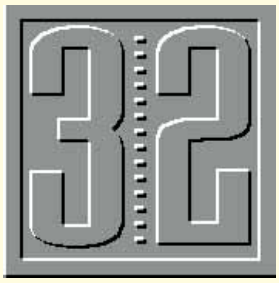

Linux on the hoof

Chris Bidmead managed to load Linux onto his trusty portable so he could take it everywhere with him. Here, he relates his trials, tribulations and triumphs.

My mate Marcus said "Get a life" when I told him I was taking Linux on holiday. He can talk: he's just bought a Gateway 2000 120MHz Pentium machine with 2Gb disk space and a sub-woofer for making Doom sound more dangerous. And all he runs on it is Windows 16-bit. I ask you; is that a life?

I only told him because I was proud to have got Linux onto Tonto, my trusty portable. Marcus has Linux, too, but he runs it on an old 386 machine (where, unlike Windows, it gets by just fine).

A good conversationalist is witty and wise; a *really good* conversationalist brings out the best in others so that *they* feel witty and wise. In this sense, Linux is a brilliant conversationalist because the exercise of getting it across to Tonto has left me feeling very smart indeed, although really of course I'm only piggybacking on the shoulders of the giants who built (and who are continuing to build) the thing.

Caldera, the Linux in question, comes on a CD-ROM. Tonto doesn't have a CD-ROM drive, and there isn't even a SCSI interface to hang one on. The Caldera manual says you can install the operating system over a network: you make a boot diskette and a root diskette in the usual way, and then evoke NFS (the Network File System) to get the thing across from a CD-ROM installed on another machine running Caldera.

The other machine was standing by, but how to connect to it? Although Tonto takes SCSI or network PCMCIA cards I didn't have any handy and there were only a few hours to go before the taxi to the airport. I dug out an old DLink Pocket Ethernet Adaptor — the kind you screw onto the parallel port. I now had the makings of a network machine.

The preview version of Caldera comes with a selection of ready-built kernels, and there's a map in the 125-page Getting Started booklet that tells you which one to use for the combination of network adaptor, SCSI card and CD-ROM device you're using. There isn't a pre-built kernel for every combination: each kernel contains multiple drivers and sniffer software to home in on your particular hardware. This approach avoids tons of unnecessary drivers being included in your particular kernel but will still leave some redundancy. Linux doesn't (as yet) offer full support for dynamic driver loading.

Recompilation

Alas, the map didn't mention the old DLink adaptor. I hoped it might be in there anyway but when I tried the pre-cooked kernel in the "Other Brand not listed here" category, it wasn't recognised. There was nothing for it but to build my own kernel. It's a condition of the Linux licence (and fundamental to its operation as well as its philosophy) that distributions come with their own source code. To cook up your own kernel you recompile the source — all 13Mb of it.

If you're new to Linux this might sound a formidable task. And yes, it is, but not if you follow the footsteps of those giants. They've made recompilation easy by writing scripts that can be read by "make", the utility that controls compilation. No more about "make" now; suffice it to say that I simply read the instructions in the kernel "How To" document, followed those big footsteps and made a few very simple yes/no decisions *en route*.

New kernel

About an hour later I ended up with a new kernel image called zImage (the "z" is a reminder that it's compressed). Compressed is the standard format for Linux kernels these days — "lilo", the boot loader, is primed to deal with these automatically.

Note that zImage is an image of the kernel — unlike the boot images on the Caldera CD, which are images of the whole floppy you use to boot. It's important to be clear about this if you're substituting a home-made kernel of your own. And don't try using "rawrite" to put zImage onto a floppy in the same way that you initially created the boot floppy.

There are a couple of ways of getting your new kernel onto a boot disk. The simplest is probably to create a boot floppy of some kind (I used the dud one I had already tried) and substitute your new zImage for the file called "vmlinuz" already on the floppy. There should be enough room on the floppy, so all you have to do is rename "vmlinuz" — Unix thinks of this as a "move" operation, so the command is something like

```
mv vmlinuz vmlinuz.old
```

— and copy zImage over to the floppy, giving it the new name "vmlinuz". If you've mounted your floppy on /mnt/floppy, for the Caldera distribution the appropriate command to do this will be

```
cp /usr/src/linux/arc/i386/boot/
zImage /mnt/floppy/vmlinuz
```

Linux copes with a whole variety of file systems. Its standard file system is called "Ext2" but the Caldera boot floppies use the cheap and cheerful Minix file system. When you boot with this diskette it first invokes lilo (the Linux boot manager) which will still be looking for the old kernel (it finds it via a disk map, not through the file system, so the change of name won't fool it). You need to re-run lilo to change the disk map. Ordinarily, if you're installing a new kernel on your base system, all you need to do to fix the map is run /sbin/lilo. But on this occasion we want to fix up lilo on the floppy.

Even if you're not a Linux fan it's worth knowing about lilo because it can boot a number of other operating systems as well, and is more flexible than OS/2's Boot Manager and more economical with partitions. We'll be investigating lilo in the coming months, but let's concentrate on this particular problem for now. In theory

we could close everything down, boot from the floppy and run lilo from there, but it's much simpler to evoke our hard disk-based lilo and ask it to regard the floppy as the root file system on which we want to work. lilo accepts a parameter

```
-r <newroot>
```

which allows us to do this, so we can fix up the map with the command

```
/sbin/lilo -r /mnt/floppy
```

This new boot disk, in conjunction with the standard root disk, was all that Tonto needed to get going. Initially, it still didn't recognise the Pocket Ethernet adaptor, but this was easily fixed by reconfiguring the parallel port as LPT2. A second minor problem (translate this as "a hitch that held me back for the best part of an hour") was that the giant who wrote the Caldera installation script hadn't taken on board an official change in the name of the Pocket Ethernet Adaptor port; from /dev/dl0 to /dev/eth0. The simplest way around this is to lie to the installation script and pretend that you're using an internal rather than an external adaptor.

Hair today

Hooray! I now had a small Linux system connected to the network. The next trick was to use NFS to hook it to the CD-ROM running on the desktop Caldera machine. Like much of Linux, this is one of those things that sounds impossibly hairy yet turns out to be astonishingly simple. As long as the appropriate NFS daemons are running on both machines (which by default they are) there are only two things to be done. Firstly, "export" the CD-ROM from the desktop machine. You do this by naming the appropriate directory (in my case, /mnt/cdrom) in a file called exports that lives in /etc. In the case of a first-time NFS installation you may have to create this file (any text editor will do). Then you simply use the standard mount command to attach this directory to a mount point you create on the other machine (my Tonto) on the other side of the network. In this case the file type will be -t nfs and the source device will include the network name of the desktop machine; caldera:/mnt/cdrom. In fact, because this way of writing the source device name is unique to NFS and therefore implies it, you don't need the -t parameter under Linux. Actually it is even simpler because the Caldera install script acknowledges that you might need to do an NFS install, and automates it all for you. I only got to know about the weird science underpinning all this because of the bug-ettes that needed sorting out in this preview version.

Foreign climes

I ran the rest of Caldera's Express install, which loads and configures everything you need to get going, including the X Window

Fig 1 *An alternative way of evoking emacs with the required font is to use Looking Glass's ability to attach command lines to icons. Unfortunately the dialogue box where you set this up refuses to be a target for cut and paste, so the whole tedious font name must be inserted by hand*

Fig 2 *The Caldera font server can hand out Speedo, Adobe Type 1 and TrueType fonts — one of the key features of Caldera I didn't get to play with on my X-less holiday machine*

System. Because Tonto has a double-scan LCD screen I wasn't surprised that the automated X configuration came unstuck and left me with a text screen full of error messages instead of the colourful windowed display. I knew (well, I thought I knew) that I now had everything I needed on board Tonto and could work out the X configuration later.

Alas, I was wrong. It was only when I got to foreign climes that I discovered that the X server I'd installed by default was the wrong one, and the one I needed was still back home on the Caldera CD. An Internet connection could have fixed this in a jiffy but the holiday home didn't even have a phone. Result: my explorations of Caldera were limited to a character interface.

The bad news is that it turns out there isn't really very much to learn about Caldera from this viewpoint because, apart from the NetWare client software, the added Caldera features like the Looking Glass desktop (*Fig 1)*, the Font Server (*Fig 2*), the Icon Editor and the Control Panel, are all X-based. But if you don't know much about Linux (and I'm still close to being a raw beginner) there's tons to be learnt using a simple text screen. As I discovered, Linux, like other modern Unixes, looks very pretty but the real power is under the hood.

A line about Linux

Here's part of a missive from Mark Jessop (m-jessop@dircon.co.uk) that came in just as I was putting the finishing touches to this month's effort: *"Dear Chris, may I start by saying how good your* Hands On *section is; it's one of the first sections of the magazine I turn to."*

Mark goes on to concur with what I wrote a couple of months ago about Linux support: *"The way I see it, Linux is all about having a go yourself and it rather defeats the object to start screaming for support the minute something doesn't quite work as it should, or when you mess it up. You can't really expect vast amounts of free support on a product that is essentially free."*

Mark reckons that: *"Most of the fun is in learning, and the satisfaction gained from making the thing work. I was under the impression that this was why Linux was created in the first place."*

Of course that's true of Unix too — it started out as a very humble platform for Ken Thompson's Space Invaders game. But like Unix, Linux has grown up. These days it's pretty much a professional operating system; witness the Caldera project.

● Complimentary correspondence about this column continues to arrive at a steady pace, so I must be doing something right — at least I'm getting better at responding promptly. Sorry I haven't the space here to thank you all by name.

PD — Phase Change Dual Drive

Add an extra parameter to the AHA15XX device driver so it can handle logical unit numbers (LUNs) and a new DEVICE=OPTICAL.SYS line, and the job's done.

I was roundly impressed with OS/2's ability to cope with the new PD drive that turned up for review. The Plasmon PD 2000 is a curious hybrid: a standard quad-speed CD-ROM drive that also takes 650Mb read/write optical cartridges. I don't normally rave about hardware, but this is a handy SCSI peripheral that you'll probably see a lot of during the coming year or so.

Everybody needs a CD-ROM drive these days and everyone has always needed more storage space. The PD-2000 fits the bill on both counts and unlike previous read/write opticals it isn't frighteningly expensive: street price is around £550 and the cartridges cost about £35 each.

One of the supposed advantages of being a computer journalist is the steady parade of manufacturers touting their wares in front of you — this may sound good but the crunch comes when you ask: "And what operating systems does it work with?" At which point the product manager will usually look at you as if you'd accused him of beating his wife: "Windows," he says. "And?…" you ask encouragingly. The product manager shakes his head — you're obviously too weird for further consideration.

Not so with the new Plasmon drive. A driver for OS/2 is included along with all the usual DOS and Windows utilities. Actually it's a generic IBM optical drive driver but it works fine, giving you a pair of drives; one for the CD-ROM and another for the phase change optical drive.

They both use the same tray and laser but OS/2 handles the changes automatically, as long as you don't jump in and try to access a drive before it has run up to speed. A pair of LEDs on the front panel helps you to track this: the busy light flashes while the new disk or cartridge is settling in, and the PD/CD light shows orange or green according to the media loaded and hence the drive letter you'll be accessing. It would be nice if the operating system could block access to whichever drive happens to be invalid but the generic driver isn't capable of such refinements. The other catch is that it only handles FAT (file allocation table) formatting, not HPFS (high performance filing system).

The PD-2000 also works with a Mac but I was more interested in seeing what Linux would make of it. Because it's SCSI I was moderately hopeful — the various Linux docs breezily state: "All SCSI-based drives should work if the controller is supported." I guessed that Linux would be able to handle the PD-2000 as a SCSI CD-ROM but would be baffled by the optical read/write side of things. But when I swapped out my faithful old NEC CD-38 attached to the Caldera machine and installed the PD-2000, this assumption turned out to be upside down.

I created a mount point called PD in the /mnt directory, and the command

```
mount -t msdos /dev/sda /mnt/PD
```

gave me 650Mb of instant new read/write disk space. It's somewhat slower than a hard disk drive (categorising it as secondary storage) but is actually faster than most of the hard disks among which I spent my early computing years, and perfectly usable as a primary drive if you're pushed. Don't mix this up with the older magneto-optical drives that need a separate turn to erase each track before they can write to it again. Phase-change writes a lot faster because it can overlay old data immediately.

The reason I haven't managed to get it working as a CD drive could simply be that I might just need to recompile the kernel for a Panasonic/Matsushita device. The problem will be more complicated if the SCSI hard drive doesn't know how to give way to a second logical unit on the same SCSI address.

But the joy of Linux is that the source code is all there to be hacked if a problem like this arises. Unfortunately, the joy of Bidmead is that he's a hopeless C programmer. Or perhaps *hopeful* C programmer might be more accurate. Anyway, one way or another I'm fairly optimistic about sorting this out and I will report back to you in next month's column.

PCW Contacts

Chris Bidmead is a consultant and commentator on advanced technology. He can be contacted at **bidmead@cix.compulink.co.uk**