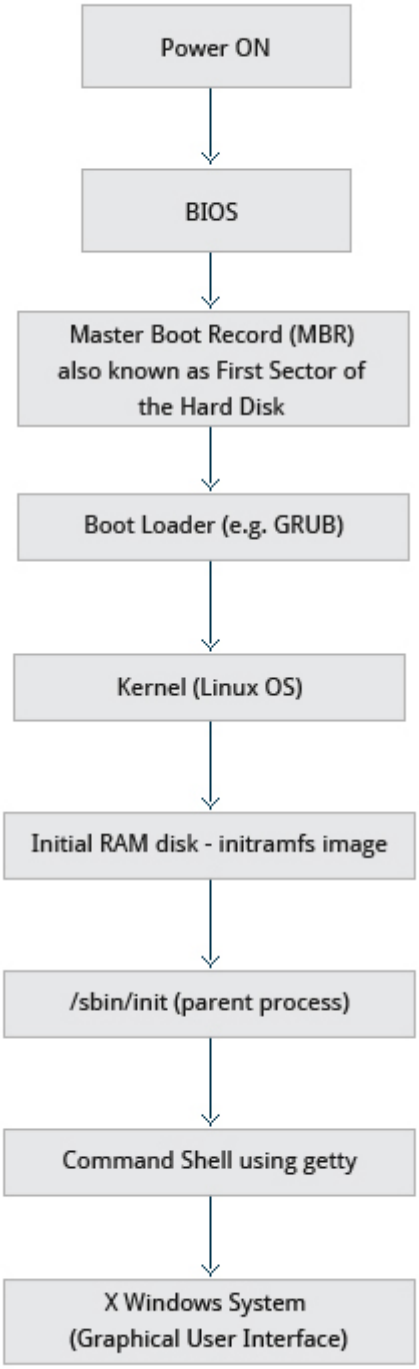
Introducción a Linux

El proceso Boot

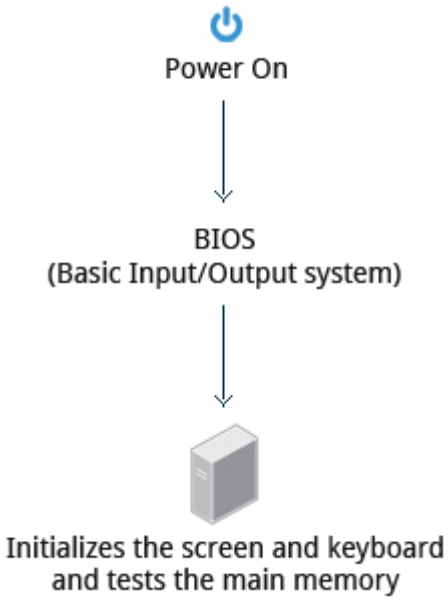
El proceso Boot es el procedimiento que inicializa el sistema. Abarca todo lo que pasa desde que se enciende la computadora hasta que la interfaz de usuario está completamente funcional.

Saber todo lo que abarca el proceso de boot puede ayudarlo con problemas de inicio o de ejecución, también como con adminstrar y mantener el rendimiento de la computadora según los requerimientos del usuario.



Primeros pasos

Encender una computadora con sistema x86(*computadoras de 32bits/ las de 64 bits se les dice x64*) de Linux, requiere una serie de pasos. Cuando la pc es encendida, la Basic Input/Output System(BIOS/Sistema básico de entrada/salida), inicializa el hardware, incluyendo la pantalla y el teclado, y prueba la memoria principal. Este proceso es llamado ***POST***(Power On Self Test/ Auoprueba de poder o encendido).



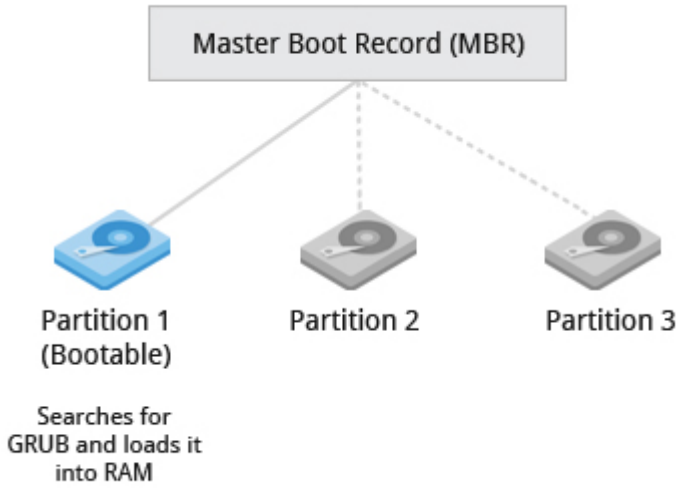
El software o programa de la BIOS esta almacenado en un chip ***ROM(Read Only Memory)*** en la placa madre o motherboard.Luego de esto, lo que queda del encendido lo hace ya, el OS(Operative System/Sistema Operativo).

Master Boot Record (MBR) y Boot Loader

Una vez que el ***POST*** es completado, el control del sistema pasa desde la ***BIOS*** a el ***Boot Loader***. El *Boot Loader*/ *Cargador de Boot* es guardado generalmente en uno de los discos duros del sistema(Hard-drive disk), ya sea en la sección Boot(para sistemas tradicionales de BIOS/MBR), o la partición ***EFI / UEFI (Extensible Firmware Interface / Unified Extensible Firmware Interface)***. En esta parte del proceso de encendido, el sistema aún no accede a ningun archivo de almacenamiento masivo, por eso cualquier información sobre la fecha, hora y los mas esenciales periféricos son cargados desde los valores CMOS(se usan baterias especiales para mantener la memoria de esta infrmación con tal de que se pueda acceder aún despues de apagar el equipo, sin que se borre).

Existen varios cargadores de boot(Boot loaders) para Linux; lo smás comunes son **GRUB(Grand Unified Boot loader/cargador)**, **ISOLINUX**(para cargar linux desde almacenamientos externos como usb o disco), y **DASU-Boot**(para cargar o iniciar en dispositivos/aplicaciones embebidos/ embedded devices/appliances). La mayoría de cargadores boot Linux pueden presentar una interfa

de usuario para escoger que sistema operativo iniciar ya sea de Linux o algun sistema operativo como windows. El cargador boot de Linux también se encarga de cargar o leer la imagen Kernel y el disco inicial RAM disk(Random Access Memory) o archivo de sistema(Filesystem/ el cual contiene archivos esenciales y drivers de dispositivos que se necesitan para iniciar el sistema) en la memoria.

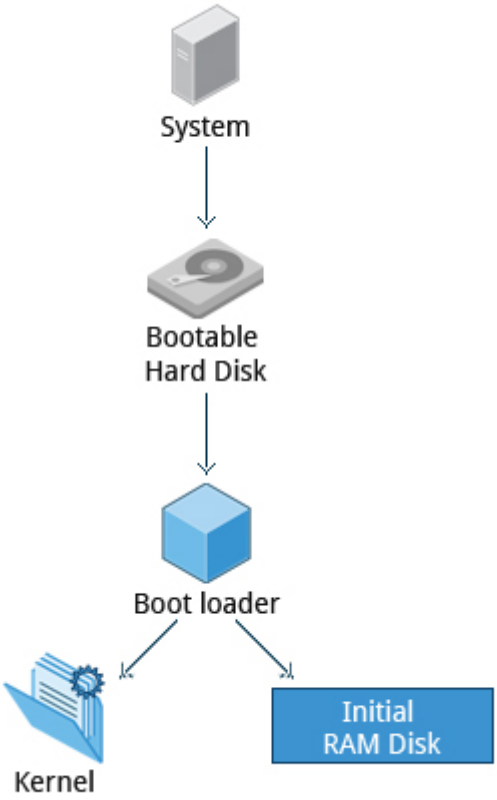


El cargador de boot tiene dos procesos distintos:

Para sistemas usando el método BIOS/MBR, el cargador boot se encuentra en el primer sector del disco duro, también conocido como el Master Boot Record(MBR).Su tamaño es de 512 bytes. En este proceso, el cargador boot examina la **partition table** y encuentra una partición booteable. Una vez que la encuentra, entonces busca el cargador de boot del segundo proceso, por ejemplo GRUB, y lo carga dentro de la RAM.

Para sistemas que usan el método EFI/UEFI, el Firmware UEFI lee los datos del Boot Manager para determinar que tipo de aplicación UEFI va a ser procesada y desde donde(desde que disco y partición el EFI puede ser encontrado). Es entonces cuando el firmware inicia la aplicación UEFI, por ejemplo GRUB, tal cual se definió en la entrada boot en el firmware administrador de boot(firmware's boot manager). Este procedimiento es complicado, pero es más versátil que los métodos viejos MBR.

El cargador boot para el segundo proceso se encuentra bajo la dirección `/boot`, Una pantalla inicial es desplegada, la cual nos permite escoger que sistema operativo(OS) queremos iniciar. Luego de escoger el sistema operativo, el cargador boot, carga el kernel del sistema operatibvo seleccionado dentro de la RAM y y le concede el control. Luego de esto, revisará y analizará el hardware del sistema e inicializará cualquier driver instalado de dispositivos en el kernel.

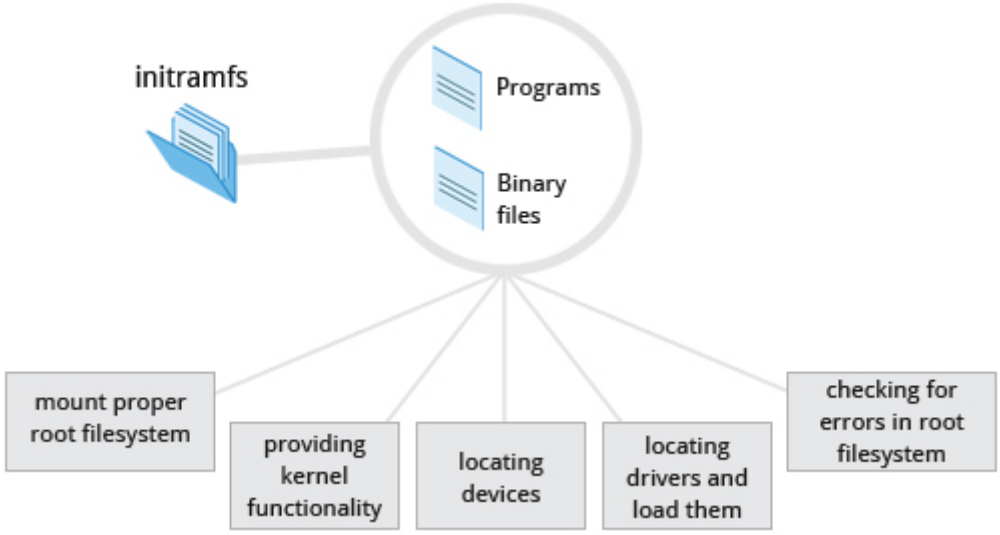


Disco Inicial RAM / Initial RAM Disk

El **initramfs(initial ram file system)** contiene programas y archivos binarios que procesan todas las acciones requeridas para montar el propio archivo root del sistema(root filesystem), como proveer funcionalidad kernel para los filesystem y drivers de dispositivos requeridos para los controles de almacenamiento masivo(mass storage), con una instalación(facility) llamada **udev**(user device), el cual es responsable de saber cuales dipositivos están presentes, localizar los drivers de dispositivos que necesiten para operar apropiadamente, y cargarlos.Después de que el root filesystem ha sido encontrado, se le revisa en busca de errores y luego se monta.

El programa montado instruye al sistema operativo de que un filesystem está listo para uso, y lo asocia con un punto particular en la jerarquía general del filesystem(*el punto de montaje / mount point*). Si funciona bn, el intramfs es borrado de la RAM y el programa **init** es ejecutado en el **root filesystem**(`/sbin/init`).

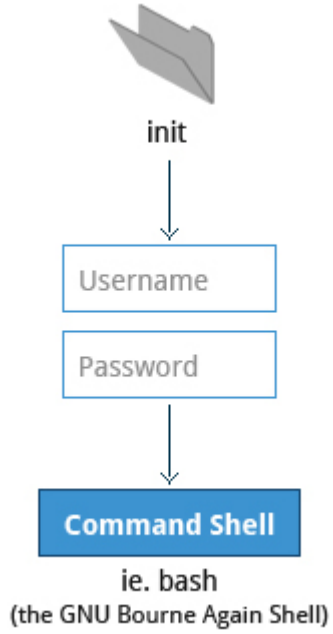
init lleva el montaje(mounting) y el *pivoting* hacia el **principal root filesistem**. Si algúnos drivers de hardware especiales son requeridos antes de que el almacenamiento masivo sea accesado, deben estar en la imagen **initramfs**.



Inicio de sesión text-mode / Text-mode login

Ya finalizando el proceso de boot, **init** inicia un número de peticiones text-mode de inicio de sesión(login). Estas se encargan de permitir escribir el propio nombre de usuario(username), seguido de la contraseña, y eventualmente ejecutar un **command shell**. Sin embargpo si se está ejecutando un sistema con grafical login interface(interfaz gráfica de usuario), no se podrá ver esto al comienzo.

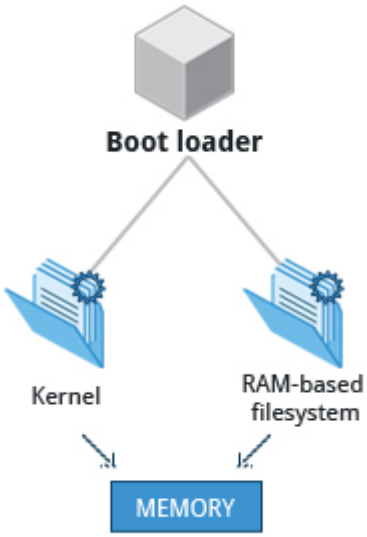
Como se podrá aprender más adelante, las terminales que ejecutan command shells pueden ser accesadas usando la tecla **ALT** + una tecla **function**. La mayoría de distribuciones inician seis terminales de texto y uno de gráficas usando las teclas **F1** ó **F2**. Dentro de un ambiente gráfico(graphical environment), cambiar a una consola de texto requiere de presionar las teclas CTRL-ALT + la tecla de función que se requiera (con F7 + ó F1 para entrar a la GUI / Grahical User Interface).



generalmente, el command shell predeterminado es **bash**, pero hay también algunos command shells avanzados disponibles. El shell imprime una petición de texto, indicando que está listo para aceptar comandos; luego que el usuario escribe el comando y presiona la tecla **enter**, el comando es ejecutado, y otra peticion es desplegada después de que el comando termina de ejecutarse.

El Kernel Linux

El cargador boot(Boot Loader), carga tanto el **kernel** como la archivo de sistema inicial RAM-base(initramfs) dentro de la memoria, de manera que pueda ser usada directamente por el kernel.

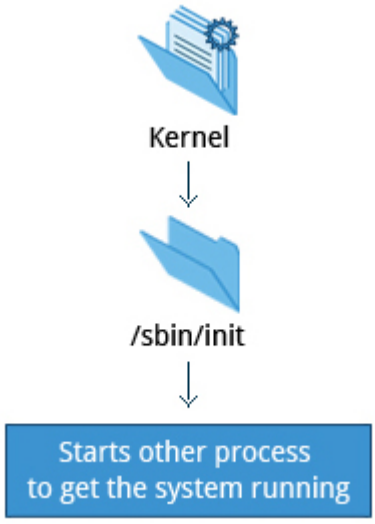


Cuando el kernel es cargado en la RAM,inmediatamente inicializa y configura la memoria del computador y configura todos los hardware o físicos instalados en el sistema. Esto incluye todos los procesadores, subsistemas I/O, dispositivos de almacenamiento,etc. El kernel también carga algunas aplicaciones de espacio de usuario necesarias.

- **/sbin/init and Services**

Una vez que el kernel ha configurado todo el hardware y montado el firoot filesystem, entonces kernel ejecuta `/sbin/init`. Esto,se convierte en el proceso inicial o principal, el cual inicai ortos procesos para mantener el sistema funcionando. La mayoría de otros procesos en el sistema registran su origen en **init**; a excepción de los llamados procesadores kernel. Estos son iniciados o ejecutados por kernel directamente, y su trabajo es manejar los detalles del sistema operativo interno.

Aparte de inciar el ssitema, **init** es responsable de mantener el sistema funcionadno y de apagarlo correctamente. Otra de sus responsabilidades es actuar cuando sea necesario como administrador de todos los procesos que no son kernel; los borra después de que terminen, y rreinicia los servicios de inicio de sesión de usuario cuando sea el usuario incie o cierre sesión, y hace lo mismo para otros servicios del sistema que corren en segundo plano(background).



Tradicionalmente, este proceso de inicio fue hecho usando procedimientos que remontan de regreso a los 1980 y el System V variety de UNIX. Este proceso serial hzio que el sistema pasra por una secuencia de **runlevels** que contenían colecciones de scripts que iniciaban y detenían servicios. Cada runlevel soportaba un modo diferente de correr el ssitema. Dentro de cada runlevel, servicios imndividuales podrían ser dispuestos a ejecutarse, o a ser detenidos(shut down) si estaban ejecutandose.

Sin embargo, todas las mayores distribuciones han dejado a un lado este método runlevel de iniciación de sistema, aunque usualmente dichas distribuciones simulan o emulan muchoas utilidades de System V por razones de compatibilidad. Más adelante se hablará de los nuevos métodos los cuales el **systemd** se ha vuelto el principal.

Startup Alternatives

Sysvinit Veía las cosas como un proceso serial, dividido entre una serie de estados secuenciales. Cada estado requería finalización antes de que el siguiente pudiera comenzar. Por lo tanto, el inicio no podría tomar fácilmente ventaha del ***procesamiento paralelo(parallel procesing)*** que podría haberse realizado en múltiples procesadores o núcleos.

Además, apagar e iniciar see veía como un evento relativamente poco usual. Esto ya no sigue siendo cierto, especialmente si hablamos de dispositivos móviles y sistemas Linux embebidos(embedded systems). Algunos métodos modernos, como el uso de **containers**, pueden requerir de casí instantáneos tiempos de inicio o booteo. Por lo tanto, los sistemas de ahora requieren métodos con capacidades más rápdias y mejoradas. Finalmente, los métodos antiguos requerían más bien instrucciones complejas de inicio(startups), las cuales eran difíciles de implementarse como universales a través de las versiones de distribución, versiones de kernelm arquitecturasa, y tipos de sistemas. Las dos principales alternativas desarrolladas fueron:

Upstart

- Desarrollada por Ubuntu e incluida en 2006.
- Adoptada en Fedora 9(en 2008), y en RHEL 6 y sus clones.

systemd

- Adoptado primeramente por Fedora en 2011.
- Adoptado por RHEL 7 y SUSE.
- Reemplazó Upstart en Ubuntu 16.04.

Linux command line

Comandos comunes

Aunque Linux Desktop usa GUI(Grafic User Interface) para las gráficas parecido a windows, la mayoría de veces las instalaciones usan un command line, sobre todo en servidores.

Un Shell provee al usuario de una interfaz sencilla para el Unix system. Sin embargo Bourne Shell es el que viene preinstalado.

pwd - muestra el directorio de la dirección en la que se encuentra el sistema.ejm:

```
[root@nombredeusuario Linux lite]# pwd
/home/linuxlite
[root@nombredeusuario Linux lite]#
```

cd cambio de directorio:

- *cd..* (con dos puntos) se usa para mover un directorio hacia la carpeta o directorio que le antecede.
- *cd* para ir directamente a la dirección main.
- *cd-* (con un guión) para mover el directorio a una dirección previa.

ls se usa para visualizar el contenido en un directorio:

- *ls -R* Enlistará todos los archivos en los subdirectorios también.
- *ls -a* muestra los archivos ocultos.
- *ls -al* muestra los archivos y los directrios con información detallada como permisos, tamaño, propietario, etc.

cat enlista el contenido de un archivo en la salida standar(standard output):

- *cat > nombrearchivo* crea un nuevo archivo.
- *cat nombrearchivo1 nombrearchivo2>nombrearchivo3* junta los dos primeros archivos descritos en un tercer nuevo archivo.

cp copia archivos.

mv mueve o renombra los archivos.

mkdir crea in nuevo directorio en el directorio en el que esté actualmente.

rm remueve archivos y directorios:

- ***rm -r*** remueve el directorio y los archivos dentro.