Software Libre

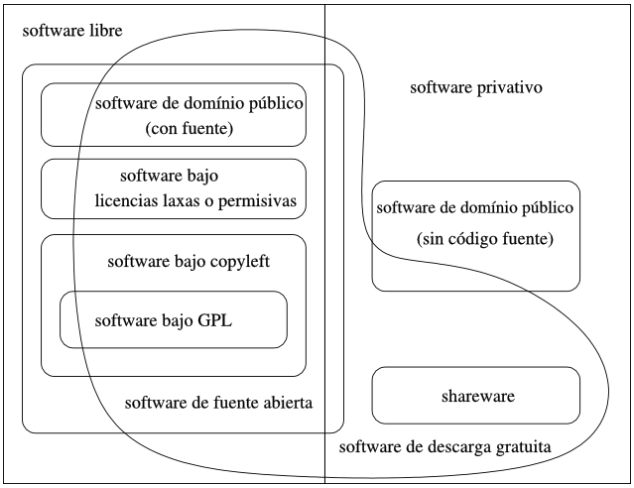
Definición y conceptos

En muchas ocasiones se confunde el verdadero significado del Software Libre. “Software Libre” se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. Para ser más específicos el termino software libre se refiere a que un usuario goza de las siguientes libertades:

- La libertad de usar el programa, con cualquier propósito.
- La libertad de estudiar cómo funciona el programa y adaptarlo a sus necesidades. El acceso al código fuente es una condición previa para esto.
- La libertad de distribuir copias.
- La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. El acceso al código fuente es un requisito previo para esto.

Un programa es un software libre si los usuarios tienen todas estas libertades. Así pues, uno debería 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. También se puede hacer modificaciones para utilizarlas de manera privada o en el trabajo sin tener que anunciar que dichas modificaciones existen.
Esta libertad de usar el software no se limita a personas, sino que también las organizaciones pueden usarlo en cualquier tipo de sistema informático y para cualquier tipo de trabajo.
La libertad de distribuir copias debe incluir tanto en formatos binarios o ejecutables del programa como su código fuente, sean versiones modificadas o sin modificar, ya que ésta es una condición necesaria para el software libre.
Si bien el software libre debe poder distribuirse libremente, pueden definirse reglas sobre la manera de distribuirlo, siempre y cuando no entren en conflicto con las libertades centrales. Así surge el copyleft (en contraposición al copyright)como una regla que implica que al momento de redistribuir el software no se pueden agregar restricciones que nieguen a otras personas las libertades centrales.
“Software libre” no significa “no comercial”. Un programa libre debe estar disponible para uso comercial, desarrollo comercial y distribución comercial.

Categorías del software



Software Libre

El software libre es software que viene con autorización para que cualquiera pueda usarlo, copiarlo y distribuirlo, ya sea literalmente o con modificaciones, gratis mediante una gratificación. En particular, esto significa que código fuente debe estar disponible.

Software de Dominio Público

El software de dominio público software que no está protegido con copyright, ósea no tiene derechos de autor.
Un programa ejecutable puede ser de dominio público pero no disponer libremente del código fuente. En ese caso no es software libre, porque el software libre requiere accesibilidad al código fuente. Por otro lado, la mayoría del software libre no está en el dominio público sino bajo los derechos de autor, y los titulares de esos derechos han dado el permiso legal para que todos puedan utilizarlo en libertad, usando una licencia de software libre.

Software protegido con Copyleft

El software protegido con copyleft es software libre cuyos términos de distribución no permiten a los redistribuidores agregar ninguna restricción adicional cuando éstos redistribuyen o modifican el software.
Copyleft es un concepto general: para poner un programa bajo copyleft, es necesario adoptar un conjunto específico de cláusulas para la distribución. Existen varias maneras de redactar las cláusulas de copyleft, por lo que en principio pueden existir muchas licencias libres con copyleft. Sin embargo, en la práctica, para casi todo el software con copyleft se usa la Licencia Pública General de GNU (GNU General Public License).

Software abarcado por GPL

La GPL(General Public License/Licencia Pública General) de GNU es un conjunto específico de términos de distribución para publicar programas con copyleft.

Software GNU

Es software que es liberado bajo el auspicio del Proyecto GNU. La mayoría del software GNU está protegido con copyleft. Todo el software GNU debe ser software libre.

Software Semilibre

El software semilibre es software que no es libre, pero viene con autorización para particulares de usar, copiar, distribuir y modificar(incluyendo la distribución de versiones modificadas) sin fines de lucro.

Software Propietario

El software propietario es software que no es libre ni semilibre. Su uso, redistribución o modificación está prohibida, o requiere que usted solicite autorización o está tan restringida que no pueda hacerla libre de un modo efectivo.

Freeware

El término “freeware” es usado comúnmente para paquetes que permiten la redistribución pero no la modificación(y su código fuente no está disponible). Estos paquetes no son software libre.

Shareware

El término “shareware” es software que viene con autorización para la gente de redistribuir copias, pero dice que quien continúe haciendo uso de una copia deberá pagar un cargo por licencia.

Software Comercial

El software comercial es software que está siendo desarrollado por una entidad que tiene la intención de hacer dinero del uso del software. “Comercial” y “Propietario” no son la misma cosa. La mayoría del software comercial es propietario, pero hay software libre comercial y hay software no libre no comercial.

Freemium

El modelo freemium es un modelo de negocio en el que la mayor parte de los servicios se ofrecen de manera gratuita (freemium), aunque existe un pequeño paquete de servicios de pago (premium) para algunos clientes que lo deseen.
El modelo freemium es cada día más recurrente y utilizado en las startups y nuevas aplicaciones que van surgiendo. El objetivo principal que se pretende alcanzar con la parte freemium o gratuita es atraer a un gran número de usuarios, realizar una captación masiva de base de datos y que un pequeño porcentaje de estos usuarios paguen por los servicios premium, y serán estos los que rentabilicen el negocio. Las cuentas premium no suelen tener unos precios elevados, son micropagos que aplicados a una gran masa de usuarios consiguen grandes volúmenes de facturación.

Sistema Operativo Linux

Introducción

En una era de cambios en el ambiente computacional, de una amplia oferta en sistemas operativos e interfaces gráficas y sobre todo, del costo que representa contar con un sistema operativo que interactué con el software sin problemas, surge con fuerza inusitada Linux.
Linux es un sistema operativo multitarea y multiusuario que corre en diversas plataformas de hardware: Intel, Alpha, SPARC, MIPS, PowerPC y que se distribuye libremente bajo los términos de la licencia GPL de GNU. Fue creado por Linus Torvalds a comienzos de los ’90 y actualmente es desarrollado y mantenido por una multitud de desarrolladores distribuidos por todo el mundo.
Su objetivo principal es propulsar el software de libre distribución junto con su código fuente para que pueda ser modificado por cualquier persona, dando rienda suelta a la creatividad. El hecho de que el sistema operativo incluya su propio código fuente expande enormemente las posibilidades de este sistema. Este método también es aplicado en numerosas ocasiones a los programas que corren en el sistema, lo que hace que podamos encontrar muchísimos programas útiles totalmente gratuitos y con su código fuente.

Un poco de historia

Linux fue creado originalmente por Linus Torvalds en la Universidad de Helsinki en Finlandia, siendo él estudiante de informática. Pero ha continuado su desarrollo con la ayuda de muchos otros programadores a través de Internet.
Linus originalmente inició el desarrollo del núcleo como su proyecto favorito, inspirado por su interés en Minix, un pequeño sistema Unix desarrollado por Andy Tannenbaum.
Linus nunca anunció la versión 0.01 de Linux(agosto 1991),esta versión no era ni siquiera ejecutable, solamente incluía los principios del núcleo del sistema, estaba escrita en lenguaje ensamblador y asumía que uno tenía acceso a un sistema Minix para su compilación.
El 5 de octubre de 1991, Linus anunció la primera versión "Oficial" de Linux, versión 0.02. Con esta versión Linus pudo ejecutar Bash(GNU Bourne Again Shell) y gcc(El compilador GNU de C) pero no mucho más funcionaba. En este estado de desarrollo ni se pensaba en los términos soporte, documentación o distribución. Después de la versión 0.03, Linus saltó en la numeración hasta la 0.10 y más y más programadores a lo largo y ancho de Internet comenzaron a trabajar en el proyecto y después de sucesivas revisiones, Linus incremento el número de versión hasta la 0.95 (Marzo 1992). Mas de un año después (diciembre 1993) el núcleo del sistema estaba en la versión 0.99 y la versión 1.0 no llegó sino hasta el 14 de marzo de 1994. Desde entonces, muchos programadores han respondido a su llamada y han ayudado a construir Linux como el sistema operativo completamente funcional que es hoy.

Características

Software Libre

Linux se distribuye bajo los términos de la Licencia Pública General de GNU, lo cual garantiza que puede ser usado, estudiado, copiado y modificado sin restricciones de ningún tipo, salvo aquellas que impiden que esto deje de ser así(copyleft).

Multitarea

Se pueden llevar a cabo múltiples tareas(o procesos) en forma simultánea y se pueden acceder múltiples dispositivos al mismo tiempo.

Multiusuario

Varios usuarios pueden acceder a las aplicaciones y recursos del sistema Linux al mismo tiempo. Y, por supuesto, cada uno de ellos puede ejecutar varios programas a la vez(multitarea).

Multiplataforma
Linux puede correr en casi cualquier plataforma Actualmente se ejecuta en plataformas Intel, Sparc, Alpha, MIPS, PowerPC, ARM y hasta en Mainframes de IBM (S/390).

Protección de memoria entre procesos
De manera que ninguno de ellos pueda colgar el sistema.

Carga de ejecutables por demanda
Linux sólo lee de disco aquellas partes de un programa que están siendo usadas actualmente.

Memoria virtual con paginación por demanda
Linux utiliza una porción del disco como memoria virtual, incrementando así la eficiencia del sistema al mantener los procesos activos en memoria RAM y ubicando los procesos que son utilizados menos frecuentemente o porciones inactivas de memoria en el disco.

Librerías compartidas
Cada aplicación comparte una librería común de subrutinas que puede llamar en tiempo de ejecución.
Compatible POSIX.1 y UNIX.

El Kernel

El kernel o núcleo de Linux se podría definir como el corazón de este sistema operativo y cuando hablamos de Linux, en realidad nos deberíamos referir al Kernel. Es el encargado de que el software y el hardware puedan trabajar juntos.
Sus funciones más importantes, aunque no las únicas, son:

- Administración del sistema.
- Administración de la memoria física y virtual para los procesos en ejecución.
- Manejo de periféricos y dispositivos de nuestro ordenador.

Versiones del kernel de Linux
Versión de producción
La versión de producción, es la versión estable hasta el momento. Esta versión es el resultado final de las versiones de desarrollo o experimentales.
Cuando el equipo de desarrollo del kernel experimental, decide que ha conseguido un kernel estable y con la suficiente calidad, se lanza una nueva versión de producción o estable.
Esta versión es la que se debería utilizar para un uso normal del sistema, ya que son las versiones consideradas más estables y libres de fallos en el momento de su lanzamiento.
Versión de desarrollo.
Esta versión es experimental y es la que utilizan los desarrolladores para programar, comprobar y verificar nuevas características, correcciones, etc. Estos núcleos suelen ser inestables y no se deberían usar, a no ser que uno sepa lo que hace.

¿Cómo interpretar los números de las versiones?
Las versiones del kernel se numeran con 3 números, de la siguiente forma: XX.YY.ZZ, donde:
XX: Indica la serie principal del kernel. Hasta el momento sólo existen la 1 y 2. Este número cambia cuando la manera de funcionamiento del kernel ha sufrido un cambio muy importante.
YY: Indica si la versión es de desarrollo o de producción. Un numero impar, significa que es de desarrollo, uno par, que es de producción.
ZZ: Indica nuevas versiones dentro de una versión, en las único que se ha modificado, son fallos de programación o bugs.
Como se dijo anteriormente, Linux es el Kernel creado y mantenido por Linus Torvalds. Pero el kernel por sí solo no podría hacer mucho sin aplicaciones que se ejecuten en él. Es por eso que surgen las distribuciones, como un modo de empaquetar el kernel con un conjunto de aplicaciones de modo tal que se logre un sistema completo y funcional.

Así, múltiples distribuciones han surgido desde el nacimiento de Linux, llegando en la actualidad a varias decenas. Cada distribución tiene una finalidad o filosofía determinada que hace que se destaque sobre las otras. Están las que buscan la facilidad de uso, las que propician el software 100% libre, las que buscan performance, las diseñadas para negocios, etc.
Algunas de las más conocidas son:

- RedHat:http://www.redhat.com
- Debian:http://www.debian.org
- Slackware:http://www.slackware.com
- SuSE:http://www.suse.com
- Gentoo:http://www.gentoo.org

Arranque y cierre del sistema

Una de las características más importantes y poderosas de Linux es el método abierto y configurable para el inicio y cierre del sistema operativo. Los usuarios son libres de configurar muchos aspectos del proceso de arranque, incluyendo qué programas se lanzarán al momento de arranque. De forma parecida, el cierre del sistema finaliza los procesos de forma organizada y configurable, aunque la personalización de este proceso casi nunca es necesaria.

El proceso de arranque

El proceso de arranque en un sistema Linux ocurre de la siguiente manera:

1. La BIOS del sistema comprueba y lanza la primera etapa del gestor de arranque del MBR(por Mater Boot Record) del disco duro primario.
2. La primera etapa del gestor de arranque se autocarga en memoria y lanza la segunda etapa del gestor de arranque desde la partición /boot/.
3. La segunda etapa del gestor de arranque carga el kernel en memoria, el cual en su momento carga los módulos necesarios y monta la partición root para sólo-lectura.
4. El kernel transfiere programa /sbin/init.
5. El programa /sbin/init carga todos los servicios y herramientas de espacio del usuario y monta todas las particiones listadas en /etc/fstab.
6. Se presenta al usuario un prompt de login de comandos para el sistema Linux apenas arrancado.

La BIOS

En el momento de arranque de una máquina x86, el procesador busca al final de la memoria del sistema el programa de la BIOS y lo ejecuta. La BIOS controla no sólo el primer paso del proceso de arranque, sino que también proporciona una interfaz de bajo nivel para dispositivos periféricos.
Una vez que se haya cargado, la BIOS chequea los periféricos y localiza un dispositivo con el que arrancar el sistema desde el que carga en memoria cualquier programa que resida en el primer sector de este dispositivo, llamado Master Boot Record o MBR.
La MBR sólo tiene 512 bytes de tamaño y contiene las instrucciones de código de máquina para el arranque del equipo, llama un gestor de arranque así como también la tabla de particiones. Una vez que la BIOS haya encontrado y cargado el gestor de arranque en memoria, le deja el control del proceso de arranque a éste.

El gestor de arranque

Dependiendo de la arquitectura del sistema, el proceso de arranque diferirá ligeramente. Los gestores de arranque de Linux para la plataforma x86 se dividen en dos etapas.
La primera es un código binario de máquina pequeña en el MBR. Su única función es la de localizar el gestor de arranque de la segunda etapa y cargar la primera parte de éste en memoria.
La segunda etapa del gestor de arranque es usar la información del MBR para determinar las opciones de arranque disponibles para el usuario. Esto significa que cada vez que se produzca un cambio en la configuración o se actualice el kernel de forma manual, se debe ejecutar un comando para escribir la información apropiada al MBR.
Una vez que el gestor de arranque de la segunda etapa está en memoria, presenta al usuario la pantalla inicial mostrando los diferentes sistemas operativos o kernels que se han configurado para arrancar. En esta pantalla el usuario puede usar las flechas direccionales para escoger el sistema operativo o kernel con el que desea arrancar y presionar la tecla [Enter].

Si no se presiona ninguna tecla, el gestor de arranque carga la selección predeterminada luego de un período de tiempo de espera(también configurable). Una vez que el gestor de arranque de la segunda etapa haya determinado qué kernel arrancar, localizará el binario del kernel correspondiente en el directorio /boot/.
El kernel binario es llamado usando el siguiente formato:
/boot/vmlinuz-<kernel-version>
Donde
<kernel-version>: corresponde a la versión del kernel especificada en las configuraciones del gestor de arranque.
El gestor de arranque luego coloca la imagen apropiada de initial RAM disk, conocida como initrd, en la memoria. El initrd es usado por el kernel para cargar controladores necesarios para arrancar el sistema.

El kernel

Cuando el kernel se carga, inmediatamente se inicializa y configura la memoria del ordenador y los diferentes dispositivos de hardware conectados al sistema, incluyendo procesadores, subsistemas de entrada/salida y dispositivos de almacenamiento.
A continuación buscará la imagen initrd en una ubicación predeterminada en memoria, la descomprimirá, la montará y cargará todos los controladores necesarios. A continuación inicializa los dispositivos virtuales relacionados con el sistema de archivos, tal como LVM o software RAID antes de desmontar la imagen del disco initrd y liberar toda la memoria que la imagen del disco ocupó anteriormente.
El kernel luego crea un dispositivo root, monta la partición root como sólo lectura y libera cualquier memoria no utilizada. Llegados a este punto, el kernel está cargado en memoria y operativo. Sin embargo, como no hay aplicaciones de usuario que permitan la entrada significativa de datos al sistema, no se puede hacer mucho más.
Para configurar el entorno /sbin/init.

El programa /sbin/INIT

El programa /sbin/init(también llamado init) coordina el resto del proceso de arranque y configura el ambiente del usuario. Cuando el comando init arranca, se vuelve el padre de todos los procesos que comienzan automáticamente en el sistema Linux.
Primero, ejecuta el script /etc/rc.d/rc.sysinit, que establece la ruta a otros programas, activa el swap, controla los sistemas de archivo y se encarga de todo lo que el sistema necesita tener hecho al momento de la inicialización.
El comando init lee su configuración de /etc/inittab, que describe cómo el sistema debería configurarse en cada nivel de ejecución de SysV init. A continuación, el comando init configura la librería de funciones base /etc/rc.d/init.d/functions. Esto indica el modo en el que se debe empezar o matar un programa y cómo determinar el PID del programa.
El programa init inicia todos los procesos de fondo buscando en el directorio apropiado rc por el nivel de ejecución especificado por defecto en /etc/inittab.
Los directorios rc están numerados para corresponder al nivel de ejecución que represente. Por ejemplo, /etc/rc.d/rc5.d/ es el directorio para el nivel de ejecución 5. Cuando se arranca el nivel de ejecución 5, el programa init consulta el directorio /etc/rc.d/rc5.d/ para determinar qué procesos iniciar o parar.
Casi todos los archivos en /etc/rc.d/rc5.d/ son enlaces simbólicos apuntando a los scripts localizados en el directorio /etc/rc.d/init.d/. Los enlaces simbólicos se usan en cada uno de los directorios rc de manera que los niveles de ejecución puedan ser reconfigurados al crear, modificar y eliminar los enlaces simbólicos sin que afecte a los scripts actuales a los fuese refiere.
El nombre de cada enlace simbólico inicia con K o S. son procesos eliminados en ese nivel de ejecución, mientras que inician por S son procesos iniciados.
El comando init en primer lugar detiene todos los enlaces simbólicos de K en el directorio mediante la ejecución del comando /etc/rc.d/ init.d/<command> stop, en el que <command> es el proceso a matar. A continuación inicia todos los enlaces simbólicos S al ejecutar /etc/rc.d/init.d/<command> start.
Cada uno de los enlaces simbólicos se numera para dictaminar el orden de inicio. Puede cambiar el orden en el que los servicios inician o paran al cambiar el número. Mientras más bajo es el número, más rápido se arrancará. Los enlaces simbólicos con el mismo número se inician de modo alfabético.