

Entornos chroot en Debian

Jesús M. González-Barahona.
Universidad Rey Juan Carlos, Móstoles, España
jgb@gsync.escet.urjc.es
jgb@debian.org

Septiembre de 2001

Versión 0.1, 2001.08.15, jgb

Versión 0.2, 2001.09.05, jgb

Resumen

Un entorno chroot sirve para trabajar como si tuviéramos un árbol de ficheros diferente. Esto puede ser muy útil para simular, por ejemplo, que tenemos instalado un cierto conjunto de paquetes, o incluso distribuciones enteras. Por ejemplo, en una máquina instalada con Debian stable podemos crear un directorio donde instalemos Debian unstable para probar algunos paquetes más actualizados. Una vez que tenemos montado el entorno chroot, podemos preparar de forma sencilla unos scripts para ejecutar aplicaciones que residen en él de forma casi transparente. Así podemos utilizar aplicaciones de una versión más nueva de Debian sin tener que reinstalar todo el sistema.

(Este artículo ha sido publicado por La Espiral y su versión mas reciente se puede encontrar en <http://www.laespiral.org>¹)

1. Introducción

Desde los primeros tiempos de Unix hay una llamada al kernel de nombre **chroot**. Su función es sencilla pero muy útil: cambiar la idea que tiene el proceso de cuál es el directorio raíz del sistema de ficheros. Si entiendes algo sobre la gestión de ficheros en Unix (y en Linux), te servirá saber que lo que se hace es cambiar el número de inodo raíz que se proporciona al proceso que se ha sometido a **chroot**. A partir de este momento, el proceso cree que está funcionando en

1

Copyright (c) 2001 Jesús M. González Barahona. Se otorga permiso para copiar, distribuir y/o modificar este documento según los términos de la Licencia GNU Para Documentación Libre (GNU Free Documentation License), versión 1.1 o cualquier versión posterior publicada por la Free Software Foundation. Esta licencia está disponible en <http://fsf.org/copyleft/fdl.html>.

un entorno donde todo lo que hay es el sistema de ficheros que cuelga del directorio que se ha elegido como raíz.

Por lo tanto, para que el proceso pueda funcionar una vez se ha hecho el **chroot**, es preciso que en su nuevo entorno de ficheros disponga de todo lo necesario para poder ejecutar su labor. Por ejemplo, si incluye bibliotecas dinámicas, habrán de estar disponibles en él con el nombre de fichero absoluto que el proceso espera. Si necesita ficheros de configuración, también habrán de estar disponibles. Y así para todos los ficheros que pueda necesitar.

De aquí en adelante, llamaremos "entorno chroot" al entorno en el que se desenvuelven los procesos después de ejecutar **chroot**, y "entorno base" al entorno que tenemos antes de ejecutarlo.

2. La orden chroot

El funcionamiento de la orden chroot es muy sencillo:

```
chroot <nombre-directorio> [<nombre-orden>]
```

<**nombre-directorio**> es el nombre del directorio que va a ser el raíz para el nuevo proceso lanzado. <**nombre-orden**> es un parámetro opcional, que indica el nombre del proceso que se va a lanzar en este entorno con el raíz cambiado. Si no se especifica, se lanza una shell en modo interactivo.

El resultado de esta orden es, por lo tanto, un nuevo proceso (el especificado, o un intérprete de órdenes) funcionando con el directorio raíz especificado. Así, si la orden fue:

```
# chroot /installed/testing ls /tmp
```

el resultado será un listado del directorio **/installed/testing/tmp** (suponiendo que en el directorio **/installed/testing** tenemos todo preparado para que **ls** pueda funcionar, incluyendo el propio **/bin/ls**

chroot sólo puede ejecutarse si se tienen permisos de administrador (root).

En Debian, **chroot** está en el paquete **shellutils**. Hay información sobre él en forma de página de manual (tanto para la llamada como para la orden), y algo más completa en página info (aunque en todos los casos no es muy extensa).

3. Preparación de un entorno chroot: instalación en partición separada

Una de las formas más simples de preparar un entorno chroot es usando una partición del disco duro donde se instalará todo lo necesario. Aunque conceptualmente simple, este método sólo es posible si dispones de esa partición, y si estás dispuesto a seguir en ella el proceso estándar de instalación. Por ello es normalmente poco utilizado. Pero se introduce primero porque es más sencillo de entender.

La idea es sencilla. Se hace una instalación estándar en una partición libre, y luego se usa esa partición completa como entorno chroot. Por ejemplo, partimos de un ordenador donde ya está instalado Debian GNU/Linux, y que tiene al menos una partición del disco duro libre. Se comienza por instalar la versión de Debian que queramos para lo que será el entorno chroot. Lo haremos como cualquier instalación normal, simplemente indicando la partición que nos interesa (que deberá estar libre) cuando nos pregunta la partición a usar para el directorio raíz (/). Se puede utilizar como swap la misma partición que ya se usa para el Debian con el que se trabaja normalmente, o no usar partición de swap (como luego usaremos la partición desde el chroot, no se usará el swap más que durante la instalación). Una vez que la instalación está lista, ya tenemos nuestro entorno chroot. Basta con reorganizar de la partición de trabajo normal, y (suponiendo que la partición que hemos instalado para el chroot es **/dev/hda7**) podemos escribir:

```
# mount /mnt /dev/hda7 /mnt
# chroot /mnt
```

Como puede verse, instalar el entorno de esta forma es fácil, pero tiene sus inconvenientes:

- Se instalarán cosas que normalmente no hacen falta en un entorno chroot (por ejemplo, el kernel).
- Es preciso usar una partición separada, lo que puede ser un problema si no se tiene una libre del tamaño suficiente.
- Hay que reorganizar la máquina para hacer la instalación del entorno, y mientras estamos en ello no se puede hacer otra cosa que instalar.
- Es complicado tener varios entornos chroot contruidos de esta forma.

En términos generales esta solución, aunque posible, en la mayoría de los casos será exagerada, una forma de matar moscas a cañonazos.

4. Preparación de un entorno chroot: uso de debootstrap

Lo ideal si se quiere trabajar en un entorno chroot es poder instalarlo en cualquier directorio, de forma cómoda, y sin abandonar el entorno de trabajo habitual, de forma que se puedan seguir haciendo otras tareas mientras se hace la instalación. Precisamente esto es lo que permite el paquete **debootstrap**.

La utilización de este paquete es simple. Una vez instalado, se ejecuta indicando los parámetros de la instalación del entorno chroot que se desea. Por ejemplo, para instalar lo básico de la distribución Debian sid (que al escribir este documento es la distribución inestable de Debian) en el directorio `/usr/local/chroot-envs/sid`:

```
# debootstrap sid /usr/local/chroot-envs/sid
```

Si se ejecuta esta línea, **debootstrap** se encargará de bajar los paquetes básicos de sid del lugar preconfigurado (<http://ftp.debian.org/debian>), de instalarlos y de configurarlos (haciendo las preguntas necesarias para ello, de forma similar a como se haría si estuviéramos instalando una máquina de forma normal. Cuando acabe, podemos usar el entorno recién instalado:

```
# chroot /usr/local/chroot-envs/sid
```

El intérprete de órdenes (shell) que obtenemos al ejecutar esta línea está corriendo en el entorno sid.

En la versión actual de **debootstrap** pueden instalarse entornos básicos de las distribuciones de Debian potato, sid o woody. Además puede incluirse como parámetro el espejo (mirror) que se desee. Por ejemplo:

```
# debootstrap woody /var/tmp/woody-chroot http://ftp.es.debian.org/debian
```

Naturalmente, con este método también puede instalarse Debian en una partición, con un resultado muy similar al obtenido usando el método descrito en el apartado anterior.

Nota: **debootstrap** no está disponible en Debian 2.2 (potato). Para usarlo en esta distribución será preciso instalarlo a partir del paquete fuente disponible como parte de Debian unstable, bien usando **apt-get source** o **dpkg-source**. Se espera que este paquete sea parte de Debian 3.0 (woody).

5. Cuando la instalación básica del entorno chroot ya está lista

Lo que se instala con cualquiera de los métodos descritos es un entorno básico, con pocos paquetes. Normalmente hará falta instalar más cosas. Para ello, basta con obtener una shell en el entorno chroot, y desde ella usar **dselect**, **apt-get** o la herramienta de instalación que se prefiera. Eso sí, hay que tener en cuenta que ciertos paquetes no se deben instalar, o no tiene sentido hacerlo. Por ejemplo, si se instala un nuevo lilo, puede haber problemas después de configurarlo. Algo similar ocurre si se instala un kernel.

Si cuando ejecutamos **chroot** obtenemos una shell en el entorno chroot, podemos trabajar en ella normalmente, lanzando aplicaciones, como lo haríamos en el entorno base. Sólo hay que tener en cuenta que desde esa shell estará disponible sólo lo que hayamos instalado en el entorno chroot. Cuando queramos dejar de trabajar en él, basta con salir de la shell (por ejemplo, usando la orden **exit**).

Naturalmente, el entorno chroot que hemos instalado es persistente. Esto es, todos los cambios que hagamos a sus ficheros permanecerán después de abandonar el entorno. Así, si instalamos un paquete o modificamos un fichero desde el entorno chroot, la próxima vez que entremos en él el fichero estará modificado y el paquete instalado. Lo mismo ocurre si entramos varias veces en el entorno chroot (por ejemplo, desde varias ventanas diferentes). Hay que tener en cuenta también que todos los ficheros del entorno chroot son accesibles normalmente desde fuera del entorno: basta con cambiarse al directorio adecuado.

6. Scripts para ejecución transparente de programas en un entorno chroot

Cuando se instalan aplicaciones en un entorno chroot puede ser conveniente disponer de scripts que permitan su ejecución desde el entorno base, por cualquier usuario y de forma lo más transparente posible. Eso permite, por ejemplo, que usuarios de un sistema estable puedan usar paquetes de la distribución inestable, sin que perciban ninguna diferencia con los paquetes del sistema estable.

Entre los scripts que pueden ser útiles en estas circunstancias, comenzaremos por uno que permita la ejecución en el entorno chroot de una aplicación cualquiera. Es el programa **exec_chrooted.sh** (suponemos que el directorio donde tenemos el entorno chroot es **/disks/hda7**):

```
#!/bin/bash

id=$SUDO_USER
dir=`pwd`
chroot_dir=/disks/hda7

/usr/sbin/chroot $chroot_dir /usr/local/bin/init-chroot.sh $id $dir $*
```

Este pequeño programa debe poder ser ejecutado por cualquier usuario, pero chroot sólo puede ser ejecutado por el usuario root. Por eso usamos los servicios de **sudo** (en el paquete del mismo nombre) al lanzarlo. Para poder hacerlo, una vez instalado **sudo** damos permisos en el fichero **/etc/sudoers**, añadiendo la siguiente línea:

```
ALL    ALL = NOPASSWD: /usr/local/bin/exec_chrooted.sh
```

De esta forma, al escribir la siguiente línea, sudo se encargará de ejecutar **exec_chrooted.sh** como root, pero sin pedir la contraseña correspondiente (ojo, como el script **exec_chrooted.sh** se va a ejecutar como root, es muy importante que no tenga ningún agujero de seguridad). La línea de ejecución es la siguiente (para ejecutar el programa **command** con las opciones **options_string** en el entorno chroot):

```
sudo /usr/local/bin/exec_chrooted.sh command options_string
```

Para que **exec_chrooted.sh** funcione, hay que tener instalado en el entorno chroot el programa **/usr/local/bin/init-chroot.sh**, que se encarga de hacer la inicialización necesaria para que los programas puedan funcionar de forma lo más transparente posible. Por ejemplo, en un hipotético caso, ese programa puede ser el siguiente:

```
# Get some variable from command line
id=$1
shift
```

```
dir=$1
shift
command=$1
shift
# Mount some directories
mount /disks/hda2
mount /home
mount /proc
# Chdir to simulate caller environment
cd $dir
# Run command
echo su $id -c "$command $*"
su $id -c "$command $*"
```

Es importante darse cuenta de que este programa ha de estar disponible *dentro* del entorno chroot. Por ejemplo, en nuestro caso, el directorio del sistema base donde estará es **/disks/hda7/usr/local/bin/init-chroot.sh**

Un a vez tenemos **exec_chrooted.sh** funcionando, podemos preparar un script muy sencillo para ejecutar cualquier programa dentro del entorno chroot:

```
sudo /usr/local/bin/exec_chrooted.sh $0 $*
```

Podemos llamar **exec_chrooted_sudo.sh** a este programa, y colocarlo, por ejemplo, en **/usr/local/bin**. Cuando queramos ejecutar un nuevo programa dentro del entorno chroot, bastará con enlazar **exec_chrooted_sudo.sh** con el nombre de ese programa. Por ejemplo, para poder ejecutar **xine** (instalado en el entorno chroot) desde el entorno base, podemos hacer lo siguiente:

```
ln -s /usr/local/bin/exec_chrooted_sudo.sh /usr/local/bin/xine
```

Si el script **init-chroot.sh** prepara bien las cosas, cualquier usuario podrá ejecutar ahora **xine** sin notar que está instalado dentro de un entorno chroot.

También puede ser útil un programa que nos de una shell dentro del entorno chroot, basado en **exec_chrooted.sh** (llamémosle **exec_chrooted_bash.sh**). Es muy fácil:

```
sudo /usr/local/bin/exec_chrooted.sh bash $*
```

7. Algunos consejos y trucos

- Directorio /proc

Es muy interesante tener el directorio **/proc** montado también dentro del entorno chroot. Para ello puede usarse simplemente la orden **mount** desde una shell que ejecute en el entorno chroot:

```
# mount /proc
```

- Directorio /tmp

También es conveniente que el directorio **/tmp** del entorno chroot y el del sistema que estamos usando como base sean el mismo. Esto permite que desde el entorno chroot se vean los ficheros que se crean en el **/tmp** del entorno base. Entre ellos, por ejemplo, **/tmp/.X11-unix** (si se ha lanzado XWindow en el entorno base). Lo más cómodo para esto es tener **/tmp** como una partición independiente (lo que proporciona también otras ventajas), y simplemente montarla en el entorno chroot. Si esto no es posible, habrá que recurrir, por ejemplo, a esquemas más elaborados basados en enlaces simbólicos.

- Directorio /home

Permite tener acceso a los ficheros de los usuarios. En particular, permite que cuando un programa se ejecuta en el entorno chroot pueda acceder a ficheros que el usuario tiene en su directorio hogar, ayudando mucho a la transparencia de ejecución de ese programa. También permite que el proceso que ejecuta en el entorno chroot tenga acceso a ficheros de configuración de usuario que puede necesitar, o intercambiar información con otros programas que corren en el entorno base (es el caso, por ejemplo, del fichero **.Xauthority**, que permitirá a un proceso que use el sistema XWindow acceder a información de autenticación que le servirá para mostrar su salida en la pantalla gestionada por el servidor XWindow (que normalmente ejecutará en el sistema base).

Pero hay que tener cuidado cuando se hacen las cosas de esta manera. El formato de los ficheros de configuración puede cambiar entre versiones, y eso puede hacer que una aplicación que use ficheros de configuración de otra versión no funcione, o incluso los corrompa. Esto puede ser especialmente peligroso si tenemos la misma aplicación en dos versiones diferentes en el entorno base y en el entorno chroot.

- Desmontajes

Si se hacen montajes dentro del entorno chroot es conveniente recordar que hay que desmontar las particiones reales antes de apagar la máquina. Si se sigue el procedimiento de apagado normal en el entorno base si haber hecho estos desmontajes, la cuenta de montajes de las particiones reales montadas en el entorno chroot quedará mal. Por ejemplo, si estamos montado la partición **/tmp** desde el entorno base y desde el entorno chroot, la cuenta de montajes de esa partición estará a 2. La secuencia normal de apagado la desmontará sólo del sistema de ficheros base, con lo que la cuenta quedará a 1. Al volver a arrancar la máquina, **fsck** detectará que la cuenta de montajes es 1, y forzará una comprobación de la partición pensando que no se desmontó correctamente antes de apagar (con el consiguiente retraso en el arranque).

Para evitar este problema, basta con desmontar las particiones reales montadas en el entorno chroot. Puede usarse simplemente el comando **umount** desde el entorno chroot. Como tratar

de desmontar una partición que no está montada no causa problemas, puede ser buena idea tener un script que desmonte todas las particiones reales que montemos normalmente en el entorno chroot. Cuando usemos el entorno chroot, habrá que recordar ejecutar ese script la última vez que lo vayamos a abandonar, antes de apagar la máquina.

- Ojo con los puertos

No debe perderse de vista el hecho de que **chroot** sólo afecta a los ficheros sobre los que tiene visibilidad el proceso. Otros elementos del sistema operativo no se ven afectados. En particular, los puertos de comunicaciones son los mismos en el entorno base y en el entorno chroot. Así, si en el entorno base tenemos un servidor de web trabajando en el puerto 80, y tratamos de lanzar otro en el entorno chroot, tendremos un error de "puerto ya en uso".

- Ficheros de configuración del sistema

Puede ser útil preparar algunos ficheros de configuración del sistema en el entorno chroot. Aunque no vayamos a usar ese directorio como entorno base (y por lo tanto esos ficheros no afectarán al funcionamiento general de la máquina), pueden simplificar mucho nuestra vida dentro del entorno chroot. Algunos de estos ficheros son los siguientes:

- /etc/fstab

Para poder hacer los montajes de forma más sencilla

- /etc/passwd (y compañía)

Si se crean usuarios dentro del entorno chroot, quedarán reflejados en la familiar de ficheros **/etc/passwd**. Si tengo usuarios creados dentro del entorno, podré usarlos haciendo **su** (ver fichero **init-chroot.sh**). Si además los identificadores de usuario son los mismos que en el entorno base, se podrán usar los ficheros como se usarían en ese entorno (lo que simplifica mucho los scripts para la ejecución transparente de aplicaciones instaladas dentro del entorno chroot).

- /etc/resolv.conf

Permite a las aplicaciones dentro del entorno chroot encontrar la lista de servidores de DNS que tienen que usar para resolver nombres en Internet. Si apt usa almacenes remotos accesibles vía Internet hará falta tener bien configurado este fichero antes de usar **apt-get**.

- Seguridad

Cuidado con la seguridad. Todo lo que instalemos en el entorno chroot estará accesible para el resto del sistema, como ya se ha dicho. Eso quiere decir que si en el entorno chroot tenemos, por ejemplo, programas "suid root" con errores que afectan a la seguridad, alguien podría usarlos desde el entorno base para romper la seguridad de todo el sistema (no sólo del entorno chroot).

8. Bibliografía

1. *chroot - change root directory* Linux Programmer's Manual **Section 2 (man)**
2. *chroot - run command or interactive shell with special root directory* GNU sh-utils documentation **Section 8 (man)**