Drive time

A wild-goose chase for an old SCSI 2 drive throws up a few partition problems for Chris Bidmead.

I've been trying to get hold of a small SCSI 2 hard disk to refurbish a Canon object.station very kindly bequeathed to this column by Martin Reed of Apple. Alas the days of SCSI 2 drives (with the 50-pin connector) have long gone and nobody seems to have these around.

I was talking to John Fox, IBM product marketing manager for storage, about this and he promised to hunt around and see what he could come up with. He returned somewhat apologetically a few days later to say that the smallest drive he could find was 9GB, and it was Ultra2 Wide SCSI, with a 68-pin D-type connector.

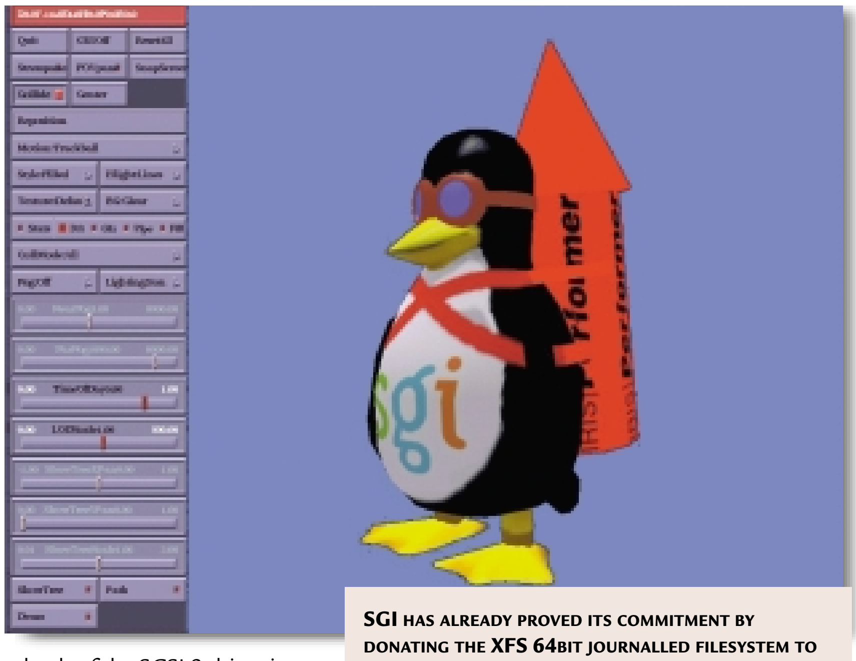
'Capacity should not pose you any integration issues,' John emailed me, and indeed 9GB is a nice advance on the object.station's original 500MB.

'Nor would 68-pin, beyond the necessity of sourcing a 50-to-68-pin adaptor,' he added. That was useful info, because I didn't know you could fit Ultra2 Wide to an old-style SCSI 2 host bus adaptor (HBA). Ah, but there is one issue, John went on to mention. Ultra2 uses LVD (low voltage differential) and needs active rather than passive termination of the SCSI bus signals. 'The bottom line is you need to source an LVD active terminator,' he added.

Well, the problem is I can't swap out the HBA on the object.station because it's an old VL-bus component. The busmastering VL-bus slot on the object.station motherboard is the fastest available – the rest are old ISA slots. Luckily I had another option.

My favourite workstation is still the two-year-old IBM PC315 with its 200MHz Pentium Pro processor. Oddly, although it has an Adaptec Ultra2 Wide controller, it's driving a couple of 2GB SCSI 2 hard disks (or DASDIs, as IBM insists on calling them, although no-one can remember what the acronym stands for). So the logical thing would be to put the new 9GB drive in there and move one or both of the SCSI 2 drives into the object.station.

The PC315 is in active use, and has a hand-tailored Mandrake 6.1 Linux installation on it. Did I feel confident about being able to shift everything across to the new drive without messing up my carefully crafted setup?



SGI has already proved its commitment by donating the XFS 64bit journalled filesystem to the Linux community and the company has also been working with the Linux kernel team to add support for OpenGL graphics libraries to the OS. But now SGI has gone even further by offering its flagship high-performance graphics software toolkit, IRIS Performer 2.3 for Linux, a real-time rendering toolkit for Intel-based systems. This is available as a free download at: www.sgi.com/software/performer/downloads.html. I haven't been able to discover the exact licensing terms – does anybody know?

Alas the days of SCSI 2 drives (with the 50-pin connector) have long gone

Drive migration

It turned out to be easy enough to attach the new 9GB drive. The Adaptec HBA has 50-pin and 68-pin outputs, and I used an Ultra2 cable to hook in the new drive, ensuring it was on a different SCSI device number from either of the other drives (which were, as I expected 0 and 1) and the CD-ROM drive, which was 6. I set the new drive to 5. Linux attributes SCSI drive device names in the order that they're found on the SCSI bus, so when the system booted the new drive was identified as /dev/sdc.

I used FDISK to create new partitions on the drive corresponding to my existing partitions housing /, /usr, /opt, /var and /home. Then I mounted each of these partitions onto temporary mount points and copied data across from the existing partitions on /dev/sda and /dev/sdb. I did this using the cp command with the -a switch, which preserves as much info as possible about the files, and works down through all the subdirectories.

There are a few things to watch out for here. Firstly, cp -a will copy the contents of mounted directories. So cp -a /mnt /tmp/mnt copies not just the mount points (which is what you probably want to do), but also the entire contents of any directories mounted there. Similarly cp -a /dev /tmp/dev is definitely not what you want to do. You won't recreate the device names: you end up copying their entire contents, so /dev/sda, say, will end up as a file containing the whole of your first hard

Netscape: Using Samba

http://www.oreilly.com/catalog/samba/chapter/book/

O'REILLY Online Catalog

PRODUCT INDEX
SEARCH THE CATALOG

Using Samba

Robert Eckstein, David Collier-Brown, Peter Kelly
1st Edition November 1999
1-56592-449-5, Order Number: 4495
416 pages, $34.95

Table of Contents

Back to: Using Samba

O'Reilly Home | O'Reilly Bookstores | How to Order | O'Reilly Contacts
International | About O'Reilly | Affiliated Companies

© 1999, O'Reilly & Associates, Inc.

File Edit View Go Communicator Help
Bookmarks Location What's Related
Search Lookup News Freshmeat Slashdot.org PC Webopaedia NewHoo Topic Internet Server Devon
Samba
100%

Another bird, although I don't know why publishers O'Reilly have chosen a hornbill to represent Samba, the open software file and print server for heterogeneous networks (ie, it provides Windows-type shares). The book is a must for readers like Laurence Crummay

disk drive. (The correct way to recreate these devices, if you ever need to, is to run the script called MAKEDEV you'll find in the /dev directory. This builds the devices in whichever directory you run it in. The Linux man page for MAKEDEV will fill you in on the detail).

Copying /proc is a nonsense, as this is a kind of pseudo filesystem that reflects the current state of play. All you need is an empty directory of that name – the running kernel does the rest.

To stave off email from Unix diehards, I'd better point out that traditional versions of cp on other Unixes, like AIX, don't understand the -a switch. The canonical way of copying a complete hierarchy of directories has always been to use tar through a pipe, something like this:

```
cd /var ; tar cf - spool |
(cd /mnt/spool ; tar xvf - )
```

This is moderately unpleasant to get your mind round, and unnecessary if you use the GNU version of the cp utility, which is portable across every Unix in the known universe.

The general idea was to duplicate my existing setup as closely as possible onto the new drive. But I quickly realised there would be horrendous complications if I went all the way and tried to create a lilo setup on /dev/sdc that would work properly when I removed the other drives, renumbered the new drive as SCSI device 0 and booted it up in its new role as /dev/sda.

Yes, it is theoretically possible. There are some very cunning things you can do with lilo (for clarity, I'm using lilo to mean the setup utility that comes with the operating system, and LILO to denote the actual boot code that gets installed in the boot sector of your choice), and I've done some of them. For example, the -c switch can point lilo to an alternative configuration file, in which you can use the "boot=" parameter to install the LILO boot code somewhere other than your present working LILO setup. So if, for example, you have a BIOS that allows you to choose which drive you boot from, you can preserve (say) /dev/sda as the boot sector for your working setup up, while experimenting with exotic LILO installations on /dev/sdb.

But I realised that figuring this one out was going to take a long time to get right, with multiple reboots in between. So I decided to cheat. To heighten the excitement, I wasn't sure if the cheat would work.

The idea was, having got my system across to the new drive, to see if I could remove the old drives, reset the new drive to SCSI 0 and then boot off the Mandrake 6.1 installation CD, telling it I wanted to upgrade my system. Unfortunately, Mandrake wasn't fooled by this, and failed to recognise an existing Linux system. Perhaps I should have spent some time investigating why, but instead I did the second best thing: a new installation, but without any reformatting of the partitions.

I used FDISK during the installation to assign the various partitions to the main Unix directories I'd planned them for (I've always found pencil and paper essential here, and luckily I had the directory to partition assignments all jotted down). And I minimised the amount of Mandrake material that would pour in from the CD-ROM by choosing Custom Installation and avoiding installing any packages.

If you do this, the basic system stuff still gets installed, and a few key config files get overwritten. The advantage of reinstallation is that this makes it easy to get the LILO boot stuff right. And once I was rebooted with the new kernel, it wasn't hard to reinstate the odd munged config file by hand. I seem to remember

Traditional versions of cp on other Unixes don't understand the -a switch

there were four — /etc/hosts, /etc/resolv.conf), /etc/HOSTNAME and /etc/sysconfig /network. And that was only because I couldn't be bothered to go through the network section of the installation.

The new setup is great, and notably faster, thanks to the Ultra2 drive. I'll let you know how I get on with reinstating the object.station.

Watch those partitions

A general point about partitioning emerged from the exercise and it is worth relaying here. You'll know that PC-style disk drives have a top limit of four primary partitions. The standard way round this is to set up one of these as an 'extended partition', within which you can create multiple 'logical partitions'. Linux labels these from five upwards, so in the case of SCSI drive 0 they'll be /dev/sda5, /dev/sda6 and so on. Logical partitions are more flexible than primary partitions because the information that defines them is stored in a linked list. A side effect of this is you have to be careful when deleting logical partitions. Take a look at Fig 1.

/dev/sda1 is a primary partition, and the extended partition, /dev/sda2, occupies the rest of the drive. Logical partitions five through 10 lie physically within /dev/sda2, but if you inspect the start and end addresses you'll notice that /dev/sda9 and /dev/sda10 are really seventh and eighth in the physical order. How so?

The original /dev/sda7 and /dev/sda8 were equal sized, and allocated to /var and /usr respectively. However my /usr was huge and stuffed full, whereas /var was barely two per cent used. So I went into single-user mode (telinit 1 from the command line), backed up the whole of each partition, unmounted them, and then entered FDISK.

What is the GIMP

The GIMP is the GNU Image Manipulation Program. It is a freely distributed piece of software suitable for such tasks as photo retouching, image composition and image authoring. This site contains information about downloading, installing, using, and enhancing GIMP. This site also serves as a distribution point for the latest releases, patches, plugins, and scripts. We also try to provide as much information about the GIMP community and related projects as possible.

What's this site about?

Well, The Gimp is an image manipulation program, written by Peter Mattis and Spencer Kimball, and released under the GNU General Public License (GPL). It is a program for UNIX and X. To learn more about this program, just read on on this page.

SGI MAY BE PORTING ITS HIGH-END GRAPHICS TOOLS TO LINUX, BUT THE FREE SOFTWARE COMMUNITY HAS ALREADY DEMONSTRATED WITH PRODUCTS LIKE THE GIMP THAT PROFESSIONAL-QUALITY GRAPHICS APPS CAN BE WRITTEN FROM SCRATCH IF NECESSARY

Now it was simply a matter of deleting /dev/sda7 and /dev/sda8 and recreating them with their sizes allocated more appropriately. When I say simply...er... well, it's certainly easy enough, and this kind of low-level messing about is exactly what runlevel 1 (single-user mode) is for. But watch out for those linked lists!

If you delete partition 7 and then partition 8, the second deletion will actually remove /dev/sda9. This is because following the first deletion, the partitions numbered above it all decrement by 1. So always begin your deletion with the highest number and work downwards. Of course, until you actually write the new partition table to disk you haven't changed anything, so provided you're careful you can experiment extensively with partition deletion and creation. A useful tip if you're inexperienced with partitions (which I haven't followed here) is to make every partition a different block size. Then you can associate that unique number in column 4 with its functional mount point when you jot down your directory allocations. /usr = 1510078 will remain true while you delete and create other partitions. As we've seen /usr = /dev/sda9 is written on water.

You'll notice that no Windows partitions were involved – the complication of trying to preserve those wretched DOS-style drive letters as I swapped partitions around would probably have scared me from getting started. If you're using a dual-boot Windows/Linux system, as I know many readers are, you may be inclined to accuse me of short-changing you: I should have faced the challenge. Alternatively, you might see this as another example of how half-baked ideas in Windows hold you back from control of your system. In which case you'll appreciate the value of my ongoing advice to reduce your dependency on Windows.

FIG 1

```
[root@pc315 bidmead]# fdisk -l/dev/sda

Disk /dev/sda: 255 heads, 63 sectors, 1115 cylinders
Units = cylinders of 16065 * 512 bytes

Device Boot   Start    End      Blocks Id System
/dev/sda1 *   1        115      923706 83 Linux
/dev/sda2     116      1115     8032500 5 Extended
/dev/sda5     116      230      923706 83 Linux
/dev/sda6     231      345      923706 83 Linux
/dev/sda7     592      608      136521 82 Linux swap
/dev/sda8     609      1115     4072446 83 Linux
/dev/sda9     346      533      1510078+ 83 Linux
/dev/sda10    534      591      465853+ 83 Linux
```

PCW CONTACTS

Chris Bidmead welcomes your comments on the Unix column. Contact him via the PCW editorial office or email unix@pcw.co.uk