

Derecho de Autor © Guillermo Antonio Abate

Permiso para copiar, distribuir y/o modificar este documento bajo los términos de la Licencia de Documentación Libre GNU, Versión 1.1 o cualquier otra versión posterior publicada por la Free Software Foundation; con las Secciones Invariantes siendo Mayix Stage4 Howto, Sin Texto de Cubierta Frontal así como para la Cubierta Posterior . Una copia de la licencia es incluida en la sección titulada "Licencia de Documentación Libre GNU".



Creacion de Live-cd Alpha-0.2

Aviso importante este documento esta en su fase de prueba y creación, se debe de tener en cuenta que muchos de los procedimientos aquí listados estan dirigidos a usuarios con experiencia en el sistema operativo linux!

Requerimientos:

Necesitamos tener por lo menos 4GB de espacio libre para construir nuestro sistema que se convertira despues en el livecd.

Preparacion:

Ahora de acuerdo a nuestro sabor y complacencia vamos a crear los directorios donde construiremos el sistema.

En mi caso he utilizado un disco duro que tenia por alli para estos fines el cual lo tengo montado en la / en la carpeta /Mayix alli he creado los directorios pero la verdad que lo que realmente es importante es el espacio puedes crear los directorios en tu /home o en /opt la verdad que no importa.

```
mkdir -p livecd/sources
```

en sources es donde vamos a descomprimir y construir.

Nos debemos hacer de un stage lo puedes hacer con el stage2 o stage3 alli depende de tu gusto lo que si es que debes tomar en cuenta que si el livecd es para que cualquiera lo use debes usar un stage generico de x86 o i686 para que sea compatible con cualquier arquitectura.

```
cd livecd/sources
tar -xvjpf stage3-i686-2004.0
mkdir newroot
```

Ahora es necesario que bajemos el ultimo portage disponible y que lo descomprimamos en livecd/sources/usr

```
cd usr
tar -xvjpf portage-xxxx.tar.bz2
```

Debemos tomar en cuenta que hay archivos que seran necesarios en *livecd/sources/etc* a la hora que hagamos el *chroot* como por ejemplo el *resolv.conf* para nuestra coneccion a internet.

Instalando el Sistema:

Debes tener en cuenta que para esta construccion debes saber instalar gentoo desde el stage que haz escogido, sino no sabes seria conveniente que antes de seguir le des una leida al handbook de gentoo el cual puedes encontrar en www.gentoo.org/doc/es/handbook/index.xml.

Ahora haremos algunos montages para poder seguir con la instalacion.

```
cd livecd/sources
mount -o bind /proc proc
mkdir usr/portage/distfiles
mount -o bind /usr/portage/distfiles usr/portage/distfiles/
```

Antes de proseguir con el *chroot* debemos verificar otros archivos importantes en *livecd/sources/etc* como lo es *make.conf* y verificar las USES oportunas para construir nuestro sistema como por ejemplo *USE="X gtk gtk2 gnome alsa -doc"* la arquitectura para la cual vamos a construir nuestro sistema, debes estar seguro agregar **livecd** tambien a USE.

```
cd livecd/sources
chroot . /bin/bash -login
env-update
source /etc/profile
```

Ahora podemos empezar a compilar tanto los archivos necesarios como los opcionales que deseamos en nuestro sistema.

Si has utilizado un stage2 deberias empezar con:

```
emerge system
```

luego deberias continuar con paquetes como *coldplug*, *vixie-cron*, *metalog*. Para despues agregar servicios a los niveles de arranque como *net.eth0*, *coldplug*, *hotplug*, *vixie-cron*, *metalog*, y algun otro que tu creas conveniente que se encuentre en el arranque.

Seria bien importante que tomaras en cuenta estos paquetes que seran de mucha ayuda cuando tu sistema arranque desde el cd

```
hwdata-knoppix
hwsetup
kudzu
livecd-tools
udev
```

Puedes compilar cualquier otra paqueteria que quieras incluir en el nuevo sistema.

Antes de compilar el kernel es importante que modifiquemos otro archivo del /etc como lo es el fstab

```
/dev/loop0      /          squashfs      ro,defaults   0 0
none           /proc      proc          defaults       0 0
none           /dev/shm   tmpfs         defaults       0 0
none           /dev/pts   devpts        defaults       0 0
```

Es el momento para que tambien modifiquemos otros archivos de /etc como (hostname, rc.conf, conf.d,....)

Compilar el *squashfs-tools* es realmente importante ya que ese sera el sistema de archivos de la imagen comprimida de nuestro sistema.

El siguiente paso es que compilemos el kernel con soporte para las diferentes arquitecturas debes tomar en cuenta que debe contar con los siguientes soportes.

1. **iso9660 cdrom filesystem**
2. **initrd support, a 8MB**
3. **loopback block device support**
4. **IDE/ATAPI cdrom device support**
5. **ext2 filesystem support (para ser usado en tu imagen de initrd)**
6. **tmpfs filesystem support**

Ahora vamos a compilar el kernel

```
emerge gentoo-dev-sources
cd /usr/src/linux
genkernel --makemenuconfig
make all && make modules_install && make install
```

Compilaremos el grub como gestor de booteo a grub en su version 0.95

```
cd /usr/portage/sys-boot/grub/
ACCEPT_KEYWORDS="~x86" emerge grub-0.95.xxxx.ebuild
```

Aqui esta el /boot/grub/grub.conf

```
default 0
timeout 5
splashimage=(cd)/boot/grub/splash.xpm.gz

title=Gentoo Linux
root (cd)
kernel (cd)/boot/kernel-xxxx video=vesafb:1024x768-32 root=/dev/ram0 rw init=/linuxrc cdroot
initrd (cd)/boot/initrd
```

Nota: Kernel-xxxx es la imagen del kernel que compilamos

Creacion de Imagen INITRD

El initrd es el archivo que usara mas a lo largo de el arranque del livecd, ahora lo haremos con un tamaño de 8 mb tu lo puedes hacer mas grande todo depende las necesidades que tengas pero recuerda que debes poner el mismo tamaño en la opcion del initrd en el kernel.

```
touch /boot/initrd
dd if=/dev/zero of=/boot/initrd bs=1024k count=8
losetup /dev/loop0 /boot/initrd
mke2fs /dev/loop0
mkdir /mnt/initrd
mount /dev/loop0 /mnt/initrd
```

Ahora vamos a crear los directorios y archivos necesarios

```
cd /mnt/initrd
mkdir etc dev lib bin proc new cdrom
touch linuxrc
chmod +x linuxrc
touch etc/mstab
touch etc/fstab
```

linuxrc es donde debemos tener los ejecutables que seran usados durante el proceso de arranque de la computadora desde el livecd

Necesitamos copiar los archivos necesarios dentro de bin y lib para bin copiaremos lo siguientes archivos

```
/bin/sh
/bin/cat
/bin/mount
/bin/mkdir
/bin/chroot
/bin/tar
/bin/ls
/sbin/pivot_root
```

Para lib necesitamos saber que librerias son necesarias para cada uno de los binarios que acabamos de copiar esto lo haremos con *ldd con cada uno de los archivos y copiar los requeridos en lib. Por ejemplo*

```
ldd /bin/mount
linux-gate.so.1 => (0xffffe000)
libc.so.6 => /lib/libc.so.6 (0x4002e000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
cp /lib/libc.so.6 /mnt/initrd/lib/
cp /lib/ld-linux.so.2 /mnt/initrd/lib/
```

Ahora necesitamos crear los dispositivos necesarios en el directorio de dev

```
mknod /mnt/initrd/dev/console c 5 1
mknod /mnt/initrd/dev/null c 1 3
mknod /mnt/initrd/dev/hda b 3 0
mknod /mnt/initrd/dev/hdb b 3 64
mknod /mnt/initrd/dev/hdc b 22 0
mknod /mnt/initrd/dev/hdd b 22 64
mknod /mnt/initrd/dev/tty c 4 0
mknod /mnt/initrd/dev/loop0 b 7 0
```

Finalmente necesitamos crear el script de linuxrc. El script debe hacer lo siguiente:

1. salva cualquier cosa que pasa por el kernel para pasar despues por el /sbin/init.
2. Localice el dispositivo del CDROM(buscaremos unicamente en hda,hdb,hdc y hdd los IDE mas comunes para los CDROMS)

3. Monte el cdrom en /cdrom
4. Monte la imagen del Sistema de Archivos Zisofs (La cual es toda nuestra instalacion de gentoo en un archivo) en /new. Esta sera el sistema de archivos / con propiedades de solo lectura.
5. Crear los puntos de montaje de escritura-lectura necesarios(etc, var, tmp, y root) como tmpfs y llenarlos
6. Finalmente el cambio de el sistema de archivos en una nueva raiz e iniciar el verdadero proceso del init.

```
#!/bin/sh
export PATH=/bin

# Adquiere kernel CMDLINE
mount -t proc none /proc
CMDLINE=`cat /proc/cmdline`
umount /proc

# Monta dispositivo de CD
CDROM=""
for x in hda hdb hdc hdd
do
mount -t iso9660 -r /dev/${x} /cdrom > /dev/null 2>&1
if [ "$?" = "0" ]
then
    CDROM="${x}"
    break
fi
done

# no encuentra el CD
if [ "${CDROM}" = "" ]
then
exec /bin/sh
exit
fi

# Monta root y crea los directorios de lectura-escritura
mount -t squashfs -o loop /cdrom/files/source.img /new > /dev/null 2>&1
mount -t tmpfs -o size=32m none /new/var > /dev/null 2>&1
mount -t tmpfs -o size=64m none /new/etc > /dev/null 2>&1
mount -t tmpfs -o size=10m none /new/tmp > /dev/null 2>&1
mount -t tmpfs -o size=10m none /new/root > /dev/null 2>&1
cd /new/var && tar xpf /cdrom/files/var.tar > /dev/null 2>&1
cd /new/etc && tar xpf /cdrom/files/etc.tar > /dev/null 2>&1
cd /new/root && tar xpf /cdrom/files/root.tar > /dev/null 2>&1

# Cambia root y empieza el init real
cd /new
pivot_root . newroot
exec chroot . /bin/sh <<- EOF >dev/console 2>&1
exec /sbin/init ${CMDLINE}
EOF
```

Nota: Yo he utilizado 4x32 tmpfs, esto es un monton de Memoria Ram, siquieres lo puedes reducir. Es que todos los equipos en que trabajo tienen un monton de Ram(servers) asi que no me preocupó.

Ya esta el Directorio de fuentes, pero antes de moverlo necesitamos haer algunas modificaciones. Primero en */var* estan directorios del portage “cache” y “db” que son muy grandes asi que los vamos a mover a */usr/lib/portage* para ahorrar espacio en */var*

```
cd /var
mv cache db /usr/lib/portage/
ln -s /usr/lib/portage/cache cache
ln -s /usr/lib/portage/db db
```

Hemos terminado con el directorio de sources debemos salir del chroot y desmontar /proc y /usr/portage/distfiles, para crear el livecd.

Creando el Livecd

Para crear el livecd debemos seguir los siguientes pasos:

1. Limpiar los directorios inecesarios como (/tmp /var/tmp).
2. Cree el directorio que utilizaremos para crear la imagen ISO (/mnt/mayix-iso).
3. Guardar los directorios de Lectura/Escritura (como /etc y /var) en tar en el Directorio.
4. Usar las herramientas squasfs, para convertir el el directorio de sources en una imagen squash.
5. Y finalmente crear el iso.

Para automatizar estos pasos crear un simple “build” script dentro del directorio “livecd”.

```
cd livecd
touch build
chmod +x build
```

Aqui esta el scrip:

```
#!/bin/bash
rm -rf target
mkdir target
cp -a source/boot target/
mkdir target/files
rm -rf source/var/tmp/*
rm -rf source/var/run/*
rm -rf source/var/lock/*
rm -rf source/tmp/*
rm -f source/etc/mtab
touch source/etc/mtab
cd source/etc/
tar cvpf ../target/files/etc.tar * .[:alnum:]]*
cd ../var/
tar cvpf ../target/files/var.tar * .[:alnum:]]*
cd ../root/
tar cvpf ../target/files/root.tar * .[:alnum:]]*
cd .././
mksquashfs source/ target/files/source.img
mkisofs -R -b boot/grub/stage2_eltorito -no-emul-boot -boot-load-size \
4 -boot-info-table -iso-level 4 -hide boot.catalog -o ~/livecd.iso target/
```

Siempre chequeen el tamaño de los archivos tar que serán los que se almacenen en el tmpfs.

Espero que les sirva este manual ;)

Guillermo Abate (Abasme)
guillermoabate@gmail.com
www.gentoo-la.org
www.mayix.net
www.la-granja.org