



# Grub a fondo

Federico Pacheco

Antes de comenzar con el tema que nos convoca, es necesario asegurarnos de comprender la verdadera necesidad de lo que a continuación analizaremos. Para comenzar, diremos que todo dispositivo electrónico pasa en algún momento por una etapa de encendido, lo cual resulta obvio hasta para quien no conoce sobre tecnología. Ya sea oprimiendo un botón, girando una perilla o accionando un switch, éste es el primer paso para obtener acceso a la funcionalidad del dispositivo.

linux@software.com.pl

**D**e esta manera, cuando más complejo sea el dispositivo, más procesos internos realizará a nivel electrónico antes de quedar completamente funcional. Por ejemplo, un teléfono celular, debe encenderse y esperarse a que tome señal, cargue los datos de la tarjeta interna, ajuste las preferencias del usuario, etc. En aparatos más complejos, como smartphones o handhelds, el proceso es similar, y en el caso de los ordenadores, tampoco es tan distinto, solo que se nota más la diferencia entre lo que ocurre con el software y lo que ocurre con el hardware. En líneas generales, este proceso de inicialización de cualquier aparato se denomina en tecnología: *bootstrapping*, o lo que como caso particular de una PC, nosotros denominamos *booting*.

Entrando entonces en el tema de los ordenadores, existirá un proceso llamado POST (*Power On Self Test* – Auto test de encendido) en el cual un pequeño programa llamado BIOS (*Basic Input Output System* – Sistema de entrada y salida básico) realiza diversas comprobaciones para verificar que el hardware esté en condiciones de arrancar. El código del BIOS permanece almacenado en un firmware, y una vez que arranca el equipo y verifica su funcionamiento, comienza

a ejecutarse su código desde la memoria (o a veces desde el mismo firmware) para luego cargar el primer sector del disco rígido (cilindro 0, cabeza 0, sector 1) y comprobar que su contenido sea válido, para pasarle el control al próximo



## Lo que deberías saber

- Uso del sistema operativo GNU/Linux o alguna de sus variantes
- Instalación de software en sistemas GNU/Linux



## En este artículo aprenderás

- Los conceptos relacionados con el arranque de un sistema GNU/Linux
- La necesidad y utilidad de un cargador de arranque
- El uso, instalación y configuración del cargador de arranque GRUB



programa que comenzará la carga del sistema operativo. Todas estas rutinas están escritas en lenguaje ensamblador, y sus funciones básicas continúan siendo las mismas que hace 30 años, lo cual puede constituir una limitación hoy en día.

## El cargador de arranque

Ahora que hemos definido algunos términos básicos, podemos pasar al próximo paso. Un cargador de arranque (en inglés *boot loader*) es un pequeño programa diseñado para preparar todo lo necesario para que un sistema operativo pueda funcionar. Por lo general cuentan con diversas etapas internas. El cargador de arranque le transferirá en su última etapa el control del equipo al sistema operativo así como el BIOS lo hace con éste. Finalmente, el sistema cargará su propio núcleo, los controladores, aplicaciones,

y demás, para luego quedar en funcionamiento normal, y considerar el proceso completo con el equipo en condiciones de responder a los requerimientos externos.

En principio un cargador puede hacer arrancar cualquier sistema operativo, ya que es independiente de éste. Podría sin embargo haber limitaciones en cuanto al medio de almacenamiento, por lo que debe tenerse en cuenta muchas veces.

Por supuesto que existen diversas alternativas, con distintas funciones para el gusto de los usuarios, pero en este caso nos dedicaremos al estudio de una en particular: GRUB. Pero antes de volcarnos a él, es bueno recordar que en un comienzo no todo era como hoy, y la hegemonía de los cargadores la ejercía el veterano LILO.

Sin entrar tanto en detalle, podemos decir que LILO (Linux LOader) es un gestor de arranque para Linux desarrollado inicialmente por Werner Almesberger, y actualmente a cargo de John Coffman. Funciona con diversos sistemas de archivos y puede arrancar desde el disco duro o flexible, permitiendo seleccionar entre 16 imágenes en el arranque. LILO puede

instalarse también en el Master Boot Record (MBR).

En las primeras distribuciones de Linux, era el gestor de facto. En la actualidad es realmente una segunda opción, en favor de GRUB. Algunas de sus limitaciones respecto al actual favorito son por ejemplo, que no tiene una línea de comandos interactiva, ni permite arrancar por red.

Otro gestor muy conocido en la actualidad es el llamado SYSLINUX, cuya información la podemos encontrar en el sitio <http://syslinux.zytor.com>.

## El MBR

El MBR (*Master Boot Record*) es el primer sector (es decir, el sector 0) de un dispositivo de almacenamiento de datos, como lo es un disco rígido. Puede utilizarse para contener al cargador de arranque o bien para almacenar directamente la tabla de particiones. En general, hablaremos del MBR como los primeros 512 bytes.

Antes de comenzar a jugar con un gestor de arranque, es una buena práctica recomendada hacer una copia de seguridad del MBR. En caso de que todo falle, podremos restaurarla

### Listado 1. Un ejemplo de configuración del archivo menu.lst

```
# Archivo de configuración de GRUB

timeout 10
default save
splashimage /dev/sdaX/ruta/
imagen.xpm.gz

title GNU/Linux Debian
root (hd0,0)
kernel /boot/vmlinuz-2.6.18 root=/dev/sda1
savedefault

title GNU/Linux Debian - Modo
Single
root (hd0,0)
kernel /vmlinuz-2.6.18 single
root=/dev/sda1 ro

title Gentoo GNU/Linux
root (hdd,0)
kernel /boot/grub/linux-1.6.22
root=/dev/sdb1
savedefault

title Windows XP
root (hd0,3)
makeactive
chainloader +1
savedefault

title Test de memoria
root (hd0,0)
kernel /boot/memtest86.bin
```

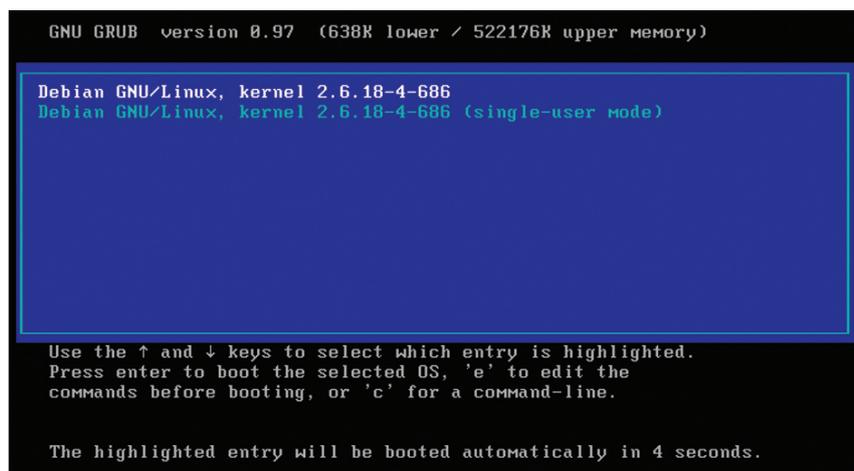


Figura 1. Menú estándar con las opciones de inicio de los sistemas

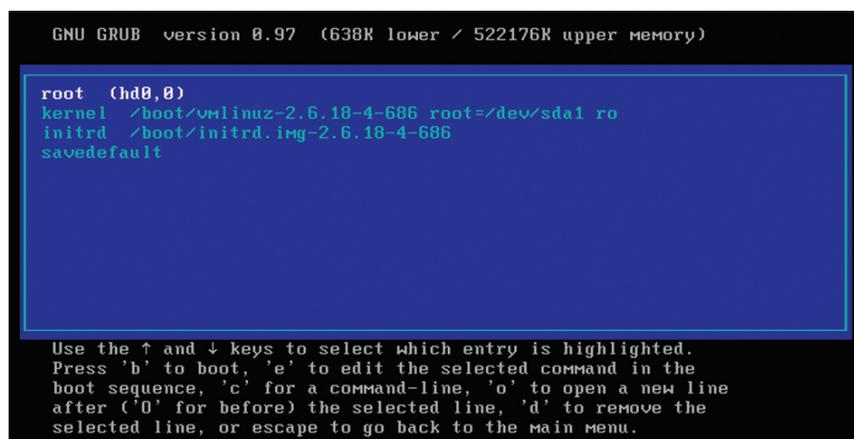


Figura 2. Modificación de la configuración en el propio momento de inicio



y hacer de cuenta que nada ha ocurrido. Para esto podemos utilizar el comando `dd` (*Dataset Definition*), desde una consola.

Realizar la copia de seguridad:

```
# dd if=/dev/dispositivo of=mbr.bkp
bs=512 count=1
```

Esto significa en español: copiar bit por bit los primeros 512 bytes del dispositivo ubicado en `/dev/dispositivo` (`hda`, `sda`, `sdb`, o cualquiera) a un archivo de nombre `mbr.bkp`. No debemos confundirnos y colocar por ejemplo, `/dev/hda1`, ya que de esa forma estaremos haciendo referencia a la primer partición, y no al disco completo, por lo cual pasaremos como parámetro su nombre solo `/dev/hda`.

Para restaurarlo, utilizamos el mismo comando con los parámetros `if` (input file) y `of` (output file) invertidos:

```
# dd if=mbr.bkp of=/dev/dispositivo
bs=512 count=1
```

Si quisiéramos borrarlo, podemos utilizar como parámetro `if` el dispositivo especial `/dev/zero`.

Para manipular el MBR y crear tablas, podemos utilizar el comando `fdisk` (o más amigablemente, el comando `cfdisk`).

## GRUB

Llegados al tema que nos congrega, comenzaremos por decir que GRUB significa *GRand Unified Bootloader*, que fue creado por Erich Stefan Boleyn bajo licencia GPL y que está alojado en <http://www.gnu.org/software/grub>. La versión que se utiliza comúnmente es la que pasara a llamarse *GRUB Legacy*, que por el momento continúa recibiendo correcciones de bugs, pero ya no se le agregan nuevas ca-

racterísticas. El foco de los desarrolladores fue puesto en GRUB 2, completamente reescrito, para poder hacerlo más robusto, potente, estable y portable. Es válido aclarar que no estamos hablando de *la segunda versión de GRUB*, sino de un nuevo proyecto, con licencia GNU, por supuesto. Algunas de sus ideas renovadas son por ejemplo, el soporte para plataformas que no sean x86, internacionalización y localización, uso de caracteres no ASCII, módulos dinámicos, gestión de memoria, un pequeño lenguaje de scripting, arquitectura orientada a objetos, y algunas cosas más. Muy interesante, sin duda.

Inicialmente GRUB 2 comenzó como proyecto PUPA, apoyado por la Information Technology Promotion Agency (IPA) de Japón. Luego, se integró al desarrollo de GRUB 2 en el año 2002, cuando la versión 0.9x de GRUB se renombró como *GRUB Legacy*. Al lanzarse, pasará a llamarse simplemente *GNU GRUB*. Al día de la fecha, aun no fue lanzado ni existe documentación al respecto. Como hemos visto anteriormente, el proceso de inicialización se subdivide en distintas partes, las que competen a GRUB se las llama Stages (Etapas). En general podemos distinguir entre las siguientes fases:

- El BIOS busca un dispositivo para iniciar y pasa el control al MBR.
- El MBR contiene la fase 1 (Stage 1) de GRUB. Como el MBR es pequeño la fase 1 sólo carga la siguiente fase del GRUB (ubicado físicamente en cualquier lugar del disco). La fase 1 puede cargar la fase 1.5 o la 2.
- En ocasiones, cuando la partición que contiene el núcleo está más allá del cilindro 1024 del disco o se está utilizando discos en modo LBA4, debe existir una etapa intermedia (Stage 1.5) que sirve de puente entre ambas y que depende del sistema de archivos. Esta etapa 1.5 es un pequeño

archivo de alrededor de 20 Kb. La fase 1.5 luego carga la fase 2.

- La fase 2 (Stage 2) de GRUB recibe el control y presenta el menú de inicio.
- GRUB carga el kernel seleccionado en memoria y le pasa el control del equipo.

Para realizar la pre-instalación en el sistema, podemos utilizar el sistema de paquetes de nuestra distribución preferida. En Debian y derivados está disponible en formato `.deb` accesible mediante la herramienta `apt-get`, en Red Hat y derivados lo encontramos en RPM, en Gentoo lo obtenemos mediante el sistema portage, cuya herramienta es `emerge`, y por si fuera poco, por supuesto que siempre es posible bajar el código fuente en `.tar.gz` y compilarlo manualmente.

Un tema que debe quedar claro desde el comienzo es la nomenclatura que utiliza GRUB para las unidades de disco. Esto es, el primer disco que reconoce el BIOS será (`hd0`), el segundo será (`hd1`), etc. sin hacer diferencia entre discos IDE, SATA o SCSI. En cuanto a las particiones, se numeran también desde el cero. Por ejemplo, (`hd0,0`) representa la primera partición del primer disco, y (`hd1,2`) representa la tercera partición del segundo disco. Las particiones extendidas comienzan a numerarse a partir de 4, tal como lo hace el propio Linux, independientemente de las particiones primarias existentes. Por ejemplo la primera partición extendida del primer disco sería (`hd0,4`). Los disquetes se denominan con el mismo criterio: (`fd0`), (`fd1`), etc.

Si bien es recomendable realizar la instalación y configuración de GRUB cuando se instala el sistema, dada la simplicidad que ofrecen los complejos instaladores actuales, también es posible realizarlo posteriormente una vez que tenemos el Linux en funcionamiento. Para casos como éste, podemos comenzar por ejecutar el script:

```
# grub-install /dev/dispositivo
```

Esta operación sobrescribe el MBR del disco seleccionado. Si tenemos el directorio `/boot` en una partición separada, podemos incluir el modificador: `--root-directory=/boot`.

## La consola de GRUB

La consola de GRUB es un elemento muy potente con el que contamos para configurar un arranque a medida de nuestras necesidades. Además, permite realizar distintas operaciones como las que veremos a continuación. Al igual que la mayoría de las consolas de GNU/Linux (`bash`, `tcsh`, `ksh`, etc.) admite el auto completa-

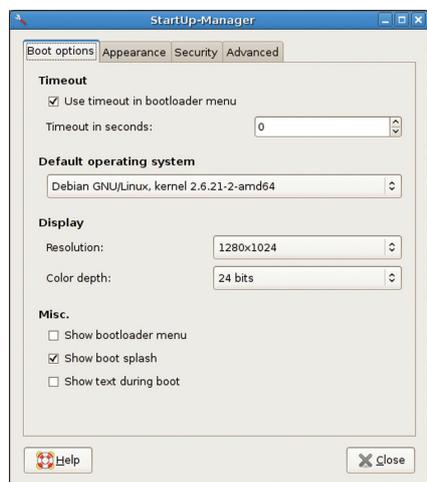


Figura 3. StartUp Manager

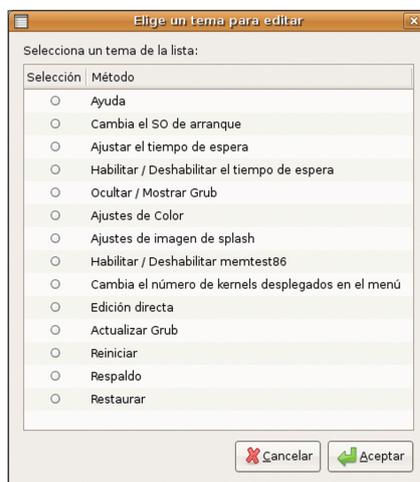


Figura 4. GrubED



do de comandos y sentencias mediante la tecla TAB. Sin duda una función casi indispensable para los que nos hemos acostumbrado a utilizarla en la consola del sistema.

La consola la obtenemos al ejecutar `grub` en la línea de comandos, y obtenemos el prompt `grub>` preparado para recibir sus propias órdenes. No olvidar que utiliza por

defecto distribución de teclado US. GRUB necesita que le indiquemos dos parámetros básicos para iniciar un kernel de Linux: la ubicación física del directorio raíz y la imagen



## Mensajes de error de GRUB

Entre las dos etapas principales del cargador (Stage 1 y Stage 2) hay mensajes de error típicos. Los errores en Stage 1 se limitan por lo general a mostrar un error y detener el sistema, en tanto que los de Stage 2 interrumpen la operación, muestran un mensaje y si es posible continúa con la operación. La siguiente lista está ordenada respecto al código de error.

- Errores de Stage 1
  - *Hard Disk Error*: Se intenta leer el disco pero no se consigue averiguar su geometría ni tamaño. También puede aparecer Floppy Error.
  - *Read Error*: Error de lectura al intentar leer stage2.
  - *Geom Error*: La ubicación en disco de stage2 no está en la parte del disco accesible por el BIOS. Esto ocurre al cambiar un disco, por ejemplo.
- Errores de Stage 2
  - *Filename must be either an absolute filename or blocklist*: Error de sintaxis en un archivo solicitado.
  - *Bad file or directory type*: Un archivo solicitado por grub no es regular, es un enlace simbólico o directorio por ejemplo.
  - *Bad or corrupt data while decompressing file*: El código de descompresión devuelve un error. Normalmente se trata de un archivo dañado.
  - *Bad or incompatible header in compressed file*: La cabecera de un archivo comprimido es incorrecta.
  - *Partition table invalid or corrupt*: La verificación del estado de la tabla de particiones falla.
  - *Mismatched or corrupt version of stage1/stage2*: Versiones incompatibles o dañadas de stage1 o stage2.
  - *Loading below 1MB is not supported*: La dirección más baja de un núcleo está por debajo de 1MB.
  - *Kernel must be loaded before booting*: Se quiso ejecutar algo sin haber indicado un núcleo.
  - *Unknown boot failure*: motivos desconocidos de error. ¡Muy grave!
  - *Unsupported Multiboot features requested*: El núcleo requiere una gestión especial que GRUB probablemente no tiene o no reconoce.
  - *Unrecognized device string*: No se cumplen las reglas de nombres de dispositivos.
  - *Invalid device requested*: El nombre del dispositivo es válido y tiene un error no contemplado por otro error.
  - *Invalid or unsupported executable format*: La imagen del núcleo no se reconoce como formatos admitidos (Linux, FreeBSD, etc.).
  - *Filesystem compatibility error, cannot read whole file*: Alguno de los sistema de fichero que tiene que leer GRUB tiene limitado el tamaño de lectura de ficheros.
  - *File not found*: No se puede encontrar un archivo solicitado pero la ruta de acceso es correcta.
  - *Inconsistent filesystem structure*: Error interno debido al estado de la estructura del sistema de archivos. Puede deberse a un sistema de archivos dañado.
  - *Cannot mount selected partition*: Una partición existe pero GRUB no la reconoce.
  - *Selected cylinder exceeds maximum supported by BIOS*: Una lectura intenta ir más allá del área direccionada por el BIOS.
  - *Linux kernel must be loaded before initrd*: Hemos colocado initrd antes que kernel.
  - *Multiboot kernel must be loaded before modules*: La orden de carga de módulos se utilizó antes que la de cargar un kernel.
  - *Selected disk does not exist*: Todo o parte un nombre de dispositivo no está presente o no se reconoce por el BIOS.
  - *No such partition*: No existe la partición especificada
  - *Error while parsing number*: Se espera un número y encuentran datos incorrectos.
  - *Attempt to access block outside partition*: Una dirección lineal de bloque está fuera de la partición. Normalmente se debe daños en el sistema de archivos.
  - *Disk read error*: Error de lectura en el disco.
  - *Too many symbolic links*: Demasiados enlaces simbólicos. El número máximo es 5. Es posible tener enlaces con bucle.
  - *Unrecognized command*: Comando no reconocido por GRUB. Puede tratarse de un error de tipeo.
  - *Selected item cannot fit into memory*: Se intenta cargar algo en memoria y no cabe.
  - *Disk write error*: Se intentaba escribir en disco y no se pudo. Sólo puede ocurrir durante la instalación o con la orden `makeactive`.
  - *Invalid argument*: El argumento para alguna orden es incorrecto.
  - *File is not sector aligned*: Se accede a una partición ReiserFS por lista de bloques.
  - *Must be authenticated*: Se trata de acceder a un arranque con "lock". Se debe introducir una contraseña válida.
  - *Serial device not configured*: Se intenta cambiar el terminal a uno serie pero no se ha activado ningún dispositivo serie.
  - *No spare sectors on the disk*: No queda espacio libre.



del kernel (y eventualmente una imagen para `initramfs` o `initrd`). Si estamos en la consola, indicamos esto línea por línea como en el siguiente ejemplo:

```
grub> root (hd0,0)
grub> kernel /vmlinuz root=/dev/sda1
grub> boot
```

El comando `root` hará que todo sea relativo a la ubicación de la raíz y el comando `kernel` permite indicar la imagen del kernel, el parámetro `root` dentro de esta línea nos permite indicar la ubicación de la imagen del mismo.

Adicionalmente, podemos indicar una imagen `initrd` mediante el comando `initrd`, que es un sistema de archivos temporal que utiliza el kernel en el arranque hasta que el sistema de archivos raíz pueda ser montado: `grub> initrd /initrd.img.`

Finalmente, `boot` inicia la carga. Aquí podemos utilizar la característica de auto completar de la consola, si es que por ejemplo, no tenemos el nombre completo de la imagen del kernel, o no recordamos la distribución de particiones del disco.

Una vez instalado y configurado, se creará un archivo llamado `menu.lst` (en algunas distribuciones llamado `grub.conf`) que contiene la información de la configuración previamente generada, la cual puede modificarse directamente editando el archivo, sin la necesidad de utilizar la consola.

Este archivo debe estar en la partición `/boot` que es la única inicialmente accesible por

`grub` y donde espera encontrarlo. Muchos de los comandos también podrán ser añadidos como etiquetas en este archivo de configuración.

También podemos hacer que todo el proceso se realice de manera manual indicando luego de `root (hd0,0)` el comando: `grub> setup (hd0)`

Si todo ha funcionado de manera correcta, al iniciar veremos el menú estándar con las opciones de inicio de los sistemas que hayamos agregado (Figura 1).

Aquí podemos modificar la configuración en el propio momento de inicio, accediendo a la edición de cada línea, cada una de las cuales nos llevará a todas las opciones de dicho sistema. Para esto utilizamos la tecla “e” (Figura 2).

Una vez ingresado, podemos editar la línea seleccionada (presionando “b”) pasar a la consola (presionando “c”) abrir una nueva línea (presionando “o” para una línea anterior y “O” para una posterior) borrar la línea seleccionada (presionando “d”) o iniciar el sistema con la configuración editada (presionando “b”).

### Opciones de configuración

Como hemos dicho, el archivo `menu.lst` admite ser editado directamente en texto plano. A continuación tenemos una serie de opciones típicas que pueden utilizarse.

No es necesario volver a ejecutar GRUB cada vez que realicemos una modificación de este archivo, como ocurre con LILO.

Para incluir un título para cada sistema que queramos iniciar: `title` seguido del nombre

que queremos que figure. Ejemplo: `title GNU/Linux Kernel 2.6.XY.`

Para indicarle el sistema que se cargará por defecto: `default X`. Donde `X` es un número, tomado desde 0 (primer sistema que se encuentra en el archivo).

Para agregar un tiempo de espera antes de la carga por defecto: `timeout X`. Donde `X` es un número en segundos.

Para ocultar el menú (por ejemplo si solamente tenemos un sistema): `hiddenmenu`

Para insertar una pausa: `pause`

Para cambiar los colores del menú de inicio, agregamos `color`, seguido por un color de entre los permitidos. Ejemplo: `color cyan/blue white/blue`

Para cargar un sistema Windows: agregar las líneas `rootnoverify` (arrancar la partición pero sin intentar montarla) `chainloader +1` (encadenar el cargador propio de otro sistema operativo) y `makeactive` (setea la partición como activa).

Para colocar una imagen de fondo, en formato `xpm`, y comprimida con `gzip` (recomendado): `splashimage /dev/sdaX/ruta/imagen.xpm.gz`

Para pasarle parámetros al kernel, contamos con algunas etiquetas, como en la siguiente línea: `kernel /boot/vmlinuz root=/dev/ada1 vga=ask hdc=ide-scsi`

Para modificar la asignación de particiones: comando `map`. Ejemplo para que el sistema vea el disco `Y` como si fuera el `X`: `map (hdX) (hdY)`.

Para establecer que la entrada por defecto sea la última seleccionada, agregamos la etiqueta `savedefault` entre las opciones del

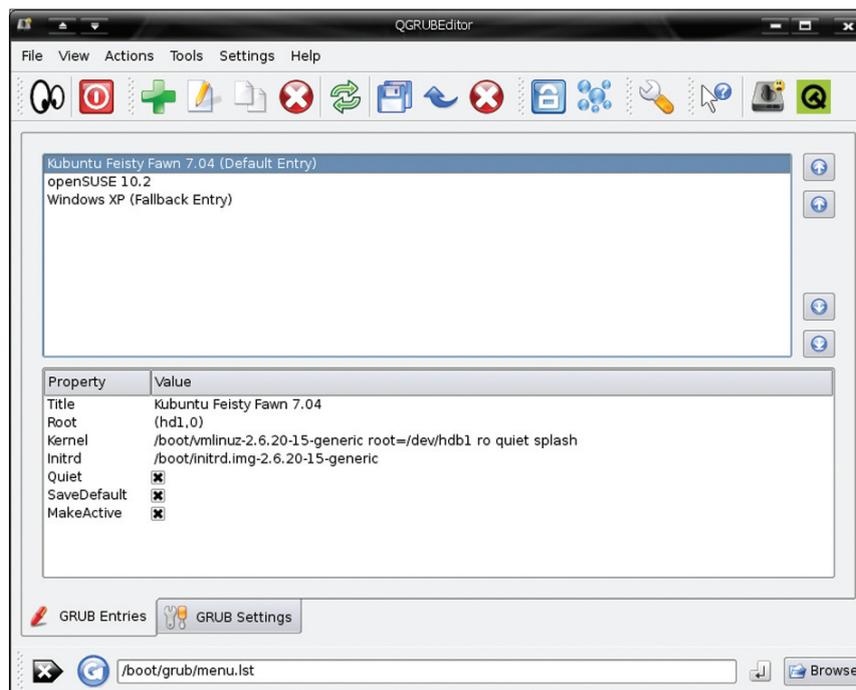


Figura 5. QGRUBEditor



### En la red

- Configuración de Grub - Una guía básica. Pedro Pablo Fábrega Martínez: <http://dns.bdat.net/documentos/grub/book1.html>
- Wikipedia: <http://es.wikipedia.org/wiki/Grub>, [http://en.wikipedia.org/wiki/GNU\\_GRUB](http://en.wikipedia.org/wiki/GNU_GRUB)
- GNU GRUB: <http://www.gnu.org/software/grub>
- GNU GRUB Manual: <http://www.gnu.org/software/grub/manual>
- IBM - Boot loader showdown: Getting to know LILO and GRUB: <http://www.ibm.com/developerworks/linux/library/l-bootload.html>
- PUPA Home: <http://www.nongnu.org/pupa/>

sistema deseado e indicamos: `default save`. Una buena idea es utilizar un gestor gráfico que permita manejar las opciones y generar el archivo `menu.lst`, como por ejemplo, el StartUp Manager (Figura 3), GrubED (Figura 4) o QGRUBEditor (Figura 5).

Por supuesto que existen muchas más opciones y comandos que podemos utilizar para nuestra configuración, pero solamente hemos mencionado aquellos de mayor utilidad general.

En función de todo esto y como ejemplo de utilización de muchas de las opciones anteriormente mencionadas, veremos a continuación un ejemplo de configuración del archivo `menu.lst` (Listado 1).

## Reinstalación de GRUB

Si hemos tenido problemas con el arranque, puede ser necesaria la reinstalación de GRUB. Esta situación puede darse por ejemplo, al cambiar las particiones, instalar otro disco o instalar otro sistema operativo.

Considerando la poca utilidad actual de los disquetes, no entraremos en detalles de la explicación de la creación de un disco de arranque, pero si fuera necesario, debemos formatear un disquete en algún formato de Linux (`ext2` por ejemplo) e indicarle la ruta de la disquetera en lugar del disco duro a `grub-install`. También es posible escribir los archivos de stage 1 y stage 2 directamente con el comando `dd o cat`, por ejemplo.

La primera medida realmente funcional de salvataje será entonces iniciar el sistema con un Live CD y reinstalar GRUB desde allí. Para esto deberemos arrancarlo, sin olvidar de montar la partición del sistema original, y luego conectar el directorio de dispositivos `/dev` del sistema del Live CD al de nuestro sistema, haciendo:

```
# mount --bind /dev /mnt
/nuestro_linux/dev
```



### Sobre el autor

Federico Pacheco ([federicop@elitech.com.ar](mailto:federicop@elitech.com.ar)) es especialista en seguridad informática y se desempeña como consultor en la empresa Elitech en Buenos Aires, Argentina. Es profesor de la misma especialidad, así como también de temas relacionados con los sistemas operativos GNU/Linux y colabora en diversos proyectos de investigación y difusión de tecnologías abiertas y el software libre.

A continuación cambiamos el origen del directorio raíz:

```
# chroot /mnt/nuestro_linux/
```

Para finalizar ejecutamos nuevamente el script `grub-install` indicando lógicamente la ubicación del disco, o realizamos la instalación manualmente desde la consola. También podemos utilizar el comando `find` para ubicar el stage 1.

Ejemplo en el MBR:

```
grub> find /boot/grub/stage1
grub> root (hd0,0)
grub> setup (hd0)
grub> quit
```

Otra opción es utilizar la utilidad *Super Grub Disk*, un restaurador del GRUB que se puede instalar en casi cualquier dispositivo externo (disquete, CD, DVD, USB) y es muy intuitivo y fácil de usar. Sólo esperamos no necesitarlo jamás.

## Un poco de seguridad

Si bien existen muchas medidas de seguridad que se pueden tomar para conseguir un arranque seguro del sistema, las mismas pueden aplicarse en distintos momentos, comenzando por el BIOS, que puede solicitar un password para permitir solamente el acceso a usuarios autorizados. La próxima medida de seguridad que podemos tomar, es limitar el acceso o las características de acceso, mediante el cargador de arranque. GRUB puede realizar esto, por supuesto, y cuenta con tres formas para diferentes necesidades. La primera característica de seguridad se refiere el hecho de evitar que pueda accederse a la consola integrada, ya que ésta proporciona acceso al sistema de archivos, por lo cual se puede acceder a información almacenada y esto podría ser muy peligroso. Esto se realiza mediante la siguiente sintaxis:

```
password [--md5] contraseña
[archivo_de_configuración]
```

Así evitamos el acceso, hasta tanto presionemos la tecla “P” e insertemos la contraseña correcta. Claro que nunca es una buena práctica almacenar contraseñas en texto plano, por lo tanto será mejor utilizar la opción `md5` para que admita como contraseña un hash MD5. Para obtener el hash de la contraseña podemos utilizar el comando `md5crypt` en una consola de `grub`:

```
grub> md5crypt
Password: *****
```

```
Encrypted: 1A/$%FJKLYUIY
I$8$/!&! (¿#=H71JZ&&
```

Claro que será necesario ahora escribir todo el hash MD5 en el lugar del password (mejor cortar y pegar):

```
grub> password --md5 1A/$%F
JKLYUIYI$8$/!&! (¿#=H71JZ&&
```

También podemos indicar un archivo de configuración como argumento de `password`, para que se cargue un archivo en particular al ingresar la contraseña válida:

```
grub> password contraseña
/boot/grub/menu-seguro.lst
```

La segunda opción de seguridad es el agregado de la etiqueta `lock`. Si tenemos varios sistemas operativos instalados y no queremos que todos puedan utilizarlos, agregaremos la etiqueta `lock` exactamente después de la etiqueta `title`. Ahora, al momento de intentar iniciar el sistema seleccionado, GRUB pedirá la contraseña global definida con `password`:

```
title Gentoo Linux
lock
root (hd0,0)
```

La tercera opción de seguridad se refiere al agregado de una contraseña específica para cada sistema operativo. Para esto agregamos la etiqueta `password` en la sección correspondiente. En este caso también se solicitará una contraseña para realizar el arranque, pero la misma se habrá agregado en texto claro en el mismo archivo de configuración, de la siguiente manera:

```
title Gentoo Linux
password mi-contraseña
root (hd0,0)
```

## Conclusiones

El mundo de los gestores de arranque está claramente dominado por GRUB, dada su potencia y flexibilidad. Mas allá de necesitarlo para inicializar diferentes sistemas, incluso con uno solo para arrancar, nos ofrece muchas alternativas que pueden sumar opciones variadas al proceso de inicio convencional. Si bien aun estamos a la espera del lanzamiento de la próxima versión, la existente está realmente muy madura, en lo que se refiere a software, y su amplia difusión hace que encontremos fuentes de información y resolución de problemas en muchos sitios, facilitando de esta manera el correcto uso y operación. 