.

2. Personalice su ambiente con definiciones especificas de bash. Habilitaremos permanentemente la edición con vi de la línea de comandos para todas las futuras sesiones de login (en bash) y agregaremos algunos alias muy útiles, que también serán permanentes, como los siguientes:

```
dir         ls
h           history
lf          ls -FC
la          ls -la
(R) set -o vi
    alias dir=ls
    alias h=history
    alias lf='ls -FC'
    alias la='ls -la'
```

Salga del sistema e ingrese de nuevo para verificar los cambios.

Ejecute un segundo shell de bash, usando el comando:

$ bash y asegúrese que su alias aún son reconocidos. Salga del shell bash.

(R) Asegurece que su variable de ambiente ENV está establecida como .bashrc en su directorio home.

Revise su archivo .profile para ver si las siguientes líneas existen. Sino, entonces agreguelas asi:

```
ENV=$HOME/.bashrc
```

export ENV

3. Manipule opciones en su shell interactivo. Ajuste su sessión de login para deshabilitar CONTROL+D como combinación de teclas para salir de sessión pero permita que CONTROL+D salga de otros shell pero no de la sessión Salga y entre de sessión para verificar los cambios. Ejecute un segundo comando:
$ bash

Asegúrese que CONTROL+D es aún reconocido y sale del shell.

(R) Puede establecer la opción ignoreeof en la línea de comandos para experimentar como el shell va a reaccionar los CTRL+D subsecuentes.

```
$ set -o ignoreeof
$ ^D
Use 'exit' to leave bash
$
```

(R) Como todo cambio de ambiente los cambios efectuados desde la línea de comandos sólo afectan la sessión actual.

Si deseamos que los cambios sean permanentes, debe agregarlo a su archivo .profile (no a su archivo .bashrc). En este ejemplo puede usar vi y agregarle la siguiente línea:

```
set -o ignoreeof
```

4. No todos los comandos son iguales. Alguno de ellos no existen como un programa por separado en el disco para éstos en el shell no usa la variable PATH para encontrarlos.

Salve y restablezca su ruta de búsqueda con :

```
$ SAVEPATH=$PATH
$ echo $SAVEPATH
$ PATH=
```

Ahora compruebe algunos comandos estándares:

```
$ pwd
$ ls
$ wc pass1
$ history
```

¿Por que algunos comandos funcionan y otros no?

Use el comando type para verificar.

Restaure su ruta de búsqueda con éste comando:

```
$ PATH=$SAVEPATH
```

(R) Esto comprueba que no todos los comnado son iguales. Algunos no existen fisicamente en el disco y otros sí. Para los que existen necesitamos la variable PATH para poder localizarlos. Los que son Built-In del shell no utilizan ésta variable.

5- Vamos a practicar configurando y exportando variables. Defina la variable Var1, dandole un valor que usted escoja y no la exporte.

Revise el estado de las opciones de su shell actual. Identifique y establezca la opción que obliga que todas las variables nuevas sean exportadas. Defina la variable Var2 y dele otro valor y no la exporte.

Inicie un nuevo shell e imprima las 2 variables. Pruebe los comandos set y env en éste nuevo shell.

Reconocen los comandos sus variables?

Inicie un shell más y repita la exploración.

Finalmente antes de cerrar las secciones adicionales, ejecute el comando ps -j y identifique la relación

padre/hijo entre los 3 Shells.

¿Quién es el proceso padre de su Shell principal?

```
$ Var1=abc
$ set -o                    # On significa SI; off significa NO
$ set -o allexport          # No hay necesidad de exportar variables aún

$ Var2-123
$ bash
$ echo $Var1 $Var2
123                         # allexport funcionó en Var2; Var1 fué definido antes de establecer allexport
$ set                       # Var2 está listada (existe)
$ env                       # Var2 está listada (existe y está exportada)
```

(R) La opción allexport permite que todos los hijos de procesos subsecuentes vean las nuevas variables.

(R) Verificar los valores PID y PPID con la salida del comando ps j nos permite identificar el PID de su shell de login. Podrá ver que el padre de su shell principal es un proceso con PID=1. Este proceso es de nombre init, el cual es responsable de iniciar la mayoría de programas y servicios al inicio del sistema.

(R) Para confirmar esto ejecute un lista largo de todos los procesos activos en su sistema:

```
$ ps lax | more
```

(R) Podrá ver el proceso init (proceso numero 1) cual es el padre a un gran numero de procesos de host.

Ejercicio 7-3: Generación de Nombre de Archivo

¿Qué se lograría con los siguientes comandos?

1. $ ls *.?

   (R) Listar todos los archivo en el directorio actual que terminan con punto seguido de un único caracter.

2. $ more [A-Z]*

   (R) Desplegar todos los archivos en el directorio actual que su nombre empieza con letra mayúscula y seguido por cualquier número de caracteres incluyendo ninguno.

3. $ ls /etc/[!a-m]*

   (R) Liste todos los archivos en el directorio /etc que empiezan con cualquier caracter menos en el rango de la a a la m (a-m) seguido por cualquier número de caracteres incluyendo ninguno.

4. $ file /usr/bin/*x*

   (R) Clasificar el contenido de todos los archivos en el directorio /usr/bin/ que sus nombre contienen un x.

5. $ ls [a-z]*[0-9]

   (R) Listar todos los archivos en el directorio actual que su nombre empieza con letra minúscula, seguido por cualquier número de caracteres incluyendo ninguno y termina en un dígito.

6. $ ls -a .[!.]*

   (R) Lista todos los archivos en el directorio actual que su nombre empieza con un punto, seguido por cualquier número de caracteres que no sea un punto y seguido por cualquier número de caracteres.

7. $ ls -d /etc/*.d/*

(R) Lista todos los directorio del directorio /etc que su nombre termina con un punto seguido por una d (.d), y lista todos los nombres de archivos contenido en ellos.

Este ejemplo ilustra el hecho de que podemos usar comodines para generar nombres de rutas de directorios asi como los nombres de los archivos.

Ejercicios 7-4: Uso del Shell Bash

1. Primero generamos unas cuantas líneas en el comando history (simplemente ejecutando unos cuantos comandos en el CLI); entonces practicaremos los mecanismos del comando history.

```
$ w
$ pwd
$ ls -1
$ more abiertos.txt
$ more linux
$ wc -l abiertos.txt linux
$ history
```

Ahora escriba:

```
$ !ls
$ !wc
$ !more          #¿Cuál de los comandos fueron ejecutado?
$ !his
$ !n             #Reemplace n con cualquier número del archivo history.
```

(R) El mecanismo history esta disponible desde que el shell bash se inicia. Pero, hasta el momento que le informamos al shell cual editor deseamos usar para la edición de líneas, la única manera que podemos usar los comandos anteriores es llamándolos explicitamente, usando el número de línea desde el archivo history.

2. Ahora utilizaremos el editor vi para visualizar nuestro archivo history, localizaremos un comando y lo editaremos, y lo prepararemos para otra ejecución. Habilite la edición vi con éste comando:

```
$ set -o vi
```

Una vez iniciado el mecanismo con el comando set -o vi, usted puede usar el mecanismo history por completo.

Podemos también navegar en todo el archivo history y editarlo usando el editor vi. Despues de sus últimos comandos su history debe reflejar algo similar a esto:

```
100 w
101 pwd
102 ls -1
103 more abiertos.txt
104 more linux
104 wc -l abiertos.txt linux
105 history
106 ls -l
107 set -o vi
```

(R) Fíjese como los comandos "!comando" no son parte del history, sino el comando al cual este hiso referencia.

3. Generación de nombres con patrones:

```
p*               # empiezan con "p".
*y               # Termina con "y".
m*d              # Empieza con "m" y termina con "d".
```

```
[egm]          # Empieza o con “e”, “g” o “m”.
*o*u*          # Contiene una “o” seguida por (pero no necesariamente de inmediato) por una “u”.
*.conf *       # Contiene una cadena “conf” en cualquier lugar del nombre del archivo.
s*n*           # Empieza con una “s” y contiene una “n”.
????           # Contiene exactamente cuatro caracteres.
[!a-z]*        # No empieza con letras minúsculas.
*[0-9]         # Contiene un dígito en algún lugar del nombre del archivo.
```

4. Vamos a seguir experimentando con los wildcards. Debe traerle a un entedimiento del concepto de quien hace que en los caracteres especiales en la línea de comandos.

   Pruebe con los siguiente comandos:

   ```
   $ cd
   $ echo *[a-z]*
   $ ls -d *[a-z]*
   ```

   Luego pruebe:

   ```
   $ echo *.*
   $ ls -d *.*
   ```

   ¿Entiende usted la salida?

   (R) El comando echo * [a-z]* imprimirá todos los nombres de archivos en el directorio actual seguido por todos los archivos que sus nombre empiezan de la a hasta la z seguido por cualquier número de caracteres.

   El comando ls tiene el mismo efecto excepto que la salida que rinde es diferente.

   (R) Los dos conjunto de comandos deben ilustrar el que hace que, el shell en este caso queda demostrado es el que efectúa la expansión del wildcard.

   (R) Los nombres de archivo generados son pasados al comando como sus argumentos y dependiendo del comando, los nombres serán usados y presentados diferente.

5. Vamos a probar definiendo alias. Cree un alias y llámelo h para el comando history el nuevo alias:

   ```
   $ alias h=history; h
   ```

   Agregue nuevo alias con los valores mostrados debajo, y entonces pruébelos>

Nombre del alias	Valor
cambdir	cd
dir	ls -d
minuz	ls -d [a-z]*

   (R) Estos son algunos posibles pasos:

   ```
   $ alias h=history
   $ h
   $ alias cambdir=cd
   $ alias dir=’ls -d’
   $ alias minuz=’ls -d [a-z]*’
   $ cambdir  /etc
   $ cambdir
   $ dir
   $ dir p*
   $ minuz
   ```

Ejercicio 7-5: Expansión y Wildcards

1. ¿Qué cree usted que efectúan los siguientes comandos?
   $ ls -d /*/p*wd

   (R) La cadena /*/p*wd es expandida a todos los archivos en todos los directorios del segundo nivel de la jeraquía de la raíz, que su nombre empieza con p seguido por cualquier número de caracteres y termina en wd. Esto expandiera a $ls /dir-cualquiera/p*wd (ej. /bin/pwd /bin/passwd /etc/passwd)
   $ cd /usr/*term*/v
   $ cd /usr/*/term*/[ab]

   (R) Estas dos líneas demuestran que si el shell genera nombres que no son compatibles con el comando, es el comando quien tiene que enfrentar el problema.

   El primer comando cd esta correcta, y el comando cd es efectuado con éxito.

   El segundo cd, debe haber generado un mensaje de error. La razón es que esta generá más de un directorio y cd no puede recibir dos directorio como argumento.

   Si queremos probar esta misma expresión lo podemos hacer con el comado ls que si puede recibir más de un directorio como argumento asi:
   $ ls -d /usr/*/term*/[ab]
   /usr/lib/terminfo/a /usr/libr/terminfo/b

   Que este funciona es muestra que es el comando cd quien debe protejerse de que el bash le pase argumentos erróneos.

2. Para probar las cadenas wildcard, use el comando ls o echo. Use solamente rm cuando usted está seguro de los resultados.

   • En su directorio home deberá crear un nuevo directorio y llámelo backup
   $cd ; mkdir backup

   • Cambiese a éste directorio
   $cd backup

   • Copie a éste directorio backup todos los archivos en el directorio /etc que su nombre empiecen con la letra l.
   $cp /etc/l* .

   • Liste y luego clasifique el contenido de los archivos en el directorio backup e identifique todos archivos de data o binarios. Y tome nota de esto.
   $ls ; file *
   $ file l?[.c]*[!f]

   Todos los archivos que su nombre empiezan con una l, sseguido por una caracter único “?”, seguido por o un punto “.” o una “[.c]”, seguido por un número de caracteres (*), siempre y cuando el último no sea una f ([!f]).

   • Dando uso a un sólo comando y con especificaciones de wildcard elimine todos los archivos de data o binarios anteriormente identificados
   $ rm l?[.c]*[!f]

Ejercicio 7-6: Uso de find

¿Qué efectúan los siguientes comandos find?

1. $ find . -print

(R) Lista todos los nombres de archivos en el directorio actual; nota la ausencia de criterio de búsqueda.

2. $ find . -type d -print

(R) Lista todos los nombres de los archivos tipo directorio en el directorio actual; como la búsqueda es recursiva, los despliega en forma de árbol.

3. $ find /home -name .profile -print

(R) Búsca todos los archivos cuyo nombres igualan .profile bajo el directorio home.

4. $ find /home -name .bash_profile -print

(R) Búsca todos los archivos cuyo nombres igualan .bash_profile bajo el directorio home.

5. $ find . /tmp /usr/tmp -name core -exec rm {} \;

(R) Búsca todos los archivos cuyo nombres igualan core bajo los directorios actual, /tmp y /usr/tmp.

6. $ find . -name "*.o" -ok rm {} \;

(R) Búsca todos los archivos cuyo nombres trerminan con "o" bajo el directorio actual, y se lo pasa como argumento al comando rm, el cual pide confirmación para su pronta eliminación.

7. $ find / -type f -size +1k -print >/tmp/grandes 2>/dev/null &

(R) Búsca todos los archivos que son del tipo archivo en todo el disco, y que además superen el tamaño de 1KB; almacena su nombre en un archivo llamado /tmp/grandes y descarta todos los mensajes de error.

Ejercicios 7-7: El Uso de Expresiones Regulares con grep

1. ¿Qué efectúa el siguiente comando?

$ ls -l | grep '^d'

(R) Lista solamente los directorios en el directorio actual.

$ grep '^user[0-9]' /etc/passwd

(R) Búsca todas las líneas en el archivo /etc/passwd que empiezan con user seguido de un número.

$ grep '^[A-Za-z]*[0-9]$' archivo.txt

(R) Búsca todas las líneas en archivo.txt que empiezan con un alfacaracter mayúscula o no, seguido por un numero de caracteres y terminando con un dígito.

$ Is -a | grep '^\.[^.]'

(R) Búsca todos los archivos desde el lista cuyo nombres empiezan con un punto "." seguido por cualquier cosa menos otro punto. Esta sentencia busca todos los archivos ocultos menos el padre y el mismo (. ni .. son listados).

$ grep '^.*,[0-9]\{10,\}' archivo.txt

(R) Búsca todos las lineas en archivo.txt que contengan un número mayor que un billón (nueve números).

2. ¿Qué buscamos en éste ejercicio?

$ grep '^.*.[0-9]\{1,\}' estudiantes.txt

Jose Paredes,19750726,Interior,27,Herrera,1975,Sistema,Programacion,Base de datos

Francis Francis,19750727,Interior,27,Rosal,1974,Sistema,Programacion,Base de datos

Ejercicio 7-8: Uso de Expresiones Regulares para Buscar en vi

1. ¿Qué línea puede ser localizada con el siguiente?

```
/^{                          # En esta línea búscamos hacia alante una que empieza con {
/;$                          # En esta línea búscamos hacia alante una que termina con ;
/^TERM                       # En esta línea búscamos hacia alante una que empieza con TERM=
?^p[<tab><espacio>]*TERM=    # En esta línea búscamos hacia atrás una que empieza con TERM= aunque este
                             indentada con espacios o tabs.
/$'\$                        # En esta línea búscamos hacia alante una que contiene $$
```

Ejercicio 12-9: Uso Avanzado de Expresiones Regulares

1. Prueba lo siguiente:

```
$ find /usr/bin -name p* -print
```

¿Puede explicar, por qué no parece estar funcionando como esperábamos?

(R) El shell expande el asterisco en el directorio de donde se ejecuta la sentencia antes de ejecutar el comando en /usr/bin con, claro esta, resultados impredecibles.

• Si la p* no iguala nombres de archivo, el wildcard no es expandido y se pasa sin tocar para encontrar archivos. En este caso find funciona por accidente como se esperaba.

• Si la p* iguala a un solo archivo en su directorio home, el wildcard es expandido y el nombre del archivo es pasado a find. El comando estaría buscando un sólo archivo en la estructura de directorio /usr/bin.

• Si la p* iguala más de un archivo en su directorio home, el wildcard es expandido y todo slo nombres encontrados son pasados al comando find. Esto causa que find arroje un error de sintaxis, ya que find espera sólo un nombre de archivo no varios.

Una vez usted establezca el trazador en su shell asi:

```
$ set -o xtrace
```

Repita el comando find anterior. Ahora el shell le devuelve cierto tipo de error, algo parecido a este:

```
$ find /usr/bin -name p* -print
+ find /usr/bin -name passwd passwd2 -print
find: passwd2: unknown expression primary
```

Para preparar el comando find para que le arroje los resultados esperados, claro esta es un problema de comillas, ejecutaremos el comando find ya correjido.

Pruébelo!, entonces compare los resultados del siguiente comando find, claro ahora corregido y su sentencia ls:

```
$ find /usr/bin -name "p*" -print
```

El comando ls equivalente es:

```
$ ls -R /usr/bin/p*
```

Este no recogerá los archivos en esos subdirectorios que no empiezan con "p", como por ejemplo **/usr/bin/folder/pentagono.**

2. En ésta pregunta de éste ejercicio ponemos a prueba su manejo del comando find. Encuentre y liste todos los archivos en su directorio home.

```
$ cd
$ find . -print
```

Repita el último comando, pero, redireccione la salida a un archivo y redireccione además, cualquier mensaje de error, al dispositivo null.

$ find . -print > listado-archivos 2>/dev/null

Modifique su última sentencia en la línea de comandos para que liste solamente los archivos que han sido modificados hoy y déle salida a un archivo diferente. Claro está, ésto demanda una nueva opción del comando find; use las páginas man para encontrarla.

$ find . -mtime 0 -print > listado-archivos-hoy 2>/dev/null

(R) La opción mtime también pudo haber sido -1, la cual tuviese el mismo efecto.

3. En éste paso experimentaremos con el uso de expresiones regulares.

Primero debemos preparar el ambiente. Ejecute los siguientes comandos:

$ cd /etc
$ ls -a > ~/datos.txt
$ cd

Este conjunto de comandos le almacenará una lista de los archivos en el directorio /etc en un archivo datos.txt en su directorio home.

Para lo que queda de éste ejercicio, necesitará usar el comando en el siguiente formato:

grep 'regexp' datos.txt

Donde el patrón 'regexp' será apropiado para el tipo de búsqueda que le pediremos llevar a cabo.

Escriba expresiones regulares que encontrarían todas las líneas en el archivo de data que:

• Empiezan con la letra "p".
grep '^p' datos.txt

• Terminan con la letra "y".
grep 'y$' datos.txt

• Empiezan con una "m" y terminan con una "d".
grep '^m.*d$' datos.txt

• Empiezan con una "e", "g", o "m".
grep '^[egm]' datos.txt

• Contienen una "o" seguida (no necesariamente de inmediato) por una "u".
grep 'o.*u' datos.txt

• Contienen una "o", entonces cualquier caracter único y entonces "u".
grep 'o.u' datos.txt

• Empiezan con una letra minúscula.
grep '^[a-z]' datos.txt

• Contienen un dígito.
grep '[0-9]' datos.txt

• Empiezan con una "s" y contienen una "n".
grep '^s.*n' datos.txt

• Contienen exactamente 4 caracteres.
grep '^....$' datos.txt
grep '^.\{4\}$' datos.txt

• Contienen exactamente 4 caracteres, pero, ninguno de ellos es un ".".
grep '^[^.]\{4\}' datos.txt

4. Dé solución al siguiente problema usando sed:

• Use el comando sed para eliminar todo, menos el nombre del usuario, de la salida del comando who.

who | sed ‘s/[ ].*$//‘ datos.txt # En los brackets un espacio y un tab

• Use el archivo estudiantes.txt como archivo fuente, elimine todo excepto, los nombres de las carreras.
sed ‘s/,.*$//‘ estudiantes.txt

• Usando el archivo estudiantes.txt como fuente, imprima toda la data de los estudiantes que estudian Sistemas.
$ sed -n ‘/Sistemas/p‘ estudiantes.txt

• Lo mismo que el anterior, pero, sólo imprima el nombre de los estudiantes. Use sólo un comando sed.
$ sed -n ‘/Sistemas/s/ „.*$//p‘ estudiantes.txt

• Igual que el anterior, pero, agréguele la cadena: “Estudiantes de:” antes de cada línea o estudiante. Puede usar más de un comando sed.
$ sed -n ‘/Sistemas/s/,.*$//p‘ estudiantes.txt | sed ‘s/^/Estudiantes de:/’

Una manera alternativa con el uso de la facilidad de sed para agrupar:
$ sed -n ‘/Sistemas/ {
> s/,.*$//
> s/^/Estudiantes de:/p
> }’ estudiantes.txt

Ejercicios 7-10: Uso Adicional de Herramientas Poderosas

1. Pruebe los siguientes comandos:
$ REV=$(tput rev)
$ NRM=$(tput rmso)
$ DAY=$(date +%e)
$ cal | sed “s/$DAY/$REVDAYNRM”

2. Use sed para traducir múltiple espacios a sólo una coma en la salida del comando ls -1.

(R) La cadena de origen contiene dos espacios (el asterisco después del segundo cero significa cero o más espacios despues de un espacio). El patrón más largo de por lo menos un espacio es igualado y reemplazado por una única coma.
$ ls -l | sed ‘s/ */ ,/g’

3. Busque todos los usuarios del sistema que usan bash o tcsh.

(R) Usamos el comando egrep ya que este ofrece la función OR:
$ egrep “bash$ | tcsh$” /etc/passwd

4. Use una combinación de los comandos tty y sed para definir una variable y llamela TTY, la cual contiene sólo el nombre de login de su dispositivo.

(R) Primero definimos una variable llamada TTY, la cual contiene el nombre de su dispositivo de login:
$ TTY-$(tty | sed ‘s!^.*/!!’) o $ TTY-$(tty | sed ‘s!/dev/!!’)

(R) La primera solución es más flexible ya que permite a dispositivos de nombres, como son /dev/pts/000.

Fijate como el uso de simbolos de admiración son usados para separar las cadenas viejas de las nuevas en el comando de sustitución de sed. De hecho, cualquier caracter puede ser usado siempre y cuando no sea parte de las dos cadenas.

PREGUNTAS POST-EXAMEN

1. Especifique como podemos escapar los caracteres especiales del shell.

   (R) Las comillas sencillas (`), deshabilitan el recognocimiento de todos los caracteres especiales.

   Las comillas dobles (“), protejen la mayoría de los caracteres especiales.

   Barra invertida (\) escapa cualquier significado especial del próximo caracter.

2. ¿Qué comando es usado para crear abreviaciones de comandos o nombres alternativos?
   $ alias

3. ¿Qué comando es usado para hacer que una variable del shell sea global?
   $ export

4. ¿Qué comandos son usados para desplegar texto en GNU/Linux?
   $ vi, less, more, view, pico, nano, joe

5. ¿Qué comando es usado para efectuar búsquedas de cadenas de texto dentro de archivos?
   $ grep

CAPÍTULO 8

PREGUNTAS PRE-EXAMEN

1. ¿Cuáles son los shells más comunes usados en GNU/Linux?
   GNU Bash y tcsh

2. ¿Cómo puede usted lograr que el comando ls escriba su salida a archivo.txt?
   $ ls > archivo.txt

3. ¿Cuál es la diferencia entre un script del shell y un comando regular?

   (R) Muy poca. Por lo general no se puede diferenciar. Los shell scripts son interpretados, mientras que los comandos son compilados a código de máquina.

4. ¿Qué es un filtro?

   (R) Un filtro toma su entrada desde una entrada estándar, la modifica y la escribe a la salida estándar. Los filtros son útiles en sentencias de tuberías o pipes para extraer información específica de la salida de un comando.

Ejercicio 8-1: Entrada y Salida de Comandos

1. Utilice el comando cat para leer el contenido del archivo /etc/passwd y redireccione la salida a un archivo de nombre archivo.1 en su directorio home.

   (R) Leer el contenido del archivo /etc/passwd y almacenar la salida en archivo.1 en su directorio home.
   $ cat /etc/passwd > archivo.1

   Lea el contenido de archivo.1 para confirmar que la redirección funcionó.
   $ cat archivo.1

   Recuerde que el comando cat normalmente abre y lee un archivo existente que es proveído como argumento en la línea de comandos. Si no le suministramos dicho archivo para abrirlo para leerlo, la entrada

estándar del comando cat se revierte a la por defecto, ¿Qué es cuál?

(R) La pantalla (/dev/tty# que nos encontramos)

Use el comando cat para crear un nuevo archivo y llámelo archivo.2 y coloque algún texto en éste.

Escríbale alguna línea de entrada y observe que pasa. Para terminar escriba (^D) en una línea sola.

```
$ cat > archivo.2
Usando cat como un editor de texto.
^D
```

(R) Muchos otros comandos de GNU/Linux trabajan muy parecido a este comportamiento de cat. Algunos otros comandos que efectúan este mismo comportamiento son:
pg, wc, more, sum, sort, awk y sed.

2. Usando el mismo método, del ejercicio anterior, deberá crear tres archivos. Nómbrelos archivo1, archivo2, y archivo3. Coloque sólo una línea de texto, respectivamente, en cada uno:

```
Este es el archivo1
Este es el archivo2
Este es el archivo3
$ cat > archivo1
Este es el archivo1
^D

$ cat > archivo2
Este es el archivo2
^D

$ cat > archivo3
Este es el archivo3
^D
```

Ahora pruebe con los siguientes comandos:

```
$ cat archivo*
$ cat archivo* > archivo123
$ cat archivo123
```

(R) Este ejemplo muestra la funcionalidad de cat para concatenar o combinar archivos en una sola salida.

3. Los siguientes dos comandos aparentan tener el mismo efecto. ¿Puede usted explicar cuál es la diferencia en el tiempo de ejecución?

```
$ cat archivo1
$ cat < archivo1
```

(R) En el primero el archivo fué abierto por cat; en el segundo el archivo es abierto por el shell.

Los dos siguientes son casi idénticos, pero ¿porqué es la salida un poco diferente?

```
$ wc archivo1
$ wc < archivo1
```

(R) Este sigue los mismos principios que el anterior en el primero el archivo fué abierto por wc; en el segundo el archivo es abierto por el shell.

Puede usted explicar ¿porqué el segundo de los dos siguientes comandos no funciona?

```
$ cat archivo*
$ cat < archivo*
```

(R) El segundo falla porque no podemos redireccionar más de un archivo ya que sólo hay un puntero al archivo. Si sólo existe un archivo que iguale el patrón entonces funciona, Fijese que es el shell quien devuelve el error, no cat.

4. (R) La secuencia de eventos del siguiente comando:
$ cat archivo1 archivo2 > archivo1

Es como explicamos:
- El shell ve el símbolo de > y abre el archivo1 para escribirle, lo cual sobre escribe su contenido.
- El comando cat trata de escribir archivo1 a la salida estándar y se da cuenta que el stream de entrada es el mismo de salida, luego genera un error pero sigue adelante.
- El comando cat escribe el contemido de archivo2 a su flujo de salida el archivo1.

¿Cómo puede hacer que el contenido de archivo2 pase a archivo3, sin alterar el contenido de archivo3?

(R) Para agregar el archivo2 al archivo3 mientras que no se elimine el contenido del archivo3, deberá usar el siguiente comando:
$ cat archivo2 >> archivo3

5. ¿Cómo redireccionamos los mensajes de error a un archivo?
$ wc /etc/* 2> archivo.error

¿Cómo se deshiciera usted de los mensajes de error?
$ wc /etc/* 2> /dev/null

¿Cómo almacenaría usted la salida a un archivo y se deshiciera de los mensajes de error?
$ wc /etc/* > archivo.salida 2> /dev/null

Ejercicios 8-2: Más Redirecciones de las E/S de los Comandos

1. Escriba los siguientes comandos:
$ cat >> archivos
Este es el archivo5
Este es el archivo6
^D

(Recuerde que eso es un CONTROL+D al final.) ¿Qué efecto tuvo éste comando?

(R) Agregar las extras líneas escritas en el teclado a archivos.

Y el siguiente:
$ ls -l archivo* >> archivos

(R) Agregar (append) la salida del comando ls a archivos.

Este es un método muy común de añadir líneas al archivo de configuración, en su directorio home, .profile o .bash_profile sin tener que recurrir al uso de vi. Siempre asegúrese de usar los caracteres dobles (>>), y no el simple (>).

Asi:
$ cat >> .profile
lineas adicionales que deseamos agregar al archivo .profile
^D

(R) Para agregar desde la línea de comandos se hace asi:
$ echo “lineas adicionales que deseamos agregar al archivo .profile” >> .profile
$

2. $ wc /etc/* 2>&1 > archivos.etc

¿Porqué seguimos viendo mensajes de error en pantalla? ¿Qué debió haber escrito para enviar los mensajes de error al mismo archivo que la salida estándar?

(R) El flujo de error debe ser redireccionado después que la salida estándar; sino, los errores se direccio-

narán a la pantalla. La forma correcta del comando es:

$ wc /etc/* > archivos.etc 2>&1

¿Qué hace el siguiente comando?:

$ wc /etc/* > archivos.etc 2>&1 > archivos2.etc

(R) Este comando envia los errores al archivos.etc y la salida estándar a arcivos2.etc. El shell le permite redireccionar el flujo estándar de I/O más de una vez desde una sóla línea de comando. La última redirección entrada es la que se aplica; debe tener mucho cuidado al, por lo general, no es una buena idea redireccionar el I/O más de una vez desde una línea de comando.

3. Establezca la opción de noclobber para la redirección del I/O del shell y escriba los siguientes comandos:

$ set -o noclobber
$ cat archivol > archivo.nuevo
$ cat archivo2 > archivo.nuevo
$ cat archivo2 >> archivo.nuevo
$ cp archivo2 archivo.nuevo

¿Cuáles comandos funcionaron y porqué?

(R) Con la opción noclobber habilitada, el shell no puede sobrescribir un archivo existente; ella puede, como quiera, agregar a un archivo, noclobber sólo aplica a las redirecciones del shell de las I/O; esto no afectaría los como es el cp.

El segundo comando cat el de archivo2 debe ser el único que falle.

¿Qué cree usted que se logra con el próximo comando?

$ cat archivo1 > | archivo.nuevo

(R) Este comando ignora el modo de noclobber, permitiendo el shell sobrescribir un archivo aunque la opción de noclobber ha sido establecida.

4. ¿Cree usted que el siguiente comando funcione?

$ 2> /dev/null < archivo.nuevo >> /tmp/archivo.log cat

(R) Si funcionó

¿Puede usted explicar?

(R) De la manera que el shell hace su parseo de esta línea el comando cat, será lo primero que se ejecutará.

De hecho, en este ejemplo, este será el unico contenido en la línea.Recuerde los siguientes puntos:

- El mecanismo de redirección es manejado por el shell, no el comando.
- Todos los caracteres especiales, incluyendo los símbolos de redirección y sus argumentos, son removidos de la línea de comando por el shell.

Ejercicios 8-3: Tuberías y Filtros

1. Listemos directorios con la asistencia de filtros:

$ ls -l /usr/bin | more
$ ls -l /usr/bin | wc -l
$ ls -l /usr/bin | grep root | more
$ ls -l /usr/bin | tee lsbin.txt | grep root | more
$ more lsbin.txt
$ ls -l /usr/bin | grep root | | tee lsbin.txt | more
$ more lsbin

2. Usando los comandos aprendidos, escriba un comando asistido por tuberías que nos reporte cuántos usuarios se encuentran ingresados en el sistema (login). No debe listar los nombres, sólo el total.
   $ who | wc -l

3. Experimentemos diferentes maneras de ver un mismo archivo:
   $ grep bash /etc/passwd | sort |more
   $ cut -d: -f1 /etc/passwd | sort |more

4. Para ésta práctica deberá crear un archivo en el editor (vi) y llamarlo estudiantes.txt, éste archivo debe tener el siguiente formato de texto y separado por coma, y tener los campos de data que listamos más adelante. Este archivo nos servirá para los siguientes ejemplos, así pues que primero, escriba uno con los nombres del ejemplo que aquí se provee:
   - Nombre
   - Matricula
   - Zona
   - Edad
   - Barrio
   - Año de Nacimiento
   - Año de Inscripción
   - Carrera
   - Mención

   Si tenemos un campo el cual no ésta definido, deberá usar un guión (-), para ocupar su lugar. Si un alumno cursa más de una mención, columnas adicionales deberá ser añadidas por cada mención adicional, así pues que, no todas las entradas tienen el mismo número de columnas.

   Aquí le mostramos entradas de ejemplo, para que usted pueda crear su propio archivo de práctica:
   Jose Paredes,19750726,Interior,27,Herrera,1975,Sistema,Programacion,Base de datos

   En éste ejemplo vemos el nombre (Nombre Apellido) seguido por su fecha de nacimiento (añomesdia), su Zona de procedencia (Interior o Capital), etc. debe crear un buen número de entradas en su archivo por cuestión de brevedad lo incluiremos en el Apéndice FFFFF.

   Usando los lectores paginadores less o more, examine el archivo que digitó en vi, para familiarizarse con su contenido.

5. ¿Cuál sentencia de grep nos mostraría todos los alumnos de la Capital?
   $ grep Capital estudiantes.txt

   ¿Cuál sentencia nos mostraría los de la carrera de Sistemas?
   $ grep Sistemas estudiantes.txt

   Usando grep, muestre todos los estudiantes que no estudian Sistemas.
   $ grep -v Sistemas estudiantes.txt

6. Usando el comando cut muestre las provincias y las edades de los estudiantes.
   $ cut -d, -f1-2 estudiantes.txt

   ¿Cuál comando nos mostraría el nombre del estudiante y su Zona?
   $ cut -d, -f1,3 estudiantes.txt

   ¿Cuál comando nos mostraría las Zonas, matriculas y carreras?
   $ cut -d, -f3-4,8 estudiantes.txt

   Liste todos los alumnos, barrios, y las menciones.
   $ cut -d, -f1,5,9- estudiantes.txt

   Liste las primeras tres letras de cada nombre de estudiante.

$ cut -c1-3 estudiantes.txt

7. Usando los comandos cut y grep, muestre los estudiantes que estudian Sistemas, despliegue el nombre del estudiante y las todas las menciones.
$ cut -d, -f1,9- estudiantes.txt | grep Sistema

Muestre todos los nombres de estudiantes cuya zona de procedencia es la Capital.
$ grep 'Capital' estudiantes.txt | cut -d, -f1,3

Muestre edad, fechas de nacimiento, y carrera para todos los estudiantes de la Capital.
$ grep 'Capital' estudiantes.txt | cut -d, -f1-2,9-

8. ¿Cuál es la sentencia del comando sort para ordenar alfabéticamente por nombre de estudiante?
$ sort estudiantes.txt

¿Cuál sentencia del comando sort ordenaría por matriculas de menor a mayor?
$ sort -t, +1n -2 estudiantes.txt

¿Cuál sentencia del comando sort ordenaría con la mayor matricula primero?
$ sort -t, +1nr -2 estudiantes.txt

¿Cuál sentencia de sort lo ordenaría por orden de fecha de inscripción?
$ sort -t, +5n -6 estudiantes.txt

9. Muestre todos los estudiantes de Sistemas, las edades, y las carreras de todos los estudiantes del interior, ordenados alfabéticamente por el nombre del estudiante.
$ grep 'Sistemas' estudiantes.txt | cut -d, -f1-2,9- | sort

Muestre los estudiantes de Sistemas, edades y las carreras, ordenadas por las edades de los estudiantes.
$ grep 'Sistemas' estudiantes.txt | cut -d, -f1-2,9- | sort -t, +1n -2

Muestre los estudiantes del Interior, listando sus fechas de inscripción y ordenadas en orden reversa.
$ grep -v 'Capital' estudiantes.txt | cut -d, -f1,6 | sort -t, +1nr -2

10. Cuente el número de estudiante de la Capital.
$ grep 'Capital' estudiantes.txt | wc -l

Cuente cuántos estudiantes son de Sistemas y del interior.
$ grep 'Sistemas' estudiantes.txt | grep -v 'Capital' | wc -l

11. Muestre todas las diferentes carreras del archivo completo.
$ cut -d, -f5 estudiantes.txt | sort | uniq

Muestre todas las diferentes menciones.
$ cut -d, -f8 estudiantes.txt | sort | uniq

Ejercicios 8-4: Tuberías y Filtros (Cont...)

1. Muestre todos los nombres de estudiantes y la Zona del Interior ordenada por Zonas.
$ cut -d, -f1,5 estudiantes.txt | sort -t, +1 -2 +0 -1

Muestre todos los estudiantes, listando sus nombres, zona, y su carrera oficial; ordenado, primero por carrera, y entonces, la zona, y finalmente, por el nombre del estudiante.
$ cut -d, -f1,5,8 estudiantes.txt | sort -t, +2 -3 +1 -2 +0 -1

Repita el último filtro pero no incluya esos estudiantes que no tienen menciones.
$ cut -d, -f1,5,8 estudiantes.txt | grep -v '-' | sort -t, +2 -3 +1 -2 +0 -1

2. Muestre los 5 primeros estudiantes, por mayoría de edad, listando sólo el nombre y la edad.

$ cut -d, -f1,2 estudiantes.txt | sort -t, +1nr -2 | head -10

Muestre sólo los nombres de los estudiantes y el barrio, pero, ordénelo por la edad, sin mostrarla.
$ cut -d, -f1,3-4 estudiantes.txt | sort -t, +2nr -3 | cut -d, -f1,2

Muestre sólo los estudiantes y su zona, pero, ordénelo por la edad del estudiante y sólo despliegue los cinco de mayor edad.
$ cut -d, -f1,2,5 estudiantes.txt | sort -t, +2nr -3 | head -10 | cut -d, -f1,2

3. Liste el estudiante de mayor edad de cada Zona.
$ cut -d, -f1,2,5 estudiantes.txt | # Corta por Carrera, Region, Edad
sort -t, +2 -3 +1nr -2 | # Ordena por Edad, luego Region
sort -t, -um +2 -3 # Ordena por Región única

Ejercicios 8-5: Shell Scripts

1. Déle permisos de ejecución al script, para ser ejecutado:
$chmod +x rev_usuario
./rev_usuario nombre-del-usuario

El comando pasa información de quien esta ingresado en el sistema al comando grep. El comando sólo imprime las líneas que contienen el primer argumento al programa.

Por ejemplo:
$ rev_usuario ivelis

Imprimirá cualquier línea del comando **who** que contenga la cadena ivelis.

El comando no funciona cuando cambiamos de directorios porque el directorio mis-scripts no está en su ruta o PATH. Anteriormente le funcionó porque el directorio mis-scripts era su directorio actual de trabajo y por ende estaba en su PATH. Para poder agregar a mis-scripts a su ruta de ejecución simplemente ejecute este comando:
$ PATH=$PATH:$HOME/mis-scripts

Si lo desea hacer permanente agreguelo a su archivo de perfil .profile.

2. Para listar un directorio en la línea de comandos:
ls =FC $1

Para permitir que más de un directorio sea nombrado en la línea de comandos:
ls =FC $*

3. Copie todos los scripts desde el directorio mis-scripts a un directorio bin en su directorio home. Claro, si no está, deberá crearlo.
$ cd
$ mkdir bin
$ cp mis-scripts/* bin
$ rm -rf mis-scripts
$ vi .profile
$ PATH=$PATH:$HOME/bin

4. $ cp -i $1 $2

Establezca que el script sea ejecutable:
$ chmod +x mi_copia

Mejorado para que el segundo parámetro sea opcional, o sea que podamos copiar asi:
$ mi_copia /usr/share/doc/Readme.txt

```
src="$1"
dest="$2"
if [[ -z "$dest" ]]; then
dest=.
fi
cp -i $src $dest
```

Mejorado para que nos pida los parámetros en el prompt:

```
src="$1"
dest="$2"
if [[ -z "$dest" ]]; then
echo -n "¿Que archivo deseas Copiar? "
read src
if [[ ! -r "$src" ]]; then
echo "$0: No se puede leer el archivo $src"
exit
fi
fi
if [[ -z "$dest" ]]; then
echo -n "¿Dónde?"
read dest
if [[ -z "$dest" ]]; then
dest=.
fi
fi
cp -i $src $dest
```

5. Ahora vamos a mejorar el script rev_usuario que creamos en la primera pregunta para que use un bucle if para probar el éxito de las tuberías y dar como salida un mensaje de "Ingresó al Sistema" o "No Ingresó al Sistema", dependiendo del éxito o el fracaso del bucle.

```
# Comando para revisar si el usuario ingreso al sistema
# Fijese que los corchetes encierran un <tab> y un < espacio>,
# ya que esto es lo que usa el comando who para separar los campos.
if [[ $# != 1]]; then
echo "$0: Error Missing username"
echo "Uso: $0 username"
exit 1
fi
if who | grep "^$1[ ]" > /dev/null
then
echo "$1 Esta logueado"
else
echo "$1 No especifico nombre de usuario"
fi
```

6. Mejore ahora, de nuevo, el script rev_usuario de la pregunta anterior para que permita múltiples argumentos desde la línea de comandos y con el uso de bucle for pasará por cada argumento, revisando si cada usuario especificado ha ingresado (logged in) al sistema con éxito.

```
# Comando para chequear que los usuarios nombrados estan logueados en el sistema.
#
if [[ $# = 0 ]]; then
echo "$0: Error no especifico nombre de usuario..."
echo "Uso: $0 username"
exit 1
fi
for user in $*;do
# Fijese que los corchetes encierran un <tab> y un < espacio>
```

```
if who | grep "^$1[ ]" > /dev/null
then
echo "$1 Esta logueado"
else
echo "$1 No especifico nombre de usuario"
fi
done
```

Pruebe el comando de la siguiente manera:

```
$ rev_usuario root nombre_usuario1 nombre_usuario2 ....
```

7-

```
# Ejecuta el comando mailx y asigna toda su salida a
# los cabezales de las variables
#
headers='mailx -H'
# Si la variable no es de longitud cero
if [[ -n "$headers"]]
then
echo "Aqui estan los encabezados de sus mensajes"
echo "$headers"
else
echo "No hay mensajes para hoy"
fi
```

Ejercicio 8-6; Más Scripts del Shell

1. Escriba un script que liste los archivos en el directorio actual, y que le indique si el archivo es un directorio, un archivo ejecutable, o si es un archivo normal. Dé salida a cada archivo en su propia línea, empezando con los siguientes simbolos especiales:

```
/ Si es un Directorio
* Si es un Ejecutable
- Si es un archivo Normal (todos los otros)
# Comando para listar los archivos indicando su tipo
if [[ $# = 1 ]]
then
cd $1
status=$?
if [ $status !=0]
then
exit $status
fi
elif [[ $# !=0 ]]
then
echo "$0: Error demasiado argumentos"
echo "uso: $0 [directorio]"
exit 1
fi
# Este bloque es una version modificada la cual nos coloca en el directorio correcto
# No se uso una sentecia else asi que si no hay parametros,
# entonces sales sin cambiar de directorio
# Regresa a la version premodificada
#
echo "Archivo en el directorio 'pwd'"
for file in *
do
# Debe probar si el directorio existe antes de ejecutar
```

```
if [[-d $file]]
then
echo "/$file"
# Con el sufijo '-' echo "${file}/"
elif[[-x $file]]
then
echo "*$file"
# Con el sufijo '-' echo "${file}"
else
echo "-$file"
# Con el sufijo '-' echo "${file}-"
fi
done
```

Use un bucle for y un comodín (wildcard) para pasar por la lista de archivos. Modifique el script para que despliegue el símbolo al final del nombre del archivo como lo hace el comando ls -F.

Modifique el script para que tome un directorio como un parámetro y que use el directorio actual si ninguno es suplido como argumento en la línea de comando.

2. Escriba un script, llámelo agregar_path, script que agrega un único parámetro a la ruta (PATH) actual, e imprime a la consola el nuevo valor de la variable PATH. Asegúrese de que el script escribe un error a pantalla si, el script, se ejecuta con cualquier otra cosa que no sea un sólo parámetro. Ponga el script a prueba, recuerde que el script se ejecuta en un subshell, así que no modificará su shell actual.

   Crée un alias y llámelo path (minúscula), el cuál, se ejecutará en el ambiente del shell actual, usando el comando (.). Ponga su script a prueba para asegurarse de que la variable PATH ha sido extendida en su ambiente actual de trabajo. Escriba algo como lo que sigue a continuación:

```
$ path /usr/sbin
PATH=/bin:/usr/bin:/home/usuario/bin:.:/usr/sbin
$ echo $PATH
/bin:/usr/bin:/home/usuario/bin:.:/usr/sbin
```

   Modifique su script para que acepte múltiples argumentos de directorio y que imprima el valor actual, si no se suple ningún argumento.

   Defina su alias asi:

```
$ alias ap=". agregar_path"
```

   Script requerido agregar_path:

```
# Script para anadir directorio al path
# Disenado para se ejecutado desde '.'
if [[ $# !=1]]; then
echo "$0: Error omitio directorio"
echo "uso: $0 directorio"
exit 1
fi
PATH= $PATH:$1
echo "PATH= $PATH"
```

   Mejorado para argumentos variables:

```
# Script para anadir directorio al path
# Disenado para se ejecutado desde '.'
for dir in $*; do
PATH= $PATH:$dir
done

echo "PATH= $PATH"
```

Modifique su script para aceptar a -d como un parámetro. En éste caso que elimine el último directorio en la variable PATH.

Mejorado para el argumento -d:

```
# Script para anadir/remover directorio al path
# Disenado para se ejecutado desde '.'
#Verifique que haya un solo argumento -d
if [[ $# =1 && "$1" = -d ]]; then
PATH ='echo $PATH | sed 's/:[^:]*$//"
fi
for dir in $*; do
PATH=$PATH:$dir
done
echo "PATH=$PATH"
```

Modifique su script para que acepte el parámetro -i.

Mejorado para el argumento -i:

```
# Script para anadir/remover directorio al path
# Disenado para se ejecutado desde '.'
#Verifique que haya un solo argumento -d o -i
if [[ $# =1 ]]; then
if [["$1" = -d ]]; then
PATH= 'echo $PATH | sed 's/:[^:]*$//"
elif [["$1" = -i ]];then
# Cambia los ':' por espacio
SPATH= 'echo $PATH | sed 's/:/ /g"
# Fije el nuevo path
NPATH=
```

En éste caso, que imprima a pantalla cada componente de la ruta y que pida al usuario que escriba "d" para eliminar el directorio; y sino, el directorio es retenido. Reconstruya un nuevo PATH basado en las respuestas del usuario.

3. Defina una función del shell que localice un archivo en particular en un directorio elegido. Aunque no se ha cubierto profundamente el tema de las funciones, le damos un ejemplo para que usted sigua su sintaxis.

Observe los espacios alrededor de la llaves ({}).

```
$fn() { find $1 -name "$2" -print 2>/dev/null; }
```

¿Puede usted descifrar cómo trabaja esta función?

(R) Las funciones son parte del ambiente actual. Las acciones pueden ser efectuadas por una función son listadas dentro de las llaves. Dentro de las llaves se pueden incluir todos los comandos necesarios. Una gran ventaja de las funciones es que entiende los parámetros posicionales.

El sistema posee un archivo debajo de la estructura de directorio /etc de nombre inittab. Ejecute su función para encontrarlo.

(R) Para encontrar el archivo inittab en la estructura de directorio /etc escriba la sentencia asi:

```
$ fn /etc inittab
```

Para ejecutar una función, escriba su nombre sin los paréntesis (seguido por ...?)

Las funciones expiran cuando usted sale del sistema, asi que para que sean parte de su ambiente, debe incluirlo en su archivo de perfil (.profile).

PREGUNTAS POST-EXAMEN

1. ¿Cuál comando usaría usted para traducir de mayúsculas a minúsculas?
   tr

2. ¿Cómo puede usted lograr que la salida estándar y la de error, ambas, sean enviadas a un archivo de nombre ls.salida?
   $ ls 2>&1 > ls.salida

3. Defina la primera línea de un script bash.
   #!/bin/bash

4. ¿Cómo podemos usar el comando w para imprimir los usuarios ingresados al sistema desde el dominio abc.com?
   $ w | grep abc.com

5. ¿Qué símbolo podemos usar en vez de test en un script de bash?

   (R) La llave cuadrada o brackets ([) trabaja igual que el comando test en bash.

CAPÍTULO 9

PREGUNTAS PRE-EXAMEN

1-¿Qué es vi?

(R) Un editor de texto.

2.- ¿Qué es emacs?

(R) Un editor de texto, que compite con vi.

3-¿Cómo puede ser grabado en vi, un archivo con un nombre diferente?

:w nombre-del-archivo

4-¿Puede usted verificar faltas ortográficas en vi?

(R) NO

5- Usando el editor vi, nombre maneras diferentes de entrar en el modo de inserción de texto.

a	Agregar después del cursor
A	Agregar al final de la línea
i	Insertar antes del cursor
I	Insertar al principio de la línea
o	Abrir una nueva línea debajo del cursor
O	Abrir una nueva línea encima del cursor

6-¿Cómo usted entra en modo de insertar en el editor pico?

(R) Pico siempre está en modo de iserción de texto.

PREGUNTAS POST-EXAMEN

1.- Describa la función del comando view archivo.txt

(R) Abre a archivo.txt con el editor vi en modo de solo lectura.

2.- ¿Cúal secuencia de comandos puede ser usada para desplegar todas las líneas que empiezan con número en el editor vi?
:set number

3.- ¿Cómo puede usted ingresar el contenido del archivo /etc/passwd en el archivo actual que esta editando dentro de una sessión de vi?
:r /etc/passwd

4.- ¿Cómo puede usted insertar la fecha y hora en la sessión actual de vi?
:r!date

5.- ¿Cómo encuentra usted como ejecutar un comando dentro del editor pico?

(R) Si el comando no esta en la línea del prompt en la parte inferior de la pantalla, use CTRL+g para dirigirse a las pantllas de ayuda. ****