Debian GNU/kFreeBSD inside native FreeBSD jail

It has been some time now since development on Debian GNU/kFreeBSD started, which aims at bringing together the FreeBSD kernel with a GNU userland.

There exists a similar implementation called Gentoo GNU/kFreeBSD, although I had no time yet to review it.

The Debian developers made some notable progress since last year, but there are still lots of issues. Check out the project website for further details.

Now let's look at an abscure idea, that struck me when I first looked at Debian GNU/kFreeBSD last spring.

Why not try to run Debian GNU/kFreeBSD inside a native FreeBSD jail (or inside a Debian GNU/kFreeBSD jail)?

One might argue where's the point in doing so. But under that premise I could also ask, what's the point then in having hundreds of Linux distros which basically do the same thing -- only in a different way?

Let's imagine the possibilities:

- Run native GNU/kFreeBSD hosts with both GNU/kFreeBSD and native FreeBSD userland jails
- Run native FreeBSD hosts with both GNU/kFreeBSD and native FreeBSD userland jails

If you're familiar with the concepts of OpenVZ/Virtuozzo or the Linux VServer project, you'll see some similarities and maybe even the reason for trying out this weird idea.

I must admit, that the current implementation is far away from production grade. There's a lot of dirty handwork involved. Aparently there is no point in trying to streamline the process right now, since GNU/kFreebSD is still under heavy development. Automated setups should be easy to implement as soon as the Debian Installer is ported and package support is fixed in debootstrap.

Are you ready to read on? Ok, let's go to work then...

Let's have a look at how to setup at installing Debian GNU/kFreeBSD inside a native FreeBSD jail.

First of all, get the Debian GNU/kFreeBSD netinst cd. You'll find it at http://glibc-bsd.alioth.debian.org/install-cd

wget http://glibc-bsd.alioth.debian.org/install-cd/kfreebsd-i386/20061213/debian-20061213-kfreebsd-i386-install.iso

Now attach the iso image to a vnode and mount it.

```
#mdconfig -a -t vnode -f debian-20061213-kfreebsd-i386-install.iso
md0
#mkdir /mnt/debinst
#mount_cd9660 /dev/md0 /mnt/debinst
```

Prepare your new root directory for the jail and untar the base dist. The base dist will take up about 120 MB on disk.

```
#mkdir /var/jails/debian_jail
#tar -xzpvf /mnt/debinst/base/base.tgz -C /var/jails/debian_jail
```

Edit the installer file at /var/jails/debian_jail/native-install and comment out the 'set -e' directive on line 31. The script will throw lot's of errors when running inside the jail. If you leave the 'set -e' directive as is, the script will abort and you will need to start over.

You're ready now to start the jail. Please do it manually as shown, you will need to fiddle around with it.
You can add it to /etc/rc.conf for regular startup later.

#ifconfig fxp0 inet alias 192.168.0.13/32
#mount_procfs procfs /var/jails/debian_jail/proc
#jail /var/jails/debian_jail debian_jail 192.168.0.13 /bin/sh

You are now inside the jail. Run the installer script from the root directory.
The installer will ask you for time zone settings, set them as required.
Answer to the popularity contest question to your own preferences.

#/native-install

The script will install and configure system packages. This will take a while.
Don't worry about errors and warnings, they're ok for now.

After the script has finished it is time for some further work. Don't leave the jail yet!

First create a valid /etc/resolv.conf. Then make sure your /etc/fstab is empty to prevent mount errors.

#echo>/etc/fstab

And don't forget to set the root password.

#passwd

Make sure init won't tamper with /dev.

#update-rc.d -f makedev remove

Replace /etc/init.d/freebsd-utils by a dummy file as shown below and reconfigure the freebsd-utils package.

#echo exit 0>/etc/init.d/freebsd-utils
#dpkg-reconfigure freebsd-utils

Now you must leave the jail for a minute. 'exit' should return you to your host.
Back on the host you need to attach devfs to your jail.

#mount_devfs devfs /var/jails/debian_jail/dev

Then re-enter the jail:

#jail /var/jails/debian_jail debian_jail 192.168.0.13 /bin/sh

It's time to import the gpg keys into apt and refresh your package list.

#gpg --keyserver subkeys.pgp.net --recv CD02E583 && gpg --export CD02E583 | apt-key add -
#apt-get update

As ssh is not part of the base system, I'd recommend to install it now.

#apt-get install ssh

Maybe you'd like to have aptitude as well? Feel free to install whatever you like.

#apt-get install aptitude

Depending on your software choice your jail will now use around 220 MB or more on disk.
Leave the jail by entering 'exit' on the prompt.
Then try to start the jail as shown below.

#jail /var/jails/debian_jail debian_jail 192.168.0.13 /etc/init.d/rc 2
Not starting internet superserver: no services enabled.
Starting OpenBSD Secure Shell server: sshd.

Check if your jail shows up and your processes are running.

#jls
JID IP Address Hostname Path
71 192.168.0.13 debian_jail /var/jails/debian_jail

#pgrep -lfj 71
95086 /usr/sbin/sshd

Your jail should now be accessible through SSH.

#ssh root@192.168.0.13
The authenticity of host '192.168.0.13 (192.168.0.13)' can't be established.
DSA key fingerprint is c5:02:ad:c6:a8:43:30:25:4a:d0:bd:71:ca:cc:f0:25.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.0.13' (DSA) to the list of known hosts.
root@192.168.0.13's password:

The programs included with the Debian GNU/kFreeBSD system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/kFreeBSD comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
debian_jail:~#

So far, so good. Your jail Debian GNU/kFreebSD is now up and running, sort of...

There's a caveat that you may already have noticed. init(8) is not yet working as supposed to, so there's actually no parent init process as it should. Neither is there proper resource or runlevel control. This is definitely an issue which would require further investigation.

Furthermore when playing around with it you'll notice that there are other things not working properly, like process information tools (ps, top, etc), a multitude of errors in init scripts, most of them being access errors due to the jail enviroment, erratic behaviour on package configuration when it comes to accessing device nodes, and many, many more.

However one must consider Debian GNU/kFreeBSD still being under development on one hand, and it was never meant to be actually run inside a jail on the other.

It would be nice if it became a standing feature in the future, maybe even for sidekicks like Gentoo GNU/kFreeBSD and other distros out there.

Another task would be checking the jail functionality vice-versa to see, how a native FreeBSD userland can be run inside a jail on a Debian GNU/kFreeBSD host.
As to my understanding of the existing implementation that shouldn't be too tricky as FreeBSD userland is already fully jail-aware.

I'm looking forward to trying this on my own as soon as the jail utilities become available on Debian GNU/kFreeBSD one day.