Installing Debian onto a Software RAID

(and) Supporting a Promise Tx2000 IDE RAID Card

Kevin Griffin
Vancouver Community Network (http://vcn.bc.ca)

griffink@vcn.bc.ca

Ian Clelland

ian@veryfresh.com

How to install Debian GNU/Linux onto a software RAID system, using a recently-made Promise IDE RAID controller. This particular case covers the scenario wherein:

You have a Promise IDE controller that sports a PDC20271 chip, and

You have two western digital hard drives (40 GB+).

This hardware is not supported by the current Debian bf2.4 installation disks (kernel version 2.4.18). Descriptions of proposed solutions.

1. Background

The mission was supposed to be simple: install Debian onto a pair of hard drives and a hardware RAID controller. It’s easy to do, if the Promise controller has the PDC20270 chip on it. This one had a PDC20271 - which is not supported before kernel 2.4.19. So how do you get a customized 2.4.19 kernel into the installation floppies?

2. Building Custom Installation Media

2.1. Motivation

We need to install Debian onto some hardware which is not supported by any of the debian installation sets. To overcome this, we make a custom installation floppy.

2.2. Prepartion

Stuff you’ll need:

Files:

`rescue.bin`, `root.bin`, a kernel configuration file from the "bf" branch of Debian, and a Linux kernel.

- rescue.bin (ftp://ftp.debian.org/debian/dists/woody/main/disks-i386/current/images-1.44/bf2.4/rescue.bin).
- root.bin (ftp://ftp.debian.org/debian/dists/woody/main/disks-i386/current/images-1.44/bf2.4/root.bin).
- kernel-config (ftp://ftp.debian.org/debian/dists/woody/main/disks-i386/current/bf2.4/kernel-config).
- A kernel(2.4.22) (ftp://ftp.kernel.org/pub/linux/kernel/v2.4/linux-2.4.22.tar.bz2)

You will also need two floppy disks. 1.44 MB, 3.5 inch. Don’t try this with your old 5.25 floppies. You’ve been warned.

100-200MB of spare hard drive space, to compile the kernel.

A compiler. GCC 2.95 is highly recommended.

2.3. Compiling the Kernel

Rename `kernel-config` to `.config` in the kernel source directory. Run

make oldconfig

Accept the defaults for every question. The `kernel-config` file is designed for a 2.4.18 kernel. We need to let it know about the new options available in 2.4.22. You can either

1. press "ENTER" for every option
2. **/usr/bin/yes "" | make oldconfig**

then

```
make menuconfig
```

You will defintely need msdos filesystem support - the boot floppy uses this. Also, include support for your network card, and whatever other hardware you might need to install the operating system. Turn on the "experimental code" option in the kernel.

Enable the PDC20271 ("PROMISE PDC202{68|69|70|71|75|76|77} support") under the "ATA/IDE/MFM/RLL support" menu. We had to enable the high-memory support option in the "Processor type and features" menu, because we were in the enviable position of having 1GB of RAM. You may need to delete other options for now; getting the kernel small enough to fit onto an install floppy is tricky. The existing kernel is 1,263,339 bytes long, and your custom kernel should be no bigger than that. Compile the kernel with

```
make dep && make modules && make bzImage
```

make may lie about the kernel image being too big. Check the file size with:

```
ls -l arch/i386/boot/bzImage
```

2.4. Putting it together

Mounting the image:

```
mount -o loop rescue.bin /mnt
```

copy `bzImage` to `linux.bin` under `/mnt`

```
gzip -9 < System.Map > /mnt/sys_map.gz
```

Not necessary but nice:

```
gzip -9 < .config > /mnt/config.gz
```

update `debian.txt` to indicate new kernel version. umount `/mnt`.

```
fdformat /dev/fd0
```

```
dd if=rescue.bin of=/dev/fd0
```

```
dd if=root.bin of=/dev/fd0
```

3. Adding Software RAID support

3.1. Motivation

We’re sorry. If you’ve been following along at home, and bought two large Western Digital hard drives, you’ll notice this didn’t work. For some reason Promise controllers working under Linux don’t play nicely with Western Digital drives over 40 GB, and refuse to recognize the BIOS-defined RAID partitions. The cause of this is unknown to us. However, the particular drives that we were working with are fast and have three year warranties. So, to make use of these drives we’ll install Debian with software RAID support.

3.2. Prepartion

Stuff you’ll need:

go get `lar1440.bin` from http://people.debian.org/~blade/install/lvm/

And one more floppy disk.

3.3. Putting it together

Build a kernel with all of the software RAID options built as modules, and IDE-RAID support removed.

```
gzip -d < lar1440.bin > image.bin
```

```
mount -o loop image.bin /mnt
```

```
cd /mnt
```

```
mkdir -p lib/modules/2.4.22/kernel/drivers/md/
```

```
cp <linux-src-directory>/drivers/md/*.o lib/modules/2.4.22/kernel/drivers/md/
```

```
umount /mnt
```

create a new blank image:

```
dd if=/dev/zero of=lar1440v1.bin bs=1k count=1440
```

compress our expanded lvm and raid tools disk into a floppy size image:

```
gzip -9 < image.bin | dd of=lar1440v1.bin conv=notrunc
```

Time to format a new floppy!

```
fdformat /dev/fd0
```

write it out:

```
dd if=lar1440v1.bin of=/dev/fd0 bs=1024
```

You now have a disk that contains LVM &: RAID tools, supporting kernel 2.4.22!

Now it’s time to follow some instructions from: http://people.debian.org/~blade/install (http://people.debian.org/~blade/install/) on how to use this handy extra modules disk.

4. Sanity, or how to do this with a third drive.

4.1. Motivation

Unfortunately, the above didn’t work for us. I tried installing on a small spare WD hard drive, but the system was still not bootable.

However, the hardware gods are merciful. I found a Maxtor 20GB hard drive that I could use to install the system. From there, I was able to recompile a kernel that supported the PDC20271, initialize a software RAID, and move most of the filesystem onto said RAID.

4.2. Preparation

Stuff you’ll need: A spare hard drive. A Debian bf2.4 installtion CD-ROM. A CD-ROM drive, and a BIOS that will let you boot from there.

4.3. Putting it together

Installation was pretty typical, because the WD and the ide-raid conttroller weren’t even noticed. Nothing to report there.

Rebuilt the kernel, with 2.4.22 and necesssary options built in. (software raid, no ide-raid, etc.)

At this point I should refer you to the Software Raid Howto:
http://www.tldp.org/HOWTO/Software-RAID-HOWTO.html

I used the mdadm command specified there to create a series of RAID partitions that would be useful in this installation:

This information is exported from a gnumeric spreadsheet.

mount point	**RAID level**	**Size (in MB)**	**Physical Size**	**Disk(s)**	**device, primary**	**device, secondary**	**raid device name**	**chunk size in kib (kibibytes)**	**filesystem**
/home	RAID 1	30720	61440	WD 1,2	hde5	hdg5	/dev/md0	32	ext3
/export	RAID 1	5120	10240	WD 1,2	hde6	hdg6	/dev/md1	32	ext3
/var	RAID 1	40960	81920	WD 1,2	hde7	hdg7	/dev/md2	32	ext3
/opt	RAID 1	1024	2048	WD 1,2	hde8	hdg8	/dev/md3	32	ext3
/tmp	RAID 0	1024	1024	WD 1,2	hde9	hdg9	/dev/md4	32	ext2
/var/local/amanda	RAID 0	2048	2048	WD 1,2	hde10	hdg10	/dev/md5	32	ext2
swap	N/A	1024	1024	WD 1,2	hde11	hdg11	N/A	32	swap

Special Notes:

If you create and use a `/tmp` like this, you will need to run **chmod +1777** on it so that the sticky bit is set, and you’ll be able to generate man pages.

Swap needs no RAID (not even RAID0). Disk striping for swap partitions is supported natively by the kernel.

`/tmp` is assumed to be, well, temporary, and is not mirrored. As well this partition was given an ext2 filesystem to save space. The AMANDA holding disk is also not mirrored, in the interests of space.

After initializing and formatting the various RAID partitions, I began mounting them onto the existing filesystem in temporary locations and then copying , for example, **cp -aR /var /mnt/var**. After a successful copy, it remains only to umount the raid from the temporary location and remount it in the permanent one. Edit your `/etc/fstab` appropriately.

5. Thanks goes out to...

The following organizations,people, and programs deserve some kudos:

- Vancouver Community Network (http://vcn.bc.ca), For being a great place to learn.
- The cap (http://cap.ic.gc.ca/english/4000newpro.asp) program, for hiring me (Kevin).
- Fincon (http://www.finbyte.com/) and the IEOGI (http://www.finbyte.com/IEOGI/index.htm) project, for supplying hardware, space, and no small amount of coffee.
- Ian Clelland (http://www.veryfresh.com/ian/), who took a great many calls and suffered mightily during the process of getting this server running.
- The co-ordinators down at CAP-VCN, and their patience. They put up with more stuff from above and below than I care to imagine.
- Ryan Sadler, whose page (http://ttul.org/~rrsadler/linux-promise/) was invaluable. This document was the essential starting point for the rest of the research.