GET STARTED: AMAZON WEB SERVICES

LINUX FORMAT

Happy birthday to us!

15 YEARS OF LINUX!

How it changed the world

» The rise of Ubuntu
» Raspberry Pi revolution
» Steam gaming arrives

+ **Init Wars:** How Systemd won over SysV and ate Linux

SPECIAL ANNIVERSARY EDITION!

200 BEST EVER LINUX TIPS!

» Desktop speed ups » Terminal shortcuts
» Sysadmin secrets » App tricks and more!




200TH ISSUE!

72 pages of expert Linux reviews, features & tutorials

Open source Heroes

"Oh, it's an iPhone... screw you! I'm not talking to you anymore."

Linus, Stallman, Shuttleworth, Kroah-Hartman... all on p40

Complete guide
Master Debian 8
» Discover the secrets of this powerful distro

Live patching
Inside kernel 4.0
» Take a next-gen Linux build for a test drive now!

4.0

Future

CYBERHOSTPRO
www.cyberhostpro.com

RESELLER HOSTING

START YOUR WEB HOSTING BUSINESS TODAY FROM JUST £9.99 MONTH

- Host Unlimited domain name
- 100% Branded to your business
- Easy to use web based control panel
- Wordpress installer
- Plus much more

WEB HOSTING FROM £29.99 year

- Up to 10GB Space
- Premium Email Accounts
- Host up to 30 websites
- Web Based Control Panel
- Windows 2012 R2 or Cloud Linux 6
- Instant activation

CLOUD VPS SERVERS FROM £15 month

- Build a server to your exact requirements
- Scale up to 10GB Ram
- Fail safe hardware
- One click backup & restore
- Catalogue of Operating systems
- Plesk or CPanel also available
- Instant activation

DEDICATED SERVERS FROM £89.99 month

- Latest Dell Poweredge servers
- Multiple Intel Processors
- Up to 4TB disk space
- 10TB monthly bandwidth
- Windows or Linux OS available
- Plesk or CPanel also available
- 1Gbit port

Hosting Heroes

Our heroes are available to answer queries and provide technical support 24 hours a day, 7 days a week.

No Contracts

All our products are contract free. You can cancel at any time by giving us notice before your next billing due date.

99.9% uptime guarantee

To ensure your website is always online, we provide a 99.9% uptime guarantee.

Use coupon code **LINUXMAG** to receive 30% of todays order.

0845 527 9345

LINUX FORMAT
Get into Linux today!

What we do

» We support the open source community by providing a resource of information, and a forum for debate.

» We help all readers get more from Linux with our tutorials section – we've something for everyone!

» We license all the source code we print in our tutorials section under the GNU GPLv3.

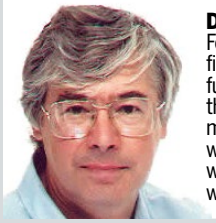
» We give you the most accurate, unbiased and up-to-date information on all things Linux.

Who we are

This issue we asked our experts: What was your Linux highlight over the last 15 years of **LXF** coverage?

Jonni Bidwell
I remember when desktop effects were introduced, and you could enable all manner of crash-inducing, productivity-depleting eye candy through Compiz. Wobbly windows were a favourite, some mornings I think they have come back. But mostly that's just my eyes acting a bit wobbly.

Neil Bothwick
I know I should be thinking of something incredibly geeky and techy to spout on about, but the real high points have all involved meeting people from various parts of the Linux and open source communities at conferences, LUG meetings and similar. The software is great but the wetware is better.



Dr. Chris Brown
For me the turning point came when I figured out how to declare a pointer to a function that returns a pointer to a function that returns an integer. Another interesting moment for me and the Brown household was when I decided to replace Windows with Ubuntu on my wife's laptop. I'm still waiting for her to notice.

Paul Hudson
It took me fifteen long, death-filled years to figure out that *Nethack* doesn't actually have an end. Instead, I have concluded that the Amulet of Yendor is an elaborate internet ruse that is designed to keep me stuck at a Linux terminal for the rest of my life, and nothing you can say will convince me otherwise.

Mayank Sharma
My word that's a wide timeframe, but if I have to pick one highlight it'll have to be the advent of live CDs and more specifically Knoppix. That one distribution has perhaps introduced more people to Linux than all others combined. That and easy to use desktop virtualisation software like *VirtualBox*.

FOSS forever

» *Linux Format* is 200 issues old. That's no mean feat in the current magazine climate of continuous decline, but the recipe for success remains the same as the day it launched: pleasing you, the discerning *Linux Format* reader. We wouldn't have remained the best-selling Linux magazine for 15 years without your continued support. So have a heartfelt thanks for all here at *Linux Format* Towers, especially Effy, you've bought him a lot of tequila over the years...

The success of *Linux Format* is simply a reflection of the success of GNU/Linux itself. While at their time, best-selling magazines such as *PC Answers*, *PC Plus*, *Amiga Format* and many more, have all fallen by the wayside, *Linux Format* marches ever on. In fact the number of Linux and FOSS-based titles on the newsstand has exploded, covering the Raspberry Pi, Ubuntu, Android and more. Despite the competition *Linux Format* has remained firmly the best-selling UK Linux magazine.

So sitting here at this milestone we're looking back over an amazing 15 years of Linux coverage, while looking forward to what the future holds for the Linux kernel and the FOSS community. If you're the nostalgic type jump to our Linux Then and Now feature on page 46 for 15 years of world-conquering Linux coverage. We dig through the archives on page 40 to bring you interviews from our Open Source Heroes.

Reading interviews with giants such as Stallman, Torvalds and Shuttleworth it's clear how their passion and worldview plays into how they've shaped their particular area of the FOSS world. We've even dusted off our own ex-Linux giant, Latin-speaking Paul Hudson on page 88, so he can revive his love for the not-so dead PHP.

If you're more interested in the new, we've a guide to the Linux kernel 4.0 and how you can give live-patching a trial run, all on page 76. For the system admins out there we're looking at the future of Amazon Cloud Services on page 52, alongside how you can get started with Debian 8 and switch to not-so-new IPv6 while we wait for it to take over IPv4. Here's hoping you enjoy *Linux Format* for another 200 issues...

Neil Mohr, Editor
» neil.mohr@futurenet.com

LINUX FORMAT

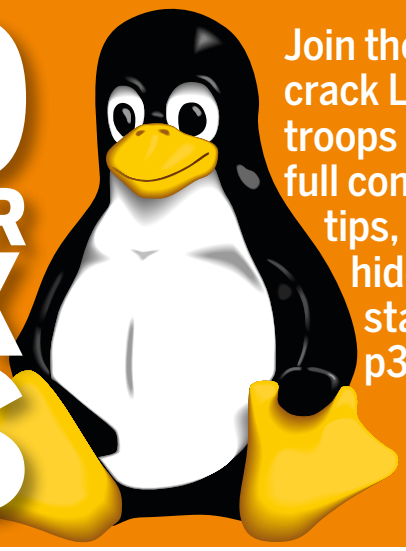
Contents

Go retro! Free issue 1 on DVD

"Value your freedom or you will lose it." – Richard Stallman

Reviews

200 BEST-EVER LINUX TIPS



Join the ranks of crack Linux troops with our full company of tips, tricks and hidden gems starting on p32

Roundup: Alternative OSes p24

Interview

"And then I get really excited, and by excited I mean I curse at people."

Classic interviews with open source giants p40

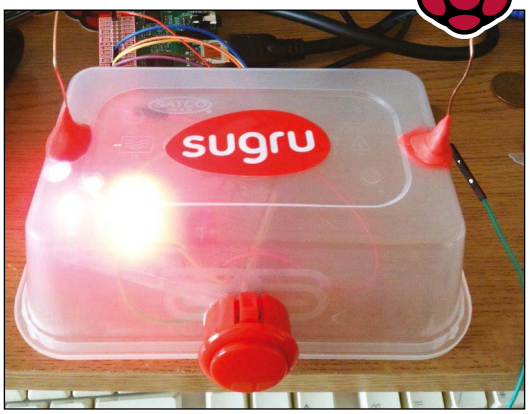
On your FREE DVD

Fedora 22 64-bit, Sabayon 15.06 64-bit, Ubuntu 15.04 32-bit,

» Only the best distros every month.

PLUS: Hotpicks, Roundup and more! p96

Raspberry Pi User



In-depth...

» Join us as we don our nostalgia goggles.

Coding Academy

Tutorials

» Proxmity lock. Mines not included.

Regulars at a glance

» Our subscriptions team is waiting for your call.

LINUX FORMAT
Newsdesk

OPEN SOURCE NEWS

Is SourceForge up to its old ticks?

Accusations emerge that is has now hijacked Nmap's account to spread adware.

Sourceforge is a website that users will recognise as a place where they may have download software from at some point or other, but on a number of occasions in its history it's gained the ire of developers and users alike by peppering pages with fake download buttons. These buttons install different – and on the whole unwanted – software.

Gimp, the open source image editor used to use Sourceforge to distribute its Windows installer, but had a public falling out with the site regarding this practice. To make matters worse, Sourceforge continued to distribute the software using *Gimp*'s account, which included an additional installer that bamboozled users into installing other software. The story appeared to end with Sourceforge promising never to include this extra unwanted software which it rather creatively calls 'offers' **http://bit.ly/SFAdwareResponse**) without the express consent of a developer, except that *Nmap* now says SourceForge has taken over its account.

In a blog post Gordon Lyon, (**http://bit.ly/SFHijacksNmap**) the program's author has alleged that the original download page has been made blank, and a new page hosting the download – along with many other download links that will install unwanted programs – has been created by Sourceforge. Although Lyon has confirmed that as of yet no additional software has been included with the *Nmap* installer, he's understandably annoyed at Sourceforge hijacking his account, and he's wary that it might include Trojan installers in the future. Sourceforge is likely to defend its actions as a legitimate way of offering its users choice, while gaining advertorial money to keep the site going, but for now we would suggest not downloading anything from Sourceforge until we find out more.

Oh dear, is SourceForge being a bit naughty again?

SOFTWARE NEWS

Gimp gets a huge update

The GEGL image processing engine gets a serious revamp.

There's great news users for those who user *Gimp* for all their image-editing needs as the Generic Graphics Library (GEGL) which powers the program has been given a huge update – the first in three years.

The update brings a number of features and improvements, including better thread-safty and experimental multithreading support, which is still in its early stages but could bring performance improvements later on. Support for mipmaps (which loads large image files in the background while you view a smaller preview version) is also included. Though, again, this is in an early experimental phase. When the feature is ready and incorporated in a future version of *Gimp*, it should make working with large image files feel a lot smoother. A new default tile back-end has also been added, which lets *Gimp* write to disk in a separate thread, so the image-editing software will be faster and feel more responsive when saving and exporting image files.

A huge number of additional features and improvements that have been included with GEGL 0.3 is listed over at *Libre Graphics World* (**http://bit.ly/GEGLUpdate**), and it's well worth a read. You won't find the additions on your currently installed version of *Gimp*, but they'll be making their way to future versions, and it gives us a good idea of what we can expect from the popular open source image editor.

MOBILE DEVICES

BQ announces another Ubuntu Phone

The Aquaris E5 HD Ubuntu Edition goes on sale for €200.

After the roaring success of the BQ Aquaris E45, the world's first commercially available Ubuntu Phone that launched earlier this year, it looks like BQ is going to be sticking with Canonical's mobile operating system as it has just announced the Aquaris E5 HD Ubuntu Edition smartphone.

You can buy it right now from **http://store.bq.com/en/ubuntu-edition-e5** and it will set you back €200. While the E45 was a more budget handset aimed at getting developers on board and creating apps for Ubuntu Phone, the Aquaris E5 HD Ubuntu Edition is a more mainstream handset.

This means it's got a nice, large 5-inch display that's 720p high definition, with a quad-core 1.3GHz Cortex A7 processor and 1GB of RAM to keep things ticking over. Canonical has previously told us about its desire to make Ubuntu Phone popular among mainstream and casual smartphone users, and while the previous phone was rather niche, could the Aquaris E5 be the one to break out into the wider market?

We haven't had a chance to have a play with it just yet, but by looking at the specifications, it has the potential to be a decent all-round smartphone. The front camera is 5 megapixels, while the rear is 13MP and comes with autofocus and two flashes, so it could appeal to people who like using their phones to take photos. Although it's not the highest resolution for a smartphone, the 5-inch screen should look pretty sharp at 720x1,280 – and it shouldn't drain the battery too severely either.

› The Aquaris E5 HD Ubuntu Edition looks a lot like the E45, but it has a bigger screen and better specs.

MOBILE DEVICES

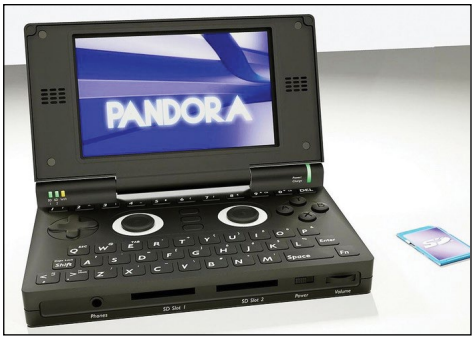
DragonBox Pyra is alive and well

The handheld Linux device overcomes some big hurdles.

The DragonBox Pyra, a handheld device that has its roots with the Open Pandora Linux project, has recovered from a spate of problems, and although we don't have a release date the projects now seems to be in good health.

The Rotation chip, which powers the graphics of the DragonBox Pyra, has now become more-functional, which might not seem like much cause for celebration, but it's an improvement. Work can now begin on implementing anti-tearing and other graphical effects to smoothly power the LCD screen of the DragonBox Pyra with some attractive emulator graphics.

As this will be a portable handheld, the case of the device is incredibly important, and the good news is that the prototype has almost been finalised, which will allow the LCD screen to fit comfortably, along with a built-in battery to help power the device. The KeyMat – how you will control the device – is also apparently coming along nicely, but although there has been some recent progress with the project, it's looking unlikely that it will make its Q2 2015 release date, with a late 2015 or early 2016 release being a more realistic target. If you'd like to find out more about the project, and see how you can help fund and contribute to it, check out the project's website at **www.pyra-handheld.com**.



› Don't give up on the DragonBox Pyra – yet.

Newsbytes

If you've been interested in trying Mozilla's new mobile operating system Firefox OS then you no longer need to wait for a new smartphone. The b2gdroid project (**https://people.mozilla.org/~fdesre/b2gdroid**) is an experiment to get the user interface of Firefox OS, known as Gaia, on to Android devices. That means you'll still be running Android underneath, but you'll get an idea of how Firefox OS works.

After 17 years we're sad to hear that the French company Mandriva is being liquidated. Although Mandriva had a respectable history of getting people and organisations to move to Linux from Windows, such as its deal back in 2007 with the Nigerian government to bring Linux to 17,000 computers used by schoolchildren, but has also been plagued with financial problems along the way.

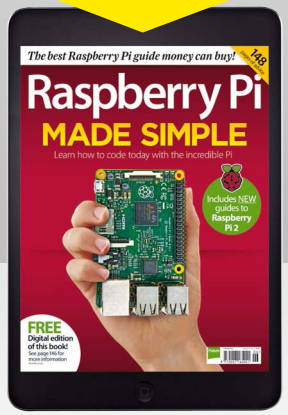
› RIP Mandriva.

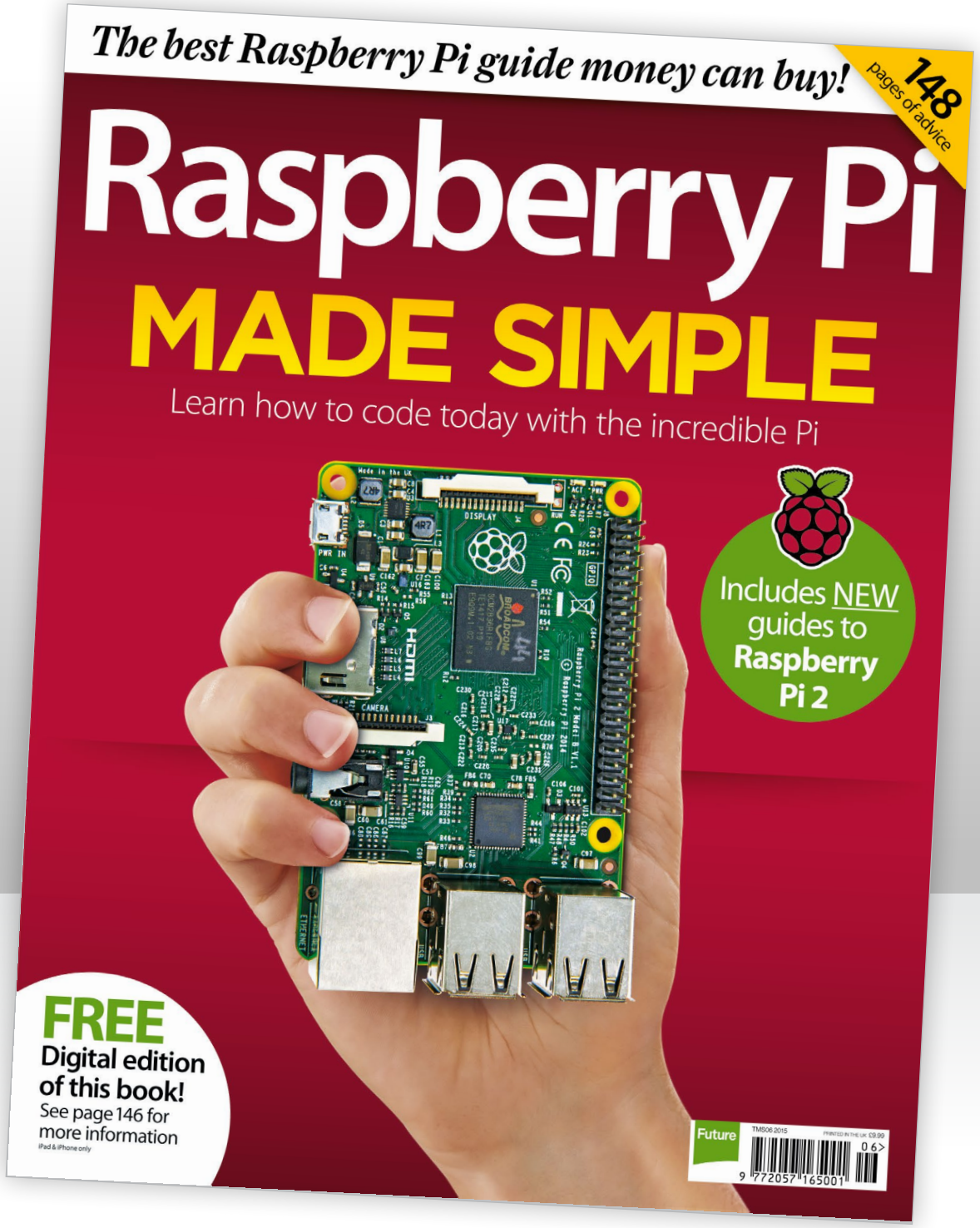
Roboto, the family of fonts which Google uses as default on Android and ChromeOS has now been made open source and can be downloaded from GitHub at **https://github.com/google/roboto**. The GitHub page contains download and installation instructions, so if you're working on a project and want to give it a Material Design feel, you now can.

Life without Linus? What will happen when the creator of the kernel finally decides to retire? In a candid interview with *Bloomberg* – which offers quite a lot of insight into his aggressive driving style – Linus Torvalds talked about the future of Linux without him: "The technical know-how these days is less," Torvalds says. "It's more about being trusted and being available. Greg [Kroah-Hartman, *see p44*] is the obvious number two. He could take it up, and then there are a couple of other people." Catch the whole interview at **http://bit.ly/LifeWithoutLinus**.

GET THE MOST FROM YOUR RASPBERRY Pi

OUT NOW!
WITH FREE DIGITAL EDITION





DELIVERED DIRECT TO YOUR DOOR

Order online at **www.myfavouritemagazines.co.uk**
or find us in your nearest supermarket, newsagent or bookstore!

pcspecialist

PC Specialist recommends Windows 8.1

Get a PC that's more **you**

Rutilus

- AMD Athlon X4 860K
- ASUS® A88XM-PLUS
- 8GB HyperX FURY RAM
- AMD Radeon R9 270X
- 120GB Samsung EVO 850 SSD
- 1TB Hard Drive
- **Windows 8.1**
- 3 Year Standard Warranty

THIS SPEC FROM **£549***

Vanquish HAF III

- Intel® Core™ i5-4590
- ASUS® H81-Gamer
- 8GB HyperX FURY RAM
- 2GB NVIDIA® GeForce® GTX 960
- 120GB Kingston SSD
- 1TB Hard Drive
- **Windows 8.1**
- 3 Year Standard Warranty

THIS SPEC FROM **£749***

Minerva

- Intel® Core™ i5-4690K
- ASUS® RoG MAXIMUS VII RANGER
- 16GB HyperX FURY RAM
- 4GB NVIDIA® GeForce® GTX 970
- 120GB Kingston HyperX 3K SSD
- 2TB Hard Drive
- **Windows 8.1**
- 3 Years Warranty

THIS SPEC FROM **£999***

Ultimate Ti

- Intel® Core™ i7-5930K
- ASUS® Rampage V Extreme
- 32GB HyperX PREDATOR RAM
- 6GB NVIDIA® GeForce™ GTX980Ti
- 6GB NVIDIA® GeForce™ GTX980Ti
- 240GB HyperX SAVAGE SSD
- 1TB Western Digital Black Hard Drive
- **Windows 8.1**
- 3 Years Warranty

THIS SPEC FROM **£2,999***

Optimus Nebula

- Intel® Core™ i7-4720HQ
- 17.3" Matte Full HD LED
- 8GB HyperX IMPACT RAM
- 2GB NVIDIA® GeForce® GTX 960M
- 120GB HyperX 3K SSD
- 1TB Hard Drive
- **Windows 8.1**
- 3 Year Standard Warranty

THIS SPEC FROM **£849***

Defiance 17

- Intel® Core™ i7-4720HQ
- 17.3" Matte Full HD LED Screen
- 16GB HyperX IMPACT RAM
- 4GB NVIDIA® GeForce® GTX 980M
- 240GB Kingston SSD
- 1TB Hard Drive
- **Windows 8.1**
- 3 Year Standard Warranty

THIS SPEC FROM **£1,349***

USE DISCOUNT CODE **PCF02** FOR £15 OFF YOUR ORDER

GAMING DESKTOPS & LAPTOPS

www.pcspecialist.co.uk • 0333 011 7000

Mailserver

Write to us at *Linux Format*, Future Publishing, Quay House, The Ambury, Bath BA1 1UA or **lxf.letters@futurenet.com**

Winlinux

First I'd like to say how much I enjoyed the issue of *Linux Answers*, I'm still picking it up for a re-read! Not many PC magazines can claim the same attention. I found the new *Linux Format* very well put together and nicely balanced for new and more experienced users.

Now I'm not a GNU guru, but I can honestly say I've installed numerous distros of Linux over the last couple of years and Mandrake was the least painful, but Winlinux 2000 was really the easiest to install. You mention in the article that there were problems with the installer. I was under the impression that this was just a problem on the original version and not the current one. I myself had no problems installing – I only had to force settings for my mouse and keyboard, but everything else worked like a dream.

I create a dual-boot menu, to avoid having to start from Windows or a DOS prompt, and now have the perfect OS marriage. I admit there's a slight performance hit, but it's not too noticeable. I only really use Linux for the Internet, design and layouts, although I have become addicted to playing a game called *The Black Penguin* of late!

With Winlinux 2000, new users don't have to worry about FIPS and/or dead possible errors. Winlinux 2000 goes onto the system cleanly and comes off again exactly the same.

I don't know if it's my imagination but I get a much better and improved Internet connection through Winlinux 2000 compared to Windows 95, which is strange.

I think 4/10 was a little unfair and that Corel Linux should've been closer to that mark instead of the 6/10. Corel Linux has been my only failure to date.

Andy Goz, via email. (issue 2)

› **Be our guest and try and argue the Atari ST was a better computer.**

Who's the fairest?

I would really like to see a triple-header in-depth review of three of the most beautiful, elegant, and downright enjoyable and usable, Linux desktop distros out there: Evolve OS, Ozon OS and elementary OS.

All of them really shine, and despite being betas they definitely deserve some cover-page love right now! They all show, in different ways, what craftsmanship and TLC can do for the Linux desktop and should be celebrated as real contender distros for 2015. Disclosure: I already use elementary OS day-to-day on my own not so powerful desktop, as well as on my wife's new home-office desktop, and even on the new desktop of my slightly 'techno-challenged' in-laws; for all of them elementary OS is going down an absolute storm and is rock-solid. Please, please, give these three distros the cover-page attention they deserve!

Diccon Spain, via email

Neil says: There's sometimes a rather perverse dismissal of any distro that dares put an emphasis on elegance and aesthetics in the Linux community, and some people do seem to take pride in using the ugliest, most user unfriendly interfaces in a bid to show just how hardcore they are. Of course, they do have a point in some regards – for many of us using the terminal and command line allows us to complete tasks far more quickly than any graphical user interface would. Also many people equate fancy themes and desktop wallpapers with bloat – why spend time prettying up a distro when it would be better to make something fast and stable?

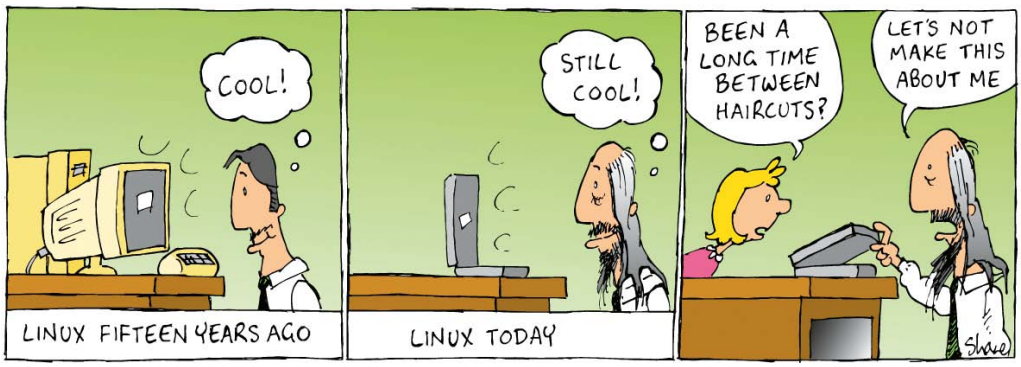
However, as the distros you've mentioned demonstrate, you don't have to sacrifice usability or stability for a great looking OS. We spend so much of our time staring at the screen, there's no harm in wanting a good-looking interface, and if these attractive UIs convince people to move to Linux that's even better! You can be sure we'll be keeping an eye on the distros you've mentioned and will feature them in the future.

Amiga Support

I just got my first copy of *Linux Format* as a replacement for my *Amiga Format* (sniff) [*Amiga*

Helpdex

shane_collinge@yahoo.com



Computing was better! – Ed]. It is very nice to see the words PPC Amiga in the magazine, and I hope the Amiga will continue to be mentioned in *Linux Format*.

The mag looks great and it was nice to see a round-up of all the different Linux packages. I myself have Corel Linux (kernel 2.2.14) and Red Hat for the PC, and now I also have Definite Linux 7.

However, these are for the PC so I wonder if you would also do a round-up for the Amiga and Mac? I haven't seen any round-ups of Linux for the Amiga and I'm not planning on buying every single copy out there. I don't want to spend days configuring, just to get it running like you mentioned. That's why I got a copy of Corel Linux for the PC. Just a few simple questions and the whole thing gets installed and works correctly.

Remco Komduur, The Netherlands (issue 2)

Neil says: No promises!

Money up front

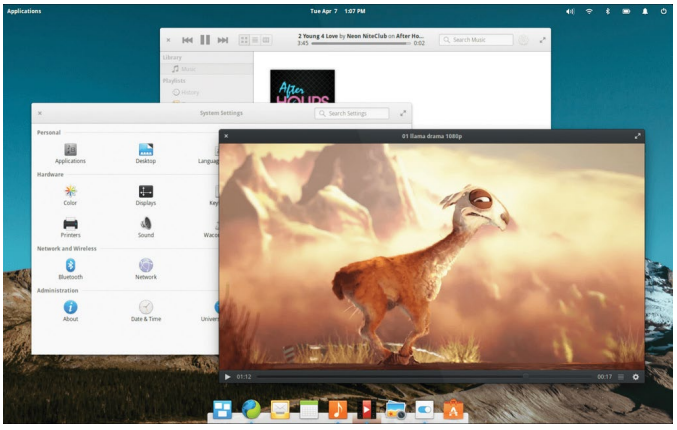
On the elementary OS money request, I seldom, if ever, donate at time of download. That kind of defeats the 'try before you buy' element encouraged by most Linux applications and distros.

I may download (or try from LXFDVD) many options. If I had to pay for each, my testing would certainly be reduced. I would 'read more' and try less.

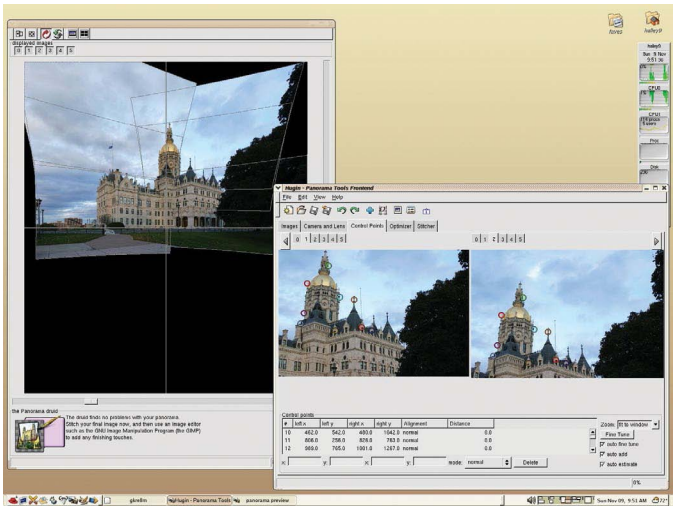
By the way, I don't mind the Android method of enabling more features with a fee. As long as the trial version is so limited, you don't know what it can do. I do find 'timers' annoying.

Rob Giuliano, Saline, MI USA

Neil Even a great looking interface can't distract from some worrying changes and elementary OS' new method of asking for funds – by placing the Download button behind a donation box that has $10 as default has certainly divided opinion. You're completely right that when trying distros you should be able to test it out first, then consider donating if you like what you see. However, many distros do find fund raising difficult – and without funds those distros would likely cease to exist. According to elementary, 99.875 percent of downloaders don't donate. It's not an ideal situation, but you can still download and install it for free – you just need to type 0 into the text box.



› Distros like elementary OS can be both attractive and powerful.



› Hugin [*see tutorials*, LXF184] is a great app for creating panoramas, but just missed out on our top 100. [*see cover feature*, LXF196]

102 best Linux apps

What a good line up of useful applications in Top 100 Linux Tools in LXF196 [*see p36*]. The competition to get into the 'top 100' would be pretty severe, so I can sympathise with the task you must have had in whittling the list down to 100.

No doubt there will be a lot of bleating and moaning about favourite apps that didn't make the cut, and just to be sure about that – here is my two bob's worth!

***Darktable* and *Hugin*... two very nice, and well-developed applications for RAW image development and post processing, and panorama stitching (and much more), respectively. Keep up the good work.**

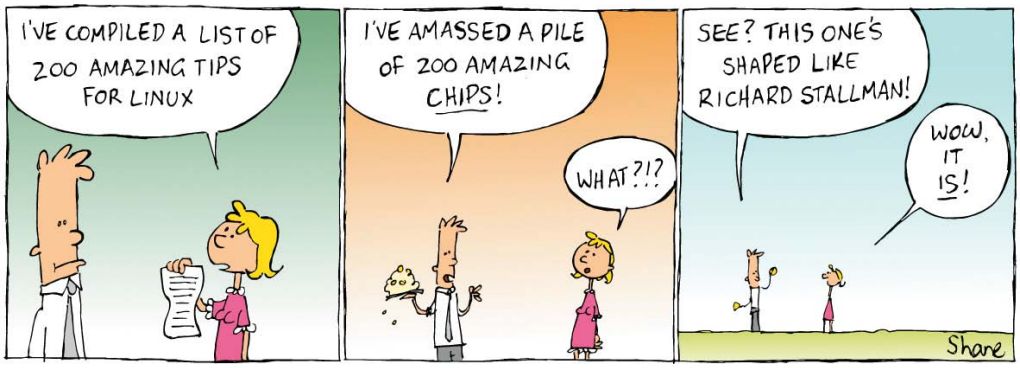
Terry Duell, via email

Neil We're glad you enjoyed the feature Terry! As you rightly pointed out whittling the list down to the essential 100 is no mean feat – we really do have an abundance of excellent free and open source software to choose from. Not a bad situation to be in really! Thanks for your suggestions as well, I particularly like *Hugin* for creating sweeping panoramas from my disparate snaps as I wonder our green and pleasant land.

Embedded Linux

Let me first warn you that I'm an ADDICT. YES!!! I'm addicted to your AWESOME, FABULOUS... Errrrr... YaDiDaDiDah... Actually NO WORDS can completely describe your magazine. Yes... I am an avid fan of your magazine here in Sydney, Australia. Ssshhhhh, just don't tell my boss because I work in a company with a completely Microsoft environment. However at home, it's a friendly and stress free Linux environment... Beat that Microsoft!!!

Anyway, the purpose of my email is to ask you guys if you could possibly look for a Linux thin client using Raspberry Pi or equivalent without a full load distro installed. All I want is (most likely to many others too) to boot directly to the logon screen and can toggle to connect to RDP, VNC etc or utilise a dedicated bootup server where client can just PXE boot »



» straight to logon screen. Can this be possible?

Emman Espinas

Neil Thanks for your kind (and enthusiastic) words , Emman! As with most things Linux-based, having a thin Raspberry Pi client is not only possible, but thanks to the hard work of dedicated coders, you're actually a bit spoiled for choice. Your first port of call should be the Raspberry Pi Thin Client project (**http://rpitc.blogspot.co.uk**), which is doing some cool things that should suit your needs. *BerryTerminal* (**www.berryterminal.com**) is also worth checking out, as its aim is to create a minimal Linux distribution that turns the Raspberry Pi into a thin client. There are plenty of other projects out there, but these should suit your needs.

› **The Raspberry Pi Thin Client project aims to make a thin yet functional client for the diminutive computer.**

Modern Linux

About six weeks ago I was browsing in a newsagent when I discovered and purchased the Build Your Own PC issue of the magazine [LXF199]. Mistake number one.

As grinding sounds were emanating from my beastie [Ed – Not sure what that means, but lets carry on] and after reading this magazine I was tempted to try building my own PC. Ding. Mistake number two.

After some, but not enough, research online I decided the best and cheapest way was to buy a pre-built gaming computer. That was mistake number three.

Seven days later a brand new and fully tested computer was sitting on my desk with no pre-installed software, because I was going to install the Arch Linux distro on it. And, yes, that was mistake number four.

After following the instructions on the Arch Linux website, I prepared a USB flash drive for the latest Arch install. I tried and was unable to boot the computer in the UEFI mode. Searching the Arch wiki I discovered a warning on the Gummiboot page "...In case you have issues booting EFISTUB kernels you should use a boot loader which does not use EFISTUB..." Researching the internet I discovered that the Linux Mint distro was bootable with the new motherboard.

Downloading and booting into the Linux Mint distro was a success. Mistake number four.

'Why?' You may ask, because I may as well have booted into MS Windows 8.1. Linux Mint being a Ubuntu derivative is a 'what you see is what you get' system with a very pretty graphical user interface and very limited system admin facilities. It doesn't even allow you to login as the root superuser.

When I first discovered the power of Linux twenty years ago I marvelled at being set free from closed propriety operating systems and software applications. Today some Linux distros appear to have lost their way. By chasing new users the distros have joined the mainstream propriety systems in shackling the users to their software systems rather than allowing them the freedom to walk their own path.

John Briggs, via email.

Neil Mistake number six: you have two mistakes number fours. Building your own PC isn't for everyone, but we're sorry to hear our Build Your Own PC edition didn't help. As for your comment on modern Linux distros, you're right that some contain similar elements of other OSes, but they can help people who are new to Linux get used to the OS. They might not be for seasoned pros for yourself, but any distro that helps convince more people to move away from Windows or Mac OSX is a good thing in our books.

You lament some distros chasing new users, but if they didn't then Linux would still be a niche OS stuck in the '90s. Some people need their hands held a little before they walk their own path, and for others there are still some excellent distros that won't patronise or limit you. LXF

Write to us

Do you have a burning Linux-related issue you want to discuss? Want to let us know how much time you've wasted on UEFI or maybe you just want to suggest future content? Write to us at *Linux Format*, Future Publishing, Quay House, The Ambury, Bath, BA1 1UA or **lxf.letters@futurenet.com**

Letter of the month

Cold call scams

Recently I had a cold call at home, where the very helpful individual at the other end explained that he was calling me because "Your computer has got a virus" and he was ringing to stop me having further problems, including potentially being charged with criminal offences... I think we all know how it goes.

I would normally give them short shrift and get them off the phone, but I had a bit of time on my hands and was feeling mischievous. I think anyone reading *Linux Format* could imagine the fun to be had and I kept them on the phone for 20 minutes before they realised I did know about computers and was not the gullible fool I pretended to be.

It was only later that it occurred to me that while these villains were on the phone to me, they were not ringing someone who may have been taken in by the patter and potentially taken advantage of.

Could I suggest that anyone reading this, who has the time and inclination, to warm to the task and see how much of their time and money you can waste and maybe save someone less able.

Maybe the *Linux Format* could contact other computer magazines and give the whole idea some impetus? Like a campaign?

Harry Sales, Stoke on Trent

Neil says:: Good on you Harry for being a nuisance to nuisance callers! It's always nice to toy with scammers trying to get your money (or worse), and you're right; while they were on the phone to you, they weren't tricking someone who might not have been so aware of what they are doing. Luckily for most of us, when these scam artists realise we're running Linux they usually know the jig is up! Perhaps more of our readers have stories of stringing these scammers along?

United Linux!

The intrepid **Les Pounder** brings you the latest community and LUG news.

Find and join a LUG

» **Bristol and Bath LUG** Meet on the 4th Saturday of each month at the Knights Templar (near Temple Meads Station) at 12.30pm until 4pm.
www.bristol.lug.org.uk

» **Blackpool Makerspace** Meet every Saturday, 10am to 2pm. Crossways Hotel, Tyldelsey Road, Blackpool, FY1 5DF
http://blackpool.lug.org.uk

» **Edinburgh LUG** Meet on the first Thursday of the month at the Southsider pub, West Richmond St.
www.edlug.org.uk

» **Hull LUG** Meet at 8pm in Hartleys Bar, Newland Ave, 1st Tuesday every month.
http://hulllug.org

» **Lincoln LUG** Meet on the third Wednesday of the month at 7pm, Lincoln Bowl, Washingborough Road, Lincoln, LN4 1EF.
www.lincoln.lug.org.uk

» **Liverpool LUG** Meet on the first Wednesday of the month from 7pm onwards at DoES Liverpool on Hanover Street, Liverpool.
http://liv.lug.org.uk/wiki

» **Manchester Hackspace** Open night every Wednesday at their space at 42 Edge St, in the Northern Quarter of Manchester.
http://hacman.org.uk

» **Surrey & Hampshire Hackspace** Meet weekly each Thursday from 6.30pm at Games Galaxy in Farnborough.
www.sh-hackspace.org.uk

OpenTech 2015

Mingling with FLOSS friends in London.

OpenTech 2015 in London was a day of talks covering our digital lives and liberties. This low cost event in June provided a glimpse into open source technologies to come. Organised by the FLOSS UK team, the event sees Python hackers rub shoulders with open data enthusiasts and artists and it was exciting to see projects being formed in the corridors.

Admittedly, this is all part of conferences such as this, but what makes OpenTech different is the mix of politics, social justice and open source, and it's refreshing to see these different groups working together.

Highlights this year included the Open Rights Group celebrating 10 years and discussing how the group has been shaped by events in its history, eg the London bombings in 2005. The group originally started with 1,000 members who pledged £5 each to found the group. We were particularly pleased to see a great talk by Rhys Phillips who is working with schools to promote engineering to girls by enhancing the topic with projects that focus on lightning. Other talks included the retention of medical data – how it's used and how problems faced in the development process can be mitigated by creating a liaison between the customer and the development team.

Each OpenTech is different and this is a clear indication of the diverse nature that open source has brought to the community. This is now our third OpenTech and it was wonderful to meet old friends and make some new ones, we're sure that we will be back for future OpenTech events. Well done to the organising team for another great and packed event. LXF

› OpenTech had a diverse mix of talks that reveal the richness of the open source community.

Community events news

PyConUK 2015
After the mention last month, the PyConUK team got in touch to announce that the ever-popular education track will return this year; bigger and better than previous conferences.

On the Friday, teachers from around the UK can take part in a day of Python training courtesy of the Python developers and members of the Raspberry Pi Foundation's Education team. Tickets for teachers are subsidised and are on sale for £50. There's also an allowance available to help pay for cover. PyConUK takes places in Coventry between 18-21 September and for more details head to the website.
www.pyconuk.org

Pi Wars
Pi Wars returns for 2015 with lots more robotic combat and challenges. The event pits the best robots in their class against each other in a series of challenges designed to test much more than their construction. Programming skills as well as build quality will be challenged in multiple arenas. This event is set for December and for tickets along with more details head to their website.
http://piwars.org

Manchester Raspberry Jam
Has it been three years already? Launched in 2012 by Ben Nuttall, Manchester Raspberry Jam was one of the first Jams in the world. Ben has since left Manchester to work in Cambridge for the Foundation but Jack Kelly continues the good work. A mix of talks, workshops and hands on hacking, the Jam has plenty for everyone. It's also child and newbie-friendly with many hackers eager to help.
http://mcrraspjam.org.uk

Fast & flexible Cloud Servers

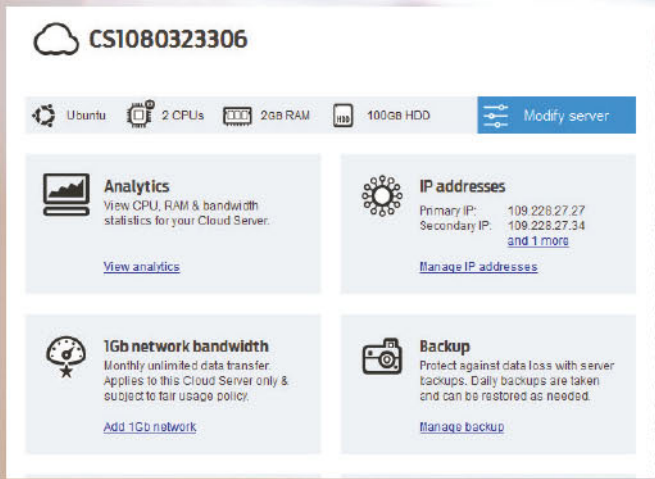
Richard Cullen, Managing Director at bluebox
Fasthosts customer since 2002

Cloud Servers – ultimate speed and resilience

Fasthosts' UK-based Cloud Servers are fast – really fast – and with 3-way mirroring, they give you the resilience you need to stay in control.

The platform we've created with Dell and Microsoft already has around 1,500 users. It's the first of its kind, using the latest Microsoft Cloud Platform System (CPS) which combines a proven software stack of Windows Server 2012 R2, System Center 2012 R2, Storage Spaces & Dell hardware.

Whatever application you're building, and however big it gets, 24/7 UK support and competitive pay-per month pricing make Fasthosts your provider of choice.



The Fasthosts control panel completes the picture, with easy analytics and server management.

Call **0333 0142 709**
or visit **fasthosts.co.uk/cloud-servers**

SERVERS · WEB HOSTING · DOMAIN NAMES · EXCHANGE EMAIL

Cloud Servers

Custom-build an advantage for your business:

Ultimate speed

Microsoft Storage Spaces and enterprise grade SSDs give you the highest throughput and lowest possible latency – for the ultimate user experience.

99.99% uptime guarantee

3-way mirroring gives extreme resilience. Create your own disaster recovery, with server backups automated for you.

Latest OS

Change configuration whenever you need: 1-24GB RAM (+ 33% bursts), up to 16vCPUs...**and** optional installed MS SQL Server on Windows 2012 R2.

Expert UK support

Phone, text, email and chat support from technical specialists – all the help you need to resolve issues and implement better projects.

Microsoft Partner
Gold Hosting

Cloud Servers from:

£11.99
per month

ex VAT charged at 20%.
12 month minimum term contract.

fasthosts


T3
HELPING YOU MAKE THE MOST OF LIFE IN TODAY'S CONNECTED WORLD
ONLINE • PRINT • TABLET
AUDIO TECHNICA ATH-MSR7
Escape to a world of high resolution audio
SAMSUNG JS9000
Relax with the latest home entertainment
CARL ZEISS VR
Embrace a new era of virtual reality gaming
SONY XPERIA Z3
Putting you in control of your life and home
APPLE WATCH
Your health and fitness upgraded
LIFE'S BETTER WITH T3
t3.com

All the latest software and hardware reviewed and rated by our experts

Acer Chromebook 15 C910 54M1

It's the biggest Chromebook **Juan Martinez** has ever seen, but is size everything?

› **Not as portable as some Chromebooks but more powerful.**

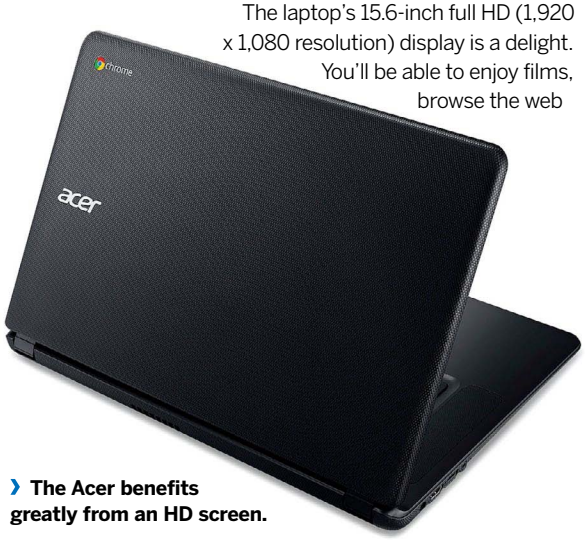
In brief...

- **CPU:** Dual-core 2.2GHz Intel Core i5-5200U
- **GPU:** Intel HD Graphics 5500
- **RAM:** 4GB, DDR3L SDRAM
- **Screen:** 15.6-inch full HD (1,920x1,080)
- **HDD:** 32GB SSD
- **Ports:** HDMI, 1x USB 2.0, 1x USB 3.0
- **Comms:** Wi-Fi 802.11a/b/g/n/ac
- **Camera:** 720p HD
- **Size:** 1.0 x 15.1 x 10.1 inches, 2.19kg

Compared to most other ChromeOS-powered portables, the Acer Chromebook 15 (C910 54M1) has a bigger screen, bigger processing power and comes with a bigger price tag. Specifically geared toward students and teachers – thanks to its rugged design and gorgeous visuals – the Acer is suitable for any consumer who doesn't mind lugging around a few extra pounds and inches.

The first thing you'll notice about this model is how big it is. Tipping the scales at a whopping 2.19kg. This is a substantial notebook that's not ideal for people who will be running from place to place. It is, however, perfect for students and teachers who will be sitting at desks for long periods of time. Housed in a gorgeous black, fabric-textured chassis with a diamond matte finish, the C910 immediately improves the style of any desk space.

The laptop's 15.6-inch full HD (1,920 x 1,080 resolution) display is a delight. You'll be able to enjoy films, browse the web and play games for long stretches without feeling much eye strain. Those who enjoy watching movies with others will appreciate the screen's wide viewing angles, which enables you to seat three people five feet away without noticing any shadowing along the edges.



› **The Acer benefits greatly from an HD screen.**

Acer dedicated about three inches of the keyboard deck's 15.1 inches of width to the laptop's adequate but not marvellous top-facing speakers. The Touchpad, which is somewhat hollow and creaky, takes up three inches, and the keyboard takes up a little more than five inches. This makes navigation somewhat uncomfortable, especially for people with tiny fingers and little experience typing (ie children). Acer should have expanded the keyboard and keys, reduced the width of the speakers, and cut the touchpad in half.

CPU clout

Acer made waves when it first installed Broadwell processors as they are faster and more energy efficient than the Celeron processors you'll typically find on Chromebooks. This addition essentially elevated the Chromebook class from a group of notebooks only suited for light use to devices that could be used to get a little work done.

When compared to the Pixel 2, the C910 narrowly edges out the more expensive notebook in performance terms. In the Octane benchmark, which measures a JavaScript engine's performance, the Pixel 2 received a 24,564 while the C910 hit 25,240. For those interested in benchmarks it achieved 1,301.5 in Mozilla Kraken and 192.5 in SunSpider

This bad boy was also able to crank out a whopping 8 hours and 48 minutes of continuous video playback with the volume and screen brightness set at 50%. That's not quite the 10 hours of performance promised by Acer, but it's close. When compared to the Pixel 2, the C910 slightly loses its edge. We were able to push that to 8 hours and 22 minutes in a more extreme test. LXF

LINUX FORMAT Verdict

Acer Chromebook 15 C910

Developer: Acer
Web: www.acer.co.uk
Price: £379

Features	7/10
Performance	10/10
Ease of use	9/10
Value	8/10

» *The Acer Chromebook 15 C910 is a durable workhorse that will reward you for focusing on its performance not size.*

Rating 9/10

Ultimaker 2

Alastair Jennings investigates quiet, high-resolution 3D printing at home – he's been duplicating an army of himself.

In brief...

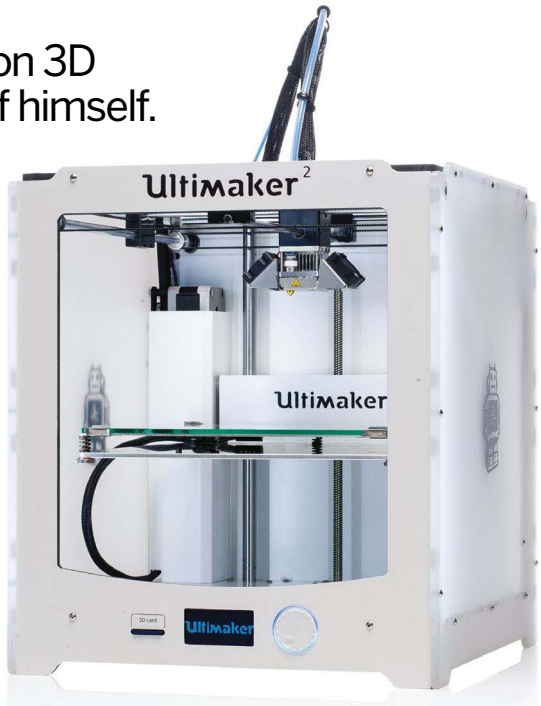
» An evolution of the original that offers a much quieter printing and excellent results. Also comes with the well-designed and intuitive Cura software for loading models. See also: RepRap, Lulzbot Mini.

The Ultimaker 2 is based on open source hardware and software, but is very much a finished product, and is only available in its assembled form. There are three versions: Ultimaker 2 Go, Ultimaker 2 and Ultimaker 2 Extended. Gone are the laser-cut wooden sides and control panel that gave the Ultimaker its distinctive built-in-a-shed look, and instead the new materials used for the casing are more refined and heavyweight. The Ultimaker 2 is an evolution of the original design and although it reflects a similar look to its predecessor, every aspect is a level up in terms of quality.

Print area is an important factor, and the Ultimaker 2 offers an impressive 23x22.5x20.5cm from its heated build plate. The top and front of the printer are open which does mean that you need to be a little careful with fingers during the print process, so if used in an educational environment this does need to be taken into consideration.

3D models can be loaded by a supplied SD card via a slot in the front, and the model to print is selected using the LCD and dial on the front. The models themselves are loaded onto the card along with quality settings via the *Ultimaker Cura* software. On the back is the filament holder, power switch and a USB port that's used for updating firmware rather than tethered printing.

Lifting the Ultimaker out of the box reveals that aside from attaching the filament holder on the back and removing the packaging, the printer is pretty much set to go. The holder simply clicks into place, and then using the control panel on the front the material is selected and filament is fed into the feeder at the back – as this happens instructions on the LCD tell you exactly what to do and after a couple of minutes the filament is loaded and ready to go. Then it's just a case of preparing the build plate with stick glue, slotting in the SD card, clicking print, selecting one of the pre-loaded models, and the print process starts. Our test sample didn't require any adjustment and initial prints turned out well and at an exceptionally high quality.



› **A complete 3D printing package for home, work or education.**

Appealing printer

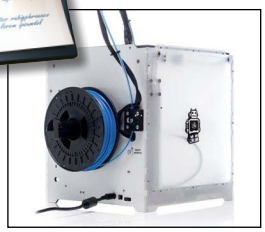
When you're ready to print models on the Ultimaker 2, you first need to load your model into the *Cura* software. Ultimaker is at present the official maintainer of the application but it's also used by other manufacturers such as Lulzbot, and is exceptionally well-designed and intuitive. It enables the adjustment of size, rotation and scale which are all shown on the virtual build plate, plus quality and print settings can all be adjusted through this software.

The Ultimaker 2 is surprisingly quiet and although it's by no means silent you can leave it running without disturbing the rest of the office, house or neighbours. The volume of operation is a major consideration especially if you have to work in close proximity to the printer as the noise does become a distraction. Again, here the SD card loading of files means that you can happily leave the printer in another room, which is a huge advantage over printers that need to be tethered to a computer in order to print.

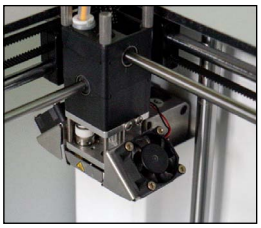
Our test prints ranged from one hour at the Fast Print setting up to 10 hours at the Ulti setting to print out a gear box – these print times are good and compare well against other printers such as the Lulzbot Mini. Print quality at the highest setting of 20 microns is excellent and although you can see the layering, a bit of a clean-up and the models quickly transform. The Normal setting also produce good quality prints, but the Low setting is best used only for very simple objects or roughly checking complex models.

The Ultimaker 2 takes 3D printing to the next level. Design, features and software give you a complete package. If you're looking for a printer at home then the small footprint and large build plate are really appealing, and for work and education the quiet operation is very welcome. LXF

Features at a glance



Filament holder
The filament is stashed around the back on the standard roll for continuous feeding.



Quiet operation
A huge improvement for the model is the quiet operation thanks to its smooth running.

LINUX FORMAT Verdict

Ultimaker 2

Developer: Ultimaker
Web: https://ultimaker.com
Price: £1,749

Features	9/10
Performance	8/10
Ease of use	9/10
Value	6/10

» *Easy to set up, and offers quality prints, ABS and PLA support as well as standalone and quiet operation.*

Rating **8/10**

Fedora 22

Shashank Sharma tests the latest release of the popular Gnome distro and discovers that it's ready to shed its image of being for advanced users only.

In brief...

» The latest edition of the Red Hat-sponsored but community driven distro. Available in three flavours, the regular desktop release is called Workstation. The distro continues to incorporate changes to attract less experienced users and yet please advanced ones. See also: OpenSUSE and Korora.

For its last release, Fedora has abandoned its long established six-month release cycle to bring about a number of organisational and systemic changes. The chief among these was the release of three distinct editions: Workstation, Cloud and Server. The other major change was the announcement of Fedora.next, which is the name for the Fedora Project's roadmap for the next decade.

With the latest release, the distro returns to a six-month release cycle but continues to produce the three editions. Each edition builds on a common base and all the necessary packages are then added to make the edition suitable for the assigned function.

We've chosen to review the Workstation edition, which is available as an ISO for 32- and 64-bit machines. This live installable edition is designed for home users, but is perfectly suited for professionals and can quickly be turned into a developer's haven.

Cutting edge

As a test bed for Red Hat Enterprise Linux (RHEL), Fedora is uniquely positioned to deliver cutting edge technologies to new users. It's always been a distros that ships with the latest software and this release is no different.

The *Gnome Display Manager* (GDM) now defaults to *Wayland* display server instead of *X.org*. While the default Gnome session still uses X, the next Fedora release will most likely default to Wayland.

» **Fedora 22 builds upon the solid foundation laid by the previous release and ships with some new features and useful improvements.**

The distro features Gnome 3.16 which has a large number of visual improvements and new features. The notification area has been revamped and notifications now appear anchored to the centre of the top bar. An unobtrusive marker informs you of any unread notifications, such as background terminal jobs, chat messages and updates etc.

The Files application, login screen, Activities overview and various other parts of the desktop feature an updated look. *Boxes*, the default application for managing virtual and remote machines features has, among other things, a revamped box creation assistant and various improvements to the Preferences dialogs.

The Software app, which has come a long way since its introduction in Fedora 20, now also lets you install extras, such as fonts and codecs, apart from installing updates and fetching new software. It's a robust tool that is as friendly as *Ubuntu Software Center*, and just as fast and efficient.

Fedora 22 is the first release that doesn't use *Yum* as the package manager. Instead, it uses *DNF* – a fork of *Yum* – that offers similar functionality and relies on the same RPM package repositories as always. Under the hood, *DNF* uses an improved dependency resolver, hawkey and since DNF is command-line compatible with *Yum*, most of the *Yum* commands, such as `install`, `groupinstall`, `remove` etc work the same way with *DNF*.

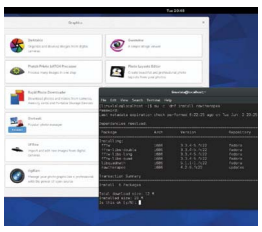
Fedora has often been described as not ideal for new users but the improved *Anaconda* installer is quite friendly. We note that although it doesn't feature major changes, compared to the last release, it seems a bit sluggish. The positioning of the buttons at the top in the installer continues to boggle the mind, but the installer itself is as good as alternatives found in other distros.

Existing Fedora users will appreciate the various visual improvements and feature additions in the latest release. For those already running a RPM-based distro, Fedora 22 is robust, stable and fast enough to tempt you away from your current distro. LXF

Features at a glance

Notifications

Pop-ups, called Banners, are now repositioned to the top of the screen, making them more noticeable.



Package manager

Dandified *Yum*, which is abbreviated to *DNF* is a fork of *Yum* and the default package manager.

LINUX FORMAT Verdict

Fedora 22 Workstation

Developer: Fedora Project
Web: www.fedoraproject.org
Licence: Various

Features	9/10
Performance	9/10
Ease of use	10/10
Documentation	9/10

» *Solid office, internet and multimedia software, coupled with user friendly tools make it an all-round distro.*

Rating 9/10

Sabayon 15.06

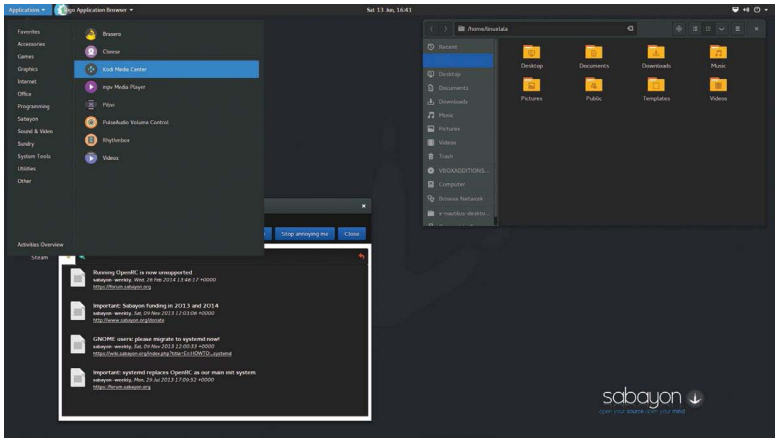
Shashank Sharma tests the latest release of the distro that's named after an Italian dessert – will it leave him craving for more?

In brief...

» Based on Gentoo, Sabayon is a live installable distro along the lines of Fedora and Ubuntu. The rolling release distro only ships 64-bit ISOs, but produces several variants favouring different desktop environments such as Gnome, KDE and Xfce. The latest release focuses on the looks and desktop appearance. See also: Fedora, Ubuntu.

Keeping true to its Gentoo heritage, Sabayon 15.06 Gnome is incredibly fast, especially when compared with its more famous peers. While a Gentoo installation can sometimes last for many days depending on your hardware setup, Sabayon's desire to provide a ready to use desktop out of the box, makes it ideal for those looking to work with a rolling-release distro with the less effort.

Sabayon provides a perfect blend of advanced features like source-based rolling releases with the ease of live installable medium with pre-packaged binaries. As well as minimal edition that's light on graphical environment and uses an *Openbox*-powered desktop, the distro offers 64-bit images for Gnome, KDE and XFCE variants.



› **Although light on some essential applications and with a few minor flaws, the distro can help new users overcome their fear of rolling releases.**

User friendly

The distro wants to be used and is driven by a need to please users. Therefore, it features applications across categories like multimedia and internet and a vast collection of useful accessories. But, despite weighing in at 1.7GB, the Gnome edition doesn't feature an office suite or even a word processor, and various other mainstream applications, eg *VLC*. Worse still, several applications, such as *Videos* and the *Steam* installer refuse to launch and the distro doesn't even spit out a crash report or any errors.

The distro makes up for the missing applications with its impressive package manager, *Rigo*. Although it looks nothing like *Ubuntu Software Center* or *Synaptic*, *Rigo* is a powerful app. It can be used to install updates, configure repositories (repos), in addition to searching and installing extra packages. It also serves as a bulletin board, informing you of community messages.

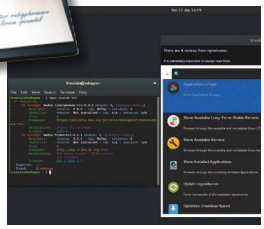
While *Rigo* connects to the Sabayon's software repos that carry binary packages, you can also use Gentoo's repos and install source packages on Sabayon using *Portage*. This unique feature gives users the choice to run a source- or binary-based distro.

The distro uses *Anaconda* installer, but unlike the recent releases of Fedora, Sabayon's installer comes with buttons positioned in sensible places. The install appears to be a little slow, compared to other distros, and while the distro will automatically detect your timezone, it doesn't let you choose what packages to install, as is the standard operating procedure for most distros. You can also use *Gparted* from the live system to carve space for the distro if you dislike *Anaconda*'s partitioner. On our test machines, the installation crashed when we chose LVM partitioning, however there are no forum posts on the subject, suggesting it might be a rare abnormality and not a full-blown bug. The progress bar on the install also freezes once it gets to the halfway point even if the installation continues.

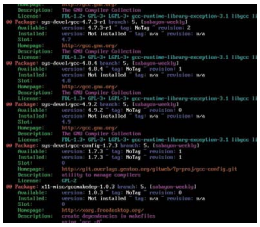
The distro does well to hide its Gnome credentials. The desktop features an Applications menu on the top-left of the screen and features a categorised list of included apps. You can click the 'Activities Overview' button at the bottom for the traditional Gnome 3 look. The distro also ships with support for Nvidia and AMD GPU drivers along with Kernel 4.0, *Kodi Media Center*, *Chrome* and various other internet and multimedia app with out of the box support for various open source and proprietary formats.

Sabayon is like Arch or Gentoo but with a far easier installation. The latest release looks as good as any modern distro even if it could have done with a bit more testing. We recommend this distro if you're looking for a highly customisable but easy to install feature-rich distro. LXF

Features at a glance



Binary or source
The choice of *Entropy* and Portage gives users access to one of the most comprehensive repos.



Highly customisable
Given time, you can use *Portage* to turn the Sabayon install into a perfect Gentoo machine.

LINUX FORMAT Verdict

Sabayon 15.06 GNOME

Developer: Fabio Erculiani and others
Web: www.sabayon.org
Licence: GPL and others

Features	8/10
Performance	7/10
Ease of use	9/10
Documentation	9/10

» *An excellent choice for users looking to move towards Gentoo but are wary of its learning curve.*

Rating 8/10

CrunchBang++ 1.0

While reminiscing over memories of hours spent coding, **Shashank Sharma** tackles CrunchBang++ to see if there are enough pluses to justify the suffix.

In brief...

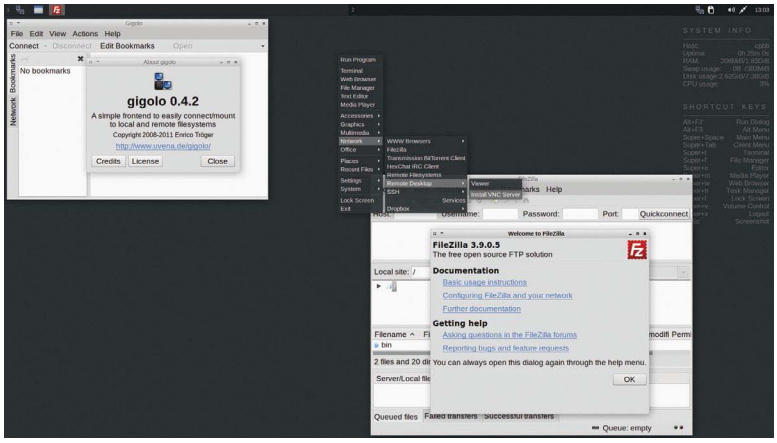
» Alternatively known as CBPP or #!++ and based on Debian 8, the distro features the lightweight *OpenBox* window manager and offers a good balance of speed and functionality. Also features popular apps like *VLC* and *Gimp* along with lightweight alternatives for most others. See also: Puppy Linux, Porteus, Slitaz.

Linux projects come and go. There have been many distributions (distros) with thriving communities that were discontinued for one reason or another. Yet, the announcement of CrunchBang's demise was pretty shocking. The disbelief and the desire for the project to continue spurned a successor, known as Bunsen Labs. Although unaffiliated with any past CrunchBang developer, CrunchBang++ (#!++) is an attempt to continue the legacy of one of the most loved minimalist distros.

#!++ is based on Debian 8 and uses its installer as well. While this may have given users pause a few years ago, the Debian installer has become increasingly user-friendly in recent releases. The distro, however, is unable to run a live system, which is a downer for some new users trying to decide whether the distro is worth the trouble of installing, but it does have installable medium for both 32- and 64-bit machines.

Extra bang?

Once installed the distro drops you to a minimalist *OpenBox*-based desktop and automatically launches a post-install configuration script that guides you through some configuration options, such as installing printer support or Java Runtime Environment, *LibreOffice*, LAMP stack etc. It's not compulsory to run the 12-step script, as it requires a working internet connection, and you can launch it at any time from the terminal with `cbpp-welcome`.



» **Although it looks bland, CrunchBang++ is a perfectly packaged distro that can be moulded easily into a highly functional desktop or running a LAMP server.**

The desktop lacks a dedicated application menu so a right-click on the desktop reveals the menu in true *OpenBox* fashion. Although there's a sizeable collection of default apps, such as *IceWeasel* browser, *VLC* media player, *Gimp*, *Abiword*, *Atril* document viewer, *Transmission*, *Filezilla* etc, the more featureful and commonplace apps, such as *LibreOffice* and *Chrome* browser are missing. The distro does provide an installer script for these prominent apps, which saves you the trouble of manually installing them using the graphical *Synaptic* Package Manager or *apt-get* tool. Incredibly, the distro similarly provides installers for DropBox and *VNC* server.

If you're used to any of the popular flashy looking modern distros, you'll find the bare grey-themed look rather depressing. But *OpenBox* is highly configurable; you can change just about every element of the desktop to get a more colourful desktop if you want.

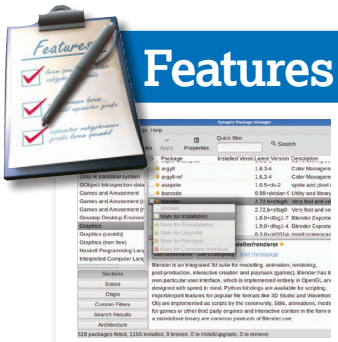
Apart from reporting the essential system stats, the *Conky* system monitor on the right of the desktop also lists a number of default keyboard shortcuts, eg for launching the file manager, terminal, media player etc. Note: The default shortcuts make use of the Super key so you'll have to reconfigure if you don't have one.

Even though the distro ships with only two multimedia tools: *VLC* and *Xfburn* to burn discs, it supports a number of formats out of the box and Debian's non-free repository is enabled by default.

If you spend a lot of time on the command-line, you'll find commands such as `ifconfig` can't be run as a non-root user. This is because **/sbin** isn't included in the $PATH. Run `<em>export PATH=$PATH:/sbin</em>` to clear up the problem.

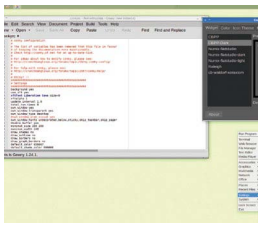
There are plenty of lightweight distros that swap KDE or Gnome for Xfce or LXDE. There are also a host that leave out the desktop environment entirely and build the GUI along a window manager, such as *OpenBox*. While #!++ firmly falls in the latter category, the distro has done much more to deliver a very fast system that's low on resources. LXF

Features at a glance



Debian based

Being Debian-based it's incredibly stable. It's also lightning fast and can be easily moulded for any use.



Highly configurable

Runs the very hackable *OpenBox* and also features tools like *Conky* system monitor and *Thunar* FM.

LINUX FORMAT Verdict

CrunchBang++ 1.0

Developer: Ben Young
Web: www.crunchbangplusplus.org
Licence: Various

Features	8/10
Performance	9/10
Ease of use	9/10
Documentation	5/10

» *CrunchBang++ is a young distro that doesn't even have a forum board (it uses Reddit) but works as advertised.*

Rating 8/10

Kerbal Space Program v1.0

We go boldly where no man has gone before with the help of **Phil Savage** and his brave crew of little green people.

Specs

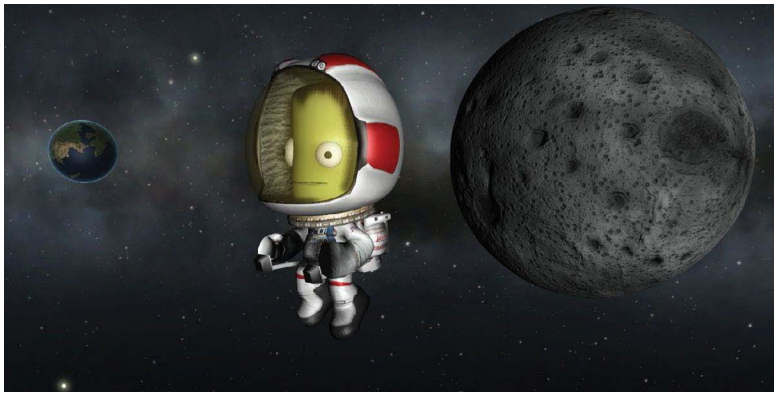
- OS: Ubuntu 14.04 (most distros supported)
- CPU: Intel Core i5
- RAM: 4GB
- HDD: 2GB
- GPU: Any with Shader Model 4, 1GB VRAM

Kerbal Space Program (*KSP*) is about building and flying rockets into space. Chances are you already knew that, because it was first released, in alpha, back in 2011. The latest update brings some significant improvements, but none of its individual additions dramatically change the game. *KSP*'s major pillars have been in place for a while now. Updates were designed to extend these pillars, and over the course of years have shaped the game into what it is now. There are new missions types, more ship parts, female Kerbal astronauts, revised graphical effects, and a new 'Engineer Report' panel that advises about obvious errors in a ship's construction

Sandbox mode is the purest distillation of KSP's essence. In it, you have an unlimited budget and full access to the game's many ship parts. Sandbox games are, by definition, without structure. At the same time, it has a natural progression you can choose to follow. There's a moon, and then planets—each far enough away that visiting the next is a significant step up in difficulty.

The persistent nature of the solar system is also important for those players who aren't incompetent. After a certain point, it becomes impossible to build a rocket big and efficient enough to reach nearby planets. Through the use of permanent orbital stations and tankers, it's possible to create a series of refuelling pit stops for you to dock at along the way.

Will we stay straight through the 10,000m ascent? Maybe?



The Kerbals take the edge off what could have been cold and dry simulation.

If you yearn for more structure, there's also a career mode. At the start of the mode, you're extremely limited in what you can build. It is, however, a slightly smoother and easier start. With less ship parts available, you're given time to figure out what everything does. The contract types can pose scenarios that can prove challenging even to those familiar with the game.

Sim with soul

As a simulation, it would be easy for *KSP* to feel cold and dry. That it isn't is all down to the presentation, and the nature of the Kerbals themselves. They're clumsy, bumbling tinkerers—rated in-game for their courage and stupidity. It's the Kerbals that bring physics down to the layman's level. In a more serious game, the mistakes and failures would feel tragic, or worse, purely theoretical. The Kerbals bring much needed heart to the simulation, and also an element of slapstick. You feel bad when one dies, but not too bad.

It's also easy to see *KSP* as being about engineering and design. Often it is that: a game of tweaking a ship's centre of mass, of increasing stability, or of using the orbital map to perform a controlled burn to a distant destination.

It's not an especially good-looking game. Planetary textures are basic and low-res. It looks utilitarian, which feels apt. Occasionally, though, you'll be floating in space and the sun will emerge from behind Kerbin, or you'll catch the distant glint of another planet. At distance, the engine's lighting excels. In motion, *KSP* is fully able to sell the majesty and awe of space exploration.

It's a reminder: we did that. On the back of science, yearning and, yes, a nearly apocalyptic competition between two superpowers, we strapped people into giant combustible machines and shot them out of our planet. We did that, and it was amazing. It's a rare and wonderful game. LXF

LINUX FORMAT Verdict

Kerbal Space Program

Developer: Sqaud
Web: https://kerbalspaceprogram.com
Price: £30

Gameplay	10/10
Graphics	7/10
Longevity	10/10
Value	8/10

A perfect blend of science and slapstick, and a robust and compelling sandbox of possibility. Outstanding.

Rating **9/10**

SERIOUS ABOUT HARDWARE?

NOW ON APPLE NEWSSTAND & GOOGLE PLAY

Download the day they go on sale in the UK!

Delivered direct to your door

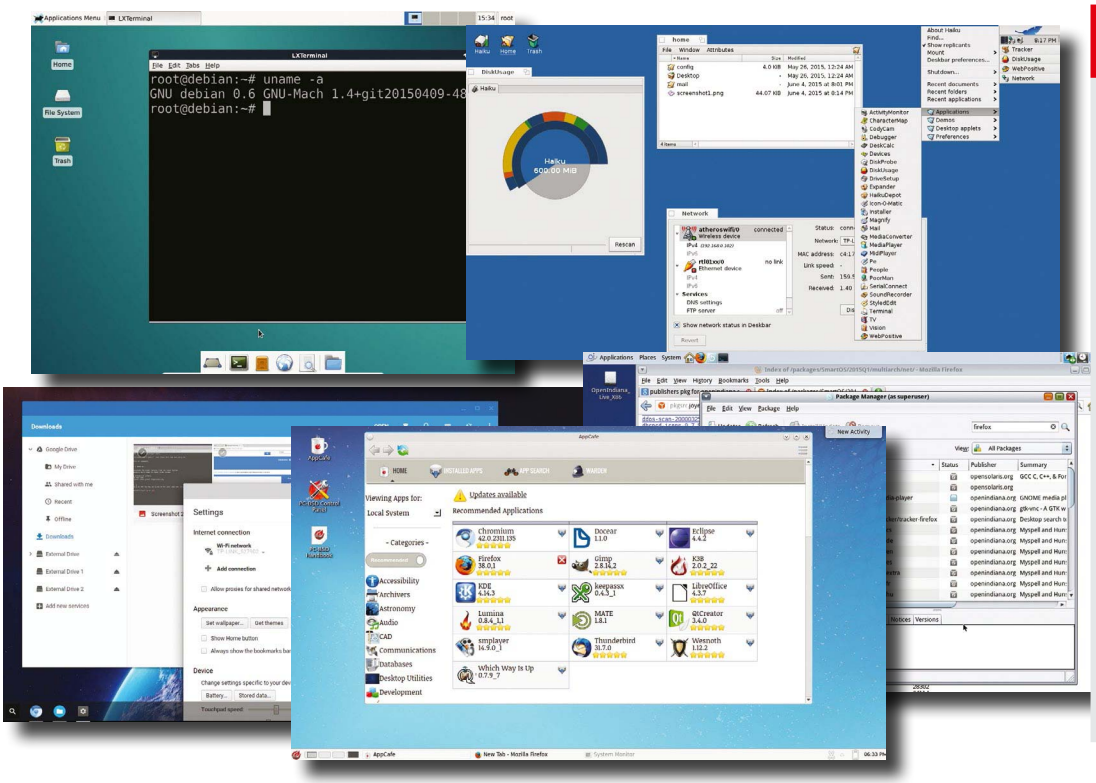
Order online at www.myfavouritemagazines.co.uk
or find us in your nearest supermarket, newsagent or bookstore!

Roundup

» Every month we compare tons of stuff so you don't have to!

Alternative OSes

We love Linux in all its flavours, but it's not the only game in open source town, so for our 200th issue **Alexander Tolstoy** looks at some alternatives.



How we tested...

It's no secret when assessing operating systems that the testing and comparison methods used can affect the results quite significantly. For this Roundup we've tried to negate this as much as we can by testing on both virtual machine (in *VirtualBox*) and on real hardware (an HP laptop with a dual-core AMD CPU and Radeon graphics).

Testing a niche OS on a real-world computer may return controversial results, because the user experience will rely on the actual drivers, but we believe that while some people will play with OSes in safe virtual environment others will be curious enough to run them on a spare partition or a separate hard drive. We'll be comparing these OSes in terms of performance, usability, number of available features and applications, online support and development status.

Our selection

- Chromium OS
- Debian GNU/Hurd
- Haiku
- OpenIndiana
- PC-BSD

There are plenty of operating systems that are open source but don't use the Linux kernel or, at least, have their own user-land software stack. But why on earth would you want to try them out? Well, it can be useful to study different OS designs; their system tools set and generally how they work, and it has to be admitted that some alternative OSes are very strong in particular tasks. For example, OpenIndiana offers enterprise-grade storage features (thanks to Sun Microsystems of old), PC-BSD has all the advantages of FreeBSD and is very good for web servers (and more), Haiku is a unique project, and not related to Unix-based systems at all, but is very fast, and Chromium OS is the open version of Google's Chrome OS, which powers the increasingly popular, fast and battery-conserving Chromebooks.

"We'll highlight OSes that offer the best practical application for the average Linux user."

So can Google's cloud-based OS compete with classical approach of others? Let's see, and we're also going to discover the purest open source project of GNU/Hurd and put it on the line with our other contenders. Our perspective is going to tend to be more desktop-specific and our goal will be to highlight the OSes that are best offering some practical application for the average Linux user.

Hardware specs

Will they run on your PC natively?

OpenIndiana will likely boot fine from live USB stick or DVD and most of system components will work. There is a community-maintained Hardware Compatibility List (**http://wiki.openindiana.org/oi/Components**), which indicates that there's even an official Nvidia proprietary driver for certain chips in OpenIndiana. Radeon chipsets are supported with basic VGA driver, and most of Wi-Fi chips are reported to work.

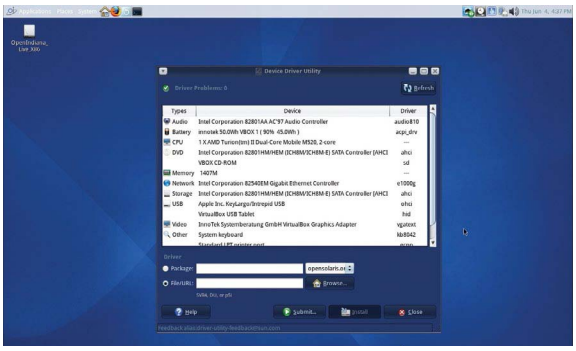
PC-BSD provides the best hardware support and is very close to what we have on Linux. The system offers official Nvidia binaries and Intel drivers for hardware acceleration and a Gallium3D support for most Radeon chips. However, the best OpenGL performance is delivered by *Kwin* in the Plasma desktop environment.

In other aspects PC-BSD matches the hardware compatibility tables of FreeBSD as it uses the same kernel. That means you can install PC-BSD on a real computer and find most of its components working out of the box, including wireless network, printing etc. Of course, not everything is complete: eg Nvidia Optimus will work, but without comfortable switching options between chips, and also some peripherals with Linux-specific blobs can be left unsupported.

Chromium OS has a Linux kernel under the hood and it should deal with most devices acceptably. However, due to forced limitations in the cloud-based OS, it's missing some vital features, such as touchpad support – which is strange considering the OS is targeted at laptops – on some models. In other aspects Chromium OS showed smart chops with perhaps the best support for external peripherals (thanks to the Linux kernel again).

Haiku is a different story. Only two developers work on Haiku's code full-time, so we can't really demand decent hardware support from this tiny OS. Nevertheless, Haiku dealt perfectly with various Wi-Fi adaptors we could find around and surprised us with instant access to WPA2-protected network.

The worst case in comparison with all the others in terms of hardware support is the Debian GNU/Hurd. There's no AGP GART support within Mach, so almost any video chip will be used with the VESA driver; a maximum of 1.7GB of RAM will be used (the rest will be silently ignored); there's no sound support at all; and no USB support (though some keyboards and mice will work thanks to the BIOS emulating legacy interfaces). Finding the right PC configuration on which GNU/Hurd will run would be very tricky.



› OpenIndiana has the Device Driver Utility to show you which drivers are currently in use.

Verdict

Chromium OS ★★★★★
Debian GNU/Hurd ★☆☆☆☆
Haiku ★★★☆☆
OpenIndiana ★★★★☆
PC-BSD ★★★★★

» *You can try booting all the systems, but skip Hurd.*

Ease of installation

What does it take to get them up and running?

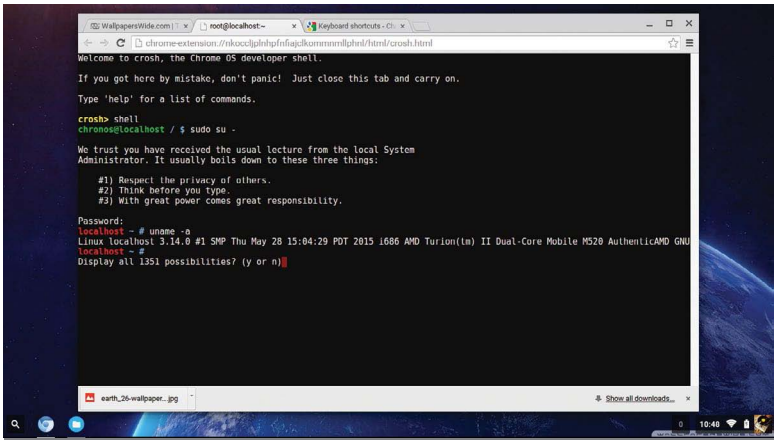
All five contenders in this Roundup were all easy to set up in virtual environment. Selecting an ISO as a primary boot device in *VirtualBox* enabled us to run all of them, either in installation or live mode. We also wanted to challenge each OS on real hardware, writing an ISO on a physical media; a USB stick, for instance.

The OpenIndiana website offers a USB image, which, it turns out, is a little tricky to write on USB. Instead, a regular ISO is easier to use, if you know where to download it. (There is a selection on this FTP page here: **http://bit.ly/1lfPr1m**). This may be an obstacle for people new to the OS.

PC-BSD is flashed to USB easily and offers a clean and very good-looking *Qt*-based installer. Logging into the freshly installed BSD system only takes a few minutes with no hassles.

Chromium OS is disappointing in this area as there's only one mode it runs in. Once the OS image is flashed onto a USB drive, it automatically becames a bootable device with a ready-to-run system. The Chromium OS developer guide has an option to install it on the hard drive, but it's not a real installer, but rather simple scripts that flash a driver from a working Chromium OS environment.

Haiku is the simplest OS to run and install. It offers both live mode and a very good (and fast) installer. Debian GNU/Hurd offers several installation modes, as you'd see in Debian Linux, but it has no live mode, and it took about an hour to install the system.



› Chromium OS is so simple to use, yet hard for a non-developer to set up.

Verdict

Chromium OS ★★★☆☆
Debian GNU/Hurd ★★★☆☆
Haiku ★★★★★
OpenIndiana ★★★★☆
PC-BSD ★★★★★

» *It's tie between PC-BSD's and Haiku's hassle-free installations.*

Performance

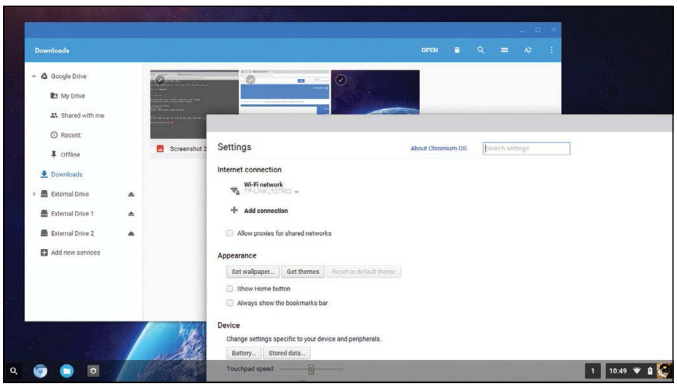
How snappy they are?

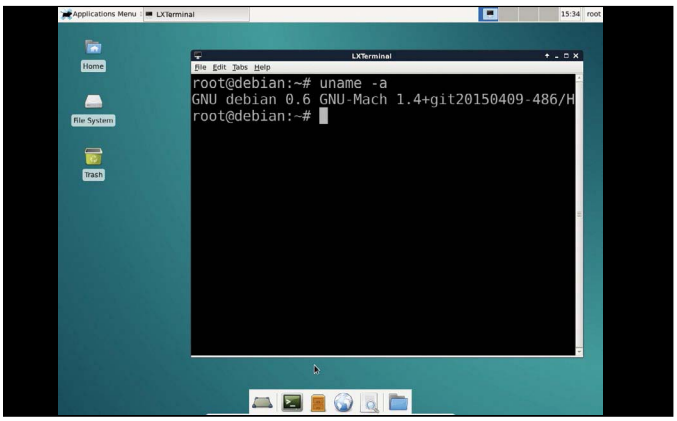
Being used to booting your lightning-fast Linux OS in tens of seconds tends to foster the expectation you can do the same in any other OS. Performance can greatly influence the impression we have of an OS, even if it fails in terms of features.

Performance matters since we want fast installation, fast boot and low latencies between a mouse click on an application's icon and its actual start up. If there's a deviation, we want to find out where it comes from and if it prevents a user from a comfortable computing experience. The difference between th OSes' performance was apparent in the virtual environment, and it was starkly apparent on bare-metal, although your experience will be a little skewed depending on your actual hardware.

Chromium OS ★★★★☆

There can be little or no complaints regarding Chromium OS's performance: it uses the Linux kernel, is based on Gentoo and uses the most recent versions of all system components, wiping off nearly all local applications. After the *X.org* server starts, the rest is handled by the browser, including user login and session, managing windows etc. Chromium OS tries to use pure versions of Gentoo source packages, however a significant number of patches come from Google and the community to optimise Chrome/Chromium OS on certain target hardware, such as Chromebooks. So the OS's performance is very good and if something lags in the Chromium desktop, it's the fault of the browser code, not the underlying base system. Bearing that in mind, you can play with the OS on hardware with 1GB of RAM and a low-end CPU.





Debian GNU/Hurd ★☆☆☆☆

Debian GNU/Hurd has finally brought *X.org* support to Hurd as an out-of-the-box experience. However, from the desktop user perspective the system is incredibly buggy and unstable. To start with, it doesn't bring the graphical desktop under a regular user – we managed to get to the LXDE desktop by issuing `$ startx` under root.

A system running Hurd also feels sluggish and slow to respond, and it isn't easy to determine whether this is due to the unhurried 2D performance with the VESA driver or microkernel I/O issues. We tried to run Phoronix Test Suite for Debian/GNU Hurd, but it turned out that only a few tests would run, such as LAME MP3 encoding, C-Ray, 7-Zip compression etc – and they all indicated a small under run of 4-6% behind the regular Debian GNU/Linux distribution, but still didn't shed any light on why Hurd was so very slow.

Support level and quality

How much help can you get on the web?

Stepping outside the Linux world means that sooner or later you will encounter some problems, so the important question will be: where do you get answers?

OpenIndiana has the large website (**http://wiki.openindiana.org**) with detailed chapters on building, installing and using the system, there is one caveat – the information is targeted mainly for developers and sysadmins, and there isn't anywhere else to look too, other than googling around.

PC-BSD is significantly better in terms of support, as it has a gorgeous community support page (**www.pcbsd.org/en/community**) with an abundance of links to forums, mailing lists, IRC rooms, blogs etc. There are also many non-official PC-BSD resources and Free-BSD websites, that are relevant to both.

Chromium OS has a number of guides at **www.chromium.org/chromium-os**, including Quick Start instructions, but it feels like very little information is being shared with the general public, while the main action still takes place inside Google.

The Haiku project has an official user manual, developer guide and other materials at **www.haiku-os.org/guides** and all information is quite concise.

Finally, the documentation at **www.gnu.org** which makes good on pages at **www.debian.org/ports/hurd** is perhaps the best element in the whole GNU/Hurd project. No wonder, as it's existed since the mid-1990s.

Verdict

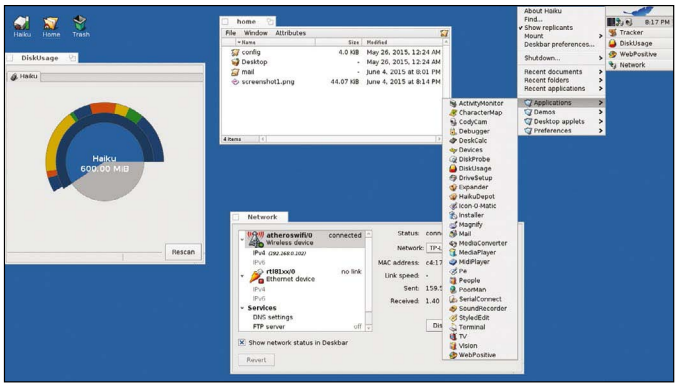
Chromium OS ★★★☆☆
Debian GNU/Hurd ★★★★☆
Haiku ★★★★☆
OpenIndiana ★★★☆☆
PC-BSD ★★★★★

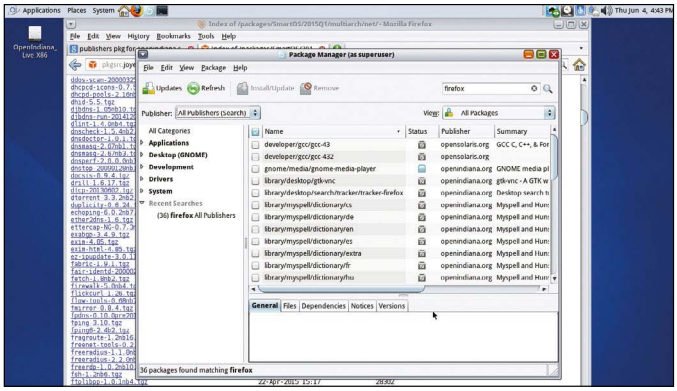
» *BSD systems are strong rivals to Linux in terms of support.*

Haiku ★★★★★

We praised Chromium OS a lot for being very fast and fluid, so you might think it would turn out to be fastest OS in the Roundup. Why only four stars then? Well, Haiku runs faster than Chromium OS; faster than any Linux flavour and out and a way faster than other system in our tests.

Haiku surprised us, showing the best figures for each and every task thrown at it. It takes 10 seconds to boot and 1-2 seconds to open any application. Bearing in mind Haiku is 32-bit only with no options, built largely with the ancient GCC2 compiler and without graphic acceleration, which makes our results astonishing. Haiku is a clear winner here with a perfectly optimized graphics stack and tiny footprint in all aspects. Haiku won't shine on CPU-heavy operations and compression, but it's blazingly fast forordinary desktop operations.





OpenIndiana ★★★☆☆

OpenIndiana's performance dips when system configurations use the basic VESA video driver, which lowers the desktop responsiveness. Regardless of video driver, the OS shows noticeable latency when starting and running various applications. The OS relies on ZFS filesystem on its root partition, which adds some marvellous features (such as snapshots) but adds a desktop performance overhead.

OpenIndiana also uses a mixed 32/64-bit mode. The Unix kernel can run in fully 64-bit while most system components are 32-bit – that's why they perform slower. PC-BSD also uses ZFS and while the two are different in most other ways, there is no visible difference on the desktop performance side: file operations and 2D graphics are slower than in Linux, but the lag isn't excessive.

PC-BSD ★★★☆☆

Once installed, PC-BSD boots to the login screen at a pedestrian speed taking a minute or so. The overall desktop performance in KDE4 is rather good, however, in both native and virtualised mode (and PC-BSD automatically enables Guest Additions). It's not as fast as the average Linux distro due to slower ZFS desktop performance compared to ext4 and more basic Gallium3D support, but it's still very usable. Applications such as *Firefox* or *LibreOffice* would start in a few seconds, but *AppCafe* (the PC-BSD software manager) took minutes to initialise, fetch the updates and finally install them – a very unpleasant experience. In many other respects the operating system performed well, it automatically enabled *VirtualBox* Guest Additions and provided accelerated graphics for our Radeon chipset.

Development status

Is the team behind your alternative OS thriving?

An actively maintained OS is crucial for the future of any OS, and each of our OSes has a differing number of developers beavering away on them, and so the time between releases will differ greatly.

OpenIndiana may still be strong thanks to the massive legacy from OpenSolaris community, but the current pace of development is snail-like. The latest release is 151a8, which plopped out of the snail's shell in August 2013; a year after the previous one. The development branch oi_151a9 seems to be alive, but we're not confident about its future.

PC-BSD is much more sprightly, with a new version released every 3-5 months, while Chromium OS boasts hundreds of developers, and its version is synced with the *Chromium* browser releases. However, there are no official ISO images for the Google OS, but rather a set of random builds from various enthusiasts, which resembles a semi-rolling release model.

Haiku OS development is extremely slow, with the latest 'official' release (Alpha 4) dating back to 2012. But the Haiku movement is much more promising with regular events and participation in Google's Summer of Code. Nightly builds of Haiku show off a constant development, even if official releases are far less frequent.

Debian GNU/Hurd had a new release in 2015, based on Debian 8 (Jessie) codebase, which offers hope that Richard Stallman's dream is still alive.

Verdict

Chromium OS ★★★★☆
Debian GNU/Hurd ★★★★☆
Haiku ★★★★☆
OpenIndiana ★★☆☆☆
PC-BSD ★★★★★

» *We're just a little worried about the future of OpenIndiana*

Features and applications

How many useful desktop apps do they offer?

OpenIndiana offers a basic set of desktop applications in its fresh installation (the ISO is less than 900MB) and a few more in two repositories (repos): main and a legacy mirror of old opensolaris.org). There are extra repositories at **http://sfe.opencsw.org** and at **http://smartos.pkg.ec**, but there are very few desktop applications there.

PC-BSD comes with *AppCafe*, a gateway to the system's own repository, and a classical FreeBSD ports support (from command line only). Also, let's not forget a splendid Kldload technology, which enables a BSD system to run Linux binaries, including *Skype*, *Adobe Flash* and some other components, which are available for Linux but not BSD (at least officially).

```
File  Edit  View  Bookmarks  Settings  Help
[atolstoy@pcbsd-5823] ~% su -
Password:
[root@pcbsd-5823] ~# kldstat
Id Refs Address            Size     Name
 1  124 0xffffffff80200000 14eb000  kernel
 2    2 0xffffffff81996000 24e76    drm.ko
 3    1 0xffffffff819bb000 409cc    vboxguest.ko
 4    1 0xffffffff819fc000 105a     vboxvideo.ko
 5    3 0xffffffff819fe000 337a9    crypto.ko
 6    1 0xffffffff81a32000 4ef9     aesni.ko
 7    1 0xffffffff81a37000 1f848    geom_eli.ko
 8    1 0xffffffff81a57000 24dd67   zfs.ko
 9    2 0xffffffff81ca5000 5a75     opensolaris.ko
10    1 0xffffffff81cab000 10725    tmpfs.ko
11    3 0xffffffff81cbc000 a84e8    linux.ko
12    1 0xffffffff81d65000 22949    geom_journal.ko
13    1 0xffffffff81d88000 22bc2    geom_mirror.ko
14    1 0xffffffff81dab000 638a     ums.ko
```

The kldstat command shows what Linux modules are being used by BSD now.

Chromium OS has the only one place to install extra applications from and that's the Chrome Web Store, which is an open marketplace for web apps for both Google Chrome and Chromium OS. But the sad thing is that those web apps can't compete with classic local applications. They aren't entirely awful, but they don't match 'desktop computing applications and it would be totally irrelevant to compare desktop heavyweights, such as *Gimp* with browser extensions on the store.

Haiku has a very modest set of applications, mostly accessible from *HaikuDepot*, its system's package manager. Antiquated sites such as BeBits and Haikuware have been discontinued, though you can find some random apps available for Haiku, such as *Scribus* and a few *Qt4*-based apps. It's not much, but still something.

The GNU/Hurd, Debian team, in contrast, managed to port about 78% of Debian packages to run on the GNU/Mach kernel, but there still aren't any desktops environments other than Xfce and LXDE.

Verdict

Chromium OS ★★☆☆☆
Debian GNU/Hurd ★★☆☆☆
Haiku ★★★☆☆
OpenIndiana ★★★☆☆
PC-BSD ★★★★★

» *Chromium OS lacks the desktop applications of the others.*

First time experience

Are they easy to get used to?

OpenIndiana has a live mode and it welcomes you with a hardened Gnome 2.30 desktop and *Firefox 10* but no *LibreOffice* in its repositories, though *OpenOffice* is there. If you don't mind the antiquated versions of OpenIndiana packages, then you'll feel comfortable. The only real obstacle can be the network card – if a connection isn't found automatically, you'll be faced with a manual setup. (Head to **http://bit.ly/1SYuPk2** for help.)

PC-BSD doesn't have a live mode, and the only challenging part to its install is the partitioner. BSD systems use a different naming convention for disk drives (eg, **/dev/sda**1 will be **/dev/ada0s1a**), but aside from that PC-BSD can be safely installed alongside a Linux distro on different partitions on the same drive. The installer uses KDE4 as default but others are in *AppCafe*.

Chromium OS is tricky to get started with but is easier to work with later on. If you're not a Chromium developer, you can get a pre-built image at **http://bit.ly/ArnoldtheBat** and flash your USB drive with it using `dd`. After you boot, you'll need to access a command prompt with Ctrl+Alt+t to bring up the shell and entering `shell` to access the classic CLI. After that you issue `$ sudo /usr/sbin/chromeos-install` followed by the root password (that's **password** for the images from the link above) and select target device (Note: it will wipe the whole device clean).

As long as Haiku detects your network card you'll be surfing the internet from the *WebPositive* browser in seconds. Haiku is very easy to use and set up, and its interface logic is quite similar to OS X, which is largely thanks to Jean-Louis Gassée, a former Apple executive.

Haiku default browser, WebPositive, is a capable web application based on WebKit.

Debian GNU/Hurd requires extra post-install steps, such as `$ dpkg-reconfigure x11-common` to let users start an X session and some other tricks (see **http://bit.ly/HurdConfig**). But generally the system delivers a horrible experience, with no live mode.

Verdict

Chromium OS ★★★☆☆
Debian GNU/Hurd ★☆☆☆☆
Haiku ★★★★★
OpenIndiana ★★★★☆
PC-BSD ★★★★☆

» *Haiku can outperform any lightweight Linux distro and it also looks cool!*

Alternative operating systems

The verdict

The abundance of open source operating systems proves that a community of open-minded developers can do great things, which are worth at least trying out on your home PC. We don't insist that you eventually switch from Linux to another OS, as we love Linux but almost all of them are more or less capable for desktop computing.

PC-BSD is the winner overall with very good performance in almost all the tests we threw at it. The OS is fast, reliable and able to recognise nearly all hardware components and peripherals. It may be missing the live mode, which could garner it even more attention from open source enthusiasts, but the desktop experience with PC-BSD is nearly the same as we'd expect in a decent Linux distribution.

Haiku is a smart OS and really unlike the other OSes. There are builds made with an ancient GCC 2 compiler, which can still run the original BeOS applications together with relatively modern *Qt4* apps. Haiku development is not fast, however, but small changes have accumulated into features that are commendable, such as working WPA2-protected Wi-Fi connections, better USB support and more.

OpenIndiana is ageing, there's no doubt about that. In the past there were great hopes for OpenSolaris and later on for the OpenIndiana/Illumos project, which was supposed to breathe a new life into the 'true' Unix System V and bring it to desktops. However, there's little work being done now and no fresh releases in recent years. OpenIndiana is still worth trying out though, as it has decent support for modern hardware.

Chromium OS is an effort to replace classic desktop computing with so-called 'cloud' computing. If your PC activity fits into the *Chromium* browser, then it may be the system for you. It's fast, sleek and intentionally hides away your system's settings. We appreciate this approach but the truth is that cloud computing using thin clients can't beat classic local apps in terms of features and flexibility.

The fifth place belongs to GNU/Hurd, an infamous attempt to create the most pure basement of a GNU system. Debian developers made a great job of delivering a working distribution with a GNU/Mach microkernel, but it is still far from being stable and usable.

› You probably won't notice that you're not in a Linux distribution until you get into a terminal.

> **"PC-BSD is fast, reliable and able to recognise nearly all hardware components and peripherals."**

1st PC-BSD ★★★★★

Web: www.pcbsd.org **Licence:** BSD licence **Version:** 10.1.2

» Our first choice after Linux, when choosing an open source OS.

2nd Haiku ★★★★★

Web: www.haiku-os.org **Licence:** MIT licence **Version:** Nightly

» A surprisingly usable, ultra-fast and stable OS for computers of all ages.

3rd OpenIndiana ★★★★☆

Web: http://openindiana.org **Licence:** Mostly CDDL **Version:** 151a8

» An old ox, which makes a straight furrow, with some rough edges.

4th Chromium OS ★★★☆☆

Web: www.chromium.org/chromium-os **Licence:** BSD **Version:** 41

» A browser instead of a full-featured OS? Not this time…

5th Debian GNU/Linux ★★☆☆☆

Web: www.debian.org/ports/hurd **Licence:** GPL **Version:** Hurd 0.6

» So many years, but still it's in the early stage of development.

Over to you...

What is your favourite open source OS, if not Linux? We'd love to hear about your experiences. Write to us at **lxf.letters@futurenet.com**.

Also consider...

We could include a vast array of different operating systems but we will restrict ourselves to mention just four. If you love what Google does, and that isn't everyone we know, why not try Android x86 (**www.android-x86.org**) an unofficial Android port, which runs perfectly on desktops PCs. It doesn't have the drawbacks of Chromium OS while still being a Linux kernel based operating system. Android x86 might not ever become an OS of choice on a desktop, but it runs all those thousands of Android apps perfectly, which could be a painless cure if you don't have an Android-based smartphone.

Another option is ReactOS, which is an open source Windows clone. It has a lot of benefits, even if we're afraid it could lead an average Linux user back in the wrong direction. There are plenty of other open source systems, from the tiny KolibriOS to the massive Darwin forks, so feel free to explore them all. LXF

200th issue – Special offer

Save up to 50%!

Choose your LINUX FORMAT package

#1 for Free Software

Print £27.50

Every issue comes with a 4GB DVD packed full of the hottest distros, apps, games and a lot more.

Digital £22.50

The cheapest way to get *Linux Format*. Instant access on your iPad, iPhone and Android device.

Bundle £37.00

- Includes a DVD packed with the best new distros.
- Exclusive access to the *Linux Format* subscribers-only area – with 1,000s of DRM-free tutorials, features and reviews.
- Every new issue in print and on your iOS or Android device.
- Never miss an issue.

Your subscription will then continue at £37 every 6 months – saving 27% off the print price and giving you up to a 82% on a digital subscription.

*Only available in certain territories: http://bit.ly/LXFwhere

200 BEST-EVER LINUX TIPS

Mayank Sharma celebrates *Linux Format*'s 200th issue by sharing 200 of his best tips collated after years of navigating the nooks and crannies of Linux.

Getting started

Testing and installing Linux distros like a pro.

1 Create a Live distribution with persistent storage
The most popular distros, such as Fedora and Ubuntu, ship with tools that earmark storage space on the live USB disk for saving data that will be available on subsequent reboots.

2 Put multiple live distros on one disk
If you want to test several live distributions you can put them all onto one USB flash drive using either the MultiCD script (which you can find here: **http://multicd.us**) or by using the French MultiBoot LiveUSB tool (**http://liveusb.info/dotclear**).

3 Use an external partitioning tool
While the partitioning tools within the distributions have improved considerably in terms of achieving better control over your disk, it's best to prepare partitions for a Linux installation using third-party tools, such as *Gparted*, which is also installed in the live versions of several distros.

4 Use LVM partitions
One of their biggest advantages to using LVM (Local Volume Manager) is that unlike standard partitions you don't have to estimate partitions at install as you can grow (or shrink) an LVM volume without losing any data.

Get more from the desktop

Be more productive on your favourite desktop.

5 Middle-click to paste
When you highlight some text with your mouse, the text is copied to a special buffer. When you middle-click in a text entry area, a copy of the text that you originally highlighted is pasted into the text entry field.

6 Define keyboard shortcuts
Almost every mainstream desktop allows you to define custom keyboard shortcuts. You'll find the option under the keyboard setting in their respective configuration panels.

7 Touchpad tricks
Move your finger up and down the right side of the touchpad to scroll vertically and tap in the lower-right corner of the touchpad to perform a right-click.

8 Enable workspaces
To enable workspaces in Ubuntu, head to System Settings > Appearance > Behavior and toggle the Enable workspaces option.

9 Install a Dock
On desktops, such as Gnome, cut down the time that it takes to launch your favourite apps by placing them on a Dock, for example the lightweight *Cairo-Dock* which is available in the official repos of most distros.

10 File manager context-menu
The right-click context-menu that is found inside the file manager on most desktop distros are full of useful options that you might have missed, such as the ability to email, compress or restore them to an earlier version.

11 Create Favourites
Place your favourite apps in Ubuntu's Launcher and Gnome's Dash by dragging them from the desktop's respective applications view.

12 Put icons on the desktop
To alter Gnome install the handy *Gnome Tweak Tool* from your distro's repos. Launch the app, head to the Desktop tab and toggle the Icons on Desktop option.

13 Quick Launch menus
Right-click the icons in Ubuntu's Launcher or an app's name in top bar in Gnome to reveal application specific options and actions.

14 Launch commands from the Mint menu
Right-click the Menu applet, choose Configure > Open the menu editor. Then select a sub-menu or create a new one and select 'New Item'. Enter the command in the space that is provided and toggle the launch in the terminal checkbox for CLI apps.

15 Alter power button behaviour
To tweak the setting of the Power button in the GTK-based Cinnamon, head to System Settings > Power Management and use the power button pull-down to select how it responds.

16 Change Panel Layout
To change Cinnamon's default panel layout head to Settings > Panel and use the Panel Layout pull-down menu to select a different style.

17 Add Applets to Panel
Cinnamon ships with several interesting applets that you can add to any panel by right-clicking the panel and selecting 'Add Applets' to the Panel option.

18 Enable compositing
For some bling, enable compositing on Mate by toggling the 'Enable software compositing window manager' option from under Control Center > Windows.

19 Get different widgets on each desktop
To customise the virtual desktops in KDE, right-click the Pager, switch to the Virtual Desktops tab and toggle the option. Now each desktop can have different widgets etc.

20 Run applications as another user
To get an application running as another user (like root) in KDE, right-click the menu icon and select 'Edit Applications', select an existing entry and click 'Copy'. Then navigate to where you want the new entry, click 'New Item', give it any name and click 'Paste'. Switch to the Advanced tab and toggle the 'Run as a different user' option and enter the username of any user.

21 Slideshow wallpaper
Right-click the KDE desktop and click 'Default Desktop Settings'. Use the Wallpaper pull-down menu to select the Slideshow option and point it to a set of images.

22 Enlarge windows horizontally
Xfce users can right-click the Maximise window button to horizontally stretch it across the screen.

Useful keyboard shortcuts

23 Alt+F2
Bring up the Run dialog box.

24 Alt
Search through an app's menu via Ubuntu's HUD.

25 Alt+~
Switch between windows on the same app.

26 Alt+Ctrl+ Up/Down/ Right/Left
Switch between workspaces.

27 Alt+PrntSc
Take a screenshot of the current window.

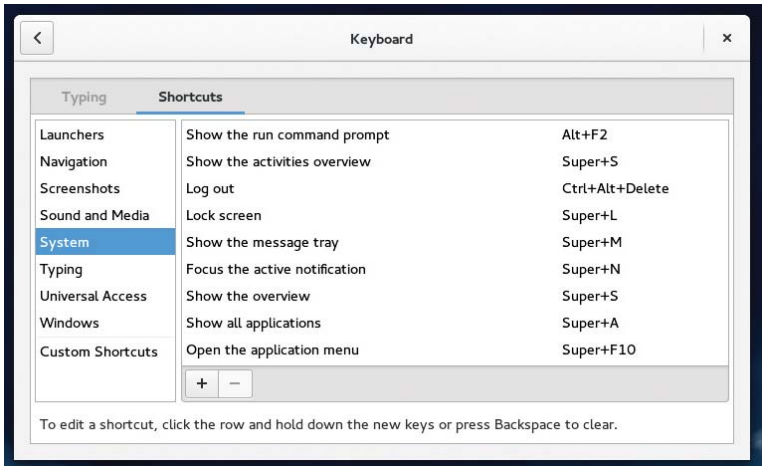
28 Shift+Ctrl+ Alt+r
Record a screencast in Gnome.

29 Super+Up
Maximise windows in Gnome.

30 Super+Down
Minimise windows in Gnome.

31 Super+ Left/Right
Snap windows in Gnome.

32 Super+m
View any missed notifications in Gnome.



› Customise and use keyboard shortcuts to save the time navigating menus.

Tips for your favourite apps

Save time and be more productive with these hidden gems.

LibreOffice

33 Quick change case
Select the words, right-click and head down to Change case menu and select the required option.

34 Enable word completion
Go to Tools > AutoCorrect Options > Word Completion and toggle the 'Enable word completion' and 'Collect words' options.

35 Define Keyboard control
Go to Tools > Customise and click the Keyboard tab to modify any of the shortcuts.

36 Play media files
Head to Insert > Media > Audio or Video and select a media file. Select the media icon in the document to enable media controls.

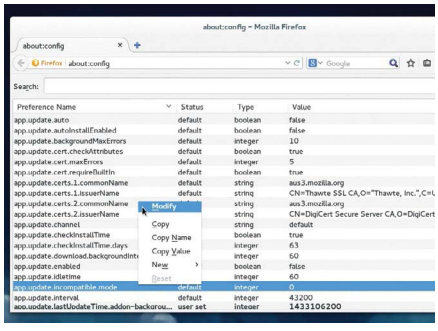
37 Use the Navigator
To swiftly navigate any documents or spreadsheet with the navigator window under View > Navigator.

38 Auto format tables
To auto format them, select some cells and head to Format > Autoformat to choose a different formatting for them.

39 Conditional formatting
Format the cells based on conditions specified under Format > Conditional Formatting > Condition.

40 Protect Sheet
Go to Tools > Protect Document > Sheet to lock access to the sheet with a password.

41 Status bar values
By default the status bar shows the sum of the values in the selected cells. Change the behaviour by right-clicking on the status bar.



› Pop up the hood and take a look inside any Mozilla app with the about:config feature.

Evince

42 Autoscroll PDFs
Right-click inside a document and select the 'Autoscroll' option and use the mouse to control the speed.

43 Make text easier to read
Head to View > Inverted Colors to display white text on a black background.

44 Add Annotations
Select the Annotations option from the drop-down menu in the side pane and switch to the Add tab to add annotations.

Internet apps

45 Speed up the browser (Firefox)
Type `about:config` in the address bar. Then type `network.http` in the filter field and set the network.http.pipelining and network.http.proxy.pipelining parameters to True.

46 Limit RAM usage (Firefox)
Go to `about:config`, filter browser.cache and set the browser.cache.disk.capacity parameter to 30000 if you have 2GB of RAM.

47 Repair folders (Thunderbird)
Right-click the damaged folder, head to Properties and click the 'Repair Folder' button.

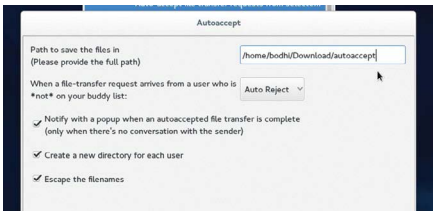
48 Create a mailing list (Thunderbird)
Head to Tools > Address Book > New List and specify which address book list to add addresses to and start adding addresses.

49 Store less mail locally (Thunderbird)
Head to Edit > Account Settings > Synchronisation & Storage for the desired account. Toggle the Synchronise the most recent option and choose the period.

50 Search all messages (Thunderbird)
To search through all mail, including mail only available in full on the server, head to Edit > Find > Search Messages and toggle the 'Run search on server' option.

51 Insert a background image (Evolution)
Toggle the Format > HTML option, head to Format > Page and click 'Browse' under Background image section and pick an image.

52 Advanced search (Evolution)
Head to Search > Advanced Search to create complex search rules. Use the 'Add Condition' button to define parameters.



› Use Pidgin's Autoaccept files plugin to drop files in a folder that you can use with Tip No. 54 to add torrents remotely.

53 Optimise Torrent speed (Transmission)
Use **http://bit.ly/AzureuaUploadCalc** to determine the recommended settings that you can then enter in the Edit > Preferences > Speed and the Network tabs.

54 Monitor directory (Transmission)
Head to Edit > Preferences > Downloading and toggle the 'Automatically add .torrent files from' option and pick a directory.

55 Remote control torrents (Transmission)
Transmission ships with a browser-based interface that can be enabled from Edit > Preferences > Remote.

56 Use a privacy-centric profile (Firefox)
JonDoFox is a *Firefox* profile that automatically integrates with the installed browser and allows you to browse the internet anonymously using a proxy server.

Media players

57 Auto-fetch subtitles (Gnome Videos)
Press Ctrl+Shift+s to open the Movie Subtitles dialog. Now select the language and click 'Find' to look for subtitles on the **www.opensubtitles.org** website.

58 Covert media files (VLC)
Head to Media > Convert/Save, add a file and click 'Convert/Save' button and select the desired codec to convert to.

59 Download online videos (VLC)
Go to Media > Open Network Stream and enter the URL of the video and use the Play pull-down menu and choose 'Convert'. Then select a preset Profile, enter the filename to save and click 'Start'.

60 Record desktop (VLC)
To enable desktop recording, go to Media > Convert / Save > Capture Device. In the Capture mode drop down menu, select

Desktop, then select your frame rate. Finally, click 'Convert/Save', give it a name and click 'Start'.

61 Remote control VLC from a browser (VLC)
Go to Tools > Preference and toggle the 'All' button under Show settings. Now go to Interface > Main Interfaces and toggle the 'Web' option. Then under Main Interface > Lua, set the Lua HTTP Password.

62 Identify a song (Amarok)
Right-click the song you can't recognise, head to Edit Track Details > Tags and click 'Get Tags from MusicBrainz'.

Image editors

63 Move the selection mask (Gimp)
Make a selection, then click the Move tool. Make sure that the Move option is set to 'Selection' in the panel and you can now drag the selection into a new position.

64 Rounded corners (Gimp)
Go to Filters > Decor > Rounded Corners. Then select the 'Edge Radius', which is the amount of curve and optionally customise the other options.

65 Batch process images (Gimp)
Grab and install David's Batch Processor plugin (**http://bit.ly/DavidsBP**) to enable all kinds of tweaks.

66 Automatically write metadata to images (Shotwell)
Head to Edit > Preferences and toggle the Write tags, titles and other metadata to photo files checkbox.

67 Organise photos by events (Shotwell)
By default, *Shotwell* clubs all photos uploaded in one go in a single event. For better organisation you can create new events from a selected group of photos from under Events > New Event.

68 Render RAW files correctly (Shotwell)
To ask *Shotwell* to use the camera's RAW developer, just open an image and toggle the Photo > Developer > Camera option.

KDE apps

69 Bookmarks locations (Konsole)
Use the Bookmarks menu to bookmark any directory. The 'Bookmark Tabs as Folder' option lets you bookmark all open tabs in a single folder.

70 Label tabs (Konsole)
If you've bookmarked a bunch of tabs that you use regularly, you can name them by double-clicking on the tab.

71 Run command on multiple sessions (Konsole)
Use Edit > Copy Input To All Tabs in the Current Window, or Select Tabs if you wish to run the same command, eg on multiple SSH'd hosts.

72 Monitor activity (Konsole)
Enable the View > Monitor for Activity option and KDE will notify you with a popup in the taskbar whenever there's any activity in that Konsole tab.

73 New tab in custom directory (Konsole)
Head to Settings > Edit Current profile and first disable the Start in the same directory as current tab option and then enter the location of the custom start directory in the field above it.

74 Create custom profiles (Konsole)
You can create new profiles with custom fonts and permissions by heading to Settings > Manage Profiles > New Profile. Then customise it by switching to the other tabs such as Appearance.

75 Read-only editor (Kate/Kwrite)
Toggle the Tools > Read Only Mode option to prevent accidentally making changes to an important document.

76 Change highlighting (Kate/Kwrite)
Choose the appropriate highlighting mode for the currently open document by heading to Tools > Highlighting.

VirtualBox

77 Create VM snapshots
To save the current state of a VM, switch to the Snapshot tab in the main interface and click the 'Take Snapshot' button. You can restore snapshots from this interface later.

78 Use USB devices
Head to the Devices > USB Devices and select the USB devices you want to connect which will then be disconnected from the host and made available to the VM.

79 Forward virtual ports
Setup Port Forwarding to ensure any server software inside the VM is accessible from the Internet by heading to Settings > Network > Advanced > Port Forwarding.

80 Enable remote display
If you run *VirtualBox* on a headless server, you can enable the remote display by heading to Settings > Display > Remote Display and toggling the Enable Server checkbox.

81 Manage VirtualBox from a browser
Another useful application for managing *VirtualBox* from a remote computer is *phpVirtualBox*, which recreates the interface inside a web browser.

82 Share clipboard
If you've installed the Guest Editions enable the appropriate option under Devices > Shared Clipboard to copy/paste text between the guest and host.

File manager shortcuts

83 F4 (KDE Dolphin)
Displays the in-line command line.

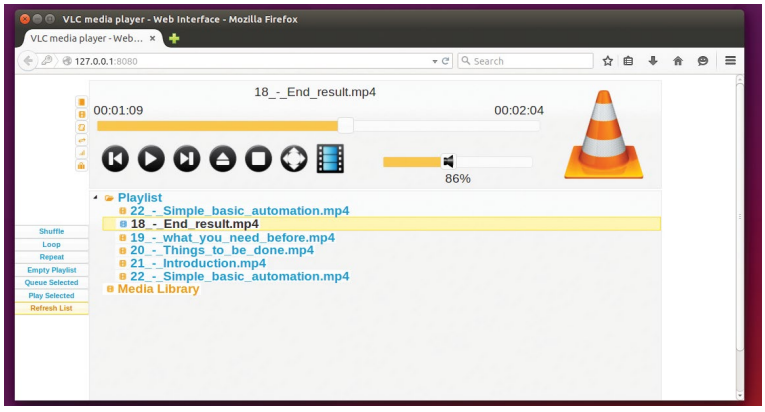
84 F3 (KDE Dolphin)
Splits a single window into two different views.

85 Ctrl+I (KDE Dolphin/ Gnome Nautilus)
View the location bar if hidden (Note: lowercase L).

86 Shift+Enter (Gnome Nautilus)
Open the selected folder in a new tab.

87 Ctrl+Shift+ drag the file (Gnome Nautilus)
Creates a soft link to the file.

88 Spacebar (Gnome Nautilus)
Preview the selected file if the *Sushi* previewer is currently installed.



Once activated, the *VLC* web interface is available at localhost:8080.

Better manage software

Use the command line to get more from your distro's package manager.

Tips for RPM/Yum/Fedora

89 Install RPMs with Yum
To resolve and fetch dependencies install RPM packages with `yum install <package.rpm>`.

90 Update a particular package
Use `yum check-update <package>` to check for updates for the package which you can install with `yum update <package>`.

91 Search for packages
Use `yum whatprovides <name>` to fetch the name of the package that provides the mentioned file.

92 Install package groups
List all available groups with `yum grouplist` and install any with `yum groupinstall <group-name>`.

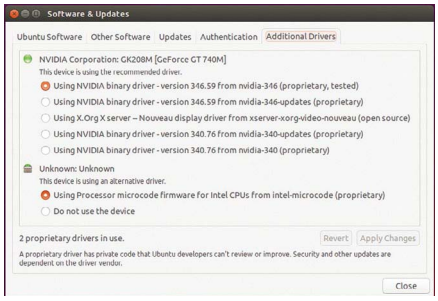
93 Rollback updates
Get a list of actions along with their update IDs with `yum history` and undo one with `yum history undo [update-id]`.

94 Speed up Yum
Install the fastestmirror plugin with `yum install yum-plugin-fastestmirror` and always use the closest mirror to install a package.

Tips for Apt/DPKG/Ubuntu/Mint

95 Backup package list
To install the same packages on another machine, create a list of installed packages with `dpkg --get-selections > pkgs.list`.

96 Replicate on another system
On a fresh installation, first import the list of packages with `dpkg --set-selections < pkgs.list` and then install them with `apt-get dselect-upgrade`.

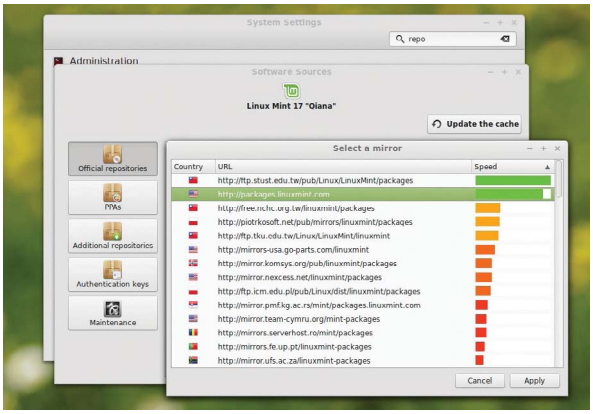


› Use Ubuntu's Additional Drivers tool to install proprietary drivers for your graphics card and other hardware.

97 Uninstall apps
To completely uninstall apps along with their configuration files, use `apt-get remove --purge <app>`.

98 Downgrade packages installed from PPAs
Install the PPA purge tool with `apt-get install ppa-purge` and revert upgraded packages with `ppa-purge <ppa-repo>`.

99 Install dev libraries
To compile a newer version of an app fetch the dev libs of the version in your repos with `apt-get build-dep <app-name>`.



› Linux Mint has great custom software management tools for easy management of mirrors and PPAs.

100 Remove archives
Use `apt-get autoclean` to remove downloaded archives of packages that have since been upgraded to newer versions. You can also get rid of them all with `apt-get clean`.

101 Remove unnecessary packages
The `apt-get autoremove` command zaps all dependencies no longer in use.

102 Fix broken dependencies
Use `apt-get -f install` if you get an error while trying to install a Deb package without installing its dependencies first.

103 Use fastest mirror
In Ubuntu's *Software & Updates*, select 'Other' from the Download from the menu and click the 'Select best server' button.

Tips for URPMI/Mageia

104 Fetch a list of dependencies
The command `urpmq -d <pkg-name>` will get a list of required package dependencies.

105 Update all media
Use `urpmi --auto-update` to update the list of available packages.

106 Saves the RPMs
Append the `--noclean` option to prevent urpmi from automatically deleting the downloaded rpms after installing an app.

107 Install from a local directory
Drop RPMs inside a directory and then add it as an installation medium with `urpmi.addmedia backup <directory>`.

108 Install from a URL
Instead of first downloading packages you can install them directly from the web with `urpmi <URL-to-the-rpm>`.

Tips for ZYpp/OpenSUSE

109 List installed packages
The `rpmqpack` command displays a list of all installed packages.

110 Update a package
Use `zypper in <app-name>` to update a package. The command will also install the package if it isn't yet installed.

111 Faster zypper
Use `zypper sh` to enter the Zypper shell which installs packages faster as it keeps all relevant data in memory.

112 Simulate an upgrade
Before you upgrade your installation do a dry run with `zypper -v dup -D`.

113 Backup repos
Save all the configured repos with `zypper lr --export ~/backup-repos.repo`.

114 Restore repos
Use `zypper ar ~/backup-repos.repo` to restore repos from the backed up file.

115 View required patches
Fetch a list of required update patches with `zypper lp`.

116 Install patches
Upgrade apps by applying all available patches with `zypper patch`.

Power user tips

Become a master of your domain.

System administration

117 Monitor remote systems
Launch KDE's *KSysGuard* and go to File > New Tab. Then switch to the new tab and head to File > Monitor Remote Machine and enter the target machine's IP address and connection details.

118 Mount ISO files
Use mount `-o loop <path-to-ISO-file> /tmp/iso-file` to explore the contents of an ISO image.

119 Create virtual consoles
With *tmux* you can create multiple sessions, run different tasks in each, and then switch from one session to another without interrupting the task running inside them.

120 Use tar efficiently
The tar archiver can detect compression formats and `tar xf <compress-file>` is all you need to unpack a file.

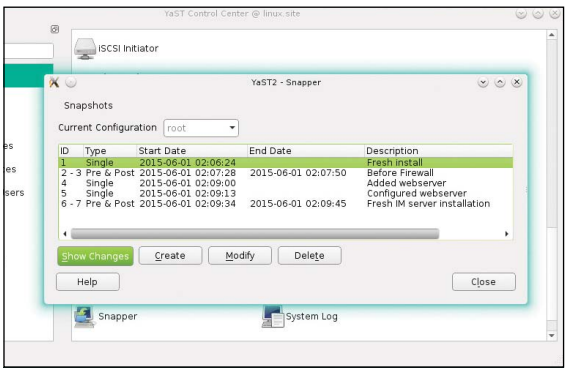
121 Set one-off reminders
You can use `at` with `notify-send` to set short time reminders, such as `echo notify-send "Check on the tea" | at now +4 min`.

122 Schedule a job for multiple times
Use a comma in the crontab file specify multiple times. For example `00 11,16 * * * <task>` executes the task everyday at 11am and again at 4pm (11,16).

123 Run a job within a specific duration
Similarly use a hyphen to specify a range. eg, `00 10-17 * * 1-5 <task>` performs the task from Monday-Friday (1-5) between 10am and 5pm (10-17).

124 Execute a command after every reboot
Use the `@reboot` keyword to run a job whenever the computer starts up.

125 View multiple log files simultaneously
You can install *multitail* from the repos to view multiple file in the following way, eg `multitail /var/log/syslog /var/log/boot.log`.



› OpenSUSE's *Snapper* tool helps you manage snapshots of the distro's btrfs filesystem.

Bash tips

126 View commands matching a pattern
Search through previously executed commands that match a pattern using `history | grep -i <first-few-letters-of-command>`.

127 Reuse arguments from an earlier command
You can use the colon (:) key to reuse the options from the previous command, such as `!!:2` points to the second argument in the previous command.

128 Preview a command before executing
Test your complex *Bash* statements by appending `:p` at the end, such as `ls -l !tar:3:p`.

129 Create shortcuts for commands
You can roll often repeated complex commands into custom ones with `alias`, such as `alias sshbox1='sudo ssh bodhi@192.168.3.111'`. To make aliases permanent add them to the **~/.bashrc** file.

130 Autocorrect CLI typos
You can use `shopt` to autocorrect any common *Bash* typos you tend to create. First, enter `shopt` to display all the available patterns and enable any with `shopt -s`. For example, using `shopt -s cdspell` will find nearest match to misspelt directory names.

131 Create files that are tough to delete
A file with a leading or trailing space in it's name or a hypen (-) cannot easily be zapped from the CLI.

132 Delete tough to delete files
Once you've create a tough file to delete, here are several ways to get rid of files with peculiar names. You can wrap the filename in quotes or use double hyphens, such as `rm " example"` or `rm -- -example`.

133 Delete all files except some
Use the ! operator to remove all files except those that match the specified pattern. For example. `rm ~(*.txt)` will remove all files in the directory that don't end with .txt.

Performance

134 Get details about the hardware
The `dmidecode` command will spit out detailed information about your computer's hardware. For example, using `dmidecode -t 16` will list details about the physical memory. Browse the dmidecode man page for a list of supported DMI types.

135 List process in a hierarchy
You can use `ps --forest` to represent the process tree in ASCII art and clearly identify parent and child processes.

CLI shortcuts

136 Ctrl+a Send the cursor to the start of the command.

137 Ctrl+e Send the cursor to the end of the command.

138 Ctrl+l (lowercase L) Clear the screen but retain what's on the current prompt.

139 Ctrl+k Cut text starting from the command prompt.

140 Ctrl+y Short for 'yank'. Paste the text in the buffer.

141 Ctrl+Shift+c/v Copy and Paste text to the CLI.

Bash shortcuts

142 Shift+PgUp/PgDown Scroll the console.

143 Ctrl+r Search command history.

144 ! <event-number> Repeat a command from history.

145 !! Repeat the last command.

146 Alt+. (dot) Prints the last argument of the previous command.

147 > <filename> Empties specified file.

148 Find memory leaks
To figure out which processes are hogging up the RAM, use `ps --sort mem` which arranges processes in ascending order of memory consumption with the heavy consumers at the bottom.

149 Memory of a particular process
View a detailed memory consumption report of a particular process with `pmap -x <PID>` which displays the amount of resident, non-shared anonymous, and locked memory for each mapping.

150 Trace the execution of a binary
If you have an unknown binary, trace its execution with `stace <binary>` to view all the system calls and signals it makes.

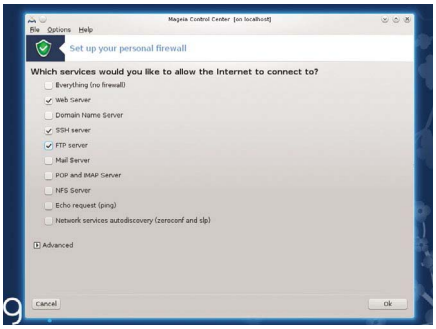
151 Track logged in users
Use the `w` command to get a list of logged in users and their processes. Add the `-f` option to include the hostname of remote users in the output.

152 Kill a graphical app
Type `xkill` in the terminal or the run dialog box which changes the pointer into a cross-hair cursor. Now click on any non-responsive window to kill it. Right-click to dismiss `xkill` without killing a process.

153 Decrease use of swap
If you've got ample RAM, optimise swap usage by editing the **/etc/sysctl.conf** file and changing the value of the vm. swappiness parameter to 10.

Backup

154 Backup the boot sector
A boot sector backup comes in handy when you accidentally wipe out your MBR. Make a backup of a healthy boot sector with `dd if=/dev/sda of=disk.mbr count=1 bs=512` and restore it with `dd if=disk.mbr of=/dev/sda`.



› **Mageia also includes a Parental Controls for time-based and app-based restrictions.**

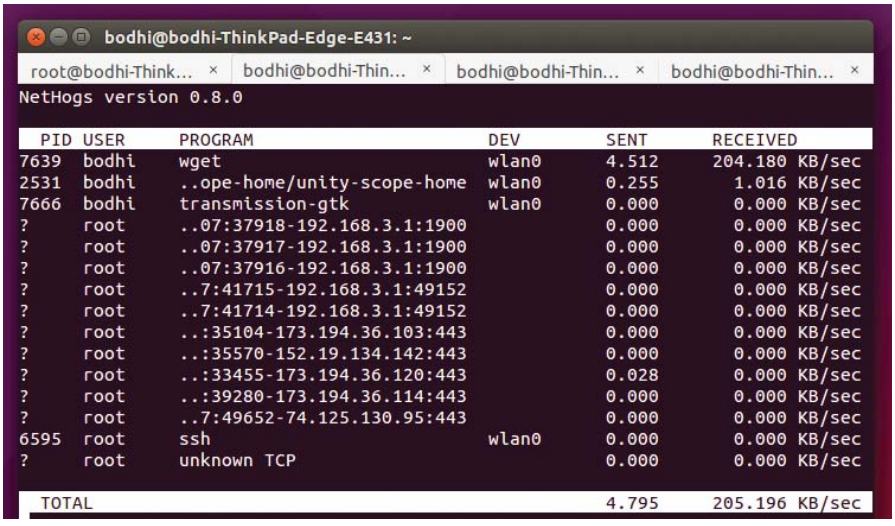
155 Backup partition table
You should also keep a backup of your partition table in the event when a mishap or other zaps this crucial bit of information. Use `sfdisk -d /dev/sda > disk.sf` to backup the table and `sfdisk /dev/sda < disk.sf` to restore the partition table.

156 Monitor the progress of dd
Install the *Pipe Viewer* (pv) tool from your distro's repos and use it to monitor `dd`. For example, `pv -tpreb some-distro.iso | sudo dd of=/dev/sdb bs=4096`.

157 Speed up backups on slower machines
If bandwidth isn't a problem, use `rsync -W` to transfer whole files and save time spent computing the changed blocks and bytes.

158 Track rsync progress
Append the `--progress` option to the `rysnc` command to keep an eye on the data transfer.

159 View changes between source and destination
Use the `-i` option to view the list of items modified by an *rsync* operation, such as `rsync -avzi [source] [destination]`.



› **Use *Nethogs* to get a real-time view of the bandwidth being consumed by an application.**

160 Use rsync over ssh
To transfer *rsync* data over SSH use the `-e ssh` option, such as `rsync -avhze ssh [source] [destination]`.

161 Exclude files
Rsync also lets you skip over certain files that you can specify with the `--exclude` option, like `rsync -avhz --exclude '*.tmp*'` will ignore files with the .tmp extension.

162 Test rsync
First time users should append a `--dry-run` option to all *rsync* operations and scan the output for any unexpected outcomes before running it for real.

163 Limit bandwidth
To make sure the *rsync* operation doesn't hog all the bandwidth restrict its usage with the `--bwlimit option, such as rsync -avhz --bwlimit=50`.

164 Don't backup files on external filesystems
Tar is a popular choice for creating an archive of the disk. Use the `--one-file-system` option with tar to make sure it doesn't backup any mounted partitions (**/media**) or virtual partitions (**/proc**, **/sys**).

Security & Firewall

165 Find which port a program is running on
Use `netstat -ap | grep [app-name]` to see a list of ports that a particular application is communicating from.

166 Disable ping reply
Pings can be used to flood the network and cause network congestion. Disable it temporarily with `echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_all` or permanently by editing the `/etc/sysctl.conf` file to add `net.ipv4.icmp_echo_ignore_all = 1`.

167 Backup iptables
If you've spent customising the kernel's iptables firewall, make sure you back it up with `iptables-save > ~/iptables.backup`

168 Block a particular domain
First, you need to figure out the domain's IP address with `host -t a www.example.com`. Then use the IP address to get its CIDR with `whois [IP Address] | grep CIDR`. Then use the CIDR to block access to the domain, such as `iptables -A OUTPUT -p tcp -d [CIDR] -j DROP`.

169 Change password for any user
If you've forgotten a password for a user, you can set a new one with `sudo passwd [username]` without being prompted for the current password.

170 Replicate permissions
Use the `--reference` option to copy the permissions of one file to another, such as `chmod --reference=[copy-permission-from-this-file] [apply-on-this-file]`.

171 Securely delete files
Install and use the *shred* utility to delete files so that they cannot be recovered. For example, `shred [file]` will overwrite the file's block with random data several times.

172 Enable built-in Firewall
Some distros such as Ubuntu ship with a simpler front-end to *iptables* firewall, called *UFW*. It's disabled by default but you can enable it with `ufw enable`.

173 Allow incoming connection
UFW denies all incoming connections by default. To tweak this policy and allow connections for common servers do `ufw allow ssh, sudo ufw allow www, ftp`.

Network & Internet

174 Run commands remotely
You can also use SSH to execute commands on a remote machine, such as `ssh [hostname] [command]`.

175 Copy SSH keys to another machine
Use `ssh-copy-id [remote-host]` to securely copy the public key of your default identity to the remote host.

176 Keep connection open
If you frequently get disconnected from remote SSH sessions due to lack of activity, you can enable the KeepAlive option by adding the `ServerAliveInterval 60` line in the **/etc/ssh/ssh-config** file.

177 Browse via SSH tunnel
First create an SSH tunnel to a remote host with `ssh -f -N -D 1080 user@remotehost`. Then change your web browser's Proxy settings and set the SOCKS host to **127.0.0.1** and the port to **1080**.

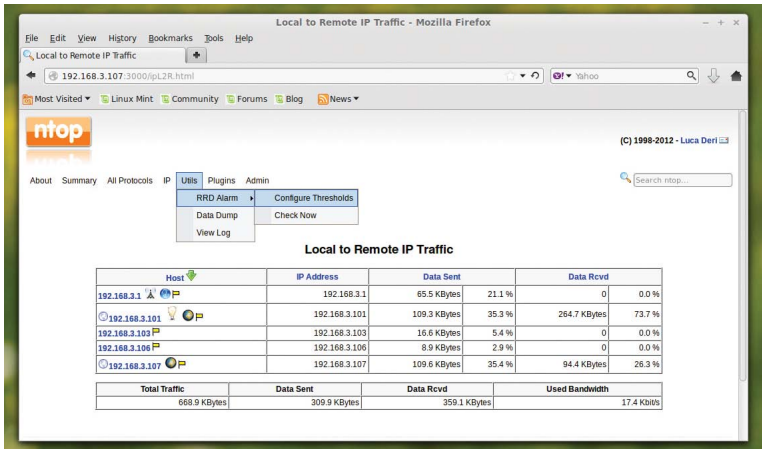
178 Play music over SSH
The command `ssh user@remotehost cat ~/Music/audio.ogg | mplayer` will redirect the output of the remote media file to mplayer on the local machine.

179 Mount partitions over SSH
Use *sshfs* to mount remote partitions such as `sshfs user@remotehost:/home/bodhi /media/remotefs` to mount the remote home directory under the local filesystem.

180 Better monitor network traffic
Ntop is available in the official repos of most distros and gives you detailed analysis of the network traffic via its web-based interface running on port 3000.

181 View network statistics
Use `netstat -s` to view statistics for all protocols or `netstat -st` for only the TCP protocol.

182 Save a webpage
Use `wget` to properly download a webpage. eg, `wget -r -np -k http://www.tuxradar.com/content/dear-edward-snowden` will download all images and change the links in the HTMl and CSS files to point to local files.



› ***Ntop* is a versatile tool that can be extended with plugins.**

183 Save multiple files
If you have saved links to multiple downloads in a file, use `cat isos.txt | xargs wget -c` to download them all.

184 Limit data transfer rate
Prevent *wget* from hogging all the bandwidth by imposing limitations, such as `wget --limit-rate=2m` will limit the transfer rate to two megabytes per second.

185 Download files based on modification date
Use curl with the `-z` option to only download files that have been modified after a particular time. For example, `curl -z 29-May-2015 [download-location]`.

186 Upload files
You can use *curl* to connect to a FTP server and upload files, such as `curl -u [user:pass] -T upload.txt ftp://ftp.example.com`.

187 Get definitions
Curl can fetch the definition of a word from a directory server. List them all with `curl dict://dict.org/show:db` and then query one with `curl dict://dict.org/d:shell:foldoc` which fetches the definition of the word 'shell' from the Foldoc dictionary.

188 Simple web filtering
To prevent your computer from accessing a website enter its URL in **/etc/hosts**, such as *127.0.0.1* **www.addictivewebsiet.com**.

189 Mirror entire websites
Use the graphical *WebHTTrack* tool available in the official repos of most distros to mirror entire websites and automatically modify links

190 Regulate bandwidth
You can use *Trickle*, lightweight userspace bandwidth shaper, to control both upload and download speeds. It can also regulate speeds for package managers such as `trickle -d200 apt-get install`.

191 Monitor bandwidth
To monitor bandwidth used by individual network applications use the *nethogs*, asmall net top tool that's available in the repos of most distros. Instead of breaking traffic down by protocol it groups bandwith by process. LXF

Top command shortcuts

192 Shift+m Sort by RAM utilisation.

193 k Kill a task from within *top*.

194 1 Track all cores individually within *top*.

195 Shift+w Save the modified configuration permanently.

less command shortcuts

196 / Search forward for a pattern.

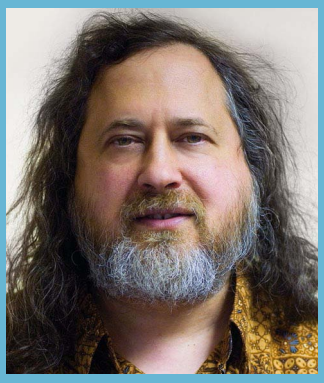
197 n Next match.

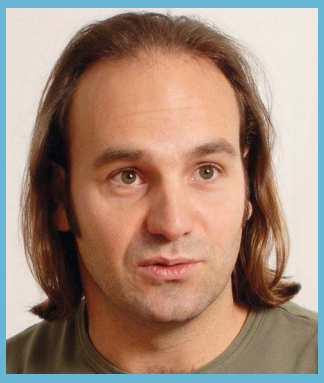
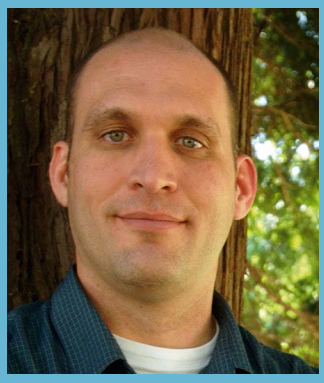
198 Shift+f Displays new content as it's appended to the file.

199 v Edit the file with your system's configured editor.

200 h View the complete list of shortcuts.

Open Source Heroes



Over 15 years *Linux Format* has talked to the biggest names in the open source world, here are just some of the highlights...

Richard Stallman.
Creator of GNU.
LXF145, June 2011

Linux Format: Most of our readers are passionate about free software...

RMS:: But do they think of it as free software?.

LXF: Well, when they contact us, many readers use the term 'free software' – some use 'open source'...

RMS: Ah, that's different, you see. Open source refers to different ideas – a different philosophy. And the difference is fundamental, *on* freedom because it's at the level of values. It's not a disagreement over some detail; it's a disagreement over the most basic thing. We are aiming for a free society, where the users have freedom. Open source organisations and leaders say they're aiming for better quality code. These are about as far apart as you can get, because we're saying it's for freedom and social solidarity, and they're saying it's for quality.

LXF: Isn't that one way you can lead people from one system to another, though?

RMS: I don't understand – they're different.

LXF: But if you have a company that makes proprietary software, it might be hard to adjust its mentality towards the GPL and free software. If you can ease it into the idea of open source, through talking about the benefits of quality – once it gets used to the idea, you can expand...

RMS: They're not the same, and the first doesn't usually lead to the second. In fact, when the open source philosophy spreads a lot – which it has – it tends to close people's minds to the ideas of free software. It even tends to cover up our existence. Most of the articles that talk about the GNU system, they don't call it the GNU system and they don't call it free software. They describe it as open source, and they give the impression that we – its developers – agree with the open source ideas that the readers have heard of already, and would never guess at what we're really standing for.

LXF: When some people hear 'free software', they think of rubbish spyware on Windows machines.

RMS: It took me time to recognise that this distinction was vital. In 1983, when I announced [GNU], I hadn't separated these concepts. It took a few years before I did. So again, in The GNU Manifesto, posted in 1985, there's still some confusion between the two meanings of 'free'. It was after that that I became aware of the need to emphasise that it's free as in freedom, not free as in price. Think of free speech, not free beer. Sure, it would've been better if I had realised that earlier. Although exactly what I would have said – it's not clear, because the English language doesn't have a word that uniquely means what I want to say. The only common

word for free in the sense of freedom is free, so that's why we say 'free/libre', because with that word we can clarify the point.

I notice there's a statement here in your magazine [**LXF143**] about *LibreOffice*, which is an important illustration. Sun acquired *StarOffice*, and released it as free software under the name *OpenOffice.org*. But the people at Sun who did this were not supporters, politically, of the ideas of free software. They were indeed open source supporters. So their goal was to make their program good quality and a success – not to give the users freedom. That wasn't their goal, although since their source code was free software, it did respect the user's freedom, but they weren't thinking about it in those terms. So they made a list of extensions, and in it they put proprietary ones.

So, what could I do about it? Well, we asked people, let's make our own list of extensions. *LibreOffice* uses our list of extensions – they've taken it over. That problem is solved, and the reason that they did this was that the people who are making this version of the program are free software activists – they care about freedom. They will take decisions for the sake of freedom. This shows that people who don't think about freedom or value freedom will sometimes do things for other reasons that help our freedom.

THE TWO MEANINGS OF FREE

"It's free as in freedom, not free as in price. Think of free speech, not free beers."

LXF: You see these purely free GNU/Linux distributions, such as Trisquel and gNewSense, and a lot of them are falling back – they're really scattered projects. Is there room for an official GNU? GNU's GNU/Linux?
RMS: I think it would be good if more of them started working together. But I don't want to start another distro that would be GNU – because that would be a slap in the face to all of those people working on those distros now, and I don't like taking a side among them, having a preference among them. It would be sort of unfortunate to do that.

LXF: Many of our readers want to know what exactly you run as an example. There are the photos on your site of you working with a ThinkPad, but you don't recommend that now.
RMS: I don't use the ThinkPad – those photos are from years ago. Now I'm using this Lemote machine – Yeeloong – you can think of that as 'remote' with a Chinese accent! I chose this machine because it's free all the way down to the BIOS. It has a MIPS-type processor, a Chinese version of the MIPS. In any case, the point is, it solves that problem.

LXF: Going back to the bigger picture, what would you say is the biggest threat to free software in 2011?
RMS: There are several. There are legal prohibitions, such as software patents in some countries that have foolish policies. And there are laws that censor free software explicitly, such as the Digital Millennium Copyright Act in the US, which censors free software that you can break digital handcuffs with. The European Union has similar laws. Both the US and the EU try to push nasty laws like that on to other countries, through treaties that they ask them to sign. So these are malicious governments.

Then there are the obstacles created by manufacturers, working together often with Microsoft. For instance, there are many pieces of PC hardware that can only be used from Windows. And typically the specs of that hardware are not available, so that of course is an unethical practice – to sell someone a product and refuse to tell them how to run it. That shouldn't be allowed. Another obstacle is the tendency to sell computers with bundled Windows. I would recommend prohibiting that practice, too. Then there's the tendency of some companies to donate gratis, or nearly gratis, copies of their non-free programs to schools. Microsoft does this, Apple does this – and I've read that the Gates Foundation does this. Bill Gates's idea of charity is to get school students hooked on Windows, so that he can make more money. That's not charity, I think.

LXF: A question from a reader: Is a world of only free software still feasible? Should that still be the ultimate goal?
RMS: Yes, it's the goal, I think. That's my goal. Now, it may be impossible to totally eradicate the last little bits of non-free software. After all, in almost 200 years of abolitionism, we haven't eliminated slavery. There are places where people are effectively slaves. I've read claims that some foreign workers in the UK are effectively slaves, because if they were to complain, they would end up getting deported. So it's hard to totally eliminate some form of abuse, but I'm sure that a society in which proprietary software is an unusual exception is possible if we demand one together.

LXF: A lot more people are using smartphones and tablets as their primary computing platforms, with their app stores...
RMS: A smartphone is a computer – it's not built using a computer – the job it does is the job of being a computer. So, everything we say about computers, that the software you run should be free applies to smartphones just the same. And likewise to tablets. Now, what should we say about app stores? Well, first of all, the Apple and Microsoft app stores forbid free software. They only allow non-free software. This shows how evil they are.

Now, Android is a different case. The source code of Android is free as Google releases it, but they use a non-copyleft licence, except for the case of Linux – which is under GPL v2. So the result is that the licence doesn't protect the users from lock-down, or Tivoization – which is the practice of making a free program's executable effectively non-free, by stopping the user from installing and using his own version.

LXF: My parents for instance – it's a case of trying to find the right approach...
RMS: I use the analogy of recipes. It's a good analogy, because a program is a lot like a recipe. They're both a series of steps to be carried out to get some desired result. And if you look at the way that cooks use recipes, you'll see that in practice they enjoy the same four freedoms in the way they use recipes. Cooks cook recipes freely, they study and change them when they wish, they redistribute copies, and if they make a modified version, they might distribute copies of their version. So imagine if businesses and the state decided to impose proprietary recipes. Suppose the state said: starting tomorrow, if you copy or change your recipe we will put you in prison and call you a pirate. Imagine how angry all cooks would be. A lot of people who don't know anything about programming will understand this. The state hasn't tried to do it with recipes – but that's exactly what it's tried to do with software. »

Linus Torvalds
Creator of Linux
Linux Format 163, November 2012

LXF: Has the Linux desktop failed because there's too much choice? [Laughter]
Linus Torvalds: I don't think the desktop is doing too well, and there's technical reasons. You've probably seen my rants about how, to some degree, I think the desktop is going in the wrong direction,but the big reason is normal people don't want to install an operating system. You can't get a desktop unless you have pre-installs, and that hasn't happened. There are cases where, if you knew where to look, you could get Linux pre-installed if you bought Dell. But, realistically, nobody has done pre-installs.

LXF: But if the KDE teams and the Gnome teams hadn't spent so much effort creating failed first versions of their desktops?
LT: I know people who decided to give up on the Linux desktop even though they're technical people, just because they got so fed up with Gnome and KDE, so that has been a negative. But at the same time, even if they hadn't done that, I don't think you'd get the normal... the grandmas, people who don't actually like computers, wouldn't have used the Linux desktop. I'm very unhappy with what Gnome and KDE have done, but in the big picture, I think that's a small, small detail.

LXF: Why you don't use the Linux trademark to create a default Linux environment?
LT: I'm not interested. I never wanted to do anything about the technical side. I'm perfectly happy complaining, because it's cathartic, and I'm perfectly happy arguing with people on the internet because arguing is my favourite pastime – not programming. But at the same time the trademark, in particular, I want to have as little as humanly possible to do with that because it's just been a huge pain. It was a pain from the very first, when we had the whole trademark squatter person, but trademarks are ludicrously bad. And, in fact, the legal situation in trademarks encourages corporations to do stupid things, because their lawyers feel if they don't do the stupid things they will lose control of the trademark, and it's bullshit, but lawyers are paid to be anal about things. So I wouldn't want to use the trademark anyway.

LXF: We read that you mostly spend your time committing the merges to the kernel. How do you manage to stay enthusiastic when it's not coding anymore?
LT: Most of my merges, technically they take two seconds to do the actual merge. It takes me more time to read and copy and paste the Git address, and read what's going on...

LXF: And that's enough?
LT: That's enough for most of the code. I get more excited about the... I mean when I get really upset about something, it's when someone does something stupid ... I mean the code is important, but realistically what I maintain these days is not the code but the workflow for people. And that sometimes gets my goat when somebody does something stupid in a big way, and then I get really excited, and by excited I mean I curse at people.

LXF: What happens when Google drops a big Android, as it did early this year after being separate, do you get a heads up on this?
LT: I expected it to be much more contentious. We discussed this before it happened, and I mean a lot of Google Android issues were not so much Google issues as kernel developer issues. Android is doing really well and it's working, and nobody actually had alternative working code that was something Google would accept, because they had issues that nobody else was solving for them.

LXF: Was there any pressure on Google to remerge the Android kernel they were working on?
LT: Kernel developers inside Google hated the fact they had their own patches. I mean, they hated it because they don't like being outside the kernel in the first place, and they hated (it) because it's extra work, too.

LXF: So is it the people responsible for those separate sub-systems that take on the responsibility for Google's part?
LT: Once Google stuff gets working, most of it, so for example [Linus reaches for the phone we're using to record the interview]... Oh, it's an iPhone... screw you! I'm not talking to you anymore... So one of the things that Android had as an issue was the whole wake clock thing, where they want from the system standpoint, to go very aggressively to sleep; but at the same time, they have to be careful with an application that's just about to do something that will wake it up.

It didn't mesh nicely with some of the other power management code, and it didn't really fit some of the theory. And it impacted a lot of the drivers. There was this cascading effect of having this interface that Google needed, that they'd done their way that was not in the standard kernel, and then that affected driver writers, they're using an interface that doesn't even exist in the standard kernel.

Nobody really minds being told what to do. People really minded that the split itself caused pain for no reason except for a minor disagreement. So we decided 'Hey, screw it, maybe we don't love everything Google does or the way they do it', but on the other hand, people didn't really hate it hate it...

THE YEAR OF THE DESKTOP

"The big reason is normal people don't want to install an operating system."

Mark Shuttleworth
Ubuntu founder
Linux Format 71, October 2005

Linux Format: What were your reasons for starting Ubuntu?
Mark Shuttleworth: A number of different things. First, a strong desire to give back. I was incredibly lucky during the dotcom boom [he sold his company Thawte to VeriSign for $575 million in 1999], and a lot of the reasons for that luck came from the fact that I was exposed to open source software at just the right time. I was able to build Thawte on Linux and *MySQL* and *Apache*. I had a very strong philanthropic programme – I kind of need to get rid of everything that I've acquired. I'm quite keen to do that in my lifetime, or in a time that would reasonably approximate what that should be. So I've got to get cracking on it.

To me, this is something that I both feel great about as philanthropy and also see as a speculative investment in having at least some role to play in what the future of the software industry looks like. I can't tell you what I think the software industry will look like, because there is too much hidden in the fog of war at the moment.

LXF: There are many Linux distributions already. Might it not have made more sense to steer one of those towards your vision?
MS: The only one that I thought was really compelling is not steerable! In a very real sense, what we do is the product of my having thought about that. I considered standing for Debian Project Leader, but I figured that there's another way to have the same effect really, and that is to create something that really executes the vision. Make those ideals freely available, and let other people take from that what they will – take the best bits.

LXF: Yes, it is really popular, and it seemed to come from nowhere...
MS: Well, to contrast it with some of the other ventures out there, we just didn't compromise on some of the technical attributes. And I give the team a very free reign to run it technically the way they want to. Building on Debian obviously helps tremendously. So we've been lucky, all of that stuff came together very nicely for the first round. The other group that I think find open source really attractive is at the complete opposite end of the spectrum. They're people who know little about computers and don't want to know anything about computers.

LXF: Some people have said that Ubuntu has been successful at the expense of Debian. Do you think that's a fair criticism? Do you think it matters?
MS: I certainly think it does matter that people feel that way, because I need Debian to be successful and for people to see what we do as constructive. So it worries me whenever that gets said. At the same time, I knew when I started this that it would be controversial within the community, because there are often knee-jerk reactions.

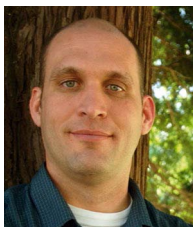
One of the reasons I again decided not to do this within Debian was because I firmly believe that there's nothing an open source team or community can't do – except do everything. There's no particular mountain that you could point to that an open source group couldn't climb, but they couldn't be on top of all of the mountains at the same time. The beauty of this being open source is that we can dispatch different teams to climb different mountains. We can win both battles. So I see Ubuntu as narrowing the scope of Debian, and losing something in the process in order to gain something somewhere else.

LXF: It fits on to one disc...
MS: That's a tremendous narrowing. We've sacrificed a tremendous amount to do that. If you care passionately about everything that's not on that disc, we're not much help to you. But the flipside is in that manner that we gain some other things. We're in the 21st century now, we should act like it. We should really figure out how distributed organisations work. And that's really hard: it's very hard to build to build a company in this completely distributed fashion. We should figure out if it really is possible to build a company around a purely open source vision. I don't know the answer, but I think it's really interesting and worth trying.

LXF: Do you think Linux must conquer the desktop to be considered an achievement?
MS: Linux is absolutely ready for some desktops. So I don't really get stressed trying to convince somebody that Linux is ready for the desktop. I really shift the discussion towards figuring out which desktops they're responsible for. And depending on the scale of your organisation, it's almost always ready in some way. It's very hard for me to suggest to a father of three that his home computer should shift to Linux, as he only has one. »

WHY DEBIAN ISN'T DEBUNTU

"The only one that I thought was really compelling is not steerable!"



Greg-Kroah Hartman
Linux Kernel stable-branch maintainer
Linux Format 81, July 2006

Linux Format: You have mentioned before that other operating systems are using Linux drivers. Syllable, IBM K42...
Greg-Kroah Hartman: Hurd...

LXF: And Hurd, yes! Do you think that embodies the whole spirit of code sharing, or is it just a bad thing in the long term?
GKH: No, I'm surprised that we're not sharing entirely. The IBM K42 guys don't want to write a driver – they want to work on whatever they're working on in their experimental kernel. Nobody likes writing drivers. Some of us do, but a lot of people who are researchers don't, yet they want to get up and running without worrying about playing drivers.

LXF: If no one likes writing drivers, is that because it's so hard to do? Hard to debug?
GKH: I don't think so. I enjoy it, that's what I do. It's different; people traditionally look at drivers as low-end, bad, something you give to the new person coming into the company. Hopefully, over the years Linus has got a bunch of really good people who have changed that, and our drivers are known for stability overall, so we're known for some really good stuff. Networking has been really, really good; SCSI, really good; USB is excellent – we support more new devices quicker than any other operating system. We did USB 2.0 support before anybody else did. Lots of other odd things we got before any other OS. Bluetooth, for instance.

LXF: I've had to install Windows on this thing [indicates his laptop]. It's not easy.
GKH: Yeah, we support hardware faster. All the hardware developers use Linux to do the hardware bring-out. IA-64 was done on Linux, x86-64 was developed on Linux. They can do that, so the hardware system-level guys love Linux. They have the source code, they can see what's wrong with their hardware... The PowerPC guys have been doing some great work. They just released a paper on bringing up Linux on giant multi-processor PowerPCs with no firmware, no BIOS on there. They didn't have to wait for the BIOS guys, the hardware guys can get straight in to it.

LXF: You've said that at the Kernel Summit in 2004 you touched a third of the kernel. I worked this out – that's 1.2 million lines and 850,000 lines removed, which is extraordinary. That seems like a gigantic rewrite, almost.
GKH: You have to take those numbers with a grain of salt. They're metrics. They could be adding and replacing the same lines – but generally they're not. They're adding new drivers, revising kernel APIs, making things better.

LXF: One of the things you're very firm on is binary drivers. Could you explain why binary drivers – say, the Nvidia driver – are illegal?
GKH: The Nvidia driver on its own is not illegal. It's very simple; talk to a lawyer. I'm not a lawyer. The GPL explicitly defines linking.

LXF: Merging GPL code with non-GPL code?
GKH: Yes, when it links – because you need to do that when you load a module – when you're linking code into the kernel you get one system image that's covered under the GPL. It's not a grey area.

LXF: How do some drivers get around this?
GKH: You can do things that are illegal if nobody sees you.

LXF: Sure, but you said that the Nvidia driver isn't illegal itself.
GKH: Because they don't ship anything that's illegal. You as a user do all the compiling and linking together. You cannot pass that compiled object on to somebody else without breaking the GPL.

LXF: Does it make your life easier?
GKH: No, it doesn't make my life easier at all. Binary drivers make our lives hell. Users report problems in their kernel and if they've got a binary driver in there we don't know what it is – it could be writing over any part of the kernel and cause it to oops [crash] or something bad to happen and we wouldn't know. Now, if you crash the kernel we do a report if you're running a binary driver and we'll basically say that we can't support that... if you have a problem, you're on your own. It has been known for people to modify that oops message to show that they're not running a binary driver because they know they won't get any support for it.

LXF: [On the SCO lawsuit] There were no few weeks of doubt, spent quickly checking through source code?
GKH: No. Linux is the best documented large codebase around. It's all known exactly where this stuff comes from. You can trace the history of all these public changes back forever. So it's not like we don't know where the stuff comes from, we've always known where it comes from, it's all been out in the open. You could turn around the other way: all these closed source operating systems, where are they getting their code from? How do we know they're not taking our code? I'm not saying that they are, but you know. I don't fear for our stuff at all. LXF

SOME THINGS NEVER CHANGE

> **"No, it doesn't make my life easier at all. Binary drivers make our lives hell."**

Future

lifehacker | UK

Helping you live better & work smarter

LIFEHACKER UK IS THE EXPERT GUIDE FOR ANYONE LOOKING TO GET THINGS DONE

- Thousands of tips to improve your home & workplace
- Get more from your smartphone, tablet & computer
- Be more efficient and increase your productivity

www.lifehacker.co.uk

twitter.com/lifehackeruk facebook.com/lifehackeruk

15 Years of Linux

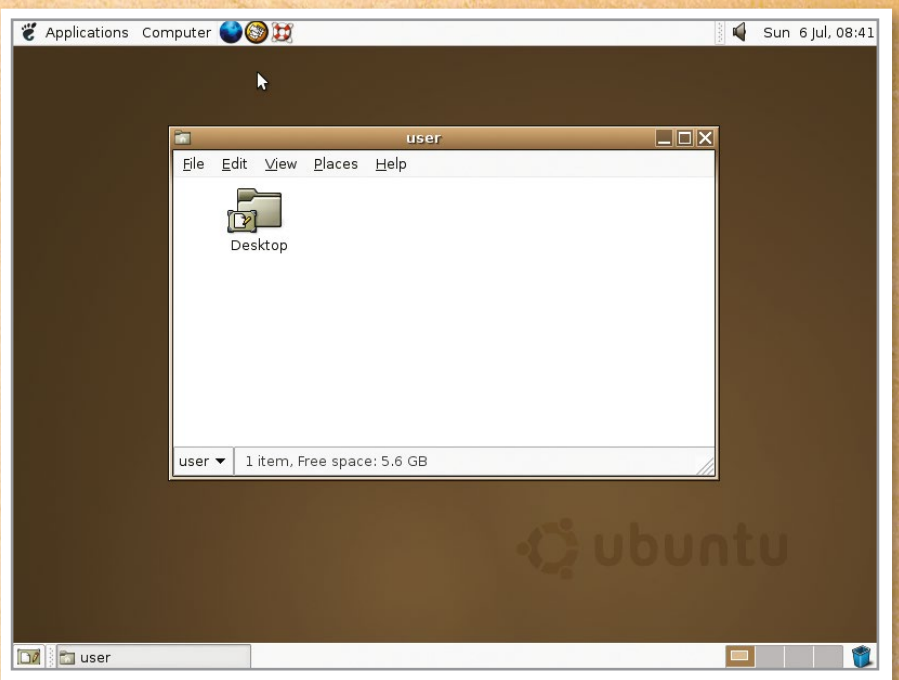
Take a walk down memory lane as **Jonni Bidwell** examines how Linux has changed over the magazine's lifespan.

It was a cold grey morning in May 2000. Winter should have departed but that doesn't happen in Britian. So Reader Zero, seeking respite from the icy rain and miserable population, stumbled into their local newsagent. Zero was hoping for some stimulating and edifying reading material, but was mostly resigned to the notion that the shelves would be populated with the usual feuilletons, corrupt gaming magazines and various 'zines pandering to interests Zero did not possess.

And then he saw it, fluorescent orange, a light in the darkness: "Join the revolution!" the coverline told our enraptured reader. Amazed that frustrated tinkerings at the terminal, considered by their peers an affectation rather than a hobby, could be part of something so exciting and dynamic as a 'revolution', Zero was powerless to resist. There was a free disc too, a whole Linux distribution (Definite Linux) was on there! That would take about a month to download over dial up. And there would be another one in four weeks, and eventually there would be not just a CD but a DVD. Zero's life was changed, and while Definite Linux definitely didn't last long, and the magazine would change hands many times over the next 15 years, it remained a bastion of quality publishing [until Jonni joined – Ed] that would inform, entertain and delight.

> **"A light in the darkness: 'Join the revolution!' the coverline told our enraptured reader."**

Back when Zero was having their cathartic moment in the newsagents, Linux was already about nine-years old. Some distributions (distros) had already established themselves, and one of the earliest was Softlanding Linux System (SLS), which appeared in May 1992. Unlike its contemporaries, SLS provided more than just the kernel and some GNU tools for preparing filesystems, instead it shipped with a networking stack and the X display server. This was considered ambitious and buggy, and efforts to fix this culminated in Slackware's release in 1993. Also that year, and again in response to frustration with SLS, Debian came into being. Red Hat Commercial Linux appeared the following year, which would engender many popular distros of the late 90s, including Mandrake, Yellow Dog and Definite Linux. KDE was released in 1998, with Gnome following in 1999. Gnome was in part created due to KDE's reliance on the then non-freely licensed *Qt* toolkit. By May 2000, the most popular distributions were Debian 2.1, Red Hat 6.1, Linux-Mandrake 7.0 (this was how it addressed itself back then), Slackware 7.0 and SUSE Linux 6.3. Some of these featured in the very first **LXF** Roundup, and you can read all about them in the exclusively digitised issue **LXF1** on the **LXFDVD** this month.



The Human theme was an attempt to make Ubuntu Linux more friendly, because as everyone knows brown is beautiful, especially if you're a warthog.

What's user experience?

If you're a recent Linux convert who's had to engage in combat with rogue configuration files, misbehaving drivers or other baffling failures, then spare a thought for those early converts whose bug reports and invective utterances blazed the trail for contemporary desktop Linux. Up until comparatively recently, it was entirely possible to destroy your monitor by feeding X invalid timing information. Ever had problems with *Grub*? Try fighting it out with an early version of *Lilo*.

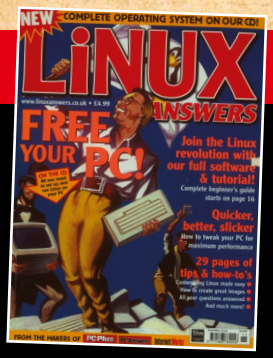
In the early days, even getting a mouse to work was non-trivial, requiring the user to do all kinds of manual calibration. Red Hat released a tool called *Xconfigurator* which provided a text-mode, menu-driven interface for setting up the X server. It was considered a godsend, even though all it did was generate an **XF86Config** file which otherwise you'd have to write yourself. So while Windows users whined about Windows ME being slow and disabling real mode DOS, your average Linux user would jump for joy if their installation process completed. Even if you got to that stage, it would be foolishly optimistic to suppose the OS would boot successfully. Hardware detection was virtually non-existent, and of the few drivers that had been written for Linux, most weren't production quality. Yet somehow, the pioneers persisted – many were of the mindset that preferred the DOS way of working, which began to be sidelined as the millennium approached. Windows users were having their files abstracted away – 'My Computer' epitomises this movement.

> **"Even getting a mouse to work was non-trivial, requiring all kinds of manual calibration."**

In January 2001 Kernel 2.4 was released and with it came support for USB and exciting new Pentium IV processors, among other things. It was of particular importance to desktop users thanks to its unified treatment of PCI, ISA, PC Card and PnP devices as well as ACPI support. The dot-com bubble was just about to burst, but all the excitement and speculation around it meant that many computer enthusiasts had a broadband connection in their home, some even enjoyed the luxury of owning more than one computer. This solved some major entry barriers to Linux: people could now download it much more easily; up-to-date documentation was easily accessible; and when Linux saw fit to disappear one's internet connection (or render the system unbootable), the other machine could be used to seek guidance. But the user experience was still, on the whole, woefully inhospitable. While some installers had evolved graphical »

Timeline

Pre-history – *Linux Answers*
In late 1999 Future plc published a one-off magazine, this was borne off the back of the success of, the now closed, *PC Answers* and *PC Plus* [the flashbacks! – Ed]. It's on the **LXFDVD.** All we'll say is that this was successful enough to launch a monthly magazine...



May 2000 – *Linux Format* #1
Renaming the title in line with Future's most successful print magazines: *Amiga Format* and *PC Format*, *Linux Format* was released with editor Nick Veitch of *Amiga Format* fame and writing talent from *PC Plus* mag. It came with a CD and was an instant hit.

» capabilities, these more often than not were more trouble than they were worth. Users were expected to understand the ins and outs of disk partitioning, and be able to discern which packages they required from often terse descriptions.

Windows XP was released around October 2001, and while this was seen as a vast improvement over its predecessor, many users found that their machines weren't up to running it. After all, it required 64MB RAM and a whopping 1.5GB of disk space. Remember that BIOSes had only recently gained the ability to address large drives (there were various limits, depending on the BIOS, 2.1, 4.2 and 8.4GB were common barriers). So many people couldn't install it on their hardware, and many that met the minimum specs found the performance rapidly degraded once the usual pantheon of office suites and runtime libraries were installed. This provided the motivation for another minor exodus to Linux, and the retro-hardware contingent continue to make up an important part of the Linux userbase (and berate us for not including 32-bit distros). Before 2006 all Macs had PowerPC processors, and many of these (as well as early Intel Macs), long-bereft of software updates from Apple, now run Linux too.

The Gnome 2 desktop environment was released in 2002 and this would become a desktop so influential that some still seek (whether out of nostalgia, atavism or curmudgeonly dislike of modern alternatives) to reproduce it. It aimed to be as simple, tweakable and intuitive, and it's hard to argue against its achieving all of these adjectives.

Oh, we're so pretty

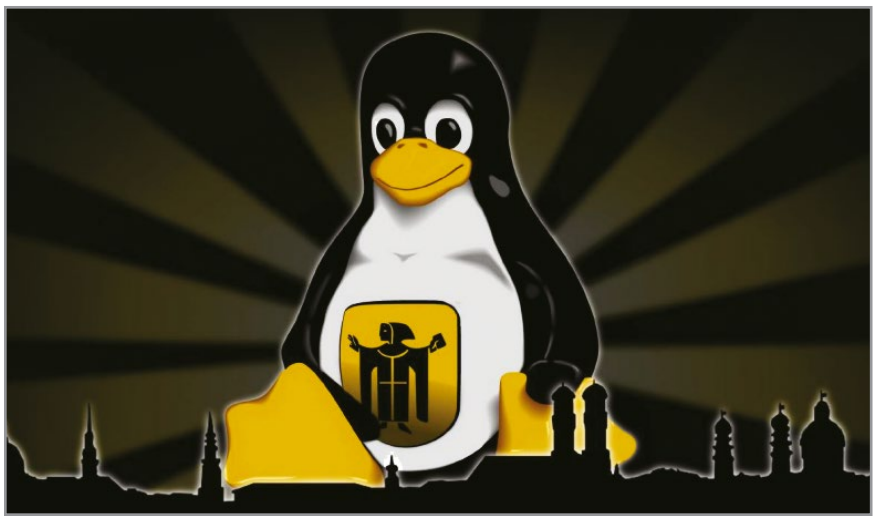
One of the major enablers was its strict adherence to the Gnome Human Interface Guidelines which set out some key principles for application designers. This meant the desktop was consistent not just internally, but in respect to all the *GTK* apps that people would go on to write for it.

> **"Gnome 2: A desktop so influential that some still seek to reproduce it."**

Also released was KDE 3, which vaguely resembled Windows – in that it was cosmetically similar and slightly more resource-demanding than Gnome. People and distributions sided with one or the other. SUSE Linux (predecessor of openSUSE) always aimed to be desktop agnostic, but most of its users preferred KDE. Heeding this, though not until 2009, it changed position and today is the leading KDE-based distro.

In late 2002, 'DVD' Jon Johansen was charged over the 1999 release of the DeCSS software for circumventing the Content Scrambling System (CSS) used on commercial DVDs. This software enabled Linux users to play DVDs, a feat they had been hitherto unable to do since DVD software required a licence key from the DVD Copy Control Agency, one of the plaintiffs in the suit. It later emerged that CSS could be broken much more trivially and Johansen was eventually acquitted. By this time iPods and piracy meant that MP3 files were commonplace. These were, and still are, dogged by patent issues with a number of bodies asserting ownership of various parts of the underlying algorithm. As a result, many distros shipped without patent-encumbered multimedia codecs. The law is murky though, and rights holders have shown restraint in filing suit against FOSS implementations of these codecs. Most distros are prudent and leave it up to the user to install these, although Ubuntu offers users the licensed (but proprietary) Fluendo codecs on install. Fortunately, many of the MP3 patents have expired and many more will have done so by 2017, it doesn't really matter – we have plenty of open formats and codecs now (OGG, FLAC, VPx and x264). It's still technically a DMCA violation to use libdvdcss (a modern and much more efficient way of cracking CSS, used by the majority of media players on Linux) to watch a DVD, but that only applies in some [backwards – Ed] countries and to date, no one has challenged its use.

The city of Munich announced in 2003 that it was to migrate all of its infrastructure from Windows NT to Linux. As well as saving costs, the Bavarians claimed the main impetus for the move was freeing them from vendor lock in. Steve Ballmer visited the mayor personally,



› **The LiMux project branded Tux with Munich's emblem, the Münchner Kindl. Apparently it didn't a hurt a bit. The project is estimated to have saved around €11 million.**

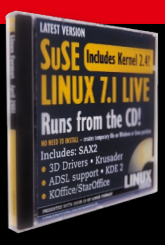
Timeline

November 2000 – LXF007

Corel Linux, the Debian-based distro, was on the CD. The OS may have failed but it was a super-easy introduction to Linux and pointed the way forward for distro developers.

April 2001 – LXF013

The beginning of a new century called for new media, so the DVD age finally hit *Linux Format*! On the first **LXFDVD** you could find SuSE Linux 7.1 and Red Hat 7.0.



May 2001 – LXF014

First a DVD and next an all-new look for **LXF**! The first redesign of the magazine cemented favourites such as HotPicks, Roundup and Answers. The International Space Station was in the Linux news and AMD64 was on the cards.

Graphics drivers and their discontents

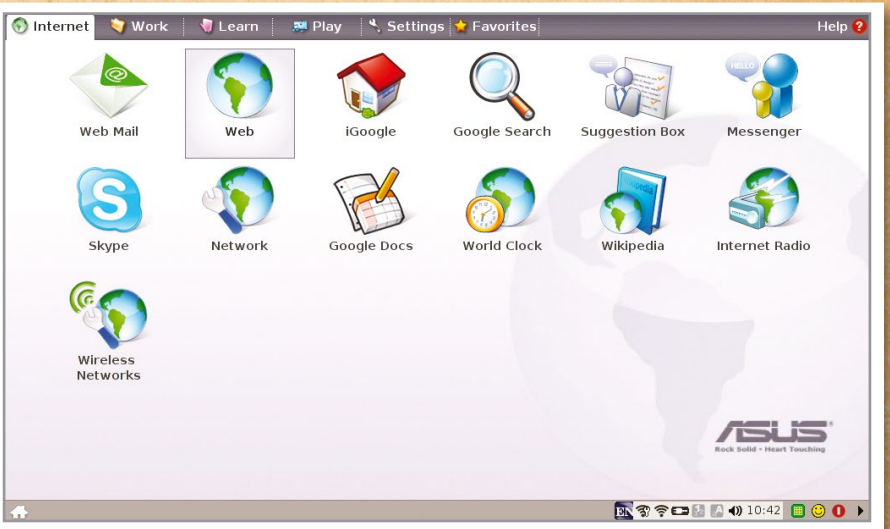
By 2003 Ati (now part of AMD) and Nvidia had both released proprietary drivers to leverage the 3D capabilities of their latest hardware (in 2005 flagship cards were the X1800 and Nvidia's 6800 series). There were open source drivers available, but performance was poor.

ATI were much more forthcoming in releasing device specifications than their opponents, as a result of which their open source drivers developed much more rapidly. Nvidia, through its nv driver, released only some obfuscated source code which left developers puzzled and frustrated. Binary drivers proved troublesome, even with helpful management tools such as Ubuntu's *Jockey*. Repositories would lag behind the latest release, which spurred users into downloading packages direct from the AMD or Nvidia. These were notoriously badly-behaved (we still don't like them now) and would wreak havoc with existing driver arrangements. Since they existed outside the package manager's purview, whenever there was a kernel update the driver module would need to be recompiled. Otherwise there would be no graphics next reboot, which, understandably, some users found upsetting.

This particular situation has been ameliorated thanks to DKMS, but graphics woes continue to be a major source of teeth-gnashing for many users. The story is in many ways still the same: open source drivers are slow and binary ones break things.

In response to the poor performance and lack of 3D support through the nv driver, the nouveau project was announced in 2006. This was a mammoth effort of clean room reverse-engineering, which relied in part on crowd sourced data: Participants would download the REnouveau program which would prod some registers, draw some graphics and then take a snapshot of the register space for developer analysis. It took until 2012 for nouveau to reach a stable release, but it appeared in some distributions some three years earlier, since even in its buggy state it proved superior to nv.

but even his charm and eloquence (and, presumably, offers of hefty discounts) weren't enough to convince the revolutionaries. The project was completed ten years later with some 15,000 machines migrated to the custom 'LiMux' distro. A scare story emerged in 2014 that the city was to revert to Windows, but turned out to be false. It's estimated that the move saved Munich some 11 million euros.



› Asus' EeePC Linux was based on Xandros and *IceWM*, but beginners didn't like it, and professionals just replaced it.

O kernel! My kernel!

After two years in development Kernel 2.6 was released in 2003. This was a vastly different beast to 2.4, featuring scheduler enhancements, improved support for multiprocessor systems (including hyperthreading, NPTL and NUMA support), faster I/O and a huge amount of extra hardware support. We also saw the Physical Address Extension (PAE) so that machines could address up to 64GB of RAM, even on 32-bit architecture. Also introduced was the venerable Advanced Linux Sound Architecture (ALSA) subsystem, which enabled (almost) out-of-the-box functionality for popular sound cards, as well as support for multiple devices, hardware mixing, full-duplex operation and MIDI. The most far-reaching new feature was the old device management subsystem, devfs, being superceded by udev. This didn't appear until 2.6.13 (November 2003), at which point the **/dev** directory ceased to be a list of (many, many) static nodes and became a dynamic reflection of the devices actually connected to the system. The subsystem udev also handled firmware loading, and userspace events and contributed to a much more convenient for desktop users. Although you still relied on such arcana as HAL and ivman in order to automount a USB stick with the correct permissions.

Linux (having already been ported to non-x86 64 bit processors) supported the Itanium's IA64 instruction when it was released in 2001. This architecture was doomed to fail though, and Intel eventually moved to the more conservative AMD64 (or x86-64) architecture, which (we delight in reminding our readers) has been around since 2003. Thanks to open source software, Linux users were running 64-bit desktops right away, while Windows users would have to wait until 2005 for the x64 release of XP. Various proprietary applications (notably *Steam* and its games) run in 32-bit mode, which provides »

April 2002 – LXF026
The second new design for the magazine in as many years! This issue also ran a very popular interview with Samba co-engineer, Jeremy Allison.

May 2002 – LXF027
This issue saw the long awaited results to the reader-voted *Linux Format* Awards 2001. Mozilla won and *Apache* too, while Mandrake picked up best distribution.

February 2003 – LXF037
We asked possibly for the first time: Is this the year of Linux on the desktop? To quote us back then: "I expect 2003 to be a real breakout year." We reviewed LindowsOS 3.0, *Unreal 2003*, while we still liked *IceWM*, KDE and *WMaker*.

» some motivation for distributions to maintain 32-bit releases, but the day will come when these are no longer tenable to maintain, and eventually they will go the way of the 386, no longer supported on Linux since 2013.

Enter the archetype

The 2004 release of Ubuntu 4.10 ('Warty Warthog') was, without a doubt, a major boon for Linux on the desktop. Using the megabucks he'd amassed from creating and selling Thawte, Mark Shuttleworth formed Canonical Inc. The goal was to sell server products and support and at the same time make a desktop Linux "for human beings". Using Debian (it having proven itself by this point) as a base, Canonical added driver tweaks, a very brown Gnome 2 theme and an ambitious six-month release cycle. We also saw the launch of **http://ubuntuforums.org**, where well-meaning but ill-informed members of the community would post 'solutions' to various Ubuntu problems.

In 2004, a sound server called Polypaudio was released by a hitherto unknown developer called Lennart Poettering and some others. At this time desktop environments relied on sound servers to overcome shortcomings in ALSA's dmix system: Gnome was using the Enlightened Sound Daemon (ESD) and KDE was using the analogue Realtime synthesizer (aRts). Polypaudio was designed to be a drop in replacement for ESD, providing much more advanced features, such as per-application volume control and network transparency. In 2006 the project, citing criticism that nobody wants polyps, renamed itself PulseAudio (it was in fact named after the sea-dwelling creature, not the medical condition).

With its new name and increased demand for a sound system comparable with that of OSX or the newly released (and much maligned) Windows Vista, PulseAudio enjoyed substantial development and began to be considered for inclusion in many distros. As is traditional, Fedora was the first to adopt, incorporating it as the default in version 8, released in late 2007. Ubuntu followed suit in 8.04, although its implementation attracted much criticism and resulted in much anti-Pulse vitriol. Poettering at one stage even described his brainchild as "the software that currently breaks your audio". It took some time but eventually Ubuntu (and other distros) sorted out implementation issues, and it now mostly works out of the box.

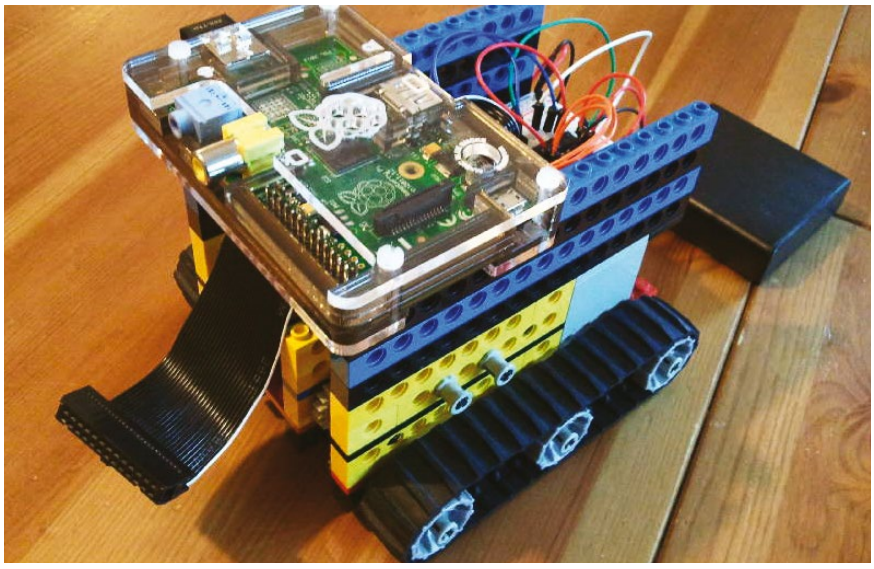
Before tablets, and smartphones that people could afford, netbooks were the pinnacle of portable computing. The first one was the Asus EeePC 701. Due to its low hardware spec (it had a 700MHz processor, 800x480 display and 512MB of RAM) running Windows on it was not an option. Instead it came with a customised version of Xandros Linux, which was functional, but lacking in polish. On the whole most people were unhappy with it, but netbooks still proved great platforms for more experienced Linux users. As newer netbooks were released

Raspberry Pi revolution

The Raspberry Pi was released in 2012. Inspired in part by the success of the BBC Micro (hence the monogram model names) in the early 1980s, the Raspberry Pi aimed to bring practical computer science to the classrooms and bootstrap the UK electronics industry. The low-cost, credit-card sized computer has sold in excess of 5 million units.

While many of these are now empowering young coders, a great deal have become part of diverse man cave projects: The 30-somethings who cut their teeth on BBCs, Spectrums, C64s reliving and reviving the thrills at the interface of coding and creativity. The Pi's GPIO pins mean that all manner of add-ons have been developed, so that the Pi can power anything from robots to remote watering systems.

The lingua franca of Pi projects is Python which, like Basic, is easy to learn. Unlike Basic, though, it is consistent, extensible and won't need to be unlearned should users move on to more advanced languages. The Pi's support for 3D graphics is impressive, but CPU-wise it is more limited. The original Pis struggle to function as a desktop computer, even with the modest Raspbian distribution (although recent work on the *Epiphany* web browser has improved this). In 2015 the Pi received a reboot, gaining a quad-core processor and extra RAM, so now it is a truly multi-purpose computer, and it still only costs £25.



› The Raspberry Pi has inspired a whole new maker generation. When the robot overlords rise up we can blame the Foundation (and send Les Pounder in with the first attack wave).

Time line

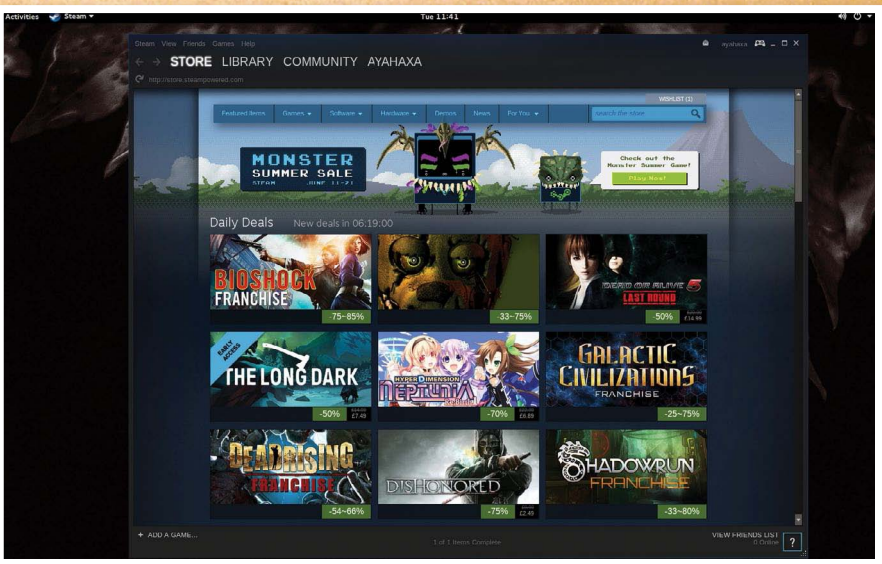
December 2004 – LXF060
The first review of Ubuntu 4.10 by a chap called Jono Bacon, scandalous we're sure; he liked it oddly enough. No, it wasn't on the disc but Mandrake 10.1 was!

January 2005 – LXF061
LXF runs the stalwart Best Distro feature and Mandrake easily wins, poor Ubuntu comes joint 9th. Some chap called Graham Morrison starts as a staff writer and we run Ubuntu on the **LXFDVD** for the first time, alongside Fedora Core 3.

October 2006 – LXF084
The last redesign of **LXF** landed, and the magazine here is largely the design still used today – with the odd section change – the **LXFDVD** was also moved inside the mag.

June 2008 – LXF106
Boy genius, Paul Hudson was promoted to editor/ High Commander and we put the Asus Eee PC centre stage, Jonni still has his running Arch Linux...

(many based around the more suitable Intel Atom chips) they started to ship with Windows XP (some seven years after its initial release) and then the crippled Windows 7 Starter Edition. Asus later backpeddled on its Linux enthusiasm: Teaming up with Microsoft it even launched an 'It's better with Windows' campaign, designed to deter people from purchasing Linux-based laptops. This smear campaign used phrases like 'major compatibility issues' and 'unfamiliar environment' to scare people away.



› Thanks to *Steam on Linux*, Tux gamers finally have thousands of games to play, and LXF writers can peruse the Summer Sale offerings and still claim to be doing work.

The cost of progress

The year 2010 may be remembered by some as the one Ubuntu started to lose the plot. Up until now, the distro had been going from strength to strength, gaining more users, more stability. It was the poster child for the (dead or irrelevant depending on who you ask) dream of Linux on the desktop. But things started to go awry in the 10.10 release. Its *Ubuntu Software Center* now included paid-for apps (the first one was Fluendo's licensed DVD player) and the Netbook remix used a new desktop environment called Unity. In the 11.04 release though, this became the new shell for the main release too. Ubuntu had long taken issue with the new Gnome 3 desktop, which at the time of the Ubuntu feature-freeze was not considered stable enough to include in the release anyway, and Gnome 2 was already a relic. So in a sense Ubuntu had no choice, but no one likes change, and users were quick to bemoan the new desktops. Ubuntu has persisted with Unity and it's much improved today, but a low point came with the 12.10 release when users noticed 'suggestions' from Amazon as they typed queries into the search lens.

> **"The last couple of years have been full of Linux developments and dramas."**

Gnome 3 is not without controversy too – the criticisms it attracted were threefold: First, many preferred the old Gnome 2 way of doing things and this clearly was not that. Second, all the fancy desktop effects required a reasonable graphics card (and also working drivers). There was a fallback mode, but it severely crippled desktop usability. Finally, this appeared to be something designed for use on mobiles or tablets, yet even today mobile Linux (not counting Android) has never taken off, so why should users be forced into this mode of thinking? Many found though, that once some old habits are unlearned and some sneaky keyboard shortcuts are learned (and *Gnome Tweak Tool* is installed), that the Gnome 3 way of working could be just as efficient, if not more so, than its predecessor. KDE users looked on smugly, having already gone through all the rigmarole of desktop modernisation (albeit less drastic than Gnome's) when KDE 4 was released in 2008. Around this point we ought to mention *Systemd* as well, but there's not much to say that hasn't been said elsewhere: the old init system was creaking at the seams, a new and better one came along, it wasn't everyone's cup of tea, but we use it anyway, the internet slanders Lennart Poettering.

There has always been a niche interest in gaming on Linux, but this was mostly done through *Wine*, which has been around since the mid 90s. Things changed when Valve released its *Steam for Linux* client in 2013. Today there are over 1,000 games available for Linux, with more being ported all the time. Granted, many of the high profile ports incorporate either a *Wine* layer or a wrapper such as eOn, but we are also seeing a good proportion of indie releases running natively. Valve even made an OpenGL version of zombie splatterfest *Left 4 Dead 2*, which outperformed the DirectX/Windows release. Linux users make up about 1% of the Steam userbase at present, but this may change if Valve's plan to conquer the living room [why not the desktop!? – Ed] through Steam boxes, running the Debian-based Steam OS, comes to fruition.

The last couple of years have been full of Linux developments and dramas too, including the Heartbleed bug, a partial resolution to the long-running SCO-IBM lawsuit and a much less adversarial stance from Microsoft. But there just isn't enough space, alas. LXF

July 2009 – LXF120
We celebrate Ubuntu 10.04 by putting it on the cover and interviewing Mark Shuttleworth, again. The man just won't leave us alone!

August 2010 – LXF134
What's this Android thing and how can it even possibly have a chance of taking on the iPhone? We explained why, plus Mint 9 and Fedora 13 on the **LXFDVD**.

April 2012 – LXF156
We reported on this thing called the Raspberry Pi back in **LXF147**, but finally the world could buy this tiny PC marvel and the world loved it.

January 2014 – LXF179…
A new editorial team lands at *Linux Format* Towers as the old team departs for pastures new. **LXF179** is the top-selling issue of the year and **LXF181** is the best seller for almost two years! Thank you for helping us keep **LXF** the UK's best seller!

Mr Brown's Administeria

Jolyon Brown
When not consulting on Linux/DevOps, Jolyon spends his time bootstrapping a startup. His biggest ambition is to find a reason to use *Emacs*.

Esoteric system administration goodness from the impenetrable bowels of the server room.

Raise a glass to Linus

My first attempt to get hold of GNU/Linux wasn't very successful. Back in the mid to late '90s I was officially 'learning on the job' looking after a small-office network made up of Windows and Novell Netware. It all seemed a bit poor compared with the Unix workstations I'd used at University, but those cost thousands and were way out of my reach. I accidentally stumbled across talk about something called 'Debian' when looking at options for an email server.

I was stunned. You mean, I could download a Unix-like operating system for free and I could look at the code? Setting to work, I tried to download Debian 1.3 (I think) using internet access that I had literally won in a raffle. Needless to say it took an absolute age and failed several times, eventually causing me to give up in disgust. Bah! It wasn't until an enterprising magazine of the time bundled a Linux distro on its cover CD that I got my first proper glimpse of this brave new world.

From there I went into the bowels of corporate IT department and watched as plucky young Linux upstarts slowly made their way into the data centre in the face of determined opposition (I recall a co-worker at the time excitedly showing me a copy of a new Linux magazine he'd just bought – the 200th issue of which you are reading right now). Of course, these days it's completely ubiquitous and Linux-based OSes are the de facto choice for running anything in 'the cloud'. It's no exaggeration to say that Linus' pet project has ended up sustaining my career (and that of many other people) for years now – I really owe him a beer or three.

jolyon.brown@gmail.com

SourceForge under fire

Plus the end is neigh – Apple open sources Swift and Microsoft finally adds OpenSSH to Powershell.

SourceForge, the long-standing software repository clashed with high profile projects, after it was discovered that downloads of software had begun to include adware bundles. These bundles offer users the chance to download additional commercial packages but some of the included packages were alleged to be malicious by users.

The site's policy is to take over accounts and projects that are no longer actively maintained – such as Gimp which has abandoned the site – and use the adware bundles to generate revenue. This 'DevShare' platform, as it's called, was previously an opt-in program but had been switched to automatic. After an outcry SourceForge issued a note (**http://bit.ly/3rdPartyOptInOnly**) stating that "In an effort to address a number of concerns ... we at SourceForge would like to note that we have stopped presenting third-party offers for unmaintained projects."

There's nothing allegedly malicious about another OpenSSL vulnerability that's been revealed codenamed 'Logjam', which is similar to the earlier FREAK attack [*Administeria, p56* **LXF197**] and just as nasty. It enables an attacker to downgrade the encryption strength between two endpoints enabling it to snoop on traffic. Don't panic though, patches are now available to address this.

In a surprising move, Apple declared that its newest language Swift will be open-sourced later this year and programs will run on Linux as well as its OS X and iOS platforms. This is seen as a move to appeal to developers to foster adoption of the new language.

Microsoft also announced that OpenSSH support is finally arriving for *PowerShell* (which I was griping about in **LXF199**). Tellingly, in the company's announcement, the team responsible said the following: "...this is the third time the PowerShell team has attempted to support SSH. The first attempts were during PowerShell V1 and V2 and were rejected. Given our changes in leadership and culture, we decided to give it another try and this time, because we are able to show the clear and compelling customer value, the company is very supportive". Unfortunately, I don't think I can take credit for this development.

› SourceForge has long supported open source but faces competition in the fiercely contested software repository market.

Amazon Web Service, meet Ansible

Mr Brown's favourite configuration management tool can help you get up to speed with Amazon's cloud platform from your command line.

In a recent Gartner report on Cloud infrastructure as a service, the Amazon Web Services (AWS) platform was ranked as being in the 'Magic Quadrant' (which is Gartner-speak for being the best). While OpenStack remains the most open of cloud 'operating systems', AWS is an important platform to know about. While my personal opinion is that a more open ecosystem is preferable, it's very likely that you will have been, or will be, asked to work with AWS to fire up Linux-based instances and for running all kinds of applications on. I feel it's only right I cover it to at least at some level. Of course, I want to do this all from the command line if possible and have everything I create stored in version control. I'm going to turn to *Ansible*, my favourite configuration management tool at the moment (and one which I covered briefly in **LXF197** and **LXF196**).

So what to build? Recently I was asked to set up a proof of concept for a client that wanted to test whether they could transfer their WordPress-based sites to Amazon (from a more traditional hosting provider). They wanted more flexibility and the ability to scale easily when one of their sites took a hit from being featured on social media. They also wanted to use a configuration management tool to handle all of these tasks. This month and next I'm going to outline, at a high level, what I did for them and how I went about it. This issue I'll cover getting the basics in place, while next month I'll look at some of the more advanced features AWS offers for this kind of scenario. I'll also gradually replace interactions with the web-based EC2 management console in favour of using *Ansible* for as much as I can.

There are a few prerequisites for following the examples below. An AWS account is needed (which is free – head to **http://aws.amazon.com** and sign up) and *Ansible* needs installing locally, as does good old *Git*. I'm running a vanilla Ubuntu 14.04 desktop (with *Ansible 1.8.2* for the record). I also need to install the **python-boto** package (via `sudo apt-get install python-boto`) which is a python interface for AWS. There are a ton of helpful guides and videos on the AWS website for anyone completely new needing an overview. I'll admit to being slightly overwhelmed with the number of products AWS has these days as demonstrated in the screenshot (*right*). The best way, as ever, to become familiar with a system is to dive in and try something out. Without further ado lets get started.

I got the key, I got the secret...

Under my AWS account name is a drop-down menu with a 'Security Credentials' option available on it. In order to do anything with AWS, I need to generate a pair of access and secret access keys. Amazon recommends setting up multiple 'IAM' (Identity and Access Management) accounts each with their own keys for greater granularity of permissions. This makes sense where a team of people might be sharing an overall corporate account with their own areas of responsibility. As I plan just to kick around a proof of concept I'll stick with the initial set, which I download as a file.

In order to use *Ansible*, I also need to use a couple of associated files: **ec2.py** and **ec2.ini**. These can be downloaded from **https://raw.githubusercontent.com/ansible/ansible/devel/plugins/inventory/ec2.py** and **https://raw.githubusercontent.com/ansible/ansible/devel/plugins/inventory/ec2.ini** respectively.

The **ec2.py** script is used for a dynamic inventory – as hosts may come and go in a cloud environment, the »

› AWS has been accused of being a little overwhelming for beginners. I can't imagine where that perception may have come from...

All the AWS TLAs!

Coming to Amazon Web Service for the first time can be very confusing, largely because of the number of acronyms involved in using it. We've used quite a few in this article, so the following terms might be useful to know:

» **VPC (Virtual Private Cloud)** An elastic network populated by infrastructure, platform and application services that share common security and interconnection.

» **EC2 (Elastic Compute Cloud)** A web service that enables you to launch and manage Linux/Unix and Windows server instances in Amazon's data centres.

» **AMI (Amazon Machine Image)** This is an encrypted machine image. AMIs are essentially a template of a computer's root drive. They contain the operating system and a number of other pieces of core software.

» **PV (Paravirtual)** A class of virtualised instance, PV AMIs run on underying hardware that doesn't have explicit support for virtualisation, but they can't take advantage of special hardware extensions, such as enhanced networking or GPU processing.

» **HVM (Hardware Virtual Machine)**. HVM AMIs are presented with a fully virtualised set of hardware, which provides the ability to run an operating system directly on top of a virtual machine without any modification, as if it were run on the bare-metal hardware. Amazon recommend HVM for best performance, especially when combined with the latest generation of instances.

» **T2, M4, M3, C4, C3, R3, G2** These are all instance types available, eg T2, M4 and M3 are all-general purpose instances. Amazon offers many different types of underlying hardware and virtual machine types. The list is here: **http://aws.amazon.com/ec2/instance-types**.

» traditional static inventory used in other *Ansible* setups isn't quite adequate. The **ec2.ini** file is the configuration for the Python script. There are a number of settings available in there (for instance, which Amazon region to use by default) but I'll leave it as it is for the time being.

Once I've downloaded **ec2.py** and **ec2.ini** and copied them into **/etc/ansible**, I set permissions on them as follows:

```
$ sudo chmod 755 /etc/ansible/ec2.py.
$ sudo chmod 644 /etc/ansible/ec2.ini
$ sudo chown root:root /etc/ansible/ec2*
```

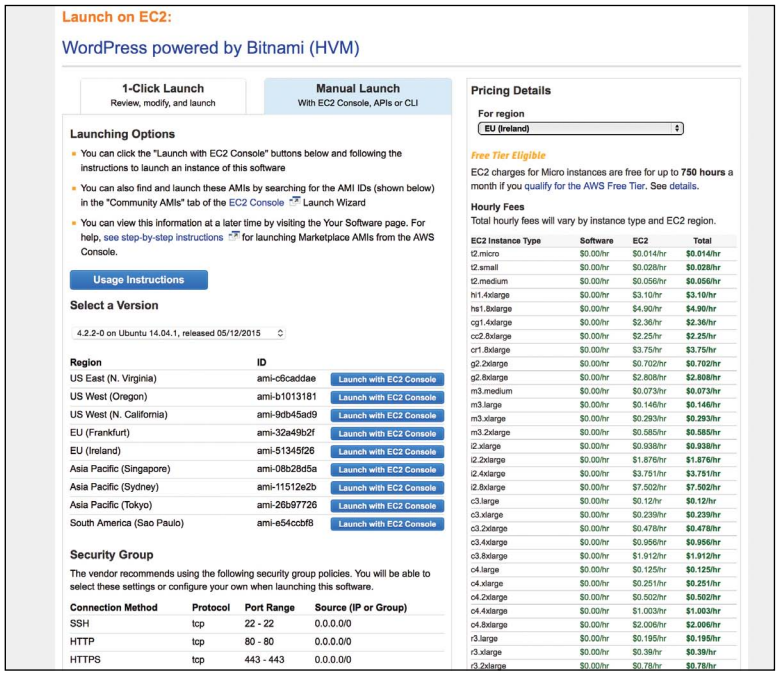
Next, I need to create a key pair so that I can login to any instances that I fire up. This is a just a traditional SSH public and private key pair, separate from the AWS access and secret access keys I've downloaded. I could, if I wished, have uploaded an existing pair created with the familiar *ssh-keygen* tool, but for this demo it's just as easy for me to use the EC2 management console to create a new set using my browser. After naming my keys (lxfkeys – very original) the browser automatically downloads the private key in PEM format.

Next, I create a new folder for my *Ansible* repo and initialise it as a git repo:

» I'm sure there are people who have memorised AMI identifiers. I hope I don't meet them.

```
$ mkdir aws-example; cd aws-example
$ git init .
```

Something I've became aware of when dealing with AWS via *Ansible* is that most of the time I end up running tasks against my local workstation. This differs from the usual *Ansible* model of running them against remote machines. The modules I've used below make calls out to various AWS APIs to then do my bidding on my behalf. In a production environment the advice seems to be that an EC2 instance should be reserved just for running *Ansible* management tasks with the AWS network itself. This leads to decisions about costs – running an instance purely to do this is a waste of money for a small installation, but might be a rounding error on a very large one. Of course, the beauty of the cloud is that instances can be stopped and started at short notice.



Please refer to the manual

Amazon do a fine list of AWS-related white papers which are well worth looking at, covering how to work with specific services to more general topics, such as security (**http://aws.amazon.com/whitepapers**). When researching how to run WordPress on AWS a couple of papers caught my eye. I have two choices: either run a set of standard EC2 nodes, or use the Elastic Beanstalk PaaS (Platform as a Service) system to configure a highly available service in a more sophisticated manner. Looking at the core *Ansible* modules, there's plenty of support for EC2, while the Elastic Beanstalk modules tend to be third-party developments hosted on GitHub. Nothing wrong with that of course, but for simplicities sake I want to stick with core for the time being. The client in question has fairly simple websites so EC2 is a reasonable choice initially. The plan being to get the basics in place first before looking at autoscaling etc.

Back to my git repo. The *Ansible* project maintains a best practices guide (**https://docs.ansible.com/playbooks_best_practices.html**), which suggests an effective directory layout for an *Ansible* installation. Of course, *Ansible* being so flexible means that a website can develop its own preferences on this front, but I've seen the suggested layout used in production very effectively and it's worth reviewing that page for information. I start populating my setup by creating a few key directories.

```
$ mkdir roles keys group_vars host_vars tools
```

For *Ansible* to work with AWS and the **ec2.py** script I need to define some environment variables to my session. For ease, I create a file in the tools directory called **env.sh** and add the following lines to it:

```
export AWS_ACCESS_KEY_ID='<my amazon key>'
export AWS_SECRET_ACCESS_KEY='<my amazon secret key>'
export ANSIBLE_HOSTS=/etc/ansible/ec2.py
export EC2_INI_PATH=/etc/ansible/ec2.ini
```

The key settings are obviously (I hope it's obvious anyway) the two values I generate immediately after logging into the EC2 console for the first time. Adding them to the script like

The business of AWS economics

One of the reasons often cited for avoiding cloud-based infrastructure is the runaway costs associated with paying an hourly rate for virtualised hardware and software.

A quick look at Amazon pricing models can make the casual reader assume they were devised by the same people who created the old gas and electricity billing system, such is their confusing nature for the uninitiated. Part of a sysadmin's job is to either shoulder responsibility for or work closely with other areas who monitor the monthly Amazon bill. The test systems we are building this issue and next should run easily within the free 750-hour usage for a t2.micro instance (but deactivate anything you don't need once you've finished with it). While the pay-for-usage instances generally give the most flexibility, they can be as expensive as other solutions, albeit without the capital cost of buying the hardware and paying for electricity. Amazon does offer other market models, which are well worth investigating.

» **Spot instances** These are resources where the user decides what price to pay. As demand ebbs and flows, the price fluctuates and can drop to the user-defined threshold at which point the instances become available. This is a useful model for tasks which are not deadline-driven or which can be interrupted.

» **Reserved instances** These are exactly what they sound like – resources which are kept for a defined period (one or three years). Amazon offer substantial discounts for these.

With only my mum likely to read this blog, the free Amazon tier should be more than adequate.

this allows me to add them to my environment by simply issuing a `. tools/env.sh` command. Running the shell's built-in `env` command confirms that they became available to my session. Next, I copy the SSH private key Amazon generated for me to the **keys** directory, renaming it **lxfkeys** and issuing a chmod 400 on the file, to make it readable only by me.

(It's at this point I came across what I suspect is an *Ansible* bug during the creation of my test system, however. I needed to create a file which I called **inventory** in my project directory containing just the following two lines – without this I got errors during *Ansible* runs later on.)

```
[base]
localhost ansible_connection=local
```

Now I have to decide what to run. Amazon comes with a large marketplace for software packaged for use on AWS (**https://aws.amazon.com/marketplace**), ranging from familiar Linux distributions to more obscure enterprise type systems. I search for Wordpress and there are a number of options. I decide to try out the 'Wordpress powered by Bitnami HVM' AMI (as it has a good rating and is free to use). After selecting it and agreeing to 'buy' it (a process reminiscent of purchasing a book for the Kindle) I look at the options for manual launch (*pictured p54*). What I'm looking for here is the AMI id, which is unique for this software in the particular Amazon data centre I want my software to run in (Ireland in my case, aka eu-west-1). I need this for use in my *Ansible* script. Amazon emails me to say the AMI is now available for use after a couple of minutes and I'm ready for my next step.

Up and instancing... a blog

Next, I need to make a note of my VPC id. This is basically the portion of the Amazon network that has been allocated to me. This can be seen on the right-hand side of the EC2 dashboard but also has a dedicated drop-down menu of its own (found by clicking on 'Services'). By investigating this I'm also able to make a note of the subnets available to me in this VPC (I have three allocated to me by default). I chose one for use in the file I then create in my *Ansible* repository (which I name **site.yml**).

```
---
# LXF AWS example

- hosts: localhost
  connection: local
  gather_facts: False

  vars:
    region: eu-west-1

  tasks:
    - name: Provision an instance
      ec2:
         key_name: lxfkeys
         instance_type: t2.micro
         image: "ami-51345f26"
         wait: true
         count: 1
         region: eu-west-1
         vpc_subnet_id: subnet-baf628df
         assign_public_ip: yes
```

This very simple file contains all the information needed by AWS for me to start up an EC2 instance. In it, I select a data centre, the size of the instance (t2.micro), the image I want to boot (taken from the AMI page) and also state that I want a public IP address. Ensuring that my environment variables are correct, I then issue:

```
$ ansible-playbook site.yml --private-key=keys/lxfkeys -i inventory
```

The Ansible run completes OK:

```
localhost      : ok=1    changed=1    unreachable=0    failed=0
```

But what's actually happened? I use the **ec2** script to list what instances I now have on the AWS cloud.

```
$ /etc/ansible/ec2.py --list
```

This results in a lot of output: including the IP address of the instance I now have up and running, its public DNS name, its type and all kinds of other information:

```
  "ec2": [
    "54.154.141.142",
  ],
```

However, if I browse to the allocated public DNS name my browser times out. One more change is needed – by default AWS blocks all external traffic. By using the EC2 console and selecting Security Groups from the left-hand side, I can see my default group. Hitting the 'Action' button at the top of the screen allows me to 'edit inbound rules' and I can add 'HTTP' from any inbound source. After this, browse to the public DNS again which brings up the default vanilla Wordpress page, ready for blogging.

In this retrospective issue of **LXF** I think it's worth finishing by noting how easy it is now to spin up a working Linux system somewhere across the world with just a few lines of text and a browser. Next month, we'll try to ditch the browser completely and look at some advanced AWS features. LXF

The best new open source software on the planet

LXF HotPicks

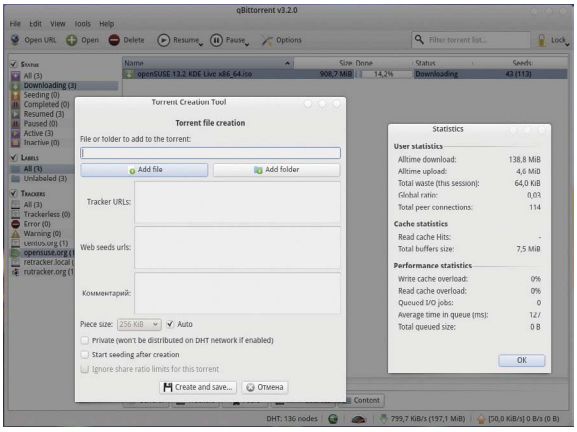
Alexander Tolstoy
has plucked the ripest apps from the vast orchard of the internet to make you a fruity FOSS pie with an extra crusty top and lashings of custard.

qBittorrent » uGet » Tmux » Qt5CT » Krita » Scribus » Kid3 » Tanks of Freedom » Dust Racing 2D » Lollypop » CVAssistant

Bittorrent client

qBittorrent

Version: 3.2.0 **Web:** www.qbittorrent.org



» qBittorrent has everything that torrent devotee might ever need.

The open source BitTorrent client *qBittorrent* is available for all major operating systems, but particularly Linux, where *qBittorrent* claims to be the closest counterpart of the proprietary *µTorrent*.

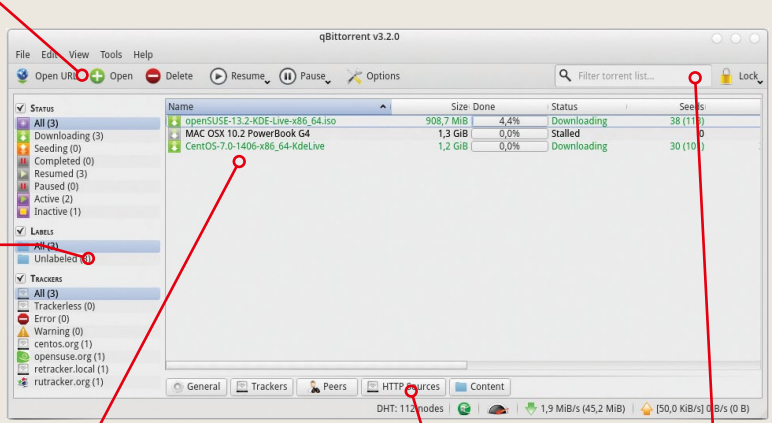
It has a classic user interface, with the current torrent downloads in the main view and a functional sidebar with status, labels and a trackers list (which is a recent feature). The simplest way to use *qBittorrent* is to offer it a .torrent file, either by selecting a downloaded local file, or pasting an URL, and you can do both actions from the application toolbar. However, *qBittorrent* offers many features for advanced users. For instance, it supports almost all BitTorrent extensions, including DHT and Magnet/BitComet links, IPv6 addresses, uPnP and NAT-PMP forwarding (if you're behind a router), IP filtering, Ajax-based web interfaces, torrents sorting, queuing and more. The handy torrent search feature is very welcome as it enables you to make search queries across a wide range of supported trackers right inside *qBittorrent*. Users can also create new torrents themselves, thanks to the built-in wizard (found under the Tools menu).

> **"Can be configured for continuous downloading instead of simultaneous."**

The new 3.2 version brings various fixes and improvements to the GUI, which can now be compiled against the *Qt5* library (Note: the version in the screenshots is still on *Qt4*). Downloaded files are highlighted in *Dolphin*, *Konqueror* and *Nautilus* now and there's also a new 'Finished' status for downloads. There's also a new .service-file for *Systemd* (for using *qBittorrent* in noX mode), which supports downloading files into network locations and tons of various fixes. The application fits wonderfully into any desktop environment thanks to its streamlined *Qt4*/*Qt5* interface and clean design. It can also be configured for continuous downloading instead of simultaneous mode, which is of great benefit for any user who is on a low-speed connection.

Getting *qBittorrent* is pretty easy – just look at your distro's repositories and find the right package. There are Ubuntu and Mint packages in the dedicated PPA (**qbittorrent-team/qbittorrent-stable**). You can also surf OpenSUSE OBS or Arch's AUR to find *Qt5* builds for a seamless installation.

Exploring the qBittorrent interface



Essential control tools
The toolbar hosts frequently used tools that apply to the currently selected torrent.

Sidebar
The Status and Labels category have been here before, but the trackers list has just arrived.

Main grid
The Mission Control Centre supplies all the information you need to track your downloads.

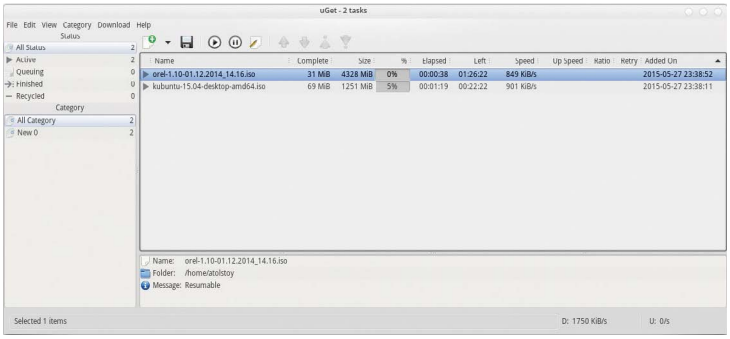
Extra features
The bottom pane offers some indicators and useful buttons for exploring network connection and torrent contents.

Search field
If you have lots of torrents running at once, you can filter them with keywords and category-specific entries.

Download manager

uGet

Version: 2.0 **Web:** http://ugetdm.com



› Control your downloads pool with tags and eliminate any disarrangement.

Here's another application which serves as an excellent download client. *uGet* is a cross-platform and open source download manager. Compared to *qBittorrent*, it has a whole host of distinguishing features and an entirely different focus.

While BitTorrent is a standalone technology combining tech and social/sharing elements, a regular download manager is a less social beast and more technical. Before the BitTorrent era – when broadband Internet connections were in their infancy and Linux users were fiddling with **/etc/ppp/options** – download managers were extremely relevant for everyone who needed to control data usage and special timeframes with lower data rates. Today those issues are still relevant for those that have to use a mobile broadband paid subscription. So if you frequently need to download a lot of massive files, your standard web browser isn't the best solution, but a standalone app can do the job better.

uGet is a lightweight GTK3-based download manager with advanced features. It can put files in a download queue, pause and resume downloads and has advanced category management. *uGet* also has *FireFox* Integration via a FlashGot extension, clipboard monitoring (it looks for URLs there) and batch downloads. The client supports a host of protocols too, including HTTP, HTTPS, FTP, BitTorrent and Metalink. It can be controlled remotely using the command line or work without notifications in Quiet Mode and use Proxy credentials and limit the number of connections and more. Downloads can be organised into categories, which are helpful for filtering items when you feel you have too many of them.

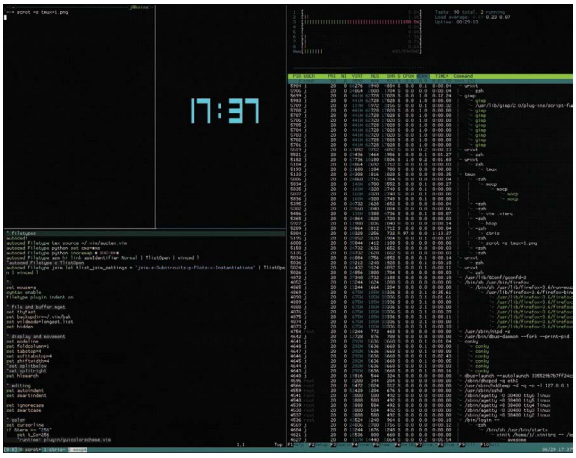
> **"uGet is an advanced and mature application from the GTK3 world."**

uGet would be a good default download manager for those who need an advanced and mature application from the GTK3 world. The application is also available on a broad number of Linux distributions, as well as other systems, including FreeBSD and Android for mobile devices.

Terminal multiplexor

Tmux

Version: 2.0 **Web:** http://tmux.sourceforge.net



› Advanced terminal multiplexor helps get rid of GUI.

The new *Tmux 2.0* was recently released after over a year of development. *Tmux* is a terminal multiplexor developed by the OpenBSD community as a more advanced and flexible replacement for *Screen* [covered in *Escape the GUI, p52*, **LXF197**]. *Tmux* lets you split your terminal into several instances to create a multi-tasking environment, which makes it ideal for command line junkies.

Terminals can be created, accessed and controlled from a single screen so that each *Tmux* instance can be detached from a screen and continue running in the background, then later reattached. When *Tmux* is started it creates a new session with a single window. A status line at the bottom of the screen shows information on the current session and is used to enter interactive commands. *Tmux* can be controlled from an attached client by using a key combination of a prefix key, Ctrl+b by default, followed by a command key. For example, % key splits the current pane into two: left and right, and 0-9 keys select windows 0 to 9. To explore the full list of keybindings, consult the dedicated section of the manual with `man mux`.

A session is a single collection of pseudo terminals under the *Tmux*'s management. Each session has one or more windows linked to it. A window occupies the entire screen and may be split into rectangular panes, each of which is a separate pseudo terminal (the pty(4) manual page documents the technical details of pseudo terminals). Any number of *Tmux* instances can connect to the same session, and any number of windows can be present in the same session. Once all sessions are killed *Tmux* exits.

> **"Tmux lets you split your terminal into several instances."**

In version 2.0 some options have been cleaned or transferred from session to server side (such as 'terminal-overrides' or 'message-limit'), plus it has better UTF-8 support. *Tmux* is in dozens of Linux distros, so getting it is very easy while mastering the 'right' configuration can take a lot longer.

Configuration tool

Qt5CT

Version: 0.12 **Web:** http://qt5ct.sourceforge.net

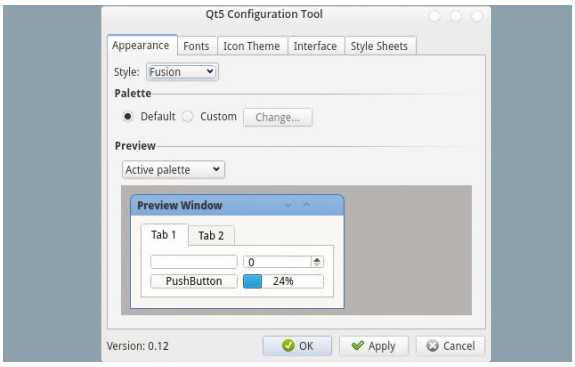
When a GUI framework upgrades to a major version, one of the usual results is an inconsistent visual mess. We recall Gnome's transition from *GTK2* to *GTK3*, and KDE throwing away its 3.x series, in 2008-2009, and going with its brand new KDE4.

Now it's time to make the transition again, as you're reading this many Linux vendors and package designs will be gearing up for the new Plasma 5 desktop which replaces 4.x. This also means that more and more *Qt*-based applications are moving to the latest *Qt5* version and sooner or later you'll get a bunch of different toolkits that will make your desktop applications look alien. To circumvent the issue, the *Qt* team has rolled out *Qt5CT*, a tool that enables users to configure *Qt5* settings (themes, fonts, icons, etc) under a desktop environment/window manager without *Qt* integration. *Qt5CT* is helpful not only in Gnome/Unity/Cinnamon and the rest, but also in KDE 4, which doesn't care a lot about *Qt5* styling.

The application has six tabs for changing widget appearance, fonts, setting icon theme, GUI effects and applying custom style sheets. By default, *Qt5* applications use the Fusion style and follow KDE4 fonts and colouring, but all this can be changed. In addition, *Qt5* accepts custom style sheets, eg the following style sheet specifies that all QLineEdits should use yellow as their background colour, and all QCheckBoxes should use red as the text colour:

```
QLineEdit { background: yellow }
QCheckBox { color: red }
```

> "configure Qt5 settings under a DE/WM without Qt integration."



› The *Qt5* configuration tool has a professional look and supplies a rich feature set.

Almost every style element is supported, so you can tweak and play with it as you wish, once you've studied the syntax beforehand (see **http://doc.qt.io/qt-5/stylesheet-syntax.html**). On some systems *Qt5CT* may not produce any changes, which means that some important variables override its work. You can fix this by entering the following lines in your **~/.bashrc** file:

```
QT_STYLE_OVERRIDE=gtkexport
QT_QPA_PLATFORMTHEME=qt5ct
```

Qt5CT is available in Arch AUR, Rosa and OpenSUSE, but strangely not in the Ubuntu family, where you're supposed to install the application from source and therefore you will need to install the Qt5 developer packages and a proper build environment.

Drawing software

Krita

Version: 2.9.4 **Web:** https://krita.org

Krita has been staying in the shadows of the less well-known office suite *Calligra* (formerly *KOffice*), and while most suites have powerful word processor and spreadsheet components as their flagships, *Calligra*'s strongest and most professional component is *Krita*, which is raster graphics software with a focus on digital painting and drawing (Krita means 'chalk' in Swedish).

Many users have tended to compare *Krita* with *Gimp* and *Adobe Photoshop*, but it has always been an artist's application, with strengths in brushes and drawing tools and a very modest set of effects and filters. In recent years, *Krita*'s development has had a significant boost and a lot of bottlenecks have been eliminated, particularly low 2D performance and random crashes. The 2.9.x series has become more impressive with every incremental release, and here's what we get this time.

The most anticipated feature in *Krita 2.9.4* is the support of *Photoshop*-like layer styles. Yes, *Krita* can now add drop shadows, inner/outer glow, bevel, emboss and lots of other skeuomorph effects to its layers and, of course, it renders third-party .PSD files much better now.

Other changes in this version are mostly bug fixes and performance improvements. *Krita* now starts a lot faster by not waiting for the presets to be loaded (shaving seconds of the launch time), plus masks and filters perform better as well as some tools involved in overlay effects, which culminates in a smoother painting experience. Of course, there's still a lot to be done; the *Krita* developers note that currently all application features can be stored in Krita's native file format only (not PSD yet). Some bugs with masks and layers will also be fixed in next minor releases.

> "The most anticipated features is a Photoshop-like layer effect."

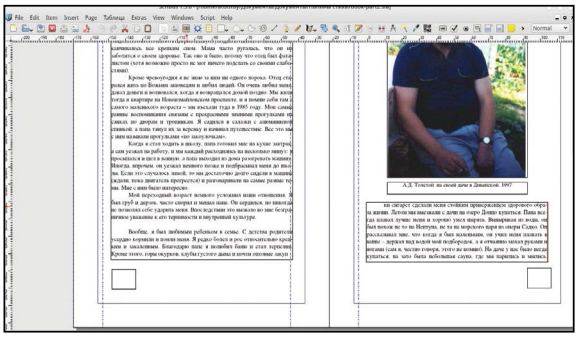
› Krita uses a dark default theme intentionally, as it is meant to help you concentrate on the content.

Getting *Krita* should be quite straightforward, there's a PPA for Ubuntu and the rest of its kin (**ppa:dimula73/krita**), and most other Linux distributions provide *Krita* packages together with the rest parts of the *Calligra Suite*.

Publishing software

Scribus

Version: 1.5.0 **Web:** http://www.scribus.net



› Scribus is finally beginning to not only look but also feel like a professional DTP suite.

Designing posters, booklets and generally all other prepareed print tasks has been an area where Linux has excelled for years. Despite this there are still very few applications that can do the job on a professional DTP level, but there are a number suitable for school projects and semi-professional publishing. Since early 2000 *Scribus* has been the most advanced DTP software of the crop, although it's still running behind *Adobe Pagemaker* of the 90s, but closing the gap in some aspects.

Scribus has hit **LXF** pages several times, [the last time being in *HotPicks, p58,* **LXF187** with version 1.4.4], but now it's time to unveil the new 1.5 version, which is a result of many years of work and more than a thousand bugfixes. Though the developers claim *Scribus 1.5* isn't stable enough for use in real production scenarios yet (while the future 1.6 is planned as a stable milestone release), in a non-critical home environment it worked very well for us without any glitches or errors.

Among the most important technical changes is the transition to the modern *Qt5* toolkit, a significant overhaul of the *Scribus* internal SLA file format, and integration with UniConvertor (we recommend installing it beforehand) for importing more vector file formats. The user interface has also been lifted with a redesigned Document Setup/ Preferences dialog and many menus now occupy less screen space.

Scribus 1.5 has also learned to embed bitmap graphics right inside its SLA format (previously they could only be linked), move several objects without grouping them and also clone master object's properties to its children. These are all features which are common in many high-profile commercial vector graphics software, although *Scribus* tends to take *Adobe Illustrator* as its guiding reference.

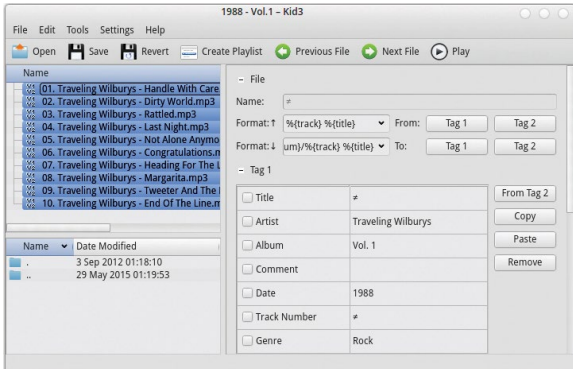
> **"An important technical change is the transition to the Qt5 toolkit."**

Scribus is available in almost any Linux distribution, though not all of them offer the latest version. If you can't find the pre-built package follow the instructions on the *Scribus* wiki to build it from source. *Scribus* has excellent documentation and a very decent compiling manual for different platforms, even for such cool things as the Haiku OS (*see Roundup p24*).

Audio tool

Kid3

Version: 3.2.1 **Web:** http://kid3.sourceforge.net



› Do away with a disorganised music library on your hard drive – use *Kid3* to wrestle with your metadata.

Collecting a vast music library on a local hard drive may seem old hat for some users in a world that has embraced streaming media so completely, but it's a way to stay in control of our music.

But while your Music folder grows, it tends to get harder to maintain directory structure and file-naming conventions. MP3 files, along with other competing formats such as OGG or AAC, store track metadata right inside a file. That's the reason why files in a directory can look accurate and quite self-descriptive, but once you add them into a media player database, they abruptly become disorganised.

Track metadata is presented in the form of tags, either ID3v1 (with a fixed set of fields of limited length) or ID3v2 (which is fully dynamical structure, not only allowing arbitrary field sizes, but also arbitrary field types).

Kid3 handles file tags and the application has a vast area on the left to show loaded audio files, a mini-file-manager at the lower-left part and a metadata editor on the right. The tool supports batch tagging of multiple MP3, Ogg/Vorbis, Opus, DSF, FLAC, MPC, MP4/AAC, MP2, Opus, Speex, TrueAudio, WavPack and WMA files (eg full albums) without typing the same information each time. This is accomplished by using wildcards in the Format fields, which lets you define the source and destination tag structure.

Once you get used to *Kid3*, you'll be able to generate tags from filenames and vice versa, automatically convert upper and lower cases and replace strings and generate playlists etc.

> **"You'll be able to generate tags from filenames and vice versa."**

Kid3 also has some helpful extra features, such as the ability to import metadata from **freedb2.org**, MusicBrainz, Discogs, Amazon and other sources, editing synchronised lyrics and event timing codes.

Though the version number of *Kid3* doesn't seem to be that progressed, the application has existed since 2003 and can be found in most Linux distributions or at project's official site, while the *Kid3* Sourceforge page of has a number of Ubuntu packages.

HotGames Entertainment apps

Turn-based strategy

Tanks of Freedom

Version: Beta3 **Web:** http://bit.ly/TanksOfFreedom

› No looting and civic disorder, but good ol' combat between armed pros.

Lovers of pixel art and indie games may just find *Tanks of Freedom* a lovely little turn-based gem. The idea is to push forward with your tanks, support with your helicopters, and claim strategic buildings with your infantry.

Each level is a part of urban landscape with roads, buildings and different obstacles. Each level sets the battle between two players with armies made up of several infantry units. The goal of the game is to take control of each other's bunker first.

A player can choose either simple or complex strategy. If you don't have much time *Tanks of Freedom* can be played like a logical board-like game where you calculate the safest and fastest path to the opponents bunker. However, it's much more fun to use tanks and aircrafts. When your units take control of plants (there are several types), you can spend money to produce various heavy weapons. A helicopter beats a tank, while three foot soldiers beat a helicopter. Tanks also have the longest cruising endurance if you use paved roads. A player has to keep balancing the different fighting elements of their army, because only infantry can occupy buildings, while vehicles provide higher fire power.

We have to admit that being tiny and super-simple to play, *Tanks of Freedom* is an easy time-killer with mature game play and lot of fun. The game is distributed as a BIN file, which you'll have to make executable and launch from the terminal like so:

```
$ chmod +x ToF_Beta3.071_
linux64.bin
$./ToF_Beta3.071_linux64.bin
```

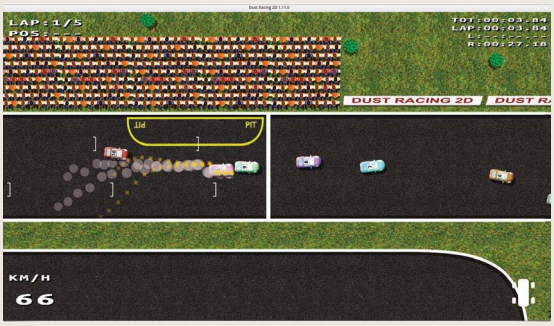
The game is lightning fast and needs about 33MBs of hard drive space, and it will run just fine on any machine that's capable of using SDL-powered games.

> **"Tanks for Freedom is an easy time-killer with mature game play."**

Racing simulator

Dust Racing 2D

Version: 1.11 **Web:** http://bit.ly/DustRacing2D



› It takes many attempts to leave those AI bots for dust.

There are plenty of racing simulators that use the first-person view, for instance we have *Project CARS* to look forward to later this year, but *Dust Racing 2D* is totally different. The game has eye-catching visuals, yet they don't pretend to be true to life and are rather cartoonish.

The game offers an arcade-style set of levels, of which only the first one is available straight away from the start. In order to unlock the next one, you'll need to finish in the top six. So what does it take to qualify? Well, the game control is extremely hard as it only uses the four keyboard arrows (or any other keys that you define in Settings), which is very difficult to get used to when you're racing against 11 AI-driven bots. The other reason is the top-down view: your accelerating key always remains the same (Up arrow by default), while the side keys act differently depending on your position on the track and your vehicle's direction. You might find it nightmarishly difficult at the beginning, as the car's handling is very slippy, but after a while you'll achieve the appropriate level of twitch response to make it round corners without skidding off the track. You can't cheat and do shortcuts as this incurs extra penalty laps. After the first callous appears on your fingers, you'll likely have perfected avoiding trees and gracefully pass hard turns and get on to the next level.

Dust Racing 2D has evolved on recent years. It starts in gorgeous full-screen mode, plays catchy music, has more logical difficulty settings, better textures and doesn't crash anymore (in our experience). There are four game modes: for one or two players (sharing a keyboard), a time trial and a duel. Lap count can also be altered to your preference.

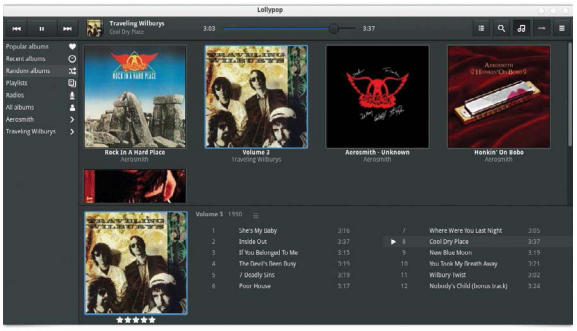
The game is widely available for many sorts of Linux distros, including Mageia, Ubuntu, Debian, OpenSUSE, Arch and lots of others. *Dust Racing 2D* relies on SDL stack and will run on any OpenGL-capable system.

> **"It starts in gorgeous full-screen mode and plays catchy music."**

Music player

Lollypop

Version: 0.9.20 **Web:** http://bit.ly/LollipopPlayer



› **It's a nice idea to start a party with a single mouse click.**

No, this is not mobile operating system, Android 5.0, but a music player with a very similar confectionery-based name. It's designed to fit just right into Gnome or any other *GTK3*-based environment with its clean design and large spaces between controls. *Lollypop* uses parts of *Totem* for parsing tracks metadata, creating playlists and playing from versatile media sources.

When the application is started for the first time, it automatically adds your **home** directory to its database and starts rustling your hard drive, so expect to wait for awhile if your music library is very large. *Lollypop* manages your music collection with a focus on dynamically updated playback statistics. On the left pane, you'll find a set of categories, such as Popular Albums, Recent Albums, Random Album, as well as static All Albums, Radios and Playlists.

The last two are placeholders for user-created content as *Lollypop* doesn't come with a predefined set of network radio sites. *Lollypop* also benefits from modern *GTK3* features, which enable you to change its colour scheme on the fly.

There's also the Party mode button, which shuts the lights down and turns the *Lollypop* window a dark shade of grey. The Gnome guidelines advise developers to use auto-lookup of the music library in the background, but *Lollypop* also has a dedicated Update music button (found in the Tools menu, in the far-right icon of the toolbar).

Album art is another feature that's supposed to be maintained manually, just select an album in any of the appropriate categories and then left-click its cover area. *Lollypop* will fetch the album cover art and will usually let you choose one from several variants.

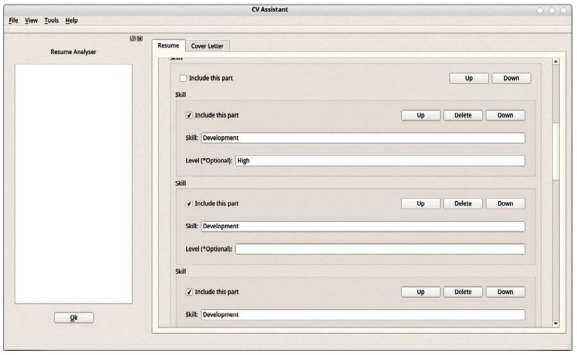
> **"Adds the home directory to its database and starts rustling your drive."**

Despite being simple to use and not excessively overloaded with features and options, *Lollypop* respects users' control of their music collections. Give it a try on your stylish *GTK3*-based desktop, and maybe you'll find that *Lollypop* will fit in better for you than *Gnome Music* and several of the other players. Lollypop is available pre-packaged for Arch, Fedora, OpenSUSE and Ubuntu/Debian. See the project's website for details.

Business application

CVAssistant

Version: 2.0.0 **Web:** http://bit.ly/CVAssistant



› **Take a more professional approach to finding a job.**

Feeling a bit punch drunk from the job knock backs? Don't give up and here's a handy app to help you with all those job application. It's no great secret that most people have to maintain more than one profile covering professional jobs in neighbouring fields, as HR departments seem oblivious to the fact that one person could do more than one type of job. This does mean that a person has to prepare several variations of their CV with different details to fit a different job profile or a specific vacancy better.

CVAssistant is specialist piece of software that helps you create custom resumes in *Word* DOCX format with less effort. The idea is to have a master CV with all skills and experiences in it, and then based on skills mentioned in a job ad, export a clean but well-formatted DOCX file as a summarised resume with only relevant skills in it.

It's definitely a useful thing, because most companies use Applicant Tracking Software (ATS) to match their job advertisement with incoming CVs, so using the right phrases in your copy can boost your chances of an interview.

CVAssistant is a *Qt*-based application that enables you to build your master CV, rearrange its fields, add your profile photo and select which fields you want to export to the target document. The most useful feature of *CVAssistant* is the Analyzer feature that compares your master CV with keywords from job advertisements and let you know if your CV matches them or not. Besides the main Resume tab, there's also an auxiliary Cover Letter tab, where you're supposed to write your eye-catching approaches.

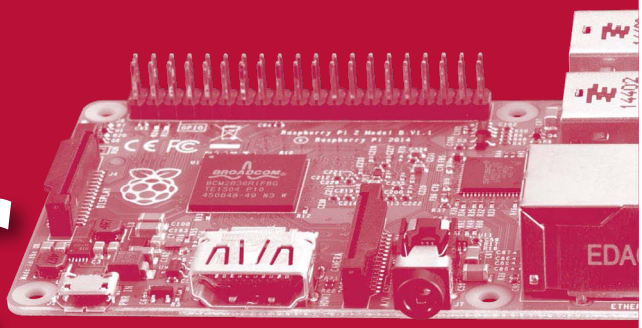
> **"Specialist software that will help you create custom resumes."**

When you finally get all fields filled correctly, press Ctrl+e to export the CV to DOCX format and Ctrl+Shift+e to export the cover letter (saved in plain text). The master CV is saved separately in the application's internal CVA format, which is XML-based. Currently *CVAssistant* isn't packaged for any distro, but it's simple to build, eg in Ubuntu, install dependencies with

```
$ sudo apt-get install qt5-qmake
libqt5-dev
```

and then build it with the `$ qmake && make` command. LXF

LINUX FORMAT
Pi user



Giving you your fill of delicious Raspberry Pi news, reviews and tutorials

BEN NUTTALL is the Education Developer Advocate at the Pi Foundation.

Welcome...

Here at the Raspberry Pi Foundation at our office in Cambridge, affectionately known as Pi Towers, we've been busy working on a number of projects in hardware, software and education.

You may have heard, we're sending two Pis to the International Space Station (ISS) at the end of the year for some space exploration educational outreach activities, as part of British ESA Astronaut Tim Peake's Mission Principia. While we've been completing the Python API for the sensor board and writing learning materials, the Model B+ Pi and its SenseHAT, are going through safety testing for flight certification, and we're in eager anticipation of getting the green light from ESA to say we're safe for space deployment.

Tim Peake will return to Earth after six months but the Pis will remain. We have no plans for them just yet, but let's see what 2016 and 2017 hold – hopefully more opportunities for school pupils around the world to run their code in space!

We've just installed an exhibit in the Foundation of Art and Creative Technology (FACT) in Liverpool as part of the Build Your Own exhibition. Neurotic Machines, the brain child of the Foundation's Creative Producer Rachel Rayns, comprises a vertical garden, a plant pot jukebox and a control panel planet simulator full of analogue dials and potentiometers, with a giant button and 10x10 dotstar pixel matrix thrown in. We had great fun putting it together and it looks beautiful. It's on display in Liverpool for three months and at Norwich Castle the following three months: **http://bit.ly/LXFneuroticPI**

Picadamy USA

The Foundation announces that it's coming to America – big plans are afoot!

Picadamy, the Raspberry Pi's Professional Development Program, has been a huge hit here in the UK, having trained around 200 teachers through just eight events. The Foundation wants to train hundreds of thousands of educators and to achieve that level of results, the program is extending its reach and heading to the USA.

The Raspberry Pi Foundation has teamed up with the Computer History Museum, Mountain View, California (**www.computerhistory.org**) to pilot a series of Picadamy USA courses in early 2016. The first course to be held in the States will be spread over two days, with 25 teachers getting a chance to see how the Raspberry Pi can be used in a classroom setting, and at the end of the course the teachers will become Raspberry Pi Certified Educators. The first courses will be held around the US, with a goal of training around 100 teachers, and if you're interested in taking part head over to **https://www.raspberrypi.org/picademy/usa** to sign up.

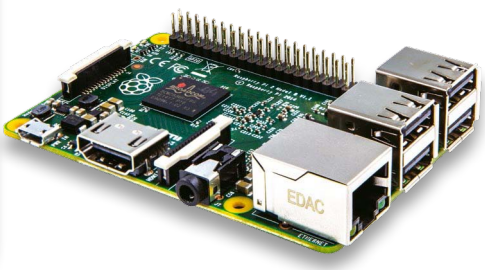
Picadamy in the UK is still going strong, so if you're a teacher here and would like to know how to get involved, make sure you go to the official Raspberry Pi website (**www.raspberrypi.org**) to find out more. No prior experience with Raspberry Pi is needed, just a desire to learn and an interest in broadening your students' horizons.

The Pi rated No.1 and 3 for hackers

The people's Pi.

Two popular sites, **LinuxGizmos.com** and **Linux.com** teamed up to ask readers what their favourite three Linux or Android-based open-spec single-board computers were, and with 1,721 respondents there was a clear winner: the Raspberry Pi 2 Model B.

Perhaps even more impressive than winning the readers' choice is that the Model B wasn't the only Pi in the top 3. While second place went to the Beaglebone Black, the Raspberry Pi Model B+ chalked up third place. Check out the survey results at **http://bit.ly/HackerSBCSurvey2015** for the full list of winners.



Fancy a Pi-powered laptop?

Build your own handheld.

Adafruit's Mini Raspberry Pi Handheld Notebook is a compact handheld device. It features a 3D-printed enclosure, a 3.5-inch touchscreen display and a 2,000mAh battery. If you don't want the screen to be covered in mucky finger prints, there's also a mini chiclet keyboard and a trackpad for controlling the device.

Adafruit suggest that it's great for checking webcams, streaming music and playing text adventures in-between monitoring prints. For more details, including how to build one yourself, including software images and blueprints, head to: **http://bit.ly/RaspPiHandHeldNotebook**.

AstroPi SenseHAT

Space, the final frontier. These are the voyages of the 'USS Enter-Pi-ise', its mission to educate and innovate. Captain **Les Pounder** takes the helm.

In brief...

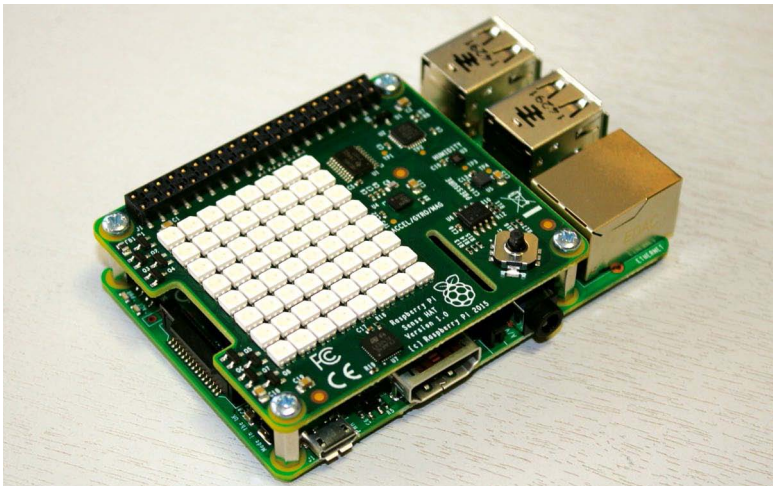
» The latest add-on board from the Raspberry Pi Foundation. It has a plethora of sensors for use in scientific projects. The board is part of a project that enables children to conduct experiments on Earth that will be replicated by British astronaut, Tim Peak on the International Space Station in late 2015.

The Raspberry Pi has travelled across the sea, powered robots, taught entire classes and is now set to reach into space with the AstroPi project. This is a science project where schools from across the United Kingdom are being asked to devise experiments which, if chosen, will be run on the International Space Station (ISS) by ESA astronaut Tim Peake.

AstroPi is also the name for the hardware platform designed for use with the A+, B+ and Raspberry Pi 2 boards (although it can work on all Pi models). The dimensions of AstroPi enable it share the same form factor as the A+ and fit neatly on top of the B+.

The AstroPi board is a data capture and science platform that offers an extensive suite of sensors. For starters, it has a temperature and humidity sensor and next to this we have a pressure sensor. The final sensor combines an accelerometer, gyroscope and magnetometer for positional data. Not content with providing sensors, AstroPi also provides an 8x8 grid of multi-colour LEDs (neopixels) and a simple joystick input device. The LED matrix is exceptionally bright (and a diffuser such as a piece of paper should be used to protect your sight).

But hardware is nothing without software, so AstroPi uses a Python 3 library, which enables easy access to the sensors and other components of the board. Each component comes with its own function to handle the collection of data, making it very accessible for children to use for hacking, eg we tested the temperature sensor and were able to output the temperature to the LED matrix, while controlling the colour of the LEDs based on temperature in 20 minutes of hacking – very easy!



› The SenseHAT is the correct name for the AstroPi sensor HAT. It's designed for the A+, B+ and Raspberry Pi 2.

AstroPi can be used in many different applications: from simple temperature logging to complex scientific experiments to monitor air pressure at altitudes using balloons or drones. In a creative application, eg AstroPi could be easily linked to MinecraftPi to use the sensors to control the game world or the LED matrix could be used to direct a player to a hidden location.

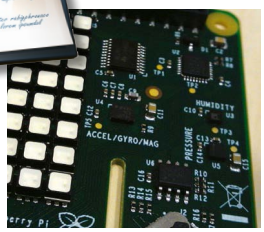
Pi me to the moon

Of course, the AstroPi's destination is the ISS, where a series of experiments will be conducted created by children in the UK as part of a national competition. The code and data generated by the experiments will be shared with every child in the world and they can recreate the experiments and compare their results to that of Tim Peake's. This will be the first time that this kind of project has ever been attempted, which is very exciting. With sensors, such as the pressure and temperature ones, we'll clearly see in the results how pressure and temperature on board the space station fluctuates through an orbit of the Earth. It's going to be fascinating to see what experiments can be devised by children and to try these ourselves.

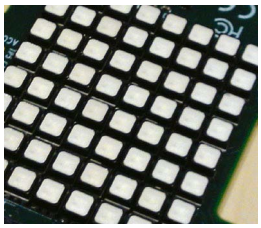
The Raspberry Pi Foundation, Airbus and the European Space Agency have worked together for just under a year to design, test and certify the board for flight to the ISS where it will be connected to a Raspberry Pi B+ to conduct the experiments, The B+ model was chosen as it takes time to certify a device. The ISS has strict safety measures, particularly as the Raspberry Pi can't be powered from the station's main power and will have to be powered via a laptop USB port.

This, then, is a fantastic board full of great potential for use in education and for hobby hackers. LXF

Features at a glance



Lots of sensors
Some of the sensors handle multiple functions, such as the joint temperature/ humidity sensor.



LED matrix
The 64-LED matrix can be used to output data from the sensors, by forming words or using colours.

LINUX FORMAT Verdict

AstroPi SenseHAT

Developer: Raspberry Pi Foundation
Web: www.raspberrypi.org
Price: TBC (around $35/£26)

Features	9/10
Performance	9/10
Ease of use	9/10
Value	9/10

» *The ease of use, matched with the selection of sensors creates an easy point of entry for scientific discovery.*

Rating 9/10

Pigame: Build a Buzzer game

Les Pounder takes inspiration from Blue Peter (but with slightly less sticky-back plastic) to build a classic fairground game using the Raspberry Pi.

Our expert

Les Pounder is part of the Raspberry Pi Foundation's Education team and loves hacking projects into life with the Pi and lots of blutack. He has a blog at **http://bigl.es**

The term 'maker' has slowly become a common word in mass media. It now embodies a spirit of tinkering and hacking that has been brought to the mainstream by single board computers, such as the Raspberry Pi and microcontrollers like the Arduino. Using these devices we can now build any amazing array of projects: from simple blinking LEDs to a bipedal robot friend. But what can we make with some arts and crafts materials and a few cheap components?

In this project we'll build a steady-hand wire game. The game will require you to guide a metal hook through a copper wire course without touching the wire at any point. If we do touch it we lose one of our three lives, and losing all of our lives will trigger game over.

For this project you'll need: any model of Raspberry Pi, the latest Raspbian release, a plastic takeaway tray, modelling clay or sugru, a breadboard, three LEDs, four 220 Ohm Resistors (RED, RED, BROWN), one buzzer, 16 jumper cables (11 male to female, 1 female to female and four male to male), one pushbutton, a 3.5mm speaker or a monitor with speakers if you decide to use HDMI, and, of course, a bare metal paperclip for the hook.

We start by making sure our Raspberry Pi is turned off and then connect the components as per the diagram [*see opposite*]. A high resolution version of this diagram can be found at **http://bit.ly/LXF200WireGame**. This project uses the latest Raspbian release which now features an enhanced desktop environment. Under the hood, one of the key Python libraries for this project, pygame, has been pre-installed ready for use with Python 2 and 3.

IDLE and libraries

So lets start our project by first opening a terminal, the icon for which is in the top left of the screen and looks like a monitor with a black screen. With the terminal open run the following command to open IDLE3 as root:

```
$ sudo idle3 &
```

The `&` is used to background the command and return the terminal prompt control to the user. We should now see the IDLE3 application on screen. By default IDLE3 will open with a shell window, this is where we can test logic, write small programs and the shell will return the answer instantly. The shell is also used as a method of output when running larger projects, such as what we will create. We don't want to be in the shell so click on File > New Window to open a blank document. In this blank document we will write the code for the project but it won't instantly return the output of code as we need to save and run the code to test it. At this time save the blank document using File > Save and call it **wire.py**. Remember to save your work often to reduce the chances of losing data.

› Our goal is to build a classic fairground game of skill where we must navigate a wire without touching it. If we touch the wire then we lose a life and trigger the Raspberry Pi to play audio and display a warning image.

We start our Python coding by importing three libraries:

```
from time import sleep
import RPi.GPIO as GPIO
import pygame
```

First we import a single function, `sleep`, from the time library which is used to delay our code in certain key points. We import in this manner as we only require one function from the library and importing all of the library would be inefficient. Our second import is the Python library that enables our project to communicate with the 40 GPIO (General Purpose Input Output) pins of the Raspberry Pi. Our last import is the `pygame` library which we will use to play music and show pictures on screen.

In order to use pygame we must first initialise both the main pygame library and the audio mixer function and this is handled via just two lines of code.

```
pygame.init()
pygame.mixer.init()
```

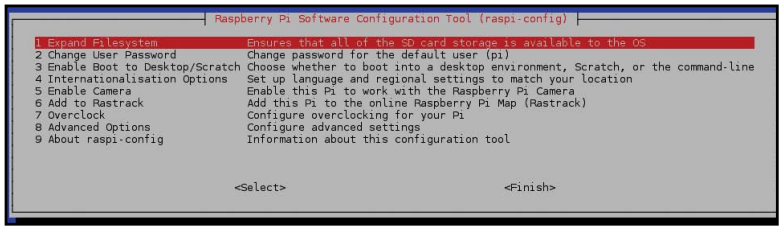
Raspi-config

For those new to Linux tinkering with configurations and settings can be daunting but thanks to the work of Alex Bradbury there's a tool to help. *Raspi-config* is a suite of tools to tinker with your Raspberry Pi. It can be used to configure most aspects of your Raspberry Pi such as overclocking, enabling SSH access and routing your audio to either the 3.5mm socket or the HDMI port. To use *Raspi-config* you will need to open a terminal window and type `$ sudo raspi-config`

The *raspi-config* menu is split into nine groups, with further options contained in Advanced Options. To control where the audio is routed navigate to the 'Advanced Options' using the cursor keys and press Enter. Inside the Advanced Options menu look for the 'Audio Options' entry, navigate to it and press Enter.

In the next menu you will be asked to select where the audio is routed. If you have connected via HDMI and your monitor/TV has speakers you can leave the default setting. But if you are using a 3.5mm speaker then you will need to select 'Force 3.5mm jack' and press Enter to enable.

Once complete navigate through the menu to exit *raspi-config*. If you play any audio it will now play via the 3.5mm headphone jack.



Now our focus shifts to setting up the GPIO ready for use and to do this we must instruct the Pi which GPIO pin layout we will be using. There are two common layouts:

» **GPIO.BOARD** This uses logical pin numbering, with two columns: one for odd and the other for even pin numbers. Pin 1 is the top left pin located near to the microSD card and pin 2 is directly next to it on the right.

» **GPIO.BC** The standard GPIO pin layout supported by the Raspberry Pi Foundation resources. This breaks out the connections from the Broadcom System on a chip (SoC) to the GPIO in an efficient order for the chip.

For this project we shall be using the GPIO.BCM format:

```
GPIO.setmode(GPIO.BCM)
```

With the GPIO pins setup we now move on to configuring the pins that will be used in the project. We will be using two inputs and four outputs. Our first input is connected to pin 2 of the GPIO and is intentionally set to a high state so the pin is active with power flowing to it. We repeat this setup for pin 21 only requiring us to change the pin number referenced.

```
GPIO.setup(2, GPIO.IN, GPIO.PUD_UP)
GPIO.setup(21, GPIO.IN, GPIO.PUD_UP)
```

We now move on to configuring the four output pins these are 14, 15, 18 and 24. Pins 14, 15 and 18 will be connected to an LED to create our life counter and 24 will be connected to a small buzzer to indicate that the wire has been touched.

```
GPIO.setup(14, GPIO.OUT)
GPIO.setup(15, GPIO.OUT)
GPIO.setup(18, GPIO.OUT)
GPIO.setup(24, GPIO.OUT)
```

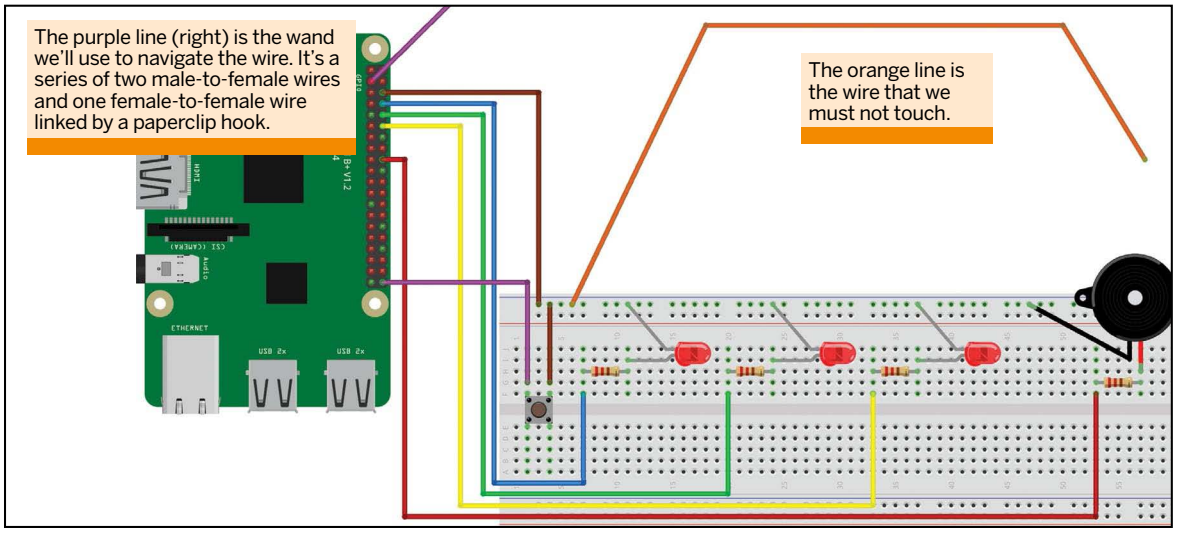
All of the pins for this project have now been configured and set up for use and now our focus shifts to creating two functions. The first function controls the number of LED that are illuminated, in effect this is our life counter. Every time we touch the wire a life is deducted and that value is compared against the values coded into this function.

Game functions

We start the function be giving it a name and then in brackets we add an argument. In this case the argument is the number of lives that we have left:

```
def life_counter(i):
```

With the function named we press Enter and the following »



› **None of the components for this project are soldered enabling us to re-purpose the components. We use connectors and a breadboard to create a semi-permanent method of joining components.**

» code is indented four spaces. This is how Python indicates that the code belongs to the function. We will create an if.. elseif conditional that compares the number of lives passed via the argument against four values 3,2,1 and 0. If the number matches any of these numbers then the correct number of LEDs are lit up using the GPIO.output function to change the state of the pin from off, 0 to on 1. Our first condition to test is a comparison of the number of lives against the value 3, if that is true then all of the LED are lit.

```
if i == 3:
    GPIO.output(14,1)
    GPIO.output(15,1)
    GPIO.output(18,1)
```

If the first condition is false then we go on to compare the number of lives against the value 2 using an else if condition, Python refers to this as elif. In this example we can see that two of the LED are lit, 14 and 15, whereas 18 has been turned off, denoting that we have two lives remaining.

```
elif i == 2:
    GPIO.output(14,1)
    GPIO.output(15,1)
    GPIO.output(18,0)
```

We further repeat the else if conditional statement against having 1 and 0 lives on the counter. Turning off the correct number of LEDs to show our remaining lives. This marks the end of this function. We now move to our last function and this function is used to display images on the screen using the pygame library.

```
def picture(img,w,h):
```

We start the function by giving it a name, 'picture' and then create three arguments in the brackets. The first, img, is used to pass the filename of an image, w and h are used to pass the dimensions of the image to the function.

With the function named we press Enter and the following code is indented four spaces, this is how Python indicates that this code belongs to the function. We next create two variables (below) the first of which is called 'pic':

```
pic = pygame.image.load(img)
background = (255, 64, 64)
screen = pygame.display.set_mode((w,h))
screen.fill((background))
```

We use pic to shorten the long function used to load an image, we then create a second variable called `screen` to set the dimensions for the window that will contain the images. Next we create a tuple, a tuple is an immutable list. A tuple can not be changed in any way once it is created. Our one is called `background` and in there we store the values needed to specify a background colour for our window.

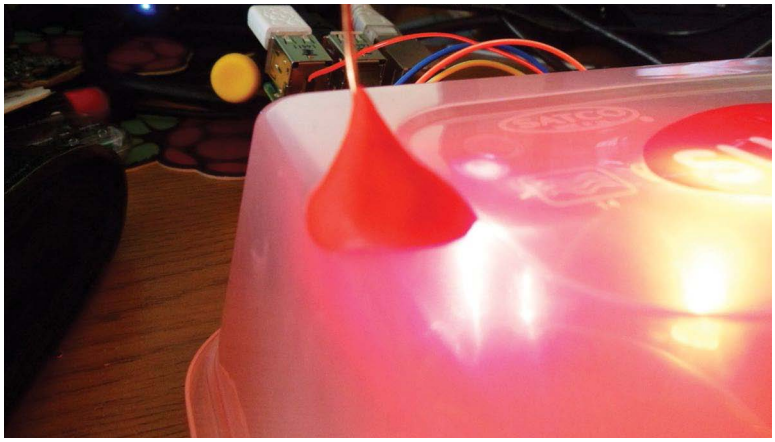
For the last part of this function we instruct pygame to show the image on screen using blitting:

```
screen.blit(pic,(0,0))
pygame.display.flip()
sleep(2)
pygame.display.quit()
```

This a fast method of displaying pixels commonly used in 2D video games. We then update the entire window using the flip function before waiting for two seconds and then closing the window.

Quick tip

We have created a Github repository for all of the code and media files used in this project. You will also find a series of high resolution diagrams illustrating the circuit diagram. All of this can be downloaded from **http://bit.ly/LXF200WireGameZIP**.



› Securing the wire is very important as even the slightest touch will trigger the alarm. We used sugru, an air curing rubber, to fix the wire in place and create insulation for the wire and a place to rest the hook.

Trying and toggling

We now move into the main body of the project and we start by using a try...except construction by adding `try:` on a new line, which, as its name suggests, will try the following code but if there any exceptions – and in this project we will later use a Keyboard Interrupt exception – it will stop the code and exit the project.

The next three lines create a variable called `toggle` and set it to be False:

```
toggle = False
picture('./pi.jpg',640,771)
while True:
```

We will use this to toggle that the user wishes to play the game. Next this calls the picture function that we created earlier and passes it to the arguments of the image that we wish to see and its dimensions. Last, we create the start of an infinite loop.

Our next section of code is an if condition that is looking to see if the button has been pressed to start the game:

```
    if GPIO.input(21) == False:
        lives = 3
        pygame.mixer.music.load('./fanfare.mp3')
        pygame.mixer.music.play(1)
        0
```

We set up an infinite loop in the previous code segment to constantly look for the button press by always checking the state of the button. Our button is linked to pin 21 (`GPIO.input(21)`), one of our input pins that has been set high at the start. When the user presses the button the state of the pin changes from high to low as the other side of the button is connected to ground and pulls the pin to low effectively turning off the pin. Once this happens code is executed to set the player's lives to three and then cue an audio file for playback, then play that audio once before changing the variable 'toggle' to True (`toggle = True`).

We now create a further indented section of code which is constantly checking to see if the status of 'toggle' is True:

```
        while True:
            if toggle == True:
                print("You have "+str(lives)+" lives left")
                life_counter(lives)
```

If that is the case then it prints out to the Python shell the number of lives that you start with and then lights up the correct number of LED using the `life_counter` function that we earlier created.

For our next section of code we are still inside the newly

created infinite loop and creating another if condition, this time looking for a change of state on pin 2 which is an input connected to the wand (paperclip):

```
            if GPIO.input(2) == False:
                GPIO.output(24,1)
                sleep(0.2)
                GPIO.output(24,0)
                pygame.mixer.music.load('./wrong.mp3')
                pygame.mixer.music.play(1)
```

if the state changes to False (low) then the buzzer attached to pin 24 will beep once for 0.2 seconds and then an audio file will be cued and played indicating that the player has touched the wire.

Elif in if

For the last chunk of code in the if conditional statement we'll trigger an image to appear on the screen reusing the `picture()` function that we created earlier.

```
                picture('shockdanger.jpg',724,634)
                lives = lives - 1
                life_counter(lives)
                print("You have "+str(lives)+" lives left")
                sleep(3)
```

Then we shall deduct one life from our 'lives' variable (`lives = lives - 1`), show the number of lives using the LED `life_counter(lives)`) and then `print` the number of lives to the screen before waiting three seconds for the player to return to the start. We now move to the elif portion of this if..elif conditional statement:

```
            elif lives == 0:
                pygame.mixer.music.load('./wrong.mp3')
                pygame.mixer.music.play(2)
                print("GAME OVER")
                sleep(3)
                break
```

The elif handles the actions if the number of lives is equal to zero. It plays an audio file three times, but you will see that `pygame.mixer.music.play(2)` so why does it play three times? Well, this is because Python starts counting from zero. While the music is playing 'GAME OVER' will be printed on the screen, before the project goes to sleep for three seconds (`sleep(3)`) and then breaks out of the infinite loop and returns to the main infinite loop, which waits for the user to press a button.

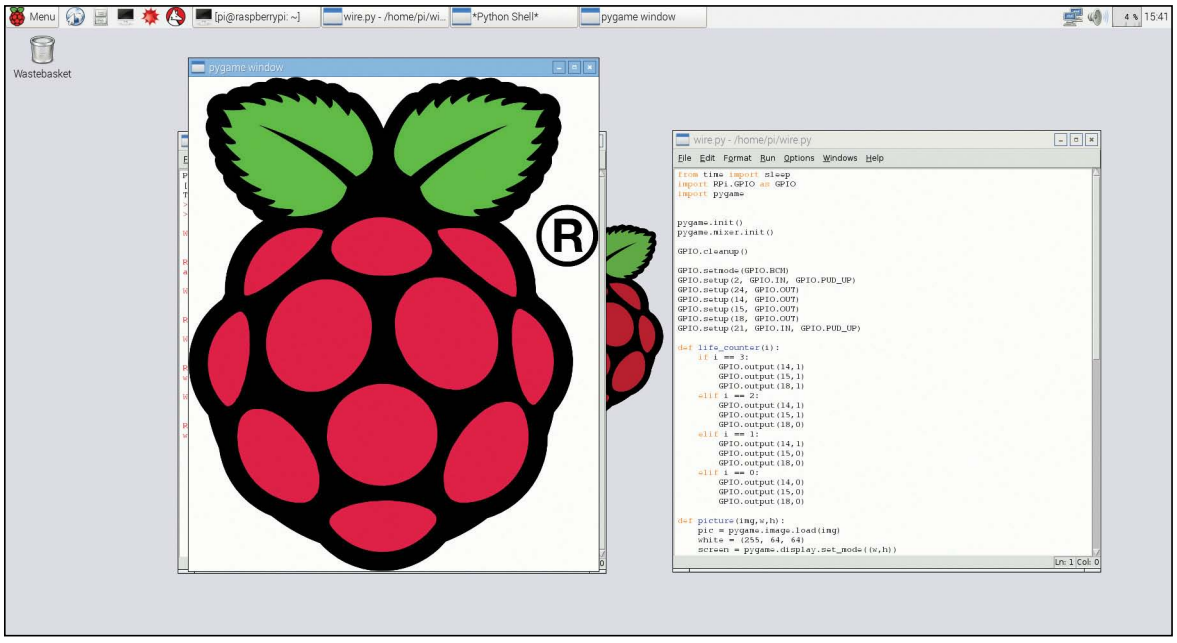
› Touching the wire triggers an alert buzzer on your breadboard, plays an audio file and displays a caution graphic on the screen.

In our final code section we will close the if conditional statement that we opened to check the state of pin 21; our button to start the game:

```
    else:
        print("Press Button to play")
except KeyboardInterrupt:
    GPIO.cleanup()
    print("EXIT")
```

If the button is not pressed the Python shell will ask the user to press the button to start the game (`print("Press Button to play")`). The next line is the exception in our try.. except construction. This will enable the player to press CTRL+c to stop the game and return the GPIO pins to their normal state and exit from the project.

With all our buzzer game code completed, remember to save your work and then run it by using the Run > Run Module menu entry. You should see a Raspberry Pi logo appear on screen. Go ahead and press the button to start your little game and be careful – Don't touch the wire or you will lose a life. If you cannot hear any sound from your speaker, please refer to the raspi-config box [*see p65*] for a detailed look at *raspi-config*. LXF



› To show that the game is ready to play we use the Raspberry Pi logo as a boot splash screen. This is a handy method of providing feedback to the user as well as looking rather nice, we think.

LINUX FORMAT Back issues » Missed one?

Issue 199 July 2015

Product code: LXFDB0199

In the magazine

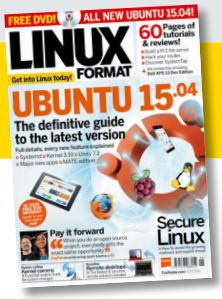
Build the ultimate Linux PC from scratch – avoid the pitfalls with our guide. Plus, which IM should you use, an interview about saving the world history with robots and open source and Unity 5 on Linux.

LXFDVD highlights

Debian 8, LXLE 14.04.02, elementary OS 0.3 and more.

Issue 198 June 2015

Product code: LXFDB0198



In the magazine

Discover all the new features packed into Ubuntu 15.04, uncover the best remote client for full desktop access, and dive into our new dedicated Raspberry Pi section. Plus: Secure your server.

LXFDVD highlights

Android-X86 4.4-r2, Grml 2014.11, Ubuntu 15.04 64- & 32-bit.

Issue 197 May 2015

Product code: LXFDB0197

In the magazine

We celebrate Raspberry Pi's third birthday with nine of the best new projects to try with your Pi. Plus, build your own Steam Machine, master Apache and code the functional way with the Haskell language.

LXFDVD highlights

Bodhi Linux 3.1, SteamOS beta, Zentyal, Netrunner 15 and more!

Issue 196 April 2015

Product code: LXFDB0196

In the magazine

Our essential guide to all the must-have open source apps as we pick the top 100. Learn how to run a Linux business, hack the kernel, make amp effects with *Guitarix* and pick up an anonymising distro.

LXFDVD highlights

Mint 17.1, KodiBuntu 14.0, Tiny Core 6.0, Scientific Linux 7.

Issue 195 March 2015

Product code: LXFDB0195

In the magazine

All the details on the new 500% faster Raspberry Pi 2! Get started with Linux: a guide to the new distros and how to install them. Port your code to Python 3 and scripting languages explored.

LXFDVD highlights

Ubuntu 14.10, Fedora 21, ArchBang 2015.01 and more!

Issue 194 February 2015

Product code: LXFDB0194

In the magazine

What's the Next Big Thing in Linux OS tech? We show the hot stuff you'll want to try. Bored of your default desktop? Take your pick of our alternatives. Plus, cake for everyone! Firefox celebrates 10 years.

LXFDVD highlights

Fedora 21 Workstation, Manjaro, ALT Linux, 4MLinux and more!

To order, visit www.myfavouritemagazines.co.uk

Select Computer from the all Magazines list and then select Linux Format.

Or call the back issues hotline on 0844 848 2852 or +44 1604 251045 for overseas orders.

Quote the issue code shown above and have your credit or debit card details ready

Future

GET OUR DIGITAL EDITION!

SUBSCRIBE TODAY AND GET 2 FREE ISSUES*

Available on your device now

Available on the App Store

Available at amazon

GET THE MAGAZINE ON Google play

nook

zinio™

*Free Trial not available on Zinio.

INTERNATIONAL READERS CAN SUBSCRIBE TOO!

Don't wait for the latest issue to reach your local store – subscribe today and let *Linux Format* come straight to you.

WHY SUBSCRIBE?

- Stop paying overseas cover prices
- Never miss another issue
- Get all 13 issues a year delivered direct to your home or workplace

TO SUBSCRIBE

- **Live in Europe?** Pay just £49.99 for a 6-month subscription
- **Live in the USA?** Pay just £49.99 for a 6-month subscription
- **Live elsewhere?** Pay just £53.99 for a 6-month subscription

TO SUBSCRIBE GO TO
myfavm.ag/LinuxFormat

Or call our helpline on +44 (0)1604 251045

Contact centre opening times are: 8am to 9.30pm (GMT) weekdays, 8am to 4pm (GMT) on Saturdays

Europe and the Rest of the World have no set newsstand prices therefore we cannot advertise the specific savings you will make. All customers will be charged in GBP. This offer is for new overseas print subscribers only. You will receive 13 issues a year. If you are dissatisfied in any way you can write to us or call us to cancel your subscription at any time and we will refund you for all unmailed issues. Prices correct at point of print and subject to change. For full terms and conditions please visit **www.myfavouritemagazines.co.uk/terms**. Offer ends 4 August 2015.

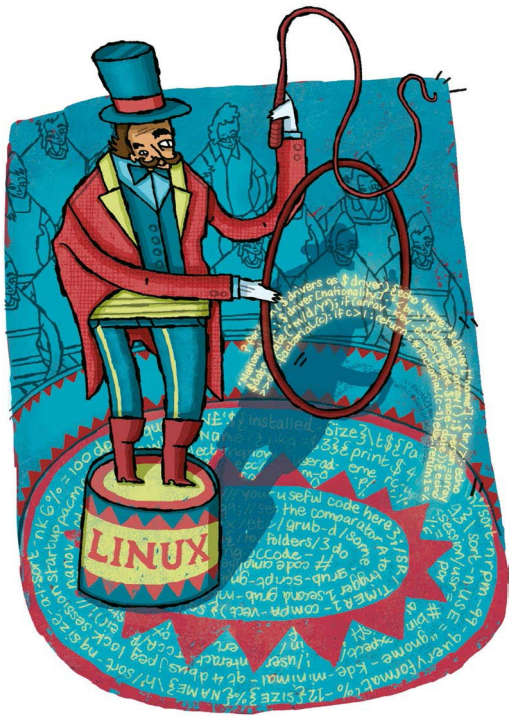
» Debian Incorporating new packages and drivers and migrating to Systemd

Debian 8: Get started

Jonni Bidwell demonstrates how to feel superior to your Ubuntu-using friends with that most venerable of Linuxes, Debian.

Our expert

Jonni Bidwell used to be an upstanding and respectable member of society with a bright future ahead of him, until he got hooked on Linux.




```
 GNU nano 2.2.6                File: /etc/apt/sources.list

#

# deb cdrom:[Debian GNU/Linux 8.0.0 _Jessie_ - Official amd64

#deb cdrom:[Debian GNU/Linux 8.0.0 _Jessie_ - Official amd64 (

deb http://ftp.uk.debian.org/debian/ jessie main
deb-src http://ftp.uk.debian.org/debian/ jessie main

deb http://security.debian.org/ jessie/updates main
deb-src http://security.debian.org/ jessie/updates main

# jessie-updates, previously known as 'volatile'
deb http://ftp.uk.debian.org/debian/ jessie-updates main
deb-src http://ftp.uk.debian.org/debian/ jessie-updates main
```

» If you live in old Blighty (aka the UK), then your vanilla sources.list file should look something like this.

Quick tip

If you are running a couple of Debian machines on your home network, then you can save some bandwidth using the *Apt-Cacher-NG* program from the repos. This will set one machine up as a caching proxy, which will act as local repository.

Last month we reviewed Debian 8 [*see Reviews, p20* **LXF199**], the latest stable release of one of the most revered distributions (distros) in the solar system. Many of our readers will be old hands at Debian, but many are no doubt eager to learn the ropes. So what follows is a guide to some everyday tasks in the most venerable of Linuxes. We'll mostly be covering the basics, but we'll also touch on some issues that Debian greybeards may run into, particularly migrating to *Systemd*. Users trying Debian for the first time are often a little disappointed that packages are deliberately stuck at stable release versions. Some people do need new packages, and we will see that there are ways to incorporate them in your Debian install. But this is not always a good idea, old doesn't mean useless, and as such it's well worth maintaining a Debian stable installation alongside a more cutting-edge distro, even if only as a fallback when you are bitten by a bug.

The installation process is largely straightforward, provided you are able to disable secure boot mechanisms. The graphical installer has been designed to be as compatible as possible, so unless your running highly obscure hardware you shouldn't need to resort to the text-only mode. If you're installing alongside Windows or any other operating system, then the manual partitioning option will let you resize any partitions appropriately. You may prefer to do your partitioning beforehand, particularly if you require, desire or already have a complicated partition layout.

Considering partitioning

It's not a good idea to move Windows partitions from Linux (eg using *Gparted*) since the Windows bootloader is sensitive to such things and generally will fail to boot afterwards. So either avoid moving Windows partitions or do so only from within Windows. Also, if you're resizing such things in *GParted*, then make sure that 'Align to' option is set to None, since otherwise the partition will be moved to the nearest cylinder or MiB, which will result in tears. It's prudent to check that Windows still boots after you've made the requisite partition changes, before you go exacerbating the situation with further OSes. It's worth noting that Debian 8.0 is also a

Get with the program

There are a number of cases where using a newer version of a package than the one provided by the Debian repos is going to be a little more desirable, eg where you are using web browsers or email clients that introduce exciting new features on every release. Backports are reworkings of packages from less stable channels. For example, Mozilla maintain an archive of *Iceweasel* packages, which you can enable as follows. As root, create a file **/etc/apt/sources.list.d/mozilla.list** containing the following line:

```
deb http://mozilla.debian.net/ jessie-backports iceweasel-release
```

You can replace `release` with either `beta` or `aurora` depending on how close to the edge you want to be. This archive is signed, so you'll need to add Mozilla's archive key to the *APT* keyring. To do this:

```
$ wget http://mozilla.debian.net/archive.asc
$ sudo apt-key add archive.asc
```

The more paranoid among you will want to verify the downloaded key. This has the fingerprint of 06C4AE2A, so verify away. Once you have added the key then you can update the package archive (you will see a few references to **http://mozilla.debian.net** during this phase) and then upgrade *Iceweasel*:

```
$ sudo apt-get update
$ sudo apt-get install iceweasel
```

Remember that the version of *Iceweasel that is* bundled with Debian is part of Mozilla's Extended Service Release, so security flaws in the browser will still be patched as a matter of urgency. Using a newer version from backports won't necessarily get you extra security, but it will get you bigger version numbers, and possibly more bugs.

There is also an official Debian backports repository so you can install packages recompiled from Testing and Unstable. You can add this with the following line in **/etc/apt/sources.list**:

```
deb http://http.debian.net/debian jessie-backports main
```

Packages from this repository are pinned with a low priority (100) so that they don't interfere with packages from stable. To install them explicitly state the backports repo with:

```
$ sudo apt-get install -t jessie-backports install "package"
```

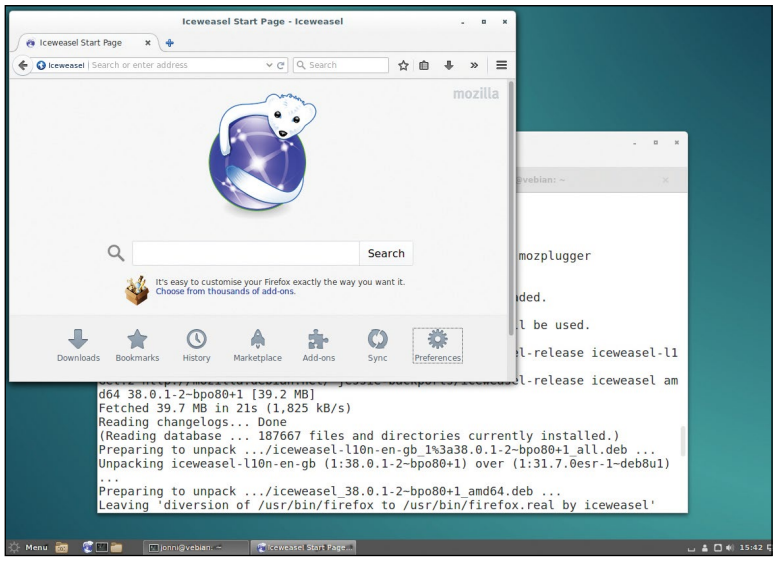
particularly good choice of OS for a VPS. Any good provider will already be providing Debian 8 images, and those not requiring powerful resources can get one for as little as $5/mo. Obviously you won't want to install a desktop environment on such a machine, but once it is deployed by the VPS provider, you'll be able to SSH in and apply some of the wisdom of this tutorial.

Just because Debian Stable eschews the latest versions of software doesn't mean that it's universally shy about package updates. These happen as frequently as in any other distro, the only difference is that the updates generally don't involve major version bumps – only bugfixes and security updates. Since the 8.0 launch a point release, 8.1, has been issued, which contains a number of these fixes. There's no need to do a re-install or anything drastic though, all of these fixes are available in the repositories. Keeping your packages up to date is vital, but also quite straightforward. Debian stores a list of repositories in the file **/etc/apt/sources.list**. You can add repositories to this file or inside a new file in the directory **/etc/apt/sources.list.d**. If you installed from a CD then your **sources.list** file will contain a line beginning `deb cdrom:`. This means that the *APT* package manager can source packages from the CD. This can save you some time post-install, if you have a slow internet connection, but once you've sync'ed with the repositories most of these packages will be out of date. So comment out the cdrom line.

Keeping current

Each line of the **sources.list** file specifies a particular repository (repos). The line begins with either `deb` or `deb-src` which determines whether binary or source packages are hosted there. The second field is the URL, the third is the distro which will be `jessie` if you've just done a new install. One may also specify a branch here, such as `stable` or `testing`. Jessie is the current stable release, but in a couple of years Stretch (the current testing release) will be stabilised. Replacing the codename `jessie` with a channel name such as `stable` throughout `sources.list` won't have any immediate effect, but when Stretch is stabilised you will be upgraded automatically. Leaving it as is means you will track the Jessie release until its end of life (usually five years after release). The remainder of the repos line stands for the components to use from the repos. Following install, only the `main` component is selected. One can add `non-free` to the repos to include various proprietary software (such as the Flash plugin – don't use this – or proprietary drivers). There are also inbetween components belonging to `contrib`. These packages are Debian Free Software Guidelines (DFSG) compliant within themselves, but have dependencies which are not. You'll notice that some lines in **sources.list** reference the distribution `jessie-updates`. These incorporate updates that it's worth installing before the next point release (at which point they will become part of Jessie proper). One can also add the distribution name `proposed-updates` to get updates scheduled for the point release, but bear in mind these will not have been fully tested.

Periodically, and whenever you update **/etc/apt/sources.list**, you'll want to run (as root) `# apt-get update` to update the local list of available packages. You'll need to use the `su` command (together with the root password you set on install) to gain root access initially, since Debian doesn't set `sudo` up for you. If you're coming from Ubuntu, then `sudo` is »



› Because the *Firefox* logo cannot be redistributed, Debian rebranded the browser as *Iceweasel*. And we updated it to version 38 using this fine tutorial.

» probably a command that is close to your heart. Setting it up on Debian is not difficult, just run the following as root to edit the file with *nano* (you can use the default editor, *vi*, if you really want, but it's not exactly newbie friendly):

```
# EDITOR=nano visudo
```

The *visudo* program edits and then checks the syntax of the file **/etc/sudoers** so that inadvertent typos don't result in everyone losing `sudo` access. Add the following line, replacing `user` with your username:

```
user    ALL=(ALL:ALL) ALL
```

Now exit with Ctrl-c and press y to save. A temporary file is saved for *visudo* to check and, if it approves, this file overwrites **/etc/sudoers**. Now you will never need the `su` command again and you can update your system exactly how an Ubuntu user would:

```
$ sudo apt-get upgrade
```

Sometimes *APT* will report that some packages will not be upgraded, which is due to some kind of dependency conflict. This will happen if, for example, you are tracking the stable channel in **/etc/apt/sources.list** at the time when a new release comes out. The solution is to use the smart dependency solver which is called with:

```
$ sudo apt-get dist-upgrade
```

We can use the `apt-cache` to search the local cache of available packages. For example, to search package descriptions matching *chromium*:

```
$ apt-cache search chromium
```

Debian package management is capable of much more than this though. The Apt suite acts as a front-end to the *dpkg* tool, abstracting away all that complicated dependency resolution and what-have-you. We can view status information for an installed or otherwise package, in this case the *Iceweasel* web browser, with:

```
$ dpkg -l iceweasel
```

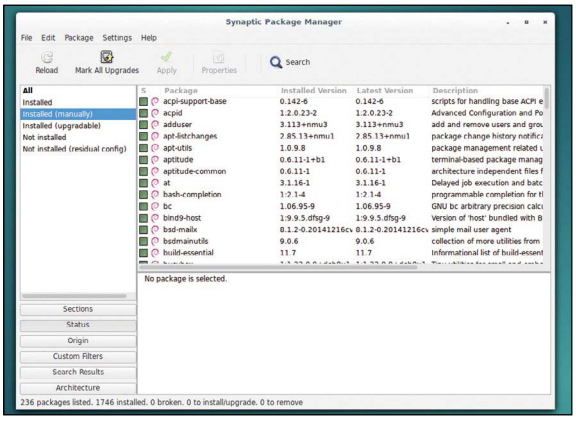
The last line will look something like

```
ii  iceweasel      31.7.0esr-1~d amd64      Web browser...
```

unless you have removed *Iceweasel* (part of the default installation) from your system. The two-letter code on the left refers to the packages status – the first `i` means that it is desired that the package is installed, and the second `i` tells us that it actually is. If a package is in a state of turmoil, then *dpkg* will show a third character, `R`, meaning that a re-install is required. Sometimes these repairs can be carried out automatically, with:

```
$ sudo apt-get install -f
```

which will attempt to make your package tree consistent.



› Synaptic is quite handy for looking at installed packages, or searching for new ones. Power users don't care for such graphical frippery though.

Removing packages

Packages can be removed (`apt-get remove`) or purged (`apt-get purge`), the latter meaning that all configuration files are removed too. Purged packages will have their status reported by `dpkg -l` as `pp`, where as those removed but still having config files will get `rc`. As new packages are installed, some dependencies will become orphaned, hence no longer required. *APT* will tell you to use

```
$ sudo apt-get autoremove
```

to do this, and you should. Also, the package cache directory, **/var/cache/apt** will fill up over time. It is good practice to periodically clean it out with:

```
$ sudo apt-get autoclean
```

Besides the lower level `dpkg` command, there is also the higher level *Aptitude* program. While you can use this instead of `apt-get` it, for every day installation and removals, it's more advisable to avoid mixing and matching front-ends, since they report their actions in different places. One of *Aptitude*'s nicest features is the ability to see why a particular package is installed, for example we wanted to know why the Xwayland server was installed:

```
$ aptitude why xwayland
```

We were using the Cinnamon desktop, which depends on the package `gnome-session-bin` (Cinnamon still uses many Gnome components behind the scenes), and therein lies the *Wayland* dependency. It will be some time before we actually see a working *Wayland* desktop in Debian, or in any other distro for that matter.

Sidestepping Systemd

Unless you've been living under a rock for the last few years, you'll be aware that some people are not so happy about *Systemd* being subsumed into almost every major Linux distribution. FUD notwithstanding, there are some reasonable criticisms of *Systemd* out there and some people will be happy to be rid of it.

Do bear in mind though that while *Systemd* can be removed and replaced with ye olde *SysVinit* for now, there is no plan to maintain the old system beyond the Jessie release. Also note that Gnome (and by extension Cinnamon) depend on *Systemd* so you may have to weigh your desire to be *Systemd*-free with your desire for those desktops.

The cleanest way to remove *Systemd* is to use a preseed line at the start of the installation process. Just press Tab on the Graphical Install menu option and add the following to the

boot command:

```
preseed/late_command="in-target apt-get install -y sysvinit-core"
```

If you've already installed, then the removal process is not so complicated. First install all the *SysV* packages:

```
$ sudo apt-get install sysvinit-core sysvinit sysvinit-utils
```

Now reboot so that *Systemd* is no longer in memory, and then purge it, possibly while burning some white sage:

```
$ sudo apt-get remove --purge --auto-remove systemd
```

You can use 'apt pinning' to ensure that *Systemd* doesn't try and sneak back onboard. As root, edit the file **/etc/apt/preferences.d/systemd** giving it the following content:

```
Package: systemd
Pin: origin
Pin-Priority: -1
```

If you run into problems trying to install package, then *Aptitude* also has a `why-not` command. There (mercifully) isn't an equivalant to the *Ubuntu Software Center*, but *Synaptic* will cater to all of your graphical package management requirements.

The standard Debian installation is designed to be as bloat free as possible. That said, there are some parts that you can probably do without, depending on how you plan on using your box. Superfluous packages will take up disk space and slow down your start up, but don't be too gun-ho about removing things. For instance, we decided that we didn't have any need for the *exim* mail server to be running, some daemons use it to send alerts to the local mail spool, but it is safe to disable it with:

```
$ sudo systemctl disable exim4
```

Trimming down startup services

Likewise, we also put the kybosh to the *modemmanager* and *speech-dispatcher* services. This shaved about five seconds off our boot time (which you can see with the `systemd-analyze` command). Debian has only just adopted *Systemd* as its init system, if you really want to disable it, then follow the instructions in the box (*see Sidestepping Systemd, bottom, p72*). But for the foreseeable future the Debian Project will maintain it in compatibility mode. This means that the old *SysV* init scripts will still work, and *Systemd* will even maintain the links in `/etc/rc?.d/` , eg if we enable the SSH server with:

```
$ sudo systemctl enable ssh
```

Then as if by magic links will appear, eg:

```
$ ls -l /etc/rc3.d/S02ssh
lrwxrwxrwx 1 root root  ... S02ssh -> ../init.d/ssh
```

If you disable the service, then that start link, and other start links in other runlevel folders, will disappear.

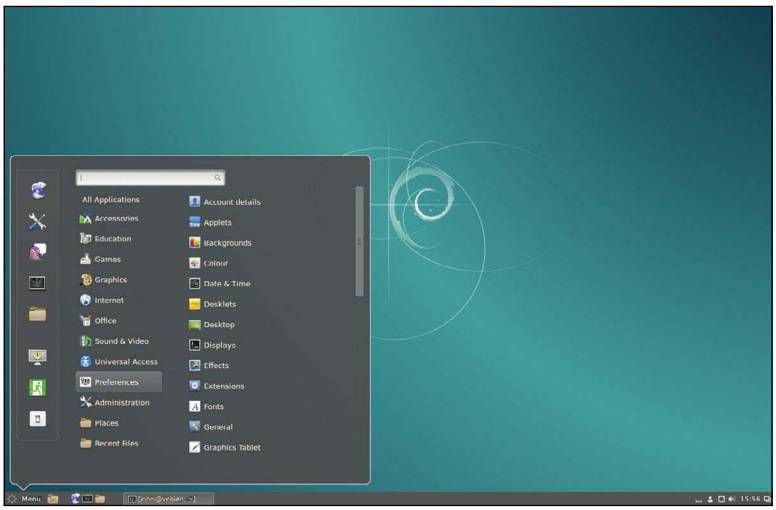
Systemd init scripts are perhaps a little tidier than their *SysV* equivalents. They are (worryingly) formatted like Windows INI files. You'll find some *Systemd* unit files in the directory **/lib/systemd/system**. Because *Systemd* is pretty smart about dependency handling, *Systemd* unit files are often much simpler than *SysV* scripts. Compare, for example, the SSH service file with its counterpart **/etc/init.d/ssh** – the *Systemd* file is 15 lines versus 154). When you enable such services with `systemctl` , links are created in the directory **/etc/systemd/system**.

Proprietary drivers

While we encourage anyone to stick with the free and open source drivers wherever possible, there remain a number of graphics cards (particular Nvidia ones) which are much better behaved when used with the proprietary ones. The first step is to add the `contrib` and `non-free` components to the `jessie` repository (the first line of substance in **/etc/apt/sources.list**). Next, you will need to add the correct version of the **linux-headers** package. If you've recently done an `apt-get upgrade` and haven't since rebooted, then it's worth doing so since the drivers need to be built against the kernel that will be running on next reboot, rather than the one currently running. If there weren't any mentions of **linux-image** or **linux-headers** in the update, then you'll be safe to proceed without a reboot. Go ahead and grab the correct headers package with

```
$ sudo apt-get install linux-headers-$(uname -r)
```

Jessie supports two branches for the Nvidia proprietary driver, 340 and 304 and just the 14.9 series for the Catalyst driver. If these are too archaic for you, which is likely, then it may be worth using the packages from the manufacturer's websites, but this is a very un-Debian way of doing things, so tread carefully. Alternatively you could explore adding the packages from the Experimental channel. This is pretty much a one-way process though, and is mostly beyond the scope of this tutorial. We will talk about picking and choosing from the less stable channels (the process is called 'apt pinning') later though. But for now we'll assume you're happy with the Debian-provided driver packages.



› The Cinnamon desktop is available from the installer, it's only version 2.2, but it looks and feels good.

For Nvidia cards and the 340 release driver (which is the last release to support GeForce 8 and 9 series CPUs) install the package as follows:

```
$ sudo apt-get install nvidia-kernel-dkms
```

Dynamic Kernel Module Support (DKMS) is an ever-so-useful system for automatically rebuilding modules whenever there is a kernel update. Without it, anyone not keeping a keen eye on package updates and manually reinstalling such packages will frequently find their graphics breaking. If you would rather have the older 304 series driver (which supports most of the same cards as 340) then use the **nvidia-legacy-304xx-kernel-dkms** package instead. *X.org* won't detect the nvidia driver, so you'll need to create a minimal **xorg.conf** file by running:

```
$ sudo nano /etc/X11/xorg.conf.d/20-nvidia
```

to edit a new file, which is a tidier way to do things. Give it the following contents:

```
Section "Device"
    Identifier "My GPU"
    Driver "nvidia"
EndSection
```

Now restart, and you should feel that shiver down the back of your spine, reminding you that your kernel is now tainted by something blobby.

Note that the Catalyst driver is incompatible with the Gnome desktop, so stick to the free driver in this case. Otherwise, to install the Catalyst driver use:

```
$ sudo apt-get install fglrx-driver
```

which will automagically pull in all the DKMS gubbins. Then, as root, create the file **/etc/X11/xorg.conf.d/20-fglrx.conf** with the following contents

```
Section "Device"
    Identifier "My GPU"
    Driver "fglrx"
EndSection
```

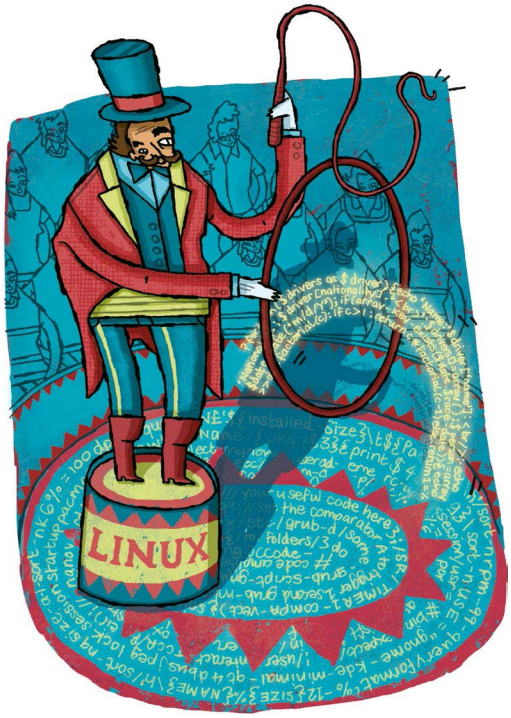
And that wraps up our guide to Debian. Be sure to check out the Debian wiki (**https://wiki.debian.org**) for guidance on your continued adventures in the Debian plains. LXF

Device security: Protection tips

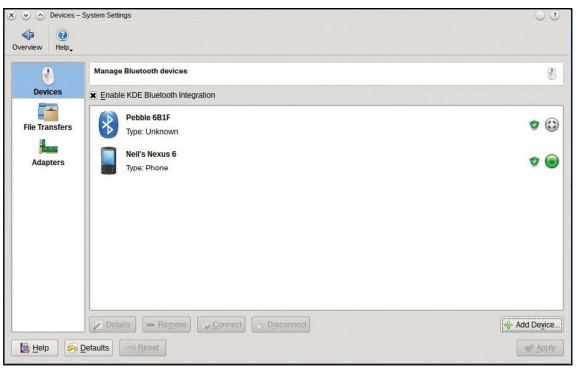
If you can't keep your laptop or smartphone out of the hands of thieves, **Neil Bothwick** shows you how to keep their contents safe.

Our expert

Neil Bothwick has a great deal of experience with booting up, as he has a computer in every room, but not as much with rebooting since he made the switch to Linux.



We often talk about security in terms of password management or encrypting data, but portable devices present another potential problem: and that's physical theft. It's easy to say that you need to 'be careful' to avoid theft of your devices, but what can you do to mitigate the damage if your laptop or smartphone is stolen? We will look at some ways of keeping your data secure should it fall into the wrong hands somehow, while avoiding adding so many layers of security that it becomes impractical or you just you don't want to use it.



› Before you can use a device as a trigger for *BlueProximity*, you have to pair it with your computer using the desktop settings program.

Strong passwords are always a good thing, but more so in the case of a mobile device. If someone takes you laptop, they will be able to try and break your passwords at their leisure. Programs like *John the Ripper* can brute-force the passwords from your system provided the attacker has access to your password files and plenty of time. You can't do anything about the time but you can make important files unreadable. Requiring root access to them is not enough, they could boot the computer from a rescue CD and read anything as root, so encrypting your hard disk should be a priority. We've covered encrypting your **home** directory [*Tutorials, p74,* **LXF196**], but encrypting the whole drive means either reinstalling or backing everything up and encrypting the drive with dm-crypt [*see Tutorials, p74,* **LXF186**]. It's a lot of work, but you should consider encrypting the whole drive when you install a distribution (distro). You should also set a BIOS password to prevent anyone changing the settings to allow booting from another medium.

Lock out

An encrypted drive or home directory is only part of the solution. It stops anyone using a live CD to read your disk, but if the laptop is stolen, while it's already switched on, your data is available for all to see. There is a simple solution to this, the easiest way to carry a laptop is with the lid closed, so could you lock your computer when the lid is closed? The answer is yes, the ACPI system reacts to various events, including closing and opening the lid. This is often used to power off the screen to prolong battery life, but you can also have it lock the screen. The details vary from desktop to desktop; in KDE you set the lid closed action in the Power Management section of the settings while Gnome users should look in Privacy and for Unity it's under Security & Privacy. It is also possible to set this up independently of the desktop environment. *Systemd* users can edit **/etc/systemd/logind.conf** and set `HandleLidSwitch` to 'lock'. Alternatively, you can have ACPI handle this directly by installing *xlock* from your package manager then creating a file called **/etc/acpi/events/lid** containing:

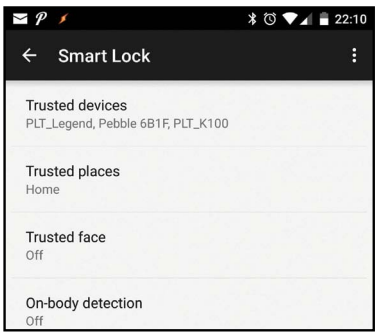
```
event=button[ /]lid.*close
action=sudo -u youruser /usr/bin/xlock -display :0
```

This runs *xlock* to lock your display when the lid is closed, you will need your password to unlock it. We use `sudo` to run it as your user because the ACPI daemon runs as root and *xlock* wants the password of the user that ran it. You will need to restart the acpid service after creating or editing this file. Most distros disable the use of Ctrl-Alt-F1 to get to a virtual console, something you should consider as *xlock* only locks

The Android options

We have mainly considered laptops running desktop Linux, but mobile devices running Android can easily be lost. Protecting the device with a PIN or pattern to unlock it is more secure, but a lot less friendly. Fortunately, the Lollipop edition of Android (5.x) has a feature called Smart Lock, found in the Security settings, that allows you to bypass the lock when certain conditions are met. Currently this works on GPS location or the presence of a Bluetooth device. For example, you can disable the lock when you are at home or in your car.

Locating and even remote wiping a lost smartphone can be done from Google's device manager at **https://www.google.co.uk/android/devicemanager**. You can make your phone more secure by enabling two factor authentication and selecting the setting to require a PIN when turning it on.



Android Lollipop's Smart Lock enables you to automatically unlock your device under particular circumstances.

the X display. You can do this by adding the following to **/etc/X11/xorg.conf** or a file in **/etc/X11/xorg.conf.d**

```
Section "ServerFlags"
  Option "DontVTSwitch" "true"
EndSection
```

Proximity locking

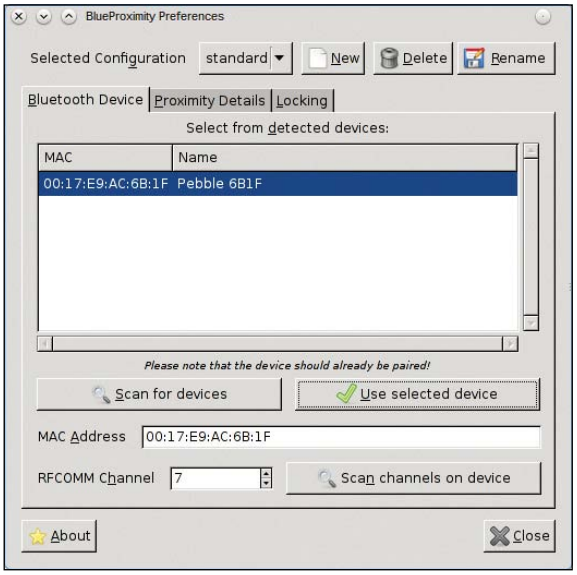
This method has two disadvantages, it relies on a thief closing the lid, which isn't such a problem, and it means you have to type your password every time you open it, and we just advised you to use a strong one. It would be handy if the laptop worked when you were using it but not for anyone else. We are not talking fingerprint scanners here but something that almost all of you already have: a smartphone. *BlueProximity* (**http://BlueProximity.sourceforge.net**) monitors a paired Bluetooth device and when it moves a certain distance away it locks the computer, when the device comes back in range it unlocks it. if your phone always lives in your pocket, or hand, it works well. It's even better if you link it to a smartwatch.

The first step is to pair your Bluetooth device with your computer, using your desktop's settings tools. Then you can start *BlueProximity*. Starting it from the launcher appears to do nothing, check the system tray as it may have started up hidden. Start by pressing the Scan button. Click on your device in the detected list and press 'Use selected device', which will copy its MAC address to the field below. Now you can set up the detection parameters in the 'Proximity Details' tab. The distance settings are actually measurements of signal strength, the higher the number he weaker the signal – or the greater the distance. The duration controls how long your device should be past that distance before the *BlueProximity* acts. If you want to set a specific distance, press the 'Reset Max/Min' button, take your device to the distance at which you want locking to occur, wait a few seconds then come back and you will see the maximum distance registered at the bottom of the display. For most uses, the defaults should be close enough, it's normal to set the locking distance to be greater than the unlocking distance, otherwise you may find it locking and unlocking when you are close to the common setting. Setting the duration too short can cause unexpected events too, because the Bluetooth signal strength reading can fluctuate.

The third tab sets the commands to run on lock and unlock, the default is to run gnome-screensaver, click the arrow button to use xscreensaver instead or type in your own command. *BlueProximity* will not run the lock commands while its window is open, but it will change the key's colour in its icon to show whether you're in range or not. Green and red have the obvious meanings, yellow means you are within lock range, but outside of the unlock distance. The proximity command is run whenever your device is in range, at the interval specified. The default is to keep gnome-screensaver from blanking the screen, even if you don't touch the keyboard or mouse. You may need to experiment a little with these settings, but you will end up with a computer that can only be used when your Bluetooth device is nearby.

Finding a lost device

Should the worst happen and your device is stolen, there is something you can do to track it or protect your data. By installing the *Prey* (**http://preyproject.com**) client on your computers and mobile devices, and setting up a free account, you will be able to get at your computer when it's online, even if someone else has it. The basic service is free, with various levels of premium service for extra features and devices, but the free option works with up to three devices and gives a good level of reassurance. There are other services, like *Cerberus*, but *Prey* has the advantages of being open source and running on all desktop and mobile operating systems. LXF



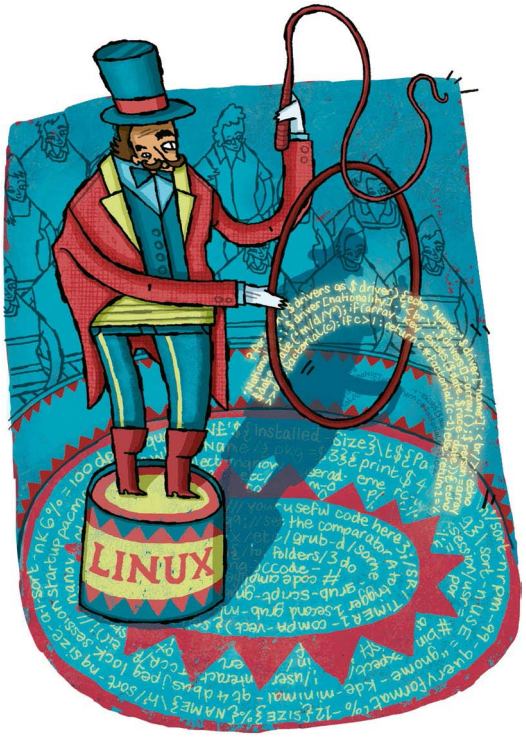
The first tab of the *BlueProximity* window is used to select the device to look for.

Kernel: Patch a running kernel

Jonni Bidwell shows you the principles behind the most exciting feature of Kernel 4.0: live patching and how to give it a go yourself.

Our expert

Jonni Bidwell wonders if the company-wide poo poo-ing of the Oxford comma also extends to that most sinister of pauses, the Miskatonic comma.



Applying software updates in Linux very rarely demands that the user reboot. Generally all one needs to do is restart any applications or services affected by the upgrade, which more often than not can be done without impacting current operations. Compare this with, say, Microsoft Windows, where enforced (or at least annoyingly reminded) reboots are par for the course. There remains though, one very big exception: the kernel.

Installing a new kernel and modules won't upset a running Linux system, but that is because they won't get loaded until the next reboot. For desktop users rebooting is at worst a minor inconvenience, but for operators of mission critical systems, reboots often have to be scheduled well in advance. Indeed, even restarting services should not be done without due consideration for users and current workloads. For this reason sysadmins tend to be very conservative in regard to updating software, so that downtime is minimised and users don't get angry.

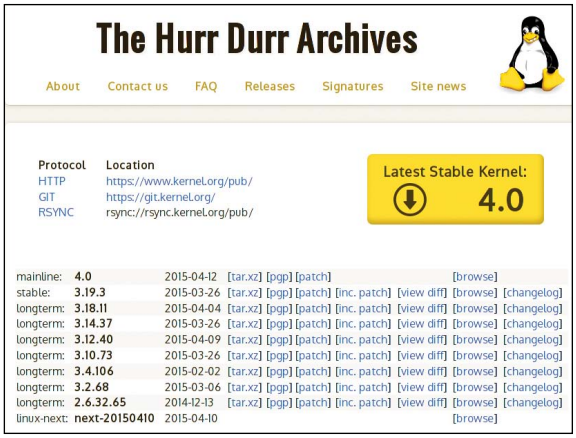
But we've entered an age of vulnerabilities bearing contrived acronyms and stylish logos, so with increasing frequency security interests (rightfully) dictate that service is going to have to be interrupted. While kernel bugs are mercifully rare (we haven't had one with its own logo yet), and doubly so for the older long-term branches, they do happen and admins of affected systems must work out some means to apply fixes as soon as possible. More often than not, this involves late nights, caffeine and desperate appeals to various deities for a successful outcome. It would be nice then, if there was a mechanism by which security patches could be applied to a running kernel, so that bugs or vulnerabilities could be trounced without any interruption to service.

Dynamic kernel patching has been something of a holy grail for sysadmins. The technology has actually been around since 2008, through a system called Ksplice. This system was (and remains) licensed under the GPLv2, but its authors also provided, under the commercial aegis Ksplice Inc, paid-for tools to simplify the installation on various distros. Ksplice Inc was acquired by Oracle in 2011, and while Ksplice itself remains an open source kernel extension, its new owners don't support anything other than Oracle Linux. So for anyone else, the dream of live kernel patching will have to be sought elsewhere.

It's been a couple of years in the making, but with Linux 4.0 (released mid-April 2015) this technology is, for all intents and purposes, here. It is known as Live Kernel Patching (livepatch) and is an amalgamation of two rival technologies: OpenSUSE's kGraft and Redhat's kpatch. Both of these projects rely on a kernel symbol `CONFIG_FTRACE` which provides function tracing, so the kernel is aware of which functions are being used at any given moment. Tracing is pivotal for both these systems since running processes need

Quick tip

If you have the smarts to compile a 4.0 kernel, then enable the SAMPLE_LIVE_PATCHING option. This will make a module **livepatch-sample.ko** which, when loaded, will live patch **/proc/cmdline** to display a message alerting you of its newly patched status.



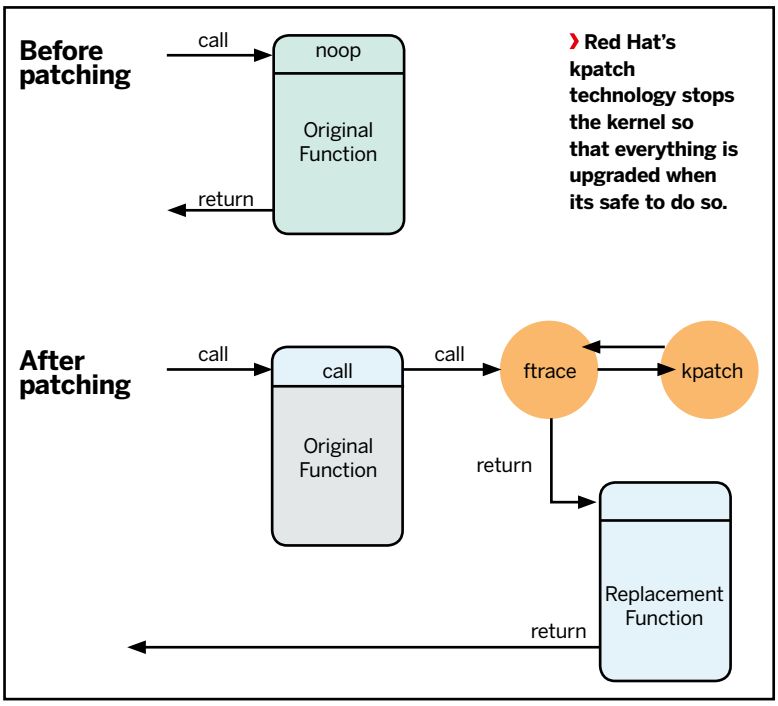
› Some prankster seems to have set up this mirror of kernel.org at http://imasheep.hurrdurr.org.

to be worked around rather than stopped. Original versions of patched kernel functions still need to be available for things that were running pre-patching, since inconsistencies, crashes or tears could result otherwise. While both parent technologies, and indeed the resultant offspring, differ in their approaches, there is a common core. Suppose we wish to apply a patch to our currently running kernel. Just to be clear this is a plaintext patch, which applies to the source code of the kernel currently in use. The patching mechanism now compiles both the patched and unpatched kernel sources and compares them by examining the resulting binaries. It might seem odd to perform this analysis at the binary level, rather than the more readable source code level, but it turns out to be a much more robust way of working, since we need to know how the patch affects the finished product. Checks are carried out to see if patching can take place safely (ie the patch doesn't exert too drastic changes) and if so the live patch is compiled as a kernel module. The module performs the required code redirection – updating old functions with jump instructions to the new ones.

Patch 'n' go

At this point, it's worth being clear about what kind of patches can and cannot be applied live. A desktop user's only experience with kernel updates might be limited to installing a package (usually called **linux** or **linux-image**) whenever it's made available by their distribution (distro). Whether the package upgrades to a new major version (eg from 4.0 to 4.1) or applies some small security fix without changing versions, the package management approach is the same: A whole new kernel is installed and springs into life on reboot. Live patching is a different beast entirely: the old kernel remains in memory throughout and any new code is patched in through a kernel module. We'll explain that process in more detail later, but the upshot is that only relatively minor kernel patches can be applied. Anything that introduces new/different data structures or new kernel functions is not a viable candidate for dynamic patching. Thus kernel upgrades (even only minor version ones) are out of the question, but this was never the problem that patching intended to solve. Rather we want a mechanism for applying security fixes, which very rarely introduce new internal semantics.

The Kpatch approach waits until all function calls have stopped before swapping in the new update, whereas kGraft selectively redirects to old functions for currently running calls and new ones otherwise. Kpatch then is a little simpler, but then there is a delay waiting for function calls to finish.



Red Hat's kpatch technology stops the kernel so that everything is upgraded when its safe to do so.

kGraft by comparison has to do some fairly high-stakes marshalling, deciding which 'universe' (old or new) each call is running in by performing a 'reality check' and mapping the appropriate functions. Even though kGraft does not introduce additional latencies, it may still take some time before all functions are replaced – there may be some long running processes which prevents update of any associated kernel functions. Livepatch, the now-official kernel solution, takes inspiration from both of these approaches and is compatible with the userspace tools for both. This is fortuitous since at the time of writing there are no livepatch userspace tools available, in fact live patching proper won't be available for some time, only the rudiments of it made it into the 4.0 kernel. More involved patches need to be tailored to suit the whims of the live patching system and currently architectures other than x86. But that's okay, very few mainstream distros are using the kernel series anyway, and both Kpatch and kGraft work with older kernels.

We're going to use Redhat's kpatch tools to show you the process. These need to be compiled from source, and you'll need to install some dependencies first. Besides the standard *make* and *GCC* you'll also need tools for working with ELF »

Not the end of reboots

As live patching work progressed, many blogs and tech news sites started peddling stories promising that you'd never need to reboot your Linux box again. This is simply not true, and nor will it be anytime soon. Packages besides the kernel need reboots too: PID1 is sensitive so init system (eg *Systemd*) upgrades still necessitate a reboot. One can attempt to reload the whole process (eg `systemctl daemon-reexec` or even `kill 1`) but this might just cause a kernel panic. Likewise, if you do a glibc upgrade then you might be able to get away with just restarting all affected services (which would be most of them), but it's so ingrained that you'd probably be better off restarting.

Graphics driver updates will certainly need a display manager restart (and possibly a restart if they're using KMS) and even adding a user to a new group won't take effect until that user logs in again. In the latter case, if the user wants to use their new group privileges in an X session, then they'll need to start a new one (or use the lesser known `newgrp` command). For many people, restarting X is just as inconvenient as a full on reboot – you still have to close everything, wait a few seconds and then type in a password. As a sysadmin you probably won't be worried about restarting the display manager, but you will be worried about restarting other services, and all the scheduling, wailing and gnashing of teeth that go therewith. So live kernel patching is no panacea for anyone irked by reboots, but that doesn't make it any less useful.

» binaries and kernel debugging symbols. You'll need plenty of disk space too – compiling two kernels will chew this up, so 15GB of free space is recommended. The exact package names, and to an extent the precise instructions, depend on what distro you are using, we'll go with Ubuntu 14.04, but instructions for Debian, Redhat flavours and even Oracle Linux can be found at **https://github.com/dynup/kpatch**.

First install all the required dependencies:

```
$ sudo apt-get install make gcc libelf-dev dpkg-dev
$ sudo apt-get build-dep linux
```

Kpatch tools

You need the kernel debug symbols for the kernel you're currently running, which requires you to add the **ddebs** repository. As root, create a file **/etc/apt/sources.list.d/ddebs.list** with the following contents, replacing `trusty` with `utopic` or `vivid` if your using 14.10 or 15.04 respectively:

```
deb http://ddebs.ubuntu.com/ trusty main restricted universe multiverse
deb http://ddebs.ubuntu.com/ trusty-security main restricted universe multiverse
deb http://ddebs.ubuntu.com/ trusty-updates main restricted universe multiverse
deb http://ddebs.ubuntu.com/ trusty-proposed main restricted universe multiverse
```

We also need to add this repository's key to the apt keyring and then update the package lists:

```
$ wget -Nq http://ddebs.ubuntu.com/dbgsym-release-key.asc -O- | sudo apt-key add -
$ sudo apt-get update
```

We can also install the kernel debugging symbols:

```
$ sudo apt-get install linux-image-$(uname -r)-dbgsym
```

Now we need to fetch and compile the Kpatch sources found on GitHub:

```
$ git clone https://github.com/dynup/kpatch.git
$ cd kpatch/
$ make
$ sudo make install
```

We also need a suitable patch to apply, here's one that changes **/proc/meminfo** to display `VmallocChunk` in capitals instead. This is a trivial change which is summed up in the following patch, which you should save as **~/meminfo-string.patch**:

```
Index: src/fs/proc/meminfo.c
===================================================================
--- src.orig/fs/proc/meminfo.c
+++ src/fs/proc/meminfo.c
@@ -95,7 +95,7 @@
        "Committed_AS:   %8lu kB\n"
        "VmallocTotal:   %8lu kB\n"
        "VmallocUsed:    %8lu kB\n"
-       "VmallocChunk:   %8lu kB\n"
+       "VMALLOCCHUNK:   %8lu kB\n"
 #ifdef CONFIG_MEMORY_FAILURE
        "HardwareCorrupted: %5lu kB\n"
 #endif
```

This is tedious to transcribe (you need to use tabs to match the original file), so by all means make your own trivial patch or copy and paste from the Quick start section of the dynup GitHub repo. You can now begin the lengthy compilation process with:

```
$ kpatch-build -t vmlinux meminfo-string.patch
```

This will take a long time, so once the patch is checked (after 'Building original kernel' appears) you are safe to amble kitchenwards and prepare the first of what will likely be many cups of tea. Kpatch will download the appropriate kernel version to **~/.kpatch/src** , so if your patch didn't work you can make a new one from here using `diff` . Eventually you'll see the cheery response:

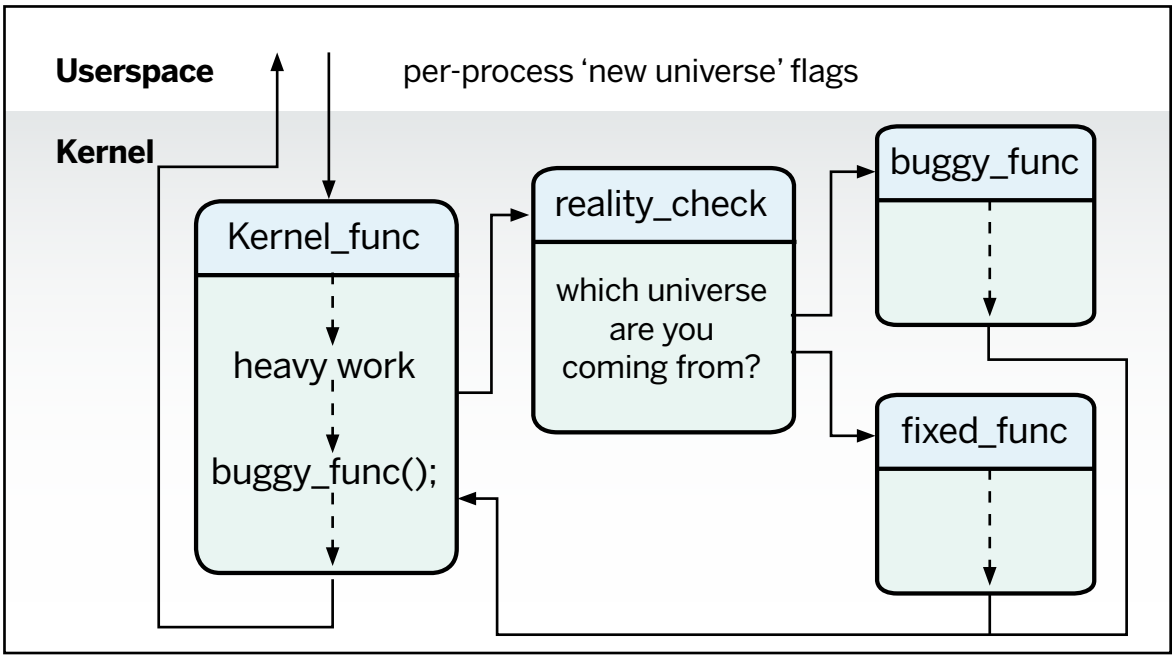
```
Building patch module: kpatch-meminfo-string.ko
SUCCESS
```

The module will be built in the current directory, but you can't load it with the usual *insmod* or *modprobe* tools, so use:

```
$ sudo kpatch load kpatch-meminfo-string.ko
```

This will load the core and patch modules, so that now if you examine the relevant lines of **/proc/meminfo** you will see something like:

```
VmallocTotal:   34359738367 kB
```



› OpenSUSE's approach is more complicated, but reality checks ensure that everything is consistent even though there are no delays.

Look to the future

By the time you read this, kernel 4.1 will probably have been released and you will be jealous of your Arch-using friends for having these and other exciting new features.

There are a lot of graphics-related updates, including GTX 750 firmware generation by Nouveau, support for Intel XenGT virtual graphics and vGEM (for accelerating Mesa software rasterisers), not to mention Radeon DisplayPort MST. We also have filesystem-level encryption for Ext4 (courtesy of Google who use Ext4 in Android), improved support for software RAID with parity (ie levels 5 and 6). There's also improved ACPI support for new Intel Atom-based SoCs and more work on the upcoming Skylake processors. Finally, the Flash-Friendly FileSystem (F2FS) gains a slew of fixes and a host of new features.

At the time of writing, the 4.2 merge window remains open, but so far we've got: Support for AMD VCE1 video encoding, as well as the new AMDGPU driver, which will be used by both proprietary and open source drivers. Kernel 4.2 will also support the EFI system resource table, which means that people with UEFI systems can update their firmware (per the UEFI Capsule Update spec) from the comfort of their desktops. Speculating now, we might also expect to see KDBUS, an in-kernel implementation of the DBUS IPC system, which would offer improved security and performance. We might even see some more livepatch work – watch this space.

```
VmallocUsed:       28348 kB
VMALLOCCHUNK:   34359700664 kB
```

Per our patch, there are capitals where once there was CamelCase. The *kpatch* tool allows us to manage which patches are available and active. Running `kpatch list` will confirm that our patch is loaded. We can also install it so that is added to the initrd image and loaded early next boot, which will be useful if a new kernel package is not available for some vulnerability against which you have just live-patched. If you decide the all caps is a bit much, then you can unload your patch with:

```
$ sudo kpatch unload kpatch-meminfo-string.ko
```

So that illustrates the theory, if you're a hardened kernel hacker then why not try some more advanced kernel patching; see what works and what doesn't. If you're new to all this, don't worry, it's pretty crazy stuff and it'll be a long time before end users have anything to gain from it.

New features, new bugs

Notwithstanding the numerical move from 3.19 to 4.0, there aren't a whole lot of groundbreaking new features to write home about in the newest Linux kernel. The same was true of the move from 2.6 to 3.0; new numbers don't necessarily mean new stuff, although lots of bugs are soundly crushed along the way. Certainly live patching is a big deal, and we've shown through our toy example that the technology is sound, but there remains a great deal of work to be done before this is ready for mainstream usage. There are some neat new features though:

» **DAX (Direct Access, eXciting)** This circumvents the unnecessary copying to kernel caches that's required when dealing with non-volatile memory devices

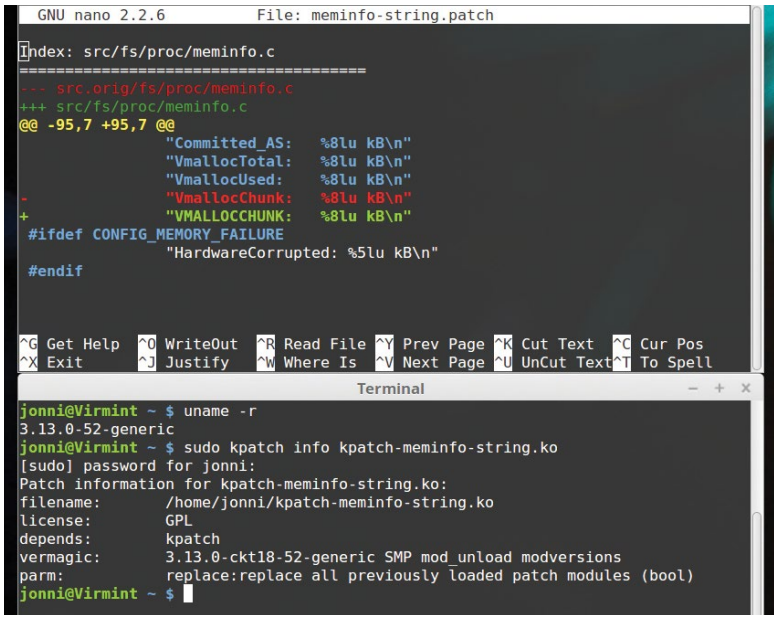
» **Lazytime** Unix systems have various different timestamps (atime, mtime) which are quite expensive to maintain. The **relatime** mount option is a workaround for minimising disruption, but it breaks some software. Lazytime keeps timestamps in the cache rather than writing them to disk, improving performance. (see an excellent article on LWN.net that explains it well **http://bit.ly/IntroToLazytime**)

» **KASan (Kernel Address Sanitizer)** This is a smart memory error detector that can find memory leaks and bugs faster than the existing kmemcheck.

» **NFS** Version 4.2 is now the default. Also debuting is Parallel NFS which separates data and metadata paths to improve scalability.

» **Dm-crypt** This is now much more scalable across multiple CPUs due to using an unbound work queue.

» **Overlayfs** Multiple lower layers are now supported for mounting several filesystems atop one another.



› With only a small amount of kernel fiddling, we managed to apply the patch on Mint 17.1.

When Kernel 4.0 (codename 'Hurr durr I'm a sheep') was released, Linus hinted that 4.1 would be a bigger release. Have a look at the box (*Looking to the Future, above*) to see what's been approved for 4.1 and what might find its way into version 4.2

At the time of writing, a data corruption bug in kernel 4.0 (and the 4.1 release candidates) has just been confirmed which affects Ext4 RAID0 filesystems. It took some tracking down, but it was due to a regression introduced with a fix for a longstanding issue (since 3.14) which led to a miscalculation when RAID 0 deals with non-power-of-2 chunksizes. Reports of corruption surfaced soon after the release, but narrowing down the problem proved tricky. A fix has been included in 4.0.3, though this will offer little consolation to those who have lost data. This is rather reminiscent of Ext4's initial release in 2012, when sporadic reports of data corruption led to a furore over the filesystem's stability. In that case most of the reported failure instances turned out to be false.

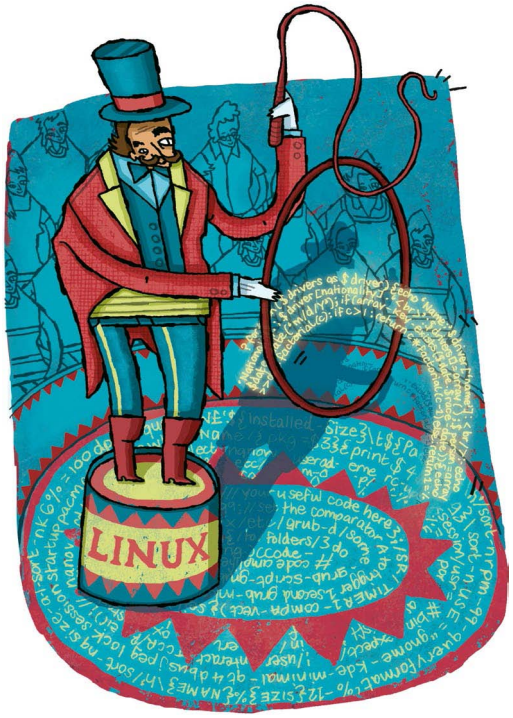
A problem was discovered, but it only affected people using non-standard mount options in a variety of uncommon situations, including rebooting twice inside a small interval. Nobody wants bugs, especially not in what is the default filesystem for many a distro, and these ones serve to illustrate why one should exercise caution when exposing their sensitive data [you mean 'bits' - Ed] to brand new kernels. Regular backups, people. LXF

IPv6: How to get connected

We did think about building a survival shelter in response to imminent IPv4-exhaustion, but **Jonni Bidwell** thought this tutorial would do the job, too.

Our expert

Jonni Bidwell is often found cowering under his desk in fear of the IPv4 apocalypse, but will usually come out for a Crunchie bar or free pizza.



February 2011 saw the IANA (Internet Assigned Numbers Authority) distribute the five remaining /8 blocks (each comprising 16 million addresses, and being the scrapings of the recovered address barrel) among the five Regional Internet Registries (RIRs). Now they have nothing left to give, they are gone. Item discontinued. Sure, you'll still be able to get IP addresses for new devices and virtual machines and what not, but companies' supplies will go dry and there ain't gonna be no re-up.

There's no need to panic, though – many years of thought and testing have already come up with a replacement addressing protocol, and it's been considered production-ready in Linux since 2005. Some major players have already activated their IPv6 networks. If you're in the UK, the chances are your ISP doesn't yet provide you with IPv6 connectivity (a handful do, and those lucky enough have no need of this tutorial), but we can work around this using a tunnel later. It's important to note that there's no plan to pull the plug on IPv4 – the two protocols can work perfectly well alongside each other. Network adaptors can be assigned both types of addresses, and dual-stack routers can marshall both types of traffic. But post-exhaustion, new hosts will only be accessible by IPv6, so those with only IPv4 connectivity will find themselves no longer able to access parts of the internet (and to some extent vice versa). While some network hardware (such as routers) may need to be upgraded, bread-and-butter network cards and switches will work fine for IPv6 – all they care about is transmitting Ethernet frames, indifferent as to the nature of the higher level packetry contained therein.

IPv4 uses 32 bits to address nodes, with the convention being to divide into four groups of eight bits (a byte, or octet) and write the decimal representation of each octet separated by dots. Thus IPv4 address space allows for about 4.3 billion addresses, which coincidentally is about one for each person on the earth who enjoys internet access. As more and more devices connect to the internet, and carrier-grade NAT address sharing regimes are not seen as an appropriate solution, we rapidly approach IPv4 exhaustion. IPv6 addresses, by comparison, use 128 bits for each address, which means that we won't be running out any time soon (there's enough for each atom on the surface of the earth to get one). The standard notation is to partition an address into eight groups of 16 bits. Each group is written as four hex digits, and separated from the next by a colon.

A whole lot of hex

Because 32 hex digits, and seven colons, are cumbersome to write down, there's a couple of shortcuts. Shortcut the first: any leading zeroes can be omitted from each group, so **0123** can be written **123**. Shortcut the second: one series of adjacent **0** (short for **0000**) groups can be replaced by **::**. As an example, consider the loopback address (the analog of **127.0.0.1** in IPv4) which, when abbreviated by the first rule, is **0:0:0:0:0:0:0:1**. Those first seven groups can be unceremoniously discarded and we need only write **::1**. Note that you can only use the double colon notation once in an address, because multiple applications would be ambiguous. An IPv6 address is divided into two parts: the first 64 bits (or four groups) form the network prefix and the last 64 are termed the host identifier. The network prefix is further divided into a routing prefix and a subnet ID, but we're not going to worry about that here.

If you're using an up-to-date distro, it'll probably have IPv6 support built in. You can check this using the `ip` command from the **iproute2** package:

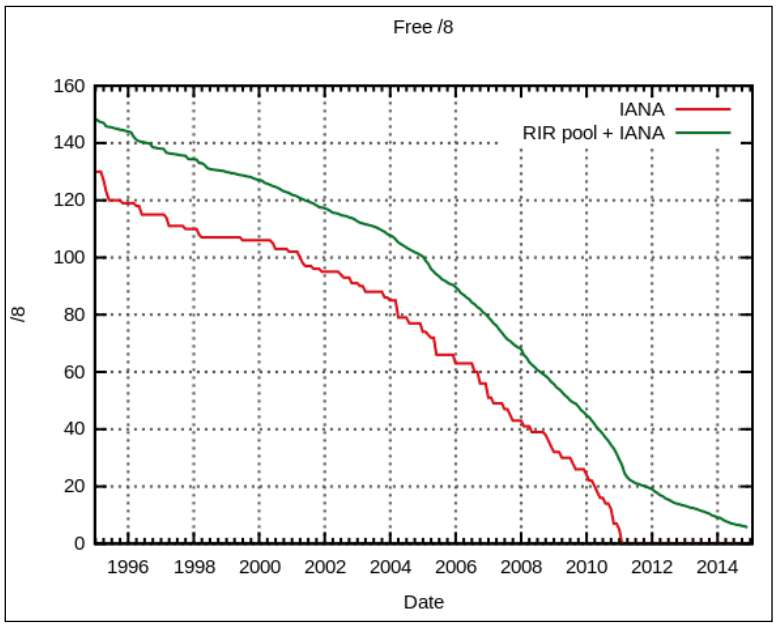
```
$ ip a
...
2: enp5s0: <BROADCAST,MULTICAST,UP,LOWER_UP>
mtu 1500 qdisc fq_codel state UP group default qlen 1000
```

```
    link/ether 90:2b:34:aa:bb:cc brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.144/24 brd 192.168.1.255 scope global
enp5s0
       valid_lft forever preferred_lft forever
    inet6 fe80::922b:34ff:feaa:bbcc/64 scope link
       valid_lft forever preferred_lft forever
```

Behold, our Ethernet card (**enp5s0**) has an **inet6** address. But it's not a globally accessible one – the **fe80::** prefix (where the **::** stands for three groups of zeroes) implies that this is a link-local address. This is somewhat like the **169.254.x.x** IPv4 auto configuration address family, which you may have seen when something goes wrong with your DHCP server. If you compare the host identifier with the MAC address (**90:2b:34:aa:bb:cc**), you should notice more than a passing similarity. This is no coincidence, because the link-local address is generated by inserting **ff:fe** into the middle of the MAC address and complementing the seventh most significant bit of the resulting string (which will either add or subtract two from the second hex digit). The link-local address is necessary for any IPv6 host – applications rely on it being there. Furthermore, because MAC addresses are unique(-ish, though they can be forged or spoofed), link-local addresses are unique for each device. This method of forming the host-identifier – because it happens for some types of non-link-local connections, too – has lead to privacy concerns. Chief among them is that IPv6 addresses (or the latter half of them, anyway) can be tied to individual devices and hence individuals. Fortunately, privacy-enhancing mechanisms exist which can generate unique ephemeral host identifiers as frequently as desired.

Configuring your route

Of course, your home router may not be configured to provide (or even support) any kind of IPv6 connectivity. But fear not because we can easily set up a simple internal IPv6 network or just a single machine (if you're lazy) and/or use a tunnelling arrangement to talk IPv6 to the outside world. Your router will allow for IPv6 traffic on the internal network, even if it doesn't have the facility to route it outside, or even know what it is. We first need assign what is called a Unique Local Address (ULA) prefix to the machine on the internal network that will act as our router. We can then set this up to advertise itself and provide SLAAC information to other machines. ULA prefixes come from a reserved block



(addresses beginning **fd**]), which does not get routed from the open internet. You can generate your own at **www.simpledns.com/private-IPv6.aspx**, or simply make one up. Now, suppose we want to use the following prefix: **fd00:dead:bad:1dea::/64**. For simplicity, we'll initially configure the host identifier (the rest of our router's IPv6 address) statically, setting it to **::1** for brevity. The `ip` command is a good method to use to achieve this:

```
$ sudo ip addr add dev int0 fd00:dead:bad:1dea::1/64
```

Here, **int0** is the name of your network interface (ours was **enp5s0**, while yours might be **eth0**). Now you should be able to ping yourself over IPv6:

```
$ ping6 -c 5 fd00:dead:bad:1dea::1
```

To make the address assignment persist across reboots depends very much on your distribution – it can be done through the *NetworkManager* GUI or various networking scripts (for example, **netctl** on Arch Linux). If you just want a local IPv6 network, skip ahead (*see Router advertising section, p82*), otherwise read on.

In order to communicate with the rest of the IPv6-speaking world (and assuming that your ISP doesn't provide this connectivity already), we need to tunnel our traffic over »

› There used to be plenty, and now there are none. This same tale may be true of all of the earth's resources.

IPv4 abstention exercise

IPv6 uptake has, on the whole been notoriously slow, despite many large firms having flipped the appropriate switches and IPv6 being mandatory for the 4G mobile standard.

Part of the reason for this is that IPv4 has proven itself to be something of a champion. The IPSec and DNSSec security extensions, which were originally designed as part of IPv6, were incorporated into it, which made for one less reason to switch over. In 2014, though, many older corporate routers began to baulk as BGP tables exceeded 512K entries. It's speculated that this was the cause of downtime at Ebay and Lastpass. While everyone recovered reasonably swiftly – a few auctions notwithstanding – this provided something of a wake-up call to everyone concerned.

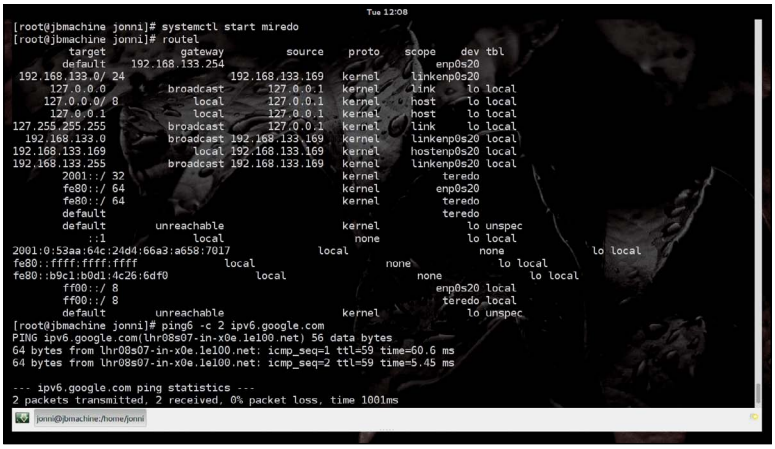
Parts of the system were beginning to creak somewhat ominously, and network bods began to act. As a result, you will find that, if you turn off IPv4 traffic, much of the internet still works as you would expect.

If you're using a Teredo tunnel, and have set it up to communicate on UDP port 3544, then the following iptables rules will do a good job of blocking any other outgoing requests, effectively cutting you off from the IPv4 internet, except for your DNS server:

```
# iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
# iptables -A OUTPUT -p UDP --dport 53 -j ACCEPT
# iptables -A OUTPUT -p UDP --sport 3544 -j ACCEPT
# iptables -A OUTPUT -j REJECT
```

You'll find some sites work as expected, some kind of work, but most don't. You probably need IPv4 connectivity, so once you've finished experimenting, you should delete these rules. If you didn't have any to begin with, you can flush the whole lot with:

```
# iptables -F
```



› Teredo tunnels are a last-resort method for getting a direct-to-host IPv6 tunnel. You shouldn't do it, but that gives it something of a forbidden fruit quality. Especially if you can't mess with your router settings...

» IPv4, using the popular 6in4 protocol. One option involves signing up to a tunnel broker service, such as Hurricane Electric or SixXS. They provide this service for free, but you still have to register, which for SixXS involves two manual human approvals. There's a number of ways that you can encapsulate IPv6 traffic inside IPv4 packets, but if you're behind a NAT layer, the easiest solution is the Anything in Anything (AYIYA) protocol. SixXS has provided the handy *Automatic IPv6 Connectivity Configuration Utility* (available in the **aiccu** package), which makes easy work of the tunnelling setup. Some distributions ask you for your SixXS credentials when you install this, while others require you to manually edit **/etc/aiccu.conf** so that it looks like this:

```
username <username>
password <password>
protocol tic
server tic.sixxs.net
IPv6_interface sixxs
automatic true
requiretls true
pidfile /var/run/aiccu.pid
defaultroute true
makebeats true
behindnat true
```

When you start the **aiccu**, you should be good to go.

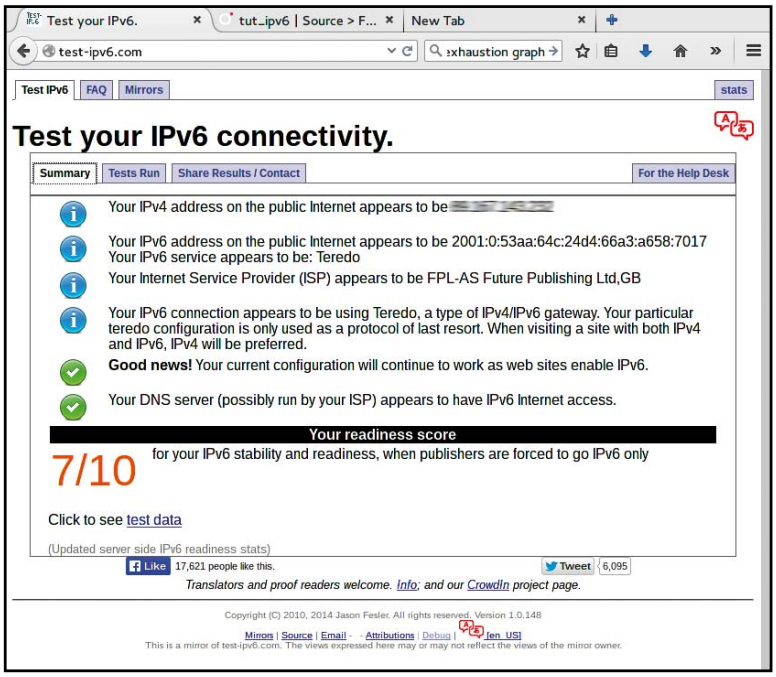
Hurricane Electric (**https://tunnelbroker.net**) requires the tunnel endpoints to be specified manually (through its web interface, where you can choose a geographically nearby endpoint – `$HE_ENDPOINT4` in the example below), as well as manual configuration of the tunnel at the client end. Furthermore, the company requires that your router's external IP be pingable from the outside world, although port forwarding should not be required. ICMP messages are necessary to negotiate IPv6 routes. Hurricane Electric assigns you a client endpoint IPv6 address beginning with **2001:** and ending with **/64**, which we call `$CLI_IP6` in the code below. You are free to set up as many hosts on this network prefix as you like; there is no need to set up a new tunnel for each. The tunnel is set up as follows, where `$INT_IP4` is your internal IP address. Note that these commands need to be done as root, so use `sudo` if necessary:

```
# ip tunnel add he-IPv6 mode sit remote $HE_ENDPOINT4 local $INT_IP4 ttl 255
# ip link set he-IPv6 up
# ip addr add $CLI_IP6 dev he-IPv6
# ip route add ::/0 dev he-IPv6
# ip -f inet6 addr
```

You need to forward protocol 41 traffic (not port 41 – this is something different) on your home router, although many don't actually let you (but not routers running DD-WRT). Instead, if you're feeling brave, you can put your machine in your home router's demilitarised zone (DMZ), so that all incoming connections are directed to it. This is a grave security risk, though, so don't do this. If you can't make it work, follow the Teredo instructions (*see top, p83*).

One should be aware that your home router's DNS facility may not return IPv6 records. This can be sidestepped by using Google's DNS servers, which you can do by adding **nameserver 8.8.8.8** to **/etc/resolv.conf** or changing the DNS Server setting in NetworkManager. DNS queries carried out over IPv4 are still allowed to return IPv6 addresses, but you can also use the IPv6 address, which is **2001:4860:4860::8888**. Test this with:

```
$ ping6 -c 5 IPv6.google.com
```



› Good news, everyone – we are apparently stable and ready. Actually, we wouldn't quite put it that way...

Router advertising

Besides setting up a static IPv6 address manually, or being assigned a prefix by a broker, as we have seen earlier, you can acquire an IPv6 address in one of two other ways. The first is stateless address auto configuration (SLAAC), wherein a host solicits a router via the Neighbor Discovery Protocol. A local router then responds to this with a network prefix and other configuration parameters. By combining this with the host identifier, either following the MAC address derivation above or using the privacy extensions described later, we obtain an IPv6 address. We can also use the more traditional stateful approach through DHCPv6, where the server keeps tabs on what addresses are assigned to whom.

Teredo tunnels

As an alternative to using a tunnel broker for a single machine, it is possible to use a Teredo tunnel. Teredo is a Microsoft-designed technique, which relies on the bandwidth of so many Teredo relays. These act as gateways, unencapsulating your IPv6 packets and sending them off, and wrapping them in IPv4 packets for return. It is intended as a last-resort technique, and because only one IPv6 address can be assigned per tunnel, it's unsuitable for use on networks, even small ones. Because it creates a tunnel that can happily sidestep your firewall, it is widely seen as an obvious security risk. In general, its use tends to be discouraged and it's being phased out. It is, however, very good at traversing NAT layers (teredo is actually a genus of creatures that bore holes through ship's hulls), so it's handy if your external IPv4 address changes regularly or your network is otherwise unco-operative.

An open source client by the name of *Miredo* is available in your distribution's repositories. Once you've installed it and started the service (**systemctl start miredo** for SystemD), you should be enjoying IPv6. Toredo addresses have a reserved **/32** prefix, and all Toredo addresses have **2001:0:** at the beginning. If all went as planned, you should have been assigned one. You can test this out by visiting **http://test-IPv6.com** which will give you (or at least your IPv6 connectivity) a rating out of 10. You aren't ever going to get full marks with a Teredo connection, and it's also possible that the *Miredo* service got your internal IP wrong. In the latter event, you simply need to edit **/etc/miredo/miredo.conf** and add a line such as:

```
BindAddress 192.168.1.10
```

In order for clients on your network to get IPv6 address information, your router needs to respond to solicitation requests with router advertisements. Without the latter, no one hears the cries of our address-less interfaces. The advertising daemon is found in the **radvd** package, which you need to install. This creates a simple local IPv6 network, with no external connectivity, though it is possible to advertise the brokered tunnels of the previous section too. The example **/etc/radvd.conf** contains many example definitions. To keep things simple, we'll back this up and use a much simpler setup (both tasks require root).

```
# mv /etc/radvd.conf{,.example}
# nano /etc/radvd.conf
```

Now you need to add the following to the configuration – replace **int0** with your interface name:

```
interface int0 {
  AdvSendAdvert on;
  MinRtrAdvInterval 3;
  MaxRtrAdvInterval 10;
  prefix fd00:dead:bad:1dea::/64 {
    AdvOnLink on;
    AdvAutonomous on;
    AdvRouterAddr on;
  };
};
```

If you set up a Hurricane Electric tunnel here, you can use the **/64** prefix assigned to you instead. This lets your IPv6 network see the outside world. You need to enable IPv6 traffic forwarding to make this work with:

```
# echo 1 > /proc/sys/net/IPv6/conf/int0/forwarding
```

Now start the service with either

```
$ sudo service radvd start
```

or

```
# systemctl start radvd
```

depending on what your **init** system is. All going well, connecting other machines to the network now sees them auto-configure themselves through SLAAC, and as an added bonus, you configure the local machine, too. To make this a persistent one, enable the **radvd** service (eg, `$ systemctl enable radvd`) and change the setting

```
net.IPv6.conf.default.forwarding=1
```

in **/etc/sysctl.conf** (or a file such as **/etc/sysctl.d/10-ip6-forward.conf**). Your router advertises to itself as well, so starting up this service statelessly auto-configures you with an IPv6 address. The standard behaviour is to generate a host identifier from the MAC address as described earlier, but some distributions have turned on privacy extensions by default, so that a random identifier is generated (usually once a day). Check how the clients are set up (again replacing **int0** with your interface):

```
$ cat /proc/sys/net/IPv6/conf/int0/use_tempaddr
```

If this returns 0, then privacy extensions are not enabled. To remedy this, as root, incant:

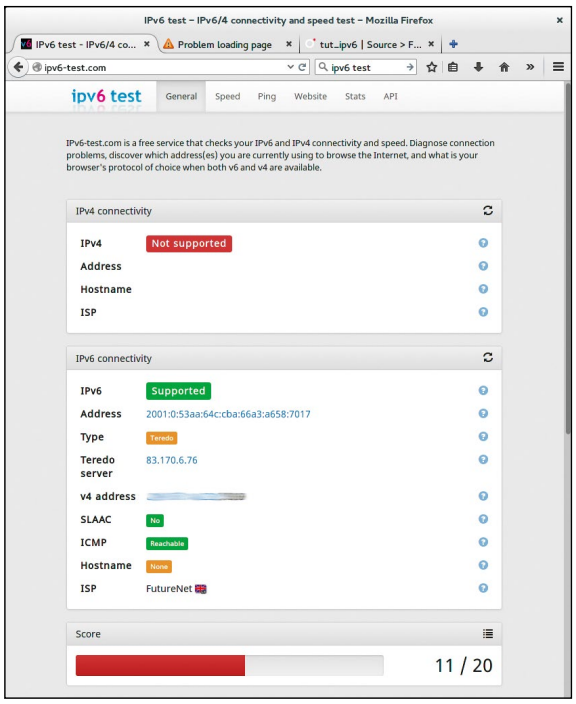
```
# echo 2 > /proc/sys/net/IPv6/conf/int0/use_tempaddr
```

As before, the setting can be made persistent by adding

```
net.IPv6.conf.int0.use_tempaddr = 2
```

say in a file **/etc/sysctl.d/20-ip6-tempaddr.conf**.

Remember, though, that once you get an IPv6 address (either through your ISP, a tunnel broker or Teredo, see boxout above), your host is accessible from the outside world – therefore, any services you may be running on your machine, depending on how they're set up, might be as well. With IPv6, our dependence on NAT is obviated, along with so many frustrations setting up BitTorrent clients. It's prudent to set up some kind of firewall here, but as long as your services are set to only listen on IPv4 interfaces, you should be fine. One slightly chilling side-effect of IPv6 is the emergence of shadow networks. These arise when IPv6 traffic is able to evade security measures which were only set up for IPv4, allowing an attacker to casually sidestep any firewall policies. Hopefully, in the very near future, ISPs will provide IPv6 connections, rendering this tutorial obsolete. We also hope that the internet isn't going to die and winter will be here, soon... LXF



There is little tangible reward for denying yourself the earthly pleasures that are offered by IPv4, but on 6 June 2014 this is exactly what many sysadmins did.

System coding: Processes

In the fifth instalment of **Dr. Chris Brown**'s guide, we explore processes – fork(), exec() and more – and a system programmer's view of pipes.

Our expert

Dr. Chris Brown provides Linux training, authoring and consultancy. He finds his Ph.D. in particle physics to be of no help in this work at all.

Here's a little thought experiment for you to consider, like that thing Schrödinger did with his cat (which we wouldn't recommend you actually do, unless you want a visit from the RSPCA). Imagine you could dismantle your computer, right down to the individual components. You would end up with little piles of wire, connectors, ICs, mounting brackets, screws, even a few resistors and capacitors. One thing you wouldn't end up with is a pile of processes. Which is strange, because if prior to taking your machine apart you had run the command:

```
$ ps -e | wc -l
```

you would have discovered that there were over 100 of them inside the machine.

So what is a process, exactly? And how do you create them? A process is a somewhat abstract concept. A common definition is 'an instance of a program in execution', though personally I don't find that terribly helpful. I prefer to think of a process as a sort of container that holds the context needed for a program to run. To put a little flesh onto that, the table (*below*) shows some of the important attributes of a process.

Attribute	Description
Code Segment	Memory region that holds the executable code
Data Segment	Memory region for global and other static data
Stack	Variable-sized region, holds local variables
Heap	Memory used for dynamic allocation of storage
Process ID	Unique numeric identifier for the process
Real User ID	Numeric identifier of the user running the process
Effective User ID	The user whose permissions the process is carrying
Current directory	Where the process will look for relative pathnames
Environment variables	List of NAME=VALUE pairs used for customisation
Open Files	The state, (eg the file position pointer) of all open files
Signal Dispositions	How the process will handle the various signal types

Quick tip

You can easily find all the named pipes in your system with the command: **$ sudo find / -type p**.

I've discussed some of these attributes in previous articles, and I'll be describing several more in upcoming months. Processes are created using the `fork()` system call. Syntactically, system calls don't come much simpler than this as `fork()` takes no arguments and returns a simple integer result. But in other ways it's the strangest of all system calls because although only one process makes the call, two processes return from it. Confused? An analogy might help. Just as a program is a list of instructions of what to do, so the script of a play is like a list of instructions of what should be said and done. Extending this analogy, an actor is like a process – he or she is the agent responsible for actually carrying out the instructions. You might say that the program is passive, but the process is active.

Think of the children

So our actor is standing alone centre-stage and reads the instruction "fork()", at which point they beckon another actor from the wings to come and join them on stage. The second actor has an identical copy of the script, and indeed, they both start reading from the line immediately following `fork()`. They are separate processes, but they are both, at least initially, executing the same program. The analogy isn't perfect. The new process isn't really waiting in the wings, pre-created and ready for action, it's created by the `fork()` call.

The original process is called the parent, the new one is the child, and to begin with it's an almost exact copy of the parent. However there's one important difference which allows the two processes to figure out which is which after the `fork()` returns – in the parent, `fork()` returns the process ID of the newly created child. In the child, `fork()` returns zero. This means that you invariably see the `fork()` call inside an if test like this:

```
if (fork()) {
  /* I am the parent */
}
else {
  /* I am the child */
}
```

Here's a more complete example in which parent and child both spin round in a loop printing messages:

```
#include <stdio.h>

void main()
{
  int i;

  if (fork()) {
```

Why fork?

Why do programs choose to fork? There are two main reasons. The first is to achieve concurrency within a single application. For example a web, mail or database server might choose to fork in order to serve multiple clients at the same time. It's true that multi-threading (which I don't discuss here) might offer a tighter, more efficient solution; nonetheless, the child-process-per-client model remains popular.

The second reason to fork is what I'll call delegation. The idea is that you want to delegate some job to another program, but you want to keep control so that you can carry on once that job is done. This is precisely what the fork/exec/exit/wait process life cycle is all about, and it's what the shell does every time you enter a command. You want to list a directory, the shell delegates the job to `ls` by exec-ing that program, but it does it in a separate child process so that the shell can keep control and prompt for the next command.

```
    for (i=0; i<100000; i++)
      printf("**    PARENT %d\n", i);
  } else {
    for (i=0; i<100000; i++)
      printf("  **   CHILD  %d\n", i);
  }
}
```

I will not waste space on the page by showing you the output but I'd encourage you to type the code in and try it out. You'll find the PARENT and CHILD lines are interleaved in an unpredictable way, as the scheduler divvies up the CPU time between the two processes. If you run the program again the interleaving will be different. It's a little worrying that the behaviour of this tiny program is non-deterministic [in a deterministic universe? – Ed], but that's what you get once you start introducing concurrency into the mix. In principle you might see output interleaved within a line, maybe like this:

```
**    PA  **    CHILD  5RENT 547713880
```

In practice I doubt you'll ever see the context switching back and forth within a line like this, but the principle remains: the two processes are running concurrently and all bets about what order things will happen in are off.

Sometimes when you run this program, the shell's next prompt appears to be missing. In fact it's just scrolled off the screen. This will happen if the child finishes after the parent, because although the shell waits for the parent process to finish before prompting, it doesn't wait for the child as well. We'll look at waiting for process termination later.

Sometimes parent and child continue to execute the same program [*see the box Why Fork? Above*], but often the child is destined to run a completely different program. It does this with one of a family of system calls which (for want of a generic name) I'll `call exec()`. Returning to our actor and script analogy, an `exec()` is like an instruction in the script that says "Go and perform the play *Macbeth*". So the actor discards whatever script he's currently performing, picks up a copy of *Macbeth*, opens it at page 1, and starts to read "When shall we three meet again? In thunder, lightning, or in rain?" He's the same actor, he's just reading a different play.

The seven versions of exec()

As shown in the decision tree diagram (*pictured right*) In fact, there are seven versions of `exec()`. The seven names are confusing, but follow a pattern:

1 If the name ends in 'e' a new environment is passed; otherwise the old one is retained.

2 If the name of the `exec()` has a 'p' in it, the executable is looked up on the search path; otherwise you need to pass an absolute pathname.

» If the name has an 'l' in it, the command line arguments are passed as an explicit list in the call; otherwise if it has a 'v' in it, they are passed as a vector (array).

Some examples might help:

```
char *argv[] = {"ls", "-l", NULL};
execl("/bin/ls", "ls", "-l", NULL);
execv("/bin/ls", argv);
execlp("ls", "ls", "-l", NULL);
execvp("ls", argv);
```

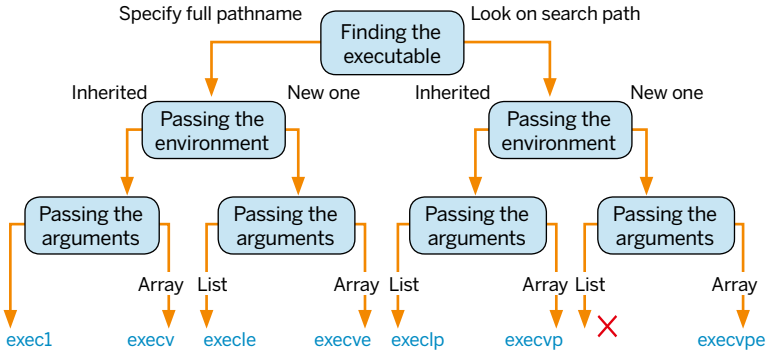
Note that a successful `exec()` doesn't return. The process is now away executing some other program. If the `exec()` does return, then it has failed, probably because it can't find the executable or doesn't have permission to run it.

Process termination

So much for creating processes. How do we get rid of them? The commonest way is for the program to terminate voluntarily by calling `exit()`. This call takes an integer argument which is made available to the parent and is called the exit status. By convention an exit status of zero means success and non-zero (1-255) implies some sort of failure. What constitutes failure depends on the program. eg, in *grep* it fails if it doesn't find a match for the pattern, returning an exit status of 1. This convention confuses C programmers, who expect 0 to be false and non-zero to be true.

The parent can choose to wait for the child to exit by calling `wait(&status)` where status is an integer variable. The call blocks until any of our child processes exits (remember we may have forked multiple times and have many children). The PID of the child that exited is returned as the value of the function. The exit status is returned in the high byte of the status variable and can be conveniently access using the macro WEXITSTATUS. We now know enough to write a simple shell. Yes really! Here it is:

```
1. #include <stdio.h>
```



› The seven variations of exec() are quite bewildering. This decision tree diagram should help you choose the right one.

» **Missed last issue?** Turn to page 68 and grab a back issue now!

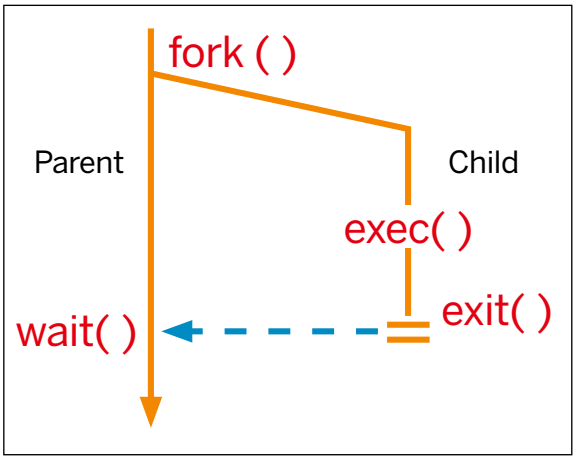
```
2. #include <string.h>
3. #include <stdlib.h>
4.
5. int prompt_and_parse(char **args)
6. {
7.   static char line[100];
8.   printf("> “);
9.   if (gets(line) == NULL)
10.    return -1;
11.  *args++ = strtok(line, " \t");
12.  while (*args++ = strtok(NULL, " \t"))
13.    /* Empty body */ ;
14.  return 1;
15. }
16.
17. main()
18. {
19.   char *args[20];
20.
21.  /* Main command loop */
22.  while (prompt_and_parse(args) > 0) {
23.    if (fork() == 0) { /* Child */
24.      execvp(args[0], args);
25.      printf("%s: not found\n”, args[0]);
26.      exit(1);
27.    }
28.    else { /* Parent */
29.      wait(0);
30.    }
31.  }
32. }
```

Here's a quick guided tour. The function `prompt_and_parse` (lines 5-15) prompts for a command, reads in a line and splits it into tokens, placing them into the array passed as an argument. Disclaimer: the fixed-length line buffer on line 7, and the use of `gets()` opens us up to a buffer overflow attack; it's asking for trouble and should not be used in production code. (The fixed-size buffer at line 19 is also poor practice.) The string-parsing loop at line 12 is a 'Marmite' line – you either love it or hate it. All the actual work is done as side-effects of evaluating the test predicate for the `while()`.

The main command loop (lines 22-31) makes repeated calls on `prompt_and_parse()`, forking each time and having the child exec the command line we just parsed. Note the choice of `execvp()` here. At line 29 the parent merely waits for its child to finish (ignoring the exit status) before looping round again. Also notice the error handling at line 25. We only come here if the `execvp()` failed, presumably because we can't find the executable. Let's give it a whirl. The initial $ prompt is from my original (*Bash*) shell; the others are from *Smallshell*.

```
$ smallshell
> date
Wed May 13 19:50:59 BST 2015
> date -I
2015-05-13
> file smallshell.c
smallshell.c: C source, ASCII text
> ls -l *.c
ls: cannot access *.c: No such file or directory
> cd ..
cd: not found
```

Well, the first three commands worked fine, so our parsing of the command line seems to be OK. The next command runs into trouble because our shell doesn't expand wildcards. Also the final attempt at `cd` fails (the `execvp()` fails) because `cd` isn't an external program, and needs to be built into the shell. But seriously, what do you expect for 30 lines of code?



The four system calls `fork()`, `exec()`, `exit()` and `wait()` are the building blocks of the process life cycle.

Programmer's pipes

Everybody knows what pipes are. They're the most widely used inter-process communication mechanism in Linux and are at the heart of the classic tool-building philosophy where commands are used in combination, eg:

```
$ ps -e | wc -l
```

Let's take a closer look at pipes through the eyes of a system programmer. Pipes are unidirectional. They have an upstream end that you can write to and a downstream end that you can read from. The pipe has a finite size and imposes a loose synchronisation between a producer (upstream) and a consumer (downstream) – the producer will block if it tries to write to the pipe when it's full, and the consumer will block if it reads from the file when it's empty. Pipes are created with the `pipe()` system call:

```
int p[2];
pipe(p);
```

You get back two file descriptors: `p[0]` on the downstream end and `p[1]` on the upstream end. Now in order to use a pipe to communicate between two processes we rely on two things: First, the open file descriptors are inherited by a child process during a `fork()`. Second, the descriptors survive an `exec()` call; that is, if file descriptor 4, for example, is open on the upstream end of a pipe prior to an `exec()`, it will be open in the new program after the exec(). (This is a simplification – you can explicitly set the close-on-exec flag on a file descriptor to stop this, but I won't get into that here.)

I've broken the business of creating and using a pipe down into four stages [*see p87*]:

```
int p[2];
pipe(p);
if (fork()) {
  /* Parent */
  dup2(p[0], 0);
  close(p[1]);
  exec( ... downstream ...);
}
else {
```

Quick tip

In modern Linux kernels the default size of a pipe is 65,536 bytes. However, there's a much lower limit (4,096 bytes) on the amount of data that you are guaranteed to be able to write atomically (ie, all in one go, without fear of it getting interleaved with stuff written by another process). This is important if you have several processes writing to the same pipe.

Adjusting the plumbing

The system calls `dup()` and `dup2()` allow re-assignment of file descriptors and are the tools of the trade of the Linux plumber. Typically they are used to re-assign the standard input or standard output of a process. The call `dup(fd);` duplicates the descriptor fd onto the lowest available descriptor, that is, the lowest one that isn't actually open. So the sequence:

```
fd = open("foo", ...)
close(0);
dup(fd);
```

will connect the standard input to file **foo**. (Remember, by definition, file descriptor 0 refers to the standard input.)

The `dup2()` system call is slightly easier to use because you get to specify which descriptor to duplicate onto:

```
fd = open("foo", ...);
dup2(fd, 0);
```

If the original descriptor (0 in the case) is already open, it will be closed first.

```
  /* Child */
  dup2(p[1], 1);
  close(p[0])
  exec( ... upstream ...);
}
```

Here's the scoop. At stage 1, our parent process (A) creates a pipe, receiving descriptors on each end. At stage 2 the process forks and the child process (B) inherits the descriptors. At stage 3, the parent process (A), which in this example is destined to become the downstream process, copies the downstream end of the pipe onto its standard input [*see the box above, Adjusting the Plumbing* for details on `dup2()`]. It then closes its descriptor on the upstream end. This turns out to be rather important, because when the downstream process subsequently tries to read from the pipe it will block if the pipe is empty as long as at least one process has a descriptor open on the upstream end. Now, of course, process A has no intention of actually writing to the pipe – it's the mere fact of still holding the descriptor open that causes the problem. It's some years ago now but I remember spending a considerable time figuring this out.

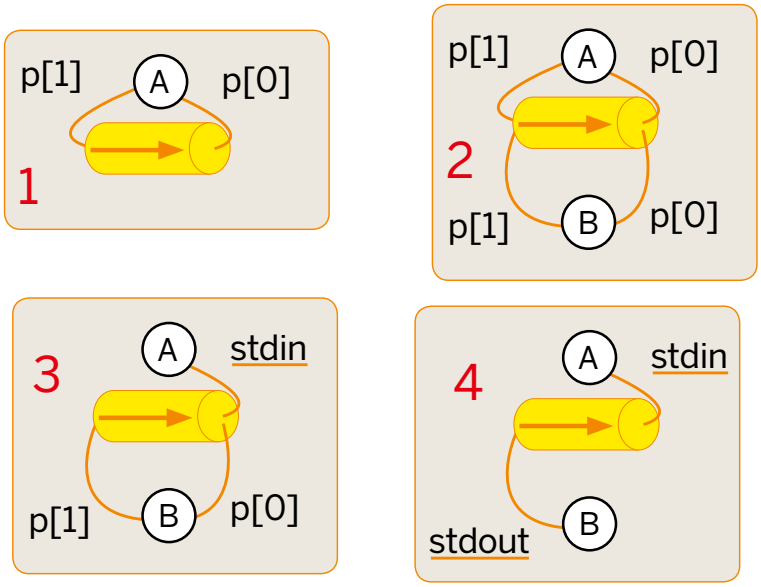
At stage 4, the child process duplicates the upstream end of the pipe onto its standard output and closes the downstream end. Now we're all set to go – process B can write to the pipe and process A can read from it.

My decision to make the parent process the downstream process and the child the upstream process is arbitrary – it could be the other way round. In fact it would be more common to have the parent fork twice and configure its two children as the upstream and downstream processes. But this example serves to illustrate how pipes work. The other point I'd like to make is that usually the child processes have no idea that these shenanigans have been going on in the parent to manipulate their file descriptors – they just read their standard input or write their standard output.

Now in most inter-process communication mechanisms there needs to be some pre-agreed 'rendezvous point' to allow one process to find the other. In the case of a TCP socket for example, the client needs to know the IP address and the port number where the server is listening. As another example, for a System-V style message queue all the processes must agree on the ID of the queue. But the pipes we've been looking at have no name or identifier; they are sometimes called anonymous pipes. They work because the two communicating processes are not really unrelated; they must have a common ancestor that created the pipe and passed the file descriptors down to them.

Jack the pipe

Which brings me neatly to the subject of named pipes. They behave like anonymous pipes but they have an entry in the file system, allowing unrelated processes to communicate, provided they agree on the name of the pipe. From the command line you can create a named pipe like this:



› The four stages of setting up a pipe. See the text for a full explanation.

```
$ mkfifo /tmp/mypipe
```

or within a program, use the `mkfifo()` call:

```
mkfifo("/tmp/mypipe", 0666);
```

In case you were wondering, fifo stands for first in, first out, which is the key characteristic of how pipes are accessed. A named pipe has a permanent existence in the file system. They will show up as type 'p' if you do a long listing:

```
$ mkfifo /tmp/mypipe
$ ls -l /tmp/mypipe
prw-rw-r-- 1 chris chris 0 May 19 18:25 /tmp/mypipe
```

In contrast, an anonymous pipe only exists as long as there is a process with a file descriptor open on it.

From a system programming point of view, named pipes are opened, read and written just like a file, except that for anything useful to happen you need to have one process open it for reading and the other open it for writing. So rather than present more C code, here's a little experiment you can try with named pipes from the command line:

1. Open two terminal windows. I'll call them A and B.
2. In window A, create the pipe **/tmp/mypipe**, as just shown.
3. In window A, run the command `$ cat /tmp/mypipe` . This will block; `cat` is waiting for some other process to open the pipe for writing, and write something to it.

» In window B, run the command `$ cat > /tmp/mypipe` .

» Now type a few lines of text into window B. As cat writes them to the pipe, they'll be picked up by the cat running in window A and displayed.

» Enter `$ ^D` in window B. This will terminate the upstream `cat` . When this happens, the downstream `cat` (in window A) receives an 'end of file' when it tries to read from the pipe (because no-one has the upstream end open) and terminates, returning you to a command prompt.

That's it for now. Next month I plan to talk about process identity and access control. See you then! LXF

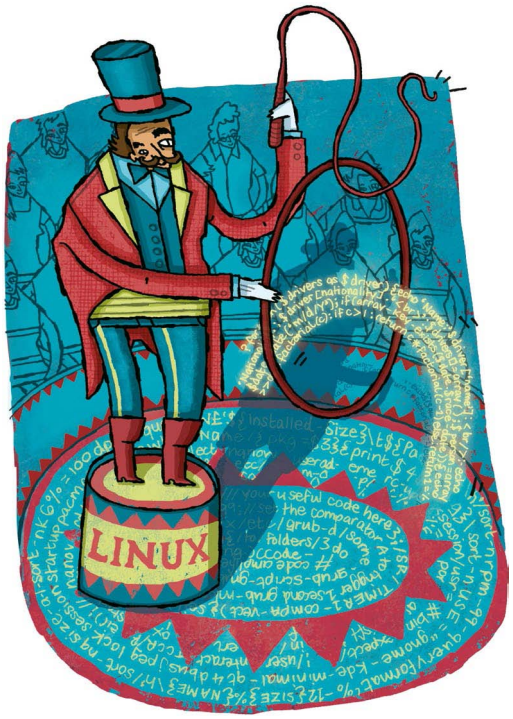
Next issue: Process & access control

PHP: Build a word game

Can't code? Won't code? **Paul Hudson**, former editor of *Linux Format*, has returned to change that by making a game in 80 lines of PHP. What a guy!

Our expert

Paul Hudson writes app coding tutorials at **hackingwithswift.com**, now uses a MacBook and largely communicates in Latin. He thinks Vim is ace.



The last time I wrote for *Linux Format*, Linux Mint 9 was the bee's knees, *Systemd* was just on the horizon, and 2010 was really honestly definitely and absolutely the year of Linux on the desktop. Today, *Systemd* is one of the largest controversies facing our community, Linux on the desktop is still yet to materialise and Linux Mint? OK, actually Linux Mint is still pretty cool – one out of three ain't bad, right?

In this special anniversary tutorial I'm going to teach you some basic PHP by helping you build a command-line word game. The game will ask players to spell out words using a variety of letter titles, rather like the smartphone game *Seven Little Words*. For example, the tiles 'MIL', 'KSH' and 'AKE' spell 'MILKSHAKE' and match the clue 'A cow in a tornado'. To make things easier, the game will tell the player the number of letters in each answer so they don't spend eternity guessing.

Don't worry if you're lacking much coding experience: I've explained everything I have room for, so even beginners should get by. But before you start you do need to install PHP. If you're using a Debian-based distribution (distro), this ought to be as simple as running `apt-get install php5-cli`.

However, that only gets you part of the way there. Other distros ought to have a similar package, but please make sure you install the command line version of PHP rather than the *Apache* module – if your package manager tells you it needs to install *Apache*, it's a safe bet something has gone wrong.

The other part of the installation is to install a package that lets us read user input from the command line. This is not built into the Debian PHP packages by default for some annoying reason, but it's trivial to add. First, run this

```
sudo nano /etc/apt/sources.list
```

and at the bottom of all your other sources, add this:

```
deb http://packages.dotdeb.org wheezy all
```

If you're using a Jessie- or Squeeze-based Debian respin, you can specify `jessie` or `squeeze` in place of `wheezy` and it will work fine. Save and exit *Nano* (Ctrl-o, Enter, then Ctrl-x) then run these commands:

```
wget http://www.dotdeb.org/dotdeb.gpg
sudo apt-key add dotdeb.gpg
apt-get update
apt-get install php5-readline
```

That will upgrade your PHP installation so that it's actually useful for command-line work. You might have thought that was the standard in something called PHP-CLI, but you'd have been wrong. Sigh. Anyway…

Level up

Before we start coding, I want to think through what our levels should look like. Each level will have seven clues, and each clue will have two or three letter tiles that spell out its answer. I don't want to complicate this project by adding the behemoth that is XML, so instead we're going to use a super simple level format:

1. Each level will have seven lines.
2. Each line will contain its letter tiles, a colon, then its clue.
3. The letter tiles are divided up using a pipe symbol: |.

With that in mind, here's what a level file looks like – save this onto your desktop as **level1.txt**:

```
MO|OSE|UM: A cow's favourite place
RO|ADH|OG: A pig that drives dangerously
TRU|NK: Where an elephant packs its luggage
MIL|KSH|AKE: A cow in a tornado
BA|BO|OM: An exploding monkey
BUL|LDO|ZER: A sleeping bull
HIS|STO|RY: A snake's favourite subject
```

Once you've finished groaning from the terrible jokes, it's down to you to create more level files – number them **level2.txt**, **level3.txt**, and so on, and you'll find it's easy to add them to your code later. Whatever you do, make sure the total number of letter tiles adds up to 20 to make things line up.

That level format is designed to be easy for people to read, easy for people to edit, and easy for PHP to parse – ie, for PHP to turn that text into a real game. The fact that it's easy for you to read and edit is important, because every time you read those terrible jokes you lose 1 IQ point. The fact that it's easy for PHP to read is important, because we can make it load the level in just 20 lines of code, including white space. Don't believe me? Here goes!

Parsing with PHP

Every Unix programming language worth its salt is good at text processing, and PHP is no different: you can load a file with just one line of code, you can split up the lines based on colons and pipes using another line of code, and you can remove any extraneous white space using yet another line of code. While it's not quite as similar to line noise as Perl, PHP is definitely concise.

We're going to write a function called loadLevel(). When it's called, we'll need to tell it the name of the level file to load (**level1.txt** is what we just made), but it will do all the hard work of finding clues and solutions in the text and returning it neatly organised.

To make the function work, we're going to need three arrays, or collections of values. The first will hold all the letter tiles, for example, 'MIL', 'KSH', 'AKE', 'RO', 'ADH' and 'OG'. The second will store all the clues, but in doing so it will need to store three things: the clue itself ('A cow in a tornado'), the hint showing how many letters the answer has ('9 letters'), plus the letter tiles that belong to this answer. That last value is needed so that we can remove used tiles from the game when a player guesses correctly. The third array will store just the answers for each level so that we can find them easily.

Before I dive into the code itself, there are six PHP fundamentals you'll need to know:

1. All values start with $. This is not because they are worth lots of money, but if you want to impress your friends here's a neat fact: the $ marker for variables is called a sigil.
2. You can make an empty array by writing two square brackets, like this: []. You can also use that to add a value to an array. PHP arrays count from zero, because... well, actually there is no sensible reason for this other than history.
3. You can read individual items from an array using a loop. We'll be using three different loop types, but I'm not doing this just to hurt your brain – the loops are all useful, honest!
4. The file() function loads a file into an array, where each line of the file is a value in the array. This is perfect for our purposes, but be warned: it leaves a line break on the end of each line, so we'll need to remove that.
5. The explode() function converts some text into an array by breaking on a specific character. We'll be using it first to explode our line on colons, because the left half contains the letter tiles and the right half the clue, then we'll use it a second time because the tiles need to be split up by a pipe symbol: |.
6. When a function wants to return a value to the code that called it, you just write 'return' then the value you want to send back.

There are a couple of other functions that we need, but they are named sensibly so you might be able to look at the code and guess for yourself. Write the following into **words.php** and save it onto your desktop alongside **level1.txt**:

```
<?php
function loadLevel($filename) {
        $tiles = [];
        $clues = [];
        $solutions = [];

        $input = file("level1.txt");

        foreach ($input as $line) {
                $line = explode(":", $line);
                $parts = explode("|", $line[0]);
                $tiles = array_merge($tiles, $parts);

                $solution = str_replace("|", "", $line[0]);
                $solutions[] = $solution;
                $clues[] = ["clue" => trim($line[1]),
"length" => strlen($solution) . " letters", "parts" => $parts];
        }
        shuffle($tiles);
        return ["letters" => $tiles, "solutions" => $solutions,
"clues" => $clues];
}
```

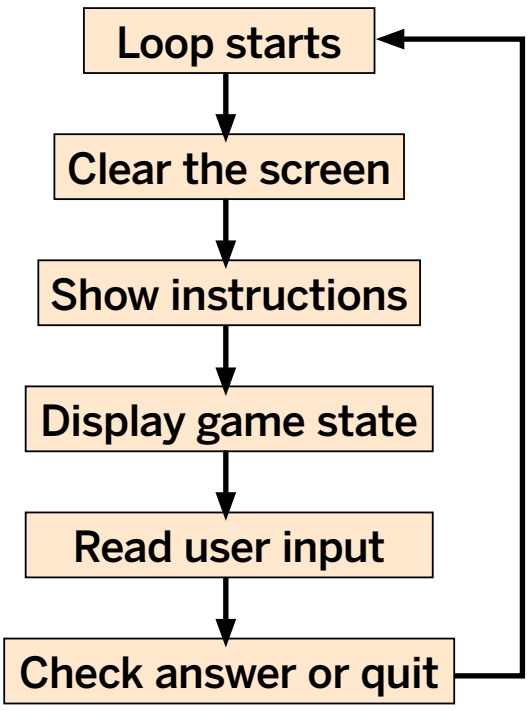
Once that's saved, you can run it from the command line with `php words.php` but unfortunately it won't do anything. It might seem a bit demotivating to write all that code and see nothing in return for all your hard work, but relax: it's coming soon. First, though, I want to explain the other functions just in case your guesses were off:

» **array_merge()** combines two arrays together. We use this to join each clue's letter tiles to the existing set of all letter tiles.

» **str_replace()** replaces text in a value. In our case, we're replacing the pipe symbol (that's |, remember) with an empty value. The code $line[0] means 'the first value in the array', and is what I meant when I said PHP arrays count from zero.

» **trim()** removes any excess whitespace from text values. This is required because the file() function leaves the line break on the end of the values that it reads.

» **strlen()** measures the length of text. »



› We use an infinite loop so that the game doesn't end until the user types quit or rage quits their terminal.

» **Want more PHP coding?** Follow Paul's tutorials on hackingwithphp.com

Comparing against false

The `array_search()` function (*see p91*) tells us where a value can be found in an array, but it has a catch that you need to be aware of: PHP considers a value of 0 to mean false, and a value of 1 to mean true, so writing 'if (1)' is equivalent to writing 'if (true)'.

If `array_search()` finds a value at position 0, it will return 0. If it can't find the value at all, it returns false. Therein lies the problem: if we had written `$position != false` it would mean if $position is false or a value that is equal to false, such as 0". Using `!==` instead (that's two equals signs) makes that condition mean "if $position is false", and means that 0 won't be considered the same as false.

» **shuffle()** randomises the order of an array, which makes the game a little more challenging.

The only complicated thing in the `loadLevel()` function is the way I use arrays, eg the function returns an array like this:

```
return ["letters" => $tiles, "solutions" => $solutions, "clues" => $clues];
```

I already explained that arrays hold value collections, but there are two ways it can do that. The first way is when those values are stored numerically, as used in `$line[0]` and `$line[1]`. The second way is what you can see in the code above, and it means 'rather than use position 0, 1, 2 and so on, give those positions names instead.' So the array we return can be accessed using `$array["letters"]` or `$array["solutions"]`, which is easier to remember. If you were wondering – and I hope you were! – the `<?php` means 'everything from here on is PHP code, and should be treated as such.' You can exit PHP mode with `?>`, at which point anything that's found will just be printed to the screen.

The state of play

The `loadLevel()` function prepares our game to be played, but it doesn't have any game logic. We still need to:

1. Show the current game state. This needs to show all the letter tiles, plus all the clues. Next to the clues we're going to show how many letters are in the answer when it hasn't been guessed, or the correct answer when it has been guessed.
2. Prompt the user to guess until the game is complete.
3. When the user guesses correctly, remove the used letter tiles and update the clues to show the correct answer.

The first thing to do is show the current state of play: all the letter tiles and all the clues. We're going to do this with a new function called `printStatus()`, which receives three values: the array of letter tiles to draw, plus our array of clue data.

```
File  Edit  Tabs  Help
You need to spell seven words from the letter tiles below.
To play, just try to spell a word out of two or more tiles.
You can type 'quit' to exit at any time.

BUL     ZER             OG      MO

RY      OSE     RO      KSH     ADH

        HIS     TRU     MIL     NK

AKE     STO     UM              LDO


CLUES
1. A cow's favourite place (7 letters)
2. A pig that drives dangerously (7 letters)
3. Where an elephant packs its luggage (5 letters)
4. A cow in a tornado (9 letters)
5. An exploding monkey (BABOOM)
6. A sleeping bull (9 letters)
7. A snake's favourite subject (8 letters)

Enter your guess:
```

› With some careful tabs and line breaks, our game status looks easy to understand and easy on the eye. Next stop: colour?

To make the letter tiles print neatly, we're going to use a second type of loop called a 'or loop'. This lets us count through a range of numbers, which in our case will be from 0 to the number of letters there are. We're also going to use a technique called modulus, which is more commonly known as 'division remainder'. A modulus operation means 'what value is left when you divide number X by number Y?' eg, 20 modulus 5 is 0 because 20 divides equally into 5 with no remainder, whereas 22 modulus 5 is 2 because 20 divides into 5 four times, with a remainder of 2.

Modulus is useful for this function because we want to lay out the letter tiles neatly. In combination with the for loop, we'll count from 0 to the number of letter tiles (which is 20), and print each tile to the screen followed by a tab character to space them out. Then, using modulus, every time the current number in our loop divides by 5 with a remainder of 4, we'll add two line breaks. In practice, this will give us a layout showing a grid of five tiles across and four down. Printing the clues is straightforward in comparison: we'll use a second for loop to count from 0 to 7, reading out the 'clue' and 'length' value for each clue. You need to know three more things before getting into the code:

1. To add one to a number, you write `++$yourNumber.++`, which means 'add one to this number, then store it back in the same value.'
2. If you want to write a tab in some text, you use the special value `\t`. For new lines, you use `\n`.
3. Accessing arrays inside text can be messy, so PHP prefers you to put your values inside braces. For example: `"Here is a value: {$array["myValue"]}"`.

Now for the code – place this into your PHP script below `loadLevel()`:

```
function printStatus($letters, $clues) {
        for ($i = 0; $i < count($letters); ++$i) {
                echo "{$letters[$i]}\t";

                if ($i % 5 == 4) {
                        echo "\n\n";
                }
        }

        echo "\nCLUES\n";

        for ($i = 0; $i < 7; ++$i) {
                echo $i + 1, ". {$clues[$i]["clue"]} ({$clues[$i]["length"]})\n";
        }

        echo "\nEnter your guess:\n";
}
```

The most complicated line of code is inside the clue loop, because it has to find particular pieces of information within a big array. `$clues[$i]["length"]` means 'get the $clues value, find the item at position $i, then pull out its length value,' but it does so all in one line.

Now, I have some good news and some bad news. The bad news is that if you run your script, it'll still print nothing. But the good news is that you are now only a few minutes away from things starting to come together, but to do that you need one more loop type...

So far we've written two functions, but they happen to be the two that form the backbone of our own program. To make

Variables in strings

PHP makes extensive use of a technique called string interpolation, which means you can write 'Foo bar $variable meh' and have $variable replaced with its value inside the text. This is clearly very useful, but it has problems: how can PHP know where your variable ends and text begins? In the previous example, it was clear: PHP variables are not allowed to have spaces in them, so when PHP spots the space after $variable it knows it's looking for $variable. When it comes to arrays, you would use $variable["wombat"], but does that mean 'the contents of $variable followed by the text ["wombat"],' or 'the contents of the "wombat" value inside the $variable array'? PHP has a simple solution for this, and it means wrapping your variables inside braces: { and }. This isn't needed for simple things like $variable, but it is required in our code because we're using $clues[$i]["clue"] and its meaning is ambiguous. When you find yourself in this situation, just put braces around your variable like this: {$clues[$i]["clue"]}, and the brackets will clear up the ambiguity perfectly.

the whole game limp into action, we need to execute the loadLevel() function and store its return value, then start asking users to guess.

To make this work, we need a new loop type called an infinite loop. This is a loop that runs forever, or at least until we tell it to stop, which is perfect for a game where the user keeps guessing until they win. Inside this loop we need to clear the screen, print out the game instructions and current game status, then read some player input and act on it.

We're going to return to the player input shortly, but for now we're going to take the player input and remove any spaces from the start or end, then convert it to uppercase. This is important because we don't really care whether users type 'milkshake', 'MILKSHAKE' or any combination in between. Once this is done, we look for two special cases: if the input was empty (ie, if the user just hit Return) then we will continue the loop from the beginning, and if the input is 'Q' or 'QUIT' (uppercase, remember!), we'll exit the program.

To make things a little easier to read, this next piece of code will split the return value from loadLevel() up into three separate values. It's not necessary for the computer, but it is necessary to maintain your sanity – trust me on this!

Anyway, put this code at the end of your PHP script:

```
$currentLevel = loadLevel("level1.txt");
$letters = $currentLevel["letters"];
$solutions = $currentLevel["solutions"];
$clues = $currentLevel["clues"];

while (true) {
system("clear");
    echo "\nSEVEN LINUXY WORDS\n";
    echo "You need to spell seven words from the letter tiles below.\n";
    echo "To play, just try to spell a word out of two or more tiles.\n";
    echo "You can type 'quit' to exit at any time.\n\n";
    printStatus($letters, $clues);

    $line = strtoupper(trim(readline()));
    if ($line == "") continue;

    if ($line == "QUIT" || $line == "Q") {
        echo "Thanks for playing!\n";
        exit;
    }
    // check answer
}
```

There are several things that deserve extra explanation.

First, the infinite loop is triggered by having while (true), which translates to 'for as long as true is equal to true, run this code.' True is of course always equal to true, so this loop runs forever. Second, to clear the terminal screen we use the system() function and pass it 'clear'. This acts as though the user had typed 'clear' themselves, and it's a great shortcut.

Third, the strtoupper() function is new, but it's also trivial: it just converts text to uppercase and returns it. Fourth, the readline() function is also new, but it's also wonderfully self-descriptive. Fifth, || inside an if condition means 'or', so this line means 'if the user's input is Q or is QUIT...' Finally, near the end is a line of code that says // check answer. PHP ignores anything starting with '//', as this is a comment.

Searching for answers

If you run the code now, you'll see things are actually working now – and about time too, quite frankly. But the game isn't really a game yet: you can type whatever you like and it just refreshes the screen as if nothing happened. But you are, thanks to some cunning shortcuts, just minutes away from finishing this game and sitting back with a big grin on your face. All that's left to do is to replace that // check answer comment with the actual code to check whether the player's answer is good. If the answer is good, then we need to remove its tiles from play, update the clues list to show the answer, then stop the player using the answer in the future.

To make this work we're going to use a new function called array_search(), which tells us where an item exists in an array, if at all. We already have our $solutions array, which contains all the possible answers, so we use array_search() to look for the user's answer in that array. And here's where the neat shortcut comes in: the position of answers in our $solutions array is identical to the position of that solution in our clues array. This means to update the clues list to show the answer rather than the word length, we need to change the clue at the right position in the array. Then to remove the tiles that were used, we'll use a similar solution: use array_search() to find each of the tiles in the answer inside the master $letters array, then set it to be some empty text so that it's no longer visible. This code replaces the comment:

```
$position = array_search($line, $solutions);

if ($position !== false) {
    $clues[$position]["length"] = $solutions[$position];

    foreach ($clues[$position]["parts"] as $letterPart) {
        $letterPosition = array_search($letterPart, $letters);

        if ($letterPosition !== false) {
            $letters[$letterPosition] = " ";
        }
    }

$solutions[$position] = "";
}
```

That's it! Your game is finished, and ready to play. LXF

Quick tip

You've got a full game on your hands, but there's much more it can do. At the very least, have a go at making another level, then modify the code to change levels when the player guesses all the words. When that's done, try keeping score: add 10 for a correct answer, and subtract 1 for an incorrect one.

Next issue: Web apps in Haskell

Answers

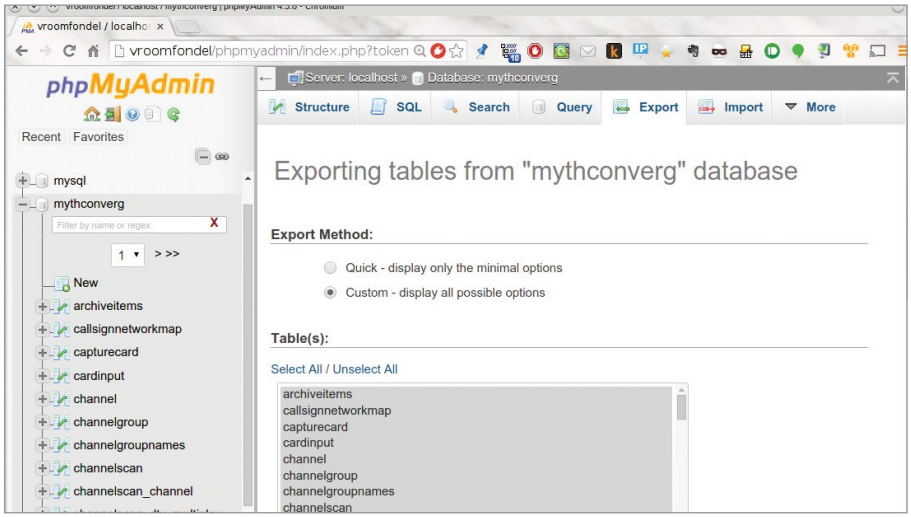
Got a question about open source? Whatever your level, email it to **lxf.answers@futurenet.com** for a solution.

This month we answer questions on:

1 Database preservation

Q By the time you read this email Ubuntu 15.04 will be out. Although I like to keep up to date with the new versions of the distro, one big gripe I have is searching for the latest instructions to restore my LAMP stack and *MySQL* database. These seem to change every time I upgrade, so if you are building a database this becomes an unwanted distraction and one which I think I will forgo this time. Why make life difficult in this way?

Mike Higgins



› Backing up a MySQL database the GUI way, with phpMyAdmin.

A It is fairly easy to backup a *MySQL* database with *mysqldump*, and restore it later. Using *phpMyAdmin* makes it even easier. However, for a server there is another important point. We often recommend using a separate partition for **/home** so that user files are not touched by a reinstall or update. For a LAMP stack, the home directories of the various programs are in **/var**. This means that when installing for such a system, a separate **/var** filesystem makes a lot of sense. That still leaves the settings files in **/etc** to worry about, but they are small and easily backed up, for the LAMP part you need to keep copies of **/etc/apache** and **/etc/mysql** (assuming the A in LAMP is for *Apache* on your system and not '*another* web server'). If you want to backup your database with *mysqldump* you can use this command:

```
$ mysqldump -p --all-databases --add-drop-database --create-options --opt --allow-keywords --flush-logs --hex-blob --max_allowed_packet=16M --result-file=mydatabases.sql
```

This produces a file of SQL statements that you can pass back to *MySQL* to recreate your databases, like this:

```
$ mysql -p <mydatabases.sql
```

The SQL files can get pretty big so you may want to omit the `--result-file` option and pipe the output through *gzip*:

```
$ mysqldump -p --various-options | gzip >mydatabases.sql.gz
$ zcat mydatabases.sql | mysql -p
```

Run `mysqldump` before reinstalling then copy the file somewhere safe so you can import it into your new system. You'll also need to backup the contents of **/var/www**.

An alternative is to use the `dist-upgrade` feature of `apt-get` to update your distro to the latest release without reinstalling. This will

Enter our competition

Win!

if ($letter == winner) get $books

Get your questions answered and exploit our generosity.

Linux Format is proud to produce the biggest and best magazine about Linux and free software that we can. A rough word count of LXF193 showed it had 55,242 words. That's a few thousand more than *Animal Farm* and Kafka's *The Metamorphosis* combined, but with way more Linux, coding and free software (but hopefully less bugs).

That's more than most of our competitors, and as for the best, well... that's a subjective claim, but it's one we're happy to stand by.

As we like giving nice things to our wonderful readers, the Star Question each month will win a copy of Martin O'Hanlon's *Adventures in Minecraft*, and it's a great intro to fun Minecraft API projects for you or a youngster. For a chance to win, email a question to **lxf.answers@futurenet.com**, or post it at **www.linuxformat.com/forums** to seek help from our very lively community of readers.

› See page 94 for our star question.

Terminals and superusers

We often give a solution as commands to type in a terminal. While it is usually possible to do the same with a distro's graphical tools, the differences between these mean that such solutions are very specific. The terminal commands are more flexible and, most importantly, can be used with all distributions.

System configuration commands often have to be run as the superuser, often called root. There are two main ways of doing this, depending on your distro. Many, especially Ubuntu and its derivatives, prefix the command with `sudo`, which asks for the user password and sets up root privileges for the duration of the command only. Other distros use `su`, which requires the root password and gives full root access until you type *logout*. If your distro uses `su`, run this once and then run any given commands without the preceding `sudo`.

update all packages, and the distro release version, to the latest and switch your sources setting to use the repositories for the latest release. Open a terminal and run:

```
$ sudo apt-get update
$ sudo apt-get dist-upgrade
```

The first command just informs your system of the latest packages available, the second one performs the actual upgrade. If you want to see what it will do without actually touching your system, add `--dry-run`:

```
$ sudo apt-get --dry-run dist-upgrade
```

This should leave all your data and settings untouched, although it never hurts to have a backup, whether you are updating or not.

2 Black screen of nothing

Q I updated my SolydX (from an *LXF* cover disc) system, running on an EeePC, including a kernel update from 3.13-1-486 to 3.16.0-4-586. The distro is based on Debian with the LXDE window manager. There were a couple of error messages involving signatures, so I installed the Debian signatures as advised on one of the help forums. Now, the thing boots OK, but when it gets to where the login splash screen is supposed to display, all I get is a black screen. I can boot into the previous version and all is OK. I want to roll back to the previous kernel version, but can't find anything in the apt man pages that helps.
Greg

A Kernel packages are different from most others in that installing a new version doesn't remove the old one. You have already found that you still have the older kernel and just need to make it the default for booting. If you look in **/etc/default/grub** you will find a setting for `GRUB_DEFAULT` that sets the menu item to boot by default. It is currently set to 0 as Grub counts the menu entries from zero. So to boot the third item you would set `GRUB_DEFAULT=2`. Then run `grub-update` to rebuild the menu with the new setting.

However, it would be better to try and solve the problem of the GUI not working with the new kernel. When you get the black screen, press Ctrl-Atl-F1 (or wait a few minutes) to get to a console login. Log in as your normal user then you can examine the X log file at **/var/log/Xorg.0.log**. Look for any lines starting with [EE]. You can copy the file to your **home** directory and then reboot into the older kernel, enabling you to read the old log and search the web for pointers. The first error line is usually the most significant, subsequent errors may only be a consequence of the first.

The most likely cause is that your video drivers no longer work with the new kernel and need to be reinstalled, this is a particular issue with out of tree drivers like the official Nvidia ones. Check that the symlink at **/usr/src/linux** points to your new kernel before reinstalling any drivers, as the driver packages often use this to determine which kernel to install for.

3 New user confusion

Q Having got past 80 years, the toll on my brain cells is telling. I'm now up to my ears with the fallout of the latest MS 'lost the plot' age. I got caught in the panic of Windows XP getting the boot, and bought a Toshiba 15.6-inch laptop with Windows 8 on it. I've since then made several attempts to make use of it. The only way I can improve Windows 8.1 is to delete it.

I want to install Linux, but I want to remove Windows 8.1 and start with a clean PC. All the Linux guides refer to backing up files first. I've got DVDs stacked 6cm high, and raring to go but why should I back up when there's nothing to back up? Or is there? I'm very cautious about this, as a friend who has a refurbished PC which had Windows 7 Professional on, was given a Linux magazine DVD and after he attempted an install he has ended up with a PC that won't boot up any more!
Arthur Lawrance

A The advice to back up files is based on the assumption that you have data on the computer that you want to keep. So save any documents, photos or anything else you want to a DVD or USB drive. It's very unusual for an installation to break the bootloader, but if it does you have nothing to worry about as there was nothing on the computer you wanted to keep, anyway. The one issue you are likely to have with a Windows 8 system is Secure Boot. The simple solution is to go into the computer's early boot menu – the manual will tell you which key to press to get to this – and turn off Secure Boot in the Boot section.

Once you've done that you can pick a Linux distribution (distro) that looks right for you and boot from its DVD. When you get to the question about disk usage, tell it you want to »

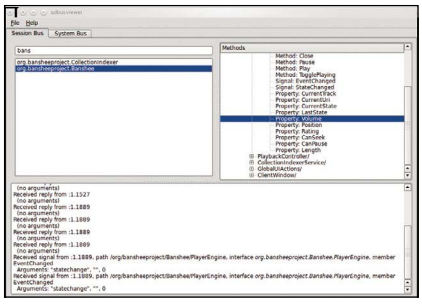
A quick reference to...

D-Bus

D-Bus, or Desktop Bus to use its full name, is a low level inter-process communication system used by Linux and other Posix operating systems. It is a means for programs to send messages to one another, either to ask for information or send commands or requests. There were previously at least two incompatible systems in use and they are KDE's DCOP and Bonobo used by Gnome.

D-Bus is heavily influenced by the design of DCOP but is desktop agnostic. In fact, despite the name, it's not reliant on any desktop at all. There are generally two buses in use on a desktop PC, a system bus used by root programs such as daemons, hardware detection software and the like, and a session bus used by the desktop.

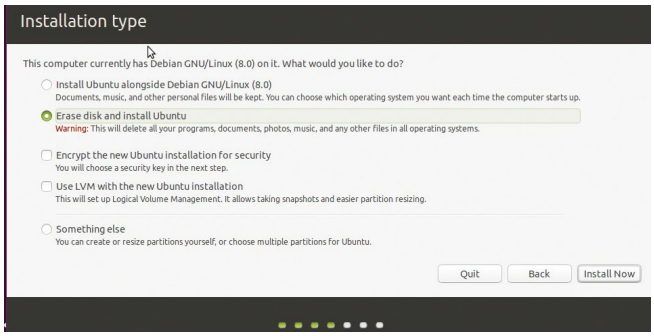
There is far more to it and many programs are using it behind the scenes, from hardware detection to system notifications. The commands used to control it from the shell may look unwieldy, what with the long service names and object paths, but that is necessary if all programs are able to provide a D-Bus interface with none of them clashing. Using *qdbusviewer* to browse the tree, run commands and view the status is the easiest way to learn. If you want to access a program's D-Bus interface through a script, qdbus is the easiest to use of the options. There is also *dbus-send* but that works at a lower level and will require some man page reading (and possibly some head scratching) before you can use it to do what you want.



› **The secrets of D-Bus revealed!**

» use the whole disk, then it will erase Windows completely before installing itself. Many systems come without Windows discs now, but have an option to create a restore DVD. I would run this before getting rid of Windows, if only because restoring it could make life easier if you need to return the laptop under warranty.

That only leaves the question of which Linux distro to choose. Most of the mainstream distros: Ubuntu, openSUSE, Fedora, Linux Mint, Debian, are suitable for new users but expect some learning curve – it's not only the name that's different about Linux. If you have a Linux-using friend, it would be best for you (if not for them) to choose the distro they use, then you can ask/pester them for help when you need it.



› **If you want to get rid of Windows when installing Linux, distro installers have an option to erase the whole disk to start afresh.**

4 Cron flakes

Q Having recently updated to Ubuntu 15.04, I am finding my way around *Systemd*. One part that confuses me is the use of *Systemd* timers to replace *Cron*. I have quite a few tasks that I run via *Cron*, do I need to stop using *Cron* and convert these to use *Systemd* timers? If so, how do I go about it?

George Walker

A First of all, there is absolutely no need to stop using *Cron*. It works exactly the way it always has and there is no need to switch if it suits your needs. Timer units are more powerful than standard *Cron* entries, but the process of using them is more complex. Instead of having everything defined in a single crontab file, you need two separate files for each cronjob. These are standard *Systemd* unit files that you should save in **/etc/systemd/system**. The first has a .timer extension and describes when to trigger an action. Here's an example, called **ssd-trim.timer**, that launches an SSD trim operation twice a week, on Monday and Thursday. First the timer file:

```
[Unit]
Description=Trims the root filesystem
[Timer]
OnCalendar=Mon,Thu *-*-* 10:00
[Install]
WantedBy=basic.target
```

This looks much like any other using file, except for the `OnCalendar` setting. This takes a string of the form:

```
Weekday Year-Month-Date
Hours:Minutes:Seconds
```

The `Weekday` and `Seconds` are optional, and the individual items can be lists or ranges, much the same as crontab. So this time will run at 10am every Monday and Thursday, but what will it run? When a timer is triggered, it starts a service with the same name (you can use a different name with the Unit option), and here is **ssd-trim.service**.

```
[Unit]
Description=Trims the root filesystem
[Service]
Type=oneshot
ExecStart=/sbin/fstrim /
```

If you omit the Type setting, it defaults to simple. For most standalone cron jobs, it makes no difference which you use. You start and enable the timer units just like any other. If you have a lot of entries in **/etc/cron** that are hourly, daily etc, you won't want to be creating separate timer and service files for them all, so create a **run-hourly.timer** like this:

```
[Unit]
Description=Hourly timer
[Timer]
OnBootSec=3min
OnUnitActiveSec=1h
OnCalendar=hourly
Unit=run-hourly.target
[Install]
WantedBy=basic.target
```

Star Question **Winner!** **This month's winner is Rojj. Get in touch with us to claim your glittering prize!**

★ Networking basics

Q I'm using a laptop running Debian, named 'Lenovo'. Upstairs I have a desktop machine also running Debian named 'Debian'. Both connect to my BT HomeHub 4 using Wi-Fi and both use the Xfce desktop. I want to be able to save docs from one to the other. Basically I want to use the upstairs machine as a server. Currently, the two machines do not detect each other.

Rojj

From the forums

A First you need to make sure that your two computers are aware of one another. Run `$ ping -c 3 Debian` from your laptop and you should see a few lines of ping responses. If you get an 'unknown host' error, try replacing the name of the computer with its IP address, which you can get from your desktop's network configuration tool or the status pages of the router. If that works, your computers don't know about each others' hostnames. You could stick with IP addresses, but it isn't very friendly, so edit the file **/etc/hosts** on each system, as root, and add

```
IP-ADDRESS Lenovo
IP-ADDRESS Debian
```

replacing `IP-ADDRESS` with the address of each machine. Now you should be able to ping them by name. In order to share files you need to set up network shares. You have two main choices here, NFS and *Samba*. NFS is the traditional Unix network file shares while *Samba* uses the Windows CIFS filesystem. Both have their pros and cons, but *Samba* enables you to share files with Windows systems too and has a web-based configuration tool. You need to install *Samba* on the server, then point your browser at **http://localhost:901** to configure it.

NFS is set up with a text file at **/etc/exports** containing one line for each folder:

```
/mnt/music 192.168.1.0/24(no_root_
squash,no_subtree_check,rw,sync,fsid=5)
```

This exports the **/mnt/music** directory to all hosts on the **192.168.1.*** network, change this if your router uses something different. The other options are explained in the *exportfs* man page but can be left like this, apart from the fsid setting. Later versions of NFS require a unique ID for each export. To apply the changes you've made run:

```
$ sudo exportfs -r
```

On the client side, you can browse NFS and Samba shares from most file managers, or you can add them to **/etc/fstab** for permanent mounting.

```
Debian:/mnt/music /mnt/music nfs
soft,noauto 0 0
//Debian/music /mnt/music cifs usernam
e=xxx,password=yyy,noauto 00
```

The first line mounts an NFS share, the second a Samba share. For NFS we give the source as the shared path, while for Samba each share has a name that we use. The `noauto` option means the shares are not mounted when the system boots, which would cause delays when you're not connected to your home network.

The `OnBootSec` option causes it to wait until three minutes after booting before trying to run anything. The next line tells it to only run a unit if at least an hour has passed since the last run, to avoid a reboot causing extra runs. This calls `run-hourly.target`, as set by `Unit=`, which contains this:

```
[Unit]
Description=Hourly timer target
StopWhenUnneeded=yes
```

Then create the directory **/etc/systemd/system/run-hourly.target.wants** and drop any number of service files in there, each of which will be run every hour. Do the same for daily and weekly cron jobs.

5 Firefox follies

Q *Firefox* has gone mental on me. The immediate problem is that the URL bar has taken on a life of its own. I click on a long URL then modify it in the URL bar but when press I Enter it reverts to the previous URL. So I copy-paste into a text editor, tweak the URL, copy-paste back into *Firefox*. But what pastes in is the original URL all over again! I purge the History cache of the last hour, close *Firefox* and reopen. Same again. In a similar animate-or-bust way it resizes the window in a slide-it-around kind of way when I open the config options, readjusts tab widths when I open and close stuff similarly.

Despite having played with classic look add-ons and stuff, every damn widget in the whole UI seems to individually need telling to stop messing with my head. Is there any kind of hand grenade I can throw into the works to stop the whole insane mess?

guy

From the forums

A The usual first step to take when a program starts misbehaving like this is to reset it to its default settings. Firefox stores its settings in **~/.mozilla/firefox**, the file **profiles.ini** contains a line like

```
Path=somerandomstring.default
```

while the directory **somerandomstring.default** contains all your settings, bookmarks and other data. By renaming **profile.ini** and starting *Firefox*, it will create a new **profile.ini** and a new **anotherrandomstring.default** directory with everything reset to the defaults. This should fix the problem, but you will have lost your settings and bookmarks. Now you can start copying files, a few at a time, from the old settings directory to the new one. Make sure *Firefox* is not running when you do this, then restart it and see whether things still work as you want. By a process of elimination, you can find the cause of just about any *Firefox* problem this way.

The alternative approach is to back up your **somerandomstring.default** directory and then start experimenting with the *Firefox* settings. You can see all of the settings by typing **about:config** into the address bar. Use the search box to narrow this down to those containing 'urlbar,' they are the settings that control the behaviour of the address bar. You could just start changing settings at random, or you could compare the settings you currently run with the defaults (take a screenshot of each to view them side by side). You can change any setting by double-clicking it, this will toggle a boolean (true or false, yes or no) setting and prompt for input for other types. This is not the first time we have heard of *Firefox* settings seemingly changing of their own accord, so it is a good idea to keep a backup of your settings – but probably not the whole directory as that can get very large, but at least the **prefs.js** file, and maybe your bookmarks too. You can also run *Firefox* with no extensions, to check if one of them is the cause of your problems:

```
$ firefox-bin --safe-mode
```

Run this in a terminal. LXF

Help us to help you

We receive several questions each month that we are unable to answer, because they give insufficient detail about the problem. In order to give the best answers to your questions, we need to know as much as possible.

If you get an error message, please tell us the exact message and precisely what you did to invoke it. If you have a hardware problem, let us know about the hardware. If Linux is already running, you can use the *Hardinfo* program (**http://bit.ly/Hardinfo**) that gives a full report on your hardware and system as an HTML file you can send us.

Alternatively, the output from *lshw* is just as useful (**http://ezix.org/project/wiki/HardwareLiSter**). One or both of these should be in your distro's repositories. If you are unwilling, or unable, to install these, run the following commands in a root terminal and attach the **system.txt** file to your email. This will still be a great help in diagnosing your problem.

```
uname -a > system.txt
lspci >> system.txt
lspci -vv >> system.txt
```

Frequently asked questions…

rsync and Unison

› What is this *rsync* program that people talk about?
It is a way of synchronising the contents of two directories, ensuring one is an identical copy of the other.

› Can't you use cp for that?
Yes, but `cp` copies everything while *rsync* only copies files that are different. With large files that have changed, it only copies the changed parts.

› That sounds good, how do I use it?

```
$ rsync --archive --delete /path/to/source/ /path/to/dest/
```

will ensure that the second directory contains an exact copy of the first. The `--delete` option removes files not present on the first and `--archive` causes all file permissions and time stamps to be copied too. The trailing slashes are important with *rsync*, they indicate that you want to synchronise the contents of the directories. If you omit them, you could end up copying one directory into another.

› What if I want to synchronise with a directory on another computer?
Provided you have SSH access to the other computer, you can use

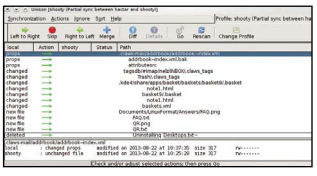
```
$ rsync --archive --delete /my/local/site/ hostname:/path/to/site/
```

When one or both of the paths contains a computer's hostname, *rsync* uses a remote shell program for the transfers. This is SSH by default, but you can change it to something else – even to RSH if you want everyone to be able to read your files when they are in transit!

› What if I have changed files on two computers and want to keep both up to date?
You need something designed more for that task, such as Unison (**www.cis.upenn.edu/~bcpierce/unison**). *Unison* uses *rsync* and SSH for the transfers, so you keep all of the above benefits, but is designed for two-way synchronisation. If you have changed some files on one computer and some on another, it will ensure each computer has the most up-to-date version.

› That sounds very clever, but what if I edit the same file on both computers, I bet it's not so smart then?
It can't read your mind and know which version you want to keep, but it can tell that both copies have been changed and ask you which one to use. *Unison* keeps track of all changes so it knows if a file has been changed on both computers since it was last run.

› This sounds a bit complex, are there a ton of command line options to learn?
There are, or you could use the GUI and just point and click to select. The GUI only runs on the computer initiating the transfer, so you can still use it to sync with your remote computer, providing you are able to install *Unison* there in the first place.



› Keeping a laptop and desktop in sync with *Unison*.

On the disc

Distros, apps, games, books, miscellany and more...

The best of the internet, crammed into a phantom-zone like 4GB DVD.

Distros

When *Linux Format* was launched – way back when the cover disc was a single CD and most distros didn't even need that much space – by LXF20 we had added a DVD option, with owners of older hardware being kept happy with two CDs. In those days an install disc was basically an installer and a software repository, you picked from the available packages during installation.

All that changed when the installable live CD appeared; first with PCLinuxOS. Now you could try out the distro without touching your hard disk, then install it if you liked it – all from the same disc. Install discs became smaller again, and for years Ubuntu had a policy that its base distro should fit on a CD. Once it stopped thinking like that, its discs grew, but most install discs remain between one and two CDs in size. That means we can bring you three or four high-quality distros every month, almost all of which can be tested from the DVD and installed only if you like what you see.

CDs may be long gone, but readers still ask for 32-bit distros, so we'll continue to include them for as long as the distro builders provide them, even if the number that do supply them is falling.

Neil

» Important NOTICE!

Defective discs

In the unlikely event of your *Linux Format* coverdisc being in any way defective, please visit our support site at **www.linuxformat.com/dvdsupport** for further assistance. If you would prefer to talk to a member of our reader support team, email us at **discsupport@futurenet.com** or telephone +44 (0) 1225 822743.

The forward-looking distro

Fedora 22

64-bit

Fedora is an important Linux distro, even if you don't use it. Red Hat are one of the major contributors to the Linux ecosystem, both in the kernel and userspace, and Fedora is its proving ground for new ideas. Gnome 3 and *Systemd* are in widespread use now and both were introduced in previous Fedora releases. *Wayland*, the perpetual next-generation graphics system, also appeared in a previous Fedora release. So Fedora is worth watching if you're interested in what is coming next – and in some ways it is the opposite of last month's lead distro, Debian.

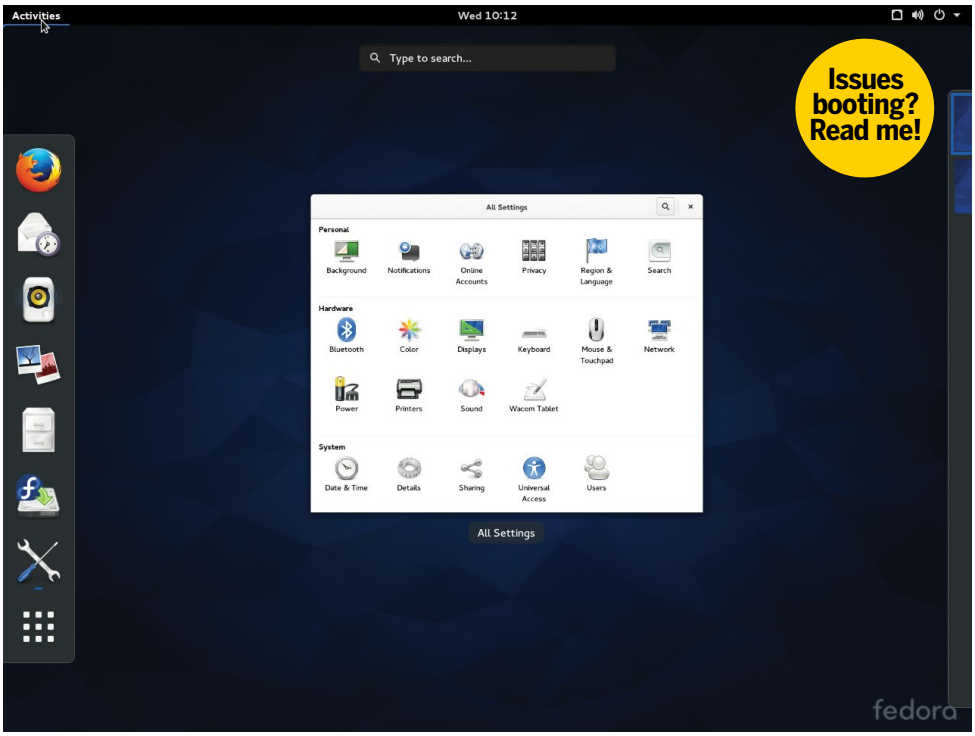
Fedora 22 continues the new software trend, with *Wayland* now being the default graphics system loaded by *GDM* (it falls back to X if your hardware is not supported by *Wayland*). This is the Gnome version, Fedora's default desktop, but all the desktops have been updated with the most major change being KDE updating to *Qt5*, you can install a KDE desktop once Fedora is on your hard drive. Behind the scenes, Fedora 22 uses the 4.0 Linux kernel and several improvements to the kernel packaging, installation and update processes have been made behind the scenes.

So if you want to see where Linux is heading in the near future, or at least one of the possibilities, you should try out Fedora 22. Because Red Hat are big in the server space, where virtualisation is so important now, Fedora is well suited to trying in a virtual environment, such as *Qemu* or *VirtualBox*, although you probably won't see the benefits of *Wayland* if you try it this way.

Naughties Nvidias

The Fedora and Sabayon live systems may have problems booting if you have an older Nvidia graphics card. This is because they both come with the newer drivers that no longer work with some cards. The solution is either to wait for the auto-detection process to time out, which can take several minutes, or to use the Safe boot option. The Safe option is often the best choice for Fedora while booting Sabayon may require a little patience.

This only applies to some cards, and only to the live disc. Once the distro is on your drive, you can install the correct drivers for your card. This isn't the fault of the distros, or our DVD, but is down to the way Nvidia manage its proprietary drivers.



› **Fedora 22 is an excellent choice if you plan to try out lots of virtual environments.**

New to Linux? Start here

» What is Linux? How do I install it?
» Is there an equivalent of *MS Office*?
» What's this command line all about?
» How do I install software?

Open **Index.html** on the disc to find out

Are you reading this on a tablet?
Download your DVD from **www.linuxformat.com**

32-bit edition – like it's 2004

Ubuntu 15.04

32-bit

Over the years of **LXFDVD**s, there have been two consistently popular distros with our readers. The first was Mandrake, which morphed into Mandriva and then went through several bouts of death throes before finally succumbing (although some of its spirit lives on in Mageia). The other is the young upstart and became popular amazingly quickly, Ubuntu. It's hard to believe that the first release was in 2004, version 4.04. At two releases a year, that makes 15.04 the 23rd Ubuntu release.

The out-of-the-box

Sabayon 15.06

64-bit

If you want to try the latest software, it helps to use a rolling release distro; those with set release versions tend not to update package between releases except for bug fixes. Arch and Gentoo are both popular for this, but they do require some effort from the user. Sabayon is a distro built on top of Gentoo, which incidentally, is what Gentoo is meant to be used for, so you get the benefits of the fast rolling release cycle without the hassle of maintaining the nuts and bolts of the system yourself. In addition to the live and install boot options, you have the choice of installing Sabayon as a Kodi media centre or as a Steam box. If you take the latter choice, it will download 200MB+ of *Steam* updates when you first boot it; something to consider if you have a metered Internet connection.

And more!

System tools

Essentials

Checkinstall Install tarballs with your package manager.

Coreutils The basic utilities that should exist on every operating system.

HardInfo A system benchmarking tool.

Kernel Source code for the latest stable kernel release, should you need it.

Memtest86+ Check for faulty memory.

Plop A simple manager for booting OSes, from CD, DVD and USB.

RawWrite Create boot floppy disks under MS-DOS in Windows.

Smart Boot Manager An OS-agnostic manager with an easy-to-use interface.

WvDial Connect with a dial-up modem.

Reading matter

Bookshelf

Advanced Bash-Scripting Guide Go further with shell scripting.

Bash Guide for Beginners Get to grips with *Bash* scripting.

Bourne Shell Scripting Guide Get started with shell scripting.

The Cathedral and the Bazaar Eric S Raymond's classic text explaining the advantages of open development.

The Debian Administrator's Handbook An essential guide for sysadmins.

Introduction to Linux A handy guide full of pointers for new Linux users.

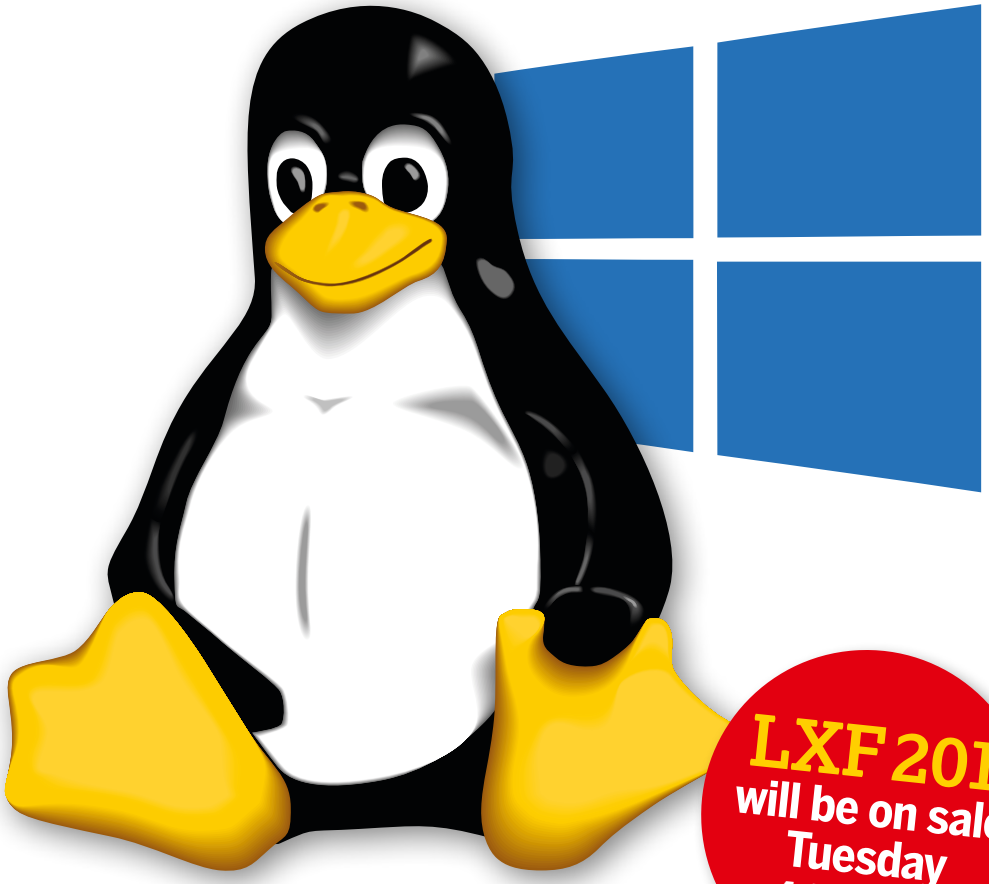
Linux Dictionary The A-Z of everything to do with Linux.

Linux Kernel in a Nutshell An introduction to the kernel written by master hacker Greg Kroah-Hartman.

The Linux System Administrator's Guide Take control of your system.

Tools Summary A complete overview of the GNU tools.

Dont Miss! Linux Format issue 1 PDF



LXF 201 will be on sale Tuesday 4 Aug 2015

Next month: Windows 10 vs Linux 4.1

We know who wins, but how good a fight can Microsoft's all-new, multi-device champion put up?

Server operating systems

Want a bullet-proof, easy-to-install, easy-to-use server system? We test the best Linux has to offer.

The Trust report

Who can you trust with your data online? The EFF has reported and it's not good reading...

Learning LaTex

Why all the fuss over an obsessive academic system? We guide you into using the best mark-up tool ever!

Contents of future issues subject to change – we're celebrating good times, come on!

LINUX FORMAT
Get into Linux today!

Future Publishing, Quay House,
The Ambury, Bath, BA1 1UA
Tel 01225 442244
Email linuxformat@futurenet.com

abc **19,000** January – December 2014
A member of the Audit Bureau of Circulations.

EDITORIAL
Editor Neil Mohr
neil.mohr@futurenet.com
Technical editor Jonni Bidwell
jonni.bidwell@futurenet.com
Operations editor Chris Thornett
chris.thornett@futurenet.com
Art editor Efrain Hernandez-Mendoza
efrain.hernandez-mendoza@futurenet.com
Editorial contributors Neil Bothwick, Dr. Chris Brown, Jolyon Brown, Matthew Hanson, Paul Hudson, Juan Martinez, Les Pounder, Alexander Tolstoy, Phil Savage, Mayank Sharma, Shashank Sharma
Illustrations Shane Collinge, iStockPhoto

ADVERTISING
For ad enquiries please contact:
Key Accounts - sales manager Richard Hemmings
richard.hemmings@futurenet.com

MARKETING
Marketing manager Richard Stephens
richard.stephens@futurenet.com

PRODUCTION AND DISTRIBUTION
Production controller Marie Quilter
Production manager Mark Constance
Distributed by Seymour Distribution Ltd, 2 East Poultry Avenue, London EC1A 9PT **Tel** 020 7429 4000
Overseas distribution by Seymour International

LICENSING
International director Regina Erak
regina.erak@futurenet.com **Tel** +44 (0)1225 442244
Fax +44 (0)1225 732275

CIRCULATION
Trade marketing manager Juliette Winyard
Tel 07551 150 984

SUBSCRIPTIONS & BACK ISSUES
UK reader order line & enquiries 0844 848 2852
Overseas reader order line & enquiries +44 (0)1604 251045
Online enquiries www.myfavouritemagazines.co.uk
Email *linuxformat@myfavouritemagazines.co.uk*

MANAGEMENT
Content & Marketing director Nial Ferguson
Head of Content & Marketing, Technology Nick Merritt
Group editor-in-chief Paul Newman
Group art director Steve Gotobed
Editor-in-chief, Computing Brands Graham Barlow

Printed in the UK by William Gibbons on behalf of Future.

DinoPC.com

Griffin 15.6" 840M

- Intel® Core™ i3 4100M processor (2x2.5Ghz)
- Windows 8.1 (64-bit)
- 8GB DDR3 1600Mhz RAM
- NVIDIA GeForce 840M 2GB
- 8x DVDRW
- 500GB S-ATAIII HDD
- 15.6" HD 1366x768 LED
- 2.0M Webcam + Card Reader
- Wifi + Bluetooth

£549 inc VAT

Elegant and slim laptop with superb all-round performance

The Winning Species

Trust us to build you a champion's PC for less.

Slayer 15.6" — £1,129

- Intel® Core™ i7 4710MQ processor (4x2.5Ghz)
- Windows 8.1 (64-bit)
- Corsair 8GB Vengeance 1600Mhz RAM
- 240GB SSD
- NVIDIA GeForce GTX 970M 6GB
- Wifi + Bluetooth
- 15.6" Matte LED (1920x1080)
- 2.0M Webcam + Card Reader

"Battery life is good...And there is plenty of horsepower under the hood"

Infernodon — £1,499

- Intel® Core™ i7 4790K processor (4x4.5Ghz)
- Corsair Hydro H100i Water Cooling
- Windows 8.1 (64-bit)
- Gigabyte Z97X-Gaming 3 Motherboard
- Corsair 16GB Vengeance Pro 2400Mhz RAM
- NVIDIA GeForce GTX 980 4GB
- Plextor 128GB M.2
- Seagate 1TB Hybrid SSHD
- NZXT H440 Chassis
- Corsair CS 650W Modular PSU

"This superb Devil's Canyon machine won't disappoint"

Microraptor — £1,049

- Intel® Core™ i5 4690K processor (4x4.3Ghz)
- Corsair Hydro H80i Water Cooling
- Windows 8.1 (64-bit)
- Gigabyte Z97N-Wifi Motherboard
- Corsair 16GB Vengeance 1600Mhz RAM
- 120GB SSD
- 1TB S-ATAIII 6.0Gb/s HDD
- NVIDIA GeForce GTX 970 4GB
- Coolermaster Elite 130 Chassis
- Corsair CX 600W PSU

"A gaming PC that delivers excellent performance for the price"

Discount Code:
PCF32G3W

Free 32GB drive +
3 working day priority

DinoPC.com

www.dinopc.com
0844 999 4123
Or visit our London office

DOMAINS | MAIL | HOSTING | eSHOPS | SERVERS

NEXT GENERATION 1&1 CLOUD SERVER

TRY FOR 1 MONTH – FREE!
Then from only £9.99 per month*

Powered by intel Cloud Technology

Top performance tested by CLOUD SPECTATOR

EASY TO USE – READY TO GO

The new 1&1 Cloud Server offers all the advantages of dedicated hardware performance combined with the flexibility of the cloud!

FLEXIBLE & AFFORDABLE

Customised configuration

- SSD, RAM and CPU can be adjusted independently, flexibly and precisely
- **NEW:** Pre-configured packages available

Windows

Transparent costs

- Billing by the minute
- Clearly structured cost overview enables efficient planning and management
- No minimum contract term

EASY & SECURE

1&1 Cloud Panel

- Innovative, user-friendly interface with smart administration

Security

- Built-in firewall to protect your server from online threats
- Backups and snapshots to prevent accidental data loss
- High-performance 1&1 Data Centres are among the safest in Europe

ALL-INCLUSIVE

Top performance

- Unlimited traffic
- Setup in 55 seconds
- Premium SSD with the highest possible performance
- Private networks, professional API, load balancers, firewalls and more, all easy to configure
- Ready-to-use applications including WordPress, Drupal™ and Magento®

1 TRIAL TRY FOR 30 DAYS

1 CLICK UPGRADE OR DOWNGRADE

1 CALL SPEAK TO AN EXPERT

0333 336 5509

1&1

1and1.co.uk

*1&1 Cloud Server powered by Intel® Xeon® Processor E5-2683 (35M Cache, 2.00 GHz). 1 month free trial, then from £9.99 per month (minimum configuration). No setup fee. Prices exclude 20% VAT. Visit www.1and1.co.uk for full offer details, terms and conditions. Windows is a registered trademark of Microsoft® Corporation. Linux is a registered trademark of Linus Torvalds. Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

9016

9000