Primer: Building the Linux kernel "The Debian Way" (includes Ubuntu)

If you are in a hurry to see it all work (for Hardy Herron), skip to the "Specific Example" section, which includes how to build the kernel for Ubuntu 8.04 with AppArmor and the nVidia kernel module.

Commands that require root privilages are preceeded with '`sudo`' for clarity; on some systems it is more typical to execute '`su -`' to become the root user before executing commands that require root privilages.

Steps:

1. Make directories in your home directory for compiling Linux

```
mkdir ~/src
mkdir ~/src/Linux2.6
cd ~/src/Linux2.6
```

2. Install required pacakges/programs to build the kernel:

```
sudo apt-get install kernel-package libncurses5-dev fakeroot wget bzip2
```

For '`make gconfig`' to work [will install ~48.5 MB of stuff, assuming Gnome is installed]:
```
sudo apt-get install libgtk2.0-dev libglade2-dev
```

For '`make xconfig`' to work [will install ~35 MB of stuff, assuming KDE is installed]:
```
sudo apt-get install libqt3-mt-dev g++
```

For downloading and maintaining source with Git:
```
sudo apt-get install git-core git-doc
```
For Git GUI visualization tools (optional):
```
sudo apt-get install git-gui gitk qgit
```

For custom patching the kernel sources:
```
sudo apt-get install quilt patch
```

3. Configure kernel-pkg.conf

Copy over the global kernel-pkg config file to your home directory:
```
cp /etc/kernel-pkg.conf ~/.kernel-pkg.conf
```
Modify `~/.kernel-pkg.conf` with your name, email address, and ADD:
```
MODULES_LOC := ~/src/Linux2.6/modules
```
This tells `make-kpkg` (the kernel building package) where it should expect to find external module sources

4. Choose and obtain the Linux kernel source code

Three (main) options for where to get the Linux sources from:

A. Get the full stable .tar.bz2 file from http://www.kernel.org (good starting point)
On the main page find the latest "stable" kernel, choose the "F" link on the right-hand side
Save the file to directory `~/src/Linux2.6/`
Or if you want a prior kernel version, download one of the many at:

```
http://www.kernel.org/pub/linux/kernel/v2.6/
```

Expand the source code with:

```
tar -xjvf linux-*.tar.bz2
```

[or `tar -xzvf linux-*.tar.gz` for a .tar.gz]

B. Install the kernel source from Debian or Ubuntu

```
sudo apt-get install linux-source
```

This is a virtual package that has a dependency on the latest linux-source-<VERSION> package available; installs a linux-source-<VERSION>.tar.bz2 file into /usr/src/

Make a softlink to the tarball and decompress it:

```
cd ~/src/Linux2.6
ln -s /usr/src/linux-source-<KERNEL_VERSION>.tar.bz2 \
    ./linux-source-<KERNEL_VERSION>.tar.bz2
tar -xjvf linux-source-<KERNEL_VERSION>.tar.bz2
```

C. Download the Linux kernel source via Git. Common respository choices (choose one):

```
git-clone git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-2.6.22.y.git
git-clone git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-2.6.23.y.git
git-clone git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-2.6.24.y.git
```

Or for more options, visit `http://git.kernel.org`

Update a respository:
Method 1:

```
git-checkout master
git-pull
```

Reason for checkout: `git pull` will complain if you are in a different branch. If you really want to update another branch than the `master` branch, use:

```
git-pull <URL> master
```

where the <URL> is the same Git URL you used to clone the download of Linux.

Method 2:

```
git-fetch
git-checkout master
```

Reason for checkout: `git fetch` gets objects but does not update the checked out files.

How does '`git fetch`' and '`git-pull`' know where to get updates from?
This can be found in `.git/config`. For example, the 2.6.24.x repository contains:

```
[remote "origin"]
     url = git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-2.6.24.y.git
```

5. Optional: Patch the kernel

Patching with the `patch` command

```
cd ~/src/Linux-2.6/
wget http://www.kernel.org/linux/kernel/v2.6/patch-<PATCH_VERSION>
gunzip patch*.bz2    [or gunzip patch*.gz]
cd linux-<KERNEL_VERSION>
patch -p1 < ../patch-<PATCH_VERSION>
```

Example of patching 2.6.24.4 with an AppArmor patch, using quilt:

```
cd ~/src/Linux-2.6/
wget \
http://forge.novell.com/modules/xfcontent/private.php/\
apparmor/kernel-patches-2.6.24.tgz
tar -xvzf kernel-patches-2.6.24.tgz
cd linux-2.6.24.4
ln -sv ../apparmor ./patches
quilt push -a
```

6. Configuration of Linux

```
cd <Linux source directory>
```

Start off with a known configuration (note the . In the 2nd filename):

```
cp /boot/config-<KERNEL_VERSION> ./.config
```

To verbosely copy the config file of the kernel that is currently running right now:

```
cp -v /boot/config-$(uname -r) ./.config
```

To add your own version number to the kernel, either:

A. Edit the top-level "Makefile" and add text to the "EXTRAVERSION = " section
B. During the kernel configuration, edit the "localversion" in
   General -> Local version
C. Make a "localversion" file (top-level) containing what you want to add
   [Note: "crk1" is just my own initials and "version 1" -- i.e. it's an arbitrary addition.]

Examples (these all do the same thing – do ***one*** of them):

In "Makefile":

```
VERSION = 2
PATCHLEVEL = 6
SUBLEVEL = 24
EXTRAVERSION = .4-686-initrd-crk1
NAME = Err Metey! A Heury Beelge-a Ret!
```

In the configuration under "General Setup", fill in the "Local version -- apppend to kernel release" [the CONFIG_LOCALVERSION option]:

```
-686-initrd-crk1
```

The contents of an example "localversion" file:

```
-686-initrd-crk1
```

Update the configuration with one of:

```
make menuconfig
```

or

```
make xconfig
```

or

```
make gconfig
```

7. Optional: Install needed/desired additional external source packages, make soft links, and expand

Examples [these all install compressed tarballs into /usr/src/ when installed via apt-get]:

```
aufs-source
cloop-src
fglrx-kernel-src
kqemu-source
ndiswrapper-source
nvidia-kernel-source
realtime-lsm-source
squashfs-source
tp-smapi-source
```

... there are currently about 70 or so of these in the tree for Debian Unstable.

Make all soft links for external modules:

```
cd ~/src/Linux2.6/
ls /usr/src/*.bz2 | while read BZFILE; do ln -sv $BZFILE ./$(basename $BZFILE); done
```

Or you can make the soft links yourself one at at time:

```
ln -sv /usr/src/nvidia-kernel.tar.bz2 ./nvidia-kernel.tar.bz2
ln -sv /usr/src/squashfs.tar.bz2 ./squashfs.tar.bz2
...
```

Expand the tarballs:

```
tar -xjvf *.bz2
```

8. Compile the kernel into packages and install

Normal build for just the kernel, no external modules, with an initrd image:

```
fakeroot make-kpkg clean
time nice fakeroot make-kpkg –revision=<KERNEL_VERSION>+<YOUR_VERSION> \
 --initrd kernel_image
cd ..
sudo dpkg-i linux-image*.deb
```

Building the kernel + external modules, but without an initrd image:

```
fakeroot make-kpkg modules_clean
fakeroot make-kpkg clean
time nice fakeroot make-kpkg –-revision=<KERNEL_VERSION>+<YOUR_VERSION> \
 kernel_image modules_image
cd ..
sudo dpkg -i *.deb
```

A full real example, kernel + external modules + initrd image:

```
fakeroot make-kpkg modules_clean
fakeroot make-kpkg clean
time nice fakeroot make-kpkg –revision=2.6.24.4+686+initrd+crk1 \
 --initrd kernel_image modules_image
cd ..
sudo dpkg -i *.deb
```

Extract contents of an ramfs initrd image

```
cd ~/src/Linux2.6
cp /boot/initrd.img-<KERNEL_VERSION> ./initrd.cpio.gz
gunzip initrd.cpio.gz
cpio -i < initrd.cpio
```

For more detail about how to mount older initrd.img types [compressed + uncompressed cramfs]:
http://www.ducea.com/2006/06/24/inspecting-the-content-of-an-initrd-file/

Specific Example for Ubuntu 8.04

Ubuntu 8.04 (Hardy Herron)

Using git to download the latest 2.6.24.y series, patching for AppArmor, + building nVidia kernel drivers

```
sudo apt-get install kernel-package libncurses5-dev fakeroot wget bzip2
sudo apt-get install git-core git-doc diff quilt patch nvidia-kernel-source
mkdir -pv ~/src/Linux2.6/
cp -v /etc/kernel-pkg.conf ~/.kernel-pkg.conf
echo “MODULES_LOC := ~/src/Linux2.6/modules” >> ~/.kernel-pkg.conf
cd ~/src/Linux2.6
ln -sv /usr/src/nvidia-kernel-source.tar.gz ./nvidia-kernel-source.tar.gz
tar xzvf nvidia-kernel-source.tar.gz
git-clone git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-2.6.24.y
wget http://forge.novell.com/modules/xfcontent/private.php/apparmor/kernel-patches-2.6.24.tgz
tar xzvf kernel-patches-2.6.24.tar.gz
mv 2.6.24 apparmor-2.6.24
cd linux-2.6.24.y
git-branch apparmor-test
git-checkout apparmor-test
ln -sv ../apparmor-2.6.24 ./patches
quilt push -a
cp -v /boot/config-$(uname -r) ./.config
echo “-testing” > localversion
make oldconfig
# Say Y for VESA 2.0 VGA graphics support
time nice fakeroot make-kpkg --revision=2.6.24.4+testing --initrd \
kernel_image modules_image
cd ..
sudo dpkg -i *.deb
```

Handout for the April 2, 2008 MHVLUG meeting
by Chris Knadle
email: Chris.Knadle@coredump.us

Files related to the talk maintained at:
ftp://ftp.coredump.us/kernel-talk_04-02-2008/
or on the MHVLUG page for the meeting at:
http://mhvlug.org/MonthlyMeetings/2008/04