

Capítulo 1: Introducción a Linux

Contenido

1.1 Introducción	1
1.2 La evolución del Linux y los sistemas operativos populares	1
1.2.4.1 ¿Qué es un Comando?	6
1.3 Elegir un Sistema Operativo	8

1.1 Introducción

En este capítulo vamos a explorar la evolución del Linux® y los sistemas operativos populares. También vamos a hablar sobre las consideraciones para elegir un sistema operativo.

Linux es de Código Abierto. ¿Qué significa eso? El código que impulsa a Linux no es propiedad de una empresa. En cambio, lo desarrolla la comunidad que lo usa. **¿Por qué es esto bueno?** Libera a los usuarios de los costos de licencia y permite modificar el código según las necesidades cambiantes.

Linux® es una marca registrada de Linus Torvalds en los Estados Unidos y otros países.

1.2 La evolución del Linux y los sistemas operativos populares

La definición de la palabra Linux depende del contexto en el que se utiliza. Linux se refiere al kernel. Es el controlador central de todo lo que pasa en el equipo (veremos más detalles a continuación). Quienes dicen que su equipo "se ejecuta con Linux" generalmente se refiere al kernel y el conjunto de herramientas que vienen con él (llamados distribución). Si tienes "Experiencia con Linux", probablemente te refieres a los propios programas, aunque dependiendo del contexto, podrías hablar sobre tu capacidad de ajustar con precisión el kernel. Cada uno de estos componentes será explorado para que puedas entender exactamente qué papel juega cada uno.

El término que más complica las cosas es UNIX. UNIX era originalmente un sistema operativo desarrollado en los laboratorios de Bell AT&T en la década de 1970. Éste fue modificado y bifurcado (es decir, las personas lo modificaron y

estas modificaciones sirvieron de base para otros sistemas). En la actualidad hay muchas variantes de UNIX. Sin embargo, UNIX es ahora una marca registrada y una especificación, propiedad de un consorcio industrial llamado Open Group. Sólo el software que ha sido certificado por el Open Group puede llamarse UNIX. A pesar de la adopción de todos los requisitos de la especificación de UNIX, Linux no ha sido certificado. ¡Eso significa que Linux realmente no es un UNIX! Es sólo... como UNIX.

1.2.1 Rol del Kernel

El kernel del sistema operativo es como un controlador de tráfico aéreo en un aeropuerto. El kernel determina que programa obtiene que pedazos de memoria, arranca y mata a los programas, y se encarga de mostrar texto en un monitor. Cuando una aplicación necesita escribir en disco, debe pedir al sistema operativo que lo haga. Si dos aplicaciones piden el mismo recurso, el kernel decide cuál de las dos lo recibe y en algunos casos, mata a una de las aplicaciones para salvar el resto del sistema.

El kernel también se encarga de cambiar entre aplicaciones. Un equipo tendrá un pequeño número de procesadores CPU y una cantidad finita de memoria. El kernel se encarga de descargar una tarea y cargar una nueva si hay más tareas que CPUs. Cuando la tarea actual se ha ejecutado una cantidad suficiente de tiempo, la CPU detiene la tarea para que otra pueda ejecutarse. Esto se llama multitarea preferente. Multitarea significa que la computadora realiza varias tareas a la vez, preferente significa que el kernel decide cuándo cambia el enfoque entre las tareas. Con las tareas de conmutación rápida, parece que el equipo está haciendo muchas cosas a la vez. Cada aplicación puede pensar que tiene un bloque grande de memoria en el sistema, pero es el kernel que mantiene esta ilusión, reasignando bloques más pequeños de memoria, intercambiando bloques de memoria con otras aplicaciones, o incluso sacando al disco bloques que aún no se hayan tocado.

Cuando el equipo arranca, carga un pequeño trozo de código llamado gestor de arranque. El gestor de arranque debe cargar el kernel y arrancarlo. Si estás más familiarizado con sistemas operativos como Microsoft Windows y Apple OS X, probablemente nunca ves al gestor de arranque, pero en el ambiente de UNIX es generalmente visible por lo que puedes modificar la manera en la que tu equipo arranque.

El gestor de arranque carga el kernel de Linux y luego transfiere el control. Linux continúa con el funcionamiento de los programas necesarios para hacer que el equipo sea útil, tales como conexión a la red o abrir un servidor web.

1.2.2 Las Aplicaciones

Al igual que un controlador de tráfico aéreo, el kernel no es útil sin tener algo que controlar. Si el kernel es la torre, las aplicaciones son los aviones. Las aplicaciones mandan peticiones al kernel, en cambio, éste recibe recursos tales como memoria, CPU y disco. El kernel también abstrae los detalles complicados de la aplicación. La aplicación no sabe si un bloque de disco es una unidad de estado sólido de fabricante A, un disco duro metálico de spinning del fabricante B, o incluso, alguna parte del archivo de red. Las aplicaciones sólo tienen que seguir la Interfaz de Programación de Aplicaciones (API - Application Programming Interface) del kernel y a cambio no tienen que preocuparse por los detalles de implementación.

Cuando nosotros, como usuarios, pensamos en aplicaciones, tendemos a pensar en los procesadores de texto, navegadores web y clientes de correo electrónico. Al kernel no le importa si se está ejecutando algo orientado al usuario, es un servicio de red que se comunice con un equipo remoto, o una tarea interna. Por lo tanto, de esto obtenemos una abstracción llamada proceso. Un proceso es solamente una tarea que está cargada y rastreada por el kernel. Una aplicación puede necesitar incluso múltiples procesos para funcionar, por lo que el kernel se encarga de ejecutar los procesos, los arranca y para según lo requerido, y entrega los recursos del sistema.

1.2.3 Rol de Código Abierto

Linux comenzó como un proyecto de pasatiempo por Linus Torvalds en 1991. Hizo la fuente disponible libremente y otros se unieron para formar este sistema operativo de vanguardia. Su sistema no fue el primero desarrollado por un grupo. Sin embargo, ya que fue un proyecto creado desde cero, los primeros usuarios podían influir el rumbo del proyecto y asegurarse de que no se repitieran los errores de otros UNIXes.

Los proyectos de software toman la forma de código fuente, que es un conjunto de instrucciones de computo legibles para el humano. El código fuente puede escribirse en cualquiera de los cientos de lenguajes diferentes, Linux ha sido escrito solamente en C, que es un lenguaje que comparte historia con el UNIX original.

El código fuente no se entiende directamente por el equipo, por lo que debe ser compilado en instrucciones de máquina por un compilador. El compilador reúne todos los archivos fuente y genera algo que se puede ejecutar en el equipo, como el kernel de Linux.

Históricamente, la mayor parte del software se ha publicado bajo una licencia de código cerrado, lo que significa que obtienes el derecho a utilizar el código de máquina, pero no puedes ver el código fuente. ¡A menudo la licencia dice específicamente, que no se intente revertir el código máquina al código de fuente

para averiguar lo que hace!

El Código Abierto toma una vista centrada en la fuente del software. La filosofía de código abierto es que tienes derecho a obtener el software y modificarlo para tu propio uso. Linux adoptó esta filosofía con gran éxito. La gente tomó la fuente, hizo cambios y lo compartió con el resto del grupo.

Junto a esto, fue el proyecto GNU (GNU, no UNIX). Mientras que GNU estaba construyendo su propio sistema operativo, eran mucho más eficaces en la creación de las herramientas que están de acuerdo con el sistema operativo UNIX, como los compiladores y las interfaces de usuario. La fuente era completamente gratuita, así que Linux pudo enfocar sus herramientas y proporcionar un sistema completo. Como tal, la mayoría de las herramientas que forman parte del sistema Linux provienen de estas herramientas GNU.

Hay muchas diversas variantes en código abierto, y los veremos en un capítulo posterior. Todos coinciden en que debes tener acceso al código fuente, pero difieren en cómo puedes, o en algunos casos, cómo debes redistribuir los cambios.

1.2.4 Distribuciones de Linux

Toma las herramientas de GNU y Linux, añade algunas aplicaciones para el usuario como un cliente de correo, y obtienes un sistema Linux completo. Se empezó a empaquetar todo este software en una distribución casi tan pronto como Linux llegó a ser utilizable. La distribución se encarga de configurar el almacenamiento de información, instalar el kernel e instalar el resto del software. Las distribuciones recomendadas completas también incluyen herramientas para administrar el sistema y un administrador de paquetes para añadir y eliminar el software después de la instalación.

Como en UNIX, hay muchos sabores diferentes de distribuciones. En estos días, hay distribuciones que se centran en el funcionamiento en servidores, computadoras de escritorio (desktop) o incluso herramientas específicas de la industria como el diseño de la electrónica o la computación estadística. Los principales actores en el mercado se remontan a Red Hat o Debian. La diferencia más visible es el administrador de paquetes, aunque encontrarás otras diferencias en todo, desde ubicaciones de archivos a filosofías de políticas.

Red Hat empezó como una simple distribución que introdujo el Administrador de Paquetes Red Hat (RPM- Red Hat Package Manager). El desarrollador eventualmente formó una compañía alrededor de éste, que intentó comercializar una computadora de escritorio Linux para negocios. Con el tiempo, Red Hat comenzó a centrarse más en las aplicaciones de servidor web y servicios de archivos, y lanzó Red Hat Enterprise Linux, que era un servicio de pago en un ciclo de liberación largo. El ciclo de liberación dicta con qué frecuencia se

actualiza el software. Una empresa puede valorar la estabilidad y quiere ciclos de liberación largos, un aficionado o un principiante puede querer un software más reciente y optar por un ciclo de liberación más corto. Para cumplir con este último grupo, Red Hat patrocina el **Proyecto Fedora** que hace que el escritorio personal contenga el software más reciente, pero aun construido sobre los mismos principios como la versión para empresas.

Porque todo en Red Hat Enterprise Linux es de código abierto, un proyecto llamado **CentOS** llegó a ser el que volvió a compilar todos los paquetes RHEL y los proporcionó gratis. CentOS y otros proyectos similares (como Scientific Linux) son en gran parte compatibles con RHEL e integran algún software más reciente, pero no ofrecen el soporte pagado que Red Hat si ofrece.

Scientific Linux es un ejemplo de una distribución de uso específico basada en Red Hat. El proyecto viene patrocinado por Fermilab siendo una distribución diseñada para habilitar la computación científica. Entre sus muchas aplicaciones, Scientific Linux se utiliza con aceleradores de partículas como el Gran Colisionador de Hadrones en el CERN.

Open SUSE, originalmente derivado de Slackware, aun incorpora muchos aspectos de Red Hat. La compañía original fue comprada por Novell en el año 2003, que entonces fue adquirida por el Grupo Attachmate en 2011. El grupo Attachmate luego se fusionó con Micro Focus Internacional. A través de todas las fusiones y adquisiciones SUSE ha logrado continuar y crecer. Mientras que Open SUSE se basa en escritorio y es disponible al público en general, **SUSE Linux Enterprise** contiene código propietario y se vende como un producto de servidor.

Debian es más bien un esfuerzo de la comunidad y como tal, también promueve el uso de software de código abierto y la adherencia a estándares. Debian desarrolló su propio sistema de administración de paquetes basado en el formato del archivo .deb. Mientras que Red Hat deja sin soporte las plataformas Intel y AMD para proyectos derivados, Debian es compatible con muchas de estas plataformas directamente.

Ubuntu es la distribución derivada de Debian más popular. Es la creación de **Canonical**, una empresa que apoyó el crecimiento de Ubuntu ganando dinero proporcionando soporte.

Linux Mint se inició como una bifurcación de **Ubuntu Linux** mientras sigue dependiendo sobre los repositorios de Ubuntu. Existen varias versiones, todas libres de costo, pero algunas incluyen códigos propietarios que no pueden ser distribuidos sin restricciones de la licencia en algunos países. Linux Mint está suplantando rápidamente Ubuntu como solución de Linux escritorio más popular del mundo.

Hemos tratado el tema de las distribuciones mencionadas específicamente en los objetivos de Linux Essentials. Debes saber que hay cientos, y hasta miles más que están disponibles. Es importante entender que si bien hay muchas diferentes distribuciones de Linux, muchos de los programas y comandos siguen siendo los mismos o muy similares.

1.2.4.1 ¿Qué es un Comando?

La respuesta más simple a la pregunta "¿Qué es un comando?" es que un comando es un programa de software que cuando se ejecuta en la línea de comandos, realiza una acción en el equipo.

Cuando tomas en cuenta un comando utilizando esta definición, en realidad estás tomando en cuenta lo que sucede al ejecutar un comando. Cuando se escribe un comando, el sistema operativo ejecuta un proceso que puede leer una entrada, manipular datos y producir la salida. Desde esta perspectiva, un comando ejecuta un proceso en el sistema operativo, y entonces la computadora realiza un trabajo.

Sin embargo, un comando se puede ver desde una perspectiva diferente: desde su origen. La fuente es desde donde el comando "proviene" y hay varios orígenes diferentes de comandos dentro de shell de la CLI:

- **Comandos integrados en el shell:** Un buen ejemplo es el comando `cd` ya que es parte del bash shell. Cuando un usuario escribe el comando `cd`, el bash shell ya se está ejecutando y sabe cómo interpretar ese comando, sin requerir de ningún programa adicional para iniciarse.
- **Comandos que se almacenan en archivos que son buscados por el shell:** Si escribes un comando `ls`, entonces shell busca en los directorios que aparecen en la variable RUTA DE ACCESO (`PATH`) para tratar de encontrar un archivo llamado `ls` que puede ejecutar. Estos comandos se pueden ejecutar también escribiendo la ruta de acceso completa del comando.
- **Alias:** Un alias puede reemplazar un comando integrado, la función o un comando que se encuentra en un archivo. Los alias pueden ser útiles para la creación de nuevos comandos de funciones y comandos existentes.
- **Funciones:** Las funciones también pueden ser construidas usando los comandos existentes o crear nuevos comandos, reemplazar los comandos integrados en el shell o comandos almacenados en archivos. Los alias y las funciones normalmente se cargan desde los archivos de inicialización cuando se inicia el shell por primera vez, que veremos más adelante.

Para considerar

Aunque los alias serán tratados en detalle en una sección posterior, este ejemplo breve puede ser útil para entender el concepto de comandos.

Un alias es esencialmente un apodo para otro comando o una serie de comandos. Por ejemplo, el comando de `cal 2014` muestra el calendario para el año 2014. Supongamos que acabes ejecutando este comando a menudo. En lugar de ejecutar el comando completo cada vez, puedes crear un alias llamado `mycal` y ejecutar el alias como se muestra en el siguiente gráfico:

```
sysadmin@localhost:~$ alias mycal="cal 2014"
sysadmin@localhost:~$ mycal
2014
```

Enero							Febrero							Marzo						
Do	Lu	Ma	Mi	Ju	Vi	Sá	Do	Lu	Ma	Mi	Ju	Vi	Sa	Do	Lu	Ma	Mi	Ju	Vi	Sá
			1	2	3	4							1						1	
5	6	7	8	9	10	11	2	3	4	5	6	7	8	2	3	4	5	6	7	8
12	13	14	15	16	17	18	9	10	11	12	13	14	15	9	10	11	12	13	14	15
19	20	21	22	23	24	25	16	17	18	19	20	21	22	16	17	18	19	20	21	22
26	27	28	29	30	31		23	24	25	26	27	28		23	24	25	26	27	28	29
														30	31					

Abril							Mayo							Junio						
Do	Lu	Ma	Mi	Ju	Vi	Sá	Do	Lu	Ma	Mi	Ju	Vi	Sa	Do	Lu	Ma	Mi	Ju	Vi	Sá
		1	2	3	4	5					1	2	3	1	2	3	4	5	6	7
6	7	8	9	10	11	12	4	5	6	7	8	9	10	8	9	10	11	12	13	14
13	14	15	16	17	18	19	11	12	13	14	15	16	17	15	16	17	18	19	20	21
20	21	22	23	24	25	26	18	19	20	21	22	23	24	22	23	24	25	26	27	28
27	28	29	30				25	26	27	28	29	30	31	29	30					

Julio							Agosto							Septiembre						
Do	Lu	Ma	Mi	Ju	Vi	Sá	Do	Lu	Ma	Mi	Ju	Vi	Sa	Do	Lu	Ma	Mi	Ju	Vi	Sá
1	2	3	4	5																

1.2.5 Plataformas de Hardware

Linux comenzó como algo que sólo funcionaría en un equipo como Linus': un 386 con un controlador de disco duro específico. El rango de soporte creció gracias a que se empezó a implementar soporte para otros hardware. Finalmente, Linux comenzó a apoyar a otros chips, incluido el hardware que se hizo para ejecutar sistemas operativos competitivos!

Los tipos de hardware crecieron desde el simple chip de Intel hasta supercomputadores. Más tarde se desarrollaron chips de menor tamaño

compatibles con Linux pensados para que quepan en dispositivos de consumo, llamados dispositivos integrados. El soporte para Linux se hizo omnipresente de tal manera que a menudo es más fácil construir hardware para soportar Linux y usar Linux como un trampolín para su software personalizado, que construir el hardware y el software personalizados desde cero.

Finalmente, teléfonos celulares y tabletas empezaron a funcionar con Linux. Una empresa, más tarde comprada por Google, salió con la plataforma Android que es un paquete de Linux y el software necesario para ejecutarse un teléfono o una tableta. Esto significa que el esfuerzo para introducir un teléfono al mercado es significativamente menor, y las empresas pueden pasar su tiempo innovando el software orientado al usuario en lugar de reinventar la rueda cada vez. Android es ahora uno de los líderes del mercado en el espacio.

Además de teléfonos y tabletas, muchos dispositivos de consumo llevan Linux. Los enrutadores inalámbricos normalmente funcionan con Linux porque tiene un gran conjunto de características de red. El TiVo es un grabador de vídeo digital para el consumidor basado en Linux. A pesar de que estos dispositivos tienen Linux en el kernel, los usuarios finales no tiene que saberlo. El software a la medida interactúa con el usuario y Linux proporciona una plataforma estable.

1.3 Elegir un Sistema Operativo

Has aprendido que Linux es un sistema operativo de tipo UNIX, lo que significa que no ha pasado por una certificación formal y por lo tanto no puede usar la marca oficial de UNIX. Hay muchas otras alternativas; algunas son tipo UNIX y algunas están certificadas como UNIX. Existen también sistemas operativos no Unix como Microsoft Windows.

La pregunta más importante para determinar la configuración de una máquina es "¿Qué hará esta máquina?" Si necesitas ejecutar un software especializado que sólo se ejecuta en Oracle Solaris, entonces eso es lo que necesitarás. Si necesitas leer y escribir documentos de Microsoft Office, entonces necesitarás Windows o algo capaz de ejecutar OpenOffice o LibreOffice.

1.3.1 Puntos de Decisión

En primer lugar tienes que decidir que rol debe tener tu máquina. ¿Estará sentado en la consola ejecutando aplicaciones de productividad o navegando la web? Si es así, necesitarás un equipo de escritorio (desktop). ¿Vas a utilizar la máquina como un servidor Web o de otra manera a parte de estar sentado en una estantería de servidor en algún lugar? Estás buscando un servidor.

Los servidores generalmente están en un rack y comparten un teclado y un monitor con muchos otros equipos, ya que el acceso de la consola sólo se utiliza para configurar y solucionar problemas en el servidor. El servidor se ejecutará

en modo no gráfico, que libera recursos para el propósito real de la computadora. Un equipo de escritorio ejecutará principalmente una GUI.

A continuación, debes determinar las funciones de la máquina. ¿Existe software específico que necesita para funcionar, o funciones específicas que debe hacer? ¿Quieres manejar cientos o miles de estas máquinas al mismo tiempo? ¿Cuál es el conjunto de habilidades del equipo que administra la máquina y del software?

También debes determinar la vida útil y la tolerancia de riesgo del servidor. Los sistemas operativos y actualizaciones de software vienen sobre una base periódica, llamada el ciclo de liberación. Los proveedores de software sólo darán soporte a las versiones anteriores del software durante un tiempo antes de que ya no ofrezcan las actualizaciones, lo que se llama ciclo de mantenimiento (o ciclo de vida). Por ejemplo, las versiones principales de Fedora Linux salen aproximadamente cada 6 meses. El Fin de vida (EOL- End of Life) de las versiones se considera después de 2 versiones principales más un mes, por lo que tienes entre 7 y 13 meses después de instalar Fedora antes que necesites actualizaciones. Compara esto con la variante del servidor comercial, Red Hat Enterprise Linux, y puedes utilizarla hasta 13 años sin la necesidad de actualizar.

Los ciclos de mantenimiento y liberación son importantes porque en un entorno de servidor empresarial las actualizaciones importantes del servidor requieren mucho tiempo y por lo tanto se hacen raramente. En cambio, el propio servidor se reemplaza cuando hay actualizaciones importantes o reemplazos de las aplicaciones que requieren actualización del sistema operativo. Del mismo modo, un ciclo de liberación lento es importante porque las aplicaciones a menudo se enfocan en la versión actual del sistema operativo y querrás evitar la sobrecarga de las constantes actualizaciones para mantenerse al día en los servidores y sistemas operativos. Hay una gran cantidad de trabajo involucrada en la actualización de un servidor, y la función del servidor tiene a menudo muchas personalizaciones que dificultan el portar a un nuevo servidor. Esto requiere mucho más pruebas que si sólo se haya actualizado una aplicación.

Si te dedicas al desarrollo de software o al trabajo tradicional de escritorio, a menudo querrás tener el software más reciente. El software más reciente tiene mejoras en funcionalidad y aspecto que contribuyen a que el uso del equipo sea más agradable. Un escritorio frecuentemente guarda su trabajo en un servidor remoto, por lo que el escritorio se puede limpiar e instalar el sistema operativo más reciente con poca interrupción.

Las versiones de software individuales se pueden caracterizar por ser beta o estables. Una de las ventajas de ser un desarrollador de código abierto es que puedes liberar tu nuevo software y rápidamente obtener retroalimentación de los usuarios. Si una versión de software está en una etapa

de muchas funciones nuevas que no han sido rigurosamente probados por lo general se denomina beta. Después de que tales funciones hayan sido probadas en el campo el software se mueve a un punto estable. Si necesitas las últimas funciones, buscarás una distribución que tiene un ciclo de liberación corto y el software beta es de fácil uso. Cuando se trate de un servidor querrás un software estable a menos que las nuevas funciones sean necesarias y no te importe ejecutar código que no haya sido completamente probado.

Otro concepto ligeramente relacionado es la compatibilidad con versiones anteriores. Esto se refiere a la capacidad de un sistema operativo más reciente de ser compatible con el software creado para las versiones anteriores. Esto es generalmente una preocupación si necesitas actualizar tu sistema operativo, pero no estás en condiciones de actualizar el software de la aplicación.

Por supuesto, el costo siempre es un factor. Linux en sí podría ser gratuito, pero tendrías que pagar por soporte dependiendo de las opciones que elijas. Microsoft tiene costo de licencia de servidor y podría aplicar los gastos adicionales de soporte durante la vigencia del servidor. El sistema operativo que hayas elegido puede que sólo se ejecute en un hardware en particular, lo que también afecta el costo.

1.3.2 Microsoft Windows

El mundo de Microsoft divide los sistemas operativos de acuerdo al propósito de la máquina: ¿escritorio o servidor? La edición de escritorio de Windows ha experimentado diversos esquemas de nomenclatura con la versión actual (al momento de escribir esto) siendo ahora simplemente **Windows 8**. Nuevas versiones del escritorio salen cada 3-5 años y tienden a recibir soporte por muchos años. La compatibilidad con versiones anteriores es también una prioridad para Microsoft, llegando incluso a agrupar la tecnología de la máquina virtual para que los usuarios puedan ejecutar software antiguo.

En el mundo de los servidores existe Windows Server. Actualmente hay (hasta la fecha de este texto) la versión 2012 para indicar la fecha de lanzamiento. El servidor ejecuta una GUI, pero en gran parte como una respuesta competitiva a Linux. Ha hecho progresos sorprendentes en la línea de comandos con capacidades de scripting a través de PowerShell. También puedes hacer que el servidor parezca una computadora de sobremesa con un paquete de experiencia de escritorio opcional.

1.3.3 Apple OS X

Apple produce el sistema operativo OS X que pasó por la certificación de UNIX. OS X está parcialmente basado en software del proyecto FreeBSD.

Por el momento, OS X es principalmente un sistema operativo de escritorio, pero

existen paquetes opcionales que ayudan con la gestión de servicios de red que permiten a muchas computadoras de escritorio OS X colaborar, tal como compartir archivos o ejecutar un inicio de sesión de red.

OS X en el escritorio suele ser una decisión personal ya que mucha gente considera este sistema más fácil de usar. La creciente popularidad de OS X ha asegurado un sano soporte de proveedores de software. OS X es también muy popular en las industrias creativas como por ejemplo la producción de vídeo. Es un área donde las aplicaciones manejan la decisión de sistema operativo y por lo tanto la elección de hardware, ya que OS X se ejecuta en el hardware de Apple.

1.3.4 BSD

Hay varios proyectos open source BSD (Berkeley Software Distribution) como OpenBSD, FreeBSD y NetBSD. Estas son alternativas a Linux en muchos aspectos ya que utilizan una gran cantidad de software común. BSD por lo general se implementa en la función del servidor, aunque también hay variantes como GNOME y KDE que fueron desarrolladas para las funciones del escritorio.

1.3.5 Otros UNIX Comerciales

Algunos de los UNIX comerciales más populares son:

- *Oracle Solaris*
- *IBM AIX*
- *HP-UX*

Cada uno de ellos se ejecuta en el hardware de sus respectivos creadores. El hardware es generalmente grande y potente, que ofrece características tales como CPU y memoria o integración de intercambio con sistemas de legado mainframe también ofrecidos por el proveedor.

A menos que el software requiera un hardware específico o las necesidades de la aplicación requieran de la redundancia en el hardware, muchas personas tienden a elegir estas opciones porque ya son usuarios de productos de la compañía. Por ejemplo, IBM AIX ejecuta en una amplia variedad de hardware de IBM y puede compartir el hardware con mainframes. Por lo tanto, encontrarás AIX en empresas que ya tienen una larga tradición de uso de IBM o que hacen uso de software de IBM software como el WebSphere.

1.3.6 Linux

Un aspecto donde Linux es muy diferente a las alternativas, es que después de que un administrador haya elegido Linux todavía tiene que elegir una distribución Linux. Acuérdate del tema 1, la distribución empaca el kernel, utilidades y

herramientas administrativas de Linux en un paquete instalable y proporciona una manera de instalar y actualizar paquetes después de la instalación inicial.

Algunos sistemas operativos están disponibles a través de un único proveedor, como OS X y Windows, con el soporte del sistema proporcionado por el proveedor. Con Linux, hay múltiples opciones, desde las ofertas comerciales para el servidor o de escritorio, hasta las distribuciones personalizadas hechas para convertir una computadora antigua en un firewall de red.

A menudo los proveedores de aplicaciones eligen un subconjunto de distribuciones para proporcionar soporte. Diferentes distribuciones tienen diferentes versiones de las librerías (bibliotecas) principales y es difícil para una empresa dar soporte para todas estas versiones diferentes.

Los gobiernos y las grandes empresas también pueden limitar sus opciones a las distribuciones que ofrecen soporte comercial. Esto es común en las grandes empresas donde pagar para otro nivel de soporte es mejor que correr el riesgo de interrupciones extensas. También, las diferentes distribuciones ofrecen ciclos de liberación a veces tan frecuentes como cada seis meses. Mientras que las actualizaciones no son necesarias, cada versión puede obtener soporte sólo para un periodo razonable. Por lo tanto, algunas versiones de Linux tienen un Periodo Largo de Soporte (LTS- Long Term Support) hasta 5 años o más, mientras que otros sólo recibirán soporte por dos años o menos.

Algunas distribuciones se diferencian entre estables, de prueba y versiones inestables. La diferencia es que la distribución inestable intercambia fiabilidad a cambio de funciones. Cuando las funciones se hayan integrado en el sistema por mucho tiempo y muchos de los errores y problemas hayan sido abordados, el software pasa por pruebas para ser una versión estable. La distribución Debian advierte a los usuarios sobre los peligros de usar la liberación "sid" con la siguiente advertencia:

- "sid" está sujeta a cambios masivos y actualizaciones de librerías (biblioteca). Esto puede resultar en un sistema muy "inestable" que contiene paquetes que no se pueden instalar debido a la falta de librerías, dependencias que no se pueden satisfacer, etc. Usar bajo el propio riesgo!*

Otras versiones dependen de las distribuciones Beta. Por ejemplo, la distribución de Fedora libera las versiones Beta o versiones de su software antes de la liberación completa para minimizar errores. Fedora se considera a menudo una comunidad orientada a la versión Beta de RedHat. Se agregan y cambian las funciones en la versión de Fedora antes de encontrar su camino en la distribución de RedHat Enterprise.

1.3.7 Android

Android, patrocinado por Google, es la distribución Linux más popular del mundo. Es fundamentalmente diferente de sus contrapartes. Linux es un kernel y muchos de los comandos que se tratarán en este curso son en realidad parte del paquete **GNU**(GNU no es Unix). Por esta razón algunas personas insisten en utilizar el término GNU/Linux en lugar de Linux por sí solo.

Android utiliza la máquina virtual **Dalvik** con Linux, proporcionando una sólida plataforma para dispositivos móviles como teléfonos y tabletas. Sin embargo, carece de los paquetes tradicionales que se distribuyen a menudo con Linux (como GNU y Xorg), por lo que Android es generalmente incompatible con distribuciones Linux de escritorio.

Esta incompatibilidad significa que un usuario de RedHat o Ubuntu no puede descargar software de la tienda de Google Play. Además, un terminal emulador en Android carece de muchos de los comandos de sus contrapartes de Linux. Sin embargo, es posible utilizar BusyBox con Android para habilitar el funcionamiento de la mayoría de los comandos.