DebootstrapChroot

https://help.ubuntu.com/community/DebootstrapChroot

DebootstrapChroot

This article shows you how to use debootstrap to build a *chroot environment* that you can use for various needs, from trying out the latest (or even oldest) Ubuntu releases, or even working with Debian releases, to utilizing the chroot as a package building environment. For an alternative approach that uses higher level tools (e.g. `sbuild`), see this page on setting up sbuild environments.

You can work anywhere - this Howto will assume you're using `/var/chroot` and gives a target system of HardyHeron. For other target versions replace *hardy* below with the first part of the release code name, *dapper* for DapperDrake or *lucid* for LucidLynx for example.

Getting and installing debootstrap

For the least pain and gnashing of teeth, please get the Ubuntu binary packages *manually* by downloading from the following links with the 'wget' command which is demonstrated below:

- http://archive.ubuntu.com/ubuntu/pool/main/d/debootstrap/debootstrap_1.0.7~dapper1_all.deb - If you want a *dapper* chroot
- http://archive.ubuntu.com/ubuntu/pool/main/d/debootstrap/debootstrap_1.0.7~edgy1_all.deb - If you want a *edgy* chroot
- http://archive.ubuntu.com/ubuntu/pool/main/d/debootstrap/debootstrap_1.0.7~feisty1_all.deb - If you want a *feisty* chroot
- http://archive.ubuntu.com/ubuntu/pool/main/d/debootstrap/debootstrap_1.0.7~gutsy1_all.deb - If you want a *gutsy* chroot
- http://archive.ubuntu.com/ubuntu/pool/main/d/debootstrap/debootstrap_1.0.9~hardy1_all.deb - If you want a *hardy* chroot (if that's not available go to http://archive.ubuntu.com/ubuntu/pool/main/d/debootstrap/ and find the newest one)

Example: Terminal session `wget`-ing and installing the latest hardy debootstrap:

```
wget http://archive.ubuntu.com/ubuntu/pool/main/d/debootstrap/debootstrap_1.0.9~hardy1_all.deb
sudo dpkg --install debootstrap_1.0.9~hardy1_all.deb
```

Note that if you are going to develop the *next* in-development version on the current stable release (e.g. you want a Quantal chroot on your Precise system), you will need to install `debootstrap` from backports. Once you've enabled backports (e.g. `precise-backports`), you'll need to explicitly install the new `debootstrap` in Ubuntu versions on or after 11.04. E.g.

```
apt-get update
apt-get install -t precise-backports debootstrap
```

Now you can easily build Quantal chroots.

Installing and configuring schroot

schroot is a convenient means of managing chroot environments; with this tool you can have both Dapper, Feisty and even Debian Sid chroots in your Ubuntu install, and using a chroot environment is a simple as `schroot -c dapper -d` 😉

To get `schroot` working in your system, just do the following in a Terminal:

```
sudo apt-get install schroot
sudo mkdir /var/chroot # Remember, we assume our chroot is here
sudo editor /etc/schroot/schroot.conf
```

Then append this line in `/etc/schroot/schroot.conf`:

```
[hardy]
description=Ubuntu hardy
location=/var/chroot/hardy
priority=3
users=doko
groups=sbuild
root-groups=root
```

Additional steps for 64-bit systems

If you are running a 64bit kernel and install a 32bit chroot (architectures i386, lpia on amd64, sparc, powerpc), add the line:

```
personality=linux32
```

and install the `linux32` package. This avoids prefixing each schroot command with the `linux32` command.

```
aliases=dokochroot,default
```

default is very useful as are aliases.

Setting up your chroot with debootstrap

If you want a 32-bit chroot on amd64 add `--arch i386` to this command line. If you use the chroot to build packages add `--variant=buildd` . Change *hardy* to according to your needs to *dapper*, *edgy*, *feisty* or leave as is for *hardy* chroot.

To actually install the base chroot, open a Terminal and do:

```
 sudo debootstrap --variant=buildd --arch i386 hardy /var/chroot/hardy http://archive.ubuntu.com/ubuntu/
```

`debootstrap` will then build a HardyHeron chroot in `/var/chroot/`, getting the base packages in `http://archive.ubuntu.com/ubuntu/`, and, depending on the given additional options (in square brackets,) `debootstrap` will build a chroot for the given architecture and variant.

If `debootstrap` finishes successfully, you'll be left with a base chroot in `/var/chroot`, which is not suitable for nearly anything. To actually get our chroot to work and be able to, say, grab packages from the network, do the following right after `debootstrap`:

```
 sudo cp /etc/resolv.conf /var/chroot/hardy/etc/resolv.conf
 sudo cp /etc/apt/sources.list /var/chroot/hardy/etc/apt/
 sudo editor /var/chroot/hardy/etc/apt/sources.list
```

If your current distribution is different than your target distribution (i.e. you use Hardy and want a Gutsy chroot), change all of the occurrences of Hardy/Gutsy/Feisty/Edgy etc. to your target distribution.

```
 sudo chroot /var/chroot/hardy
 apt-get update
 apt-get --no-install-recommends install wget debconf devscripts gnupg nano  #For package-building
 apt-get update  #clean the gpg error message
 apt-get install locales dialog  #If you don't talk en_US
 locale-gen en_GB.UTF-8  # or your preferred locale
 tzselect; TZ='Continent/Country'; export TZ  #Configure and use our local time instead of UTC; save in .profile
 exit
```

If you dont want the locale warnings in your chroot, add this to your ~/.bashrc file.

```
export LANG=C
```

You can stop here if you want a simple chroot that you use as root (`sudo chroot /var/chroot`). If you want to use your chroot as another user and have access to your normal /home and other directories inside the chroot, continue.

Note for Debian chr oot on Ubun tu

If you want to build a Debian chroot on an Ubuntu system you need to point it at a Debian archive:

```
 sudo debootstrap --arch i386 sid sid/ http://ftp.uk.debian.org/debian/
```

Getting stuff(X/ssh-agent/ect,dbus,mounting removables,modprobe,err stuff) working automagically

Append these lines to `/etc/fstab`:

```
/proc /var/chroot/hardy/proc none rbind 0 0 # Can just be mounted, comments?
/dev /var/chroot/hardy/dev none rbind 0 0 # Good thing to do, but not secure.
/sys /var/chroot/hardy/sys none rbind 0 0 # Same as proc?
/tmp /var/chroot/hardy/tmp none rbind 0 0 # This opens a lot of doors, namly X sockets are here... DRI should work assuming bit
/home /var/chroot/hardy/home none rbind 0 0 # This is optional.  As are the others, but this is more so.
/media /var/chroot/hardy/media none rbind 0 0 # Your USB stick.
/lib/modules /var/chroot/hardy/lib/modules none rbind 0 0 # You may need to load modules??  Think binfmt_misc.
/var/run/dbus/ /var/chroot/hardy/var/run/dbus/ none rbind 0 0 # Gnome likes this.
# Others??  /etc?
```

Note: fstype is **none** options are **rbind**.

Loading cron/apache/daemons

Add this(or something like it) to `/etc/rc.local` or your startup wherever you like:

```
schroot --all -- su -c /etc/init.d/rc\ 2 -
```

Setting up a dchroot (non-root) environment

`dchroot` makes it possible to use your newly-built chroot even as a non-root user. Hence, you can configure your chroot environment in such a way that you can even use your existing `/home` as the chroot's `/home`, thereby saving you some expensive moving in between homes, as well as making package building/testing a *LOT* more convenient.

To do this, first fix the user and root password:

```
 sudo cp /etc/passwd /var/chroot/hardy/etc/
 sudo sed 's/\([^:]*\):[^:]*:/\1:*:/' /etc/shadow | sudo tee /var/chroot/hardy/etc/shadow
 sudo cp /etc/group /var/chroot/hardy/etc/
 sudo cp /etc/hosts /var/chroot/hardy/etc/ # avoid sudo warnings when it tries to resolve the chroot's hostname
```

For a debian chroot, I also had to do:

```
sudo sed 's/\([^:]*\):[^:]*:/\1:*:/' /etc/gshadow | sudo tee /var/chroot/hardy/etc/gshadow
```

Then enable sudo and setup your passwords for root and the first sudo user in the `admin` group:

```
 sudo cp /etc/sudoers /var/chroot/hardy/etc/
 sudo chroot /var/chroot/hardy/
 dpkg-reconfigure passwd
 passwd <username of your first ubuntu user in the admin group>
```

Next, install the sudo package to be able to use it being in chroot:

```
 apt-get install sudo
 exit
```

Finish things up:

```
 sudo editor /etc/fstab
```

This is like the previous instructions, but different. Add these lines: (/media/cdrom is optional, of course, and you might have to create the dir in the chroot)

```
 /home           /var/chroot/hardy/home        none    bind            0       0
 /tmp            /var/chroot/hardy/tmp         none    bind            0       0
 /media/cdrom    /var/chroot/hardy/media/cdrom none    bind            0       0
 /dev            /var/chroot/hardy/dev         none    bind            0       0
 proc-chroot     /var/chroot/hardy/proc        proc    defaults        0       0
 devpts-chroot   /var/chroot/hardy/dev/pts     devpts  defaults        0       0
```

and delete these lines from before:

```
 /proc /var/chroot/hardy/proc none rbind 0 0 # Can just be mounted, comments?
 /dev /var/chroot/hardy/dev none rbind 0 0 # Good thing to do, but not secure.
 /sys /var/chroot/hardy/sys none rbind 0 0 # Same as proc?
 /tmp /var/chroot/hardy/tmp none rbind 0 0 # This opens a lot of doors, namly X sockets are here... DRI should work assuming bi
 /home /var/chroot/hardy/home none rbind 0 0 # This is optional.  As are the others, but this is more so.
 /media /var/chroot/hardy/media none rbind 0 0 # Your USB stick.
 /lib/modules /var/chroot/hardy/lib/modules none rbind 0 0 # You may need to load modules??  Think binfmt_misc.
 /var/run/dbus/ /var/chroot/hardy/var/run/dbus/ none rbind 0 0 # Gnome likes this.
```

Mount them:

```
sudo mount -a
```

The default bash path includes chroot information. To make this visible:

```
sudo chroot /var/chroot/hardy/
echo mychroot > etc/debian_chroot
exit
```

Set the chroot you just created in the dchroot.conf file

```
sudo editor /etc/dchroot.conf
```

Add the following to this file (if this is your first "dchroot" it will be a new, empty file; if there is more than one, the first item listed will be the default):

```
mychroot /var/chroot/hardy/
```

Now when you want to use your chroot (you may omit the `-c mychroot` if there's only one, or you just want the first one in the file). The `-d` parameter means that your environment will be preserved, this is generally useful if you want chrooted applications to seamlessly use your X server, your session manager, etc.

```
 dchroot -c mychroot -d
```

Tada! Now you can switch to and from your main `/` and `/var/chroot/`, without even becoming root!

Shortcuts / Usage

you can type dchroot -d "command" and it executes that command in the chroot.

I have this script do_chroot in /usr/local/bin:

```
/usr/bin/dchroot -d "`echo $0 | sed 's|^.*/||'` $*"
```

I had trouble with quoting in the above script. This one works better for me. ~JPKotta

```
args=""
for i in "$@" ; do
    args="$args '$i'"
    #echo $args
done

/usr/bin/dchroot -d -- "$0" $args
```

Then I create a symbolic link from that to the command I want to execute in the chroot, e.g.:

```
ln -s /usr/local/bin/do_chroot /usr/local/bin/firefox
```

which will execute firefox in the chroot environment when I launch it in my normal 64 bit environment. To launch my amd64 firefox I can type /usr/bin/firefox.

Instead if you want you can just create a script for launching the 32bit firefox e.g.:

```
dchroot -d "firefox"
```

put it in /usr/local/bin and add it to the gnome menu.

If you're going to start a program that only works in 32bit, first type dchroot -d and you'll be in the 32 bit environment.

Notes

Some missing points are covered on this external article: http://ornellas.apanela.com/dokuwiki/pub:multiarch.

From unknown Sun Apr 17 05:43:14 +0100 2005 From: Date: Sun, 17 Apr 2005 05:43:14 +0100 Subject: Using symlinks for passwd, groups, shadow, etc..? Message-ID: <20050417054314+0100@https://www.ubuntulinux.org>

Wouldn't it be possible to use symlinks for the files that get copied into the chroot? Like /etc/hosts? Would it work with /etc/passwd and the like?

```
  Re: You can link into, but not outof a chroot.
mv /etc/hosts /chroot/etc/hosts
ln -s ../chroot/etc/hosts /etc
... Using hardlinks is better.
```

From MichaelShigorin Sun Apr 17 13:42:38 +0100 2005 From: Michael Shigorin Date: Sun, 17 Apr 2005 13:42:38 +0100 Subject: nope Message-ID: <20050417134238+0100@https://www.ubuntulinux.org>

...but you can mount --bind them one be one.

From goofrider Thu May 12 19:26:45 +0100 2005 From: goofrider Date: Thu, 12 May 2005 19:26:45 +0100 Subject: chroot and symlinks Message-ID: <20050512192645+0100@https://www.ubuntulinux.org>

You can't symlinks from inside the chroot to somewhere outside of it, because once you chroot into it, the new chroot will becomes `/`, and all symlinks will be resolved relative to this new `/`. Use `mount --bind` instead (though hard links should work too). --GoofRider 2005-05-12

From Sam Fri May 13 09:22:44 +0100 2005 From: Sam Date: Fri, 13 May 2005 09:22:44 +0100 Subject: mount -a Message-ID: <20050513092244+0100@www.ubuntulinux.org>

You can use $ sudo mount -a for mounting all the entries in fstab instead of mounting them one by one.

From LukaszStelmach Sun May 15 00:06:59 +0100 2005 From: Lukasz Stelmach Date: Sun, 15 May 2005 00:06:59 +0100 Subject: Using symlinks Message-ID: <20050515000659+0100@www.ubuntulinux.org>

You can make hardlink to files (but only when your chroot dir is on te same partition):

ln /etc/passwd /var/chroot/etc/

From: Elmo, 21.12.05 Does anyone know howto enable DRI from inside a 32bit chroot, 'cause if I mount --bind /dev/dri chroot/dev/dri I get the following error: "DDX driver parameter mismatch: got 848 bytes, but expected 840 bytes. libGL error: InitDriver failed" (glxinfo) I'd really like to get doom3 working on my amd64 install.

26.12.05, Elmo: I know, it should work natively, but I have problems with other games aswell, so getting dri working from a chroot would be great=)

26.12.05, Elmo: At debian-amd64 list(http://lists.debian.org/debian-amd64/2005/02/msg00807.html), around February 05, is said that it's not possible at the moment. Got to find another way around my problem, will propably post to ubuntu forums.

10.06.06 Just a note from a person who ruined his system: After all this is done do not go and delete things from /var/chroot willy-nilly as it will delete the files from the linked directory as well. I found this out only after my entire /home directory was wiped out when I tried to free up some disk space by deleting the files from the chroot directory. Thanks to my foolishness I emptied root's trash before I realized what I'd done. It's been a while since my last backup so I lost everything from Documents, etc for the last year or so.

From: Murray Cumming 06.10.05: I had to do "apt-get install language-pack-en" to avoid the "Locale not supported by C library." warnings. Even "sudo dpkg-reconfigure locales" gave a "perl: warning: Setting locale failed." error until I did this. And that was even after I did a whole "sudo apt-get ubuntu-desktop" in the chroot.

Almost all the schroot config is unhelpful and irrelevant - Adding three lines to schroot.conf completely removes the need to copy anything from/to /etc:

```
run-setup-scripts=true
run-exec-scripts=true
type=directory
```

these will cause schroot itself to copy the latest versions of the required files every time, and do all required mounting to get /proc and /home working. Removes a LOT of effort and worry. And removes the risk of deleting your own home area due to stray bind mounts. -- directhex, 2007-09-21

```
 Re: This is the best method.
I see there are a few things missing from these scripts, the rbind(bind) stuff, ect.  We should identify what is missing and tr
```

The dchroot stuff here is practically obsolete. I found that it is completely possible to create a working schroot environment that does not make an individual root. Also the default setup appears to work. I tried it out when I messed up my ubuntu server install. Now my setup is relatively safe. None of the fstab stuff is required at all. I may actually create a wiki page to help out for schroot in non-root setups.

Installing and configuring dchroot (deprecated)

This section formerly appeared before the debootstrap section above. The following is here merely for reference. This use of dchroot is deprecated (no longer preferred), so you should probably use schroot as described above.

dchroot is a convenient means of managing chroot environments; with this tool you can have both Dapper, Feisty and even Debian Sid chroots in your Ubuntu install, and using a chroot environment is a simple as `dchroot -c dapper -d`

To get it `dchroot` working in your system, just do the following in a Terminal:

```
 sudo apt-get install dchroot
 sudo mkdir -p /var/chroot/hardy # Remember, we assume our chroot is here
 sudo editor /etc/dchroot.conf
```

Then append this line in `/etc/dchroot.conf`:

```
 mychroot /var/chroot/hardy
```

DebootstrapChroot (last edited 2013-10-21 17:41:47 by barry @ mail.wooz.org[216.15.33.194]:barry)