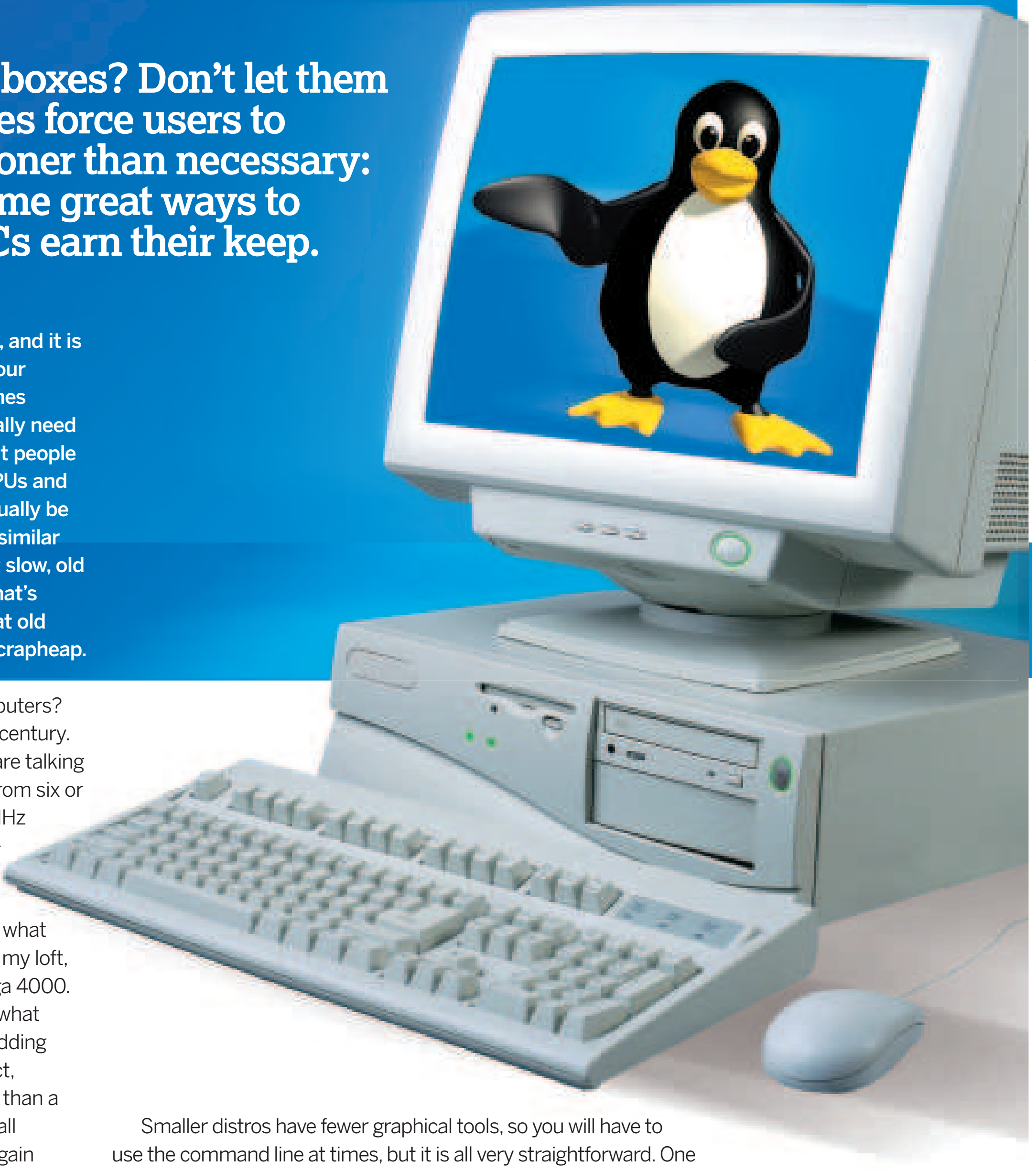
Resurrect your old PC



Nostalgic for your old beige boxes? Don't let them gather dust! Proprietary OSes force users to upgrade hardware much sooner than necessary: Neil Bothwick highlights some great ways to make your pensioned-off PCs earn their keep.

Hardware performance is constantly improving, and it is only natural to want the best, so we upgrade our system from time to time and leave the old ones behind, considering them obsolete. But you don't usually need the latest and greatest, it was only a few years ago that people were running perfectly usable systems on 500MHz CPUs and drooling over the prospect that a 1GHz CPU might actually be available quite soon. I can imagine someone writing a similar article, ten years from now, about what to do with that slow, old 4GHz eight-core system that is now gathering dust. That's what we aim to do here, show you how you can put that old hardware to good use instead of consigning it to the scrapheap.

So what are we talking about when we say *older* computers? The sort of spec that was popular around the turn of the century. OK, while that may be true, it does make it seem like we are talking about really old hardware. A typical entry-level machine from six or seven years ago would have had something like an 800MHz processor, Pentium 3 or similar, 128MB of RAM and a 20-30GB hard disk. The test rig used for testing most of the software we will discuss is actually slightly lower spec, it has a 700MHz Celeron processor, because that's what I found in the pile of computer gear I never throw away in my loft, right next to my faithful old – but non-functioning – Amiga 4000. We set the base spec at 128MB of RAM (because that is what machines of this type were typically supplied with), but adding more can provide a substantial performance boost. In fact, doubling the RAM to 256MB would give a greater benefit than a faster CPU, at far less cost. Very few people buy such small amounts of RAM these days, so they can be found at bargain prices. I needed 128MB for the test machine (it already had 256MB) and bought two brand new 64MB sticks of IBM memory for £0.99 on eBay. Where relevant, we will also cover software alternatives that will work on lesser hardware.

Smaller distros have fewer graphical tools, so you will have to use the command line at times, but it is all very straightforward. One of the great things about using an old machine for this is that you can experiment to your heart's content, and if you make a total hash of things you can just reinstall and start again without messing up your main computer with years of carefully collected data.

»

» What can you do with an older PC?

Anything was considered the domain of a powerful system a few years ago. We were quite happy using 500MHz to 1GHz computers for web browsing, email, work processing and even gaming. While the demands of gaming have moved on, the others don't require any more power than they used to, though some of the programs are now too heavyweight for the 128MB of memory that was considered plentiful in those days. First, we will look at setting up a general purpose desktop system suitable for use on limited hardware; from choosing a suitable distro through installing it, to selecting suitable apps for the main tasks. Later we look at some of the more specialist uses for older hardware, for those of you that don't need yet another desktop computer and are quite happy with your quad-core CPU system for writing emails.

Pick a distro

Ask about a lightweight distro anywhere and it will not be long before someone mentions Damn Small Linux (DSL). Well, we have mentioned it now, but we are not going to use it. DSL does a fine job of cramming a lot onto a 50MB CD, but this limit imposes some artificial restraints, such as the requirement for a 2.4 series kernel. We don't care how big the install CD is, as long as the system installs easily and runs well on our hardware. A 2.6 kernel is a requirement, because some hardware only works with newer kernels. This isn't a problem with the basic hardware, after all, it is older than 2.4, but what if you want to connect something like a webcam that has drivers that only work with the 2.6 kernels. How about the convenience of automatically mounting USB flash disks, CDs and digital cameras? Do you really want to go back to the hit-and-miss, will-it-won't-it excitement of supermount? Puppy Linux was also considered but was similarly lightweight, either of these would be suitable for less capable hardware. We also looked as Xubuntu, a variant of Ubuntu using the same Xfce desktop as Vector, but its memory requirements were higher. Xubuntu is definitely worth a look if you have 256MB of memory.

After looking at the options, we picked on Vector Linux (**www.vectorlinux.com**), a Slackware derivative that is ideal for those with less well-endowed hardware, with the website stating it is "specifically designed for use on older computers with slower processors and less RAM". There are different versions of Vector available, we used the Standard edition for this, but there is also a SOHO release. Both are at version 5.8, with a release candidate of the 5.9 standard version also available at the time of writing. The SOHO release uses a KDE desktop and includes *OpenOffice.org*, both of which are somewhat heavyweight for our old box, and the combination could bring it to its knees. The standard edition has a minimum requirement of a P200 and 96MB or RAM, making it ideal for our test system. We use the 5.9 release candidate, although a final release may be out by the time you read this.

The Vector Standard edition uses the Xfce desktop, together with the highly-rated *Thunar* file manager and a complete suite of applications. Not all of the names may be familiar to you, some of the applications are alternatives to the more resource hungry programs you already know and use, but for this environment, they are perfect. See the *Diet Software* box on page 49 for some lighter alternatives to the usual suspects.

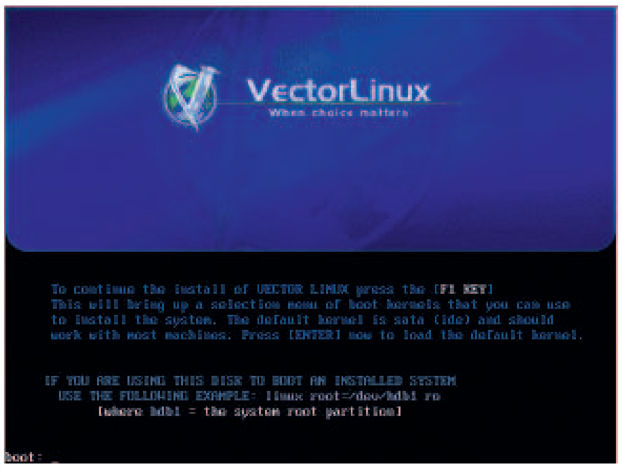
Installation

There is an ISO image of the latest Vector release candidate on the cover DVD, burn it to a CD-R in the usual way then boot from it to install. This is very much like installing Slackware, with somewhat less formal messages (like "INSTALL – come on mate, go for it!"), using the same *curses* (text) installer. This may lack some of the bells and whistles of the fancier graphical installers, but it works efficiently, even on limited hardware. There were a couple of problems that arose during installation that it would help to be aware of, rather than finding out towards the end of the process.

The 5.8 installer hung while probing the graphics hardware on two systems and needed a reboot. The only way to avoid this was to skip this step. After booting, you only see a text login, because X is not configured, but logging as root and running **xorgconfig** soon fixed that. It was easiest to log into a second console and run **lspci** on that, skipping between the two with Alt+Left/Right to get answers to the questions answered by **xorgconfig**. Vector 5.9 had no such problems, but did require more disk space for a full install; if you have a 5GB or smaller drive, either use 5.8 or be selective during the installation, there is a place to choose which package groups you wish to install.

What's in the box?

Ask people to list the classic applications used on current computers, particularly the *GTK* alternatives, and you will hear



```
To continue the install of VECTOR LINUX press the [F1 KEY]
This will bring up a selection menu of boot kernels that you can use
to install the system. The default kernel is sata (ide) and should
work with most machines. Press [ENTER] now to load the default kernel.

IF YOU ARE USING THIS DISK TO BOOT AN INSTALLED SYSTEM
  USE THE FOLLOWING EXAMPLE: linux root=/dev/hdb1 ro
      [where hdb1 = the system root partition]

boot: _
```

› **Boot from the CD to start the basic looking, but easy to use, Vector Linux installer.**

Suitable software

We have set the computer up to be safe for a child to use, but where's the fun? So far we have only disabled launching of most programs and cut out a lot of the Internet, what should you install? Naturally, that depends on the age of the child, but here are some favourites.

» GCompris
A suite of educational programs for children aged from 2 to 10.
http://gcompris.net

» Childsplay
More educational games for younger kids.
http://childsplay.sourceforge.net

» Tux Typing
Typing tutorial with lots of eye-candy.
www.geekcomix.com/dm/tuxtype

» lletters
Game that helps young kids learn their letters and numbers.
http://lln.sourceforge.net

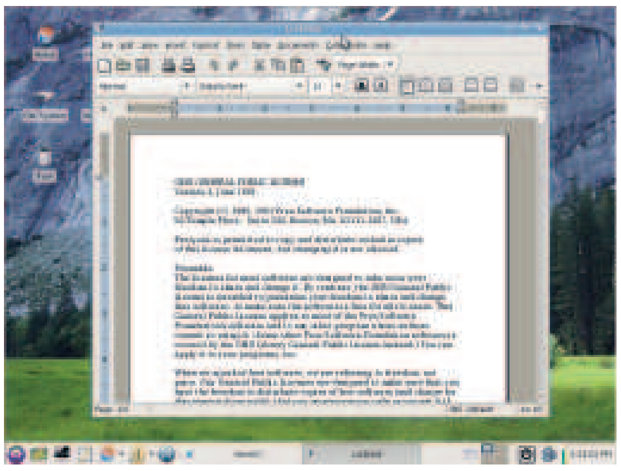
» TuxPaint
The Gimp for kids! **www.tuxpaint.org**

» Memonix
A pack of brain teasers, puzzle and memory games. Annoyingly, it only mentions Windows on the front page, but follow the download link with a Linux browser and you will get the downloads for Linux. Why hide your light under a bushel, Memonix? **www.viewizard.com/memonix**

» Extreme Tux Racer
The latest variant on the classic *Tux Racer.*
www.extremetuxracer.com

There are plenty more educational programs that are available for Linux, so many that there are at least two websites devoted to them, Schoolforge at **www.schoolforge.net** and Tux4Kids at **http://tux4kids.alioth.debian.org**.

There is also a distro dedicated to this, Edubuntu, which, if you hadn't already guessed from the name, is an Ubuntu derivative. Its requirements are too much for our poor old PC, but it is worth looking at the package list at **www.edubuntu.org** to see what's available.



› *AbiWord* **does most of what** *OpenOffice.org Writer* **can, but you can write a letter, print it and post it with** *AbiWord* **before the** *OOo* **splash screen has even finished loading.**

names like *OpenOffice.org*, *Firefox*, *Mozilla* and *Evolution* a lot. These are all highly capable programs, but all that capability comes at a price in terms of the memory, processor power and, to a lesser extent, hard disk space they need. Running *OpenOffice.org* on a 128MB 700MHz computer would be painful at best. By the time it had finished loading, you would probably have forgotten the letter you wanted to write. Running KDE applications would be even more of an overload, not because KDE needs more but Xfce is already using *GTK*, so you would need two graphic interface toolkits loaded at once.

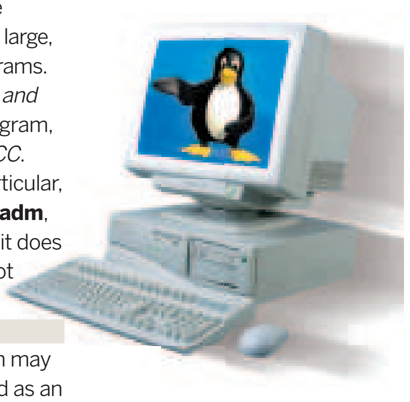
However, it is unlikely you will ever need everything that *OpenOffice.org* has to offer, certainly not all at once, and there are much lighter alternatives. Vector provides *AbiWord* for word processing, with *Gnumeric* for spreadsheets. Both of these can save and load files in *OpenOffice.org* and Microsoft formats, so you still have full portability of your data. Both are mature programs with good core feature sets, missing only the esoteric functions that are mainly used to write marketing copy. *Firefox* is present, along with *Opera*. Yes, it is a bit big and noticeably slower to load than on a modern machine, but not much else comes close if you want to be able to use a large proportion of web pages. The performance was acceptable on our test rig, but if you are running anything much slower, particularly with less memory, Vector provides *Dillo* too. *Dillo* is the browser of choice for Damn Small Linux, which gives you an idea of how compact it is, but it is also amazingly fast. Dillo doesn't do HTTPS, so you won't be able to use it for online banking, but for many sites the speed more than makes up for the limitations.

Slackware is known for its rather, well, basic package management. Vector uses the same package system but includes the *slapt* tools, which work like the Debian programs but with Slackware packages. *Slapt-get* is for command line package management while *gslapt* is similar to the *Synaptic* package management GUI so familiar to Ubuntu users. This already has a couple of online repositories set up, you just need to press the Update button to download the latest list of available packages. Then you can search for and install programs with a point and a click (or several). The range of available programs isn't as large as with some of the more well-known (and well-funded) distros, but as this is basically Slackware, you can install from their repositories too, all you need to do is tick a couple of boxes in the *gslapt* preferences. Alternatively, when a program has a Slackware package available for download, you can usually download and install it with **installpkg** – you can often find these at **www.linuxpackages.net**. Finally, the development tools are available with a standard install, so installing from source is usually straightforward, if a little slow on older hardware, when no dedicated package is available.

Mail is handled by *Mozilla Mail*, which is a surprising choice when there are excellent *GTK* mail programs like *Claws Mail* and *Sylpheed*. These are fast and lightweight, and not short of features either – personally, I use *Claws Mail* on my dual core desktop and my Nokia N800: it's nothing if not flexible.

Child's play

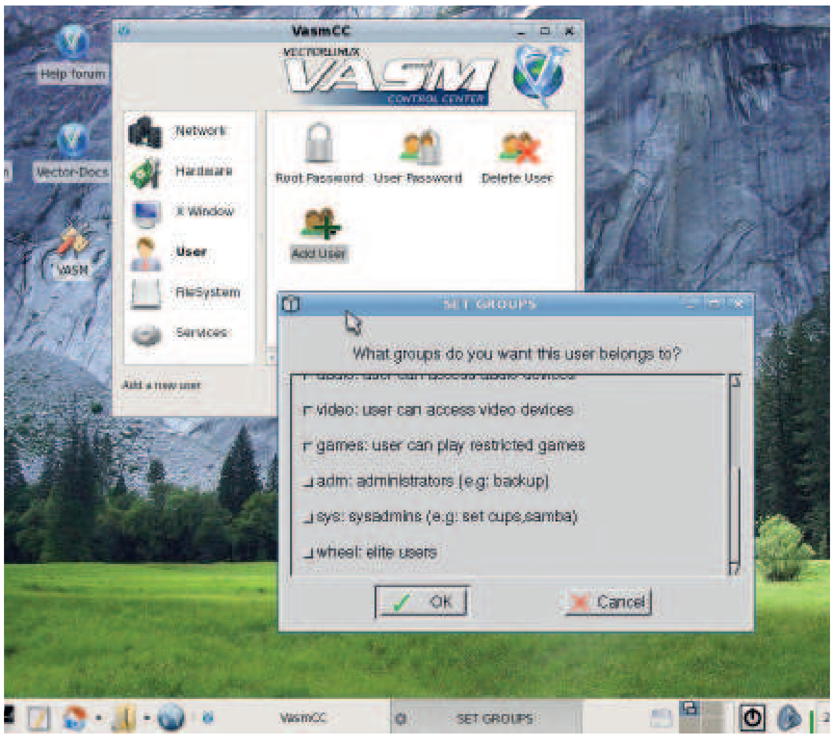
A lower-powered machine makes an ideal first computer for the younger members of the family. Unless they want to play 3D games, in which case they probably have a games console anyway; a sub-1GHz computer is ample for a child to learn the basics of using a computer and run some educational software. You need to set up a separate account for your child to limit what they can do with the computer. You certainly want to restrict their access to system directories, lock them out of using **su** or **sudo** and possibly restrict their access to the Internet. A separate account also gives the opportunity to set up a more child-friendly desktop, including large, colourful icons and a means of launching their favourite programs.



You add users using either *VASM* (*Vector Administrative and Services Menu*), Vector's original system administration program, which runs in X or text mode, or the purely graphical *VASMCC*. Create a new user but limit the groups they belong to. In particular, don't make them a member of anything administrative, like **adm**, **sys** or **wheel**. Vector creates a new group for each user, but it does make all home directories world readable, which you may not want, so open a root terminal and run

```
chmod o-rwx /home/*
```

VASM allows you to create a user without a password, which may be a good idea for a younger child, but shouldn't be regarded as an easy way out, you may as well get your children into security habits as soon as possible. If this computer is mainly for the use of one child, you can set it up to automatically login for that user. Vector uses the KDE display manager, kdm, but doesn't have the rest of the KDE infrastructure so you'll need to edit a configuration file by hand to enable this. The file is ***/usr/share/config/kdm/kdmrc*** and must be edited as root. Go to the section headed **[X-** »



› **When adding a user for your child, do not make it a member of any administrative groups, or you will get a broken computer – guaranteed!**

» :0-Core] and find the **AutoLoginEnable** and **AutoLoginUser** entries. Remove any leading comment signs (**#**) and set them to **true** and the name of your user respectively and save the file. You may also want to set **AutoLoginAgain**, to stop them trying to log out and back in as another user. It depends on how inquisitive or destructive you think your kids are, although, in my experience, it is impossible to underestimate these qualities! You can still login as your user, which you will need to do to administer the system from time to time, either by switching to a virtual console with Ctrl+Alt+F1 or using Ctrl+Alt+Backspace to kill X. Then you can log in at the console and run **startx** if you want a GUI.

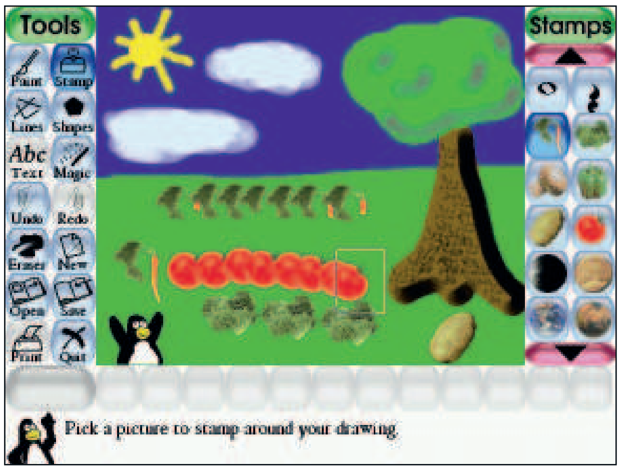
Limited options

You probably want to restrict the options available to a child, particularly a young one. You can do this by removing the panel and menu and putting icons on the desktop for each program you want them to be able to use. To remove the panel, right-click on it and select remove. Alternatively, right-click on the menu button, select remove and repeat this process for the other icons you no longer need. Then add the applications you want to the panel. To add icons to the desktop, browse **/usr/share/applications** and drag any icons you want to display into **~/Desktop**. You will need some attractive icons and wallpapers. There are plenty of Xfce wallpapers and themes at **http://xfce-look.org**, allowing you to make the desktop as child-friendly as you want.

When you have finished changing the settings, you probably want to lock them down, which you can do as root with

```
chown -R adultuser: /home/childuser/.config/xfce4
chmod -R +r /home/childuser/.config/xfce4
```

This will prevent the child user writing anywhere in the Xfce configuration directory, which is where most settings are stored, while still allowing the settings to be read. You can use the same principle to protect settings in other parts of the home directory.



› Discover the nascent artist in your child with *TuxPaint*.

Older and slower hardware?

We have been looking at using a PC in the 500-800MHz range with 128MB of RAM, but what if you are using something much more limited. We are not going to get into 386 dinosaur territory here, but there are plenty of old desktops and laptops floating around with 200-300MHz processors and 64MB of RAM. These are too limited for even the Xfce desktop of Vector Linux but there are other options. There are also lighter Windows managers, in descending order of size, you could try IceWM, Fluxbox or Ratpoison (which will run on anything with more memory than a goldfish).

Damn Small Linux and Puppy Linux will run a graphical desktop on a box with as little as 32MB and are one way to get more use out of such hardware, but you may be better off considering something more specialised than desktop use. This sort of hardware is fine for running a basic file server or a low usage web or mail server, but excellent for use as an Internet gateway. Yes, an old desktop computer is bigger and noisier than a Netgear or Linksys router, and it has less blinkenlights, but it is also far more capable and flexible when running a dedicated distro like SmoothWall or IPCop. Install one of these on the box, disconnect the keyboard and monitor and shove it in a cupboard. All configuration is done via a web browser, you don't need physical access to it once it is installed.

What do you do if your system is so old it doesn't have a CD-ROM, or has one but cannot boot from it. In the latter case, *Smart Boot Manager*, which you will find in the Essentials directory of every **LXFDVD**, can be copied to a floppy disk. Boot from this and then select the CD-ROM from the menu to transfer the boot process to the CD. Installing with no CD-ROM is less easy. Any computer old enough to have no option drive is unlikely to have USB either (particularly bootable USB) or PXE net booting, so you are stuck with one of the minimal floppy-based distros or looking for one that can d a network install from a floppy. Debian is one choice for this, with details at **www.debian.org/distrib/netinst**. Provided the floppy disk is able to bring up your network the rest of the installation should proceed as normal, albeit somewhat slower. There's also an interesting looking method of doing this without even a floppy disk at **http://marc.herbert.free.fr/linux/win2linstall.html**, although we haven't tried it.

Safer browsing

Restricting internet use is a little more tricky, unless you want to take the blanket approach of preventing any Internet access from this machine, which is generally as simple as blocking it in your router's settings. If you want to control Internet access without blocking it, you need a filtering proxy server running on another computer. You could do this with an old machine running one of the distros intended for this, such as SmoothWall, ClarkConnect or IPCop (how many old computers did you say you had lying around the place?) In this case you would probably use a combination of *Squid* and either *SquidGuard* or *DansGuardian*. *SquidGuard* and *DansGuardian* both work with *Squid*, which is a proxy server, to provide filtering, but they differ in the way they filter. *SquidGuard* uses lists of sites to block or redirect requests to those considered unsuitable, while *DansGuardian* filters on the content of requested pages. While either combination could run in the background on your desktop, they are usually used on a separate machine. They can take some time to set up.

An alternative is to run a simpler proxy on your own desktop machine, although that would be dependent on your computer always being available when internet access is required, which is fine if you don't dual boot. One such program is *Willow*, which is a content filter available from **www.digitallumber.net/software/willow**. Download the archive and, as root, unpack it to **/var** with

```
tar xf willow-3.18.tar.gz -C /var
```

copy the appropriate startup script to **/etc/init.d** and add **willow** to your default runlevel. If you get an error about being unable to open the *exefilter* module, remove the reference to *exefilter* by editing **/var/willow/willow.conf** and changing

```
filters = ['domainfilter','exefilter','contentfilter']
```

to

```
filters = ['domainfilter','contentfilter']
```

The **exefilter** isn't necessary if you are using it to protect a Linux box. Now all you need to do is set the default browser and configure it to use the *Willow* proxy. Starting the browser for the first time selects the default, select *Firefox* and then go into Edit > Preferences > Advanced > Network > Connection Settings and manually set the HTTP Proxy to the hostname of the machine running *Willow* and port 8000. On the computer running **willow**, run

```
tail -f /var/log/messages
```

and load a web page into *Firefox*. You should see some messages scroll past as *Willow* fetches the data and passes it to *Firefox* on the other computer. The content of these messages is not so important, as long as the page loads, and willow does not scream any obvious errors, you know that browsing on the old computer is protected. Now you need to make sure it stays that way by making the preferences file read-only with

```
chmod -w .mozilla/firefox/XXXXXXXX.default/prefs.js
```

Where **XXXXXXXX** is a string of random characters, you'll have to look in the directory (or use tab-completion) to see the actual name to use. It is also possible to extract this from **.mozilla/firefox/profiles.ini** with *awk* or *grep* and *sed*, but for a one-off, looking in the directory is easier, even if it scores less geek points. OK, if you really want the geek points, use this

```
chmod -w .mozilla/firefox/$(awk -F \= '/^Path=/ {print $2}' .mozilla/firefox/profiles.ini)/prefs.js
```

One word of warning about *Willow* – in order to filter effectively from the start, it comes with samples of good and bad sites and content. The bad sites include explicit pornographic sites and equally undesirable content (which is why we haven't included it on the DVD) so do not install this on the computer to be used by your child, only on the one you will use as the filtering proxy. The bad content cannot be viewed through the proxy server, only by browsing the filesystem.

You probably also want to block popups in *Firefox*, not only are these annoying and potentially confusing for children, many of them have unsuitable content. Go to **https://addons.mozilla.org** and search for "adblock", the results give you the choice of the original *Adblock* extension or the newer *Adblock Plus*, either of which will banish those annoying popups, an addition that is useful for all children under the age of 195!

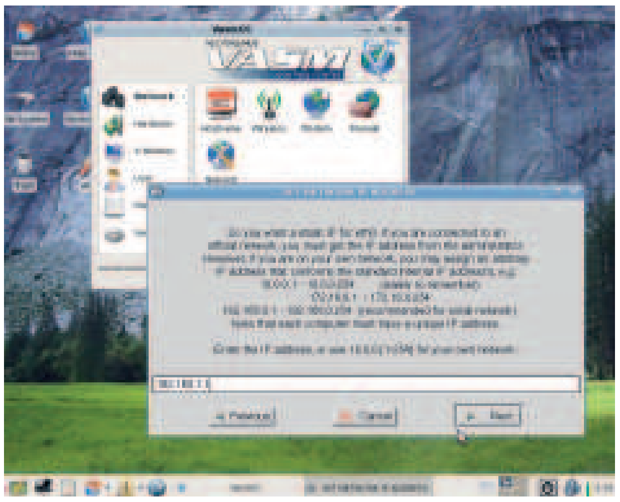
At your service

There are a number of specialised uses for an old computer. Running X uses quite a lot of resources, so anything that doesn't need a graphical display is a good candidate for a slower machine. There are various types of server that you could run on it, either because they fulfill a specific need or just because it is fun to try new things. You may want to install Vector without X in this case, although it costs you nothing but disk space to install with X but not use it.

If you are going to run any kind of server, you will need a static IP address. The default setup of almost every distro is to use DHCP to get an address from your modem/router, which is no good if you want to connect from another computer. Run **VasmCC** and go to Network > Netconf. Select static configuration and give an unused IP address in the range used by your router. Your router's configuration page should show this, but stay out of the range allocated for DHCP addresses. In general, DHCP uses the higher numbers for the last part of the address, say 192.168.1.100 to 192.168.1.200, so use a lower number to avoid conflicts. The gateway and DNS should be set to the address of your router.

Mail without spam?

Running your own mail server may seem like overkill, but there are a number of good reasons for doing so; you can



› To run a serve of any kind, you need to use *VasmCC* to give your computer a static address.

» Access your mail from more than one computer, say a desktop and a laptop, keeping both in sync

» Download mail from several email accounts, your ISP, Google mail, *etc*, and put them all in one place

» Download mail for all family members and sort it into separate mailboxes. Mail downloads are much faster, because the slow part of pulling it for the ISP has already been done in the background.

» Run spam filtering software to sort out the rubbish before you even fire up your mail program.

The last one is particularly important if you are providing Internet access for a child, since the content of some of the spam is offensive to most adults, let alone children. There are three aspects to mail serving: receiving mail from outside and delivering to local mailboxes, serving mail for the local mailboxes to mail clients, receiving and forwarding outgoing mail from the clients. For most people, the third stage is not necessary, but fell free to install something like *Postfix* if you want to try it out. For the first step, the standard program is *fetchmail*. Install this and run **fetchmailconf** to generate a configuration file. Alternatively, use this with the appropriate substitutions:

```
set daemon 300
poll mail.myisp.com with proto POP3
  user 'myispuser' there with password 'mypass' is 'myuser' here
options keep
mda '/usr/bin/procmail -d %T'
```

and set it to be readable by only your user with

```
chmod 600 ~/.fetchmailrc
```

myispuser and **mypass** should be the login and password for your ISP mailbox while **myuser** is your local username. This tells *fetchmail* to check your ISP's mail server every five minutes (300 seconds) and deliver the mail using *procmail* instead of looking for a local mail server, which should be installed by default. The keep option leaves the mail on the server, remove this once you are sure everything works. You can have multiple "poll… user…" stanzas to collect mail from more than one mailbox. Save this as **.fetchmailrc** in your home directory. Configure *procmail* to deliver the mail by putting this in **~/.procmailrc** (or **/etc/procmailrc** if you will be collecting mail for more than one user).

```
MAILDIR=/var/spool/mail
DEFAULT=$MAILDIR/$LOGNAME/
```

»

Diet software

While the likes of OpenOffice.org and Gnome are highly capable, they do work best with plenty of RAM and a good helping of CPU cycles. Here we list lighter, but still very capable, alternatives to some of the programs you are more familiar with. All of these provide the main features of their better known brethren, but are more suited to older hardware.

» *OpenOffice.org*
AbiWord for word processing
Gnumeric for spreadsheets

» KDE or Gnome
Xfce and the *Thunar* file manager
IceWM and the *ROX* file manager

» *Firefox* or *Konqueror*
Dillo

» *K3b*
Xfburn

» *Evolution* or *KMail*
Claws Mail or *Sylpheed*

It is a sign of what people want from their browsers, or how much work is involved in creating a good one, that there is very little in the middle ground between *Dillo* and *Firefox/Konqueror*. Some of the apparently lighter browsers, like *Galeon* and *Epiphany*, require several Gnome libraries and packages, making them not-so-lightweight on a non-Gnome system.

› It may be big, but *Firefox* is still the first choice of web browser.

» The trailing **/** is important, it tells *procmail* to use **maildir** storage, which is needed by the IMAP server later. As root, create the mail directory with

```
mkdir -p /var/spool/mail/myuser
chown myuser:mail /var/spool/mail/myuser
chmod 770 /var/spool/mail/myuser
```

Test it with

```
fetchmail --daemon 0 -v
```

This turns off the background mode and shows what's happening. If it works you can set *fetchmail* to run automatically by selecting Settings > Autostarted Applications, pressing Add and typing **fetchmail** in the Command box; use your imagination for the other two. *Fetchmail* will now run each time the desktop loads. However, you probably want it running all the time, irrespective of the desktop, so instead you can add a line to the end of **/etc/rc.d/rc.local**

```
su myuser -c fetchmail
```

All commands in **rc.local** are run as root, so **su** is needed to run it as your user (running **fetchmail** as root is considered bad practice). You can repeat this process for each user for whom you need to collect mail, putting their details in **.fetchmailrc** in their home directory and a line to start *fetchmail* for them in **rc.local**.

Delivering the post

Now you have the mail on your server, you need users to be able to collect it. There are a number of IMAP and POP3 servers available, my preference is for *Dovecot* (**http://dovecot.org**). This isn't in the Vector or Slackware repositories so download the package file from **www.linuxpackages.net** and install it with

```
installpkg dovecot-1.0.7-i486-1kjz.tgz
```

Once installed, configure it by copying **/etc/dovecot/dovecot-example.conf** to **/etc/dovecot/dovecot.conf** and load it into your favourite editor. Find the section for "Mailbox locations and namespaces" and add

```
mail_location = /var/spool/mail/%u
```

and change

```
#ssl_disable = no
```

and

```
#disable_plaintext_auth = yes
```

to

```
ssl_disable = yes
```

and

```
disable_plaintext_auth = no
```

You don't need SSL authentication if you are only using it on a home network. If you want to be able to access this server from outside, you need to read the documentation on SSL to allow secure logins. Comment out the line **passdb pam {** and the subsequent closing brace and uncomment **#passdb passwd {** and its closing brace, Vector does not use PAM authentication. Then move the startup script to the location used by Vector with

```
mv /etc/rc.d/rc.dovecot /etc/rc.d/init.d/dovecot
```

and tell *VasmCC* to start the service. If you want to use POP3 to collect mail in addition to the default IMAP, you need to change

```
#protocols = imap imaps
```

to

```
protocols = imap imaps pop3 pop3s
```

and set

```
pop3_uidl_format = %v.%u
```

Now you can set your mail clients to point to your Vector Linux box, using the username and password you have set on there. By using IMAP rather than POP3 to connect, your mail stays on the local server, so it will be the same whichever computer you downloaded it from, so there will be no more trying to find an email on your laptop when you downloaded it to your desktop, or having to remember to synchronise the mail directories of the two machines before you switch from one to the other.

› ***Dspam*'s web interface even draws you graphs to show how much junk you have been sent.**

De-spam with dspam

Being able to download and serve your mail locally is great, but far too much of it is spam. Wouldn't it be great if you could get rid of the spam before you even fired up your email program? When we set up *fetchmail* to collect your mail, we told it to use *procmail* to deliver the mail. An enhancement is to have it pass the mail to a spam filter, which would remove the rubbish and pass only the good (or not too bad) mails to *procmail*. One of the most popular programs for this is *Spam Assassin*, but I have found *dspam* to be more effective. Both use Bayesian filtering, where they analyse the content of the mails and give them a spamminess rating based on a number of criteria.

The default setup for *dspam* is to move the spam to a quarantine area, so you only download the mail that passes the tests. The quarantine can be viewed with a web browser, from where you can pick out any 'false positives', mails that have been incorrectly tagged as spam and delete the rest. *Dspam* learns from its mistakes, so each time you mark a false positive, you reduce the chances of that error recurring. Similarly, you can send spams that made it through the filters back to *dspam*, teaching it further. I now get very few spams leaking through, and almost no false positives, despite receiving in excess of a thousand spam mails some days.

There do not appear to be any Slackware packages for *dspam*, so download it from **http://dspam.nuclearelephant.com** and install it with

```
tar xf dspam-3.8.0.tar.gz
cd dspam-3.8.0
less README
./configure --sysconfdir=/etc
make
su
make install
```

Now edit **/etc/dspam.conf** and remove the leading **#** signs to uncomment the lines

```
#TrustedDeliveryAgent "/usr/bin/procmail"
#UntrustedDeliveryAgent "/usr/bin/procmail -d %u"
```

and edit your **.fetchmailrc** file(s) to change the **mda** line to

```
mda /usr/local/bin/dspam --deliver=innocent -- -d %T
```

Now *fetchmail* passes the message to *dspam*, which checks for spam and delivers clean mails via *procmail*. Spam mails are kept in quarantine and must be accessed via a web browser. This entails setting up a web server with the appropriate CGI scripts and authentication mechanisms, and also means that users have to log onto the web server to check for any false positives. There is another way, change **--deliver=innocent** above to **--deliver=innocent,spam**, which means that all mails are delivered to your mailbox, spam and non-spam. Before you start shouting "what's the point in that" at your copy of *Linux Format* (which is not a good thing to do when reading it on the train), consider that even though it is delivering all the mails to your mailbox, it is adding headers including one for **X-DSPAM-Result** that contains either **Innocent** or **Spam**. You can use these to filter spams into a separate folder in your mail program, meaning you can check and delete your spams quickly, without having to fire up a browser.

Bayesian spam filters like *dspam* learns from your mails, you can give it a head start by passing it a collection of good mails and another of spam mails (if you haven't deleted them). Select them in your mail program and save them as a individual files, in directories for spam and non-spam (sometimes referred to as 'ham'). Since mail programs often keep mails as separate files, this could be as simple as copying your mail and spam folders.

Then copy these files to the server and train *dspam* with

```
dspam_train username spam_dir nonspam_dir
```

You must give spam and non-spam to **dspam_train** for it to be able to learn anything useful.

A digital insurance policy

Hands up those who remember complaining about the hassle of changing floppy disk to do backups? Now 4GB DVDs are too small and large capacity tape drives cost more than the computer you are backing up, although maybe not as much as the cost of the irreplaceable data you could lose. There is an alternative, and it is sitting right in front of you (or in your loft if you haven't dug out the old box yet). Serving files over a network requires very little CPU power or memory, all you need to build a backup server (or any kind of file server) is a computer with enough hard drive space and a network card.

As hard drives are one of the few components in a computer to work physically as well as electronically, they are subject to wear and tear so a 5-10 year old drive may not be the safest place to backup your treasured files. But new drives are cheap with 160GB drives selling for around £30. Use the existing drive to install the operating system and reserve the new drive purely for storage, that way you keep your system files and data separate. If your system disk is on **/dev/hda**, put the backup drive on hdc or hdd, so the two drives are on different controllers. Your CD-ROM is probably already on **hdc**, so set the drive's jumpers to slave and connect it to the same controller as the CD-ROM.

Partition the drive with *cfdisk* or *gparted*. You will need to install *gparted* for this and it has a lot of dependencies, for this one-off job it is easier to use *cfdisk* with

```
cfdisk /dev/hdd
```

Press n to create a new partition and hit Enter twice to accept the defaults and create a partition spanning the whole disk; enter a different size if you want a smaller partition. Press W (it must be capital W) to write the changes and type "yes" to confirm, then q to exit and reboot to make sure the kernel re-reads the partition table. Now create a filesystem with

```
mke2fs -j /dev/hdd1
```

Make sure you give the right device name or you will trash the wrong partition! Now add it to **/etc/fstab**

```
/dev/hdd1  /mnt/backup  ext2  noatime 0 0
```

and mount it with

```
mkdir /mnt/backup
mount -a
df -h /mnt/backup
```

The last command should show that it is mounted and 99 per cent empty. Now you can export it to the other computers on your network by adding this line to **/etc/exports**

```
/mnt/backup 192.168.1.0/255.255.255.0(no_root_squash,no_
subtree_check,rw,sync)
```

The first three numbers in the IP address there should be the same as everything on your network, the final one is zero. Go into the services section of *VasmCC* and ensure that portmap is set to start on your run level and make **/mnt/backup** available to the rest of the network with

```
exportfs -a
exportfs
```

The first line exports the shared directory, the second lists all shares, so you can see that it worked. Now you can mount the backup directory on each computer on your network by adding this to **/etc/fstab** on each one

```
192.168.1.N:/mnt/backup  /mnt/backup  nfs  nosuid,rsize=8192,ws
ize=8192,soft  0 0
```

Where 192.168.1.N is the IP address of the server. How you actually make the backups is up to you, any program that writes backups to files can be used. It could be a simple as having a directory for each computer in **/mnt/backup** and having a *cron* job that does something like

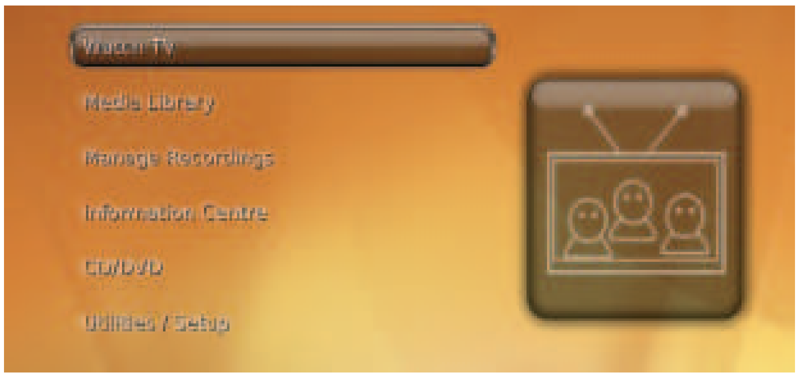
```
tar czf /mnt/backup/$HOST/home-$(date -I)-.tar.gz /home
```

that will tar up the home directory into a file containing the date, leaving you to delete older archives as you see fit. Or you could run something like *BackupPC* (**http://backuppc.sourceforge.net**) that will backup a network of machines to a single server, the choice is yours. Of course, you don't have to use a file server for backups. There are plenty of other used for shared network storage, such as storing your music or video collection. NFS works best with Unix-like operating systems (including Mac OS X). If you want to share files with Windows, you'll be better off with *Samba*.

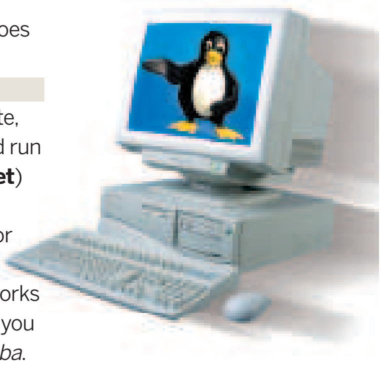
TV Time

It would be possible to use an old computer as a *MythTV* backend, but it would be limited in what it could do, transcoding and commercial detection would take a long time, but recording broadcasts that are already MPEG streams needs only modest horsepower. Add a Freeview card or two and a good sized hard drive and you could use it as a recording backend.

Running a frontend is far less demanding, the only part of this task that really works the hardware is decoding MPEG streams, and that is not a consideration if your video card has a hardware decoder. Such cards are really cheap nowadays, something like a Nvidia FX5200 can be had for less than twenty pounds. Add another ten pounds for a quieter CPU cooler and a USB remote control and you have all you need. Because modern CPUs generate so much more heat, modern coolers are more efficient, so any half-decent cooler can be used, and usually run at a slower speed to reduce noise still further.



› **Add a suitable, but cheap, video card and you can use your old box to watch TV.**



The web at home

You could also use your computer to run a web server. It won't be up to running an online commerce site, but you could use it for testing your site before uploading it, or just for the fun of seeing how a web server works. The *de facto* standard web server is *Apache*, but this is a big beast and there are lighter alternatives to play with, like *boa*, *cherokee*, *lighthttpd* and *thpptd*. We chose to use *boa*, simply because it is available to install through *gslapt*. After installation, edit **/etc/boa/boa.conf** and change **DocumentRoot** to **/home/www** then start *boa* through the Services section of *VasmCC* and you are ready to add content to **/home/www** and access it from any browser on your network.

We've only scratched the surface of what you can do with redundant hardware – use your imagination! Look at some of the specialist distros on DistroWatch (**www.distrowatch.com**) for some ideas. Or you could set up a home automation system using X10 modules, all controlled from a web browser. You could even use a text-to-email gateway to turn the lights and TV on before you get home by sending a text message from your phone! If you can think of, there is probably a way to do it – get in touch to tell us what projects we could feature in future issues. LXF