



**HARDWARE:** BUILD A PROXIMITY DETECTOR

# LINUX VOICE

**116 PAGES  
OF LINUX  
LEARNING**

February 2015

FREE SOFTWARE | FREE SPEECH

WEB DEVELOPMENT  
**FIREFOX**

Write your own plugins  
for the #1 web browser

SOCIAL ENGINEERING  
**HACKING!**

How to trick people into  
giving up their passwords

SUPERCOMPUTING  
**GPU CODING**

Unleash the power of  
your graphics card

## THE FIGHT FOR FREEDOM

How Free Software began, where it's  
going and why it's still important.

**GEEKS LIKE TO THINK THAT THEY CAN IGNORE  
POLITICS; YOU CAN LEAVE POLITICS ALONE,  
BUT POLITICS WON'T LEAVE YOU ALONE.**

**34+ PAGES OF TUTORIALS**

- PREY Track your stolen laptop across the internet
- THE LV PUB QUIZ Put your expanded neurons to the test
- WEB BROWSERS Find the swishest way to interact with the web

### OUTLAWS' LAST STAND

#### PODCASTS

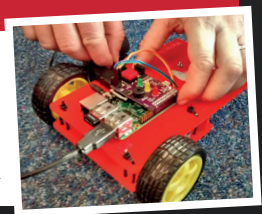
Weep, for the Linux  
Outlaws are no more



### EDUCATIONERISING

#### PICADEMY

Teachers, welcome  
to the world of geek



February 2015 £5.99 Printed in the UK

9 772054 377001

ISSN 2054-3778

029

**ANDREWS & ARNOLD LTD**

will make you the

# LINE KING



**BE THE MASTER OF  
YOUR OWN TELEPHONE**

SIP2SIM® from Andrews & Arnold allows you to treat your mobile handset as a SIP endpoint, freeing you from your desk and without the need of a smartphone app. Insert the SIM into your phone, point it to your Asterisk, FreeSwitch or other SIP server (or a commercial SIP service) and experience the reliability and call quality you're used to on a mobile, but with the flexibility of a SIP handset.

No minimum term. SIM £5+VAT and £2+VAT per month. Calls from 2p+VAT per minute.

Call 033 33 400 220, email [sales@aa.net.uk](mailto:sales@aa.net.uk) or visit [www.SIP2SIM.uk](http://www.SIP2SIM.uk) to find out more

HAKUNA MATATA

# Everything is awesome

The **February** issue

## LINUX VOICE

Linux Voice is different. Linux Voice is special. Here's why...

**1** At the end of each financial year we'll give 50% of our profits to a selection of organisations that support free software, decided by a vote among our readers (that's you).

**2** No later than nine months after first publication, we will relicense all of our content under the Creative Commons CC-BY-SA licence, so that old content can still be useful, and can live on even after the magazine has come off the shelves.

**3** We're a small company, so we don't have a board of directors or a bunch of shareholders in the City of London to keep happy. The only people that matter to us are the readers.

### THE LINUX VOICE TEAM

**Editor** Graham Morrison  
graham@linuxvoice.com

**Deputy editor** Andrew Gregory  
andrew@linuxvoice.com

**Technical editor** Ben Everard  
ben@linuxvoice.com

**Editor at large** Mike Saunders  
mike@linuxvoice.com

**Games editor** Liam Dawe  
liam@linuxvoice.com

**Creative director** Stacey Black  
stacey@linuxvoice.com

**Malign puppetmaster** Nick Veitch  
nick@linuxvoice.com

**Editorial contributors:**  
Chris Brown, Mark Crutch, Liam Dawe, Josette Garcia, Juliet Kemp, John Lane, Vincent Mealing, Sharon Mitchell, Simon Phipps, Les Pounder, Mayank Sharma, Valentine Sinitsyn



### GRAHAM MORRISON

A free software advocate and writer since the late 1990s, Graham is a lapsed KDE contributor and author of the Meeq MIDI step sequencer.

**L**inux, open source and Free Software have all become hugely successful. By and large, the coercive era of proprietary software is over. Many companies now look at open source as a distinct advantage, because they easily understand the freedoms it provides: hire a developer and build atop the shoulders of giants. It also helps that the industry is brimming with incredible talent (like you!), all of whom have cut their teeth on open source and made it part of their DNA.

But it's important not to become complacent. In particular, the technology industry of 2015 has very little in common with the 1970s, when Richard Stallman was at MIT and formulating his ideas. It reminds me of the famous George Santayana quote, "Those who cannot remember the past are condemned to repeat it." However, this is the best kind of challenge. Linux is in a dominant position because, I believe, it represents a simple truth: sharing and building things is a lot of fun. As long as we keep having fun doing what we love, its dominance will only grow.

**Graham Morrison**  
Editor, Linux Voice

**SUBSCRIBE  
ON PAGE 62**



## What's hot in LV#011



### MAYANK SHARMA

Juliet continues to open up the fascinating world of early computing. This month she's looking at mythical Cray. **p96**



### BEN EVERARD

Test out your Linux and open source knowledge with our epic 80-question pub quiz, and send us your scores! **p28**



### MIKE SAUNDERS

We cast some much-needed light on how the dark side of the internet is harnessed with the Social-Engineer Toolkit **p84**



# CONTENTS

February LV011

Happy Christmas/Burns Night/St David's Day!

**SUBSCRIBE  
ON PAGE 62**



20

## The fight for freedom

Free Software isn't just convenient and cheap – here's why it matters to everyone.

42

## Brian Behlendorf

Mozilla, the EFF, Barack Obama's election campaign and the Burning Man festival all owe a debt to this man. Say hello!



28 **PUB QUIZ**

There are no material prizes – only the best prize of all, which is geeky pride. Your starter for 10...



32 **LINUX OUTLAWS**

For years the airwaves were ruled by outlaws. Here's a swearword-free recap of their glory days.



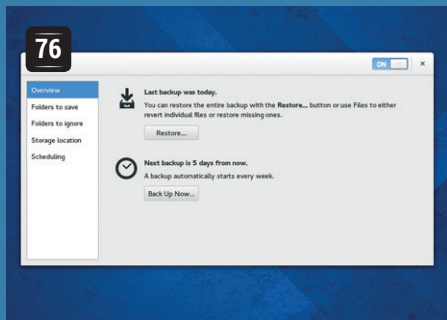
36 **PICADEMY**

Inside the Raspberry Pi Foundation's project to teach teachers how to teach computing.

## REGULARS

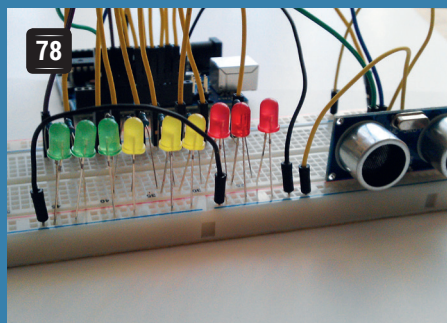
- 06 News**  
Debian has forked. FreeBSD is \$1m richer, and the Jolla tablet is doing rather well.
- 08 Distrohopper**  
ReactOS, RebeccaBlackOS and AV Linux, plus a look back at Linux Mint 1.
- 10 Gaming**  
War, zombies and space, all fictionalised, gamerised, and brought to your Linux box.
- 12 Speak your brains**  
Some kind words for our creative commons efforts, plus a note of disquiet.
- 16 LV on tour**  
Brought to you by SWAMPFest (Swansea) and the Internet Of Things (everywhere).
- 18 CloudStack**  
Brave Sir Graham bravely rides to Budapest to meet vastly important cloud people.
- 40 FAQ... ASM.JS**  
It's JavaScript Jim, but not as we know it – it's a faster embedded subset, aye!
- 56 Group test**  
We spend so much time on the web that we should really find a decent browser.
- 62 Subscribe!**  
Will Tiny Tim have goose for Christmas? You decide! Also, get a brilliant magazine!
- 64 Core technologies**  
Dr Brown interprets the signals that processes send to each other.
- 68 FOSSpicks**  
Freer than the bird that Lynyrd Skynyrd sang about in that song with the guitar solo.
- 110 Masterclass**  
Whether you're a guru or a newbie, here's how to encrypt your files with Linux.
- 114 My Linux desktop**  
Linux Voice's editor Graham Morrison invites us into his synth paradise.

# TUTORIALS



## 76 Déjà Dup: Backup for everyone

Protect yourself from data loss apocalypse the easy way.



## 78 Arduino: Build a proximity detector

Use nearby objects to activate many, many blinkenlights.



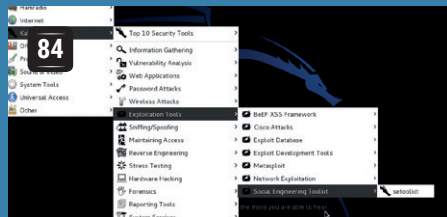
## 82 Prey: Track down stolen hardware

Keep tabs of your devices in the event of their purloinment.

**100 Firefox: Code addons**  
Add functions to the #1 browser.

**104 Code Ninja: NoSQL databases**  
Enhance your big data project.

**106 Program with your GPU**  
Unleash your graphics card.



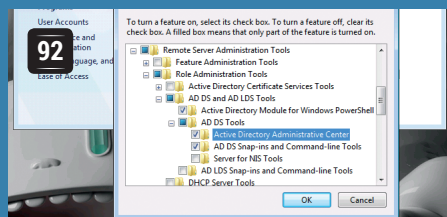
## 84 Social-Engineer Toolkit: Steal data

Understand how the criminals are trying to trick you.



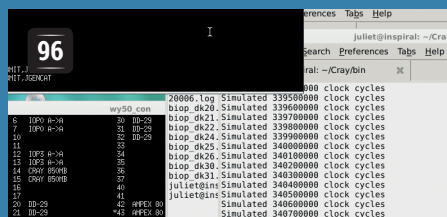
## 88 Linux 101: Emulate Windows with Wine

Bring your old applications with you when you move to Linux.



## 92 Samba 4: Use Windows shares from Linux

Admins, rejoice: Samba now works with Active Directory.



## 96 Olde Code: Seymour Cray and supercomputers

Emulate cutting-edge hardware from the time of glam rock.

# REVIEWS



**48 Entroware Proteus**  
It's lovely when companies support Linux. It's even better when their wares are this good.



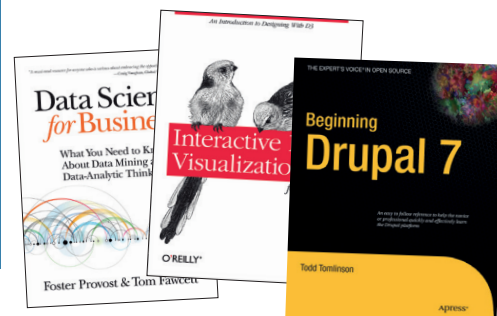
**50 The Tor Browser**  
Anonymise your web traffic the easy way with this US State Department-endorsed browser.

**51 Digikam 4.5.0**  
Manage large photo collections without Facebook, and add filters without Instagram.

**52 Mastering Vim**  
Jedi master Damian Conway will turn you into a master Vim user. He's scary, in a good way.

**53 Firefox Developer Edition**  
Developer: are you frustrated with the ever-moving target that is Firefox? Yes? Try this!

**54 Books** Now with infinite battery life, high resolution and no screen glare – books!



# NEWS ANALYSIS

The Linux Voice view on what's going on in the world of Free Software.

Opinion

## Free or open?

Which is more important? The ethics of Free Software, or the pragmatism of open source?



**Simon Phipps** is president of the Open Source Initiative and a board member of the Open Rights Group and of Open Source for America.

concrete set of criteria so that businesses could know if they had permission in advance to collaborate, plus a term that could be used with business people that did not instantly distract people from the real point.

For those with English as a first language, “free” invokes a narrative frame relating to price, and the concept of liberty can only

stops delivering the pragmatic values. That’s why “open core” software fails, for example. Its focus on making the source available omits granting permission in advance to collaborate on the whole software solution.

It fails to guarantee software freedom, and without it the pragmatic benefits of open source don’t materialise either. There are plenty of other failure modes for Free and

There’s a decades-old discussion hidden behind the terms “free software” and “open source” which surprisingly still divides people even today. What’s at the root of the division? Should we still be divided?

The Free Software Foundation was created by Richard Stallman to promote the ethical imperative of Free Software. In the late 1990s a group of experienced people concluded that the term “free software” was a problem in communicating the ideals of software freedom, because the word “free” was too often associated with getting something for nothing.

To better promote the idea that the benefits of software freedom relate to flexibility and community, they decided to coin a new term – “open source” – and start a new organisation, OSI (the Open Source Initiative), to act as the steward of the OSD (Open Source Definition) and rule on which copyright licences truly delivered software freedom.

The people who coined this term were almost all advocates of software freedom as an ethical concept as well as of its pragmatic benefits. They just wanted a

**“It’s important for every member of the community to realise that we’re part of a single movement.”**

be introduced by way of explanation. As linguistic theorist George Lakoff explains, once the narrative frame is set it’s nearly impossible to change it, so it’s better to start a conversation with a term that invokes the correct frame – I prefer to speak of “flexibility” – and then introduce other terms later – I speak of “software freedom”.

### We need flexibility

At first, Richard Stallman accepted the new term, but sadly a set of personality conflicts led to him rejecting it strongly, eventually even condemning use of the term “open source” as ethically bankrupt. But the answer to my original question is that both terms matter, and neither is effective without the other. A focus on ethics without pragmatics alienates many people by sounding “preachy”, while a focus on pragmatics without ethics drifts astray over time and in the process of becoming ethically bankrupt as Stallman asserts also

Open Source Software – the argumentative collective that insists on ideology; the company-dominated project that denies liberty to collaborators; the single-copyright-holder who changes the licence – and so on.

The dual imperative of adhering to the concept of software freedom as a reference model while articulating and securing pragmatic benefits of a collaborative development model is the only successful approach. In every case of failure, part of that dual imperative has been ignored.

### Open can also be free

I may be president of the OSI, but I am a strong and persistent advocate of software freedom. I don’t believe there’s any conflict in that, and neither do most of my good friends at the FSF. I believe it’s important for every member of the community, whether they use the term “Free Software” or the term “Open Source”, to realise we are all part of a single movement, the software freedom movement.

Neither ethics without pragmatics nor pragmatics without ethics actually deliver the software we need. Software freedom does.

**“A focus on ethics without pragmatics alienates many people by sounding preachy.”**

# CATCHUP

## Summarised: the biggest news stories from the last month

### 1 Groupon tries to nab Gnome trademark, fails spectacularly

Online voucher seller Groupon decided to launch a new point-of-sale OS called Gnome, and applied for trademarks, thereby causing serious hassle for the desktop environment. So the Gnome Foundation asked the community for financial help, raised \$102,000 in legal fees, and Groupon backed down. The moral of the story? Don't let your marketing department mess with Free Software communities – they can mobilise the troops damn quickly.

### 2 Mozilla says adieu to Google, Firefox to use Yahoo Search in future

Google and Mozilla have had a long running partnership, with much of *Firefox's* development funded by the search giant. But from December, *Firefox* will use Yahoo as its default search engine, thanks to a new deal. It's all about "promoting choice and innovation" according to a bland buzzword-saturated statement from Mozilla; more realistically, the company simply needs money to fund its browser, mobile OS and other projects.

### 3 Firefox OS comes to the Raspberry Pi

In other Mozilla news, FirefoxOS is being ported to everyone's favourite single-board computer. The developers would like to see the OS reach "parity with Raspbian" in 2015. <http://tinyurl.com/ojde3yg>



Firefox OS

### 4 Debian is forked: say hello to "Devuan"

This happened just a few hours before we went to press, so it's too soon to tell whether it's a serious effort or an elaborate troll, but a new website has been set up at <http://devuan.org> which aims to create a spin-off of Debian without *Systemd*. Devuan aims to "protect the freedom of its community of developers and users", and also "preserve Init freedom". This is a mammoth undertaking, but if it's real, it's good to see some proper work and not just flame wars on forums.

### 5 FreeBSD receives \$1m donation from WhatsApp founder Jan Koum

Happy days for the FreeBSD Foundation: the CEO and co-founder of the WhatsApp messaging service has donated \$1m to the project. "FreeBSD helped to lift me out of poverty", explained Koum, describing how access to a no-cost and robust Unix flavour helped him to get a job at Yahoo and build a career. "We'll all benefit if FreeBSD can continue to give people the same opportunity it gave me, and help more startups", he added.

### 6 Microsoft open sources .NET, go cross-platform

We've come a long way from Steve Ballmer's "Linux is a cancer" slurs from the last decade. Microsoft has announced that it's open sourcing the full server-side .NET stack, and "expanding it to run on the Linux and Mac OS platforms". This could help developers who've had headaches using Mono in the past, and the source code will be uploaded to <https://github.com/Microsoft/dotnet>. We're still cautious about the company, but it's a welcome move nonetheless.

### 7 Huge success for Jolla's crowdfunded tablet

Smartphone maker Jolla, founded in 2011 by ex-Nokia employees, has decided to take on the tablet market. The company went to Indiegogo to ask for \$380,000 – but at the time of writing, with still 12 days left to go in the campaign, almost \$1.3m had been raised. The tablet will run the Linux kernel-based Sailfish OS, and be equipped with a 1.8GHz quad-core Intel chip, 2GB of RAM and 32GB storage. The expected retail price is \$249, and it goes on-sale in May 2015.



### 8 Debian Systemd dev quits after flamewar burnout

This is rather sad. Tollef Food Heen, a Debian developer who maintained *Systemd* in the distro, has stepped back from his role after receiving a huge amount of flak for his work. "The load of the continued attacks is just becoming too much", he said on a Debian mailing list, and later remarked that conspiracy theories (that Red Hat was forcing every distro to use *Systemd*) were also making him glum. Hopefully someone else will step up and not get flamed at every turn.

# DISTROHOPPER

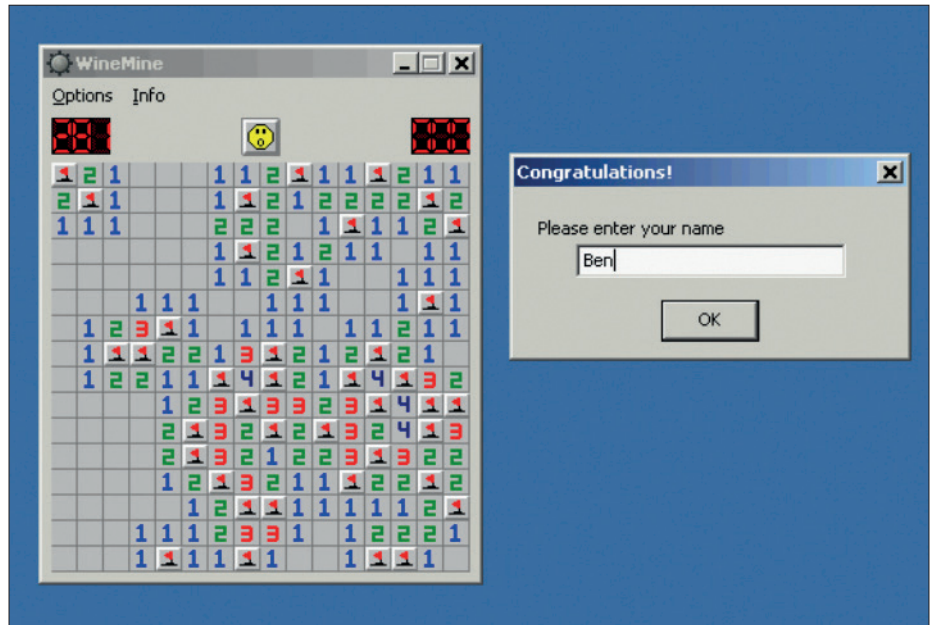
Our pick of the latest releases will whet your appetite for new Linux distributions.

## ReactOS

Like Windows, but open.

OK, this isn't Linux – it's not even based on Unix – but it is a free operating system that you can try out. ReactOS is a clone of the Windows NT kernel used in Windows XP, and some of the API. This means that in theory, you should be able to use ReactOS just like a Windows system: install the same drivers, run the same software, etc. However, in practice, the implementation is not complete enough to allow you to do this. You can run the simple tools that come with the OS, but not much else. *Wine* offers a much better chance of being able to run Windows software without a full Windows install. Even though *Wine* and ReactOS share code, *Wine* has a much better success rate.

This is a shame, because if the team had been able to create a fully working system by the time Microsoft stopped support for Windows XP, they may have found many new users. As it is, the project might have missed its chance to become mainstream.



Don't tell Linus we said this, but some games just don't look right when running on Linux.

Just because a project isn't mainstream, that doesn't mean it's not interesting. Booting up ReactOS feels like taking a trip back in time – its visual style probably has more in common with Windows 95 and 98 than XP. ReactOS does, of course, have

*Minesweeper*, the game that killed millions of man-hours worth of office-worker time in the last years of the previous millennium. Perhaps it's not the best reason to get a new OS, but for us, this dose of nostalgia made it worth booting up a virtual machine.

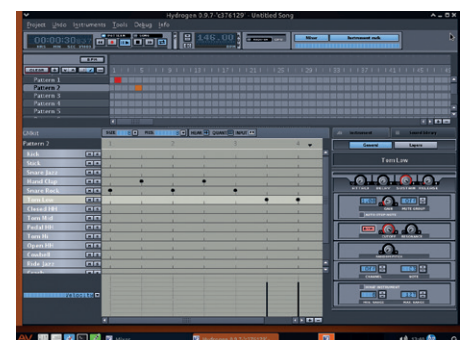
## AV Linux

The distro of choice for media creators.

AV stands for Audio Video, and this is a distro built for creating music and videos. It's jam-packed with software to help you do this – both free and commercial, but this isn't simply a distro created by installing particular packages on a base system. AV Linux not only curates the software, but also the configuration of the underlying Debian build. Much of the software – including the kernel itself – is built specially for AV Linux, and this is what makes the distro special. The result is a system that's less flexible than raw Debian, but far more suited to content creation.

This distro is probably better known for audio production than video editing. However, it is probably the best distribution of Linux for either task. There are also some useful tools for image editing, but it doesn't stand out as significantly better than other distros in this area.

If you're fed up of struggling to get a decent audio setup on Linux, AV Linux is for you. It's also great if you want to discover the best audio or video software without having the hassle of configuring the sound setup to make it work, and it uses the intuitive, smart Xfce desktop.



Fear not the configuration of *PulseAudio*, for it has been done already.

AV Linux also stands out because of its excellent documentation. You'll find several manuals as PDF files on the desktop to guide you through most things, and help you understand how AV Linux works.



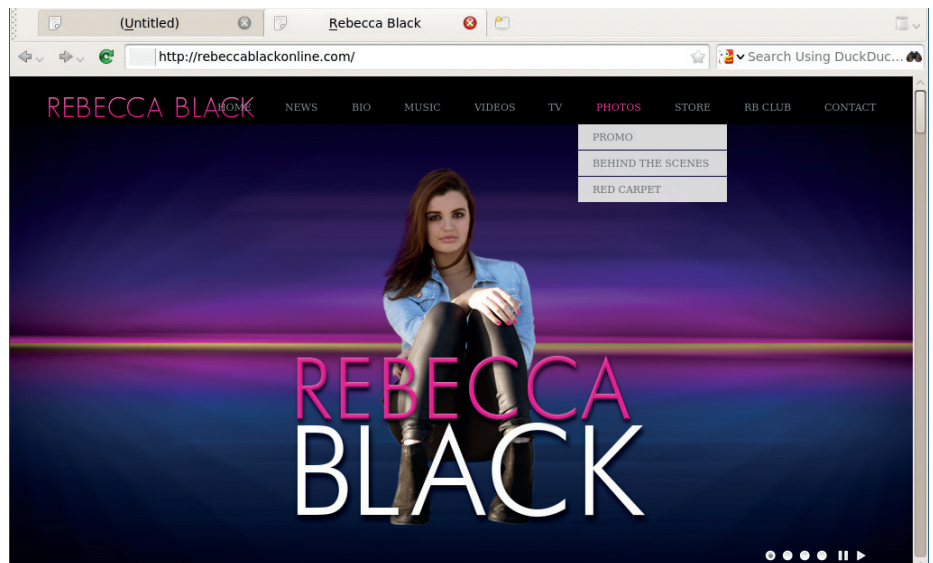
# RebeccaBlackOS

Friday, Friday ... we'll stop there.

**Y**es, this distro really is about the teen popstar who was briefly famous for the song 'Friday'. No, that doesn't mean you can discount it as an uninteresting distro. Despite its (let's be polite and say) unusual inspiration, RebeccaBlackOS (RBOS) has been pioneering Wayland on desktop Linux.

Wayland – the next-generation graphics server that should replace X Windows on almost every Linux distribution other than Ubuntu – has been around for quite some time, and is already in use on Jolla phones and some smart TVs. However, there are currently very few ways of trying it out on desktop Linux, and the RBOS live CD is by far the easiest (the second easiest is through Maynard on a Raspberry Pi: <https://github.com/raspberrypi/maynard/wiki>).

If you're anything like us, you'll have heard so much about how Wayland is the future of Linux that you'll be itching to try it and find out what the fuss is about. RBOS is the solution to this problem. It's got quite a range of software using the *Qt*, *GTK* and *EDL* (Enlightenment) widget toolkits, so you can



If you can look past the artwork, RebeccaBlackOS is the best way of trying out Wayland.

get a good idea about how each of them are working. Perhaps the most impressive thing about RebeccaBlackOS is just how normal it is. OK, the graphics are a little odd, but that's a small issue compared to getting the

display server running. Although it lacks much polish, it does seem to handle basic computing tasks without any problems, and this bodes well for the future of Linux on the desktop.

## Linux Mint 1 (Ada) Linux Mint has changed the nature of desktop Linux, but how did it start?

We're starting a new historical section to Distrohopper, where we look back at major releases of yesteryear. The first one to get dug up and dusted off is Linux Mint 1, aka Ada. Technically, this version never made it out of Beta (2.0 was the first stable version of Mint). However, it was with this unstable version that the journey began.

KDE version 3.5 greeted us after we booted Linux Mint 1. This came as a bit of a surprise, because Mint is most famous for its Gnome versions (and later Mate and Cinnamon). However, Mint made the switch to *GTK* with version 2.

For the most part, it's a fairly standard KDE desktop for the time. *KOffice* took up office duties, *Konqueror* served as a file manager, and all the other usual tools that begin with K are in their usual place. Outside of the KDE suite, *Firefox 1.5* serves as web browser, and *Gimp 2.2* takes up image editing duties.

On the whole, it doesn't feel too dated, despite being eight years old. The biggest things that stand out graphically are the lack of anti-aliased fonts and the inability of the ancient version of *Firefox* to render any modern web page. The use of floppy disk images on the install icon is another clue that this isn't from the current decade.

Linux Mint became known as one of the best-looking distros, so it's surprising to see such a graphical faux pas as an RSS reader along the full width of the screen under the taskbar. It's quite impressive though that the two RSS feeds picked back in 2006 still work (OSNews and Distrowatch).

Clem (the founder of Linux Mint) talks about the early releases in a blog post at: <http://segfault.linuxmint.com/2014/01/ada-barbara-bea-bianca-and-cassandra>.



Why oh why is that RSS feed there?

# GAMING ON LINUX

The tastiest brain candy to relax those tired neurons



## ALEA JACTA EST



Liam Dawe is our Games Editor and the founder of [gamingonlinux.com](http://gamingonlinux.com), the home of Tux gaming on the web.

Linux Gaming seems to be stronger than ever, but a question that has been plaguing us recently is; will we ever hit over 2% market share for gamers?

We have all genres of games now gracing our platform, but the uptake doesn't seem to be going anywhere just yet.

We have Valve singing our praises to developers, and GOG supporting us with some classic games to help build up our back catalogue as well, and many other smaller stores are also now supporting Linux.

Until the day your average Joe (sorry Joe, but you're not very tech smart!) can walk into their local PC store and have Linux thrust in their face the same way Windows is, it's possible our day will never come.

We've had a few small breakthroughs recently, with games like *Football Manager* having a Linux icon directly on the games box, and even on their big posters at game stores around the world. It may be small, but getting the Linux icon out there in major stores may cause more people to find out what it is. Never underestimate the small things.

We also need people to make sure they only buy Linux games once the Linux version is available; all too often we see people getting burned by developers who promise a Linux version, and then it never comes. This is a repeating problem, and one we should avoid creating for ourselves.

What are your thoughts? Do you agree with us, or do you think we are completely off the mark here?

## War Thunder

Linux finally has another good quality MMO.

**W**ar Thunder is a completely free-to-play massively multiplayer online (MMO) game, which is one of its main draws. However, free-to-play games often turn into what people call "pay 2 win", meaning people who put real money into it for extras gain the upper hand. Thankfully that's not quite the case here.

War Thunder is a mixture of intense aeroplane and land-based tank battles. You don't have to spend a penny to get any enjoyment out of it based on our testing. If you enjoy flying around smashing other human players out of the sky, then this is the game you have been waiting for.

There's no launcher just yet, so if you download it directly from Gaijin Entertainment's website, it's advisable to run it in your terminal to watch the progress. The download is actually an updater you need to run, which downloads the rest of the game. The game is also on Steam, so getting it there does make it easier.

Even with the excellent graphics, the performance of the game is stellar too, so overall the Linux port does seem very well done. The control system is one of the best things about it



as the aeroplanes are very easy to control and it makes flying a real pleasure.

There are masses of planes and tanks to choose from and, as you play more games, you unlock different nations each with their own set of units for you to try out.

The game can be quite overwhelming at the start, due to the underwhelming tutorial, which only really teaches you the basics of flight and combat, and doesn't explain much about the research and unlocking of new vehicles.

Developer Gaijin Entertainment Release Date  
6 November 2014 Website <http://warthunder.com>



Don't blame us if you get addicted – War Thunder is quite the time sink.

**"If you enjoy flying around smashing other players out of the sky, this is the game you have been waiting for."**

# Icewind Dale: Enhanced Edition

A crime for RPG fans to pass up.

**I**cewind Dale is probably a name a lot of our older readers remember as it originally came out in 2000. Thankfully some developers do go back to older games and bring them to newer game engines to support not just newer computers, but better operating systems like Linux too.

It's a role playing game from a top-down perspective, so it doesn't change any of the main features of the original release, but merely cleans it up for modern gaming. It does however include new content, like new spells, items, armour and weapons.

One of the best features of the new edition is the ability to play co-operative with your friends across Linux, Mac & Windows online. One thing that hasn't changed is the graphics; they're still low resolution compared to today's games, and it looks a little dated. Luckily though, part of the upgrade included making it fit nicely on high screen resolutions.

Fans of games like *Baldur's Gate* and other 2D RPG's will fall in love with it.

Developer Beamdog Release Date 30 October  
2014 Website <http://icewinddale.com>



A party-based RPG,  
no lone wolves here.

# Dead Island

Get ready for the zombie apocalypse.

**D**o you love zombies? Good, as *Dead Island* has them in spades. This highly popular zombie smashing RPG isn't your conventional zombie game either, as the main focus of it is the melee combat.

It's aimed at a mature audience, so we wouldn't go putting it on while you have any younger members of your family running around as it's pretty much guaranteed to give them nightmares.

Unlike *Left 4 Dead* from Valve, *Dead Island* is an "open world" zombie game, so you don't have defined paths to follow; you can go wherever you see fit to find gear, smash zombies and complete quests.

The game gets quite intense at times, especially if you manage to get surrounded, and, as ammunition is a tight resource, a lot of the time you really do need to use a melee weapon, which is not



easy. If you're getting stuck, then why not team up with another friendly penguin gamer? *Dead Island* also includes co-op play for up to four people at a time.

The Linux port isn't without issues though, if you find the graphics look like they are in a very low colour mode, try turning down the gamma.

Developer Techland Release Date 24 October  
2014 Website <http://store.steampowered.com/app/91310>

## ALSO RELEASED...



### Interstellar Marines

*Interstellar Marines*, the great looking first person shooter, has been updated a few times recently and it now includes some excellent co-op action against evil robots.

One of the new game modes sees you frantically trying to turn the power back on in different sections of the map, but those pesky robots are waiting for you in the dark, and it gets a bit jumpy.

<http://store.steampowered.com/app/236370>



### Double Action

Have you ever wanted to shoot someone while smashing through a skyscraper window? *Double Action* has you covered! It's an over-the-top, points-based shooter that can be played in first or third person, and it's hilarious.

It's completely free, with zero features to pay for, now that's what we like to see!  
<http://store.steampowered.com/app/317360>



### Transistor

If you're in the market for a beautiful and fun action RPG in a sci-fi world then you should stop here. *Transistor* is the next game from the creators of *Bastion*, and with "overwhelmingly positive" reviews on Steam, you can't pass it up. Encouragingly, the graphics are just as stylish as *Bastion*. Enjoy!

<http://store.steampowered.com/app/237930>

# LINUX VOICE YOUR LETTERS



Got something to say? An idea for a new magazine feature? Or a great discovery? Email us: [letters@linuxvoice.com](mailto:letters@linuxvoice.com)

## LINUX VOICE STAR LETTER

### THAT'S NUMBERWANG

*Mathematica* [reviewed in LV007] is nice, but there are several FOSS alternatives. *Octave* was recently reviewed in another publication, which got me interested as I've succeeded in getting my mentee enrolled in computer engineering at university and now find myself helping with homework, prompting me to review my college math books and note that we've moved on a bit from the days of hand-cranked desk calculators, slide rules, and books of tables.

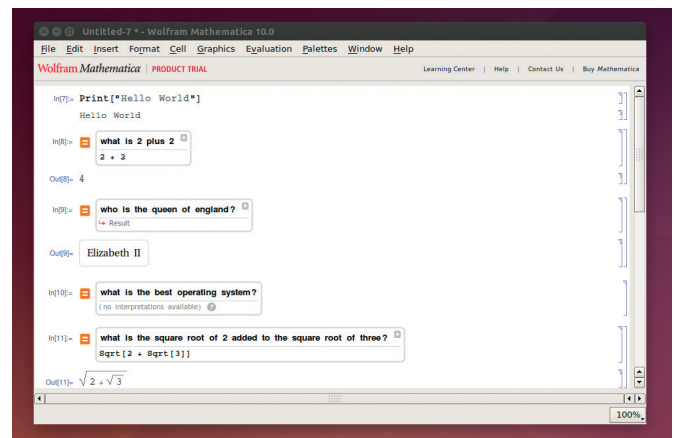
*Octave* is powerful, but not the best choice for analytical or symbolic math. I tried *Maxima*, which is in the Mint repository, but changed to *Sage* because I found a

beginner's guide to using it. *Sage* is very good and the plotting worked better than *GNU Plot* on my machine, but I know I am going to need geometry, which led to another search and the discovery of *GeoGebra* [[www.geogebra.org](http://www.geogebra.org)].

*GeoGebra* is relatively new and absolutely wonderful. Perhaps you could do a comparison of the leading FOSS computer-aided math applications in a future edition of Linux Voice.

**Andrew Shead.**

**Ben says:** As I said in the review, the idea of coding in a proprietary environment deeply troubles me. The software you mention are all good choices, but they all



Whatever the relative merits of *Mathematica*, a basic version of it now comes free of charge as part of Raspbian.

have their own shortcomings. Personally, my language of choice for mathematical computation is Python (with NumPy, SciPy and other

modules), although this also has its limitations. This does sound like an area ripe for comparison. We'll look into it for a future issue of Linux Voice.

### RELENTLESS CHEERINESS

Still loving the magazine, thanks for the *i3* tutorial – it was just what I need at the moment. I have two suggestions for the magazine.

Firstly, although the gaming section is welcome and a good read I wonder if it's time for it to become a bit more critical? As the gaming on Linux scene matures we are being asked to pay upwards of £30 for some of the games and it would be nice to know more about whether they will repay the investment.

Secondly, a plea for a rain-detection Raspberry Pi tutorial. My

wife is obsessed with line-drying clothes in the depths of winter so I would get major brownie points if I could build something that let her know if it rained. Maybe you could even cover serving requests on a web server so others nearby could access the same information? Take the British obsession with the weather and Linux-ify it.

Keep up the great work.

**Chris Beeley, Nottingham**

**Andrew says:** The gaming section is really there to provide a snapshot of what's going on, rather than an



in-depth review, and as such we don't feel there's any point including stuff that's rubbish. As for a Pi weather station, I want one too. Clothes dried outside always feel much fresher.

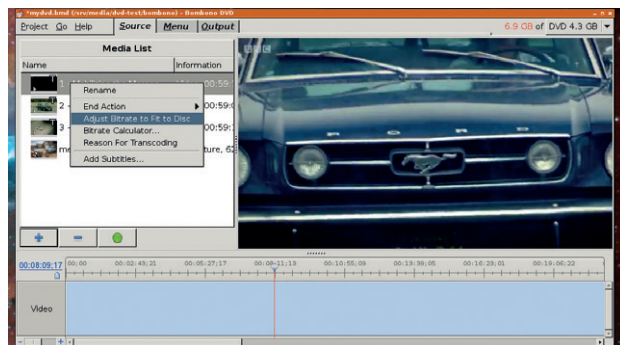
If ever we come across a big name title that's robbing people blind, we'll let you know.

## TOO EASY? TOO HARD?

Whatever the subject, be it computers, astronomy, mathematics etc, there is a tendency for those who know the subject well to forget how easy it is to confuse or 'scare off' those of us who are new to the subject, and in this case, your excellent magazine. If someone asked me, as a novice, how to copy a DVD, I would tell them that I used Kubuntu 12.04, which contains the K3b burning software as standard, and installed K9Copy from the software store. This allowed a 7.9GB protected DVD to be shrunk down to 4.2GB and burned onto a blank DVD. (K9Copy & K3b only work together

on KDE desktops; they do not appear to work on Ubuntu's Unity desktop). Whilst I have no doubt that your instructions are far more comprehensive and cover more situations, not all of us are yet ready for your 'Masterclass' approach. Please do not forget that we 'refugees from Windows' need to start at the easy end of the Linux subject.

**Andrew says:** Hmm. The masterclass section, by definition, is meant to provide an exhaustive examination of a particular application, and so it's never going to be aimed at anyone who just wants to get from metaphorical point A to



point B. That said, the balance has tipped away from new users in recent issues, and we'll have an opportunity to address that soon. Out of interest, is there anything that you'd like a beginners' guide to? Let us know and we'll do our best to provide one.

There are many ways to skin a cat in Linux; sometimes the most reliable way isn't the easiest.

## PANIC AND FREAK OUT!

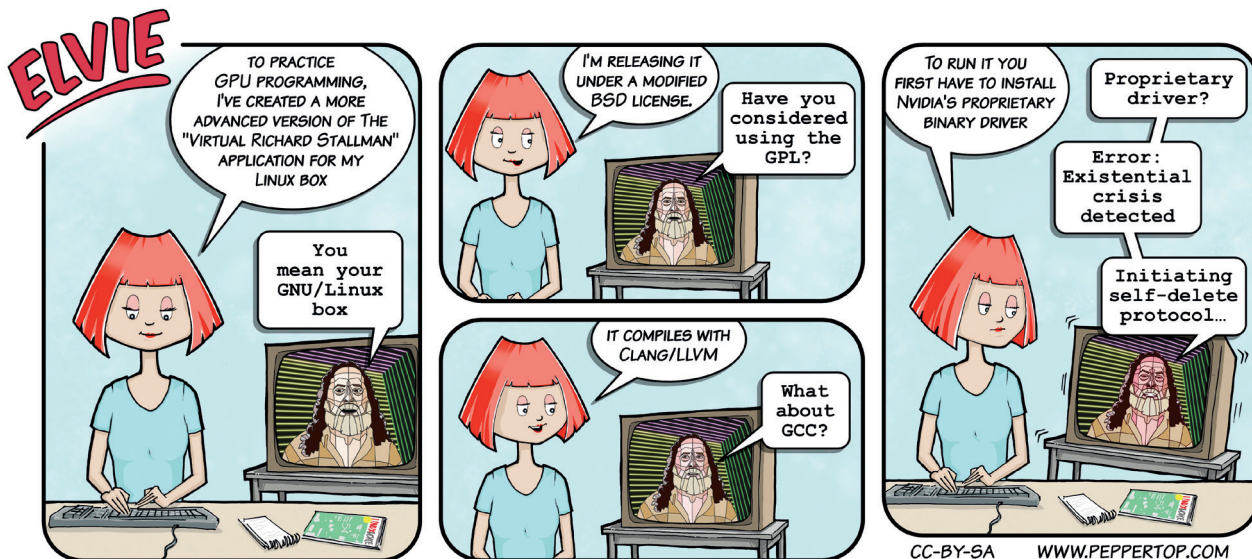
So, Symantec has discovered a new computer virus thing called Regin, and it's going to kill us all in our beds, or something. I looked on the BBC, Sydney Morning Herald and Symantec's own website and apparently this latest doomsday virus affects "computers". I have a computer! Should I be worried? Or is it actually more likely that this, like all them seem to be, is just a Windows virus and the media either doesn't know or is too craven to make the distinction that Windows computers ≠ all

computers?  
**David Watson, Melbourne**

**Mike says:** You're absolutely right David, and I tried to get in touch with the BBC's Rory Cellan-Jones to let him know about this oft-repeated error. At least, I hope it's an error: the cynic might suggest that the BBC et al are afraid of angering Microsoft by suggesting that it bears any culpability for the holes in its security, and instead is hedging its bets by using the vague (but still accurate) term "computer virus".



Far be it from us to illustrate the concept of a computer windows virus with a lazy stock photo search for the term 'computer virus'.



## CALM DOWN DEAR!

As a long time listener to your podcast (and as a listener to TuxRadar before then) I've laughed along with you on many occasions about the dafteries that free software throws up. But I'm a bit concerned over the Systemd argument that's raging on and that you've touched on recently in the podcast. I must admit that I don't understand the technical arguments (though I trust the Debian people, so if they're happy with it, I'm happy with it). But the level of personal invective that's been levelled at Lennart Poettering [the developer of Systemd] is on another level entirely, and is making me rethink the reasons that I got involved in the first place. I thought free software was supposed to be about inclusivity; instead it seems that there's a nasty element taking over. There's

a bullying culture about it now that I don't like.

**Andrew says:** Two things: yes, you're right. Technical criticism is one thing; personal abuse is quite another, and (apart from anything else) must be pretty alienating for anyone looking at free software from the outside in. But the other thing is that the pond life who are harassing the Systemd developers are representative only of themselves.

There are many, many people who contribute to free software, so the law of averages suggests that some of those people will be idiots. But they are only a minority. Actually, they're more than that; they're a vocal minority, and like all vocal minorities, they shout louder than the rest. Everyone has a different opinion, but as far as I can see the best thing to do is ignore them.



This man does not deserve the hate he gets online. Trolls, stop it! Image: Ramkrsna CC BY-SA 2.0.

## FREEEEEEEDOM!

I just wanted to get in touch to say a big well done on releasing issue 1 as Creative Commons CC-BY-SA. I subscribed when you launched before Christmas 2013 partly because I trusted your reputation from your old magazine, but what really grabbed me was your commitment to release your content free to the community. I've been looking forward to the day when you relicensed issue 1, and now you have I'm so happy I subscribed. You've done everything I wanted you to, and I'm really glad I backed you. Well done!

Marco Pahl

**Graham Says:** Modesty forbids me from bragging, but I do think we've done a brilliant job in such a short time. I'm especially proud of our giving content away under CC-BY-SA as well; it means that, as well as feeding our families, Linux Voice is producing something of value to society, and that's a really lovely feeling. Thanks a million for your warm wishes.

Issue 1 of Linux Voice is now free as in beer and speech from [www.linuxvoice.com/download-linux-voice-issue-1-with-audio](http://www.linuxvoice.com/download-linux-voice-issue-1-with-audio). Take it, share it, download it, change it, remix it, do whatever as long as you credit us, and we really do hope that it's useful to you.

**LINUX VOICE**  
The magazine that gives back to the Free Software community

April 2014

**PRIVACY PGP**  
Stop President Obama from reading your emails

**RASPBERRY PI BREWPI**  
Brew fine ales with free software. Mmm, beer...

**YE OLDE CODE LOVELACE**  
Tonight we're going to program like it's 1843

**114 PAGES OF NEURAL ENHANCEMENT!**

**THE BEST FREE SOFTWARE 2014**  
Discover the 51 best things about free software right now with our ultimate roundup

**32+ PAGES OF TUTORIALS**

FREE SOFTWARE | FREE SPEECH

**REVIEW MAGEIA 4**  
The KDE 4 desktop is now a beautiful swan

**DON'T BE EVIL OWNCLOUD 6**  
Free yourself from Google's tentacles

ISSN 2054-3778  
9 772054 377001  
April 2014 £5.99 Printed in the UK



YOUR AD  
HERE



Email [andrew@linuxvoice.com](mailto:andrew@linuxvoice.com) to advertise here

# LUGS ON TOUR

## Perl and the Internet of Things at the London Perl Workshop (LPW)

**Josette Garcia** treads London's gold-paved streets to touch the internet of things.

**S**urprise! LPW took place on 8 November – a few weeks earlier than previous years. Otherwise it followed the same pattern – Westminster University during the day and pub in the evening. 250 Perl mongers made it to London on a very damp morning to listen to the latest news and projects in Perl. The attendees had to make very quick choices as the day was split into four tracks plus two workshops – 54 talks or 28 hours' worth!

Tickets to LPW are pay what you want, so nobody is barred from attending, and the workshops are completely free. In his call for papers Mark Keating from Shadowcat Systems shows how unrestricted the conference is:

"We welcome the submission of talks, discussions, presentations and workshops using other languages, or that have separate technical or engineering objectives. The strength in developing is

knowledge from a wide variety of sources. If you use another programming language or system to connect physical and virtual devices together and want to submit or attend the event then we would be honoured to welcome you. If you are connected to a UK Hackspace and wish to attend and display then please approach the organisers with ideas of your requirements."

This year the theme for the London Perl Workshop was Perl and the Internet of Things, focusing on using Perl to control a wide variety of connected devices or the web. This theme brought a variety of very talks and workshops:

- Hakim Cassimaly (aka Osfameron) – Arduino and Perl (4hr workshop).
- Dave Cross (aka Daveorg) – Perl in the Internet of things (2hr workshop).
- Mike Whitaker (Penfold) – Perl and the green Mill House or how



The Perl mongers came from all over the world, including the USA, Germany, Japan, Ukraine and Romania.

to use Perl and some inexpensive sensors to track your house's power consumption (and generation, if you have solar panels).

Matt S Trout announced that DX, his logic programming system for configuration and deployment management appears to work well enough to try and use for something, and that in the name of maximising dogfooding the first something will be a CPAN client.

I overheard that Larry Wall will give a big presentation at FOSDEM about Perl 6; maybe he'll even mention a date on which the whole team will aim to have Perl 6.0.0 ready for release. Prepare to be delightfully surprised!



### TELL US ABOUT YOUR LUG!

We want to know more about your LUG or hackspace, so please write to us at [lugs@linuxvoice.com](mailto:lugs@linuxvoice.com) and we might send one of our roving reporters to your next LUG meeting



# SWAMPFest 2014

Sharon Mitchell reports from Swansea Hackspace's event in honour of EU code week.

**W**e were delighted to play host to 108 ticket holders, who attended for the first ever SWAMPFest held on Saturday 11 October 2014.

The event was conceived to mark the start of EU Code Week, and forge a strong partnership between Swansea Hackspace, SWLUG, Pi Cymru, and Carmarthen Coder Dojo – the fact that the event coincided with the Hackspace's First Anniversary is purely coincidental ;)

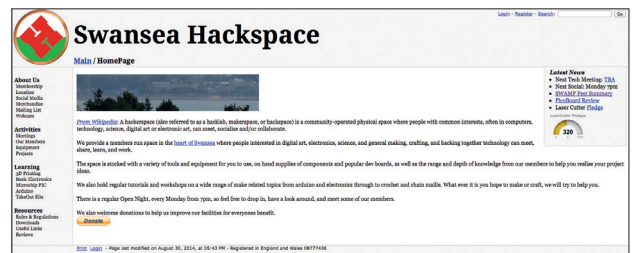
Later in the planning process, digital volunteers were recruited to provide *Minecraft* and Lego workshops on the day.

The day was well attended by male and female, young and old. It was great to see wheelchair and pushchair users being able to access the venue (TechHub Swansea) and to see parents feeling it was something they could bring their young children along to with plenty of workshops catering for all ages, tastes and aptitude

levels; around a third of the attendees were under 16 years old.

The event itself was spread over three floors of the TechHub, with Pi Cymru, Digital Volunteers, and Carmarthen Coder Dojo sharing a large room on the first floor for their all-day drop-in workshops; Pi Cymru's DIY banana piano project was a huge crowd pleaser, and of course the Minecraft and Lego helped entertain the little ones.

The second floor featured the main staging area for the speakers/talks together with a retail/exhibition area. Swansea Hackspace had a table featuring member-made projects; Colin Deady (Ethical Websites/The Mag Pi) had a stall and gave a talk; Representing Code Club was Wales Regional Coordinator Craig Thomas and Maplin's Swansea branch were in attendance. Those lads didn't arrive empty handed to the party either – Maplin has very generously loaned some fantastic new kit to



the Hackspace for members to review/workshop. In keeping with the Event's 'maker' theme, The Lurcher Gallery from Carmarthen were in attendance, showcasing their "Recycling creativity with a Steampunk theme". You can find their Steampunk'd Nerf guns on Etsy shop "Spart1cus".

Floor 4 is the current permanent home of Swansea Hackspace, whose members provided a day-long programme of short taster workshops. These were really popular with the attendees and massively over-subscribed.

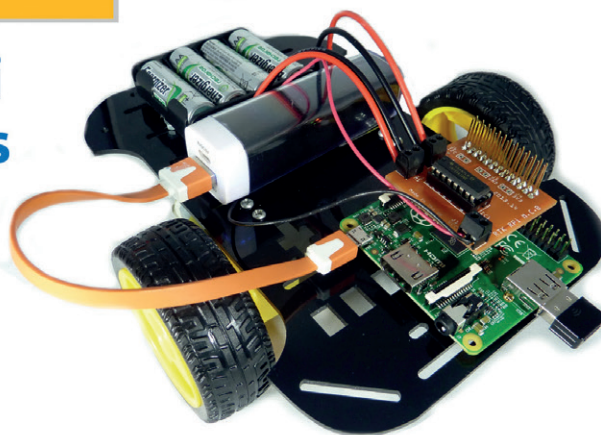
Now to have a break before we plan the next – see you in 2015!

As part of SWAMPFest, Carmarthen Coder Dojo ran Raspberry Pi based Scratch workshops and featured Raspberry Pi- and Arduino-controlled robotics.



## Ryantek LTD

**Raspberry Pi  
Robotics Kits  
Starting  
From  
£24.99**



**Available  
Pre-Soldered  
& With  
Raspberry Pi  
A+**

**Buy Today At <http://store.ryantek.uk>  
And use LINUXVOICE for 5% off your order!**





The Apache Software Foundation celebrated its fifteenth anniversary while we were in attendance.



# CloudStack Collaboration Conference Europe

## The Linux Foundation and the Apache Software Foundation join forces in Budapest.

Despite containing the word 'cloud', CloudStack has much more in common with a typical open source project than the world of big budget cloud hyperbole. It's a collection of software and a management interface that was at one time commercial and then become known as **Cloud.com**. **Cloud.com** began a transition to GPLv3 in May 2010 which was completed later in 2011 after **Cloud.com** was bought by Citrix, the large US-based server/networking company. The entire project was then donated to the Apache Software Foundation and successfully progressed through its incubator program before becoming a fully fledged part of Apache project. CloudStack is now one of Apache's

'top-level' projects, alongside its famous HTTP web server, OpenOffice, Hadoop, SpamAssassin, Subversion and many others.

CloudStack offers 'infrastructure as a service'. That means it manages and deploys the virtual machines that run yours or your customer's operating systems, and eventually, services and software. It also manages and dynamically deploys the resources needed by those machines such as storage and virtual networking. Cloud vendors like to call this 'orchestration', and CloudStack does all of this by harnessing lots of different open source tools, such as the Hadoop Distributed File System, the Mesos distributed systems kernel, the Cassandra database

and the Spark cluster computing engine. It's no coincidence that all of these projects happen to fall under the auspices of the Apache Software Foundation, so it's a natural fit.

It's also a stack of components that's often compared to OpenStack, the dominant platform for open source clouds. Both projects are IaaS solutions and both are used by a wide variety of companies. But they're quite different in implementation, community and marketing budgets. CloudStack suffers from the latter in particular, especially as the Apache Software Foundation is mostly run by volunteers. OpenStack, by comparison, famously had both NASA and Rackspace as initial investors.

"Certainly, it has ramifications," we were told by David Nalley, when we asked about marketing budgets. "In some ways, it's letting the market decide, it's letting the project decide, but not necessarily with the Apache Foundation's financial strength behind any given project."

David is both a CloudStack committer and a member of the Apache Software Foundation, as are many of the attendees here. His honest appraisal on the differences between how both projects are marketed is a great indicator for what this conference, and this community, is like.

"If you want to have a voice in where a project's going, you have to be doing something," he later said, "And the only people who have a voice are the people who have earned the seat at the table."

As we were told in the keynotes, "it's about users being developers," and this must have been why the conference was a relatively informal, collaborative, developer-centric gathering of geeks and geek-related enterprises, where the hotel's corridors are considered a central track and where its people care more about providing a genuine open alternative than whether there was a marketing budget.

**Community meets enterprise**

Remarkably, and completely in contrast to our experience, that seemed to be the prevailing attitude from companies in attendance too. When we spoke to Giles Sirett, for example, CEO of ShapeBlue, the largest independent integrator of Cloudstack, his earnestness sounded familiar.

"People don't know that 75% of the world's websites are delivered by Apache's web server and people don't really know that a good percentage of the world's public clouds are delivered by CloudStack – because it's boring plumbing and it should be boring plumbing," he told us.

It's obvious that a lot of Apache philosophy has gone into CloudStack since its migration from **Cloud.com**, and that's something you can't easily learn without being here. The custodianship of The

A subscription to our magazine was won by Michael Ducy, Global Partner Evangelist at Chef. Let us know what you think, Michael!



Giles Sirett is both the CEO of ShapeBlue, the largest independent integrator of Cloudstack, and a member of the Project Management Committee at Apache CloudStack

Budapest: home of the CloudStack Collaboration Conference and thousands of protesters fighting against an insane internet tax.



**"People don't know that a good percentage of the world's public clouds are delivered by CloudStack."**

Apache Foundation, and respect for its governance model, is seen as a badge of honour for the many of the CloudStack people we spoke to and that has resulted in a project that isn't run by a limited number of vendors, and offers far greater diversity.

Throughout 2014, there were 32,000+ unique downloads from

140 different countries. The historically Citrix-oriented Xen hypervisor shares dominance with KVM (34% to 31%), with CentOS being the most popular host (58%), followed by Ubuntu/Debian (26%) and Red Hat Enterprise (11%). OpenStack, by comparison, is apparently 95% KVM running on Ubuntu.

CloudStack is also being used by companies as varied as Farmville's Zynga, BT and 'a very large satellite broadcaster', with the biggest deployments remaining private. There's a smaller range in the size of deployment, though, with the a third of private cloud's running small 1–50 instances and another third running 100–500. But there are still some 3% with more than 10,000 instances, so CloudStack can scale.

In CloudStack, we've found a genuine open source project and that realisation has surprising consequences; not since Eucalyptus was bundled with Ubuntu server have we wanted to start playing with cloud installations and wanting to write about it. [L]

# FOR THE FIGHT FREEDOM

Free Software isn't just about getting shiny new programs for no cash – it's part of a much larger social movement. **Mike Saunders and Graham Morrison** explore the history and future of FOSS.

**T**here's a problem with the word 'free'. Specifically, it can refer to something that costs no money, or something that isn't held down by restrictions – in other words, something that has liberty. This difference is crucial when we talk about software, because free (as in cost) software doesn't necessarily give you freedom. There are plenty of no-cost applications out there that spy on you, steal your data, and try to lock you in to specific file formats. And you certainly can't get the source code to them.

To make the distinction clearer, many people refer to free (as in liberty) software as a proper noun: Free Software. But it's important to note that Free

Software didn't just pop up as an idea one day, as a "wouldn't it be cool" notion from some hackers in a pub. The principles behind Free Software go back to the early days of computing, and many people have fought long and hard to protect freedom in computing, even when all hope looked lost.

So this issue we want to delve deep into the world of Free Software: where exactly did it come from, why is it important, and what challenges are ahead. We also look at the differences in licences, one of the thorniest issues in FOSS, especially when people have different definitions of "free". But let's start by going back to the early days of computing, when the world was a simpler, happier place...

# FOSS before there was FOSS

Free Software goes back to the 1950s – it just didn't have a name back then.

The idea of releasing software as binary-only executables, without access to the source code that generated them, is relatively new. Yes, commercial software has existed for several decades, but back in the 50s and 60s, as mainframe computers started finding their way into businesses and universities, it was completely normal to get source code with a machine or software package.

Take the UNIVAC 1, the second commercial computer produced in the US: its A-2 compiler was supplied with source code, and customers were encouraged to send their modifications back to UNIVAC. This is FOSS just as we know it, but back in 1953! And it made absolute sense, because improved code was better for users, for the computer makers, and for everyone else who needed data generated by those enormous machines.

So this was the norm at the time, and there are plenty of other examples, such as IBM distributing operating system source code with its mainframes. When Richard Stallman joined the AI Lab of MIT (the Massachusetts Institute of Technology) in 1971, source code was everywhere: "Sharing of software was not limited to our particular community; it is as old as computers, just as sharing of recipes is as old as cooking. But we did it more than most."

But the times were changing. Companies started to see software as commercially viable products, and not just handy things to bundle with hardware. Stallman saw this happening at MIT, where more and more computers were being supplied with proprietary (closed) operating systems. He saw his beloved community of hackers, engineers and sharers being destroyed.

The straw that broke the camel's back was a printer driver: Stallman needed the source code to add some vital features. But



Richard Stallman started the Free Software movement not just to make low-cost programs, but to encourage sharing and benefit the world. (Image: Richard Stallman CC-BY-ND, <https://stallman.org/photos>)

in order to access the source code, he had to sign a non-disclosure agreement, which essentially prohibited him from sharing his improvements with his co-workers. What kind of a world was this becoming, where companies deliberately try to stop you from helping your fellow man? Why set hackers

other attempts by companies to eliminate collaboration and sharing. In 1983, Stallman created GNU (GNU's Not Unix), a new operating system with a Unix-like design, for everyone to share. The announcement is one of the most famous Usenet posts in internet history: [www.gnu.org/gnu/initial-announcement.html](http://www.gnu.org/gnu/initial-announcement.html).

## "Why set hackers against each other, when they could work together to make a better world?"

against each other, when they could work together to make a better world?

So a deeply despondent Stallman had a choice. He could either choose to leave the computing world altogether, or create a new project comprised entirely of software that's free from non-disclosure agreements and

GNU alone wouldn't save the software community, though. Stallman also founded the Free Software Foundation, and created the GNU General Public License, which described software freedom in legal terms and prevented anyone from taking his work and locking it up in proprietary software.

By 1991, much of the GNU system was complete, although the kernel (HURD) hadn't seen much work. However, a non-GNU kernel project called Linux was becoming usable, and paired with the GNU software, a complete operating system could be made. Stallman, and many others from the GNU project, prefer to call the operating system GNU/Linux for this reason, and to emphasise that GNU is a project for computing freedom, and not just some useful bits and bobs that run on "Linux". For brevity we use "Linux" to describe the OS in this magazine, but we appreciate the argument that it should be called "GNU/Linux".

### The BSD alternative

Even while companies were trying to monetise software, code sharing remained common in academic circles. BSD, the Berkeley Software Distribution, was a Unix flavour that started life in 1977. Its source code ended up in legal tangles in the early 1990s, as GNU/Linux was beginning to take off, but the situation was resolved and today we have three major spin-offs: FreeBSD, OpenBSD

and NetBSD. They share a lot of similarities with GNU/Linux, but the licensing is different (more over the page) and the developers tend to focus on the practical aspects of source code availability, rather than societal implications of freedom and sharing. Some BSD fans regard BSD as the original Free Software, and GNU just happened to pick up on it later.

# So many licences...

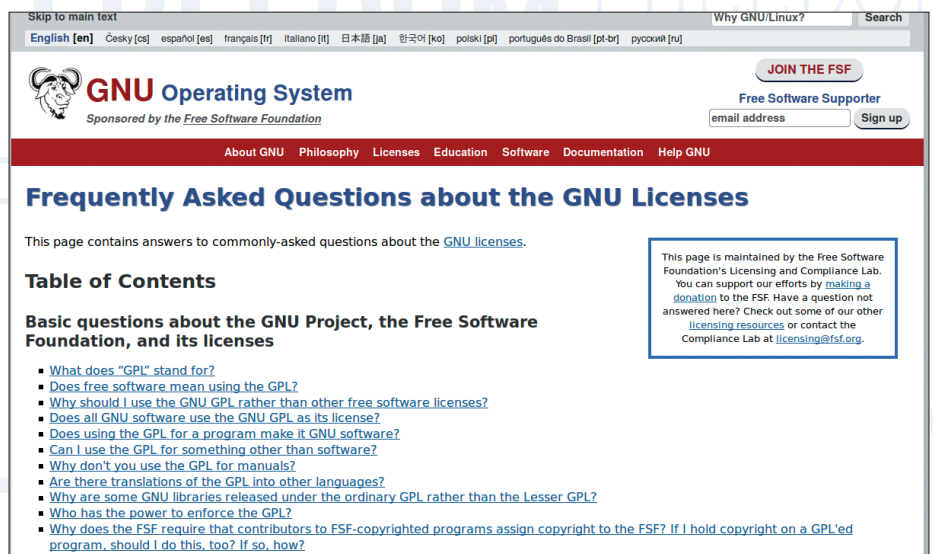
GPL, LGPL, Affero GPL, BSD... there are many ways to make code free (as in liberty).

Free Software, according to Richard Stallman, should grant users four essential freedoms:

- Freedom to run the program for any purpose.
- Freedom to study how the program works (ie look at the source code).
- Freedom to distribute copies to help your neighbour.
- Freedom to distribute your changes in source format.

Now, you could easily knock together a quick 100-word licence based on these preconditions, but to make it last over the years and have a significant legal foundation, you need something longer. This is why Stallman created the GPL, the General Public Licence, which is quite long but makes it very hard for malicious types to subvert it.

Consider, for instance, source code. A dodgy company using GPLed code could release its modifications as



The GNU project takes its licences seriously – the FAQ for the GPL is over 22,000 words long!

assembly language listings, generated by a disassembler. This is, strictly speaking, “source code”, but it’s of little use to

developers who want to incorporate modifications back into the main tree. So the GPL describes source code as the “preferred

## A quick chat with: Richard Stallman

The creator of GNU, the Free Software Foundation and the GPL

**LV** What do you see as the biggest challenges facing Free Software right now?

**Richard Stallman:** Computers designed to make it impossible to run free software. These include Apple and Microsoft phones and tablets, the modem processors of all new portable phones, computers in cars, and so on. Many of them check for manufacturers’ signatures to make it impossible for users to change the software in their own computers.

Also, services that refuse to function except through nonfree apps or nonfree code sent to the browser in a web page. Many of these are nasty in other ways too – for instance, they track people and collect dossiers, thus endangering democracy. See [www.gnu.org/philosophy/surveillance-vs-democracy.html](http://www.gnu.org/philosophy/surveillance-vs-democracy.html).

**LV** Are there any problems approaching that could make a GPL v4 necessary?

**RMS:** Not that I know of.

**LV** From a wider perspective: tens of millions (if not more) of people now benefit from Free Software, and a free platform in the form of GNU/Linux. It’s perfectly possible to do almost every mainstream computing task without being restricted by proprietary software. Obviously there are still some battles to fight, but are you satisfied on the whole? Is there anything else outside of software that you’d like to tackle?

**RMS:** The idea of the free software movement is that users should have control over their computing, so also over software they use. (See [www.gnu.org/philosophy/free-software-even-more-important.html](http://www.gnu.org/philosophy/free-software-even-more-important.html))

Given that nonfree software is nowadays typically also malware (see [www.gnu.org/philosophy/proprietary](http://www.gnu.org/philosophy/proprietary) for examples), a free society calls for replacing all nonfree software with free software.

We have advanced a long way starting from near zero in 1983, but we have a long way left to go. As of yet, we have freed only a small fraction of the inhabitants of



cyberspace, and that is mostly limited to the field of PCs.

**LV** Finally, are you still using the Lemote netbook you had for a while, or have you moved on to the Free Software Foundation-approved refurbished Thinkpad?

**RMS:** It’s called the Gluglug, and yes I have switched. In practical terms it is a lot better. ([www.fsf.org/news/gluglug-x60-laptop-now-certified-to-respect-your-freedom](http://www.fsf.org/news/gluglug-x60-laptop-now-certified-to-respect-your-freedom)).

form for making modifications" – in other words, code in the original language.

The GPL uses copyright law to make sure that the rights to distribute and modify Free Software remain in the code, and nobody can suddenly lock it down under a different license. This strategy is known as "copyleft" in the FOSS world. But there are various versions of the GPL:

- **GPL v2** Provides the rights given above, and is used in Linux (the kernel).
- **GPL v3** As above, but with extra clauses relating to software patents, DRM (you can freely break DRM that's implemented in Free Software) and the right to replace GPLed software on locked-down hardware such as TV set-top boxes.
- **LGPL** The "lesser" GPL, which allows linking with proprietary applications. The GNU C Library (**glibc**) uses this. But why does it exist, when the FSF is against proprietary software? Basically, it's better to have non-free programs using free libraries rather than proprietary equivalents – as it gives users slightly more freedom.
- **Affero GPL** Like the GPL, but if you run GPLed software on a server and users run

communicate with it (like a web app), users should also have the right to access the source code.

There are other GNU licences, such as for documentation, with the full list at [www.gnu.org/licenses](http://www.gnu.org/licenses). Interestingly, Linux (the kernel) hasn't upgraded to the GPL v3 due to objections from Linus Torvalds. He doesn't think it's wrong if hardware

licences. The most notable is the BSD licence, used by FreeBSD among other projects, at just 233 words. This basically says: do what you want with the code, but credit the original source, and don't sue us for anything that goes wrong.

Now, this leads to an involved philosophical debate about which licence is more free. From one side, FreeBSD fans would argue that their licence is the freest, as it enforces fewer restrictions on its users. You really can do what you want with the code, including folding it into proprietary software, just like Sony did with its PS4 operating system (which was based on the FreeBSD kernel).

GPL fans counter with: yes, the GPL has more restrictions, but these are put in place to maintain the user's freedom down the road. The GPL is the freer licence as it actively fights for freedom.

Who's right? The arguments will go on for years, no doubt. But the general consensus tends to be that the BSD licence provides more freedom for developers, while the GPL is better for end users.

## "Why set hackers against each other, when they could work together to make a better world?"

manufacturers want to restrict users from modifying software, noting that he installs Linux on his children's computers, and has the right to stop them from upgrading it. The GPL v3 is "overreaching" accordingly to Torvalds, and isn't "morally" where he wants to be. (See his full explanation at <http://tinyurl.com/npmfvwz>).

### Free, or even freer?

While the GPL v3 is over 5,600 words long, there are alternative and much simpler

## A quick chat with: **Microsoft!**

**Gianugo Rabellino, senior director of open source communities at MS Open Tech.**

**LV** Microsoft today is heavily involved in various open source projects and releases a lot of code under OSS licences. What brought about the change in attitude since the early 2000s? Is it a grass-roots campaign in Microsoft, or a bigger corporate strategy?

**Gianugo Rabellino:** Microsoft sees openness as a way to satisfy our customers and grow our business. This involves enabling open source applications to run better on and with our Microsoft platforms, but also to deliver great Microsoft experiences to other device platforms.

Our open source strategy has evolved based on conversation with our customers, many of whom operate heterogeneous IT environments with traditional commercial software, commercial open source software and community-based open source software working side-by-side.

So just how far we have come? Our CEO Satya Nadella recently said "Microsoft loves Linux," and described how there are 1,000 Linux virtual machines to choose from for

Microsoft Azure, and that Linux and various packages of Linux comprise 20% of Azure's workloads. But those who follow our Linux work more closely will know that we've had Microsoft engineers actively contributing to the Linux kernel for over five years.

Openness is increasingly becoming part of the company's DNA – multiple teams across Microsoft are involved in open source, standards and interoperability efforts. It's both a top-down and bottom-up approach – with customers and developers at the centre.

**LV** How is open source being used in Microsoft now? What would you describe as the company's biggest open source projects?

**GR:** Microsoft currently participates in over 800 open source projects on GitHub, and that number is growing. We work with many open source communities to identify valuable opportunities, projects and initiatives in which we want to participate, often focusing on improving open source



application interoperability with our products, and using an open source development approach when it makes sense for specific products and solutions.

One of our most significant open source projects was our recent announcement that Microsoft is open sourcing the full server-side .NET stack and expanding .NET to run on the Linux and Mac OS platforms. A large chunk of .NET was already open in the ASP.NET family of technologies, and this change builds from that successful initiative.

# Real GNU/Linux distributions

Our recommendations for maximum freedom.

**W**e know there are hundreds of distributions to choose between. But there are far fewer choices and some compromises to make if you want to use a GNU/Linux distribution that's endorsed by The Free Software Foundation for adhering to Richard Stallman's guiding principles. Choice is reduced because the Linux kernel can't contain any of the proprietary blobs of firmware that are tolerated by most other distributions. These kernels are given the name 'libre', and they can have an impact on hardware compatibility and performance.

If there's no open source driver, you'll also need to invest in different hardware. This used to be a much bigger problem 10 years ago, with many modems, wireless dongles, printers, touchpads and graphics cards rendered useless without their manufacturer's proprietary blobs. Fortunately, the Linux kernel is in much better shape, and most modern hardware we use will 'just work', which means there's a good chance a free distribution won't require new hardware unless you're using something esoteric. The main decision is which distribution to try, and while there are quite a few, not many receive the same number of updates you'd expect from an active distribution, which is why we're only going to look at four.

## Trisquel 7.0 LTS

It used to be the case that 'free' GNU/Linux distributions weren't as usable as their non-free counterparts. That could make switching difficult for non-technical users. That things have changed is partly thanks to what we'd consider the most popular GNU-centric distribution, Trisquel. Trisquel has been downloaded 344,786 times since its 2.0 release, and now uses Ubuntu as its foundation, making it an easy migration for millions of Ubuntu users. The latest release is a re-working of the 14.04 Long Term Support version of Ubuntu, which means you'll get updates until 2019. There's a wide variety of download choices, from a 3 GB ISO that includes source code to a 25MB ISO that needs a network installation. We opted for the 1.5GB DVD image, which can also operate as a live desktop. Its Gnome-based installer looks amazing, and while it's great that the *Orca* screen reader was



The biggest difference between these distributions and the ones we usually cover is their emphasis on freedom – that sometimes means sacrificing features, but it's in a noble cause.

enabled by default and speaking many of the options you make and see on-screen, we couldn't find an easy way to disable it (it's Alt+Super+S). And that's all there is to installation. Within minutes, we had our new system up and running.

Trisquel defaults to running Gnome in its classic mode, and like the installer, we really like the appearance of the default theme. Most of the default applications are identical to a standard Ubuntu release. There's *LibreOffice*, *Rhythmbox*, *Gimp* and *Evolution*. The web browser is based on version 33 of *Firefox* and it's called *Abrowser*. It worked well for us, defaulting to DuckDuckGo for searches as well as offering clear options for disabling JavaScript or installing the more privacy focussed *GNU/IceCat*.

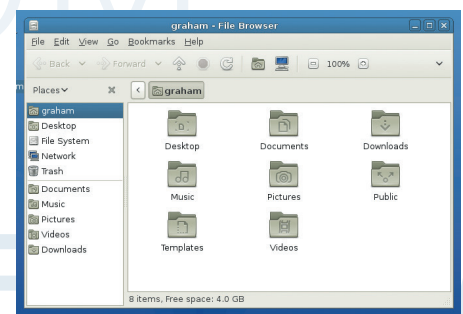
<http://trisquel.info>

## gNewSense 3.1

Second to Trisquel in popularity, gNewSense is a little more austere in the appearance category. This is primarily because it's using a little-themed version of Gnome 2.x and older packages than most distributions, and this is because of its choice of base distribution. After a few years of using Ubuntu as its base, version 3 switched to Debian 6, first released in 2011, and many of

the version numbers of packages stretch back from that point – hence Gnome 2.x. Like Trisquel's Ubuntu repositories, in Debian that many packages can be installed very easily, and because the non-free repositories are disabled automatically, you don't have to worry about. Despite its age, installation is straightforward although slightly more intimidating than Trisquel. You're asked for your network's DNS address and manual confirmation of how your partition table is going to be generated, for example.

Using this old version of Gnome is a reminder of how much has changed. It's quick and functional, but doesn't have any of the bells and whistles of newer versions



Based on an older version of Debian, gNewSense looks a little dated and may not work on the latest hardware.



– we still find *OpenOffice* here rather than *LibreOffice*, for instance, and the desktop doesn't look anything like as good as Trisquel. The older kernel, 2.6.32, is a little more worrying. Trisquel sports version 3.13, complete with low latency patches and bfq scheduling, but more importantly, many more hardware drivers and updates, making gNewSense less likely to work with modern hardware, at least until it catches up with the latest Debian release.

[www.gnewsense.org](http://www.gnewsense.org)

## Parabola

Parabola is a relatively recent addition to the Linux Foundation's list of free distributions, being ordained in 2011, and it's a little different to both gNewSense and Trisquel. This is something you find out within 30 seconds of booting the 500MB Live ISO because you're dropped as root into the command line and curtly told that if you want to install Parabola, you'd better have a working networking connection, and that to work out how this is done, open **network.html** into *Lynx*. Things could be worse. They could have insisted on loading the html file into *Emacs*.

If this sounds familiar, it's because Parabola is built atop Arch, a distribution that's spawned a couple of 'libre' kernel distributions. You can even migrate from a regular Arch installation if you'd rather rid yourself of those pesky proprietary bits. We like Arch a lot here at LV Towers, but it's not for the uninitiated. Fortunately, the barking words thrown onto the screen at login are worse than their bite, and we

```
Parabola GNU/Linux-libre 3.10.10-1-LIBRE (parabolaiso) (tty1)
parabolaiso login: root (automatic login)
```

```
-----
Parabola live media 2013.09.01
```

```
To install Parabola, the system must be connected to the internet.
For instructions, enter this command:
lynx network.html
```

```
Press the number keys while holding Alt to switch virtual terminals.
This allows entering commands without closing lynx.
```

```
-----
root@parabolaiso ~ # _
```

It's quite a shock how little a 500MB actually gets you these days. Judging from the text, maybe not even a network connection.

found networking already up and running, putting off our Herculean struggle with Systemd for another day. And while you'll need to configure and install everything else you want to use, including a graphical

## Musix v3.0.1

The three distributions we've looked at so far have been functional and modifiable just like any other distribution. Musix is a reminder that not all 'libre' distributions need to focus on sober functionality, and it does this by being a Debian-based distribution designed for music and media creation, which we think is a great idea. There's not too much choice on the download medium, and the

**“There are some compromises to make if you want to use a distro that's endorsed by the FSF.”**

environment, and carefully follow the installation instructions, this is still a wonderful distribution. In some ways, building your own installation with an Arch distribution is a great way of appreciating the amount of work that goes into creating a working system, especially when you know that every package is untarnished.

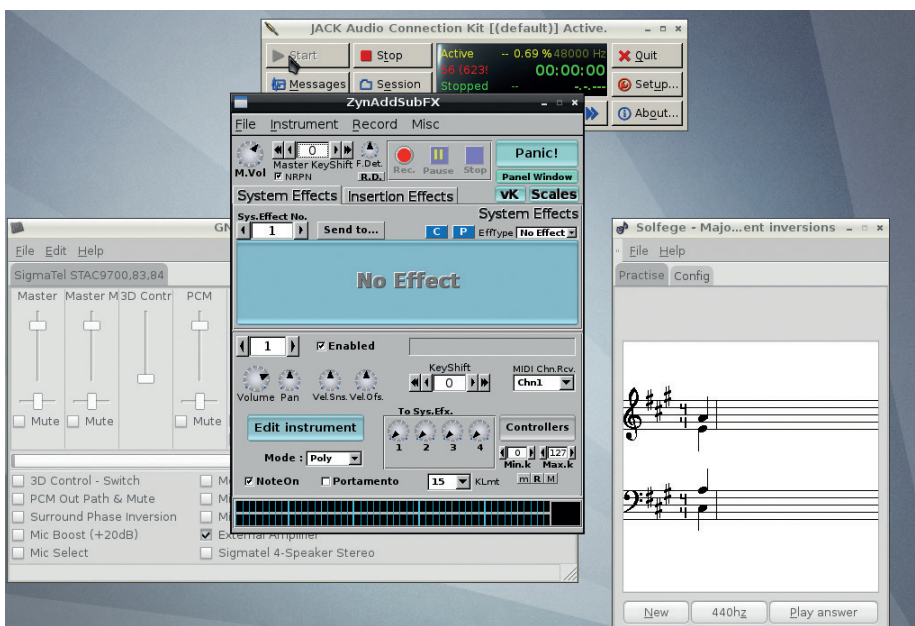
<https://www.parabola.nu>

2GB download can take a while from the limited server capacity, and unlike the other three distributions we've looked at here, there's no torrent we could fine. Installation is easy though. The Live DVD defaults to Spanish, but there's English, French and Portuguese too, and they're all selectable from the boot menu, which is an excellent idea. This is also the only distribution we've looked at that boots to an augmented KDE desktop (username: **live** password: **user**).

There's pretty much every audio application and effect you've ever heard of installed, along with some lots of other multimedia tools like *Kdenlive* for video editing and *Blender* for 3D generation. The most important feature is that the Jack audio system is already running, and you can control it's parameters and connections with the *QJackCTL* application that's also included.

One tool we'd not seen before is *GNU Solfege*. This is music educational and training tool. It can play intervals and rhythms, for example, and ask you to identify them, you can create and train yourself about scaled, and chords and keep on top of your progress. The user interface is simple, but it's crammed full of essential content that can really help.

<https://musixdistro.wordpress.com>



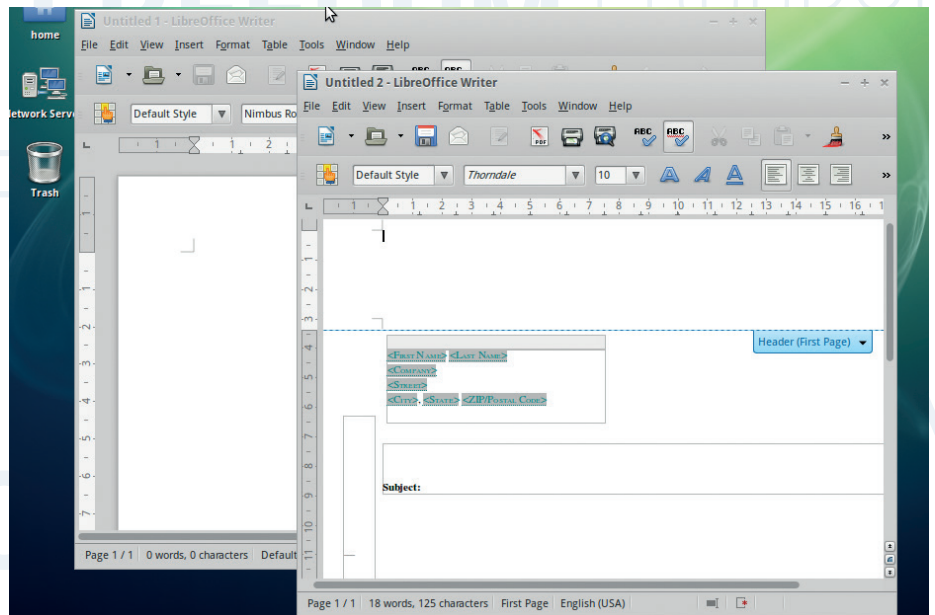
Forget about proprietary plugins and formats with a music-making distro focussed on freedom.

# LibreOffice vs OpenOffice.org

Our essential office suite proves that Free Software licences are important.

The recent history of both the *OpenOffice.org* and *LibreOffice* projects encapsulates a lot of what is good in open source philosophy, and what wider good can be achieved. We think it's also a great example that exposes many of the issues brought about when these kinds of projects are large and successful, and how they interact with both their community and their corporate sponsors.

Having an 'office' suite of applications for Linux has always been absolutely vital. At work, we all know that text documents and spreadsheets spend their lives in perpetual motion between colleagues' email accounts, and nearly of these documents will have been created by *Microsoft Word* or *Microsoft Excel*. These two applications are a cornerstone of Microsoft's still unrivalled business strategy and unapproachable influence, and they have been dominant for over 20 years. As such, they've been fiercely guarded jewels in Microsoft's crown. Microsoft famously blocked its rival, IBM, from selling Windows 95 in an attempt to undermine IBM's own office suite, Lotus SmartSuite. Late in the same decade,



*LibreOffice* has supplanted *OpenOffice* in nearly all Linux distributions, but it still faces a battle for recognition outside of the open source community.

another rival, Sun Microsystems, acquired a commercial office suite called *StarOffice* and open sourced the code in an attempt to subvert the influence of Microsoft's

*Office*, eventually releasing version 1.0 of *OpenOffice.org* in May 2002.

*OpenOffice.org* has always had a strong focus on embedding excellent import and

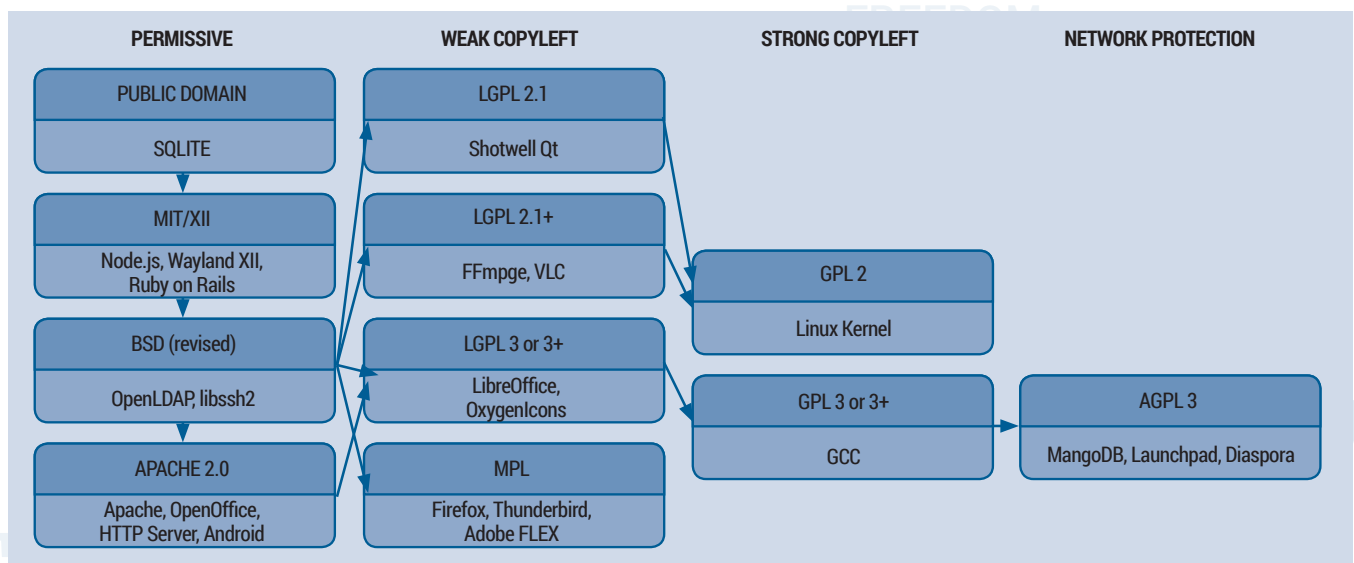
## The open source licence spectrum

Not all licences are the same. Some enable you to do more things than others. With a BSD-style licence, for example, you can often create proprietary code without having to release your own changes. This is what allowed Apple to take parts of FreeBSD and NetBSD for its own Mach kernel without having to provide its own changes.

This is great in certain circumstances, especially when a developer just wants to get their idea out there, but the Free Software movement is built upon the users of that software having the same access to the new developments, and that means having access to the code and being able to make their own modifications. These licences are less

permissive and are stronger 'copyleft'. Finally, we have the Affero General Public Licence, which is recommended by the Free Software Foundation when code is running across a network.

Based on a 2007 illustration by David A Wheeler (CC BY-SA 3)



export compatibility with Microsoft's own formats, which meant that not only did Linux inherit a fully fledged office suite, it also gained vital compatibility with the file formats everyone was emailing between themselves. This has subsequently helped Linux become a real viable alternative to Windows, as shown in many places such as Munich city council's LiMux project. But more importantly, it has also been instrumental in making its OpenDocument Format (ODF) an ISO standard, and it could be argued that *OpenOffice.org's* viability for document editing and interoperability has paved the way for a change in attitude for many institutions who previously saw Microsoft's applications and formats as the only possible options.

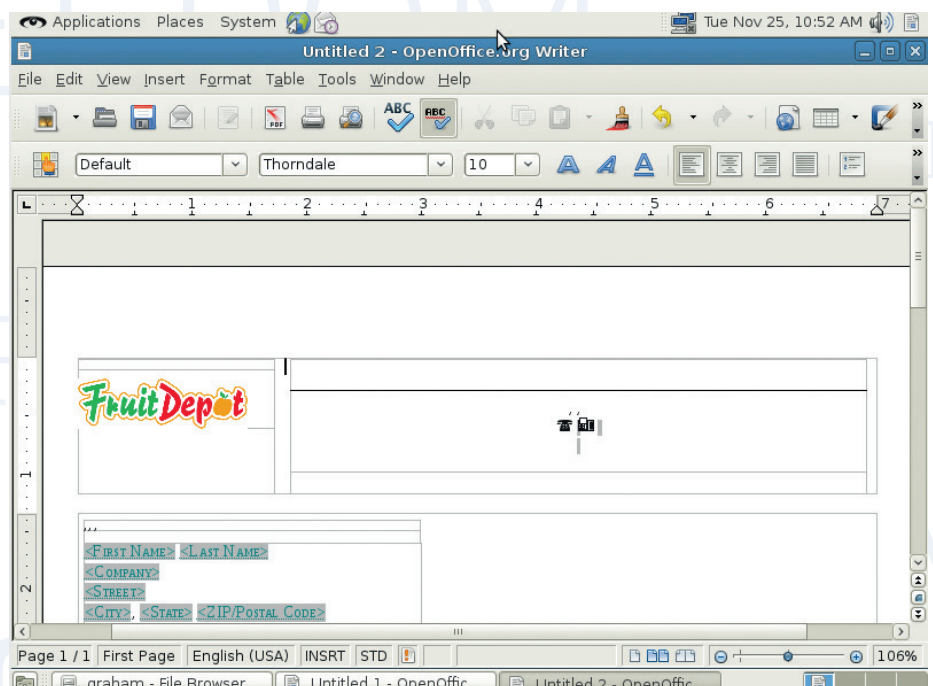
## Oracle

We'd argue that none of this would have been possible without *OpenOffice.org* being an open source project (LGPL v3 has been used since version 2), and the project is significant because the licence has both ensured its openness and its survival. Oracle acquired Sun Microsystems in 2010 and muddled the future of many of Sun's open source projects like *OpenOffice.org*, *MySQL*, *VirtualBox* and of course, *Java*.

Oracle also reduced the number of developers working on what was now called Oracle Open Office, and this apparent lack of progress led to what's best described as a 'fork', in a similar way that Xfree86 became X.org, and both forks were only possible because of the licences used to host and share the source code.

Names, such as *OpenOffice.org*, are trademarked and not typically part of the open source side of a project. The Document Foundation (TDF) was created to take control over the project, after initially hoping that Oracle would rather hand over *OpenOffice.org* to TDF than run it itself. When this didn't happen, The Foundation created its own fork of the source code and voted to rename the liberated version *LibreOffice*, which has since gone on to replace *OpenOffice.org* in all of the major Linux distributions.

However, the story doesn't end there. *OpenOffice.org* still has some powerful brand recognition and is still downloaded and used in many places, despite *LibreOffice's* superiority, and Oracle did eventually decide to give the project away, *OpenOffice.org* was given to The Apache Software Foundation, a *bona fide* repository for open source projects, and this donation must have left it



*OpenOffice* is in the very capable hands of The Apache Software Foundation, but a merge with *LibreOffice* now seems almost impossible.

with something of a quandary. On the one hand, the Apache Software Foundation now had an important piece in the free software puzzle under its control, a gateway suite for people migrating from proprietary systems. On the other hand, *OpenOffice.org* had been morally supplanted by *LibreOffice* in the hearts of many open source users. The

differences in their licensing begin to have an effect. *Apache OpenOffice* uses the generally more liberal Apache Licence, as you might expect. *LibreOffice*, by contrast, has inherited LGPLv3 and the MPL 2.0 (Mozilla Public Licence). This makes moving code from one project to other the much easier in one direction – from the more

liberally licensed code to the less liberally licensed code, and that means from *Apache OpenOffice* to *LibreOffice*.

It's a shame that we have two such similar projects, but it's difficult to see how it could have happened any other way

**“As users and advocates, we still have LibreOffice, which is by far the most important thing.”**

Document Foundation also made it clear that it had no intention of shifting direction when it made its own statement shortly after Oracle's relicensing, “The Document Foundation and *LibreOffice* represent already a future path of development for the *OpenOffice.org* community and the *OpenOffice.org* code base, as was originally announced on September 28, 2010.”

## OpenOffice rides again

The Apache Software Foundation has since developed the suite on its own, which has revealed another final twist to the saga. Without the same resources, *OpenOffice* development has been slower; both projects have worked on their own aspects to the code, but there's also some sharing. Unfortunately, this is always where subtle

without Oracle donating the code to The Document Foundation at an early stage. That this didn't happen could have simply been a lack of understanding at Oracle, or the subtle machinations of a large corporation with its wide and convoluted approach towards the open source projects it curates. Either way, as users and advocates, we still have the project and the software, which is by far the most important thing, and something that would never have happened with a similar proprietary piece of software.

Despite all this manoeuvring and strategy, and despite the long history of the software, it's open source that has ensured its survival and continued growth. And we don't think there's any other system that could have produced the same result. ☐

# THE LINUX VOICE PUB QUIZ

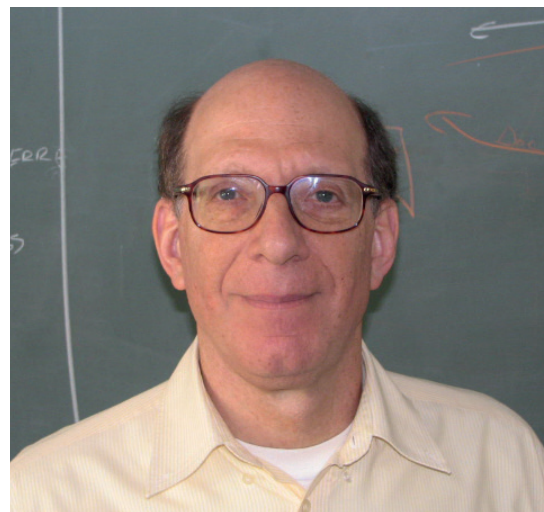
Get a few friends together, settle down by the fire and pour yourselves an ale. It's time to test your Linux knowledge with a little help from our Quiz Master, **Gnomish Armorar**.

## The Linux kernel

- 00** Andrew S Tanenbaum and Linus Torvalds famously argued about the size of their kernels. What was the subject of Professor Tanenbaum's 1992 email rebuttal?
- A) You are insane B) Linux is monolithic C) Linux is a microkernel D) Linux is obsolete
- 01** Which company did Linus colourfully describe as being the "single worst we've had to deal with" back in 2012?
- A) Intel B) Nvidia C) The Linux Foundation D) AMD
- 02** How did Linus Torvalds describe Systemd developer Kay Sievers?
- A) Drooling moron B) Pathetic moron C) F\*cking moron D) Needing a little constructive help
- 03** Linus Torvalds has said "I'm an egotistical bastard, and I name all my projects after myself," but which of the following isn't one of his projects?
- A) Scrot B) Subsurface C) Linux D) Git
- 04** In what month of 1991 did Linus announce his kernel with the words "Just a hobby, won't be big and professional like GNU."
- A) August B) September C) October D) November
- 05** How many lines of code were in the 0.01 release

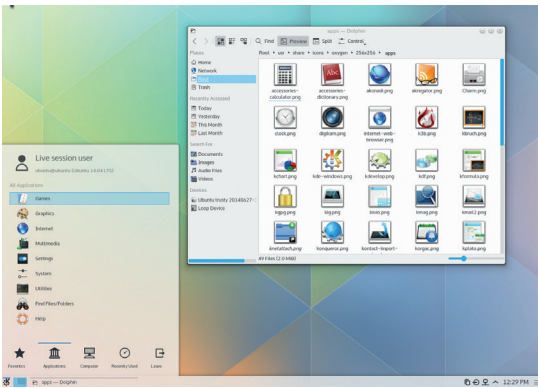
## of the kernel?

- A) 91,023 B) 23,910 C) 39,102 D) 10,239
- 06** Which of these is a real kernel error?
- A) "Here's a nickel kid. Go buy yourself a real



Andrew S Tanenbaum originally thought Linus was onto a losing gambit with his newfangled operating system.

Photo credit: GerardM (CC BY-SA)



Will 2015 be your year of Linux on the desktop?

computer." **B)** "So long and thanks for all the fish" **C)** "Game Over" **D)** "I came, I tried, I crashed"?

**07** In which order were the following Linux sound architectures developed: ALSA, OSS, PulseAudio and Jack?

**A)** AOPJ **B)** OPAJ **C)** OAJP **D)** OAPJ

**08** In The Linux Foundation's 2013 report on kernel development, who signed off the largest percentage of patches?

**A)** David S Miller **B)** Linus Torvalds **C)** Greg Kroah-Hartman **D)** Andrew Morton

**09** According to the same report, which group had the largest number of patches signed off?

**A)** Intel **B)** Red Hat **C)** The Linux Foundation **D)** Google

**Desktops**

**10** What did the K in KDE probably stand for?

**A)** Konzept **B)** Kool **C)** KDE **D)** Kitchen

**11** What does the M represent in Gnome?

**A)** Model **B)** Matthew **C)** MIME **D)** MOSAIC

**12** What does the X represent in Xfce?

**A)** X11 **B)** X.org **C)** XForms **D)** XWindow

**13** How do most Gnome developers pronounce it?

**A)** Nome **B)** Gunome **C)** Nomay **D)** Gunume

**14** And how should you say 'Qt' to Lars Knoll?

**A)** Cut **B)** Cutes **C)** Cutey **D)** Cute

**15** Which of these isn't a KDE application?

**A)** Konqueror **B)** Konversation **C)** Kanada **D)** Belle

**16** Which of these doesn't use GTK+?

**A)** Mozilla Firefox **B)** Google Earth **C)** The Gimp **D)** Xfce

**17** When was Qt finally released under an FSF approved licence?

**A)** 1997 **B)** 1998 **C)** 1999 **D)** 2000

**18** Miguel de Icaza is the co-creator of Gnome. What does he now help develop?

**A)** Mono **B)** Swift **C)** Python **D)** BlitzBasic

**19** Which desktop environment is Linus Torvalds currently using?

**A)** KDE **B)** Gnome 2.x **C)** Gnome 3 **D)** Mate

**Raspberry Pi**

**20** Which 80s computer does the Pi partly take its inspiration from?

**A)** Acorn Electron **B)** Acorn BBC **C)** Acorn Archimedes

**D)** Acorn Atom

**21** Which company did Paul Beech, the winning designer behind the Pi's logo, co-found?

**A)** Pimoroni **B)** Adafruit **C)** Ryantek **D)** Ragworm

**22** What is really beneath the two leaves in the Raspberry Pi logo?

**A)** A delicious raspberry **B)** A disco light ball thing **C)** A buckyball **D)** The meddling monk

**23** Which famous British games designer is a co-founder of the Raspberry Pi Foundation?

**A)** Andy Braybrook **B)** Geoff Crammond **C)** David Braben **D)** Jeff Minter

**24** When could you first order a Raspberry Pi?

**A)** 29 February 2012 **B)** 29 March 2012

**C)** 29 April 2012 **D)** 29 May 2012

**25** How much RAM did the original Model Bs ship with?

**A)** 32k **B)** 256MB **C)** 512MB **D)** 1024MB

**26** How many Raspberry Pis have been sold (as of late 2014)?

**A)** 1 million **B)** 2 million **C)** 3 million **D)** 4 million

**27** Which of the following hasn't been attempted with a Raspberry Pi (yet!!)?

**A)** Brewing beer **B)** Raspberry Pi submarine **C)** An autonomous Atlantic crossing **D)** Sent to near space

**28** What's the name of the Pi's easy to use operating system installer?

**A)** Noobs **B)** ezboot **C)** AcornDFS **D)** ddrescue

**29** Which software isn't available for the Raspberry Pi for free?

**A)** Scratch **B)** Mathematica **C)** Minecraft

**D)** Microsoft Office

**Ubuntu**

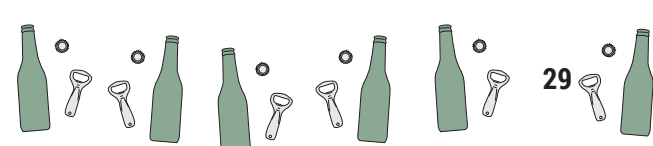
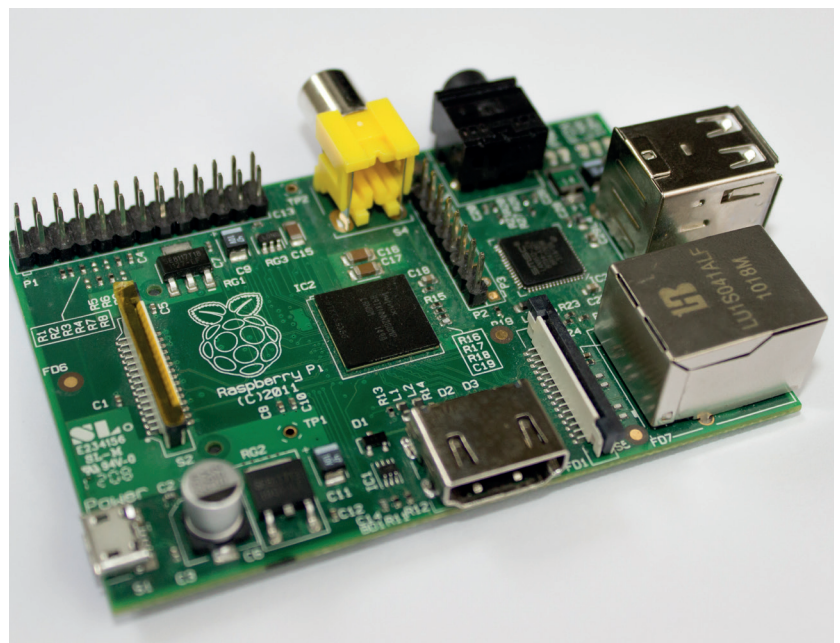
**30** What was the code name for the first release of Ubuntu?

**A)** Hoary Hedgehog **B)** Warty Warthog

**C)** Breezy Badger **D)** Dapper Drake

**31** How much did Mark Shuttleworth reportedly sell

We're still waiting for a Pi that can fit into an Altoids tin.



his security startup, Thawte, to VeriSign?

- A) \$574 million B) \$57.4 million C) \$5.74 million
- D) \$5.74

32 Mark Shuttleworth was which numbered space tourist?

- A) First B) Second C) Third D) Fourth

33 What does Ubuntu mean?

- A) I'm loving it B) I am what I am C) Do no evil
- D) Humanity to others

34 Where did Mark Shuttleworth go to strategise before launching Canonical?

- A) Antartica B) North Pole C) Isle of Man
- D) Cape Town

35 What is Mark Shuttleworth's latest project?

- A) A South African astronauts fund B) The Ubuntu Watch
- C) Botanic gardens D) Solar balloons

36 In 2011, how many Ubuntu users did Mark Shuttleworth say there'd be in 2015?

- A) 200 million B) 100 million C) 50 million
- D) 25 million

37 What percentage of the Ubuntu Edge super smartphone was funded through Indiegogo?

- A) 30 B) 40 C) 50 D) 60

38 When did Canonical announce the Ubuntu TV?

- A) 2011 B) 2012 C) 2013 D) 2014

39 Which is going to be the first Ubuntu release with Mir as default?

- A) 15.04 B) 15.10 C) 16.04 D) 16.10

**Picture Round - see inside back cover**

40 Which command produces this output?

- A) fortune | echo B) cowsay | fortune C) fortune | cowsay
- D) cowsay

41 Where on earth is Damian Conway?

- A) Bath Abbey B) Westminster Abbey C) Princess Theatre, Melbourne
- D) Buckingham Palace

42 Who is this?

- A) Richard Stallman B) Fred Durst
- C) Mark Shuttleworth D) Jono Bacon

43 Which Raspberry Pi is this?

- A) Model A B) Model B C) Model A+ D) Model B+

44 Which open source celebrity have we unleashed our Gimp skills upon?

- A) Tim O'Reilly B) Tim Bray C) James Bottomley
- D) Julian Assange

45 Which ancient Linux distro is this?

- A) Red Hat 5.1 B) Mandrake 5.1 C) Slackware 7
- D) Debian 2.0

46 Which desktop is Knoppix using in this release?

- A) GNOME B) KDE C) LXDE 7.2 D) XFCE

47 Which ancient version of Ubuntu is depicted here?

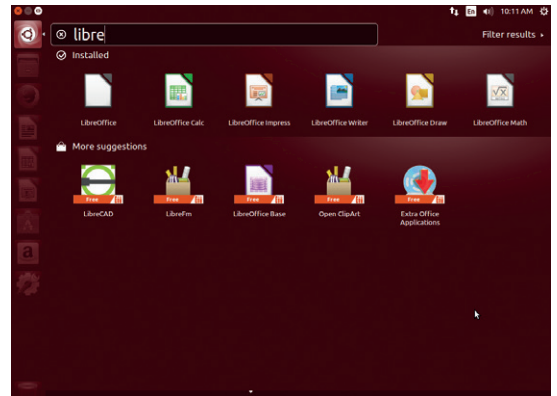
- A) 4.10 B) 5.10 C) 6.06 D) 6.10

48 What music application are we using in this screenshot?

- A) Rhythmbox B) XMMS C) Clementine D) Amarok

49 Who is this?

- A) Richard Stallman B) Fred Durst
- C) Mark Shuttleworth D) Jono Bacon



Ubuntu is still perhaps the worlds' most popular GNU/Linux distribution.

**Bash Commands**

50 What are the target file's permissions after the following command: `chmod 765 filename`?

- A) -rwxrwx-r-x B) --rw-r--r-- C) -rwxrwx-r-- D) -rwxrwxrwx

51 Which of these vim :help arguments is not an easter egg?

- A) holy-grail B) 42 C) ! D) me

52 What does the 'rev' command do?

- A) Increments a version number B) Reverses input
- C) Reverts to a backup D) Makes a car sound

53 What's the systemd equivalent to 'tail -f /var/log/messages'?

- A) journalctl -f B) tail -f /var/log/systemd
- C) systemd --viewlog D) cat /dev/sda2

54 Which character in Bash is used to redirect output to a file?

- A) < B) << C) > D) >>

55 Which character redirects a file's contents to the standard input of a command?

- A) < B) << C) > D) >>

56 What's the Bash keyboard shortcut for searching backwards through your history?



Debian is the largest community supported Linux distro.



A) Alt S B) Alt H C) Ctrl S D) Ctrl R

57 What command would you use to perform simple arithmetic?

A) \$((56 - 14)) B) echo '56-14' bc C) bc '56 - 14'  
D) calc '56 - 14'

58 If you wanted to process audio on the command line, what would you use?

A) shoez B) spex C) soundz D) sox

59 Which command would you use to change your password?

A) pwd B) pwnd C) passwd D) password

**Debian**

60 Who was Ian Murdock's girlfriend when he started Debian?

A) Debra B) Debbie C) Deborah D) Deidre

61 When was Debian first announced?

A) 1992 B) 1993 C) 1994 D) 1995

62 Which distribution isn't derived from Debian?

A) Knoppix B) Linux Mint C) Mageia D) Ubuntu

63 Which film series characters are Debian releases often named after?

A) Toy Story B) Back to the Future C) Stargate  
D) Krzysztof Kieslowski's Three Colours

64 How many CPU architectures does Debian 7.0/ Linux support

A) 11 B) 3 C) 9 D) 15

65 Which boot system does Debian use?

A) System V B) Upstart C) Systemd  
D) Don't even go there

66 Which other kernel doesn't have a Debian version?

A) GNU/Hurd B) FreeBSD C) Minix D) Linux

67 Which of the following is not in Debian's Social Contract?

A) Community politeness B) User focus  
C) non-free compatibility D) Problems aren't hidden

68 Who is the longest running project leader of Debian?

A) Ian Murdock B) Stefano Zacchiroli  
C) Bruce Perens D) Bdale Garbee

69 When was Debian 1.0 released?

A) June 1996 B) December 1995 C) It wasn't, due to a CD creation fault  
D) January 1997

**Hardware**

70 What was the first laptop top be awarded the FSF's Respects Your Freedom award?

A) Dell XPS 13 B) Gluglug X60 C) Lenovo ThinkPad



Robots distract our attention by hosting themselves within things we enjoy.

T42 D) Rhino E6540

71 Which of the following has never suffered a Linux installation?

A) a Mars Rover B) International Space Station  
C) a toaster D) a sniper rifle

72 Which kernel version re-integrated the changes made by Android?

A) 2.6.35 B) 3.0 C) 3.3 D) 3.5

73 Which Bruce created BusyBox, as used by many set-top-boxes, NASs and routers?

A) Wayne B) Forsyth C) Lee D) Perens

74 Which hacker co-created both Samba and rsync?

A) Ted Ts'o B) Andrew Tridgell C) Jeremy Allison  
D) Linus Torvalds

75 Which is the best source of entropy in a standard Linux installation?

A) /dev/random B) /dev/urandom C) /dev/null  
D) /dev/zero

76 How many of TOP500's top ten supercomputers (Nov 2014) run Linux?

A) 7 B) 9 C) 8 D) 10

77 Which laptop hasn't Linus Torvalds used for travelling (as far as we know)?

A) Macbook Pro B) Pixel Chromebook C) Macbook Air  
D) 11-inch Sony Vaio Pro

78 What is the coreboot project?

A) A clothing alliance B) An SSD cache C) A BIOS replacement  
D) A systemd replacement

79 Which of the following is not a boot manager?

A) UEFI B) Lilo C) Grub D) Refind

**Answers Let us know how you get on!**

- 70) B 71) A 72) C 73) D 74) B 75) A 76) D 77) A 78) C 79) A
- 60) A 61) B 62) C 63) A 64) A 65) C 66) C 67) A 68) B 69) C
- 50) A 51) B 52) B 53) A 54) C 55) A 56) D 57) B 58) D 59) A
- 40) C 41) B 42) D 43) C 44) A 45) B 46) A 47) A 48) D 49) A
- 30) B 31) A 32) B 33) D 34) C 35) C 36) A 37) B 38) B 39) C
- 20) B 21) A 22) C 23) C 24) A 25) B 26) D 27) B 28) A 29) D
- 10) B 11) A 12) C 13) B 14) D 15) C 16) B 17) D 18) A 19) C
- 00) D 01) B 02) C 03) A 04) A 05) D 06) A 07) C 08) C 09) B



**Les Pounder** looks back on the glory days of a Linux podcast that took no prisoners. So farewell then, Linux Outlaws...

**I**n 2007 there was a new Linux podcast on the scene, hosted by two regular Linux users who loved talking. Dan Lynch and Fabian Scherschel rose to become two of the most loved podcast presenters in the Linux community. Their podcast, Linux Outlaws, became a much loved and cherished show that both informed and entertained its listeners with a mix of news, reviews, interviews and several rants. Just recently the team behind Linux Outlaws announced that the show would be ending in December 2014 and we were privileged to have the chance to interview the outlaws for their last stand.

**“Linux Outlaws informed and entertained its listeners with a mix of news, reviews and rants.”**

From the community, may we say thank-you to Linux Outlaws for seven years of great shows and wish them good luck in their future projects.

*[Editor's note – although we intended these pages to provide a look at the best Free Software podcasts around to fill the void left by the departure of Linux Outlaws, we somehow neglected to mention the Linux Voice podcast. Rising phoenix-like from the ashes of the once-popular TuxRadar podcast, Linux Voice fills the airwaves once a fortnight with a mélange of new-ish news, opinions and contributions from our insightful, intelligent and beautiful listeners. Right, carry on...]*



# Interview: Dan Lynch & Fab Scherschel

We speak to the men behind Linux Outlaws to find out how their long-distance project started, what they plan to do next and why they have to drive so fast.

**LV** Dan, Fab great to chat with you both. So first question: how did you guys meet?

**Dan Lynch:** We met on a social network called Jaiku back in early 2007. It was a Finnish project that became popular when Leo Laporte (of TWIT fame) joined. I heard about it on one of Leo's shows and decided to give it a try. I'd never really bothered with social networks at all before. Jaiku was like Twitter in that initial posts were limited to 140 characters but it also had comments on each post, which were not limited. What came out of that were these really in-depth long discussion threads.

I met Fab and a whole bunch of great people on there. Many of them are still among my closest friends. I began podcasting with my Jaiku friends in early 2007 and we had a great time. Fab was a guest host or panel member on those shows sometimes. We both used Linux and were enthusiastic about it. It was his idea to start a podcast about Linux and he asked me if I wanted to join him. The rest as they say is history. Jaiku was bought by Google and ultimately shut down by the way. Some of the technology became Google+ and the staff were mostly absorbed into Google's team.

**Fabian Scherschel:** Pretty much what Dan said, although I don't think Jaiku was my first social network. I still have fond memories of the community there. I got on that early podcast by imitating an angry John C Dvorak, by the way. Fun times!

**LV** Linux Outlaws rose to popularity at a time when LUG Radio was winding down. As we understand it the LUGRadio hosts gave Linux Outlaws quite a recommendation?

**DL:** They did encourage people to give us a try and that was really helpful. I began listening to LUGRadio around the end of 2006, so it had been going years before I heard it. When we started Linux Outlaws in Sept 2007 they were a big influence. Initially I'm sure they had no idea who we were – I mean, why would they? But at some point one of the hosts heard our show and mentioned it. That was great. We also came 2nd to LUGRadio in a podcast roundup in *Linux Format* magazine pretty early on, it must have been 2008 I think. I believe it may even have been Linux Voice's own Mike Saunders who wrote it, I forget now though [It was Mayank Sharma]. That boosted our popularity massively and we reached a big audience quite quickly. When LUGRadio decided to call it a day about a year after we began they were really kind in directing people our way, which we really appreciated.

**FS:** I must confess that I didn't listen to LUGRadio when we started. Dan would go on and on about it



Dan Lynch is based in Liverpool and records from his home studio.

and eventually I gave it a listen and became a fan. I think I pretty much must have listened to the whole back catalogue, too. As for podcasts who helped us out along the way, we must also mention Linux Reality. I wrote an email to Chess [Griffin] when that show ended and he mentioned LO subsequently. We got a pretty big influx of new listeners from that.

**LV** From all of the shows what has been your favourite moment?

**DL:** Just one? That's very hard to answer. Possibly recording our 100th episode in a car speeding down the Autobahn as we headed to Linux Tag in 2009. Fab was driving! I don't recommend podcasting and driving, unless you have a decent hands free of course. In this case I held the recorder as he drove so it was ok. We had a really fun discussion and people seemed to like that episode a lot.

**FS:** The recording on the Autobahn at 200km/h in my dad's Renault was definitely a high point. That was really fun! Other than that, I also immensely enjoyed the live show we did in Liverpool for episode 300 as well as taking the stage at the first OggCamp. Back then, I would still feel pretty nervous when facing down a crowd of hundreds.

**LV** What legacy will Linux Outlaws leave behind?

**DL:** God I don't know. I'd like to think we showed



Fabian Scherschel (L) and Dan Lynch (R) have been the hosts of Linux Outlaws for seven years.

people that anyone can do this and you really don't need to be clever or talented. I know LO listeners have started a lot of podcasts and gone off to do their own things down the years. I like to think we inspired that in some way. We never claimed to be experts in Linux or anything else really. As the show grew popular people began to expect some level of expertise from us but they missed the point. It was always meant to be just two average guys who use Linux talking about things they think are cool and sharing experiences. If others went out and did that too because of our example then great. Also OggCamp is a legacy I'm very proud of personally. It's not an LO invention and

of course we began the event with the good folks at Ubuntu UK. It's grown way beyond the show, but LO played a crucial part in starting the event. I want to see that continue long after we hang up the microphones. **FS:** I hope that there will be more podcasts to come for Dan and myself. Maybe not Linux or software-specific ones, but hopefully they will be fun and entertaining. I'm not ready to give up podcasting just yet. As for the Linux podcasting scene, I hope there will be always a show or two that includes presenters that say what they believe, no matter how unpopular it might be. That's the one thing that was always most important for me and it's the one unique aspect that independent podcasts add to the media landscape.

**LV** If you could have done one thing differently with Linux Outlaws, what would it be?

**DL:** That's hard to say. I definitely don't regret much to do with the show. This may be a rubbish answer but I don't think there is anything obvious I'd want to go back and change really.

**FS:** Actually, one of the things I've always been proud of with LO is that we've adapted throughout the show's run. We've tried not to cling to predefined templates and segments. We tried to change with the times. In that way, I think, we actually did things differently whenever we saw the need. This did get us in conflict with some of the listeners several times, but I do think we generally did a good job.

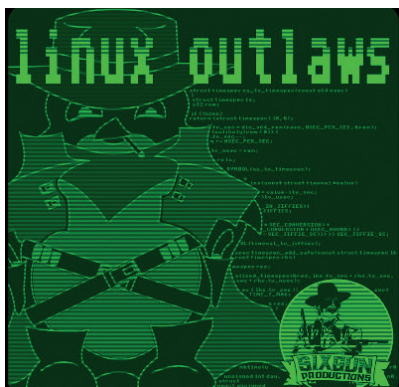


**Top** Fabian Scherschel is based in Germany and also records from a home studio that has grown along with the podcast.

**Bottom left** The Linux Outlaws album art has changed many times in the course of the podcast.

**Bottom right** Linux Outlaws were part of the team that created the popular Oggcamp unconferece.

PHOTO CREDIT: Tris Linnell





# EVEN MORE PODCASTS



## FLOSS Weekly

Hosted by Randal Schwartz and his rotating panel of presenters which includes Linux Outlaws' very own Dan Lynch. FLOSS Weekly is part of the larger TwiT (The Week in Tech) network that covers everything from Open Source technologies to other proprietary platforms. This podcast is of exceptional quality and by that we mean the content and knowledge of the hosts and the production quality of the finished product. The show often has special guests from the tech scene and they provide great insights into their projects.  
<http://twit.tv/show/floss-weekly>

## Ubuntu UK Podcast

Well known as the Radio 4 of Linux Podcasts, the Ubuntu UK podcast has a long tradition of quality and insightful content. Hosted by Alan Pope, Tony Whitmore, Laura Cowen and Mark Johnson, the Ubuntu UK podcast, as you may have already guessed, provides news and content from the Ubuntu community, but their scope is not just limited to Ubuntu and they feature content from the wider Open Source community. This podcast is well produced with a snappy (the team try to keep episodes to under 40 minutes) and well tested format.  
<http://podcast.ubuntu-uk.org>

## Raspi.today

A relative new kid on the block hosted by Russell Barnes, who used to be the editor of *Linux User and Developer* magazine. Russell's site is dedicated to the Raspberry Pi and his enthusiasm and passion is plain to see with lots of great content in his podcast. He has interviewed many different people from the Raspberry Pi community including 4tronix, PiBorg and Ben Everard, author of the best-selling *Learning Python with Raspberry Pi*. If you are into your Raspberry Pi, then this podcast is the one for you.  
[www.raspi.today](http://www.raspi.today)

## Linux Luddites

Linux Luddites is a podcast presented by Joe, Paddy and Jesse, who twice a month dip a toe into the Free Software waters, try new Linux software, then decide that they liked the old stuff better. The format of the show is similar to Linux Outlaws with a dash of LUG Radio. The news, reviews and feedback sections denote the pace and content of the show. If you like your Linux news with a dash of humour then the Luddites are happy to oblige, just don't be too bashful when the odd profanity pops out.  
<http://linuxluddites.com>

# LINUX LUDDITES

## The Linux Link Tech Show

TLLTS is a weekly podcast with a myriad of presenters that each delve into the latest news and issues in the Linux and Open Source communities. Presented in a laid back and relaxed manner TLLTS is a refined yet casual show that seeks to entertain and inform the listener. A great podcast to listen to on your commute to work. The Linux Link Tech Show is released under a Creative Commons Attribution, Share-Alike Non-Commercial licence, so you're free to mix it up, copy it and use it as you see fit as long as you fulfil the terms of the licence.  
[www.tlts.org](http://www.tlts.org)

## MintCast

In a similar manner to the Ubuntu UK podcast, the MintCast is a podcast created by and for the many Linux Mint users, of whom there are a growing number. Weighing in at around two hours long this is an informative show presented by Rob, Scott and James who are all keen Mint users, along with guest hosts from the Mint community.  
<http://mintcast.org>



## Going Linux

Going Linux is currently hosted by Larry Bushey, the creator of the podcast, and technology advocate Bill Smith, and has in the past been presented by Tom Chadoir and Serge Rey. This is a quality podcast that provides great content to suit all levels of Linux users, along with reviews on all the latest free software and general news from Linux communities. The show provides a fresh and innovative approach to delivering the latest news thanks to their extended team of presenters and diverse content.  
<http://goinglinux.com>

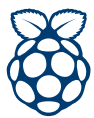
## Everyday Linux

The community that surrounds Linux is what sustains and nurtures Linux and this podcast is there to help illustrate this facet of the Linux eco-system.  
<http://elementopie.com/everyday-linux-episodes>





# PICADEMY



LEARN TO TEACH



**Les Pounder** travels to Pi Towers to find out more about what the Raspberry Pi Foundation is doing to educate the educators.

Computing in schools has become a hot topic in the last few years, and teachers are keen to learn new skills to expand their knowledge of the subject. Typically this is achieved via self-learning, but when teachers need quick results they reach for training via established training providers. Continual Professional Development (CPD) is not new to the teaching world, and providers are looking to meet the needs of teachers who are eager to learn what the Raspberry Pi can do.

The Raspberry Pi Foundation has its own education team, and its champion of CPD is Carrie Anne Philbin, an ex teacher, author of *Adventures in Raspberry Pi*, and creator of a popular series of YouTube videos under the title "Geek Girl Diaries".

Linux Voice was privileged to be given behind-the-scenes access to Picademy, the free professional

development event provided by the Raspberry Pi Foundation, and the team that make it happen. We asked them all about the genesis of Picademy and the future of CPD training for UK teachers.

## Genesis

When Michael Gove, the former Secretary of State for Education, announced that there were to be changes to the ICT curriculum many teachers around the UK felt that they were ill-prepared to teach the new computing curriculum and searched for specialist

CPD training to help bridge their skills gap. Around the time of this announcement the Raspberry Pi was on sale with a mission to help children learn more about

**"Teachers are keen to learn new skills and to expand their knowledge of the Raspberry Pi."**

Computing via creative means. What was missing was a support system of lessons and ideas for teachers to work with, and so the Raspberry Pi

Foundation created its education team with the goal of providing support material for teachers to use with their Pis. With the hiring of Carrie Anne Philbin, the Foundation education team created their own CPD under the name of “The Raspberry Pi Academy for Teachers” informally known as Picademy. The first Picademy took place at the Raspberry Pi HQ in April 2014 and 24 teachers from around the UK took their place as Raspberry Pi Certified Educators (RCE). Since April, there have been four more PiCademies, each training 24 more teachers who are spreading their knowledge to other schools around the UK.

### Structure of Picademy

Typically, CPD is structured around a single day and is lead by a single trainer who is an expert in the subject. Picademy is a little different in that it is split over two days and is lead by a team of specialists from across the Raspberry Pi community.

On day one, the 24 teachers from around the UK arrived at Pi Towers in Cambridge for an early start to a full day of Pi-based training. Each of the teachers had previously gone through a rigorous selection process that involves a written application to find out about their skills and aspirations for the Pi, which is then followed up with a video application. From the hundreds of applications the chosen 24 make it to Cambridge and are put into teams that encapsulate the essence of Pi, with team names such as GPIO, Scratch, Python and Minecraft becoming precursors to the content of the next two days.

The four teams were then introduced to the Picademy team, consisting of Carrie Anne Philbin, Sam Aaron, Ben Nuttall and Dave Honess, and to the community members, which on this occasion included James Robinson, Martin O’Hanlon, James Hughes and your humble narrator. Over the course of the two days these people were on hand to guide each of the teams through their learning.

The first training session was with James Hughes, who led a quick guide to Linux and the command line, both of which are essential skills to learn when hacking the Raspberry Pi.

The second training session is with me, making my début with my favourite Pi add-on board, the Pibrella.



Using Pibrella with a little Python 3 we all made our own version of the old quiz show game “Wheel of Fortune”, which uses a wheel to randomly select a question or prize.

Session two, led by Clive Beale, introduced the class to the excellent Scratch GPIO application maintained by Simon Walters. Using Scratch GPIO, the class quickly built their own traffic lights using breadboards, LEDs and wires.

### Hacking the camera module

Session three was an introduction to the Raspberry Pi camera unit and the corresponding Python module, led by Ben Nuttall. This session was a swift yet succinct introduction to the camera and how to use it with Python 3. Ben instructed the class on how to install and use the hardware using the **raspistill** command and the Python library using a hardware button to trigger the camera.

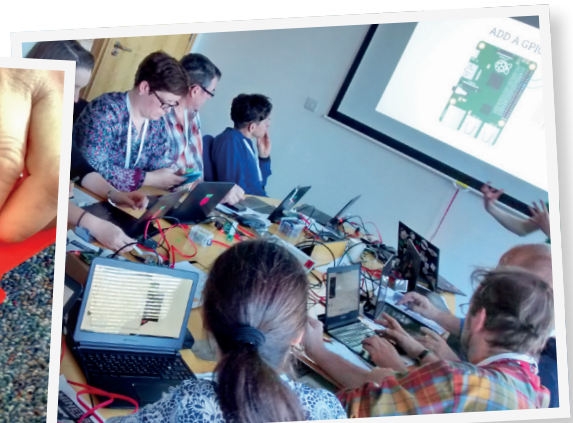
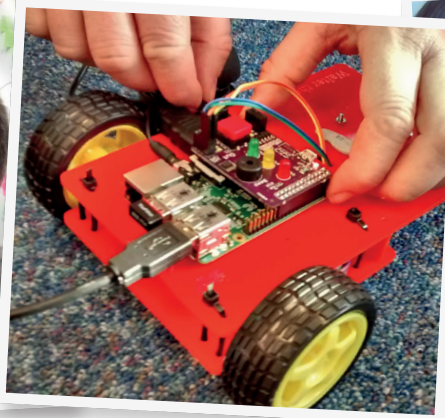
Session five was led by Martin O’Hanlon, and demonstrated how *Minecraft* can be used to teach Python in a fun and inventive manner, starting with teleporting and the x y z positioning system, and ending with the creation of diamond walkways that enable the player to walk in thin air. Martin, along with David Whale, has written a book in a similar fashion to Carrie Anne’s successful *Adventures in Raspberry Pi*, wonderfully called *Adventures in Minecraft*.

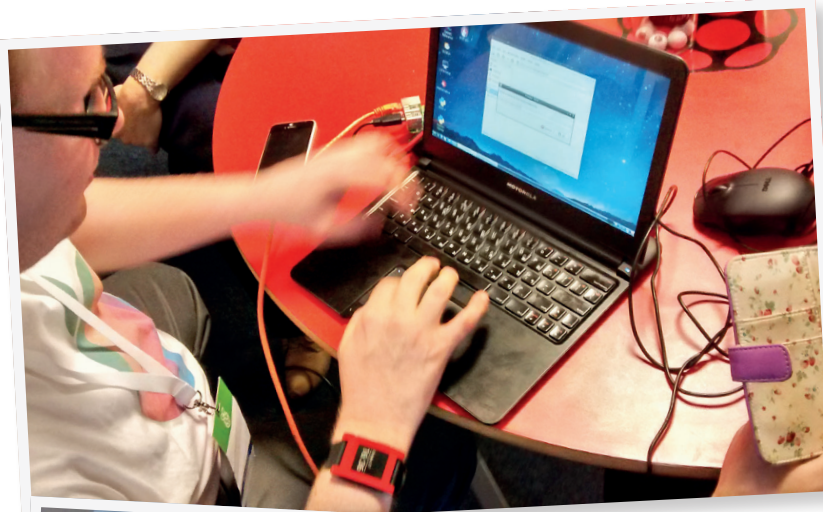
Picademy gives teachers the time and resources to try out their Pi-based projects in a supportive network.

**Below left** At the end of day 1 the teachers added their projects to a large wall of projects from previous Picademies.

**Centre** Robots are a great way to demonstrate coding concepts in an exciting manner. Why show off a boring loop with “Hello World” when you can have a robot drive around?

**Below right** Constantly changing the location and trainer for sessions helps the class to retain their focus along with a notebook full of notes and sketches.





**Top** Picademy uses Motorola lapdocks to enable teachers to be fully mobile with their Raspberry Pi projects. **Above** At the start of day 2 there were speakers from the Foundation and the Community. Here we see the Foundation's artist in residence, Rachel Rayns talking about art and the Raspberry Pi.

Sonic Pi, led by Sam Aaron, was session six, and this proved to be the most popular session of the day. Sonic Pi, the popular music composition/programming suite, is an exceptionally powerful tool in the hands of practitioners such as Sam. After demonstrating a series of ambitious projects, Sam let

the class compose their own tunes using samples and programming logic.

The last session of the day was a double feature, first led by James Robinson, who talked about integrating the

Raspberry Pi into the classroom. In the second half of the session Dave Honess demonstrated a classroom solution known as LTSP created by Andrew Mulholland that enables the Raspberry Pi to be used in a similar fashion to a thin client.

With all of the sessions complete for day one, the Picademy teachers were issued a challenge – in order to complete their Picademy training they were asked

**“The teams at Picademy get all of the toys they need to build any type of project.”**

to think of a project and complete it in the second day of Picademy.

Day two starts bright and early with a few presentations from key members of the Raspberry Pi team and community. Our first speaker was Rachel Rayns, an artist who works for the Raspberry Pi Foundation to further the use of the Raspberry Pi in creative and artistic projects. Rachel talked about her journey from being a traditional artist to using digital media and tools from the maker community. Our next speaker was Sam Aaron, the main developer of Sonic Pi, who works at a supersonic pace to improve the project, which is clearly visible in the changes made between version 1 and 2 of the application. Sam talks about how coding and music are interlinked with similar concepts that complement each other. Sam also demonstrated how live coding with Sonic Pi can be used for creative DJ sessions, something that will add an extra incentive for musically minded programmers. The last speaker is Matthew Manning aka Raspberry Pi IV Beginners, and his talk focused on the various communities that are present in the Raspberry Pi world.

### Hands-on

With the talks over and the teams eager to get hacking on their projects, they set to work creating their Pi-powered inventions.

The Raspberry Pi Foundation provide lots of equipment and access to the engineers behind the Raspberry Pi, so each of the teams get all of the toys that they need to build any type of project. At the October Picademy the theme was quite clearly Halloween, and with projects such as a robotic mobile disco that danced to Michael Jackson's *Thriller* and a Tweeting "Ghost Catcher" this is clearly evident. During the course of the day, the teachers hacked their projects into life ready for a show and tell at the end of the day, where they met Lance Howarth, the CEO of the Raspberry Pi Foundation's charitable activities, who was on hand to issue the certification for each of the team members who are now Raspberry Pi Certified Educators.

The class graduate and network amongst themselves, forging new branches to the education network via social media and traditional networking channels. The ideas created and friendships made here will go on to help others around the UK to work with the Raspberry Pi.

### How can you apply for Picademy?

Teachers from around the UK are welcome to apply for Picademy. Over the course of two days you will learn more about the Raspberry Pi and how it can be integrated into all aspects of your classes. You don't need to be an expert in the Raspberry Pi, as full guidance is provided; just arrive with an open mind ready for lots of fun and inventions. To apply for Picademy, head over to [www.raspberrypi.org/Picademy](http://www.raspberrypi.org/Picademy) where full application details can be found along with a short video introducing the Picademy training.

# Interview: Carrie Anne Philbin

During Picademy Linux Voice had the chance to talk to Carrie Anne Philbin, the lead for CPD training in the Raspberry Pi Foundation.

**LV** Hi Carrie Anne, thanks for talking to us. What is Picademy all about?

**Carrie Ann Philbin:** The Raspberry Pi Academy for Teachers, or Picademy, is a continued professional development programme for any practising teacher around the world, from any subject specialism. The two-day course leads to certification as attendees become Raspberry Pi Certified Educators and join an online community to share knowledge and good practice. The course is completely free to attend and takes place at our headquarters in Cambridge, UK. Every cohort includes a mix of primary and secondary school teachers in equal measures, as well as experienced computing teachers and those new to the subject, to share ideas.

**LV** There are many different providers for CPD but not many that specialise in the Raspberry Pi. How did the idea for Picademy come about?

**CAP:** With the introduction of the new computing curriculum in England and with many of the Google Raspberry Pis being distributed to schools through the Hour of Code Competition in 2013, we found that Raspberry Pi was being used in more formal learning settings than it had been previously. In January 2014 I attended BETT, the largest educational technology show in the UK, with the education team and found that teachers were no longer asking "Why should I use Raspberry Pi in my classroom?" but instead asking "When will you be running training courses?" I returned to work determined to create a training programme for teachers that would be inspiring, fun, creative and worth every second. So Picademy was born.

**LV** Why is CPD so important for teachers and the future of our new computing education system?

**CAP:** The changes to the curriculum are often misrepresented by the media as a "coding" curriculum, which has led to confusion and a lack of confidence from some teachers. There is also a skills gap for many teachers. The Computing At School organisation is doing a fantastic job in dispelling these myths and providing a

platform for teachers to talk. They have also developed CAS Master Teachers in order to train other teachers and share their best classroom practice.

**LV** So how is Picademy different to traditional CPD?

**CAP:** What we are doing at Raspberry Pi is different in that we are not training teachers to teach the new curriculum, but instead to see computing as cross curricular and a subject that underpins many others like Music, Art, Science, and Design Technology.

Only 44.9% of secondary school ICT teachers have a post A-level qualification relevant to ICT and the overwhelming majority of primary school teachers do not have a computing background. A recent survey found that 60% of teachers did not feel confident delivering the new curriculum.

So far, the government has provided £3.5 million for CPD, which is equivalent to £175 per school. By comparison, Jersey is investing around £5,750 per school to make a similar step change to computing. The sum also compares poorly to recent provision for CPD for teachers in maths, physics and global issues.

**LV** Education is an important part of the Foundation's mission. How can Picademy be expanded to take it to more people across the United Kingdom, and possibly the world?

**CAP:** That's the million dollar question! We have a very small education team at the foundation and have lots of projects going on that take up a lot of our time. We are very lucky to have members of our community like Sam Aaron, Martin O'Hanlon, James Hughes, Matthew Manning and Les Pounder to give up their time to come and help us with Picademy currently.

I'm looking to create more documentation for our current RCEs (Raspberry Pi Certified Educators) so that they can train others in their area, perhaps with the support of their local Raspberry Jam, to spread knowledge.

**LV** Is Picademy branching out and heading on tour?

**CAP:** In January 2015 we are moving Picademy to Wales, and having our first ever



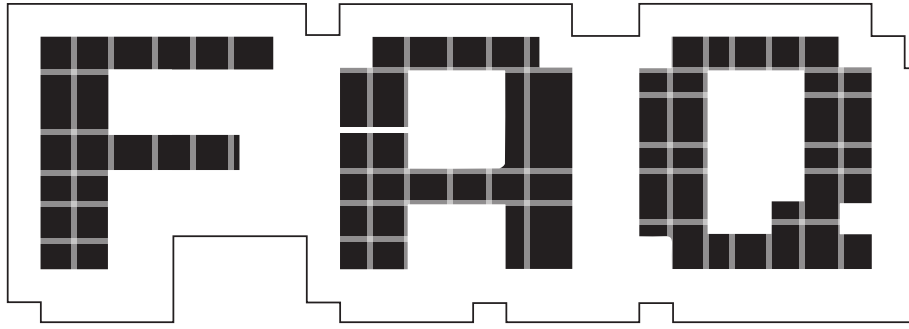
'on the road' event at the Sony factory in Pencoed where Raspberry Pis are manufactured. It's not out of the realms of possibility that we might visit other places in the UK and across the globe moving forward, but let's see how Wales goes first! There is something special about coming to Pi Towers for two days that I don't want to lose from Picademy.

**LV** What has been the response to Picademy from the Pi community?

**CAP:** The community has been hugely supportive. The MagPi magazine wrote an article about the CPD early on and offered us free printed issues of their magazines for swag bags. The Pi Hut, Pimoroni, Cyntech and ModMyPi have all donated swag for the teachers and the infamous prize box. Individuals like Les Pounder, Martin O'Hanlon, Sam Aaron, Alex Eames, Matthew Manning, Alan O'Donohoe, James Hughes, James Robinson and many more have helped to support the event through workshops, talks or videos.

**LV** If there were any changes that you could make, what would they be?

**CAP:** I'd like to be able to reach more teachers who lack confidence right now or who feel unsupported in their school. I'd like to show them that there is a great community out there ready, willing and able to help them. I'd also like to see our RCEs run more of their own training events in their region, and sharing resources with us to publish on our website. But it is early days for Picademy having only just completed our 5th event. There's loads more to come. **LV**



# ASM.JS

Bringing near-native performance to cross-platform web apps.

**BEN EVERARD**

**Q** ASM usually means assembly, and .js usually means a JavaScript library, but what on earth are those two things doing together?

**A** The idea behind asm.js is to remove everything from JavaScript that doesn't run quickly. The result is a very strict subset of JavaScript that isn't as nice to program in, but does run much faster.

Although asm.js is an interpreted language that you could program in, programmers will usually write code in another language, and then compile that language to asm.js. In other words, asm.js is designed to be a little like assembly language, but it's actually JavaScript.

**Q** So it's just another JavaScript engine?

**A** No. In essence, asm.js isn't anything other than a specification of a subset of JavaScript.

**“The idea of asm.js is to minimise performance concerns as much as possible.”**

Any JavaScript code that only uses this subset can be said to use asm.js. However, since it's valid JavaScript, it will run on any JavaScript engine. As there are JavaScript engines for almost every computing platform built in the past decade, compiling code to asm.js means it should be very cross-platform.

**Q** Wait a minute: compiling code to an intermediate language so it can run on web browsers... this sounds a lot like Java! Do we really need another option for this?

**A** There is a certain similarity in the concepts behind Java and asm.js. However, they're solutions designed for different ages. Java applets are placed on a page and given a certain area that they are allowed to interact with. In other words, they were a single item on a larger page. This means that, while they have some uses, they have never really been suitable for full-on web apps.

JavaScript (and by extension asm.js) can interact with the entire web page. It can add, remove and manipulate items in the HTML in an almost endless series of ways. In other words, JavaScript can be used to control the entire web page. This makes it a much better option for modern web apps. What's more, almost all modern web

browsers support JavaScript out of the box without the need for plugins. This means that you don't need to ask users to download anything in order to access your site.

**Q** Why bother using a subset of JavaScript, when the full language is already supported on multiple browsers on most modern operating systems?

**A** The advantage of limiting the available options is speed. Using *Firefox* (which is the browser that handles asm.js best), the JavaScript engine is able to detect when a particular script is written in asm.js, and will optimise itself accordingly. This gives it the advantage of running everywhere (because it is a subset of JavaScript), yet being able to run very quickly when the JavaScript engine supports it.

**Q** So it already works everywhere? Does that mean I don't need to change my browser?

**A** It's not essential, but like we said before, *Firefox* optimises itself when it detects asm.js code. This means that it will run much faster on recent versions of *Firefox* than it will on other browsers. Obviously, speed isn't always essential, but when it is, you're



better off using *Firefox* for asm.js web apps. *Chrome*, after a slow start, is catching up. Other browsers are likely to perform worse at the moment, but may well see improvements in time.

**Q** You mentioned earlier that programmers write in other languages, and then compile to JavaScript. What languages can you program asm.js in?

**A** So far, most of the work has focused on C and C++. The support for both of these is provided through the *Emscripten* source-to-source compiler. Since a large proportion of computer games are written in these languages, asm.js has been used to port games to the web (using WebGL for graphics). Perhaps the most famous asm.js project is the port of the *Unity* games engine (for example, *Dead Trigger 2* – <http://beta.unity3d.com/jonas/DT2> and *AngryBots* – <http://beta.unity3d.com/jonas/AngryBots>).

However, support for other languages is coming. Python has some support (via *pypy.js*), and the Lua VM can be built through *Emscripten*, but neither of these are really at the level of the C and C++ versions yet.

**Q** Why not just skip this step and write in JavaScript?

**A** There are a few reasons! There's obviously a lot of legacy code that exists already in C and C++, so why bother re-writing it in JavaScript if you can just compile it? You might want a single codebase that can compile to both native and browser. Also, compiled asm.js code tends to be quite a bit faster than hand-written JavaScript because it takes advantage of a whole host of optimisations.

**Q** It looks to me a little like most of the advantages of asm.js happen when you take something that is normally a native app and convert it into a web app. Isn't this a bad idea? I mean, wouldn't it be better just to compile the C or C++ to native code?

**A** Whether or not it's a good idea depends on many things, but basically it's always a trade-off. Putting software in web apps can make them easier to access across a range of



If something as computationally intense as an FPS game can run in asm.js, then most other software should have no problem.

devices, but on the other hand, there are privacy and security concerns, and performance can be a problem. The idea of asm.js is to minimise the performance concerns as much as possible. In fact, benchmarks show that code compiled to asm.js can run at about twice the speed of the same code compiled natively. This might sound like quite a big slowdown, but it doesn't mean that programs will run at half the speed, because only a small proportion of most software is actually waiting for a bit of code to run. Most of the time the computer's waiting for user input, or for some data to be retrieved from the disk, or (in the case of games) a 3D scene to render on the graphics card. This means that plenty of software will appear to run at the same speed when using asm.js as when compiled to native code. This doesn't change the trade-off between access on multiple devices and security, which will be highly dependent on the application and who's hosting it.

We should also point out here that although JavaScript is usually used for web apps, you don't have to use it this way. There's nothing to stop you using asm.js to create software that doesn't rely on the network, and just uses the JavaScript engine to provide portability. If asm.js takes off, we're likely to see more and more software doing this. In fact, it's already possible to compile some *Qt* software to asm.js. There are some examples at <http://vps2.etotheipiusone.com:30176/redmine/projects/emscripten-qt/wiki/Demos>.

**Q** This all sounds wonderful. How can I compile my C and C++ programs to asm.js?

Software can be compiled to asm.js using *Emscripten*. Use this in exactly the same way you would any other compiler. Asm.js is used when you set the optimisation flag to **-O1** or higher. This can output pure JavaScript or an HTML file that includes the JavaScript. See the tutorial at [http://kripken.github.io/emscripten-site/docs/getting\\_started/Tutorial.html](http://kripken.github.io/emscripten-site/docs/getting_started/Tutorial.html) for a useful look at how to get started.

**Q** What about stuff that JavaScript in the browser just can't do, like access the filesystem, and link to libraries.

**A** There's no way that asm.js can access the filesystem of a machine when running on a website – JavaScript is deliberately kept separate from the machine it's running on for security reasons. However, asm.js programs can access a virtual file system. This enables the developer to use the same C and C++ code that accesses files, but at the same time, still protect the host machine from any malicious asm.js code.

Libraries are another matter. By default, asm.js includes *libc*, *libc++* and *SDL*. If you want to work with other libraries, you could try compiling those libraries to asm.js, or re-implementing the features you need. There's some more details on this on the *Emscripten* FAQ: [http://kripken.github.io/emscripten-site/docs/getting\\_started/FAQ.html](http://kripken.github.io/emscripten-site/docs/getting_started/FAQ.html) 📄

# BRIAN BEHLENDORF

Graham Morrison geeks out about synthesizers with a kindred spirit, then remembers to ask some questions about free software.

**W**hat do the *Apache* web server, the EFF, Mozilla, the World Economic Forum and Obama's 2008 campaign have in common? The answer is Brian Behlendorf. He is one of the founding developers of **httpd**. He was the co-

founder of Collabnet, the company responsible for *Subversion*. He's been on the board at the Mozilla Foundation for over a decade, and joined the board at the Electronic Freedom Foundation in February 2013. He was a technology advisor to the 2008 Obama campaign,

and helped the Department of Health and Human Services develop open source solutions for electronic health records. He's served as CTO to the World Economic Forum, he's an entrepreneur, a fan of electronic music, and a true open source polymath.

**LV** **Larry Page said Google could save a 100,000 lives with access to big data, but is Google the right company to do this data mining? Can you see a way of doing this that respects people privacy, while still saving 100,000 lives?**

**Brian Behlendorf:** I think what disturbs people is not the sense of data being shared, but data being shared in ways they either can't quantify or can't control. And control of data is an awkward thing, because there's no physical law that allows me to take away something you know about me, nor should there be. Because you have as much right to data you've collected about me in a mutual transaction. When it comes your rights to sharing that data with others, that's where I think we can talk about appropriate rules or not. And so finding ways to actually get their consent, to share that kind of data or make people active participants in understanding where they can feed that...

**LV** **It's trust, isn't it?**

**BB:** It is, certainly. There's a project out there called the Respect Network, which is a coalition of a bunch of different companies (I think Swisscom is a part of this and a whole bunch of startups), to basically put

together a contractual network for sharing data that binds the participants into covenants with the end user. When you give data to a member of the Respect Network, you can grant them the right to share that with other groups that you talk to also within that network, but you also have the ability to ask them to remove that data, or to update it and have that update shared once with the rest of the members. So it's a way of starting to claw back a little bit on the consumer side, an understanding of how that data propagates. And then you can't share it outside of the Respect Network, right. That contractual relationship stops someone from being able to do things nefariously with your data, like share your credit card information with your health insurance provider so they know about all those trips to McDonalds.

**LV** **<laughter/>**

**BB:** That actually happens! It's not a hypothetical thing.

**LV** **Maybe we're too cynical, but that's the kind of thing we worry about. Even in anonymised data, there's still a shadow of yourself, and of the population.**

**BB:** De-anonymisation is getting better and better all the time.

**LV** **But where does the trust come into de-anonymisation?**

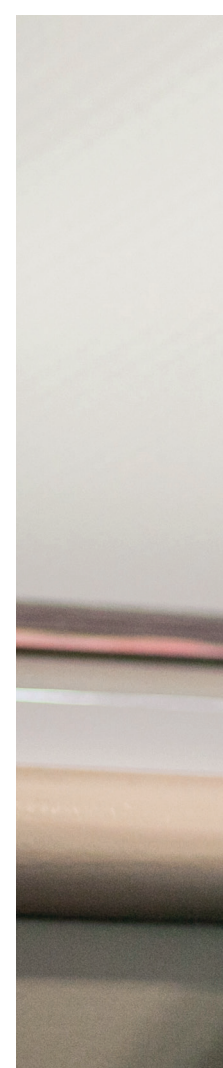
**BB:** We shouldn't pretend that you can take data about somebody's very intimate details, like places they've been, especially transit data. There's a study that showed you can uniquely identify individuals by having just four data points of their daily route.

**LV** **Wow.**

**BB:** When you think about it, the places that you work and the place that you live, how many other people take the same route to work within a couple of hundred feet to a couple of hundred feet? It's probably a very small number. If you gave me, here's anonymised transit data showing every trip inside London, I'd probably be able to go from here and here, who are the two people that live here, and then with one or two more points, discretely just get it down. So that's scary, right?

**LV** **We're right to be worried about it being in the wrong hands.**

**BB:** We're right to be cynical or question anyone who says "don't worry, we anonymise it first before we hand it on". At the same time, I do absolutely agree that better data informatics can lead to better health outcomes. But I think that only works if you've got the active





**“I’m constantly looking for folks doing interesting things who we could help put thruster rockets on and go into orbit with.”**

participation of the end users and they understand what they’re doing. I think this is what drove a lot of people nuclear with the Facebook timeline thing, is the sense that there was no consent around that, no awareness of it. Even if we know that our timeline gets played with, OK I can understand being played with to click on an ad, I’d be more likely to click on something, but played with to be more depressed or happier about the world around me, that’s pretty F’d up.

**LV** Or sicker or healthier...

**BB:** Right. So I think we have a few more iterations of this where we are going to discover I think what ways to quantify that deeper need we have.

**“What disturbs people is data being shared in ways that they can’t quantify or can’t control.”**

When we see that a friend told us about a disease they had, if you search on Google and suddenly you’re seeing ads for creams and lotions on other websites. Maybe it doesn’t happen with Google, maybe it’s other places, they’re kind of re-targeting in that kind of data utopia. What could happen with that?

I think that’s going to drive a demand for different technologies. In the same way that Apple stepped in at two different times, once in the 80s and once 10 years ago, and said there’s a need for better design when it comes to how we use computers, I think there’s an opportunity for another company to step in and say we’re going to provide technology that addresses this gap, this thing that’s in the zeitgeist that the existing leaders are saying no-one cares about, which is the desired form of trustworthy technology.

**LV** Surely the most important thing is implementing the system that could save 100,000

**people’s lives and we could do that right now?**

**BB:** I think we’re going to find upper limits to what it means to be human. We could get so precise with the data that we could tell you that for every Oreo that you eat, statistically speaking, that’s six minutes off your life. You could even come up with a wristband that would monitor everything you do and go ‘You’re going to walk out in this sun? Well that increases your risk of getting skin cancer and that’s another 10 minutes off your life’.

And pretty soon, if you have that perfect picture, insurance is no longer about creating a pool of roughly equal people to help balance out the extremes, instead it becomes about prepaying for medical expenses. And that’s the danger here, is that it’s not necessarily in the individual’s interest to preserve their life at all costs.

**LV** As a society, we can’t afford to go down that route.

**BB:** There's so much more basic information needed that it's almost not even worth thinking about right now. Like, the challenges people have just getting their medical histories transported from one place to another to another, especially people with chronic issues [such as] diabetes. I mean imagine, it would be so much better if you could create consistent, high-quality, longitudinal data pictures.

**LV** **Google's got the data.**

**BB:** Once they have those contact lenses, they may have perfect data. But I think we should help individuals with managing their health stream information, and through that, helping everything else. When I was at the Department of Health and Human Services, there was this recurring theme, which not everyone was a fan of, but it was individuals at the centre of their health information exchange, and there had been very few attempts at doing a really good personal health record system. [Microsoft's] HealthVault has probably been the best funded of them, Google gave up on it with Google Health; now it looks like they might get back into it, them and Apple, with this health metrics thing. But I think those will eventually come back around to helping the people who are trying to maintain their health, a mix of the exercise, doctors' reports and labs and that sort of thing.

**LV** **Can you tell us about your role on the board of the EFF, Mozilla and Benetech?**

**BB:** Sure. They're three non-profits that are pretty different in terms of how they go about implementing change in the world. It's largely about oversight, direction-setting and trying to bring in other individuals with interesting viewpoints. But trying to understand what's at the core of these three – as well as making sure that we tackle issues like a CEO not working out or should we shift the mission to tackle that – that's what we do.

The EFF's background is as an activist organisation. They just did a major release of something called Privacy Badger, which is for blocking cookies and things like that. It has seen a major expansion in the public attention being focused on the EFF because of the Snowden leaks. And there's recognition that things feel broken in a space at a much deeper level than just policy. That it's something about how we relate to governments and other organisations with the privacy of our data and the systems that we use. Part of the feeling is in tools, but part of it might also be in the way we relate to these organisations and the liberties we've allowed organisations like the NSA to take. And so in addition to writing software, we sue the government. We rally public attention around certain key

issues, we try to shift legislation here and there, and make people understand when there's a vote going up on an issue they care about. A lot of it though is also education, helping people understand what these issues are, or helping train journalists and people fighting for different particular points of view in different countries. Like, explaining to them what the laws are in these areas or what are the tools you can use to communicate securely as well. So it's a pretty diverse organisation, the EFF.

Mozilla is very different. Mozilla's

**“The fight for the open web used to mean fighting for HTML 5, JavaScript and CSS.”**

main thrust is the fight for the open web, and they do it by building consumer products that people love. And we've had challenges as you would expect any 14–15 year-old-organisation to have. The organisation is about 11 years old, but the project started even four years before that in '98 as the open sourcing of the *Netscape* browser. There's now about 750 people working for Mozilla. We have one major revenue stream, and we're looking for others.

**LV** **By 'one major revenue stream', do you mean Google?**



**Brian Behlendorf is potential father of the term 'Intelligent Dance Music' as he ran the famous early 90s mailing list.**

**BB:** Well it's not even that it's just one vendor, it's that we're dependent on one particular way of doing things. I have to be clear here, I'm on the board of the foundation, the foundation owns the corporation, the corporation is the one that builds the products and gets them out. The foundation licenses the trademark to the corporation, so it brings some money in that way, and we have some other investments and things. So the foundation does a lot of public benefit kinds of projects; the Webmaker project, Popcorn, that sort of thing, and fund a lot of open web, education projects. The corporation is the one that builds the product and we have to maintain this distinction because the [tax authorities in the USA] look very differently at non-profits and for-profits. So I can't tell the corporation what to build, but we can talk about this fight for the open web. It used to mean fighting for HTML 5, JavaScript and CSS, and we won that war. We were not only a faster browser, we've shown the world that these technologies are a better way to build web apps and websites, and why building a site for one browser or the other is lame.

And then the world took a tremendous step backwards and got app crazy. We as technologists out there wondered why would you want to build platform-specific apps when you have the web, and what we kind of got schooled on by Apple and others was this idea of local applications that could deal with local data that could deal in disconnected environments, that one could procure in an app store and pay real money for, generating a revenue stream for people, was interesting.

**LV** **Apple stumbled on that idea though. Their original idea on the iPhone was to have web apps.**

**BB:** It's funny how, for many of us, our biggest money makers come from happy accidents. But they took a step back, and now we do have platforms like Apache Cordova, which allow for some degree of portable development. But I think what really became clear, even five years ago, was the sense that the fight for the open web was no longer about a browser and about the presentation language, it was also about payments and also about where you store your metadata.



**"Enterprises are spending a lot of money on cyber security mitigation but no-one's talking about cyber security insurance."**

These are all things that are becoming core parts of the operating system. So the web standards needed to be updated to be able to do things like trigger the camera on your device to take a photo, which wasn't in HTML before, so rightfully that's one thing that apps had on us, so we had to come up with standards for that. But then we needed to look at not just getting Firefox to run on Android or getting it to run on iPhone, which Apple wouldn't let us do because it's a closed platform.

**LV** **It's a great app on Android.**

**BB:** So on Android we're able to do it, but even there it's not enough. Even there it's clear that we need to be so much more deeply integrated with the rest of what people expect from their phones. So that was the genesis of boot to Gecko and now Firefox OS. (Firefox OS phones are now available for sale in 15 different countries, by the way). The majority of R&D software development effort at the corporation is now focused on Firefox OS and making that work, on mobile in general I'd say, but Firefox OS is a huge part of that. It's a huge deal for us and I think it's not just phones, it'll be tablets and other things you'll see.

But thinking about fighting for the open web, it's funny how the Snowden stuff comes up again. It's almost like the battleground has shifted to talking about helping people and how their communications can be secured, and not entrenched with any one vendor.

**LV** **And you could argue that's something you've done your whole career.**

**BB:** It's a fight that I've helped with at different times and I'm still very happy to be working on that at Mozilla. And at Mithril too, hopefully. I mean, if a company comes along that is able to be an interesting part of this fight, that'd be a huge thing. The opportunity here is really big for companies, and let me be plural about that, to step in and look at providing an extra layer of security or an extra guarantee, or maybe even capturing this moment in the zeitgeist, where there is an unfulfilled demand by consumers, sometimes even not expressed directly by consumers, to understand how to trust technology. Even enterprises have this challenge too. So I'm constantly out there looking for folks doing interesting things here who we could help put thruster rockets on and go to orbit with. **LV**

Write once,  
deploy everywhere.



ubuntu<sup>®</sup>

**BUY LINUXVOICE MUGS AND T-SHIRTS!**



[shop.linuxvoice.com](http://shop.linuxvoice.com)

# LINUX VOICE REVIEWS



**Andrew Gregory**

An internet of things toaster would be just the ticket, says our hungry deputy editor.

Software freedom is important; we know that. But to most people that's only an abstract statement. Yes, we can study, modify and share the source code to *Emacs* or *GCC*, but as far as most people are concerned these may be hieroglyphics. Who cares? That stuff is for weirdos.

When the internet of things arrives though, all this will change. If your door lock or central heating software upgrades to a new version and breaks, all hell will break loose. If there's an iOS-style built in obsolescence that means your central heating will only get security updates if you buy new radiators, there's going to be a breakdown in how the world works.

## Free is cheaper

The main reason an internet-of-things company should use it is price. If you can make the source code open you cut your maintenance costs at a stroke, while at the same time reducing your liability if your smart car decides to drive you into a concrete pillar that isn't in its sat nav. If closed software does that, there's an obvious chain of liability, if free software had the same bug, it would have been fixed last week. Consumers expect their computers to be rubbish; when software gets to the real world, they'll demand a lot more.

[andrew@linuxvoice.com](mailto:andrew@linuxvoice.com)

The latest software and hardware for your Linux box, reviewed and rated by the most experienced writers in the business

## On test this issue...

48



### Entroware Proteus

Mike Saunders wants decent battery life, a 13-inch screen and a nice keyboard. Oh, and Linux pre-installed would also be nice. That's why he liked this laptop so much.

50



### Congratulations!

This browser is configured to use Tor. You are now free to browse the Internet anonymously. [Test Tor Network Settings](#)

Search securely with StartPage

#### What Next?

Tor is NOT all you need to browse anonymously! You may need to change some of your browsing habits to ensure your identity stays safe.

[Tips On Staying Anonymous >](#)

#### You Can Help!

There are many ways you can help make the Tor Network faster and stronger:

- Run a Tor Relay Node >
- Volunteer Your Services >
- Make a Donation >

The Tor Project is a US 501(c)(3) non-profit dedicated to the research, development, and education of online anonymity and privacy. [Learn more about The Tor Project >](#)

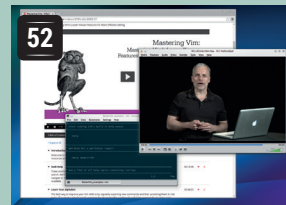
### The Tor Browser

Our friends in Cheltenham don't have a clue what Ben Everard is viewing now he's using the Tor Browser. Well here's the secret – it's Ed Sheeran videos.



### Digikam 4.5.0

Our senior vice president in charge of taking pictures Graham Morrison examines Linux's leading photo album. Better than *Kalburn!*



### Mastering Vim

Vim user Graham Morrison didn't even know how little he knew until this instructional video expanded his powers.



### Firefox DE

Fresh from writing an addon for *Firefox* on page 100, Ben Everard needs a version of the popular browser for developers.

## BOOKS AND GROUP TEST

Many people spend most of their time at a computer staring straight into the web browser – whether that's Twitter, web mail, Facebook or the latest Ed Sheeran videos on YouTube. So it makes sense to pick the right tool for the job. You can probably guess the two leading contenders, but there's something for everyone, and each application we tested has something unique going for it. Meanwhile in books, there's the usual mix of subjects – including *Data Science for Business*, which makes us wish that we'd stuck with that maths A-level.



# Entroware Proteus

A new-ish company sources ideas from the web to make this Linux-bundled laptop. Mike Saunders checks it out.

## DATA

**Web**  
www.entroware.com  
**Manufacturer**  
Entroware/Clevo  
**Price**  
From £649 (for below specs: £754)

## SPECIFICATIONS

**CPU** Intel i5-4210M @ 2.60GHz  
**RAM** 8GB DDR3 1600 MHz  
**Storage** 120GB Samsung 840 EVO SSD  
**Graphics** Nvidia GeForce GTX 860M  
**Display** 13.3-inch Matte IPS LED, 1920x1080  
**Webcam** 2.0MP  
**Battery** 5600 mAh, 62.16Wh  
**Size (MM):** 32H x 330W x 228D  
**Weight** 2.04kg

Linux and laptops don't always make for the happiest of bedfellows. Custom hardware, coupled with the reluctance of manufacturers to share driver information, means that Linux support ranges from pretty good (for example, on older Thinkpads) to utterly terrible. If you're running Linux on your laptop, you've probably encountered some kind of issue, whether it's to do with battery life, suspend/hibernate or the webcam. A few machines work perfectly – but they're rare.

So when Entroware arrived on the scene as a vendor selling PCs and laptops with Linux pre-installed, we were naturally curious. And doubly so, because the small UK-based company didn't just throw out some generic machines and try to grab cash from desperate Linux users, but actually went out to the community to ask what people wanted. Entroware asked /r/linux on Reddit: What would you like to see from Linux computer retailers? There were almost 200 responses, and Entroware has taken them into account with its new flagship laptop, the Proteus.

We were lent a review unit for a couple of weeks, so have spent quite a bit of time with it. It's a boxy, angular machine, with black plastic on the underside, silvery plastic around the keyboard, and a slightly rubberised black top. The machine's original design manufacturer is Clevo (model W230SS), and it's sold by resellers in some markets as a gaming laptop.

We love the keyboard. It's a chiclet design, quiet and with chunky Enter and Backspace keys. (Our



This image shows the US keyboard layout; the UK model has satisfyingly large Enter and Backspace keys.

review unit was supplied with Windows logo keys, but Entroware aims to change those.) The keyboard is backlit, the keys themselves have a decent amount of travel, and there's barely any flex behind them – it feels very well made. The screen exhibits a little more flexing under pressure, but not to any scary degree.

But here we come to our first minor gripe: the trackpad. It's not bad, but it's just small. Sure, this

## The origins of Entroware

We caught up with Anthony Pich, co-founder of Entroware, to find out how his company came into being and what challenges it faces.

We caught up with Anthony Pich, co-founder of Entroware, to find out how his company came into being and what challenges it faces.

**LV** **How did Entroware get started?**  
**Anthony Pich:** The idea to start the company was made after I bought a new laptop preloaded with Windows. After immediately formatting it and installing Ubuntu, due to poor hardware support, I had to mess around with drivers and configuration files with most updates. When we looked at buying machines that

were Linux compatible out of the box, we found that the UK's offerings were expensive and not customisable. Even manufacturers overseas seem to be price gouging, so we decided to source the parts ourselves.

**LV** **How many people work there, and what do you do?**  
**AP:** As we are still very much in our infancy, we have two highly trained employees, whose responsibilities vary from manufacturing and quality control to marketing and accounting. For hardware

research and testing, we like to involve the whole team. We will be taking on more staff in the coming weeks to coincide with the launch of EU shipping.

**LV** **What's the biggest challenge in selling Linux-compatible laptops?**  
**AP:** Our biggest challenges so far have stemmed from hardware compatibility. With each product launch, we thoroughly research and stress test every individual component. This includes graphics cards and SSDs to less obvious components such as Bluetooth and card readers.



is only a 13-inch laptop, but we've seen bigger on other similarly sized machines, and when you've tried the giant football-field-esque trackpads on Apple's laptops, it's hard to go back.

### You're hot then you're cold

The front of the machine contains power/activity LEDs and SD card slot, while the right provides access to three USB 3.0 ports, HDMI, VGA, Ethernet, power and a Kensington lock. On the left is an extra USB 2 port, headphone and mic ports, and a grille for the fan. And this is the second of our gripes: the fan positioning. The cooling system sucks in air from underneath the laptop, and blows it out of the left-hand side. This means you always need to use it on a flat surface (so not directly on your lap or a bed, in case you block the vent underneath). And if you're a left-hander, with a mouse plugged in, you'll feel a steady stream of warm air on your hand.

On the upside, the machine stayed cool in our testing, even when stressing both CPU cores with maximum load, and the fans weren't especially noisy unless at absolute peak. Most importantly, the fans are barely audible when playing HD video – so you can enjoy movies without being distracted.

Onto the screen: it's a 13.3-inch IPS LED panel with 1920x1080 resolution. (An ultra high-res 3200x1800 display is available for an extra £50, but as HiDPI support on Linux is a mixed bag right now, we wouldn't recommend it unless you absolutely need it.) The contrast and horizontal viewing angles are good, although we noticed a tiny amount of light bleed from the bottom of the display when showing a full black screen – it's not annoying though.

Performance will depend on the chip you choose when configuring the machine: the £649 unit is equipped with a dual-core Intel i3 at 2.5GHz, but you can ramp it up to a quad-core i7 at 2.5GHz for an extra £95, or go full whack for a 2.9-GHz i7 for an extra £374. Similarly, the base unit is supplied with 4GB of RAM, but you can bump it up to 8GB for £30 or 16GB for £90. All models ship with Nvidia GeForce GTX 860M graphics with 2GB RAM.

But what's the battery life like? On our Core i5 review machine, with middle-level screen brightness and low keyboard backlight, we did some web browsing, played half an hour of *Minecraft*, and had an internet radio station running all the time (Flash, using around 7% CPU). With this setup we eked out just over four hours from the battery. If you're doing light browsing and typing work, you can expect to get over five hours. Suspend worked out of the box, taking five seconds to suspend and the same amount of time to resume.

### Upgrade-friendly

Excellently, the Proteus is easy to upgrade and maintain: just remove four cross-head screws from the panel on the underside of the machine, and you get access to the hard drive, RAM slots, Wi-Fi card



The top has a slightly rubberised feel, which looks great but needs the occasional wipe to remove fingerprints.

slot, and even the heatsink and fan. As Linux users we like to tinker with things, so we don't want sealed-up, locked-down machines that can't be opened without all sorts of hassle (cough, Apple). So plus points for Entroware here.

The laptop is bundled with Ubuntu 14.10; that's the OS that Entroware officially supports, but the company told us that it will try to assist users if they have problems on a different distro. Even if you don't want to run Ubuntu, at least you know that the hardware has been checked for Linux compatibility and everything should work, given the right configuration.

In all, the Proteus is a good all-round portable workstation. It packs plenty of power for the price – especially if you bump it up to 8GB of RAM and add an SSD – and it's also well built with a lovely keyboard. The dinky trackpad and underside fan vent fan are slight downers, but they won't be an problem for everyone. And even if those issues are making you think twice, there's still the matter of supporting Linux-friendly companies.

Would you rather buy a laptop with a slightly better cooling layout from a giant faceless company that doesn't give a hoot about Linux, and that forces you to pay the Windows tax? Or would you rather support a new Linux-focused company that's easy to talk to? We'd say the latter makes more sense. 🐧

**“The Proteus packs plenty of power for the price, is well built and has a lovely keyboard.”**

### LINUX VOICE VERDICT

A solid workhorse with decent specs and battery life, and a smashing keyboard, from an accessible, Linux-friendly company. A good purchase.



# Tor Browser 4.0.1

Now Theresa May can't discover Ben Everard's rampant online cat-video viewing habits. Your move, GCHQ.

## DATA

**Web**  
www.torproject.org  
**Developer**  
The Tor Project  
**Price**  
Free under various free software licences

The *Tor Browser* was originally funded by the US State Department as a way for non-technical people living in countries with few internet freedoms to access their online material, such as the Voice of America news site. The goal was simple: take the best censorship-resisting online anonymity software and democratise it so that it becomes accessible to everyone, not just geeks. That was nine years and four version numbers ago. Few people at the time realised just how popular it would become.

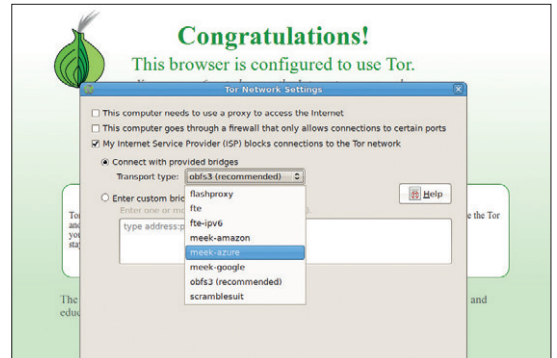
The *Tor Browser* comes as a Zip file that contains the executable. You just extract it and run. It will automatically connect to the *Tor* network, and start *Firefox* (now version 31 ESR). *Firefox* is customised to improve security. It defaults to the StartPage privacy-protecting search engine, and includes the NoScript addon. By default, the NoScript settings are quite lax, so you may want to investigate these if you're worried about attacks on your anonymity.

In future versions, *Tor Browser* will have a security slider that will enable you to easily change the security vs convenience settings (such as NoScript). There's a beta version of *Tor Browser 4.5* available now that includes this, but it's not yet considered stable.

### Problems change

The Free Software Foundation's HTTPS Everywhere addon is also installed. This forces the browser to use HTTPS whenever it's available even when following links that point to the non-SSL version of the page. So, for example, if you enter [www.wikipedia.org](http://www.wikipedia.org) into a normal version of Firefox, you'll go to the non-secure site, whereas if you enter it in the *Tor Browser*, you'll go to the SSL-secured version of the site. This is important because even through the *Tor Browser* stops people being able to link who you are to what you're browsing, unless you use SSL, it's still possible

When the *Tor Browser* starts, it will verify that *Tor* is working properly. If you don't see this page, then there's a problem



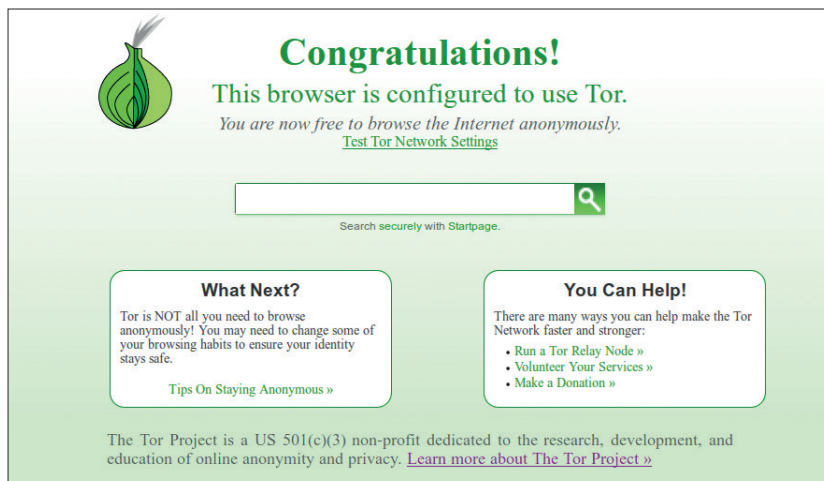
The new transports dramatically increase the difficulty for anyone trying to stopping people accessing *Tor*.

to see what data is being sent (which could well be used to identify who's sending it).

The challenges faced by the *Tor Browser* today are very different from when it launched. Originally, it was a way to access proxy content that was blocked; now the actual *Tor* network itself is blocked in some countries. This has meant that the *Tor* developers have had to find ways to access the network even when all the IP addresses of computers on the network are blocked.

Rather than try to come up with a single solution to this, *Tor* uses pluggable transport modules. These are methods of obfuscating the *Tor* communication so it's harder to block. The more pluggable transports there are, the more challenges for anyone trying to prevent people connecting to the network. These transports have been around for a while, but *Tor Browser 4* both makes them easier to use, and introduces some powerful options. The Meek pluggable transport (new in the *Tor Browser 4*) is believed to work out-of-the box in China, one of the countries that has had most success in blocking access to *Tor*. This transport diverts traffic through popular content delivery networks (CDNs) which means that if a government wants to block Meek, they have to block every site that uses these CDNs, and that includes a large proportion of the web. The idea is to make the level of collateral damage of blocking *Tor* too high for it to be feasible.

Regardless of whether you're trying to bypass censorship, or keep your private internet communication private, the *Tor Browser Bundle* should find a place in your network toolkit.



## LINUX VOICE VERDICT

Preventing snooping and bypassing censorship has never been easier.



# Digikam 4.5.0

Once a rival developer [of Kalbum, if you're interested], **Graham Morrison** imports the LV photo collection into the latest release.

Photos are becoming like dust. They gather and accumulate without anyone noticing. Over the course of a year, a collection can become unmanageable and often best left forgotten. Unless, that is, you lose some. Which is why applications that make your collection easy to store, easy to process and easy to share are more important now than ever before. And as social networks continue to dominate, the austerity of a desktop application makes a refreshing change, even when the export options let you share your creations as instantly as a Polaroid.

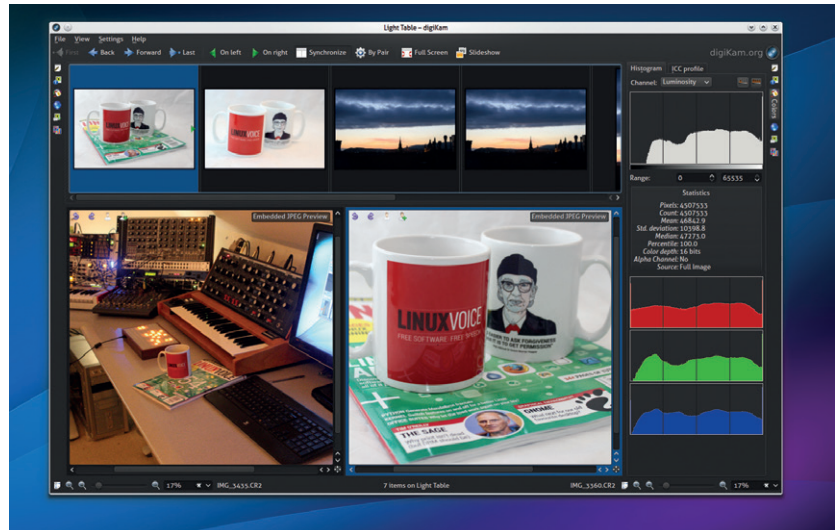
*Digikam* has been around for a long time. It predates iPhoto and Flickr. But it has also moved with the times and it's currently moving very quickly. Version 4 was released last May, and this major update follows exactly six months later. For the first time with *Digikam*, we had enough confidence to attempt importing RAW files directly from the Linux Voice Canon 600D. This worked excellently. The preview window appeared quickly and drew thumbnails within a few moments. It's a pity these aren't pre-cached for the files out of view, but we like the way you can click on a single image and skip through larger previews. The use of a Marble map view before importing images is also novel, but we'd like the ability to add keywords to images before import.

## Lazer Tag

Tags, colours, ratings and captions can be added when you get to the main application window, and it's these facilities that make management so straightforward. The main view is centred around another thumbnail view and two strips of panels that can be opened and closed to its left and right. These panels hide *Digikam's* powerhouse of features, from



If you include the plugins, there are probably more filters, effects and editing options in *Digikam* than *Gimp*.



duplicate image recognition using fingerprints, face detection, location mapping, over- and under-exposure marking and a great side-by-side light table view. Unlike many photo management apps, there's also a brilliant image editor that lets you do far more than tinker with colour balance and contrast, although we do miss the exposure, noise reduction, white/black adjustments and lens profiles of *AfterShot Pro* (a commercial alternative). The user is expected to out-source these functions to an image developer applications, but *Digikam* already does so much, we'd love to see them integrated into *Digikam*. There are 39 plugins, for example – we used Exoblend to create pseudo HDR images – so *Digikam* is already way more than a management tool.

There's been lots of bug fixing since the 4.0.0 release, and we didn't experience any stability problems with 4.5.0. The user interface suffers from over functionality, in the KDE sense, because there are too many windows, panels and options. And despite our having used *Digikam* on and off for almost 10 years, we still couldn't find a good way of seeing what each release brings. The link to bugs.kde.org doesn't help. But this is still a wonderful application. Remarkably powerful, flexible and capable of managing very complex collection. It could just do with a little pruning and rationalisation in its imminent transition to KDE 5. 🐧

We prefer a darker theme when working with photos, and like *Krita*, *Digikam* lets you change colours on the fly.

## DATA

**Web**  
www.digikam.org  
**Developer**  
KDE  
**Licence**  
GPLv2

## LINUX VOICE VERDICT

One of the best photo management applications for Linux, and even (shhh)... Windows.



# Damian Conway's Mastering Vim

After many years knowing only 'i' and ':wq' in Vim, Graham Morrison feels it's time to get some video learning.

## DATA

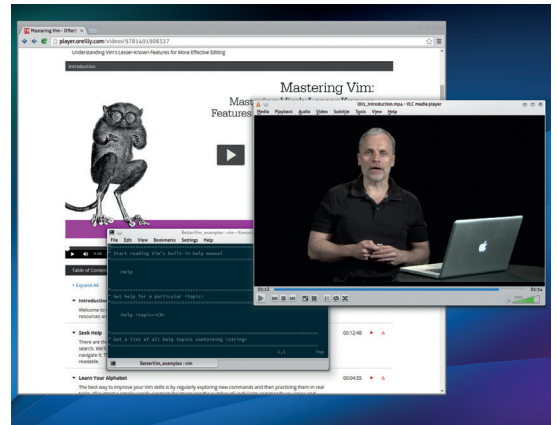
**Web**  
<http://shop.oreilly.com>  
**Developer**  
 O'Reilly/Damian Conway  
**Licence**  
 Proprietary

**D**amian Conway is a natural teacher. Actually, that's not true. We know he appears natural and relaxed and full of fragile wisdom. But this doesn't come naturally. It's the result of hard work. His 30-minute keynote presentation at QCon took over 100 hours of preparation. Likewise, we can't even begin to guess how long *Mastering Vim*, a video of three hours and 25 minutes, took to produce. We're tempted to say decades, as we've seen his dexterity with this humble text editor first hand. We've seen him meld presentations with live code demos, hack away at Perl and tell jokes, all with just a few chordal keystrokes, all at the same time. If there was going to be anyone to push us through the pain barrier of only remembering three *Vim* commands, it was going to be Damian Conway.

We've never tried 'Video Training' before, and we're rather distrustful of leaving books and words behind, especially with the costs involved. But we genuinely like the idea of an expert trainer using their experience to show us, personally, how to do cool stuff. You need an account at O'Reilly, and their training videos are purchased just like anything else they sell. All your purchases, whether they're ebooks, books or videos are tied to your account, and you access the content from the 'Your Products' page of the web portal.

## Blockbuster video

The video interface has changed considerably over the last couple of months. It works in *Firefox*, but if you use *Chrome* you also get control over playback speed. This helps in some of the more complex moments. Brilliantly, you can choose to either stream or download the video, with the end format being m4a. We only downloaded the 3:34 minute introduction, which is around 30MB, so extrapolating that to the full duration, you should expect the entire video to take up around 2GB of storage. In common with O'Reilly's ebooks policy, these files are completely free of DRM, and played back perfectly in VLC (H264 MPEG-4 video with MPEG AAC audio).

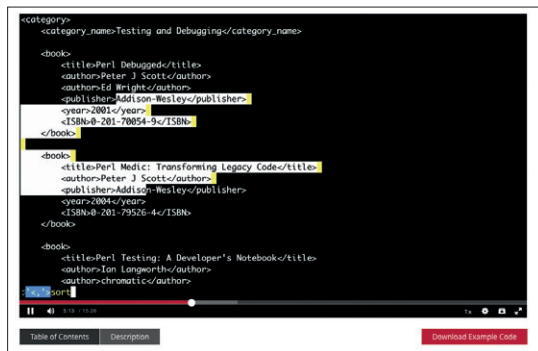


Each chapter is organised and explained brilliantly, and there's a downloadable file for the code, examples and *Vim* configurations mentioned.

We stuck with streaming as it was more convenient. The video quality is on a par with Netflix, albeit with a maximum resolution of 720p. The entire course is split into 22 chapters with some variations in duration, from just a few minutes up to closer to 20. We watched the entire video over the course of about six weeks. This is very much going to depend on your experience level. In the third chapter, for example, Damian recommends going through each key on the keyboard and trying to learn what their functions are, with and without Shift and with the Control keys! This is a tough proposition for our limited brain capacity, but we set an hour aside and tried our best. We didn't remember that many new keys straight away, but forcing ourselves to concentrate made a huge difference and did help us get the most from the remainder of the course.

The sub-heading for this video is 'lesser known features for more effective editing', and each chapter feels like a collection of Damian's hard-won productivity hacks, whether that's post processing your search results or using code completion. This is a course crammed full of practical examples, some live demos and lots of advice. It doesn't touch on the really advanced stuff, but we think it does more than enough to pull the average *Vim* flirt into a more steady, long term relationship, requiring neither the discipline of a book, or huge amounts of coffee.

Being able to watch Damian put examples into practice is another advantage of video.



## LINUX VOICE VERDICT

Beg your boss to take this from your entertainment budget. It's far cheaper than an afternoon at the Celtic Manor.



# Firefox Developer Edition

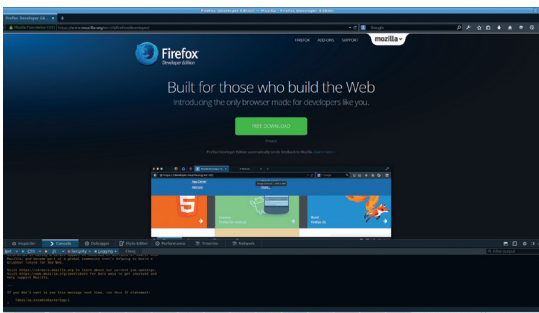
Ben Everard gets a new web browser, but will it make him a better web developer?

**F**irefox is best known as a web browser, but it also has an integrated set of development tools. These have been in place to help web developers see what's going on on their pages. The include tools to inspect particular elements, understand styles, interact with the JavaScript and more. In short, *Firefox* has become one of the most powerful desktop development environments available. All this power is included in the normal desktop version of the browser under Tools > Web Developer menu.

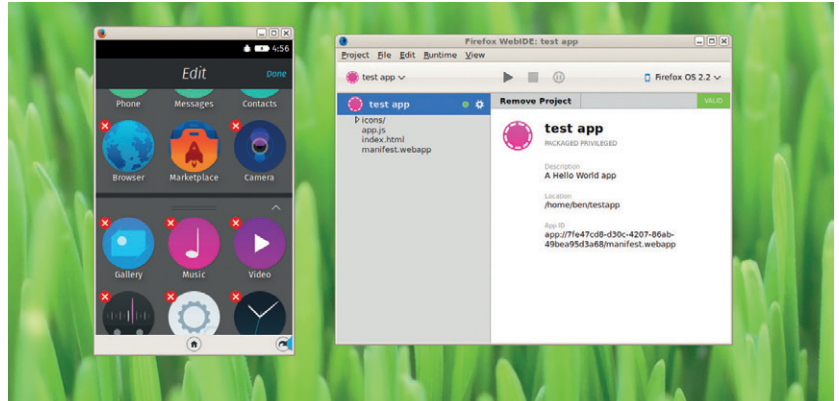
This is useful for anyone creating websites, but now the same web technologies (HTML, CSS and JavaScript) are being used in far more environments than just regular web pages. Thanks to Cordova and PhoneGap, many mobile apps are built in this way, and Firefox OS is an entire ecosystem based around these technologies. *Firefox Developer Edition (FDE)* is a new version of *Firefox* designed to make it easier for people developing on web technologies outside of the browser to get the full advantage of the *Firefox* developer's toolkit.

Most of this is done through the *WebIDE*, which enables you to connect to mobile devices (and simulators), and control them from your desktop. This gives developers a unified interface for all classes of device, and even other browsers on other devices (for example, you can control *Chrome* on Android through *Firefox* on your desktop).

There is a second area of advantage to *Firefox Developer Edition*: it includes features that haven't yet made it to the main stable build of *Firefox*. In Mozilla parlance, *FDE* takes the place of Aurora, which is the stage before Beta. In simple terms, it will get features 12 weeks before they're released in the main version of *Firefox*. Whether or not this is a good thing remains to be seen. After all, not everyone wants their development environment to include features not yet considered fully stable.




The dark blue theme is, presumably, designed to make the interface more leet, and make us developers stand out from the crowd.



In fact, the *WebIDE* environment is one of these advanced features. It is coming to mainline *Firefox* (though perhaps not installed by default), but *FDE* includes the very latest build, and will continue to be ahead of mainline *Firefox* even after it's released. Additionally, *FDE* comes with a different theme than regular *Firefox*, which also gives easier access to the developer tools (for anyone who doesn't use Ctrl+Shift+I to bring them up anyway). In reality, this doesn't add much though.

## Developers: the ball's in your court

Overall, *FDE* definitely shows some promise. *Firefox* is moving at a pace that can make it hard for developers to keep up. This gives a three-month head start on the main edition for anyone working on cutting-edge web software. The developers' tools are also advancing, so having the latest build can make your life easier. However, we'd be a little concerned about basing our development environment on software that's not yet considered stable enough for general use.

Although there are certainly some developers who will appreciate *FDE*, it may appeal more to early adopters who like to always have the latest technology. In recent years, the performance of JavaScript engines – particularly *Firefox's* – have been increasing rapidly. Although *Firefox Developer Edition* isn't designed specifically for speed, it does have more recent optimisations. When we tested the latest *FDE* against the latest stable version of *Firefox*, we found the former to be about 10% faster on JavaScript benchmarks. 

The *WebIDE* controlling a FirefoxOS simulator is a great for any developers who haven't been able to get hold of the real hardware.

## DATA

**Web**  
[https://developer.mozilla.org/en-US/Firefox/Developer\\_Edition](https://developer.mozilla.org/en-US/Firefox/Developer_Edition)  
**Developer**  
 Mozilla  
**Licence**  
 MPL

## LINUX VOICE VERDICT

New developer tools, but these come with some concerns over stability.



# Data Science For Business

Ben Everard is now applying data-driven decision making to Linux Voice.

The information age, it seems, has given way to the data age. Whereas once the internet was seen as some magic way of us getting information via machines, it's now increasingly becoming a way of businesses getting data on us via machines. This process of gathering and analysing vast quantities of data has sired an industry: data science.

*Data Science for Business* isn't a handbook for aspiring data scientists looking to break into this industry (although it could fulfill that role). Instead, it's a book for people in business wondering how they can make use of data science in their decision making. It is, the authors claim, inspired by a course on data science targeted at MBA students.

This gives away its level: it's non-technical from a programming perspective, but also unafraid to get into the maths of the problems. It's also heavily focused on how

the theory links up with real-world benefits to the business.

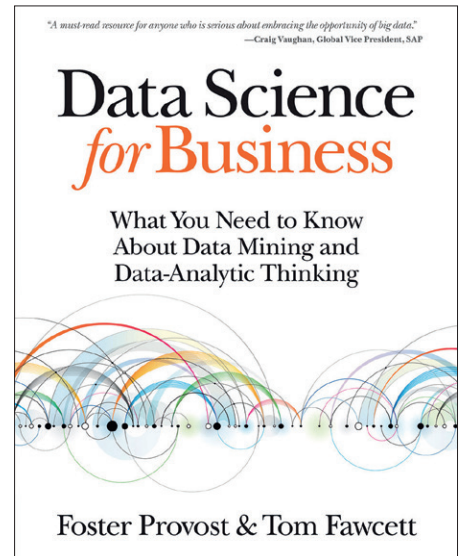
The target reader is someone looking to arrange a data science approach in an organisation, not the person actually doing the implementation. That said, the book is sufficiently broad and well written that it will be of interest to anyone interested in the area in general as long as they don't expect a detailed guide to implementing the approaches.

**LINUX VOICE VERDICT**

Author Foster Provost and Tom Fawcett  
 Publisher O'Reilly  
 ISBN 987-1-449-36132-7  
 Price £25.99

Everything you need to know about data science, except how to program it.

★★★★★



The numbers don't lie. Businesses that base their decision-making on data science are more productive than those that don't.

# Bulletproof SSL and TLS

In HTTPS we trust. But it wasn't until now that Graham Morrison understood why.

SSL isn't *that* difficult to get up and running. But because it deals with some powerful magic and requires a very specific balance of ingredients, it can be both intimidating and easy to get wrong. And when you get it wrong, the consequences can be catastrophic. Encryption and the subtle requirements of certificate exchange are difficult subjects, and as the author of this incredible book explains, it would take a couple of your Earth years to really get your head around the subject. This is what he's done on your behalf.

Despite its scary title (for us) and the subject matter, we found *Bulletproof SSL and TLS* incredibly accessible. Anyone reading this magazine should feel at home with the level and terminology, from its foundations describing network layers through to eventual deployment. That's not to say things don't get technical in the end – they do, but SSL starts to make sense way before then. In particular, we really liked the way the author goes into a lot of detail about how SSL can be compromised.

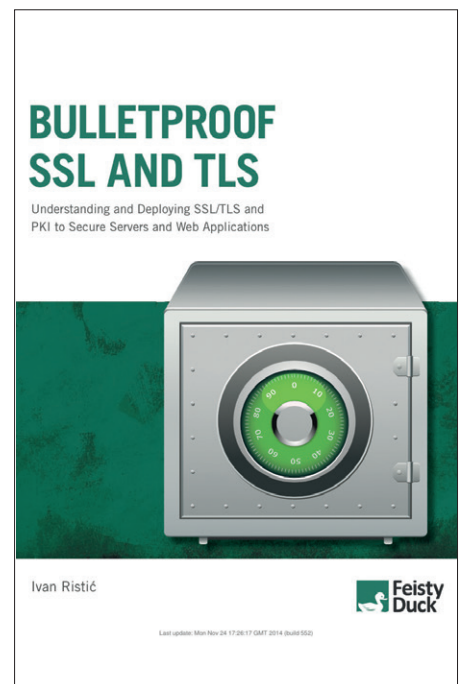
We're reviewing the digital edition, which is available through the publisher's portal as PDF, EPUB and for Kindle with a single purchase. There's no DRM, but your name and email address are used as a watermark. The PDF worked great on the desktop, and we used the EPUB to catch up from a tablet. Since its release in September, the digital versions have been updated by the author, which is another huge advantage with digital, and he actually considers the title a 'living book' that he wants to keep updating. As a result, this is an essential read, not just for sysadmins, but for anyone who puts their trust in HTTPS.

**LINUX VOICE VERDICT**

Author Ivan Ristić  
 Publisher Feisty Duck Ltd  
 ISBN 978-1907117046  
 Price (Ebook) £24 (Print + Ebook) £34

An exhaustive and practical guide to an essential part of the internet.

★★★★★



Over the last couple of months we've enabled SSL on almost our entire web presence, with **LinuxVoice.com** being the last stop.

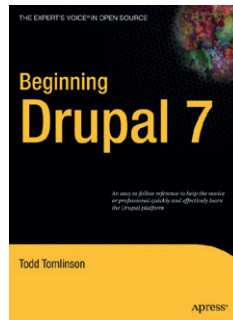
## Beginning Drupal 7

Ben Everard gets his monthly dose of web frameworks

**B**eginning *Drupal 7* is quite a modest name. In fact, this book contains easily enough information for a beginner to create a site from scratch and keep it running. Todd Tomlinson goes through everything you need to know in an orderly fashion. It's well explained, and there are plenty of pictures to guide you along the way.

The book gives an extremely thorough introduction to *Drupal 7*; however, it is also a little unexciting. While no book we've ever read on a web development framework has been a real page-turner, *Beginning Drupal 7* does feel as though it's lacking sparkle. The examples are all very functional, and the prose is clean, but unremarkable. Some people will think this a good thing, but this reviewer is easily distracted when reading technical books and finds that a little character can make it easier to learn.

Overall, *Beginning Drupal 7* is a perfectly good book for learning *Drupal*, but it falls



*Beginning Drupal 7*'s appendices take the user through eCommerce, social media and more.

short of greatness by not drawing the reader into the subject. It relies on the reader having some external motivation for reading.

### LINUX VOICE VERDICT

Author Todd Tomlinson  
Publisher Apress  
ISBN 978-1-4302-2859-2  
Price £39.49

A good, if unexciting, introduction to the *Drupal* web development framework.



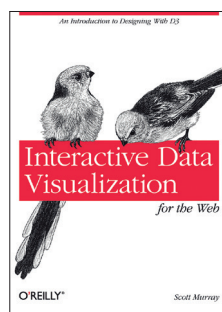
## Interactive Data Visualisation for the Web

Ben Everard is now busy drawing pretty pictures in JavaScript.

**D**3 is one of the most powerful JavaScript graphing libraries, and with that power comes quite a bit of complexity. Creating D3 graphics is not for the faint-hearted, but the reward for the complexity is the ability to create custom, interactive graphics that work smoothly across all modern browsers.

*Interactive Data Visualization for the Web* aims to give a gentle introduction to D3. In fact, it starts at the most basic level, and doesn't even assume the reader understands HTML. However, the whole of HTML, CSS and JavaScript is covered in just 44 pages, so it's alright for a refresher, but someone new to the subject would probably struggle to follow it.

None of the book requires detailed knowledge of these areas, but you do need a good understanding of the basics. Once the book reaches D3, the pace slows down, and Scott Murray takes the reader gently through the basics of how to use the library.



Alas, this book contains no useful information on how to get birds to perch on your titles.

If you want to build custom visualisations, D3 is probably the best option, and this book is probably the best option to get started with D3.

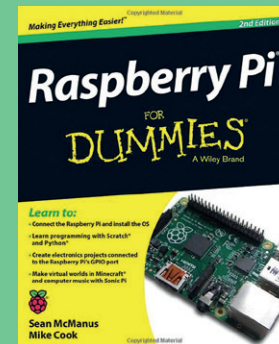
### LINUX VOICE VERDICT

Author Scott Murray  
Publisher O'Reilly  
ISBN 978-1-449-33973-9  
Price £25.99

*Interactive Visualization for the Web* an easy introduction to the complex world of D3



## ALSO RELEASED...



The second edition should be out buy the time you read this.

### Raspberry Pi for Dummies

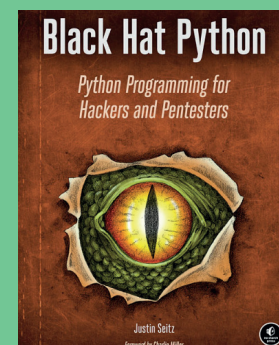
This is the second edition of a title that has proven very popular. Surprisingly, the varied and eclectic nature of the Pi makes it a good fit for the 'Dummies' brand, which may entice complete beginners into trying something new. We're looking forward to taking a look.



Put all your Lego into a single box and get inventing.

### Steampunk Lego

Forget the instructions and leave this lounging on your coffee table. Soon, your living room will be full of floating Edwardian palaces, Victorian warships and weird brass goggles. The pages we've seen look gorgeous and should be a real inspiration.



*The Battle of the Five Armies* should be out by now.

### Black Hat Python

We absolutely love the idea of this book. Use Python to sniff and manipulate network packets, create your own trojans and Windows COM automation, plus all kinds of other dark super powers. Python is accessible enough to make it fun, and vitally informative when it comes to your own defences.

# GROUP TEST

The web browser is the most indispensable software in your distro. **Mayank Sharma** looks at six of the most popular options.

## On Test

### Firefox



**URL** [www.firefox.com](http://www.firefox.com)  
**VERSION** 33.0.3  
**LICENCE** MPL 2.0  
*Can the default browser for most Linux distros stand up to the competition?*

### Chromium



**URL** [www.chromium.org](http://www.chromium.org)  
**VERSION** 38.0.2125  
**LICENCE** Various open source licences.  
*Is the browser from Google just an open source hogwash?*

### Epiphany



**URL** <https://wiki.gnome.org/Apps/Web>  
**VERSION** 3.12.0  
**LICENCE** GPL v2  
*Can GNOME's default web browser hold the fort?*

### Konqueror



**URL** [www.konqueror.org](http://www.konqueror.org)  
**VERSION** 4.14  
**LICENCE** GPL v2  
*Can this multifaceted option from KDE take on purpose-built web browsers?*

### Midori



**URL** [www.midori-browser.org](http://www.midori-browser.org)  
**VERSION** 0.5.8  
**LICENCE** LGPL v2.1  
*Will this minion be able to hold a candle to the giants?*

### Opera



**URL** [www.opera.com](http://www.opera.com)  
**VERSION** 12.16  
**LICENCE** Proprietary  
*Do we really need proprietary software in this field?*

## Web browsers

**W**eb browsers shape the way we view and interact with the internet. They have grown along with the internet as it evolved from primarily a read-only medium to a content-creation platform. As content producers explore new avenues of pushing more content and creation avenues to us users, web browsers must keep pace with the new and upcoming protocols and web technologies that piggyback the content.

It's fair to say that the web browser has become the most widely used piece of software. With the rising number of web-based apps and cloud services, the web browser is probably the first app you call upon after logging into the desktop. In fact, for some people it wouldn't be unfair to say that the performance of the browser dictates their whole desktop experience. This is why you need to make sure you pick the correct web browser for you.

A web browser is a complex piece of software, though it might not look it. You want it to be secure while you use it to pass your credit card information to an online retailer. Furthermore, you want it to be reliable when you're using a web-based email service or an online office suite or updating project specs on the corporate intranet. Finally, you want it to be able to handle all sorts of multimedia while being zippy enough so as not to sap the resources on your computer. And you want all of this in a well-integrated package that offers a great user experience.

In the good old days, the choice was simple as there were few options. For a long time, *Firefox* was the default web browser for virtually every Linux distro. However, over the years it's given users a lot of reasons to demand alternatives and the community and the larger Linux ecosystem hasn't disappointed. We evaluate some of the best options that are also easily accessible.

**“The web browser has become the most widely used piece of software.”**

### Testing the browsers

We've tested the browsers on a variety of parameters. Some parameters, such as the availability of add-ons and extensions, get more weight than, say, the availability of a feature like private mode. To test their adherence to web standards we also subjected them to popular tests such as Acid 3, which checks compliance with elements of various web standards including

Document Object Model (DOM) and JavaScript. In addition to the browser, we also look at their ancillary services such as bookmarks, users and download management. While all of the browsers on test should be available on virtually every Linux desktop distro, we've assessed them from inside the Arch-based Manjaro Linux as well as on Fedora 20.



# Plugins and add-ons

Customise your web experience with extra bits and bobs.

While all browsers are usable straight out of the box, you'll need to extend them with add-ons and plugins for a truly customised user experience. In that respect all web browsers give you access to add-ons for you to tailor the app to your needs.

The most popular browsers usually have the largest community of users and developers and as a result have the largest and most varied selection of extensions.

*Firefox* has a dedicated add-ons website (<https://addons.mozilla.org>) and, while *Chromium* doesn't have a dedicated extensions site of its own, its users can flesh it with add-ons from *Chrome's* web store (<https://chrome.google.com/webstore/category/extensions>). Both websites list hundreds of add-ons in well laid-out categories and allow you to find and install extensions in a matter of clicks. *Opera* has a dedicated add-ons website as well (<https://>

[addons.opera.com/en/](https://addons.opera.com/en/)), but its collection isn't as diverse as *Firefox's* and *Chromium's*.

The minuscule *Midori* browser ships with over two dozen extensions that you can enable and configure from within the browser itself. *Epiphany* doesn't have a traditional extensions model that allows plugging in external add-ons. Instead the extensions for *Epiphany* are shipped along with the browser itself. Similarly, *Konqueror* also ships with extensions.

## Opera 12.16

Can it conduct the choir?

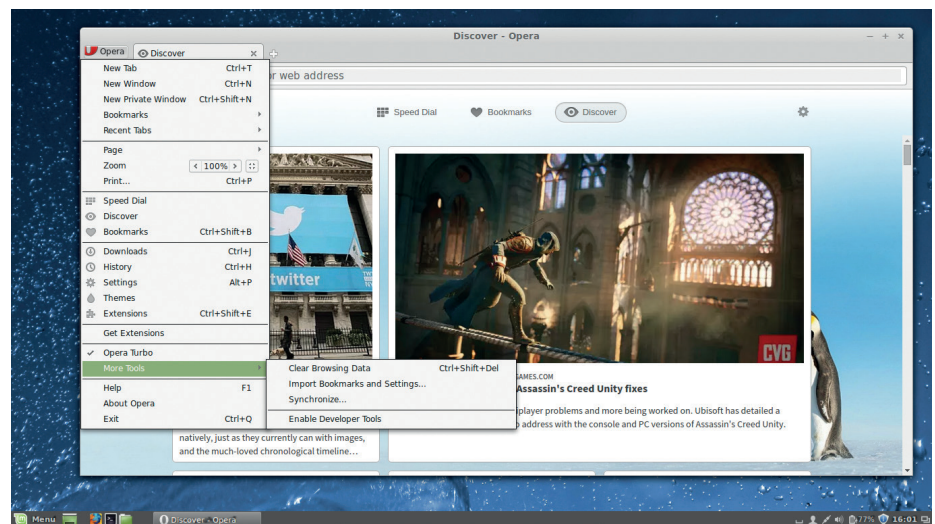
*Opera* is one of the oldest browsers, having come into being as a research project in 1994. Its first version was released in 1995 but it wasn't until 2000 that *Opera* was available for Linux, and it took another five years for the software to become ad-free freeware.

*Opera* on Linux includes all of the features you'd expect from a web browser, including support for multiple tabs, a search-engine toolbar, bookmark management, a password manager, per-site security configuration, download manager, auto-updater, and optional add-ons.

One of its best privacy-focused features is the option to delete private data, such as cookies, browsing history, items in cache and passwords with the click of a button. The browser also likes to flaunt its security-related features. When visiting a site, *Opera* displays a security badge in the address bar that shows details about the website, including its security certificates. *Opera* also checks visits against blacklists for phishing and malware websites, and displays a warning page if you visit a known offender.

### Resting on its laurels

The status bar at the bottom includes buttons to turn on the Turbo mode and Opera Link. The Turbo mode shrinks the pages before sending them to the user. This helps keep costs down on networks where you are charged for the amount of data transferred. The feature is turned off by default and even after you enable it, it'll



The *Opera Mini* browser for Android is still one of the popular mobile browsers, particularly for the Turbo feature which is known to reduce pages by up to 80% of their original size.

ignore any traffic passing on secure HTTPS channels. Opera Link is the browser's synchronisation feature, which can sync bookmarks, history, searches and more. If you want to extend *Opera*, you can download extensions and themes from the dedicated website (<https://addons.opera.com>).

In terms of performance, *Opera* is comparable to *Firefox* and *Chromium*. But while it passed the Acid 3 test, *Opera* scored lower in the HTML 5 test. That's primarily because *Opera's* current stable Linux version was released back in July 2013 and still uses the *Presto* layout engine. This has been replaced by Google's *Blink* layout engine which is used in the current stable builds for Windows and Mac OS X. On Linux, the *Blink*-based *Opera 26* is only available in beta.

The beta version is available as a binary download for popular Linux distros. It features an updated simplified interface and

an improved Turbo mode. There's also a new Discover feature that shows news and other articles much like StumbleUpon. This beta version also lets you share individual bookmarks or entire bookmark folders with anyone on the web by creating a special [share.opera.com](https://share.opera.com) URL with your shared bookmarks that's valid for 14 days.

As expected, the beta version scores higher on the HTML 5 test than the stable version. Although we didn't run into any unexpected problems, the software is beta for a reason and it must be pretty serious. What else would explain the lack of a stable Linux release for more than a year?

### VERDICT

A proprietary solution that runs well but doesn't offer anything spectacular.

★★★★★

# Epiphany 3.12

A profound experience.

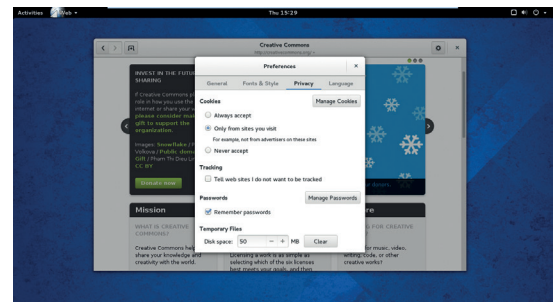
**E**piphany, now rechristened simply as *Web*, is the default browser of the Gnome desktop environment. As part of the Gnome desktop, the browser adheres to the Gnome Human Interface Guidelines (HIG) and maintains a simple user interface with only a required minimum number of features exposed to users.

*Epiphany* has all the core web browser features such as tabbed browsing, bookmark management, and an incognito mode. Since version 3.2 the browser can also be used as launchers for web apps. The launchers are standalone instances of the browsers that are listed along with the offline apps in the Applications menu of the desktop. You can access and manage the launchers with the special **about:applications** URI from within *Epiphany* itself.

*Epiphany's* built-in preference manager is designed to present user-only basic browser-specific settings.

To tweak more advanced options you need the external GSettings configurators such as **dconf** or the graphical *dconf-editor*. The 3.12 version is a major release of the browser that includes performance and user interface enhancements such as the new address bar design, which replaces the traditional URL bar with the page title once the page has finished loading. The recently released 3.14 version is just a minor update that adds support for blocking invalid SSL certificates, and improved security by warning users visiting pages with mixed content.

However, we aren't impressed by the browser's usability. For starters, you can't switch tabs by using the popular Ctrl+Tab key combination. Also, you



*Epiphany* is now the default browser in the Raspberry Pi's official Raspbian distro.

can't customise the websites displayed in the speed dial, and there is no add-ons management infrastructure. We also feel that the redesigned address bar might confuse new users who won't expect to find it there.

**“As part of Gnome, Epiphany adheres to the Gnome Human Interface Guidelines.”**

**VERDICT**  
If you're a Gnome user, use this default browser for creating nifty little web apps.  
★★★★★

# Konqueror 4.14

Too many kooks?

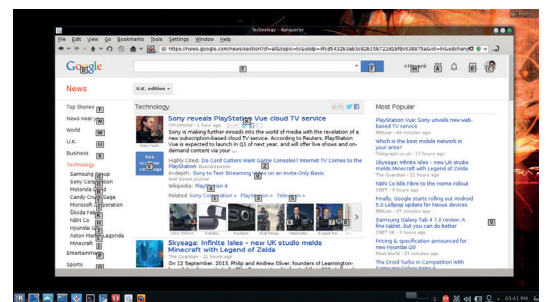
**W**hat *Epiphany* is to Gnome, *Konqueror* is to KDE. The browser is part of the KDE Software Compilation and up until the release of KDE 4 was one of the best things about KDE – a fully functional web-browser and a very capable file manager. In fact it was pitched as an advanced file manager that can display web pages. With the release of KDE4, *Konqueror* was replaced as the default file manager by *Dolphin*. However *Konqueror* can still display a web page just as easily as it can display a Samba share or a remote FTP site.

By default, *Konqueror* uses the *KHTML* rendering engine. Although this supports the latest web standards such as HTML 5, JavaScript, CSS 3 and others, you should use the browser's ability to change rendering engine and switch to *WebKit* which enhances the user experience manifold.

*Konqueror* is well integrated within KDE and uses the KParts object model to let you view various types of files from with *Konqueror* itself. So in essence it provides you with a PDF viewer, an FTP client, a text editor, a spreadsheet editor, a word document editor, an SVN client and more, all within the browser window itself.

**A many-stringed bow**  
As a web browser, *Konqueror* includes a bookmark manager, password manager and a tabbed interface, and lets you define web shortcuts for quick access to popular web services. The browser comes with a nice set of plugins such as a custom ad blocker, automatic web page translation tool, a user agent switcher, shell command panel, and more.

*Konqueror* also ships with multiple profiles each designed for a particular



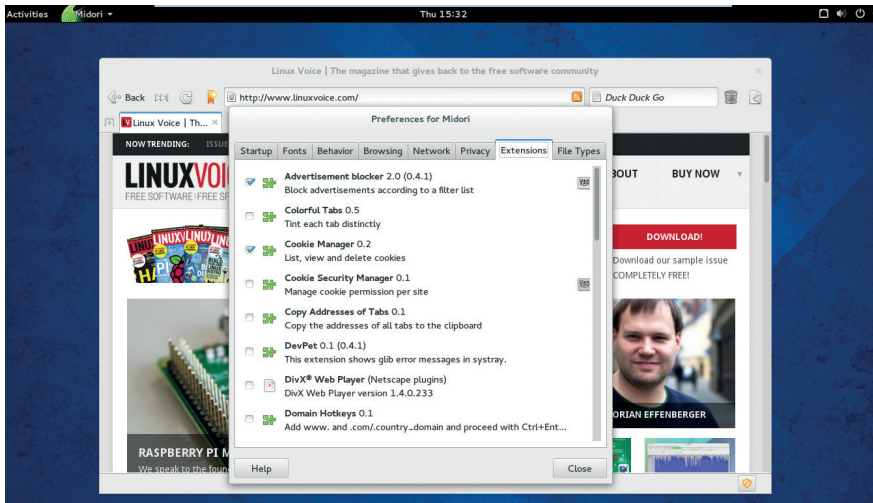
One of the good things about *Konqueror* is that it can be controlled entirely from the keyboard.

use type and you can also view multiple websites in a single window with its split view functionality. However, one of the things that's missing from *Konqueror's* repertoire is an incognito private browsing mode. You also won't find any privacy options in its long list of settings, nor will you find a long list of add-ons.

**VERDICT**  
As far as default browsers go, *Konqueror* is ace.  
★★★★★

# Midori 0.5.8

Not as green as you might expect.



In *Midori*, private sessions run as a separate process, so if your private browsing crashes, it won't affect the normal browser session.

Development on the *Midori* browser started in 2003 and the browser was designed with the idea to make most of the available resources. Quick launch speeds and minimal resource usage are the hallmarks of the browser. No wonder then that *Midori* is popular with lightweight distros and the default browsers on distros such as SLiTaz, Trisquel Mini and Elementary OS.

It uses the *WebKit* rendering engine and performs just as well as the other browsers using this engine. Yet despite its lightweight nature and design, *Midori* has all the features you'd expect from a web browser including a speed dial, tabbed interface, bookmark management, configurable web search as well as an incognito mode.

The browser can show you DOM storage items and can create multiple browser profiles. Like *Epiphany*, *Midori* can also create desktop launcher shortcuts for websites. These launchers run simple single-instance windows of the browser that have no address bar.

*Midori* can clear private data with a single click and you can also set it to clear data while quitting. You'll also find some privacy-related control in the browser's settings section. And there's a trash icon next to the address bar that lists recently closed tabs and windows along with a button to clear this list.

The browser also ships with a bunch of useful add-ons. There's an ad-blocker,

cookie manager, external download manager, feed panel, mouse gestures, custom keyboard shortcuts and more, along with support for *Netscape*-style plugins for supporting different media through external players such as *Totem* and *VLC*.

The latest release of the browser features several notable improvements to its *WebKit 2* rendering engine including improved text selection behaviour, favicons and more. The AdBlock extension has been rewritten in Vala and features a new status bar entry to toggle the add-on for individual sites and also show a list of items that were blocked on the respective site. The release also features two new extensions. One lets you use the Ctrl+Enter key combination to autocomplete URLs and the other adds a little notes panel that saves any selected text as you browse the web.

*Midori* was remarkably stable and properly rendered every popular social network we pointed it to including Facebook, Twitter, Youtube, LinkedIn, and Spotify. However it doesn't have a synchronisation feature, nor a mobile client and while its collection of add-ons is useful it's nowhere as comprehensive as the web stores of *Firefox* and *Chrome*.

## VERDICT

An impressive browser that's light only in size and not features.

★★★★★

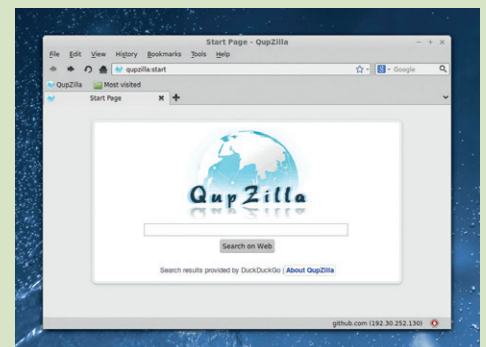
# Variants and other options

Look beyond the obvious.

In this test we've only looked at some of the most popular web browsers available and there's no dearth of alternatives. If you're looking for more than just a browser, there's *SeaMonkey*, which is the continuation of the Mozilla Application Suite and uses the same code as *Firefox* but is developed outside the control of the Mozilla Foundation. In fact, *Firefox* has spawned several browsers each with its own distinct purpose. There's the ESR version (Extended Support Release) for users who don't want a new version every few weeks (or for organisations that just can't handle the hassle of constant upgrades).

If you are a Debian user, your distro ships with the *Iceweasel* browser, which is a rebranded *Firefox* release stripped of all the Mozilla trademarks. Then there's *PaleMoon*, also from *Firefox*, that has stopped support for older hardware and optimised it for performance on newer devices.

Similarly, *Chromium* fuels a bunch of browsers as well, most notably the proprietary *Chrome* browser. The *SRWare Iron* browser is another proprietary freeware browser that aims to eliminate usage tracking and other privacy-compromising functionality in *Chrome*. KDE users should also check out the lightweight *WebKit*-based *Rekonq* browser, which is also pretty well integrated in KDE. There's also *Dillo* for resource-conscious users and some text-based browsers such as *Lynx*, which is still in active development 22 years after its initial release.



*QupZilla* is a zippy browser that's designed to provide a native feel on all supported platforms and across Linux desktops.

# Firefox 33.0.3 vs Chromium 38.0.2125

The battle of the alpha behemoths.

**F**irefox and Chromium are by far the biggest and most comprehensive web browser projects of the lot and the only real challengers to each other.

Firefox has long been considered the *de-facto* browser for the open source community. It was fast, it was innovative and it championed standards compliance while other browsers of the time (we're looking at you, *Internet Explorer*) were trying to manoeuvre the web as per their whims and wishes.

Firefox proudly continues its tradition and is still known for its technical innovations and customisations. If there's something you can do with a web browser – tabbed browsing, location-aware browsing, incremental find, smart bookmarking, managing downloads, browsing privately – you can do it with *Firefox*. The browser also strives to support the new and upcoming open technologies such as WebM and HTML 5.

Firefox is also the most localised web browser, with support for over 70 languages. The browser has an excellent security record and is known for offering bug bounties to developers who discover a security hole. It also implements Google's Safe Browsing API to safeguard its users from phishing and malware.

There was a time when other browsers scored over *Firefox* just for their looks and customisation potential. However, the latest version of *Firefox*

is highly customisable. In fact you can rearrange the entire interface as per your needs. The browser now ships with a new view for customisation that lets you rearrange the various components of the browser window.

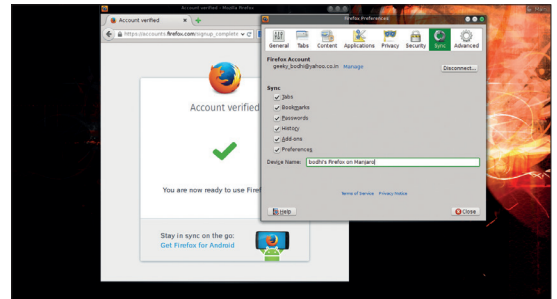
Another strength of the browser is the plethora of add-ons freely available on a dedicated website of their own. There's also the Firefox Sync service, which helps users synchronise passwords, history, bookmarks and open tabs by storing encrypted copies on Mozilla servers.

Firefox is also a wonderful platform for web developers and includes tools such as the Error Console and the DOM Inspector as well as extensions such as Firebug. Furthermore, Mozilla has recently announced a new release dubbed Firefox Developer Edition built especially for web developers.

### All that glitters ain't gold

The *Chromium* project impressed users right from its initial release in 2008. The open source project lends code to Google's proprietary *Chrome* browser, which only adds some Google trademarked and proprietary code on top of *Chromium*.

*Chromium* impressed users with its minimalist user interface. The browser won accolades and users for being faster than *Firefox* and for making judicious use of computer resources despite matching the latter for features. *Chromium* can do all the usual things you can do with *Firefox* and also



Firefox is available for all major operating systems and also works on Android smartphones.

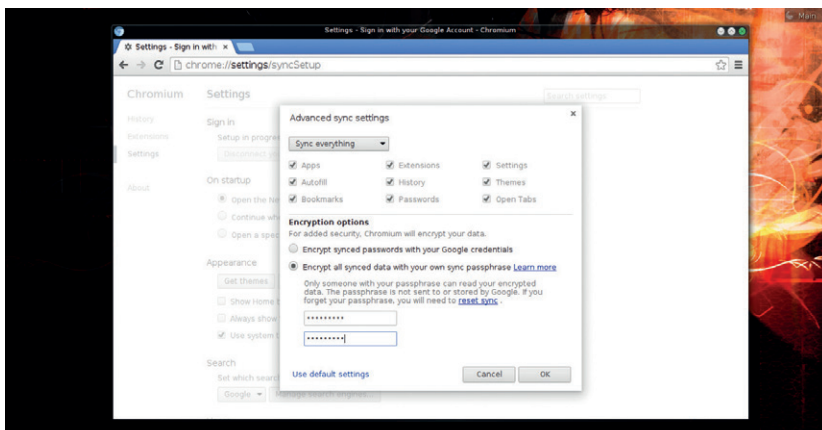
includes a PDF reader. Like *Firefox*, it's hot on new and upcoming open protocols, and supports Vorbis, Theora and WebM codecs among others.

*Chromium's* snapshots are built automatically several times a day and made available as binary releases. These are then taken by distro packagers and included in the respective software repositories. While the browser is available for multiple platforms, there is no mobile client, although you can compile one yourself. You can extend *Chromium* by

**“Like Firefox, Chromium is hot on upcoming open protocols, and supports Vorbis.”**

fetching extensions from *Chrome's* web store, which is as diverse as *Firefox's*. *Chromium* also sports a sync feature similar to *Firefox*.

*Chromium* has more privacy-related controls than *Firefox*, but that's primarily because the browser uses more privacy-invasive services than *Firefox*. For example, *Chromium* uses a web service to resolve navigation errors along with a prediction service to complete searches and URLs. The one area that *Chromium* scores over *Firefox* is user management, which is more intuitive in *Chromium* than in *Firefox*.



Chromium uses multiple processes to isolate websites in different tabs from each other.

**VERDICT**

<b>FIREFOX:</b> The open source champion does well to keep up with the competition.	<b>CHROMIUM:</b> It won users with its speed but it has to pay heed to Google's whims.
★★★★★	★★★★★

# OUR VERDICT

## Web browsers

This was always going to be between *Firefox* and *Chromium*. That's not to say that the other browsers didn't stand a chance. In fact, we were tempted to give the top honours to *Midori*. *Midori* is an impressive piece of software that's not only fast and lightweight but is also full of useful features and has enough room for extensibility. It easily obliterates the other contenders in this group test but lacks the advanced power user features that you get with *Firefox* and *Chromium*, such as a synchronisation feature.

There's not much to choose

from *Chromium* is its process model, which handles unresponsive web pages more gracefully than *Firefox*, but is also responsible for its high resource usage.

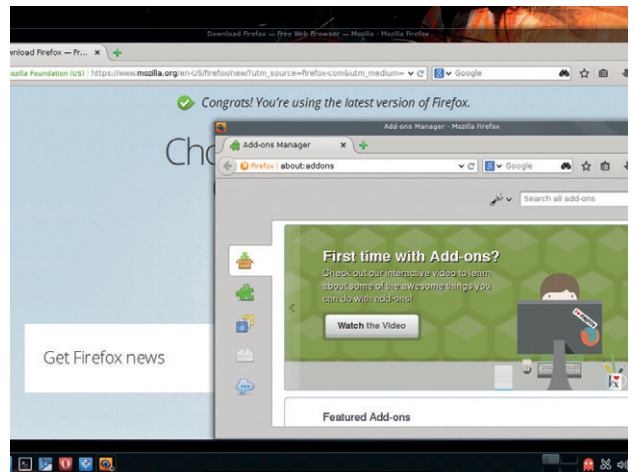
But *Firefox* is more than just a browser and has been fighting the good fight for quite some time. Its commitment to promote open standards, privacy and the open web are as important and worthy of support as the software itself. Also, unlike some projects, *Firefox* doesn't rest on its laurels and strives to innovate and adapt to the dynamic space in which it operates. The project has just celebrated the

**“Firefox strives to innovate and adapt to the dynamic space in which it operates.”**

from between *Firefox* and *Chromium*. As browsers they are almost equally matched. We like *Firefox's* implementation of a few features, most notably Firefox Sync, which encrypts all synced data and also lets you use a custom server instead of Mozilla's. Over the years *Firefox* has also been able to streamline itself and match *Chromium* for speed and performance. The one feature *Firefox* hasn't been able to adapt

10th anniversary of *Firefox 1.0* with a special anniversary release that comes with a Forget button that helps you erase recent activity if you accidentally fall down a rabbit hole on the web.

Mozilla's official mission statement is to build a better internet. The team also just happen to produce a wonderful browser that outperforms the competition on merit. We're glad to see it back at the top of the tree. 🐘



This group test wasn't lost by any browser (there's some fantastic software for Linux) as much as it was won by *Firefox*.

### 1st Firefox 33.0.3

Licence MPL 2.0 Version 33.0.3

[www.firefox.com](http://www.firefox.com)

If it keeps up the good work, it'll be hard for anyone to displace *Firefox* from the top step.

### 2nd Chromium 38.0.2125

Licence Various open source licenses. Version 38.0.2125

[www.chromium.org](http://www.chromium.org)

The only real competition to *Firefox*; loses for its focus on powering proprietary solutions instead of catering to its users.

### 3rd Midori 0.5.8

Licence GPL v2 Version 3.12.0

[www.midori-browser.org](http://www.midori-browser.org)

The go-to browser for anyone concerned about resource consumption.

### 4th Konqueror 4.14

Licence GPL v2 Version 4.14

[www.konqueror.org](http://www.konqueror.org)

Like most things KDE, *Konqueror* loses out for trying too hard. As part of an integrated desktop though, it's well worth trying.

### 5th Epiphany 3.12

Licence LGPL v2.1 Version 0.5.8

<https://wiki.gnome.org/Apps/Webe>

Gnome's default browser is good for creating web apps, but falls behind in daily usage.

### 6th Opera 12.16

Licence Proprietary Version 12.16

[www.opera.com](http://www.opera.com)

The ugly proprietary duckling pales in front of the white open source swans.



*Midori* ships with several useful plugins, and is fantastic for older machines.

# SUBSCRIBE

shop.linuxvoice.com



Introducing **Linux Voice**, the magazine that:

**LV** Gives 50% of its profits back to Free Software

**LV** Licenses its content CC-BY-SA within 9 months

### 12-month subs prices

- UK – £55
- Europe – £85
- US/Canada – £95
- ROW – £99

### 7-month subs prices

- UK – £38
- Europe – £53
- US/Canada – £57
- ROW – £60

DIGITAL SUBSCRIPTION ONLY £38

Get 114 pages of tutorials, features, interviews and reviews every month

Access our rapidly growing back-issues archive – all DRM-free and ready to download

Save money on the shop price and get each issue delivered to your door

Payment is in Pounds Sterling. 12-month subscribers will receive 12 issues of Linux Voice a year. 7-month subscribers will receive 7 issue of Linux Voice. If you are dissatisfied in any way you can write to us to cancel your subscription at [subscriptions@linuxvoice.com](mailto:subscriptions@linuxvoice.com) and we will refund you for all unmailed issues.

# NEXT MONTH IN LINUX VOICE

**ON SALE  
THURSDAY  
29 JANUARY**



## WHAT NOW FOR LINUX?

We asked a bunch of movers and shakers in Free Software what's coming in 2015 – their answers will inform, surprise and delight you.

## EVEN MORE AWESOME!



Image credit: flammkrna CC BY-SA 2.0

### Lennart Poettering

Why is Systemd necessary? Can you fix my broken audio setup? What's it like being an internet celebrity? Hear the answers from the horse's mouth.



### Scribus (again)!

We promise, this time we'll give ourselves plenty of time to look at how well this superb design software integrates with a proprietary workflow.



### Assembly language

If you're as cool as Mike Saunders (and you want to get as close as possible to bare-metal programming), you'll love Assembly language.

# LINUX VOICE IS BROUGHT TO YOU BY

**Editor** Graham Morrison  
[graham@linuxvoice.com](mailto:graham@linuxvoice.com)  
**Deputy editor** Andrew Gregory  
[andrew@linuxvoice.com](mailto:andrew@linuxvoice.com)  
**Technical editor** Ben Everard  
[ben@linuxvoice.com](mailto:ben@linuxvoice.com)  
**Editor at large** Mike Saunders  
[mike@linuxvoice.com](mailto:mike@linuxvoice.com)  
**Creative director** Stacey Black  
[stacey@linuxvoice.com](mailto:stacey@linuxvoice.com)

**Editorial consultant** Nick Veitch  
[nick@linuxvoice.com](mailto:nick@linuxvoice.com)

All code printed in this magazine is licensed under the GNU GPLv3

Printed in the UK by  
Acorn Web Offset Ltd

**Disclaimer** We accept no liability for any loss of data or damage to your hardware

through the use of advice in this magazine. Experiment with Linux at your own risk! Distributed by Marketforce (UK) Ltd, Blue Fin Building, 110 Southwark Street, London, SE1 0SU  
Tel: +44 (0) 20 3148 3300

Circulation Marketing by Intermedia Brand Marketing Ltd, registered office North Quay House, Sutton Harbour, Plymouth PL4 0RA  
Tel: 01737 852166

**Copyright** Linux is a trademark of Linus Torvalds, and is used with permission. Anything in this magazine may not be reproduced without permission of the editor, until July 2015 when all content (including our images) is re-licensed CC-BY-SA.  
©Linux Voice Ltd 2014  
ISSN 2054-3778

Subscribe: [shop.linuxvoice.com](http://shop.linuxvoice.com)  
[subscriptions@linuxvoice.com](mailto:subscriptions@linuxvoice.com)



A veteran Unix and Linux enthusiast, Chris Brown has written and delivered open source training from New Delhi to San Francisco, though not on the same day.

# CORE TECHNOLOGY

Prise the back off Linux and find out what really makes it tick.

## Signals

Get the attention of running processes by sending them signals.

In the world of Linux system programming, a signal is an event that's delivered to a process by the kernel. A signal says to the process "something has happened that you might want to respond to". A few signals are generated as a result of something that the program itself is doing (usually something bad), but most of them originate from sources external to the program itself.

Why do you need to know about signals? Well, they're important to a system administrator because they provide a way to interact with running processes (in particular, to kill them). And the most important single reason that a developer needs to be aware of signals is so that he knows how to write programs that ignore them. But there are more useful things you can do with signals, as we'll see.

There are several different types of signal. If you're running a *Bash* shell, the built-in command **kill -l** will show you a list of them. It's a slightly scary list but you don't need to know about most of them, and here we're going to focus on the 10 or so you're most likely to use.

### SIGHUP

This signal has an interesting history. The "HUP" stands for "hang up" and it harks back to the days when telephones hung on a hook on the wall and you would terminate a call by hanging up the phone. The scenario went like this: Dennis was logged into his PDP11 computer via a dial-up line and a modem. Without logging off, he simply "hung up" the connection. Later, Ken dialled in to the same modem, thus finding himself connected to Dennis's abandoned shell. To prevent this undesirable state of affairs, the

device driver for the modem would detect the loss of carrier when Dennis hung up and deliver a SIGHUP signal to terminate the shell session.

Well, dial-up logins are history now, and SIGHUP was looking forward to a peaceful retirement when it was offered a new job. Nowadays, SIGHUP is interpreted by some daemons to mean "your configuration file has been changed, please go and re-read it". One example is the system logging daemon (**syslog** or **rsyslog**) which re-reads the config file **/etc/syslog.conf** (or **/etc/rsyslog.conf**) on startup and on receipt of a SIGHUP. In some cases the daemon simply stops and restarts when it receives this signal.

### SIGINT

This is the signal that is sent to a foreground process by the terminal driver when you enter ^C on the terminal. By default, programs will terminate when they receive this signal. Some programs, especially ones that operate interactively, choose to ignore this signal.

### SIGTERM

This is conventionally used as a polite "please tidy up and terminate" request. For example, when you shut down a Linux system with the **shutdown** command, it begins by sending SIGTERM to all the running processes in the hope that they will do the decent thing and go away. If this doesn't work, **shutdown** waits for a few seconds then sends a SIGKILL. SIGTERM is the default signal type sent when you use programs like **kill** and **pkill**.

### SIGKILL

This is the most brutal signal because a

process cannot choose to catch or ignore it. A process receiving SIGKILL is instantly terminated. Best practice suggests that as a way of killing a process it should be a last resort, when more polite requests such as SIGTERM have failed. This is particularly true for services that maintain lock files or other temporary data files, because they won't have opportunity to clean them up and you may end up having to manually remove them before the service will restart.

### SIGALRM

This signal is "self-inflicted" – it's generated as a result of an alarm clock timing out. Typically a C program might request an alarm call 10 seconds from now with:

```
alarm(10);
```

and use it to implement a timeout on a potentially blocking operation.

### Setting your interrupt character

The terminal driver (the code inside the kernel that's reading characters from your keyboard) recognises a number of characters that are handled specially. Well-known examples include the "interrupt" character (usually ^C) which sends a SIGINT to any foreground processes running on that terminal, and the "end-of-file" character (usually ^D) which tells a program that's reading its standard input from the keyboard that there is no more data. You can see all of these settings with:

```
$ stty -a
```

Although most of what you'll see here harks back to the days of terminals that plugged into serial ports and is not relevant now, you can also change them. For example, to set your interrupt character to ^X:

```
$ stty intr ^X
```

The control character is entered here as two characters, a caret (^) then the X.



## SIGSEGV

This signal is generated by the kernel when a process tries to access a memory address that's outside its address space. Of course this should never happen in a correctly written program; typically it occurs in C code that makes a reference through a pointer that hasn't been initialised, as this two-liner demonstrates:

```
void main()
{
    int *p;
    *p = 0;
}
```

Assuming the code's in the file `segvdemo.c`, compile and run it like this:

```
$ gcc -o segvdemo segvdemo.c
$ ./segvdemo
Segmentation fault (core dumped)
$ echo $?
139
```

## SIGILL

Another signal that arises directly from the execution of the process. It indicates an illegal instruction, and should never occur unless your compiler is buggy or the executable has become corrupt, or maybe because it calls a function through an uninitialised pointer.

## SIGBUS and SIGFPE

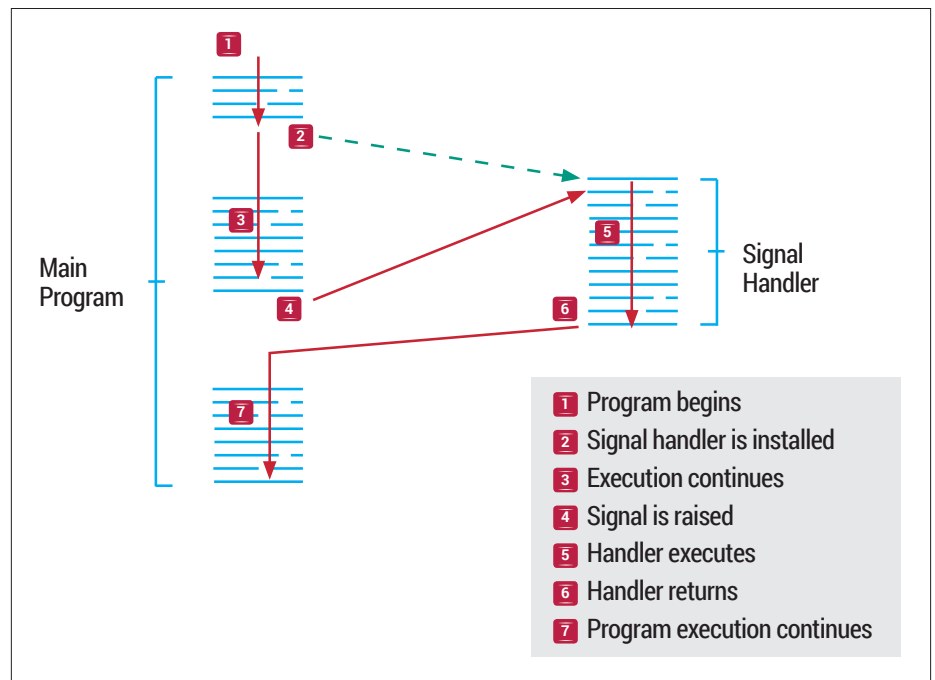
These indicate an incorrectly aligned memory access, and a floating point exception (or other arithmetic error) respectively. It's easy to deliberately generate a SIGFPE – just divide by zero:

```
void main()
{
    int a, b, c;
    a = 1; b = 0;
    c = a / b;
}
```

If you compile and run this program, you'll see something like this:

### Signals are not exception handling

Some languages support exception handling, typically with keywords like "try", "catch" and "throw". For example, if you try to open a file for writing and don't have write permission, in some environments the runtime will throw an exception that you can choose to catch in order to handle the error. We mention this because this is NOT what signals do. You can (to a very limited extent) install exception handling of a sort by catching signals like SIGFPE, but failed system calls and library routine calls do not throw exceptions; they return -1 (or sometimes a null pointer) to indicate failure and you need to explicitly test the return value to detect this.



Arrival of a signal interrupts the execution of the main program and runs the handler.

```
$ ./fpdemo
Floating point exception (core dumped)
$ echo $?
136
```

Again, notice the exit status (136). Subtracting 128 gives 8, the signal number of SIGFPE.

## SIGABRT

A self-generated signal raised when a program calls the `abort()` library function. By default it will cause immediate termination of the program.

## Sending signals

OK, we've discussed some of the signal types. We've seen that some, such as SIGSEGV and SIGFPE, are raised automatically by the kernel as the result of some misdemeanour committed by the program. These are sometimes referred to as "synchronous" signals. But others need to be explicitly generated from outside the program (sometimes called "asynchronous" signals). How do we do that?

One way is to use the command `kill`. It's not a good name really; `raise` or `sendSignal` might be better. For example, we can send a SIGHUP signal to process 12345 like this:

```
$ kill -SIGHUP 12345
```

Or you can use the short signal name, or the signal number, like this:

```
$ kill -HUP 12345
```

```
$ kill -1 12345
```

This is a good time to point out that you can only send signals to processes that you

own, unless you're running as root in which case you can deliver signals to any process.

Sending SIGHUP manually like this is commonly used to signal a service after changing its config file. Manually generated signals are also often used to terminate a "hung" process (or just one that seems to have been running for far too long), typically like this:

```
$ kill -TERM 12345
or more brutally:
$ kill -KILL 12345
```

If you don't specify a signal type, the default is SIGTERM.

As you'll see from these example you need to know the process ID to send a signal. If you're trying to kill a program called `foobar` you might get this by running:

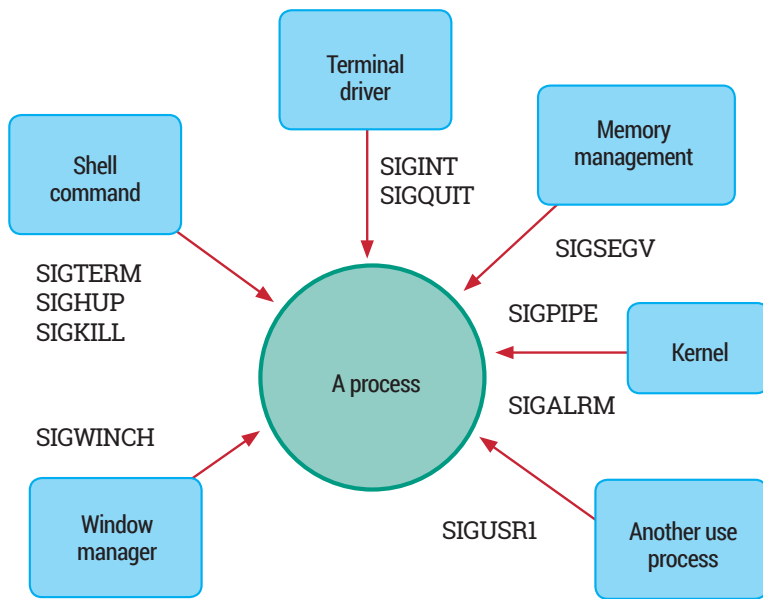
```
$ ps -ef | grep foobar
chris 4923 2586 0 18:07 pts/0 00:00:00 ./foobar
chris 4968 4924 0 18:07 pts/6 00:00:00 grep foobar
```

from which we see that the PID is 4923. (Ignore the "false positive" generated from the `grep` command.)

## Sending signals from a program

So much for sending signals from the command line. You can also send signals from within a program. Here's a little C program I wrote called "terminate"; the idea is that you give it a PID as an argument and it begins by sending a polite SIGTERM signal to ask the process to terminate. If this doesn't work it just pulls out a gun and

Signal sources



The kernel delivers all signals, but different signal types typically originate from different places.

shoots the process in the head with SIGKILL. Note that the line numbers are for reference, they are not part of the file:

```

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <signal.h>
4. #include <errno.h>
6. int main(int argc, char* argv[])
7. {
8.     int targetpid, i;
9.
10.    /* Get target process ID from command line */
11.    targetpid = atoi(argv[1]);
12.
13.    /* Check that the process exists */
14.    if (kill(targetpid, 0) < 0)
15.    {
16.        switch (errno)
17.        {
18.            case ESRCH:
19.                printf("Process %d does not exist\n",
20.                    targetpid);
21.                exit(1);
22.            case EPERM:
23.                printf("Do not have permission to terminate
24.                    that process\n");
25.                exit(2);
26.        }
27.    }
28.    /* Ask the process to terminate (politely) */
29.    kill(targetpid, SIGTERM);
30.    /* Wait for up to 5 seconds for the process to
31.        die */
    
```

```

31. for (i = 0; i < 5; i++)
32. {
33.     if (kill(targetpid, 0) < 0)
34.         exit(0);
35.     sleep(1);
36. }
37.
38. /* Asking nicely didn't work, bring out the big
39.    guns */
40. printf("SIGTERM ineffective, sending
41.    SIGKILL\n");
42. kill(targetpid, SIGKILL);
43. exit(3);
    
```

If you don't read "C", here's a guided tour: At line 11 we grab the process ID from the command line. There should really be some error checking here to verify that the user did actually supply a PID as argument. At line 14 we try to send signal number 0 to the process. The `kill()` system call is analogous to the `kill` command, though notice that the arguments are in the opposite order. (Hey, this is Linux! You want consistency?) Now there is no signal number 0, so the call will not actually deliver a signal to the process, but it will fail (returning -1) if either the process doesn't exist, or we don't have permission to signal it (ie we don't own it and we're not root). These two conditions are trapped at lines 18 and 21, where we print an appropriate error message and exit. If we make it as far as line 28, we know that the process exists and we have permission to signal it, so we send a polite

SIGTERM to the process, hoping it will oblige and go away. Then the loop starting at line 31 repeatedly probes (sending the dummy signal 0 again) to see if the process has terminated. If it has, then fine, our job is done, and we exit at line 34. We continue for five seconds, probing at one-second intervals. Finally, if we reach line 39, we forcefully terminate the process with SIGKILL. This approach (SIGTERM followed if necessary by SIGKILL) is essentially what happens to all running processes during a system shutdown.

Catching signals

So now we know how to send signals. Let's look at the other side of the story – how does a process respond when it receives a signal? Well, each signal has a default disposition ("disposition" is just a posh word meaning "what will happen when a signal arrives"). The three dispositions shown in the table are:

- 1 **Term** The process is terminated (this is the most common behaviour).
- 2 **Core** The process is terminated, a memory image (core file) may be written.
- 3 **Ignore** The signal is ignored.

However – and here it gets interesting – a program can install handlers for the various signal types – pieces of code that will run if the signal arrives.

Rather than do this in C again, we'll do it in a shell script. The purpose of this script is to count the number of prime numbers less than one million.

Now of course, doing a computation-rich thing like this in a shell script is pretty stupid, and I'm not using the most efficient algorithm either, which doesn't help. But that's not the point. The point of this example is that it represents a long-running program that gradually works its way through a data set. Here's the script:

```

1. #!/bin/bash
2.
3. function isprime()
4. {
    
```

Signals and exit codes

When a process terminates "normally" (by executing an `exit()`), it chooses an exit code to pass back to the parent – 0 to indicate success and a small integer to indicate some sort of failure. But if the process is terminated by a signal it doesn't get a choice. In this case, the exit status will be 128 plus the number of the signal that killed it. So for example a process killed by a SIGKILL (signal 9) will have exit status 137 (128+9).

```

5. n=$1
6. factor=2
7. while (( factor * factor <= n ))
8. do
9.   if (( n % factor == 0 ))
10. then
11.   return 1 # number is not prime
12. fi
13. (( factor++ ))
14. done
15. return 0 # no factors, number is prime
16. }
17.
18. trap 'echo Testing value $val, found $count
primes so far' HUP
19. trap 'echo Buzz off I am busy counting primes!'
TERM
20. trap " INT
21.
22. count=2
23. val=5
24. while (( val < 1000000 ))
25. do
26.   if isprime $val
27. then
28.   (( count++ ))
29. fi
30. (( val += 2 ))
31. done
32. echo count is $count

```

Let's walk you through this. Lines 3 to 16 define a function called **isprime**. It takes the number we want to test as an argument, and returns 0 (success) if the number is prime and 1 (failure) if it isn't. The code is not difficult, but its details do not concern us here. The script really starts at line 22. We enter a loop (lines 24 to 31), testing all odd numbers between 5 and 1,000,000 for prime-ness and counting them. (I do at least have the sense not to test even numbers.)

On exiting the loop we print out the answer (line 32).

If you want to try this out (and we hope you will) put the code into a file called **countprimes** and make it executable:

```
$ chmod u+x countprimes
```

Now run the script:

```
$/countprimes
```

It will take quite a while to run (17 minutes on my laptop). Meanwhile, go back and look at lines 18–20. These are the lines that install signal handlers for SIGHUP, SIGTERM, and SIGINT signals respectively. In these examples we have written the signal-handling actions "in line", though we could also have written them as functions, which would be easier to deal with if we wanted the handler to do several things. The SIGHUP handler prints a progress report. The SIGTERM

## Common signals

Signal name	Number	Default action	Description
SIGHUP	1	Term	Some daemons interpret this to mean "re-read your configuration"
SIGINT	2	Term	This signal is sent by ^C on the terminal
SIGTRAP	5	Core	Trace/breakpoint trap
SIGBUS	7	Core	Invalid memory access (bad alignment)
SIGFPE	8	Core	Arithmetic error such as divide by zero
SIGKILL	9	Term	Lethal signal, cannot be caught or ignored
SIGSEGV	11	Core	Invalid memory access (bad address)
SIGPIPE	13	Term	Write on a pipe with no one to read it
SIGALRM	14	Term	Expiry of alarm clock timer
SIGTERM	15	Term	Polite "please terminate" signal
SIGCHLD	17	Ignore	Child process has terminated

Each signal has a name, a number, a default "disposition" and a purpose.

handler prints a rude message, but the program continues executing. The empty SIGINT handler at line 20 simply makes the script ignore SIGINT signals. Since ignoring signals is a common requirement, we'll allow ourselves one more line of C:

```
signal(SIGINT, SIG_IGN);
```

which says to ignore SIGINT signals and is equivalent to the **trap** statement at line 20 in the script.

So go back to the terminal where

### Buzz off I am busy counting primes!

but again, the program will continue. Finally (unless you are actually interested in knowing how many primes under 1,000,000 there really are and would like to allow the program to run to completion) we can forcefully terminate the program with:

```
$ pkill -KILL countprimes
```

We haven't installed a handler for SIGKILL, and we couldn't if we wanted to because you can't catch or ignore SIGKILL, so in the

**"This example represents a long-running program that gradually works its way through a data set."**

**countprimes** is running and enter 'C'. As we've seen, this will send a SIGINT signal to the process. If we didn't have line 20 in the script this would terminate the program, but now it is simply ignored and the program continues.

Now open a second terminal window. Enter the command:

```
$ pkill -HUP countprimes
```

```
Testing value 861877, found 68481 primes so far
```

As you'll see, the SIGHUP handler tells us how far we've got in our prime-counting task. Now try:

```
$ pkill countprimes
```

which sends the default SIGTERM signal and will elicit the response:

first terminal window you'll see the message **Killed**

If you then examine the exit status in that terminal:

```
$ echo $?
```

```
137
```

you see that it's 137. Subtracting 128 as before gives 9, the signal number of SIGKILL.

That's all for this month. If you'd like to learn more, the man page for **signal** (**man 7 signal**) has a great deal more detail, but rapidly gets rather techie. There's also a good discussion in the GNU C Library manual at [www.gnu.org/software/libc/manual/html\\_node/Signal-Handling.html](http://www.gnu.org/software/libc/manual/html_node/Signal-Handling.html). Happy signalling! 📺

# FOSSpicks

Sparkling gems and new releases from the world of Free and Open Source Software



**Mike Saunders** has spent a decade mining the internet for free software treasures. Here's the result of his latest haul...

Sound file tag editor

## Puddletag 1.0.5

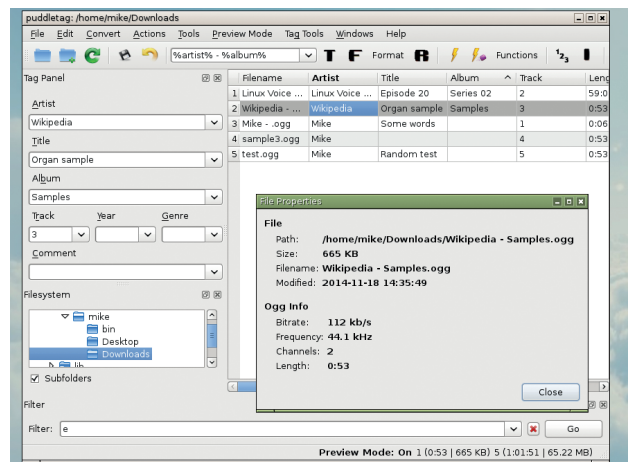
**W**e love discovering programs that ostensibly perform mundane tasks, but have so many features and options that they actually become rather cool. *Puddletag* is one such example: it's a music file tag editor. Riveting, right? But when you start exploring the interface and discover some of the complexity behind it, you actually start to admire it. And if you manage a large music collection, you might find that you can't live without it. Sure, most graphical music players on Linux include some kind of tag editing facility, but *Puddletag* is industrial strength.

It's written in Python (2) and uses *Qt 4* for the interface, so its main dependency is *PyQt4*. You'll also

need *Mutagen* – this is the library that handles the low-level operations of adding tags to music files. On Ubuntu and Debian-based distros, you can get all of the dependencies via the **python-qt4**, **python-pyparsing**, **python-mutagen** and **python-configobj** packages. Then extract the **puddletag-1.0.5.tar.gz** file, go into the resulting directory, and run **./puddletag**.

### I, spreadsheet

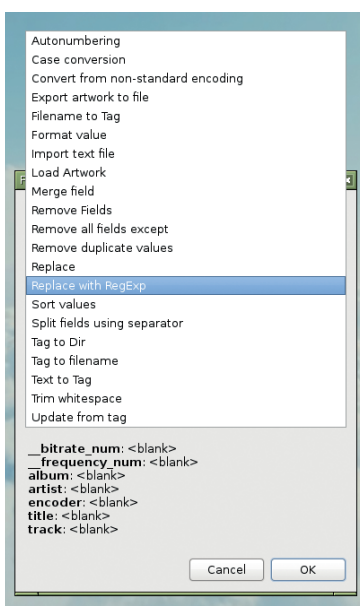
The first thing you'll notice is the unusual interface: *Puddletag* looks somewhat like a spreadsheet. This actually turns out to be a very good design when you're working on lots of files. Under the filesystem panel on the bottom-left, navigate into a



*Puddletag* can work with ID3v1, ID3v2 (MP3), MP4, VorbisComments (Ogg and FLAC) and Musepack (mpc) tags.

click on the F button in the toolbar, the file will be renamed according to the contents of its artist and track tags. In this way, you can turn **bachconcerto1.ogg** into something nicer like **Bach Concerto in G pt 1.ogg**. Excellently, you can do this for multiple files in the list by selecting them and clicking F – a great one-click way to clean up your music collection.

And that's just the start of it. You can create user-defined actions comprised of functions that sort values, merge fields, trim whitespace, convert case, and so much more. It's almost as flexible as using a scripting language, but with the convenience of a GUI, so if you have very specific requirements for your collection, *Puddletag* should handle them with aplomb.



Various functions are available to include in user-defined actions, such as regex-based text replacement.

**“Puddletag is almost as flexible as using a scripting language, but with the convenience of a GUI.”**

directory containing audio files, and they'll appear in the list on the right. Just like in a spreadsheet, you can now click into cells and edit data – that's the simplest way to do it.

Where *Puddletag* really shines, however, is in its automation facilities. Check out the drop-down list in the toolbar: there you can enter variables such as **%artist%** and **%album%**. You can use this to rename files according to tags, so say you have this in the drop-down list:

**%artist% - %track%**  
If you now select an audio file and

**PROJECT WEBSITE**  
<http://puddletag.sourceforge.net>

Operating system

# PC-BSD 10

FreeBSD is a fine server operating system, sharing many of the same qualities that Linux has: it's open source, it's reliable, it's secure and it can run thousands of FOSS programs. Some people use it as a desktop OS, but it's not the best experience out of the box – quite a lot of manual work is required to get everything set up properly. That's not a criticism, as FreeBSD just provides a base system and expects you to know what you're doing, like in Arch Linux. But if you want to try something more newbie-friendly, PC-BSD, a desktop-oriented OS based on FreeBSD, is worth a look.

Version 10.0 is available as a 3.4GB ISO which you can burn to a DVD-R. For testing purposes, though, it's much easier to boot it up in *VirtualBox*. The minimum system requirements are 1GB of RAM and 20GB of hard drive space, but the PC-BSD team recommends 4GB and 50GB respectively. This might seem excessive, given that the FreeBSD core is rather svelte, but the choice of desktop environments and supplied apps makes it a beefy package.

After booting, you're given default settings for installation, including a KDE 4.12 desktop. You can change this to Gnome, Mate, *Xfce* or some lighter window managers. PC-BSD

can take over the whole hard drive automatically; custom partitioning is also available. One reboot later and you're prompted to create a root password and normal user account, before landing at your chosen desktop.

## A bundle of joy

One of PC-BSD's most notable features is its PBI packaging system. This aims to make software installation more Windows or Mac OS X-like, in that users can download single **.pbi** packages, double-click, and get a new program. All dependencies are bundled into the package, and it's installed in **/Programs**. This is in contrast to Linux, where dependencies are used more extensively and programs are scattered around the filesystem. Is PC-BSD's approach better? Well, it's certainly easier when it comes to grabbing new apps from the web, but there's a lot of duplication. If a security hole is discovered in a widely-used library, every PBI using that library has to be updated – this is more time-consuming than the shared library approach.

**“PC-BSD's packaging system aims to make software installation more Windows- or Mac OS X-like.”**

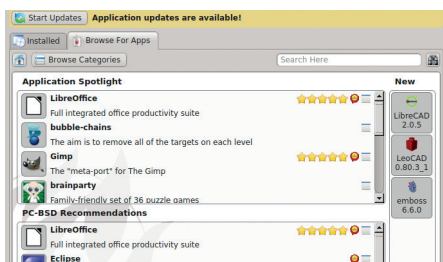


PC-BSD takes the solid server foundations of FreeBSD and adds a user-friendly desktop and packaging system.

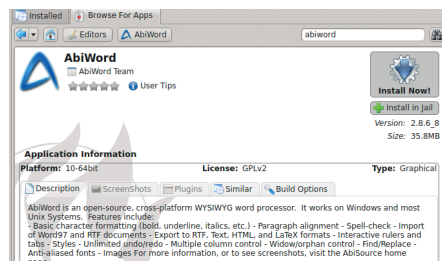
So, why else would you consider using PC-BSD rather than Linux? There's not a lot between them, although hardware support tends to be broader in Linux. Pretty much every major FOSS desktop app is available in PC-BSD – *LibreOffice*, *Firefox*, *Gimp* and so forth. There's also the usual gamut of development tools. But PC-BSD has a few aces up its sleeve as well, inherited from FreeBSD, such as excellent ZFS support. ZFS is a filesystem that features storage pools, snapshots, compression, corruption prevention and many other goodies. Meanwhile, FreeBSD jails are like chroot on steroids, and the licence makes it much easier to incorporate into proprietary software (if that's your wish).

PROJECT WEBSITE  
[www.pc-bsd.org](http://www.pc-bsd.org)

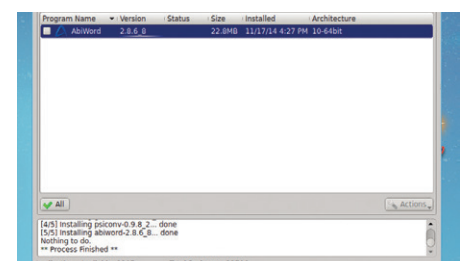
## How it works: Adding software



**1 AppCafe**  
Click the *AppCafe* icon on the desktop, and enter your password when prompted. You'll see a cluttered window – resize it to make more space.



**2 Search**  
*AppCafe* shows recommended applications by default; use the search bar to find a specific application. Click its name in the list up details.



**3 Install**  
Click *Install Now*, or *Install In Jail* to set it up in a restricted environment. The latter option is recommended for untrusted programs.

Video/audio transcoder

# Transmageddon 1.5

**W**e're pretty hardcore geeks at Linux Voice. But one thing has always terrified us: *Mencoder*. This tool, part of *MPlayer*, is tremendously powerful when it comes to converting media from one format to another. But it's insanely complicated – the man page alone contains almost 45,000 words!

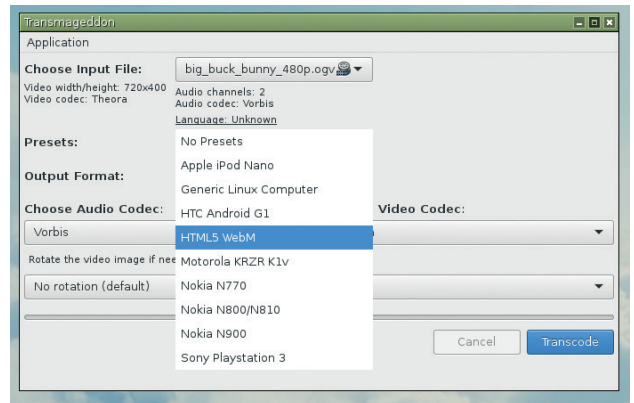
So we're always on the lookout for graphical tools that make the job a lot simpler. *Handbrake* and *VLC* work well here, but *Transmageddon* is an excellent lightweight alternative. It's built on Gnome libraries – although it ran without any problems on our minimalist *Openbox* installation – and uses *GStreamer* to handle media codecs. So if you want to transcode into the widest possible range of formats, install the "bad" and "ugly" *GStreamer* codec packages.

Bizarrely, *Transmageddon* doesn't open a file if you specify it at the command line. We're not sure if this is just a simple oversight, but anyway: you can choose the input file once the GUI appears.

*Transmageddon* will identify the audio and/or video codecs used by the file, and provide some extra information such as the resolution and number of audio channels. You can then choose the container for the output format, such as Ogg, Matroska, AVI, FLV, WebM and others, and then specify the audio and video codecs to be used inside the container.

### Hurrah for codecs!

So far so good. But what makes *Transmageddon* especially useful is its in-built profiles, which let you convert for specific devices and platforms without having to know



*Transmageddon* has a limited range of options, but it's great for quick converting tasks.

the details about them. Under the Presets drop-down menu, for instance, you can choose from various mobile phones, the Sony PS3, a "generic Linux" profile (Ogg Vorbis and Theora), HTML 5 video, and more. These profiles are stored in XML format in `/usr/share/transmageddon/profiles` – you can easily edit them to create your own for other devices.

**PROJECT WEBSITE**  
[www.linuxrising.org](http://www.linuxrising.org)

Drop-down terminal

# Guake 0.5.1

**I**f you do a lot of work in the terminal, you probably have several terminal windows at any one time, and can easily identify them in your taskbar or window list. Alternatively, you might use something like *tmux* to switch between your command-line apps inside a single terminal window. Either way, this makes sense when you have various regularly-used programs or shell prompts running all the time.

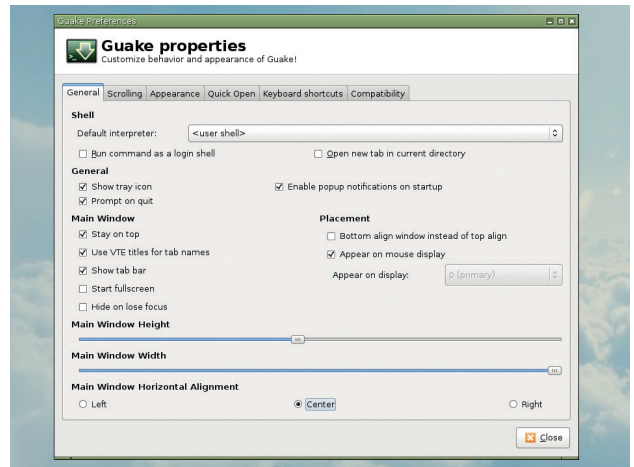
But what if you quickly need to enter a command, and don't want to open yet another terminal window? *Guake* is the answer. It's an ultra-fast terminal that pops down from the top of the screen when you hit a certain key (by default, bound to F12).

*Guake* has been doing the rounds for a while, but as a Gnome app it's

usually not well known by users of other desktops. We gave it a try on a plain *Openbox* setup, and were pleased to see that it works excellently. It's also attractive, picking up on the desktop wallpaper and showing it in the background of the terminal window, slightly darker to make the white terminal text stand out.

By default, *Guake* takes up the top half of the screen when you hit F12, but you can change this in the settings. Right-click on the *Guake* notification area icon and choose Preferences; then note the Main Window Height option. You can also make the terminal narrower, and determine where it appears.

Under the Appearance tab it's possible to change the font and transparency effect, while the Keyboard Shortcuts tab lets you



Don't like the defaults? Change *Guake's* appearance and keyboard shortcuts with just a few clicks.

redefine the key that's used to open the terminal. *Guake* does everything it's supposed to: it's fast, it's easy to configure, and it's perfect for those times when you need to run a quick command without adding yet more windows to your already busy desktop.

**PROJECT WEBSITE**  
<https://github.com/Guake/guake>

## Tiling terminal window manager

## Dvtm 0.13

There's a lot of talk 't the moment about the "Unix philosophy". In general, most people agree that this means small programs, with single objectives, that can be fitted together (eg with pipes or redirection) to solve larger tasks. This philosophy inspired the developer of *Dvtm*. It lets you split your terminal window up into tiles, with individual command line sessions inside them. But that's all it does; it stays away from session management, for instance, which is provided in a separate program (*Abduco*).

But what's the benefit of tiling? Well, tiling window managers, such as *i3*, are becoming popular on Linux desktops, especially for power users. They maximise screen space usage and let you switch layouts quickly, so you can

can have the main part of your screen devoted to, say, *Firefox*, with various terminals around it showing docs, server stats and so forth.

When you start *Dvtm*, you'll be greeted by just another terminal prompt. Hit Ctrl+G followed by C, however, and you'll see the terminal splits into two windows, one left and one right. Hit Ctrl+G followed by C once more, and the right-hand pane will be split into two. These are the basics of tiling. Ctrl+G in *Dvtm* is known as the "mod" key – it's the key combo that you press before doing any other action.

For instance, press mod followed by Space, and *Dvtm* will shift to a different layout, separating the

**"Dvtm lets you split your terminal window up into tiles."**

```

[1][2][a][4][5][
[0]
top - 15:05:07 up 32 min, 1 user, load average: 0.01, 0
Tasks: 78 total, 1 running, 77 sleeping, 0 stopped,
Mem: 0.0/0.0
Mem: 6.6/3.965
Mem Swap: 0.0/3.724

PID USER PR NO NI B RES CPU MEM%
1 root 20 0 22.2m 4.2m 0.0 0.1
108 root 20 0 57.7m 10.7m 0.0 0.3
130 root 20 0 51.5m 3.0m 0.0 0.1
156 root 20 0 14.0m 2.4m 0.0 0.1
157 obus 20 0 26.4m 2.0m 0.0 0.1
162 root 20 0 69.9m 3.4m 0.0 0.1
223 mke 20 0 13.2m 3.1m 0.0 0.1
252 mke 20 0 15.5m 1.8m 0.0 0.0
268 root 19 -1 865.5m 43.7m 0.0 1.1
256 mke 20 0 19.2m 3.1m 0.0 0.1
265 mke 20 0 171.2m 14.7m 0.0 0.4
470 mke 20 0 252.7m 19.1m 0.0 0.5
471 mke 20 0 8.2m 1.5m 0.0 0.0
472 mke 20 0 15.4m 3.7m 0.0 0.1
530 mke 20 0 15.3m 5.6m 0.0 0.1
540 mke 20 0 15.4m 3.7m 0.0 0.1
541 mke 20 0 15.4m 3.6m 0.0 0.1
550 mke 20 0 17.0m 3.4m 0.0 0.1
559 mke 20 0 12.7m 2.3m 0.0 0.1
542 mke 20 0 15.4m 3.6m 0.0 0.1
543 mke 20 0 33.6m 3.2m 0.0 0.1
219 mke 20 0 26.1m 2.5m 0.0 0.1
221 mke 20 0 71.9m 1.7m 0.0 0.0
250 mke 20 0 15.6m 1.7m 0.0 0.0
261 mke 20 0 26.2m 2.2m 0.0 0.1
272 mke 20 0 13.2m 2.0m 0.0 0.1
218 mke 20 0 165.5m 11.9m 0.0 0.2
277 mke 20 0 13.2m 2.0m 0.0 0.1
360 mke 20 0 157.7m 12.8m 0.0 0.3
293 mke 20 0 106.0m 4.2m 0.0 0.1
304 mke 20 0 107.9m 3.6m 0.0 0.1
310 mke 20 0 103.9m 3.3m 0.0 0.1
314 mke 20 0 104.6m 3.2m 0.0 0.1

df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1       25G   2.0G  22G   8% /
dev             2.0G   0 2.0G   0% /dev
run             2.0G   0 2.0G   0% /dev/shm
tmpfs          2.0G   0 2.0G   0% /sys/fs/cgroup
tmpfs          2.0G  24K 2.0G   1% /tmp
tmpfs          396M  4.0K 396M   1% /run/user/1000
mike@archbang ~$

Dvtm is a dynamic tiling manager for the console. As a console window manager it tries to make it easy to work with multiple console based applications.

OPTIONS
-- Print version information to standard output and exit.
-M Toggle default mouse grabbing upon startup. Use this to allow normal mouse operation under X.
-m modifier set command modifier at runtime.
-d delay

mike@archbang ~$

```

Don't overload your desktop with terminal windows – use *Dvtm* and run multiple programs in a single one.

screen vertically with two windows at the top and one at the bottom. Do mod and Space again for yet another layout. It's awesome: with a full-screen terminal window, you can create some very useful layouts without having to manually resize anything.

PROJECT WEBSITE  
[www.brain-dump.org/projects/dvtm/](http://www.brain-dump.org/projects/dvtm/)

## Multi-system emulator

## Mednafen 0.9.33

*Mednafen* is the mother of all emulators. It can play games from a large range of systems, which might leave you thinking: what's the point? There are already good SNES, Mega Drive, Game Boy etc. emulators out there, so why do we need a single program to emulate them all? Well, if you're a retro gaming fan who likes to play games from a range of systems, life is much easier when you only have to configure one emulator. Set up *Mednafen* with the exact graphics, sound and input options that you want, and you can then play all your games without learning a load of different tools.

*Mednafen* is included in many distro repositories, or if you're compiling from source code, the main dependencies are *SDL*, *libcdio* and *libsndfile*. And there's

something worth noting about *SDL*: when we first tried the emulator, we didn't get any sound output. To fix it, we had to edit `~/mednafen/mednafen-09x.cfg` and change the `sound.driver` line from `default` to `sdl`. If you're running an Ubuntu-based distro and have the same problem, try that fix.

## A cornucopia of consoles

*Mednafen* emulates (deep breath): Game Boy, Game Gear, NES, Master System, Super NES, Mega Drive (aka Genesis), Virtual Boy, Atari Lynx and some fairly obscure handhelds like the Neo Geo Pocket and WonderSwan. The emulation was fast and glitch-free with the games we tried. *Mednafen* supports saved states, real-time game rewinding, screenshot-taking and even the ability to make videos in various



*Mednafen* handles Mode 7 games like *Super Mario Kart* (the best version, we think) wonderfully.

formats. It can take a while to set up, but the configuration file is well documented and when it's working properly, it's one of the best multi-system emulators we've come across.

PROJECT WEBSITE  
<http://mednafen.sourceforge.net>

Android open source package manager

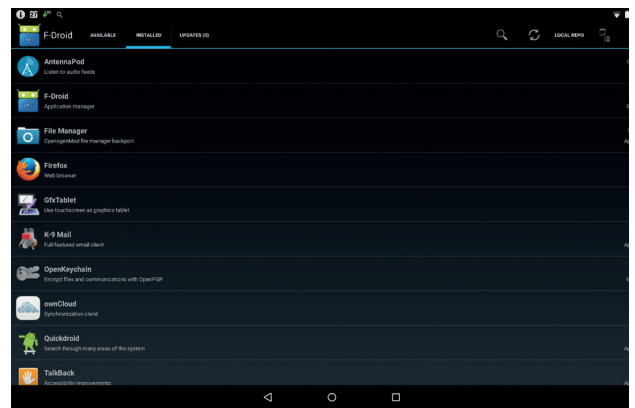
# F-Droid 0.76

**W**e thought we'd take a short break from regular service and look at a couple of Android 'apps'. "Wooah..." you might say, "Aren't the majority of apps on Android proprietary?" Yes, this is true. But open source applications are being released for Android too, and what's better is that you don't need Google's Play store to use them.

*F-Droid* is an open source package manager for Android, and it's the only package you'll need to install manually. You can do this by first enabling 'Unknown sources' in the Security panel of your device, and then by downloading the *F-Droid* apk package onto your Android device using a web browser. You can also open a QR code from a desktop browser. Android should ask whether you want this package installed

automatically, and you should soon find *F-Droid*'s copyleft-inspired icon lurking in your launcher.

As with any other Linux package manager, you need to first update the database of files held in the repositories. You can do this from the drop-down menu, and with the database populated, you'll see a list of packages under the 'Available' heading. Any of these can now be installed first by tapping on the package, then by clicking on the small 'plus' symbol at the top of the screen. If you run *F-Droid* periodically, it will check for package updates too, and these can be installed automatically. A new feature in the latest version is the



When Google Play isn't available, such as on Cyanogenmod, *F-Droid* becomes absolutely essential.

ability to share your local package cache as a local repository.

Tap on 'Local Repo' and a QR Code appears. Tap on the 'Turn on' button and then anyone who scans the QR Code with their Android device can install packages directly off your own Android device. *F-Droid* isn't as aesthetically pleasing as Google Play, but it's definitely more ethically pleasing.

PROJECT WEBSITE  
<https://f-droid.org>

**"F-Droid is an open source package manager for Android."**

Android open source email client

# K-9 Mail 5.001

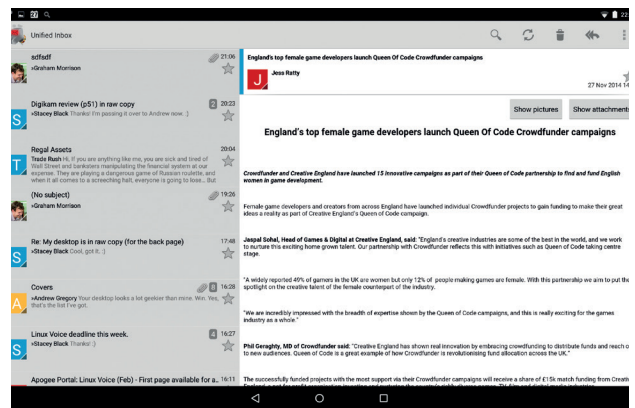
**T**his is another Android application, and perhaps not coincidentally, it can be installed easily via *F-Droid* (although you can also install the same application through Google Play). It's open source and being developed just like any other open source application. Tim Bray, who we interviewed last month, is even one of its contributors.

Put simply, *K-9 Mail* is utterly brilliant. It's like installing an old desktop email client onto your mobile device. Only better. You can easily manage more than one account from a single app, even migrating Gmail if you don't like its new responsive and configuration-free interface. Messages are still threaded and sorted and placed in a unified inbox, if you want them to be. IMAP-pushed notifications are

also powerful. We have our work email flashing the LED the colour of Linux Voice red, and you can change the sounds too.

Perhaps most importantly, *K-9 Mail* can integrate with OpenKeyChain (and APG), also available from *F-Droid*. OpenKeyChain enables you to manage your public and private keys, as well as import keys for your contacts and add and decode encrypted emails. We'd love to see PGP/MIME integration for convenience, but we know of no other Android solution that can offer the same level of security.

Outside of our encryption paranoia, *K-9* still excels, whether you're searching for emails, adding different signatures, having multiple identities, themeing, attachment saving and even keyboard



Tim Bray, who we interviewed last month, is helping develop part of *K-9 Mail* on Android.

shortcuts. It doesn't look quite as modern as the new Gmail, and attachments aren't quite so well integrated, but it can be made to look lovely

This is an app that turns your mobile or tablet from a convenience into a productivity tool. We only wish we could banish email from our lives and forget *K-9* is so good.

PROJECT WEBSITE  
<http://k9mail.org>



## FOSSPICKS Brain Relaxers

Transport simulator

# Simutrans 120

Ever since the glory days of *Sim City* on the Amiga we've really enjoyed playing sim-like games. There's something incredibly satisfying about creating your own little world, trying to make everybody happy, and then trashing the whole place if it doesn't work (or you just get bored and need some cathartic release via mindless destruction).

*Simutrans* is a transport simulator, so when you start it, the environment is populated with towns and villages. Your goal is to connect these locations with infrastructure such as roads and railways, helping people and products to move between the settlements and build a thriving economy. You have to monitor

your cash flow, investing in infrastructure for longer gains down the road (no pun intended).

## Telegraph road

When creating a new game, you can define the size of the playing area, and how many settlements it has. Then use the toolbar icons at the top to start building: if you hover the mouse over the icons, you'll see different prices for various road types. So if your money (shown at the bottom) is running low, you can opt for cheaper dirt tracks, for instance. To create a road or railway, click and drag between two points, and *Simutrans* will create the shortest path.

It's not the easiest game to get into, and you'll have to spend some time in the documentation to really



*Simutrans's* pixel artwork is nicely detailed, but hard to see on high-res monitors. You can zoom in though.

get the most of it, and the teeny-tiny icons can be fiddly to work with. But there's a busy online community supporting the game, so it's easy to get help.

PROJECT WEBSITE  
[www.simutrans.com](http://www.simutrans.com)

Action game

# Koules 1.4

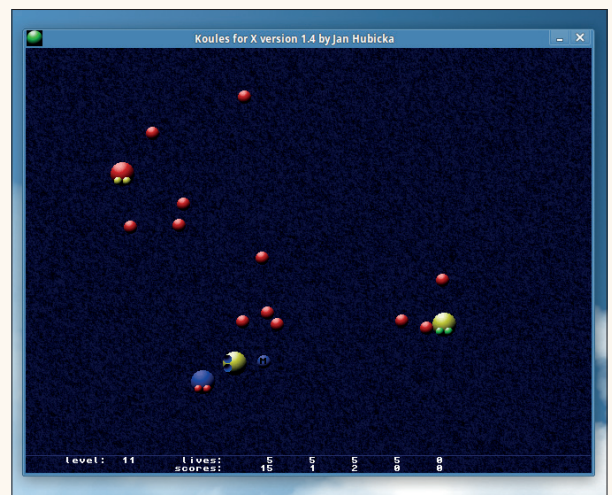
Prepare yourself: this game is hardly cutting edge or fresh by any definition. Indeed, it hasn't been updated since 1998. So why are we covering it here? Well, for three reasons: this author had a dream about it recently, an actual dream involving installing software. Secondly, the game can still be found in many distro repositories. And third, it's still a classic!

If you're running Debian, Ubuntu or Fedora, you should be able to install *Koules* from the repos like any regular program. In Arch Linux it's available in the AUR, as opposed to the normal repositories.


Anyway, once you've started *Koules*, keep tapping Enter to skip

past the *Star Wars*-esque intro text. At the main menu, select Start Game and hit Enter. Your character is the yellow ball with blue eyes, and you'll notice a number of small balls moving in your direction and bouncing off one another. Your job is to move your player with the cursor keys and bash the small balls onto the edge of the screen to destroy them.

This might sound easy, but it gets very tricky, very quickly. With each level more and more balls appear and try to bash you onto the edges, in which case you lose a life. Some balls don't immediately explode on contact with the edges, but turn into powerups giving you extra speed and mass. The game is massively addictive and makes you



A multiplayer mode is available, and there's even an Android port on the Play Store.

laugh at times, given the craziness of the gameplay – so we're glad to see that it still runs on modern Linux distros today. 

PROJECT WEBSITE  
[www.ucw.cz/~hubicka/koules/](http://www.ucw.cz/~hubicka/koules/)  
English



## **FORTHCOMING EVENTS**

**un-conference - London - Saturday 7th February 2015**

**London Hackspace, 447 Hackney Road, London E2 9DY**

**Our 5th un-conference - on-line booking opening soon.**

**see: <http://www.flossuk.org/Events/Unconference2015>**

**DEVOPS Spring 2015 - 24th, 25th & 26th March 2015**

**The Hilton York, 1 Towers Street, York YO1 9WD**

**Tuesday 24th March - Workshops • Wednesday 25th & Thursday 26th - Conference**

This event is the UK's only conference aimed specifically at systems and network administrators. It always attracts a large number of professionals from sites of all shapes and sizes. As well as technical talks, the conference provides a friendly environment for delegates to meet, learn, and enjoy lively debate on a host of talks.

**Bookings will open in December, for more information see: <http://www.flossuk.org/Events/Spring2015>**

**OpenTech 2015 - Saturday 13th June, ULU, University of London Union**

**OpenTech will be 10 years old in 2015.**

It will be back in 2015, for the usual mix of technology, experience and everything else.

For now, there's a date for your diary, and the call for talks will open soon. If there's something you'd like to hear about at OpenTech next year, we'd love to hear from you and we'll see what we can do.

OpenTech is only possible because of wonderful and generous sponsors in the past.... If you or your company is interested in Sponsoring please get in touch - [opentech@opentech.org.uk](mailto:opentech@opentech.org.uk)

The event's predecessors were low cost, one-day conferences about technologies that anyone can have a go at, from 'Open Source' style ways of working to repurposing everyday electronics hardware.

**<http://www.opentech.org.uk>**

**Dynamic Languages Conference - Saturday 20th June 2015 Manchester**

**for more information see: <http://www.dynamiclanguages.co.uk>**

If you are not a member and want to be kept informed of future events please subscribe to 'announce@ukuug.org' see: <http://lists.ukuug.org/mailman/listinfo/announce>

UKUUG - Silver Sponsor  
members include:



Why not join today? Annual Individual membership is just £42.00 inc. VAT. See: <http://www.flossuk.org/join>

As a member you can attend all our events at the specially discounted member rates and receive our quarterly Newsletter!

**UKUUG Ltd. t/a FLOSS UK, The Manor House, Buntingford, Herts SG9 9AB**

**Tel: 01763 273475 Email: [office@ukuug.org](mailto:office@ukuug.org)**

# TUTORIALS

Dip your toe into a pool full of Linux knowledge with ten tutorials lovingly crafted to expand your Linux consciousness



**Ben Everard**

is looking for some free software that stops the rain falling like tears in the rain.

**W**e've pledged to release each issue of Linux Voice under a creative commons licence within nine months. As we put issue 11 together, this deadline passed for issue 1, and so we released it under CC-BY-SA 3.0. The response has been pretty overwhelming: enough to crash our server, and the content is now also shared on BitTorrent and the Internet Archive as well as our website, which dramatically increases the potential readership we'll reach.

There remains a healthy skepticism within the media business about whether you can build a magazine by giving content away like this. As far as we are aware, we're the first company to try it, and since this is the first issue we've released, it's far too early to tell if it's a daring move that will push media into the modern age, or a poor business decision made by starry-eyed idealists. Naturally, we think it's the former, but it will be some time before we know.

However, even if it doesn't turn out to be the best business decision we've made, we still stand firm on the principal that it's the right thing to do. We need to balance our ability to make a living against the desire to support the community that makes the magazine possible, and we think this is the right way to do it.

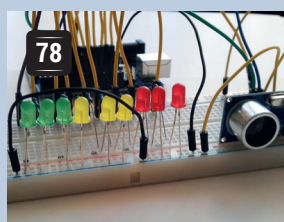
[ben@linuxvoice.com](mailto:ben@linuxvoice.com)

## In this issue...



### Déjà Dup

Like death and taxes, data loss comes to us all – but not if you follow **Mayank Sharma's** advice and use this simple backup tool.



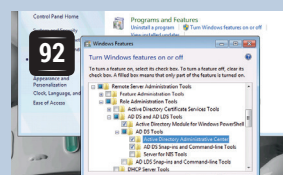
### Arduino sensors

**Les Pounder** uses an ultrasonic distance sensor and an Arduino microcontroller to replace his tape measure.



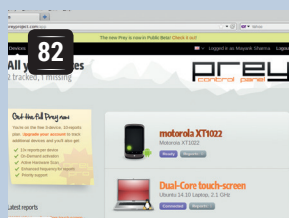
### Wine

Run Windows software the **Mike Saunders** way.



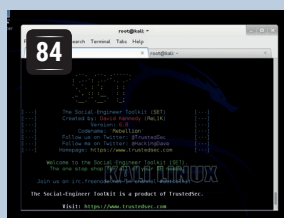
### Samba

**John Lane** shows how to share files with Windows.



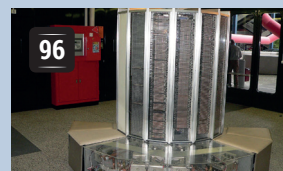
### Prey

Don't let thieves get away! **Mayank Shama** shows how to use *Prey* to track and locate your devices should they go missing.



### SEToolkit

Social engineering is one of the most effective ways of beating computer security. **Ben Everard** harvests credentials with *Set*.



### Seymour Cray

**Juliet Kemp** uncovers the man behind the computer.

## PROGRAMMING

### Firefox addons

**100** *Firefox* is good, but it's not perfect. Fortunately, it includes a mechanism for you to make it so: addons. These are written in JavaScript and can control much of the behaviour of the browser, so whatever feature you want to have, don't wait for someone else – get started with addons and create it today.

### NoSQL

**104** Money doesn't make the world go around any more, data does. So many ones and zeros are being collected now that traditional approaches to data management can crumble under the load. Big Data needs a new tools and techniques, and NoSQL databases have risen to the challenge.

### GPGPU

**106** Graphics cards are actually incredibly powerful processing units capable of much more than just drawing scenes or playing games. They can solve some problems far faster than most CPUs using General Purpose Programming on GPUs (GPGPU). Don't let your GPU be lazy; put its processing power to use today!

# DÉJÀ DUP: BACKUP FOR EVERYONE

MAYANK SHARMA

Get acquainted with the easiest backup tool on the planet to help you save yourself from the inevitable data apocalypse.

## WHY DO THIS?

- You'd really risk losing data?
- Quick to set up and easy to use.
- Designed for the everyday desktop user.

We care about you. No, we really do. Which is why you should believe us when we say that sooner or later you will lose valuable data. You can spend a fortune on a storage medium that's anti-scratch, dust-resistant, heat-proof and contains no moving parts, but what you really need to do is to invest some effort in backing up your data.

Although it isn't particularly time consuming, backing up data requires careful thought and preparation, and involves more than just zipping files into a tarball. Unfortunately this means it's often neglected. This is where *Déjà Dup* comes into play. It's different from the plethora of other tools in that it has a minimal interface so as to not overwhelm new users. But it's based on the powerful *Duplicity* command line backup tool and provides just the right number of features for desktop users who aren't used to the ways of a backup tool. On some distros, such as Ubuntu, *Déjà Dup* ships pre-installed, while it's available in the official repos of most others. You can configure the software in a matter of minutes without delving into lengthy documentation.

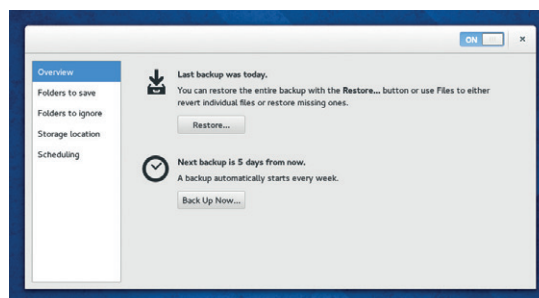
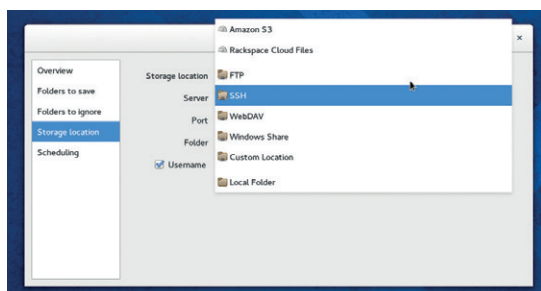
## Date with data

The first time you fire up the tool, it lands on the Overview page which, as expected, tells you that there are no recent backups nor are there any scheduled. Before it can save you from data doomsday, you'll first have to set it up.

Navigate to the Folders To Save section, which by default lists your Home directory. This is generally a safe bet for most users. If you know what you are doing you can also remove this location and add any particular directories that you wish to back up. You can also use the + button to add other folders on other mounted drives or network shares.

Then switch to Folders To Ignore and specify folders you don't need to back up. By separating the directories to include and exclude in the backup,

Use the Custom Location option to specify a remote location supported by the Gnome Virtual File System (GVFS).



Although it's changed somewhat, the main interface of the app still essentially contains only two buttons – to back up and restore data.

*Déjà Dup* gives you the flexibility to include a large directory – for instance `/home` – in your backup, while specifying parts to leave out, such as `.cache/`.

To help you get started this section already lists the **Trash** and **Downloads** folders, though many users might want to remove the latter from the list unless you really don't want to safeguard the contents of the **Downloads** folder.

## Prepare for a backup

While *Déjà Dup* takes the pain out of setting up the actual data backup process, a crucial part of the process is preparing for it which involves careful consideration. For starters, you need to decide where you want to store your data. Keeping it on another partition of the same disk isn't advisable, since the whole disk might fail and render the backup copy useless.

One solution is to keep the backup on another disk. If you have multiple disks and a spare computer you can even set up your own Network Attached Storage device using software like Open Media Vault (instructions in our tutorial in LV009). To protect your data against physical disasters, such as fires, floods and theft, make sure you keep the backup as far away from the original as possible, perhaps on a cheap cloud storage service. Each method has its advantages – hard disks are cheap and readily available while removable disks offer portability, and online storage is globally accessible. The kind of data you wish to back up also influences the choice of storage medium. A DVD might be useful for holiday snapshots, but isn't going to be much use to a professional photographer.

You'll also need to work out the appropriate backup methodology. Do you want to back up manually or automatically based on a schedule? The correct backup frequency varies based on the kind and value of data being safeguarded. Depending on the size of the files, it might not be a good idea to back them up completely everyday either.

## What to back up?

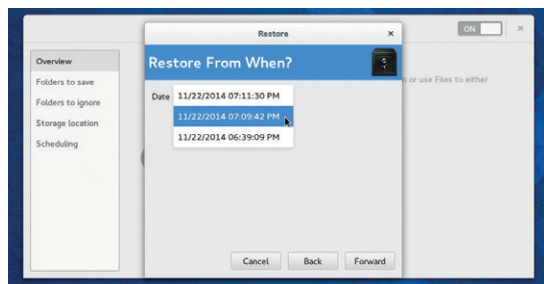
Do you really need to backup your entire home directory, or is just an overkill? Here's what you need to know while selecting directories for backup. Most modern distros keep the files you've created or downloaded under directories such as `~/Documents`, `~/Downloads`, and `~/Desktop`, so you'll want to include them in your backups. Also don't forget to check `/home` for any important documents. Some apps, such as email clients, also keep your downloaded emails, attachments and address books under hidden directories beneath your home folder, so make sure you include them as well.

Then there are apps that create their own directories to store files. Most prompt you for the location, while some may create them on their own during installation. Make sure you check for and include such directories which are usually listed under the Preferences section of the apps. Be vigilant, though. Some of these directories contain cache directories, which needlessly add to the backup's size. Finally, if there's a piece of software that's crucial to you and you don't want to spend time downloading it again, back it up by saving the cache directory for your distro's package management system.

Next, move on to Storage location and use the Drop-down list to pick one of the supported locations. This can be a local hard disk, a remote location that you connect to via FTP or SSH, or a cloud storage service like Amazon's S3. Depending on the storage location you select, the app will ask you for further details. For example, if you select the FTP option, you'll be asked to provide the IP address of the FTP server along with the authentication information and the location of folder where you want to store the backups. To save the backups to Amazon's S3 it'll need your S3 Access Key ID and for the Rackspace Cloud Files service it'll need your username.

Finally, you can switch to the Scheduling section and select a policy for keeping old backups. By default, old backups will be kept until the target storage location runs out of space, but you can also specify a different time period depending on the importance of the data. Before you can set a schedule for the backup you'll have to activate the app by toggling the button at the top right-corner of the window to On. Once the configured backup is enabled, you can use the pull-down list on the Scheduling section to either run this backup every day or every week, which is the default option.

To create the initial backup, switch to the Overview section and click on the Back Up Now button. The tool will provide a summary list of the directories involved and will begin. While creating the backup, the app will ask you to optionally encrypt the backup. You can enter a password in the space provided or choose to back up the files without a password. This initial backup may take some time, but subsequent backups are much faster because they are incremental and only back up data that has changed.



Despite being a simple app, *Déjà Dup* offers advanced features like incremental backups and stores multiple time-stamped versions of backups.


Once the backup has been created, the Overview window will inform you when the last backup was taken and when the next one is scheduled.

## Déjà Vu

To restore files from the backup, launch the app and click on the Restore button in the Overview section. The app will launch the Restore wizard, which will first prompt you for the location of the backed up files. Just like before, select the remote location where you've backed up the files and enter any associated information such as the IP address and your login credentials.

*Déjà Dup* will then scan the remote location and in the next section it'll display a time-stamped list of all the backups. Select the one you wish to restore from and move on to the next step. The app will now give you the option to either restore the backed up files to their original location or into a specified folder. Before restoring the files, the app will prompt you for the password if the backed up files were password protected.

One of the best features of the app is its ability to restore individual files as well. To do this, head to the folder from which you have accidentally lost files. Right-click inside the folder and select the Restore Missing Files option from the context-menu. The app will scan the folder against the most recent backup of this folder and display a list of files that are in the backup but currently missing from the folder. Now use the checkbox besides the listed files to select the ones that you wish to restore and the app will restore their latest versions.

It's worth noting that *Déjà Dup* is missing some of the flexibility you'd get with other backup tools. One such missing feature is the ability to create backup sets to backup different files into different locations. *Déjà Dup*, instead, is designed to back up the specified folders into the specified destination, each and every time you schedule it to run. *Déjà Dup* isn't meant for use in a complex environment like an enterprise, but is perfect for safeguarding data for home and SOHO users and also gives you the flexibility to restore individual files from the backups with ease. 

Mayank Sharma has been tinkering with Linux since the 90s and contributes to a variety of techie publications.



# ARDUINO: BUILD A PROXIMITY SENSOR

**LES POUNDER**

If you have trouble reverse-parking in the tight terraced streets of northern England, why not build one of these?

**WHY DO THIS?**

- The Arduino is a great platform for experimentation and in this tutorial we will build a device that can react to our proximity.

**TOOLS REQUIRED**

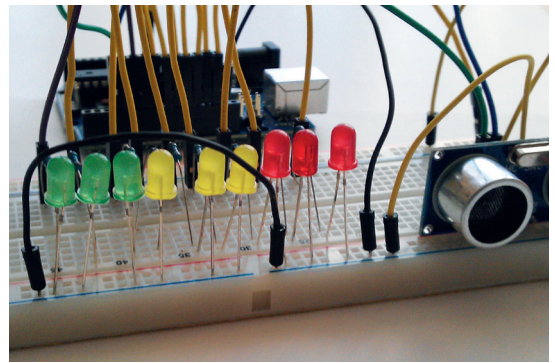
- An Arduino – ideally an Uno, but the code will work on most types, such as the Leonardo.
- A large breadboard.
- An HC-SR04 ultrasonic sensor.
- 9 x 220Ω resistors (Colour code red red brown gold).
- 3 x red LED.
- 3 x yellow LED.
- 3 x green LED.
- Male-to-male jumper cables.

In the world of hobbyist electronics there are two big names: the Raspberry Pi and the Arduino. And while the Raspberry Pi has the largest share of the spotlight we should not forget the Arduino, which was the board to launch the Internet of Things back in the mid 2000s.

The Arduino is a small microcontroller board created in Italy to enable a low-cost method for artists to use electronics. The Arduino comes as a hardware platform with an accompanying software application used to program the board. Thanks to the Arduino there has been a big change to the programming landscape, with the barrier to physical computing projects such as home automation and robotics being broken down by this cheap and flexible platform.

The Arduino platform encompasses a multitude of boards. Boards such as the Leonardo, Galileo and the Mega are available to purchase, but the most common board used by the majority of new hackers is the Uno. The Uno is a remarkable board that hits the sweet spot between functionality and price. It comes with 14 digital input/output pins and six analogue inputs all connected to a microcontroller in the form of the ATMEL ATmega328, which is programmed via a USB connection to your computer.

In this project we will be using an Arduino UNO R2, which is fully compatible with the newer Uno R3. Your Uno should also have a USB lead to connect to your computer. Insert one end into the Arduino and the



Build your own distance sensor using just a few cheap components and a little bit of code.

other into your computer; can you see an LED labelled L13 on your board? This should be blinking, which is a test preloaded into every board to demonstrate that it is working.

**Installing the Arduino software**

The most convenient way to install the Arduino software is via the package manager for your distribution. For Debian- and Ubuntu-based systems type the following into a terminal, followed by the Enter key:

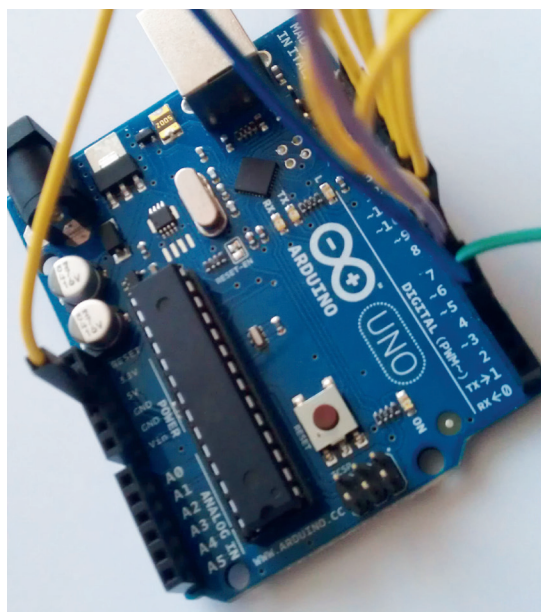
```
sudo apt-get install arduino
```

With the Arduino application installed we can now run it from the menu.

When run for the first time the Arduino application will ask for our current user to be added to a group called "dialout". This is important as only members of this group can access the Arduino hardware that is attached to the USB port. Add your user to the dialout group and then close down all of your open applications, including the Arduino software, and log out of the current session. This will ensure that your user is added to the correct group and that their privileges are reloaded at the start of the new session. Once you're logged in, open the Arduino application once more.

**Arduino coding 101**

The Arduino language is based upon a language called Processing, and is relatively easy to pick up especially if you have worked with languages such as Python before. Let's step through an example provided by the Arduino team and one that is preloaded onto every Arduino.



The Arduino Uno is a remarkable board and a great starting point for hardware hacking.

In this example an LED attached to pin 13 blinks in an infinite loop. The Arduino Uno comes with a built-in LED for pin 13, labelled L13 on your board:

```
int led = 13;

void setup() {
  pinMode(led, OUTPUT);
}

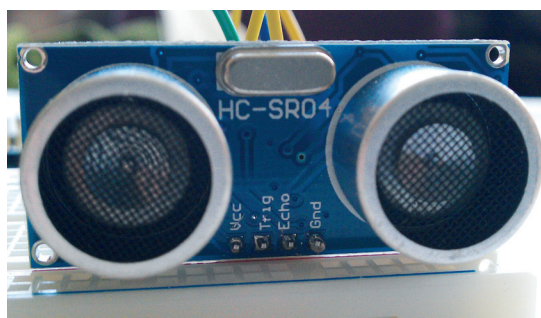
void loop() {
  digitalWrite(led, HIGH);
  delay(1000);
  digitalWrite(led, LOW);
  delay(1000);
}
```

We start by declaring a new variable, labelled as **led**, and in there we store the integer (int) value 13. With the variable created we now move to the setup section of our code and instruct the Arduino that the pin mode for our **led** pin is an output, which means that current will flow from the pin to the LED attached to it.

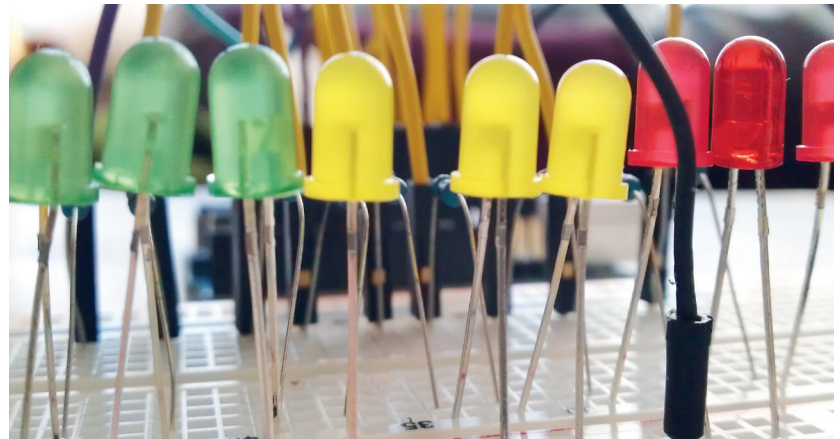
You can see that the code contained in **void setup()** and **void loop()** is contained inside **{}** brackets. Arduino uses these brackets to contain the code that belongs to that section, unlike Python, which uses indentation to denote what code belongs to what section. You will also notice that each line of code inside a section ends with a semi colon **;** this instructs the application that this line has been completed, and without them you will receive a compilation error.

Our focus shifts to the main body of code that will handle the blinking of our LED. This is contained in **loop()** and the blinking is achieved by sending power to the LED pin using **digitalWrite(led, HIGH);** We then delay the program by one second and then turn off the power to the LED pin **digitalWrite(led, LOW);** and finally we delay the program by a further one second, effectively causing the blink. This is then looped infinitely.

This small piece of code is considered the "Hello World" of the Arduino platform, demonstrating the functionality in a simple manner. For our project we're going to go much further, controlling nine LEDs via a novel input device.



Ultrasonic sensors work by firing a burst of ultrasound forwards. Any reflected ultrasound is bounced back to the sensor enabling our code to calculate the distance.



Our LEDs have their shortest leg in line with the ground rail of our breadboard and their longest leg in line with a resistor to lengthen their lifespan.

In our project we have nine LEDs attached to an Arduino via a series of wires and resistors that are connected via a breadboard. We have three green, yellow and red LEDs, which will provide a visual output when our ultrasonic sensor is triggered. An object 30 centimetres or further away will trigger the green LEDs, an object less than 30 centimetres but greater than 10 centimetres away will trigger the yellow LEDs. Finally an object less than 20cm away will trigger the red LEDs to illuminate warning us of a collision. This setup is very similar to a parking sensor.

So our logic would be as follows

```
Take a reading using the sensor.
If the distance is less than 10cm
  Illuminate red LEDs
Else if the distance is greater than 10cm but less than 30cm
  Illuminate the yellow LEDs
Else
  Illuminate the green LEDs.
```

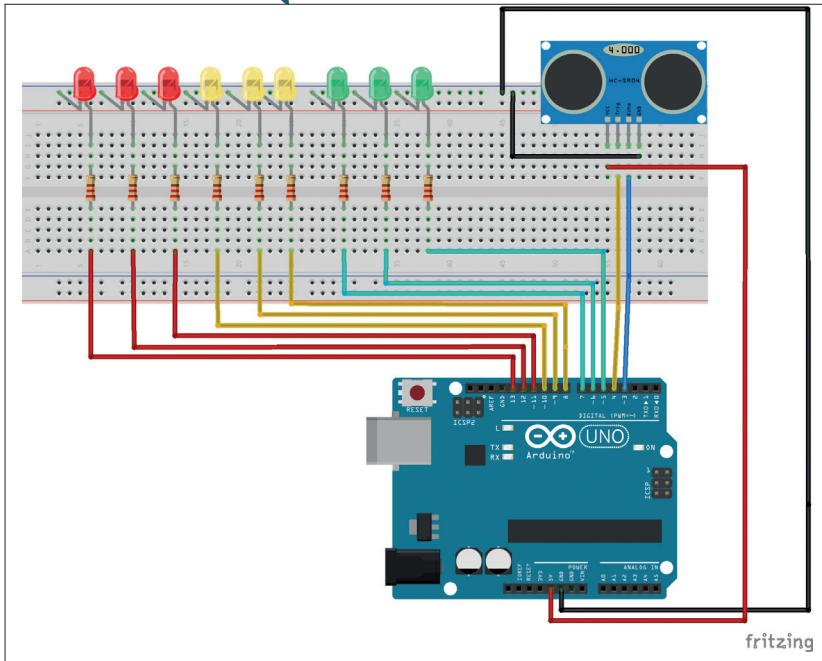
Let's step through the code to see how it works.

### Creating variables

Unlike Python, the Arduino language requires us to identify the type of data that is being stored within a variable. In this case we're storing the pin number for each of the components that are attached to the Arduino via a breadboard. This number is an integer, which is shortened to **int**. For this project we create 11 variables, nine that control each of the LEDs used and two that handle the sending and receiving of the ultrasonic pulse.

```
int trigPin = 4;
int echoPin = 3;
int red1 = 13;
int red2 = 12;
int red3 = 11;
int yel1 = 10;
int yel2 = 9;
int yel3 = 8;
int gre1 = 7;
int gre2 = 6;
int gre3 = 5;
```

In order to use the components connected to the Arduino, the Arduino has to be told that they are there, and to do that we need to provide instructions on



This diagram was created in *Fritzing*, a great tool to help you design the layout of a project. You can find a high resolution version in the repository for this project.

where they are connected and what they will do. This is achieved via the `void setup()` configuration section, and in here we use `pinMode` to instruct the Arduino on what each pin will do. We earlier created a series of variables that store the pin locations for each of the components. We will use those with `pinMode` to identify the pin that we wish to configure. The configuration is quite simple: is the pin an input or an output? An input will wait for a signal/current from an external component, while an output will send a signal/current to an external component.

```
void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(red1, OUTPUT);
  pinMode(red2, OUTPUT);
  pinMode(red3, OUTPUT);
  pinMode(yel1, OUTPUT);
  pinMode(yel2, OUTPUT);
  pinMode(yel3, OUTPUT);
  pinMode(gre1, OUTPUT);
  pinMode(gre2, OUTPUT);
  pinMode(gre3, OUTPUT);
}
```

The `void loop()` is our main body of code and contains the logic that controls the detection of an object by the ultrasonic sensor. We start the first part of this section by creating two variables, called `duration` and `distance`; these will contain long integers, so called because they have an extended size to store large numbers. We will use these variables to store the time taken for the pulse to be sent and received, and we shall use `distance` to store the answer to a calculation later in the code.

We next trigger a pulse to be sent from the ultrasonic sensor, but before we do that we must ensure that the ultrasonic sensor is not already transmitting. We do that using `digitalWrite`, which

instructs the `trigPin` to change its state from on to off (**HIGH** to **LOW**).

The code then instructs the project to wait for two microseconds, which is just enough time for the ultrasonic sensor to settle ready for use.

We now use the `digitalWrite` function to send a pulse from the sensor by setting the `trigPin` to **HIGH**, in other words sending current to the sensor. Current is sent to the ultrasonic sensor for 10 microseconds using the `delayMicroseconds()` function. We then turn off the current to the `trigPin`, ending the pulse transmission sequence.

Now we need to do a little maths. To kick things off we record the time taken for the pulse to be sent and received, and this is stored in the `duration` variable that we created earlier. Lastly we use the `distance` variable to store the answer to the calculation, `duration` divided by 2, as we only need to know how long it took for the pulse to be received. The answer is then divided by 29.1 to give us the distance in centimetres:

```
void loop() {
  long duration, distance;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = (duration/2) / 29.1;
```

For our last section of code we use the classic `if, else if, else` conditional statement to check for three different states. We'll start with the `if` statement.

The first condition that we wish to test is to check the distance between the sensor and any objects that might be in the way. At this time we're looking for objects less than 10 centimetres away, and if this condition is true we turn on the power to all of the red LEDs, and turn off the power to the yellow and green LED. This tells us that the object is really close, just like a parking sensor does in our cars:

```
if (distance < 10) {
  digitalWrite(red1,HIGH);
  digitalWrite(red2,HIGH);
  digitalWrite(red3,HIGH);
  digitalWrite(gre1,LOW);
  digitalWrite(gre2,LOW);
  digitalWrite(gre3,LOW);
  digitalWrite(yel1,LOW);
  digitalWrite(yel2,LOW);
  digitalWrite(yel3,LOW);
}
```

Our next condition to check uses an `else if` statement, and this means that if the first `if` statement is false, check to see if this `else if` statement is now true and if so run the code. So if the distance between our sensor and object is greater than 10 centimetres but less than 30 centimetres, the red and green LEDs are turned off, and the yellow LED are turned on, indicating that we are getting closer to the object:



```

else if (distance > 10 and distance < 30) {
digitalWrite(yel1,HIGH);
digitalWrite(yel2,HIGH);
digitalWrite(yel3,HIGH);
digitalWrite(ge1,LOW);
digitalWrite(ge2,LOW);
digitalWrite(ge3,LOW);
digitalWrite(red1,LOW);
digitalWrite(red2,LOW);
digitalWrite(red3,LOW);
}
    
```

Our last condition to test is rather simple, as it does not require anything to test. **else** is used when all other conditions have been tested and proven to be false. If everything is false then **else** must be true. So if the object is not less than 10 centimetres away, or further than 30 centimetres away then the red and yellow LED will be turned off and the green LED will be turned on, indicating that we are far enough away from the sensor. Our last line of code controls the speed of the project and introduces a half-second delay before the main loop is repeated once again:

```

else {
digitalWrite(red1,LOW);
digitalWrite(red2,LOW);
digitalWrite(red3,LOW);
digitalWrite(ge1,HIGH);
digitalWrite(ge2,HIGH);
digitalWrite(ge3,HIGH);
digitalWrite(yel1,LOW);
digitalWrite(yel2,LOW);
digitalWrite(yel3,LOW);
}
delay(500);
}
    
```

### Building the hardware

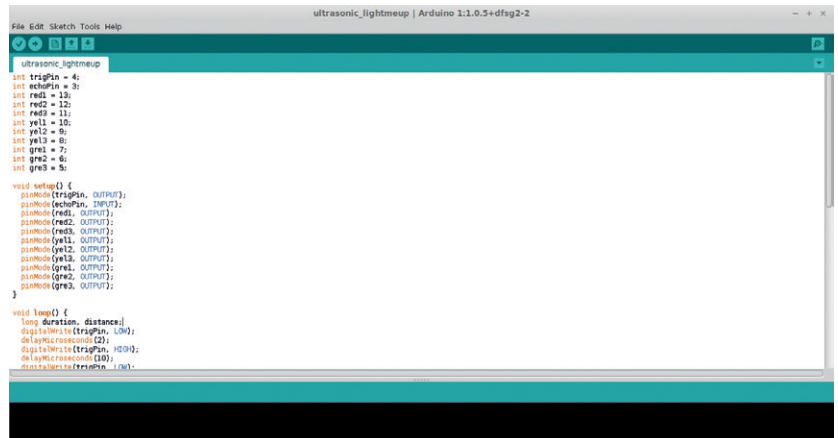
Arduino projects come as a package, with software and hardware. With the code already taken care of, our focus shifts to the hardware build of the project.

We start our build with the humble breadboard, and to the breadboard we add the HC-SR04 ultrasonic sensor, taking care to note the pin layout as we will need to connect each of those pins, using the breadboard to the relevant pins of the Arduino.

Here are the connections for the ultrasonic sensor:

- VCC connects to 5V.
- GND connects to GND (we will use the ground rail on the breadboard, marked with a "-").
- Echo connects to pin 3.
- Trigger connects to pin 4

Now we just mentioned that we will use the ground rail on the breadboard. The rails are the outer two columns of holes that are marked "+" and "-". Power, otherwise known as VCC or V+, is connected to the "+" rail, and ground, otherwise known as GND or V-, is connected to "-". In this project we just use the "-". By connecting the GND from our Arduino to the "-" rail via a jumper cable we create a common ground that any component can safely use.



With the sensor attached, now is the time to connect each of the 9 LEDs to our breadboard. LEDs come with two legs: the longest is the positive leg, commonly known as the Anode; and a shorter leg which is negative/ground and known as a Cathode. When connecting our LEDs to the breadboard, the cathode will be inserted into the same "-" (ground) rail that we used for the sensor. The longer anode leg needs to be inserted into the main breadboard area, so do this for all of the LEDs.

Our LEDs require a resistor in line from the Arduino to the LEDs. We need this to protect the LED from too much current, which can damage or shorten the life of our LEDs. For each of the LEDs use a 220Ω resistor that bridges the central channel and is in line with the LED anode leg. With the resistors inserted now grab some male-to-male jumper cables and wire to the Arduino as follows:

- Red1 = pin 13.
- Red2 = pin 12.
- Red3 = pin 11.
- Yellow1 = pin 10.
- Yellow2 = pin 9.
- Yellow3 = pin 8.
- Green1 = pin 7.
- Green2 = pin 6.
- Green3 = pin 5.

Before applying power double-check all of your connections; the worst thing that can happen is that an LED will pop, but checking your circuit is a good habit to get into. When ready, connect your Arduino to your computer via the USB lead and upload the code to your board via the upload button in the Arduino application. After about 10 seconds your project will come to life and you can move your hand in front of the sensor to trigger the different coloured LEDs. If the code does not auto start, press the reset button on your Arduino.

That's it! You've built your very own distance sensor using less than £10 of parts and around 80 lines of code. You're officially an electronics engineer! 🎉

You can find the complete code for this project at our GitHub [https://github.com/lesp/LinuxVoice\\_Issue11\\_Arduino\\_Project](https://github.com/lesp/LinuxVoice_Issue11_Arduino_Project) or as a Zip file at [https://github.com/lesp/LinuxVoice\\_Issue11\\_Arduino\\_Project/archive/master.zip](https://github.com/lesp/LinuxVoice_Issue11_Arduino_Project/archive/master.zip)

**Les Pounder is a maker and hacker specialising in the Raspberry Pi and Arduino. Les travels the UK training teachers in the new computing curriculum and Raspberry Pi.**

# PREY: RECOVER STOLEN DEVICES

Lost your laptop? Don't be a deerstalker and follow our advice to pull a fast one on the perp. The game is afoot!

## WHY DO THIS?

- Recovering a device is better than replacing it.
- The software is easy to set up and administer.
- It's also very economical and even has a very usable no-cost plan.

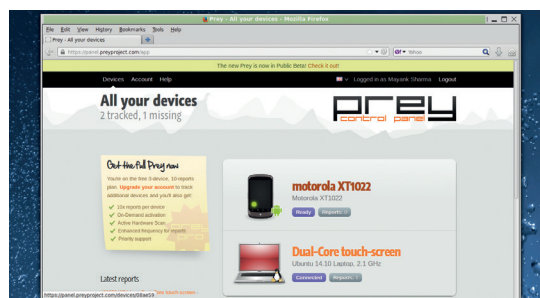
Linux does a wonderful job of insulating your computer from the electronic nasties floating about. But it all comes to naught if you leave it on a bus or lose it in a robbery. The open source *Prey* software helps recover your stolen devices by allowing you to track and control them remotely and make them unusable to anyone who's got them.

Remember, however, that to take advantage of *Prey's* abilities you'll need to install it before losing control of the laptop. *Prey* installs an agent on your device that runs in the background and periodically sends an HTTP request to check in with its online headquarters on whether it should perform any action or stay asleep. When you lose a device, you mark it as such on *Prey's* dashboard and the device then starts collecting information to help you track it down.

Besides Linux, *Prey* works on several operating systems including Windows, Mac OS X, and even Android and iOS, so you can use it to track laptops and mobile devices as well. You can use it for free to track up to three devices or upgrade to a paid Pro plan starting from \$5/month (about £3).

## Set up a device

The *Prey* project has pre-compiled binaries for Deb-based distros such as Debian and Ubuntu. To set up *Prey* on these distros, head to [preyproject.com](http://preyproject.com) and click on the 'Download now' button to grab the binary. You can double-click on the downloaded **.deb** file or use the **sudo dpkg --install prey-\*** command to install the *Prey* agent.



To prevent the thief from formatting your laptop, disable booting from removable devices and also lock the BIOS.

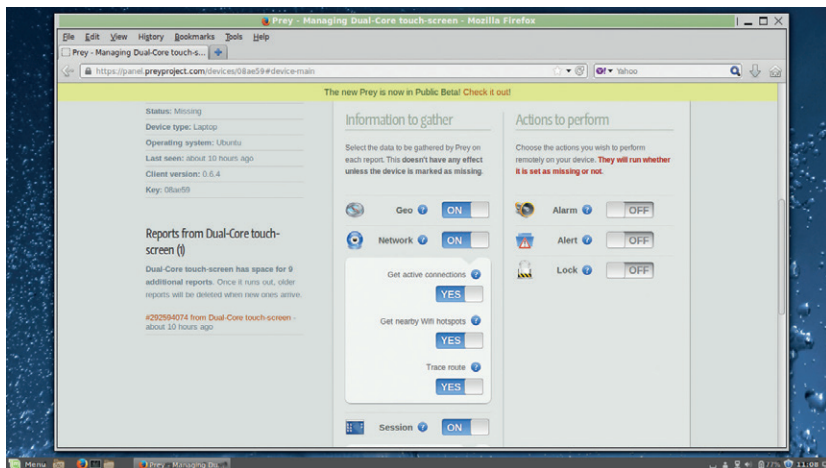
After it installs, *Prey* will fire up its graphical configuration tool and asks you to set up a reporting method. The reporting method controls how *Prey* communicates with the devices and reports back to you. The recommended method is to use *Prey's* web-based control panel, which can be accessed from any machine. You can also optionally run *Prey* in standalone mode, which would require you to set up your own SMTP mail server whose settings you need to specify in *prey's* config file under **/usr/share/prey**.

Next you need to register with the service. You can do so from within the app or by visiting **preyproject.com**. The *Prey* setup asks you for your name, email address, and a password. After the account has been set up, *Prey* will ask you to add the laptop to the list of tracked devices. It'll automatically pick up the name of the device and its type, which you can edit later from *Prey's* control panel. For subsequent installations on other devices, select the option to link the device with your existing account.

That's it. You are now ready to set up *Prey's* behaviour. If it isn't already running, launch the *Prey* Configurator (which should be under the Administration applications menu), and switch to the Main settings tab. Here you can enable a password-less guest user account to lure whoever is using your stolen device. You should also opt to activate the Wi-Fi autoconnect option, which discreetly connects to the nearest open Wi-Fi hotspot and starts sending you reports.

## Configure behaviour

After you've set up your device, you can configure its behaviour via *Prey's* web-based control panel. The control panel is broken into various sections that control different aspects of the device.



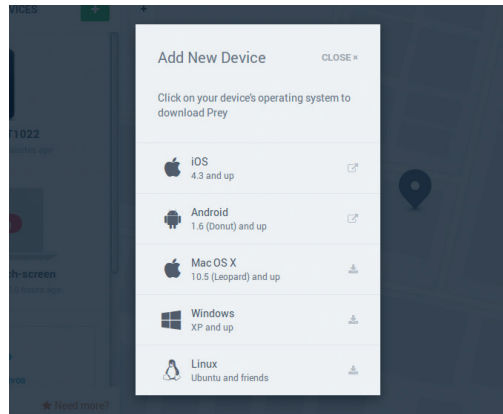
The best thing about the web-based control panel is that it allows you to configure the behaviour of *Prey* on the stolen machine even after it's been pilfered.

## Prey on an Android device

Besides netbooks, laptops, and desktops, *Prey* can also protect mobile devices. To install *Prey* on an Android device, download it from the Google Play Store. Besides Android, the app also runs on iOS and is available on Apple's App Store as well. Once it's installed, hook it to your account if you already have one, or create a new one, just like you do on the laptop version of *Prey*.

After associating it with your account, *Prey* prompts you to activate the *Prey* administrator by locking down the software with a password for extra security. Once activated, whoever's got your device will first have to revoke the privileges provided by the administrator before they can uninstall the software from your phone.

The *Prey* for Android app has the Disable Power Menu option, which when enabled prevents your device from being turned off by disabling its power menu. Also the *Prey* dashboard for the Android device has an additional option. You can toggle the Hide switch, which will then hide the icon for the *Prey* app from the Home screen.



Both the Android and iOS versions are written in their respective platform's native languages.

The main dashboard lists all added devices. Click on a device's name to set it up. Switch to the Configuration section, from where you can alter the name of the device as well as its type in case it wasn't correctly detected. Then move to the Hardware section, which gives you detailed information about the hardware on a particular device including the serial number of the device as well as details about its motherboard and other components, which helps you submit a detailed report to the authorities.

The options under the Main section are separated into two groups; 'Actions to perform' lists the actions that *Prey* will take on your device. Although these actions will be performed irrespective of the device being marked as missing or not, it's best to keep them turned off until the device is actually missing.

Some of these options are designed to dissuade the thief soon after you've lost the device. For example, the Alarm option sounds a loud alarm from your missing device to help you locate it, if it's nearby. Then there's the Alert option, which displays an alert message on the screen on the missing device. If these don't work to discourage the thief, you can use the Lock option to prevent the computer from being used until a password is entered. However, you might not want to lock out the perp as you can trace him better when he's using the laptop.

### Keep tabs on your prey

When you lose the laptop, log into *Prey's* web panel, click on the device that's missing, and use the slider at the top to mark it as such. *Prey* can discreetly gather lots of information about the missing device and its current operator. You can mark all the information you wish to gather from the missing devices' page on the web dashboard.

As soon as the device is brought online, *Prey* can use nearby Wi-Fi access points to interpolate the location on your device and mark it on Google maps. Along with this it also gathers other network-related

information such as the public and remote IP address of the network the device is connected to. You can also ask *Prey* to run a traceroute (to **google.com**) from the missing machine through the thief's router. For this to work though make sure you install the traceroute package on the missing laptop before losing it.

You can also ask *Prey* to gather information about the desktop session – including a list of running apps along with a screenshot. Sooner or later you will get a screenshot of him logging into his account on a webmail or some other website. While you won't get his password, you'll be able to clearly see his unique username, using which you can contact him.

If your device has an inbuilt webcam (most laptops and netbooks these days do), *Prey* will also secretly take snapshots of whatever's in front of the webcam. It won't take long before you catch the crook in front of your stolen device. You can set the interval after which *Prey* wakes up and collects the information you have asked it to gather. In the free version, this duration can be between 10 minutes to 50 minutes.

The Pro version allows you to take this down to two minutes or, better still, create a persistent connection with the device. One of the benefits of the Pro version is the On Demand Mode, which brings you reports from a missing device in real-time. In this mode, any changes you make to the configuration of the missing device are triggered instantly if the device is online.

You're all set. All you can do now is wait. As soon as the miscreant goes online with your laptop, the *Prey* client will alert the *Prey* web service. Although we hope you never lose your laptop, in case you do, you are now fully prepared to take on the perp who's got it, and either force him to return your device or collect enough information to build a strong case for the authorities to take action on. Happy hunting. 📍

Mayank Sharma has been tinkering with Linux since the 90s and contributes to a variety of techie publications.

BEN EVERARD

# PENETRATION TESTING: SOCIAL ENGINEER TOOLKIT

Don a stylish black hat and open up the software of choice for the discerning technical con man.

## WHY DO THIS?

- Learn the tools of online scammers so you can protect yourself.
- Begin a lucrative career as a penetration tester.
- Get a better understanding of the technologies that underpin the web.

It doesn't matter how good your computer security systems are if your users just let attackers know how to log in. Social engineering is the black art of persuading victims to tell you everything you need to know to break into their computers. Sometimes this can mean persuading them to hand over usernames or passwords, sometimes it can mean granting you physical access to their computer, and sometimes it can mean deleting any incriminating evidence. In truth, the skilled social engineer can persuade victims to bypass all sorts of computer security that would be hard to compromise using just technical means.

The *Social-Engineer Toolkit (Set)* is a piece of software that helps you set up some social engineering attacks.

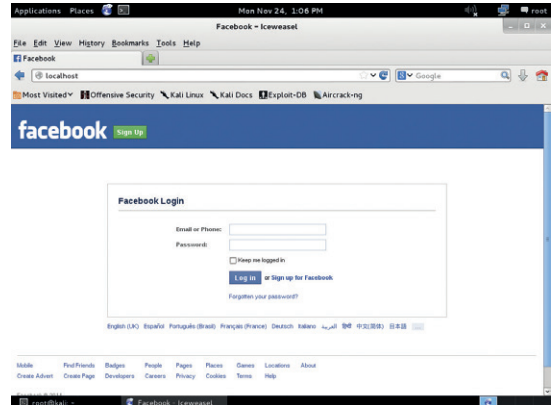
## Stealing credentials

One of the most popular uses of social engineering is tricking people into revealing their login details. This could be through a simple confidence scheme, or through some technical trickery. The *Social-Engineer Toolkit* provides some ways to make this easier.

The easiest way to try *Set* is using a security-focused distro that comes with it already installed. Kali Linux is an excellent option for this (see boxout).

Alternatively, you can grab *Set* from the Git repository. First you'll need to make sure you've installed the **git** command through your package manager. On Debian and Ubuntu-based systems, this

The *Social-Engineer Toolkit* can do far more than just clone websites. The best documentation is at [www.social-engineer.org](http://www.social-engineer.org) (under Framework).



Our login-stealing Facebook clone. Would you be fooled?

is done with:

```
sudo apt-get install git
```

Then you can clone the repository with:

```
git clone https://github.com/trustedsec/social-engineer-toolkit/set/
```

```
cd set
```

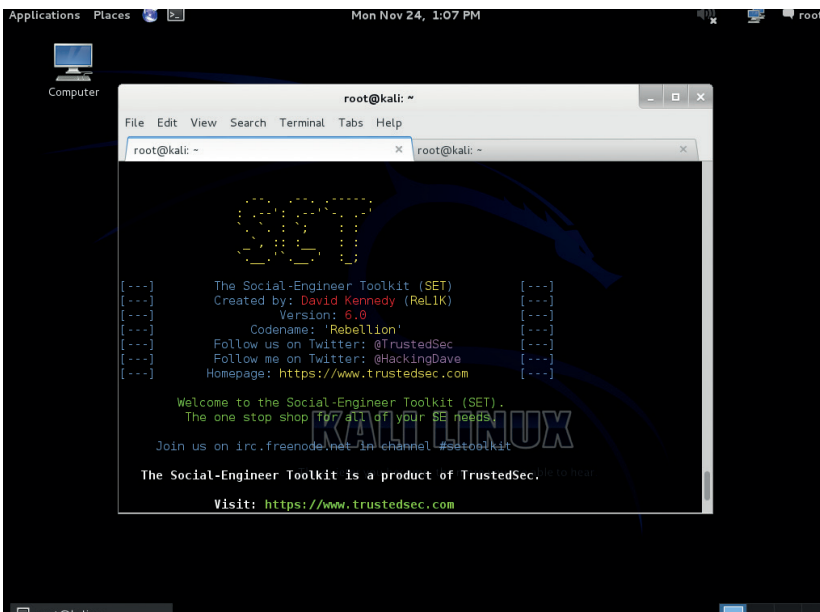
```
sudo python setup.py install
```

You can now use the command **setoolkit** to access the various features of *Set*. The attack we're going to run is called credential harvesting. It will create a clone of a website with login details, then we'll have to try and get people to log in. When people do, it'll save their username and password, then forward them on to the real site where they'll be prompted to log in again. Most people will simply assume that they entered their password incorrectly the first time, or that there's been some form of network glitch, and log in again.

In order to run this attack, you'll need a webserver with PHP running on your local machine. You can get this in Debian- and Ubuntu-based systems with:

```
sudo apt-get install apache2 php5 libapache2-mod-php5
```

If you're trying this from Kali Linux, you'll already have all this installed. If you're using another distro and can't find *Apache* in your package manager, it's sometimes in a package called **httpd**.



## Legalities

It should go without saying that the techniques we've discussed in this tutorial can be illegal if used against unsuspecting victims. While it's fine to try them out by yourself on your own local network, using them to try to break into computer networks can have very serious legal consequences. Just don't do it.

Now with everything set up, let's set up the cloned website. First, start *Set* running with root permissions:

#### sudo setoolkit

You may get a warning message about *Metasploit* not being installed. This isn't a problem for us since we won't be using any of the attacks that depend on it. However, if you want to fully investigate more of the capabilities of *Metasploit*, it's worth installing this as well (see boxout).

Note that these instructions will overwrite `/var/www/index.html` (or `/var/www/html/index.html`), so if you're already hosting any website on the machine, it's probably best to try this out in a live environment or a virtual machine.

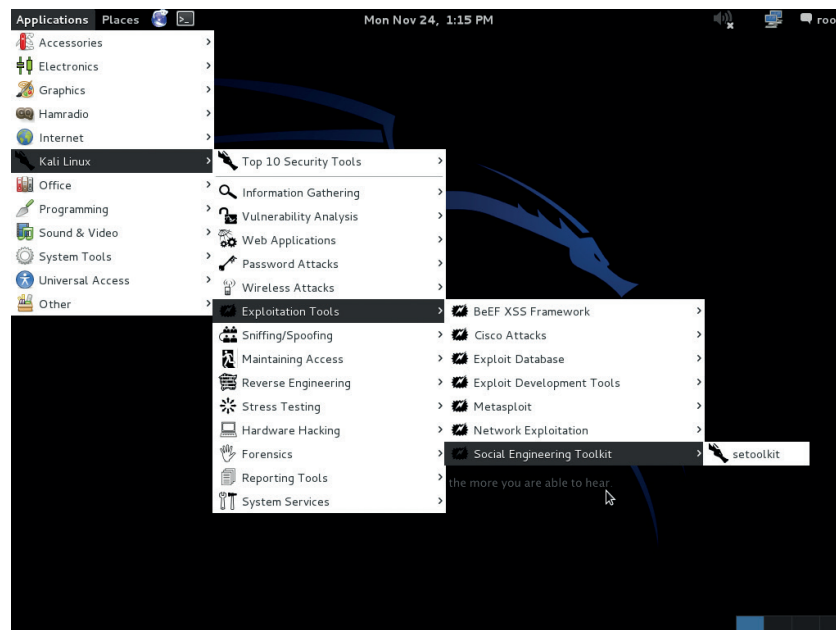
*Setoolkit* is controlled via a text-based menu system (which means it can easily be controlled remotely should you need to). The cloned website credential harvester is under: 1) Social-Engineering Attacks > 2) Website Attack Vectors > 3) Credential Harvester Attack Method > 2) Site Cloner.

Once you've selected this, you just need to enter the IP address of the machine you're running the attack from. If you're running the attack on a LAN, then this should be the local IP address of the machine on which you're running *Set*. You can find this out by running `sudo ifconfig`, and looking for something like:

```
wlan0  Link encap:Ethernet HWaddr 00:13:e8:3d:92:7b
        inet addr:192.168.0.4 Bcast:192.168.0.255
        Mask:255.255.255.0
```

In this case, the IP address for wlan0 (wireless lan – if you're using wired Ethernet, this will probably be eth0) is 192.168.0.4.

If you've got a publicly routable IP address, then you could also enter this here, but you should be careful: running a test attack on a local network is usually fine (see boxout on legalities), but if you're doing anything on the public internet, you're far more likely to run into problems with the law.



After you've entered this, you should enter the URL that you want to clone. For a test, we used Facebook (<https://facebook.com>), but you could put any website with a login here.

Once you've done that, it'll automatically make a copy of the website, host it locally, and set it to harvest the credentials.

You should now be able to point your browser to localhost and see the cloned site. If you don't see the cloned site, then it could be that *Set* has put the files in the wrong place. By default, it will put them in `/var/www`, but many modern Linux systems use `/var/www/html` as the web root. The easiest way to tell if this is the problem is by opening a new terminal and `cd`'ing to `/var/www` and seeing if the `html` folder exists. If it does, just move all three files created by *Set* to `html`:

*Set* is just one of many penetration testing tools included in Kali (see boxout).

## Metasploit

The *Social-Engineer Toolkit* is designed to work hand-in-hand with *Metasploit*. *Metasploit* isn't so much a piece of software, as a complete framework for penetration testing. It includes everything from exploits to software for recording your attacks. *Set* uses some of *Metasploit*'s features and exploits, so unless you have *Metasploit* installed, you won't get the full functionality of *Set*.

The easiest way to try out *Metasploit* is in a live environment that has it already installed, like Kali. However, you can install it on a regular Linux distro. To make this a bit easier on Debian and Ubuntu-based systems, there's a script to automate installation. You can grab it from GitHub with:

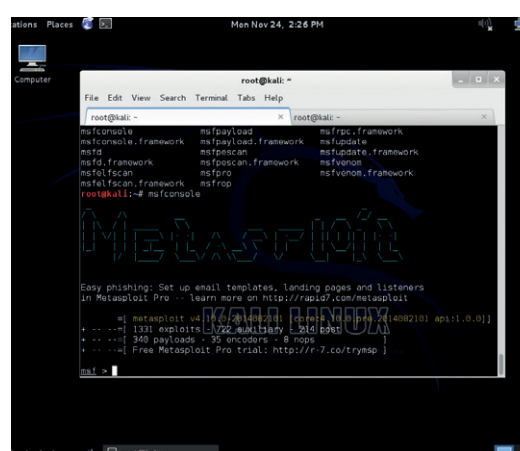
**git clone <https://github.com/darkoperator/MSF-Installer.git> msf**

Then, to install *Metasploit*, you just need to run:

```
cd msf
./msf_install.sh
```

You can find details on how to install it in Fedora at <http://fedoraproject.org/wiki/Metasploit>.

There's also a version of *Metasploit* with a HTML front-end. You can grab this from [www.rapid7.com/products/metasploit/download.jsp](http://www.rapid7.com/products/metasploit/download.jsp).



`msfconsole` (shown here) is the usual interface to *Metasploit*, but there are others including *Armitage* (graphical) and `msfcli` (for using in scripts).

## Kali Linux

If you do quite a bit of penetration testing, it's worthwhile setting up your own working environment. This is quite a worthwhile exercise; there's plenty of software that can be useful, and you'll be able to pick the ones that are useful to you. However, if you're just getting started, or if you just want to dabble, it's useful to use a ready-made penetration testing environment. In last month's Distrohopper, we took a look at Backbox. This is one good option, but by far the most popular is Kali (formerly BackTrack Linux). It comes in flavours for small ARM machines as well as x86 desktops.

You can install it, but you can also run it live, and it has almost every open source (and some closed source) penetration tool set up and ready to run. You can grab an ISO from [www.kali.org](http://www.kali.org), and then use it like any other distro. It also runs well in a virtual machine if you want to separate your penetration testing from your main desktop.

```
mv /var/www/index.html html
```

```
mv /var/www/post.php html
```

```
mv harvester* html
```

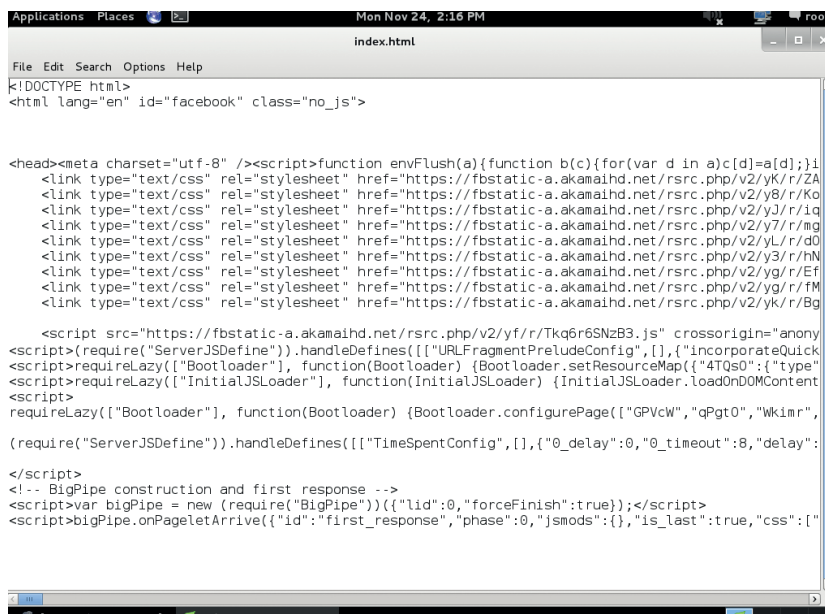
Now you should be able to point your browser to localhost and see the cloned site. If you enter dummy details in there, you should find that you get forwarded to the real Facebook page, and that whatever you enter is copied to the harvester file in the webroot.

After a visitor goes to your site, you should see an entry like the following in the harvester file:

### Array

```
(
  [isd] => AVrTM83X
  [display] =>
  [enable_profile_selector] =>
  [legacy_return] => 1
  [profile_selector_ids] =>
  [trynum] => 1
  [timezone] => 0
  [lgnrnd] => 125137_iouC
  [lgnjs] => 1416689527
  [email] => test@test.com
  [pass] => test
```

Set just sets up the HTML and PHP file. The actual attack could be run on any computer with a webserver installed.



```
[default_persistent] => 0
```

This contains all the POST data that the user sent back to the website. In this case, the important fields are **email** and **pass**.

## Getting visitors

The problem now is to get victims to go to your fake site. There are a few options here. Perhaps the simplest is simply tricking them to click on a link. The classic approach here is to send them an email with a link to Facebook that actually points to your clone. This, however, will look a little strange to anyone who clicks on the link because the URL in the address bar will be an IP number not the proper domain.

One way around this is to register a domain that looks similar to the one in you're cloning. For example, **www.facebook.com**. A casual glance won't show that there's anything wrong with that. What's more, you could even get a real SSL certificate for it so that you could encrypt the connection and make it look even less suspicious. For Facebook, you're unlikely to find a domain that looks similar that's not already taken. This approach does have the downside that it costs money, and leaves a paper trail.

Another approach is to attack the Domain Name System (DNS) in such a way that when the user enters a domain name, they get pointed to your site instead of the real site. The easiest way of doing this requires modifying a file on the victim's machine. This could be done in a few ways. You could get the victim to use your machine (with the attack already set up). If you can get physical access to their machine, you could do it using a live distro to bypass any passwords they may have (unless their hard drive is encrypted).

Whenever you type in a domain name, like **www.facebook.com** or **google.co.uk**, your computer first checks a file called **hosts**. If this domain isn't in that file, it then sends a message to its DNS server asking which machine corresponds to the domain name. To get the victim's computer to send the request to our malicious server, all we have to do is

## Protect yourself

If the attacker does this well, it can be hard to spot a cloned site. One obvious giveaway is a lack of SSL on a login page. However, even this can be faked if the attacker is either using a real domain, or has managed to get access to your computer (where they could install a fake certificate).

You can avoid getting caught out by fake domains by always typing in the domain of any important sites rather than just clicking on links, but this won't protect you if they've managed to hijack your DNS connection. The best protection against an attacker installing fake certificates on your computer is full disk encryption.

Important sites should use two-factor authentication. This combines usual login details with a second form of security (such as a code that is sent via SMS). Using something like this means that the credentials the attacker steals won't be sufficient to log them in.

add an entry to their **hosts** file. On Linux systems, this file is in **/etc/hosts**. In most versions of Windows, it's in **Windows\System32\drivers\etc**. Entries are simply a domain name, then a space (or tab), then the IP address that domain should resolve to.

If we hijack **www.facebook.com**, then whenever the user goes to **www.facebook.com**, they will be directed to our site. However, this causes a problem because they won't be able to get to the actual Facebook site, and so will quickly see that there is a problem. However, if we redirect **login.facebook.com** to our site, we can then forward them on to **www.facebook.com**, and it should be fairly trivial to persuade a victim to click on a link to **login.facebook.com** (*Set* is, after all, an aid to social engineering, not a complete hacking solution).

The line you need to add to the **hosts** file is:

```
192.168.0.4 login.facebook.com
```

If you do this on the same machine that you're running the server on, you won't be able to clone the site once you've entered this because it will interfere with the way *Set* clones the URL. However, you can disable this line in the hosts file by adding a **#** character to the start of it.

You can get your cloned web page to point the user onto whatever other page you want to by editing the **post.php** that *Set* puts in **/var/www**. The file should contain a single line that's something like:

```
<?php $file = 'harvester_2014-11-22 20:51:37.547239.txt';file_put_contents($file, print_r($_POST, true), FILE_APPEND);?><meta http-equiv="refresh" content="0; url=https://www.facebook.com/login.php" />
```

The first part of this (in the **php** tag) harvests the credentials, while the second (in the **meta** tag) forwards the user onto the correct site. In this case, we've set it to forward to **https://www.facebook.com/login.php**, but this could be anything as long as it doesn't cause a problem with your **hosts** file.

## Google dorks

If you want to use this approach on victims on your local network, you'll need to run this on a public server. This is legally dubious so it's probably best that you don't do it.

Another problem with running this sort of attack using a public server is that it uses a standard set of filenames. While you could quite easily change the **index.html**, **post.php** and harvester files, many people don't. The first two files there are generic enough that there are files with these names in lots of web apps. However, the harvester filename is quite unusual. If you can find a public server with a file whose name starts with **harvester\_2014**, then there's a good chance that it's currently running *Set*'s credential harvester.


Finding files on the internet is easy, you just use Google. In this case, if you search for "inurl:harvester\_2014" you'll find (among

some other things) servers that are currently running *Set*.

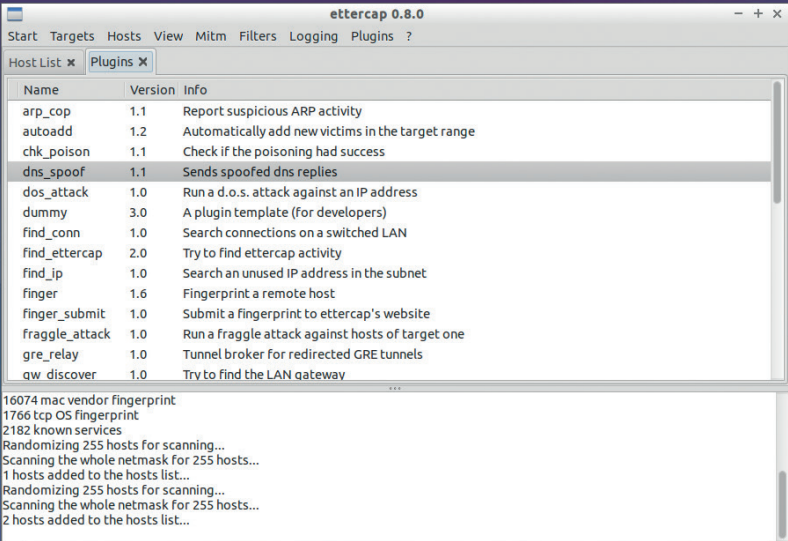
Obviously, these sites tend to go down and come back up quite regularly. When we did it, there were a couple of live harvesters, only one of which had any data in. Assuming they haven't changed the default, the cloned site will be **index.html** in the same directory. Sure enough, we found that it was a clone of Facebook. We didn't check to see if any of the credentials in the harvester file were real (and obviously, doing this would be illegal).

This style of using Google to find things that are useful to hackers is known as Google Dorks. If you know the more advanced syntax of Google searches, you can find all sorts of things that were never meant to be made public. There's a database full of useful examples at **www.exploit-db.com/google-dorks**.

Alternatively, you could change this to an HTML page that just contains an error saying that the website is down temporarily. As long as it has the first **php** tag, it will still harvest the credentials.

This approach is fairly simple to set up, but it does require you to have access to the machine that the victim's on. There are ways of getting around this requirement. To do this, you need to perform a man-in-the-middle attack which can either be physical (that is, you actually set up the network so that their connection has to flow through your computer), or by using an ARP spoofing attack, which will fool other computers on the local area network into routing their traffic through your machine. 

**Ben Everard is the best-selling author of *Learning Python With Raspberry Pi*. He really wants you to be careful on the web.**



The screenshot shows the ettercap 0.8.0 interface. The 'Plugins' tab is active, displaying a list of plugins with their names, versions, and brief descriptions. The 'Host List' tab is also visible, showing a list of hosts being scanned.

Name	Version	Info
arp_cop	1.1	Report suspicious ARP activity
autoadd	1.2	Automatically add new victims in the target range
chk_poison	1.1	Check if the poisoning had success
dns_spoof	1.1	Sends spoofed dns replies
dos_attack	1.0	Run a d.o.s. attack against an IP address
dummy	3.0	A plugin template (for developers)
find_conn	1.0	Search connections on a switched LAN
find_ettercap	2.0	Try to find ettercap activity
find_ip	1.0	Search an unused IP address in the subnet
finger	1.6	Fingerprint a remote host
finger_submit	1.0	Submit a fingerprint to ettercap's website
fraggle_attack	1.0	Run a fraggle attack against hosts of target one
gre_relay	1.0	Tunnel broker for redirected GRE tunnels
qw_discover	1.0	Try to find the LAN gateway

16074 mac vendor fingerprint  
1766 tcp OS fingerprint  
2182 known services  
Randomizing 255 hosts for scanning...  
Scanning the whole netmask for 255 hosts...  
1 hosts added to the hosts list...  
Randomizing 255 hosts for scanning...  
Scanning the whole netmask for 255 hosts...  
2 hosts added to the hosts list...

*Ettercap* can be used to manipulate a network to intercept DNS requests. Run it from the command line with **sudo ettercap -G** to get the graphical version shown here.

# LINUX 101: RUN WINDOWS APPLICATIONS WITH WINE

MIKE SAUNDERS

If you still depend on a few Windows programs, or you want to help newbies make the switch to Linux, Wine is mightily useful.

### WHY DO THIS?

- Run legacy Windows apps without rebooting.
- Create multiple configurations for better compatibility.
- Help Windows users move over to Free Software.

Question: what do you call a program that runs software designed for a different platform? An emulator, right? Well, the name *Wine* comes from “Wine Is Not an Emulator” – which is one of those recursive acronyms that are so loved in the FOSS world. But given that *Wine* lets you run Windows software on your Linux installation, why is it not an emulator? Essentially, *Wine* acts as a compatibility layer that translates Windows system calls to their Linux equivalents, and it doesn’t actually emulate a complete Windows PC, with its CPU, graphics card and so forth.

Anyway, with that naming confusion out of the way, let’s focus on the software itself. *Wine* is a godsend for many Linux users who’ve made the transition from Windows, but still need to run the occasional program

from the latter operating system. It’s completely free software – you don’t need a licence from Microsoft to use it – and it’s capable of running a wide range of programs. Not all of them, mind you, and very recent software can have problems. But some major applications like *Microsoft Office 2010* work well enough for daily use.

So if you’re still dual-booting between Linux and Windows, and would rather spend more time in the former, here we’ll show you how to use *Wine* and (hopefully) run your favourite Windows apps without rebooting. Or if you’re a full-time Linux user and don’t give a hoot about Windows, you can still use this guide when you’re helping others make the transition to Linux, set up *Wine* for them and demonstrate the awesome power of free software.

## 1 GETTING STARTED

Here’s our first app running on *Wine* – *Notepad++*. It’s a simple program and therefore has few compatibility issues with *Wine*.

*Wine* is included in almost every major distro’s repositories, so find it in your package manager or use your usual command-line tools to install it (eg **sudo apt-get install wine** on Ubuntu-based distros). We’re using Arch Linux for this tutorial – but the commands are the same across other distros. If you want the latest and greatest version and are happy compiling software from its source code, you can get it from

**www.winehq.org**. After you’ve got it installed, find a simple, standalone Windows program to test; in our case we’re going to use the rather cool *Notepad++* text editor available from **http://notepad-plus-plus.org**. This program exists as a single .exe file and doesn’t have a ton of complicated dependencies, so it’s the perfect type of program to kick tires of a new *Wine* installation.

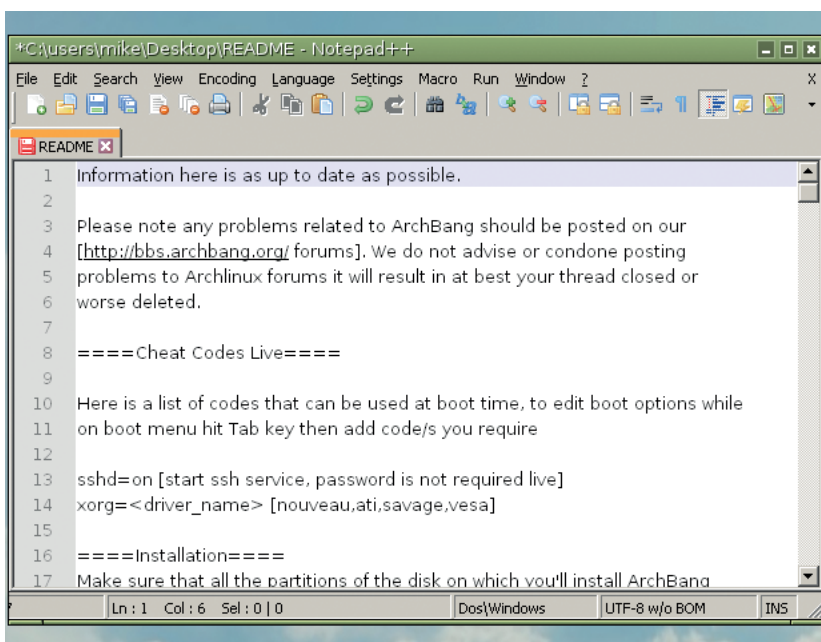
Go to the Downloads section, then grab the “minimalist package” and save it to your home directory. This is in 7-zip format, so install the tool to extract that in your distro’s package manager (it should be provided in the package **p7zip**). Then open a terminal and enter:

```
7z x -onpp npp.6.6.9.bin.minimalist.7z
cd npp
```

Here we’re extracting the download into a new **npp** directory – if there’s a newer version of *Notepad++* by the time you read this, change the version number accordingly. We then switch into the directory, and if you enter **ls**, you’ll see that there’s a file there called **notepad++.exe**. Let’s run it!

```
wine notepad++.exe
```

You may be prompted to install Mono and Gecko packages; these aren’t important for now, so just click Cancel in the dialog boxes that appear. And after a few moments, you’ll see *Notepad++*, a Windows program, in all its glory on your Linux desktop. Not bad – it’s as simple as that!





## CrossOver: the commercial alternative

If you're looking for improved compatibility, easier installation and technical support, *CrossOver* ([www.codeweavers.com](http://www.codeweavers.com)) is worth a look. This is a commercial version of *Wine* with various extras, and is especially useful if you want to run Microsoft Office (XP to 2010), *Adobe Photoshop* and several triple-A games like *World of Warcraft*. *CrossOver* includes a tool called *CrossTie*, which lets you install applications straight from the web with just a couple of clicks, and it also uses a bottles system to stop different configurations from overwriting one another. It also has some tweaks to integrate more smoothly with KDE and Gnome.

Pricing starts from €32; this gets you one month of email support and upgrades (when new versions are released). For €48 you get the whole package, which includes one year of email support and upgrades, along with a "phone support incident" – ie you can speak to one of the devs on the phone if you're having serious trouble. (Subsequent calls cost €16.95 each.) *CrossOver* employs *Wine* developers and contributes code back to the main tree, so if you find *Wine* really useful and want to support its development financially, buying *CrossOver* is a good idea. Alternatively, you can donate directly at [www.winehq.org/donate](http://www.winehq.org/donate).

Well, for small programs it's simple; we'll get to the more complicated setups later. For now, try exploring the program. As mentioned, *Wine* translates every Windows system and library call that the program makes into a Linux equivalent. So if you save a file from *Notepad++*, the program calls Windows' file saving routine, *Wine* intercepts it, and forwards on the request to the Linux equivalent. When *Notepad++* wants to draw something on the screen, it makes requests to Windows libraries – and *Wine* has its own versions of these, which then talk to the X server on Linux. It's very cool technology.

### Two worlds collide

Of course, *Wine* can't magically make some of the differences between Linux and Windows disappear. Go to File > Open in *Notepad++*, for instance, and select My Computer from the "Look in" list. You'll see C:, D: and Z: drives – but they make no sense in the Linux world. Well, *Wine* maps them to different locations in your filesystem. The Z: drive is mapped to the root directory (**/**), which is the base of everything in a Linux/Unix system. So you can use that drive to go to your home directory in **/home** and access your personal files – or use the My Documents shortcut.

But where does C: point to? If you click into it, you'll see some familiar folders from a Windows installation: Program files, windows and so forth. These were created when you first ran *Wine*, so let's explore them in more depth. They're located in **.wine/drive\_c** in your home directory, so close *Notepad++* and switch into that directory like so:

```
cd ~/.wine/drive_c
```

Enter **ls** and you'll see those folders again. Switch into the windows directory with **cd windows** and run **ls** again – this time, you'll notice some common tools like *Regedit*. Basically, *Wine* has created a very minimal Windows installation in your home directory, comprised of fully open source software, of course. So you can run the included tools like so:

### wine regedit

(Note that you can omit the **.exe**.) This looks just like the real Windows registry editor, but go to Help > About and you'll see that it's a tool written by the *Wine* developers. Back in the terminal, if you head into the **windows/system32** directory (and **syswow64** on 64-bit installations), you'll see a bunch of DLLs. These

are *Wine*'s own implementations of core Windows libraries – and again, they're fully open source.

Now, they're not always as feature-complete as the original Windows versions, so in some cases you can copy DLLs from a real Windows installation into this directory, to improve compatibility with certain programs. The only ones you must never overwrite are **kernel32.dll**, **gdi32.dll**, **user32.dll**, and **ntdll.dll** – you can only use the *Wine* versions of these.

As an aside, the ReactOS project ([www.reactos.org](http://www.reactos.org)), which aims to create an open source Windows-compatible operating system, uses many *Wine*

DLLs. The underlying structure of ReactOS is very different to Linux and Unix, as it aims to be compatible with Windows drives as well as software, but there's a decent amount of code-flow between ReactOS and *Wine*. We've been following ReactOS for many years and it's making slow but steady progress – what Microsoft's lawyers think about it, though, remains to be seen...

**"Wine translates every system call that an application makes into a Linux equivalent."**

Adobe has dropped support for *Reader* on Linux, but thanks to *Wine*, you can get some Windows versions running.

Linux+Voice-Sample.pdf - Adobe Reader

File Edit View Document Tools Window Help

68.5%

Find

WELCOME

## Welcome to Linux Voice

75,000 of words of awesome each and every month.

**LINUX VOICE**

Linux Voice is different. Linux Voice is special. Here's why...

- 1 At the end of each financial year we'll give 50% of our profits to a selection of organisations that support free software, decided by a vote among our readers (that's you).
- 2 No later than nine months after first publication, we will relicense all of our content under the Creative Commons CC-BY-SA licence, so that old content can still be useful, and can live on even after the magazine has come off the shelves.
- 3 We're a small company, so we don't have a board of...

**GRAHAM MORRISON**  
A free software advocate and writer since the late 1990s, Graham is a lapsed KDE contributor and author of the Meeq MIDI step sequencer.

**W**e've pooled some of our favourite features, reviews and tutorials from our first few issues to give potential readers and subscribers a better idea of what our magazine contains. We put together 115 pages of content like this each and every month.

Linux Voice is the magazine we launched after a trailblazing \$200,000 Indiegogo campaign that concluded in December 2013. Our success was purely down to the incredible support shown by the community, and wonderful endorsements from the likes of Simon Phipps, Karen Sandler, Eben Upton and Jono Bacon. As old hands in the Linux magazine business, we started Linux Voice because we wanted to create the best magazine we could.

**SUBSCRIBE**

## 2 INSTALLING SOFTWARE AND CUSTOM SETUPS

So far, we've just tested a simple standalone `.exe` program. But what if you want to install something more complicated, like a program that extracts and installs its own files? Let's try *Adobe Reader*, given that it's no longer supported on Linux. The first thing to do before installing any program is to check its compatibility ratings, so in this case we go to <https://appdb.winehq.org> and enter "Adobe Reader" in the search box in the top-right. When the results come up, we click on the "WineHQ – Adobe Reader" link.

Here we can see that different versions of the program have different ratings: gold means that a program works almost flawlessly, whereas silver and bronze mean that the program is usable, albeit with some glitches or other issues. (These compatibility ratings are provided by the community, and some programs have only been tested with older *Wine* releases, so it's possible that compatibility has improved in the meantime.)

*Adobe Reader 9.x* is rated as silver, so click it and then, on the following page, the first "Free Download" link on the left. Save the `AdbeRdr90_en_US.exe` file to your home directory, fire up a terminal, and enter:

```
wine AdbeRdr90_en_US.exe
```

This isn't the program itself, but rather its installer – so follow the prompts, and don't worry if the installer gets confused and crashes right at the end. (This is a common occurrence in *Wine*, when installers don't quite understand that they're not running in an original Windows environment, but usually it's not a problem.)

Now, like in regular Windows, *Adobe Reader* has been installed in a **Program Files** directory. In this case, you'll find it in `~/.wine/drive_c/Program Files (x86)/Adobe/Reader 9.0/Reader`. If you `cd` into that directory and enter `ls`, you'll see `AcroRd32.exe` – that's the program you want to run with *Wine*. So give it a go, and try opening some PDFs – by and large, it does its job without major bugs. You can now create a

launcher on your desktop that runs the program with this command (note the use of quote marks to get round spaces in directory names):

```
wine ~/.wine/drive_c/"Program Files (x86)"/Adobe/"Reader 9.0"/Reader/AcroRd32.exe
```

Another useful launcher to create is "wine explorer", which starts up a file manager, so you can then browse into Program Files (x86) yourself and launch programs by double-clicking on them.

As you'd expect, *Wine* is a hugely configurable piece of software, but fortunately there's a fairly good GUI tool that lets you tweak settings without fiddling around inside config files. Enter `winecfg` and a small Windows utility pops up with various tabs. The most important of these is Applications: here you can select `.exe` files in your *Wine* installation, and then choose the Windows version that *Wine* should emulate for them. So if you know that a certain program works best in Windows XP, or Windows 7, you can select it at the bottom. In general, *Wine*'s compatibility is more complete when it comes to older Windows versions.

Another useful tab here is Libraries. Here you can choose whether *Wine* should override its inbuilt libraries with native ones, as mentioned earlier. Under the Graphics tab you can also customise the screen resolution, which helps if certain programs are being displayed incorrectly.

### Bottle it!

Things can get complicated when you've customised your *Wine* installation for one particular program, and it's running beautifully, but then you install another program that needs different settings, library overrides, and so forth. It's a colossal pain to keep switching options manually, but thankfully, *Wine* has a solution for this called prefixes (aka *Wine* bottles). This lets you create and maintain separate *Wine* installations for your programs – albeit with a bit of extra disk space usage.

### LV PRO TIP

It's important to note that Wine programs can access your Linux system like any other app. They're not sandboxed or restricted. Of course, the inbuilt security mechanisms of Unix and Linux should stop a malicious app from completely hosing your system, but if you want to be ultra secure, use Wine on Linux inside a virtual machine!

### LV PRO TIP

Got a program that's supplied as an `.msi` file? You can install these using the `msiexec` tool along with the `/i` flag – for instance, `msiexec /i filename.msi`. This tool is provided as part of a standard *Wine* installation.

## Running DOS software with DOSBox

If you've got some really old programs that you'd like to get running again, from the days when MS-DOS ruled the (business) world, you're also in luck. FOSS platforms have had great DOS compatibility for many years thanks to *DOSEMU*, but that program hasn't seen many updates recently, and can be fiddly to set up. A better alternative can be found in *DOSBox* – and it's available in all major distro's repositories.

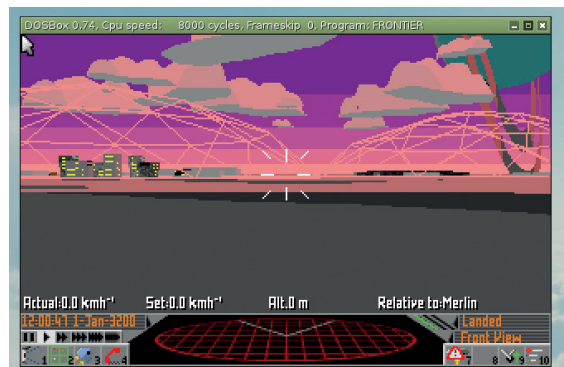
Once you have it installed, you just need to point it at a directory containing your DOS program(s). In this example we've got *Frontier Elite II* in a directory called `frontier`, so we just run:

```
dosbox frontier
```

When *DOSBox* starts, it maps its emulated C:

drive to the directory you specified. So if you enter `DIR` now, you'll see the files inside the `frontier` directory – including `frontier.bat`, which you can use to run it. *DOSBox* will often grab the mouse cursor for itself, so to free it, press `Ctrl+F10` at the same time.

To configure *DOSBox*, run it on its own (without a directory) and then enter `config -wc dosbox.conf` in the emulated DOS session. Type `exit` to leave, and you'll see `dosbox.conf` in your current directory. You can now edit this to tweak settings, such as full-screen mode, mouse sensitivity, and how quickly it should run (look at the cycles option). If you need any help, you'll find plenty of it on the wiki at [www.dosbox.com/wiki](http://www.dosbox.com/wiki).



*Elite: Dangerous* should be out by the time you read this, but we'll never stop loving *Frontier*.

The key to this is the **WINEPREFIX** environment variable. Say you want to install program **fooapp.exe** into a new *Wine* installation, and not **.wine** in your home directory. You would run this command:

```
env WINEPREFIX=~/.wine_fooapp wine fooapp.exe
```

A new *Wine* installation, with fresh settings and libraries, will be created in **.wine\_fooapp** in your home directory. Once the app is installed, you can run its executable as usual, but make sure to keep the **env WINEPREFIX=~/.wine\_fooapp** part otherwise it all gets messy. Essentially, you can make as many *Wine* prefixes as you like (at the cost of 35MB each time), but always make sure you're pointing **WINEPREFIX** to the appropriate place, otherwise one installation can overwrite settings from another.

To run **winecfg** for a particular prefix, you also need to specify the environment variable:

```
env WINEPREFIX=~/.wine_fooapp winecfg
```

If you want to completely remove a program and its prefix, just remove the directory.

Another useful environment variable is **WINEARCH**. If you're running a 64-bit distro, *Wine* will start in 64-bit mode by default; if this leads to problems with your programs, you can change this by using **env WINEARCH=win32** before your commands.

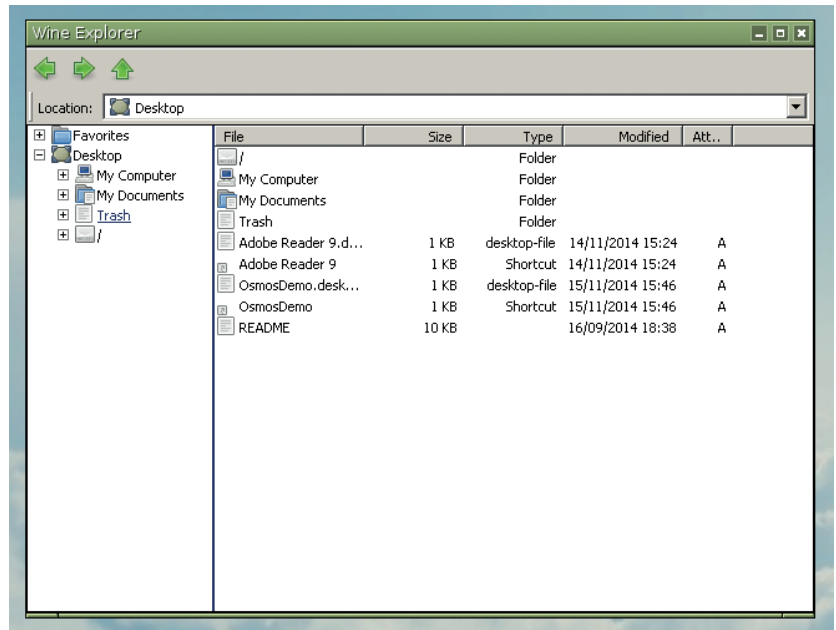
## Tricks up the sleeve

Finally, we want to give a mention to *Winetricks* ([www.winetricks.org](http://www.winetricks.org)), a very handy little script that assists you in installing various programs and games. Many distros include it in their package repositories – if you can't find it, just grab it from the website (the Installing page explains how to do it step-by-step). You'll also need some kind of utility for displaying dialog boxes, such as *Zenity* or *kdialog* from KDE.

When you run *Winetricks*, you'll be prompted to install an application, benchmark or game. Try installing an app: you'll see that many of them can be downloaded automatically (check the Media column), but in some cases, such as with Microsoft Office, you'll need the original CD or DVD. As a test, try installing the *AbiWord* word processor: *Winetricks* will download the setup.exe file and run it in *Wine*.

After the installation, *Winetricks* will return to its original menu, but you'll see a new item: "Select *AbiWord*". Click on this and then OK, and another menu will appear so that you can configure the installation. You can access the usual **winecfg** tool in this way, or also fine-tune options via the Change settings item. Note the titlebar here – *Winetricks* has installed *AbiWord* into its own prefix, in **~/local/share/wineprefixes/abiword/**. So if you **cd** into that directory in a terminal, you'll see the usual **drive\_c/Program Files (x86)** subdirectory underneath it, and then *AbiWord* under that. (The launcher, **AbiWord.exe**, is inside the **bin** directory.)

*Winetricks* also provides access to a large list of games and demos, many of which are great for testing the performance of *Wine*. Just remember that they can swallow up your disk space quickly, so it's a



good idea to remove the prefixes when you're done with them!

## The future of Wine

*Wine's* development dates back to the mid-90s, so it's one of the longest-running projects in the Free Software world. After two decades of development, though, why are there still compatibility problems with so many programs? Well, part of the problem is that Windows is a moving target. When *Wine* started, its goal was to provide compatibility with Win32 – in other words, the APIs used on Windows 95, 98 and NT. But since then, we've seen many more releases of Windows, and *Wine* developers keep trying to chase the latest APIs.

Some would argue that the *Wine* team should set a very specific goal: compatibility with Windows XP, for instance, and forget about Vista, 7 and 8. This could make sense in positioning *Wine* as a solution for legacy applications, but other users and developers want to use *Wine* to play the latest games and run recent versions of *Office*. As most *Wine* developers are hacking on the code out of a labour of love, nobody can force them to limit the compatibility to a specific Windows release.

And then, trying to be compatible with Windows APIs is an adventure in itself. Many APIs are undocumented or don't behave as expected, so it's not just about following a spec like POSIX. After all, it's in Microsoft's interests that a compatible OS from a third party isn't developed. Sure, the Redmond giant has been more friendly with the FOSS community recently, but we don't expect it to suddenly get behind the *Wine* project or open up reams of specifications to help its development. 🍷

Mike is a recursive acronym and stands for "Mike ist kein Emulator". Blame his parents.

Wine Explorer is a simple file manager that you can use to browse files and launch programs.

### LV PRO TIP

Spaces in file and directory names are common in the Windows world, but they're a royal pain in the rear on the Linux command line. You can use escape characters (backslashes) to get around them if you're a long-time Linuxer, but for new users it's best to just use quotes. So if you need to **cd** into the **Program Files (x86)** directory, enter **cd "Program Files (x86)"**.

# SAMBA 4: IMPLEMENT ACTIVE DIRECTORY DOMAIN SERVICES

Master your Windows domain from the comfortable familiarity of your Linux server.

## WHY DO THIS?

- Administer Windows machines on a network without having to abandon your Linux working environment.
- Learn one of the most important features in Samba 4.

## LV PRO TIP

Implementing Samba requires root privileges. `sudo -i` gives you a root prompt.

**S**amba is an open source implementation of the protocols for user and resource management in a Windows network. It allows Unix-like operating systems such as Linux and OS X to share files and printers, and to authenticate and manage users and resources in a Windows network.

The venerable version 3 series had long satisfied the file sharing needs of many Linux systems, until Microsoft introduced its Active Directory user and resource management infrastructure. But version 4 of Samba resolves this, because it is fully-compatible with it. In this tutorial, we'll install the Samba Version 4 server and configure it as an Active Directory Domain Controller. Up-to-date distros should have updated their Samba version, but you can always download the latest sources from the [samba.org](http://samba.org) website. We'll use the "Trusty Tahr" Ubuntu Server, version 14.04, as it's a long term support release that includes Samba 4.1.6 in its repositories. This makes installation straightforward – as root:

```
$ apt-get install samba smbclient
```

We also installed **smbclient**, the command line Samba client. We'll use it to help test our server.

Ubuntu's Samba package automatically starts the daemons upon installation. We're about to reconfigure it, so stop them now:

```
$ stop smbd
```

```
$ stop nmbd
```

Samba's main administration tool, **samba-tool**, is used to provision (set up) a new domain controller. You need to remove the pre-installed default Samba configuration file before you begin otherwise

provisioning will fail (it writes a new one and won't overwrite an existing one):

```
$ rm /etc/samba/smb.conf
```

You should also ensure that your server is configured with a static IP address and has itself listed as its primary name server. If you need help configuring this, our Network Configuration box explains what to do.

Interactive provisioning prompts for you to enter the required information but offers default values that are usually acceptable. The first question asks for a Realm, which is the domain suffix that Active Directory will apply to all hosts that join the domain. The default value is the default search domain for your network, as defined in `/etc/resolv.conf` and converted to upper case letters (eg **EXAMPLE.COM**) and it's fine to accept this suggestion.

You will also be asked to choose a DNS Backend. Samba requires a DNS server and implements one internally if you accept the default **SAMBA\_INTERNAL** option. This should be suitable for most uses but you can use an external BIND DNS server if you prefer.

The provisioning tool asks two questions that require non-default answers. You need to supply:

- The DNS Forwarder Address: the IP address of another DNS on your network, such as another name server defined in `/etc/resolv.conf`,
- An Administrator Password of your choosing that is suitably complex – it needs to have least eight characters containing three of these four kinds: lower-case letters, upper-case letters, digits and symbols. We'll use **"Pa\$\$w0rd"** in this tutorial; you should use something different.

Provisioning can be as simple as:

```
$ samba-tool domain provision --interactive
```

however, it's best to add some optional arguments to gain some additional benefits:

```
$ samba-tool domain provision --interactive --use-rfc2307 --use-xattrs=yes
```

The `--use-rfc2307` argument configures Active Directory so that it can store Unix user attributes, and this makes it possible to authenticate Linux users with Samba. The second argument allows Samba to support access control lists. These are lists of permissions that augment the basic **user**, **group** and **others** entitlements. Windows makes extensive use of them.

To support access control lists, the Linux kernel and any filesystem that you want to use with Samba

A new server hasn't got much to share, but there's no harm in looking.

```
ubuntu@samba:~$ smbclient -L localhost -U%
Domain=[EXAMPLE] OS=[Unix] Server=[Samba 4.1.6-Ubuntu]

  Sharename      Type      Comment
  -----
  netlogon       Disk
  sysvol         Disk
  IPC$           IPC      IPC Service (Samba 4.1.6-Ubuntu)
Domain=[EXAMPLE] OS=[Unix] Server=[Samba 4.1.6-Ubuntu]

  Server          Comment
  -----
  WORKGROUP       Master
  WORKGROUP       SAMBA

ubuntu@samba:~$ smbclient //localhost/sysvol -U Administrator -c ls
Enter Administrator's password:
Domain=[EXAMPLE] OS=[Unix] Server=[Samba 4.1.6-Ubuntu]
.
.
example.com
D          0 Tue Sep 16 12:27:47 2014
D          0 Tue Sep 16 12:30:35 2014
D          0 Tue Sep 16 12:27:54 2014

39293 blocks of size 131072. 4824 blocks available
ubuntu@samba:~$
```

need to have extended attribute (abbreviated to 'xattr') support. You should be fine with the ext4 filesystem, but options for various other filesystems are explained at [https://wiki.samba.org/index.php/OS\\_Requirements](https://wiki.samba.org/index.php/OS_Requirements). You'll also need the **attr** and **acl** packages. Ubuntu 14.04 includes all of this by default.

You can start Samba when provisioning completes; the Ubuntu-specific way to do this is to use Upstart:

```
$ start samba-ad-dc
```

but you can instead run the daemon directly, a distro-agnostic approach that is also useful when testing: to run it in the foreground with debug logging you can use:

```
$ samba -i -d 2 -M single
```

These, and many other, command line options are documented on the daemon's manual page (**man 8 samba**).

With Samba running, you can exercise the DNS to ensure it returns the expected results:

```
$ host -t SRV _ldap._tcp.example.com
```

```
_ldap._tcp.example.com has SRV record 0 100 389 samba.example.com.
```

```
$ host -t SRV _kerberos._udp.example.com
```

```
_kerberos._udp.example.com has SRV record 0 100 88 samba.example.com.
```

```
$ host -t A samba.example.com
```

```
samba.example.com has address 10.0.100.1
```

You may have seen the notification when the provisioning completed that "a Kerberos configuration suitable for Samba 4 has been generated". Kerberos is the authentication protocol used by Active Directory and the generated configuration allows you to interact with Samba's Kerberos services. Doing so is optional but useful for testing. If you want to use it, copy it into place and install the Kerberos client utilities:

```
$ cp /var/lib/samba/private/krb5.conf /etc
```

```
$ apt-get install krb5-user
```

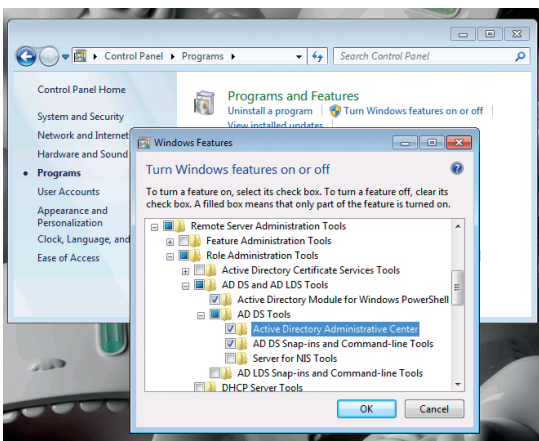
You can then run some basic Kerberos tests (the Samba server needs to be running):

```
# kinit administrator@EXAMPLE.COM
```

```
Password for administrator@EXAMPLE.COM:
```

```
# klist
```

```
Ticket cache: FILE:/tmp/krb5cc_0
```



Installing RSAT is not enough: you must also use Turn Windows Features On Or Off to enable it.

## What is Active Directory?

Active Directory, or its more complete and up-to-date name, Active Directory Domain Services, (ADDS) is a scalable, secure, and manageable infrastructure for user and resource management.

A server that provides ADDS has the ADDS 'Server Role' and is called a 'domain controller'. Its responsibilities include authentication and authorisation of users and computers in a Windows network, the assignment and enforcement of security policies and installing and updating software. The "directory" part refers to a listing of "objects". It's a database that

is managed by a "Directory System Agent" (DSA) and can be accessed using the Lightweight Directory Access Protocol (LDAP); there are also ADSI, MAPI and "Security Accounts Manager" (SAM) interfaces. The objects are either "resources" or "security principals", the latter having unique "Security Identifiers" (SIDs). Unlike the earlier Windows NT domain controllers, it's possible for there to be multiple servers with the ADDS role, all accepting read/write operations and replicating changes to remain in sync.

ADDS uses Kerberos for authentication.

```
Default principal: administrator@EXAMPLE.COM
```

```
Valid starting Expires Service principal
```

```
16/09/14 12:42:07 16/09/14 22:42:07 krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

```
renew until 17/09/14 12:41:56
```

We can use the Samba client tool to browse our domain's shares. We can list them and connect to them to see their contents (you'll need to enter the password that you chose during provisioning).

```
$ smbclient -L localhost -U%
```

```
$ smbclient //localhost/sysvol -U'Administrator%Pa$$w0rd' -c ls
```

Another way to access shares is to mount them using the cifs filesystem:

```
$ mount -t cifs -o username=Administrator,password='Pa$$w0rd' //samba/sysvol /mnt
```

## Serving time

Participants in an Active Directory domain work best when they have synchronised time clocks because Active Directory uses Kerberos for authentication, and this is extremely time-sensitive. There is an allowed tolerance of five minutes and any more than this will result in denied access. It's also essential if you have multiple servers because directory replication relies on synchronised clocks. Implementing a time server will allow clients attempting to connect to our server to synchronise their clocks from it.

Microsoft uses an extension to the standard Network Time Protocol that uses signed timestamps. It calls this the "Windows Time Service". The standard **ntpd** time server can provide such times by having Samba sign its timestamps. Install the daemon from the repository:

```
$ apt-get install ntp
```

Modify the configuration file so that **ntpd** asks Samba to sign its timestamps. You need to define the socket where the signing agent listens and add a server restriction so that requests get signed by default. If you're using a virtual server, such as LXC, you can replace the whole **/etc/ntp.conf** with the following example, otherwise amend your existing configuration so that it includes the last two lines. Restart the daemon after making your changes (**service ntp restart**).

### LV PRO TIP

If you want to use less secure passwords: **samba-tool domain passwordsettings set --complexity=off**

### LV PRO TIP

Borked install? Just delete **/etc/samba/smb.conf** and **/var/lib/samba/private** and start over.

## Network configuration

Your Samba server needs a static IP address and it should be configured to use Samba as its primary DNS name server. You can use a static IP configuration to achieve this by editing `/etc/network/interfaces.d/eth0.cfg` so that it reads like this:

```
auto eth0
iface eth0 inet static
address 10.0.100.1
netmask 255.0.0.0
gateway 10.0.0.138
dns-nameservers 10.0.100.1 10.0.0.138
dns-search example.com
```

You'll need to use values appropriate to your own network. Our server's interface is `eth0` but yours may be different and you should use your own domain name and an IP address appropriate to your network.

You can use DHCP if you prefer but you will need to make sure that your DHCP server always assigns the same IP address to your network interface. You can get your network's interface (MAC address) by doing:

```
$ cat /sys/class/net/eth0/address
```

You'll need to prepend the settings supplied by DHCP with the local DNS server entry and

there are various ways to do this. One way on Ubuntu is to add it to `/etc/resolvconf/resolv.conf.d/head` like this:

```
nameserver 10.0.100.1
```

You should also set a host name in `/etc/hostname` and its fully-qualified domain name in `/etc/hostname`. We're using `samba.example.com` so our `/etc/hostname` file contains just the host name, like this:

```
samba
```

and our `/etc/hosts` file contains a line like this:

```
127.0.1.1 samba.example.com samba
```

The easiest way to ensure your network settings take effect is to reboot after making them so that the `/etc/resolv.conf` file that DNS relies on is updated. You can then confirm your settings:

```
$ hostname
samba
$ hostname -f
samba.example.com
$ cat /etc/resolv.conf
nameserver 10.0.100.1
nameserver 10.0.0.138
search example.com
```

```
server 127.127.1.0
fudge 127.127.1.0 stratum 12
ntpsigndsocket /var/lib/samba/ntp_signd/
restrict default mssntp
```

Our example uses `127.127.1.0` as a time server address. This is a pseudo-address that NTP recognises as its own local clock and synchronises with itself. This is sufficient inside a virtual server whose clock is controlled by the VPS host.

The `ntpsigndsocket` entry defines the path to the directory where Samba places the socket file on which it will listen for signing requests. The path is determined by Samba's configuration and you can confirm the correct path with:

```
$ samba-tool testparm --verbose --suppress-prompt | grep "ntp signd socket directory"
```

```
ntp signd socket directory = /var/lib/samba/ntp_signd
```

Samba creates the socket directory but you should ensure that it is writeable by the `ntpd` daemon, which usually runs as `ntp:ntp`. You should change the directory's group to match:

```
$ chgrp ntp /var/lib/samba/ntp_signd
```

Unfortunately there is no tool to test NTP authentication from Linux but we can do so when we connect our first Windows client to our Samba server. The following examples assume that you have a clean install of Windows 7, and bear in mind that you can't join a domain from the Starter or Home editions although you'll still be able to access shares.

There are a couple of prerequisites before a client can join the domain. The first is that it must use the Samba server's DNS. The second requirement is for its clock to be reasonably consistent with the Samba

server, say within a few seconds, otherwise errors may be reported that bear no relationship to the real problem and you will not be able to authenticate. The Windows time service will keep the clocks synchronised once the client becomes a domain member. We'll assume you know how to make these tweaks or know a Windows tech who does.

Now, to add the client to the domain, go to Start > Computer > Right-click > Properties > Change Settings. This will display the System Properties dialog, where you should click on the Change button and then select Domain in the Member Of section and enter the Samba domain name before pressing 'OK'. This should request the administrator account credentials (the username is 'Administrator' and password is 'Pa\$sw0rd' if you've followed our example settings). It should finish by welcoming you to the domain and asking you to restart the computer.

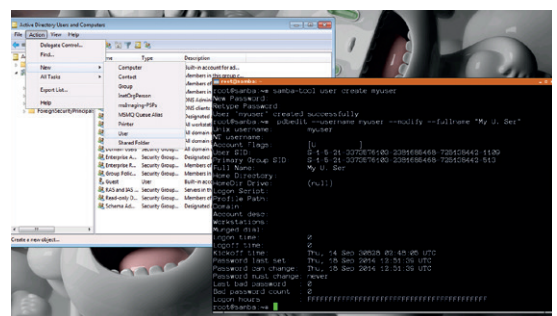
Log in as your domain administrator, (EXAMPLE\Administrator), when Windows restarts. You can now test NTP. Open a command prompt window as the Administrator (click the Start button, type `cmd` and then right-click the `cmd` icon that appears in the search results to select Run As Administrator) and then:

```
C:\> w32tm /resync
Sending resync command to local computer
The command completed successfully.
```

So, we now have Active Directory Domain Services and have joined a client to the domain. What does that give us? Well, we can now use Windows tools to administer our domain, but you need to download the Remote Server Administration Tools and install them. Do this while you're still logged in to your Windows desktop – see <http://bit.ly/ms-rsat>.

The Remote Server Administration Tools include a tool called Active Directory Users And Computers that you can use for your admin tasks. Run this, as an administrator, via the Start button: search for `dsa.msc` (this is the name of the relevant executable file that you need to run). The Action menu lists the various administrative actions that you can perform, such as adding a new user.

You can also perform these tasks using the Samba command line tools if you prefer that way of doing things. The Samba administration utility is called



Whatever your preference, you can get your admin done: you can use the native Windows tools or the various Linux command line alternatives.

### LV PRO TIP

Reload Samba's config without restarting: `smbcontrol all reload-config`. Sanity check it with `testparm`.

### LV PRO TIP

`ntp_signd` is a compile-time option. If signed requests do not work you may need to rebuild `ntpd` from source. This isn't the case with Ubuntu 14.04.

**samba-tool**, and you can use it to add users like this:

```
$ samba-tool user create myuser
```

This creates a user but doesn't enrich it with supplementary data that can be stored in Active Directory, such as their name and phone number, but you can use the **pdbedit** command line tool for that:

```
$ pdbedit --username myuser --modify --fullname "My User"
```

You can edit common user attributes with **pdbedit** but there are many more attributes in the directory that you can access. You'll need a basic grasp of how LDAP stores data and you'll need the LDAP Database Tools to access it. Install the tools and try some queries:

```
$ apt-get install ldb-tools
```

```
$ ldbsearch -H /var/lib/samba/private/sam.ldb -b CN=myuser,CN=Users,DC=example,DC=com
```

```
$ ldbsearch -H /var/lib/samba/private/sam.ldb -b CN=Users,DC=example,DC=com samaccountname=myuser
```

The first argument points at Samba's database – your Active Directory. The second argument is the Distinguished Name (DN) to search within (a DN is what uniquely identifies a record in LDAP and the base DN specifies where to start the search). What follows the arguments is an expression that selects records from the database and fields from those records. If the expression is omitted then everything beneath the base DN is returned. See **man ldbsearch** for more.

Use your preferred method to try adding a user now, we'll make use of **myuser** in the following examples. If you need to edit your user's record then **ldbedit** gives you direct edit access to the directory. Be careful not to alter any internal Active Directory data. You can edit a user like this:

```
$ ldbedit -H /var/lib/samba/private/sam.ldb -b CN=Users,DC=example,DC=com samaccountname=myuser
```

## Linux login

We recommended adding a **--use-rfc2307** option when provisioning the Samba server. RFC2307 is an internet standard that Active Directory implements so that it can store Unix attributes like usernames and passwords in a standard way. The provisioning option instructs Samba to do similarly and this allows us to use Samba to authenticate users that log in to our Linux machines. Microsoft's Active Directory implementation calls this "Identity Management for UNIX". If you want to authenticate users in this way, their computers need **winbind**, a daemon that looks up usernames and passwords in Active Directory. You need to install it, along with libraries that link it into the authentication process:

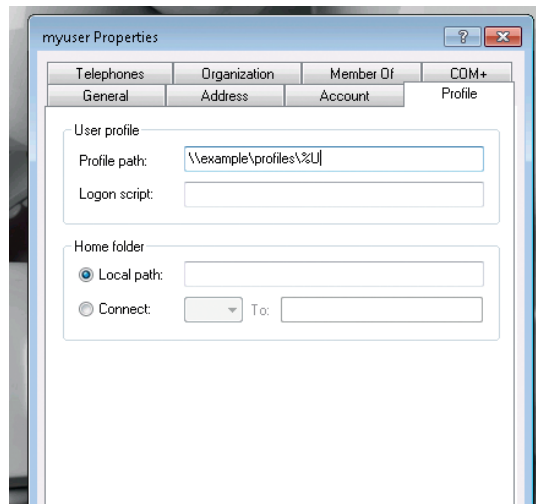
```
$ apt-get install winbind libnss-winbind libpam-winbind
```

NSS is the Name Service Switch and you need to configure it to use **winbind** as a data source by adding it after the options already in place. Our modified Ubuntu **/etc/nsswitch.conf** looks like this:

```
passwd: compat winbind
```

```
group: compat winbind
```

You can test these using **getent passwd** and **getent group**, and you can look up your user with **id**:



```
$ id myuser
```

```
uid=3000021(EXAMPLE\myuser) gid=100(users) groups=100(users)
```

Domain users have high-numbered UIDs that are assigned by Active Directory. You can modify this (or any other LDAP attribute) using **ldbedit** but they're kept separately from the main directory. You need a user's Security Identifier, or SID, to find them. The SID is another way that Active Directory uniquely identifies a user. The commands you need are:

```
$ wbinfo --name-to-sid myuser
```

```
S-1-5-21-3373576103-2381685468-725138442-1109
```

```
SID_USER (1)
```

```
$ ldbedit -H /var/lib/samba/private/idmap.ldb cn=S-1-5-21-3373576103-2381685468-725138442-1109
```


The field that you need to change is **xidNumber**; you can set this to the desired **uid** value. You only really need to do this when moving existing users from **/etc/passwd** into the directory.

You can try logging in as the user you created earlier, for example:

```
$ ssh myuser@my_linux_box
```

## When in Roam...

File and print sharing works exactly as it does when Samba is used in the classic, non-Active Directory, way by writing stanzas in **smb.conf**. One thing that a domain controller adds to this is Roaming Profiles. This feature enables your domain users to log in to Windows clients and download their user profile directory. Think about your users' habits before enabling roaming profiles. Because they are downloaded and uploaded inefficiently, users storing large amounts of data in their profile can put undue pressure on your Samba server.

There's much more to Active Directory than we've covered here, but you should be able to get your first server up and running and save yourself from one more proprietary server. 

Roaming profiles link to the **[profiles]** share configured in **smb.conf**. The **%U** in the path will be replaced with the username.

### LV PRO TIP

If you have Apparmor on your server, check that its configuration allows access to the Samba socket (it does on Ubuntu 14.04). See **/etc/apparmor.d/usr.sbin.ntpd**.

### LV PRO TIP

From a Windows command line, use **ipconfig /all** to check network settings such as DNS.

John Lane provides technical solutions to business problems. He has yet to find anything that Linux can't solve.

**JULIET KEMP**

# SEYMOUR CRAY AND SUPERCOMPUTERS

Join us in the Linux Voice time machine once more as we go back to the 1970s and the early Cray supercomputers.

Computers in the 1940s were vast, unreliable beasts built with vacuum tubes and mercury memory. Then came the 1950s, when transistors and magnetic memory allowed computers to become smaller, more reliable, and, importantly, faster. The quest for speed gave rise to the “supercomputer”, computers right at the edge of the possible in processing speed. Almost synonymous with supercomputing is Seymour Cray. For at least two decades, starting at Control Data Corporation in 1964, then at Cray Research and other companies, Cray computers were the fastest general-purpose computers in the world. And they’re still what many people think of when they imagine a supercomputer.

Seymour Cray was born in Wisconsin in 1925, and was interested in science and engineering from childhood. He was drafted as a radio operator towards the end of World War II, went back to college after the war, then joined Engineering Research Associates (ERA) in 1951. They were best known for their code-breaking work, with a little involvement with digital computing, and Cray moved into this area. He was involved in designing the ERA 1103, the first scientific computer to see commercial success. ERA was eventually bought out by Remington Rand and folded into the UNIVAC team.

In the late 1950s, Cray followed a number of other former ERA employees to the newly-formed Control Data Corporation (CDC) where he continued designing computers. However, he wasn’t interested in CDC’s main business of producing low-end commercial computers. What he wanted was to build the largest

computer in the world. He started working on the CDC 6600, which was to become the first really commercially successful supercomputer. (The UK Atlas, operational at a similar time, only had three installations, although Ferranti was certainly interested in sales.)

Cray’s vital realisation was that supercomputing – computing power – wasn’t purely a factor of processor speed. What was needed was to design a whole system that worked as fast as possible, which meant (among other things) designing for faster IO bandwidth. Otherwise your lovely ultrafast processor would spend its time idly waiting for more data to come down the pipeline. Cray has been quoted as saying, “Anyone can build a fast CPU. The trick is to build a fast system.” He was also focussed on cooling systems (heat being one of the major problems when building any computer, even now), and on ensuring that signal arrivals were properly synchronised.

## CDC 6600: the first supercomputer

Cray made several big architectural improvements in the CDC 6600. The first was its significant instruction-level parallelism: it was built to operate in parallel in two different ways. Firstly, within the CPU, there were multiple functional units (execution units forming discrete parts of the CPU) which could operate in parallel; so it could begin the next instruction while still computing the current one, as long as the current one wasn’t required by the next. It also had an instruction cache of sorts to reduce the time the CPU spent waiting for the next instruction fetch result. Secondly, the CPU itself contained 10 parallel functional units (parallel processors, or PPs), so it could operate on ten different instructions simultaneously. This was unique for the time. The CPU read and decoded instructions from memory (via the PPs), and passed them onto the functional units to be processed. The CPU also contained an eight-word stack to hold previously executed instructions, making these instructions quicker to access as they required no memory fetch.

There were 10 PPs, but the CPU could only handle a single one at a time. They were housed in a ‘barrel’, and would be presented to the CPU one at a time. On each barrel ‘rotation’ the CPU would operate on the instruction in the next PP, and so on through each of the PPs and back to the start again. This meant that multiple instructions could be processing in parallel and the PPs could handle I/O while the CPU ran its

### LV PRO TIP

For some diagrams of this setup and far greater detail about registers and functional units, see this detailed article by James E Thornton: [http://research.microsoft.com/en-us/um/people/gbell/Computer\\_Structures\\_Readings\\_and\\_Examples/00000511.htm](http://research.microsoft.com/en-us/um/people/gbell/Computer_Structures_Readings_and_Examples/00000511.htm).



Seymour Cray with the Cray-1 (1976). Image courtesy of Cray Research, Inc.



arithmetic/logic independently. The CPU's only connections to the PPs were memory, and a two-way connection such that either a PP could provide an interrupt, or it could monitor the central program address. To make best use of this, and of the functional units within the CPU, programmers had to write their code to take into account memory access and parallelisation of instructions, but the speed-up possibilities were significant.

A related improvement was in the size of the instruction set. At the time, it was usual to have large multi-task CPUs, with a large instruction set. This meant that they tended to run slowly. Cray, instead, used a small CPU with a small instruction set, handling only arithmetic and logic, which could therefore run much faster. For the other tasks usually dealt with by the CPU (memory access, I/O, and so on), Cray used secondary processors known as peripheral processors. Nearly all of the operating system, and all of the I/O (input/output) of the machine ran on these peripheral processors, to free up the CPU for user programs. This was the forerunner of the "reduced instruction set computing" (RISC) designs which gave rise to the ARM processor in the early 1980s.

The 6600 also had some instruction set idiosyncrasies. It had 8 general purpose X registers (60 bits wide), 8 A address registers (18 bits), and 8 B 'increment' registers (in which B0 was always zero, and B1 was often programmed as always 1) (18 bits). So far so normal; but instead of having an explicit memory access instruction, memory access was handled by the CPU as a side effect of assigning to particular registers (setting A1–A5 loaded the word at the given address into registers X1–X5, and setting A6 or A7 stored registers X6 or X7 into the given address). (This is quite elegant, really, but a little confusing at first glance.)

Physically, the machine was built in a + -shaped cabinet, with Freon circulating within the machine for cooling. The intersection of the + housed the interconnection cables, which were designed for minimum distance and thus maximum speed. The logic modules, each of two parallel circuit boards with components packed between them (cordwood construction) were very densely packed but had good heat removal. (Hard to repair though!) It had 400,000 transistors, over 100 miles of wiring, and a 100 nanosecond clock speed (the fastest in the world at the time). It also, of course, came with tape and disk units, high-speed card readers, a card punch, two circular 'scopes' (CRT screens) to watch data processing happening, and even a free operator's chair.

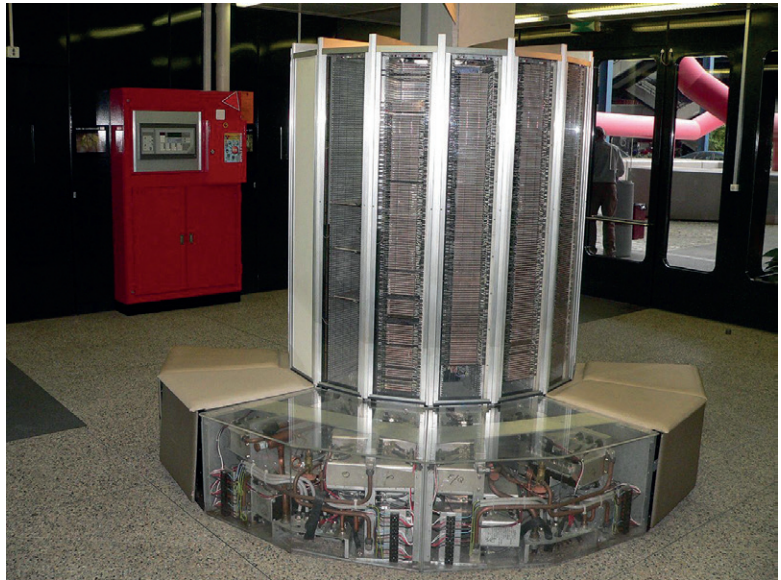
Its performance was around 1 megaFLOPS, the fastest in the world by a factor of three. It remained the fastest between its introduction in 1964 (the first one went to CERN to analyse bubble-chamber tracks; the second one to Berkeley to analyse nuclear events, also inside a bubble chamber), and the introduction of the CDC 7600 in 1969. For more detail check out the book

## Cray-1

From 1968 to 1972, Cray was working on the CDC 8600, the next stage after the CDC 6600 and 7600. Effectively, this was four 7600s wired together; but by 1972 it was clear that it was simply too complex to work well. Cray wanted to start a redesign, but CDC was once again in financial trouble, and the money was not available. Cray, therefore, left CDC and started his own company, Cray Research very nearby. Their CTO found investment on

Wall Street, and they were off again. Three years later, the Cray-1 was announced, and Los Alamos National Laboratory won the bidding war for the first machine.

They only expected to sell around a dozen, so priced them at around \$8 million; but in the end they sold around 80, at prices of between \$5 million and \$8 million, making Cray Research a huge success. On top of that, users paid for the engineers to run it.



Cray-1 with its innards showing, in Lausanne. CREDIT/COPYRIGHT: CC-BY-SA 2.0 by Rama.

(written at the time) *Design of a Computer: CDC6600*  
[www.textfiles.com/bitsavers/pdf/cdc/6x00/books/DesignOfAComputer\\_CDC6600.pdf](http://www.textfiles.com/bitsavers/pdf/cdc/6x00/books/DesignOfAComputer_CDC6600.pdf)

One important change in the Cray-1 was vector processing. This meant that if operating on a large dataset, instead of having an instruction for each member of the data set (ie looping round and swapping in a dataset member each time round the

There's a limit to what you can do on the simulator at the moment, but you could try running a test job, as described on Andras Tantos' blog.

```

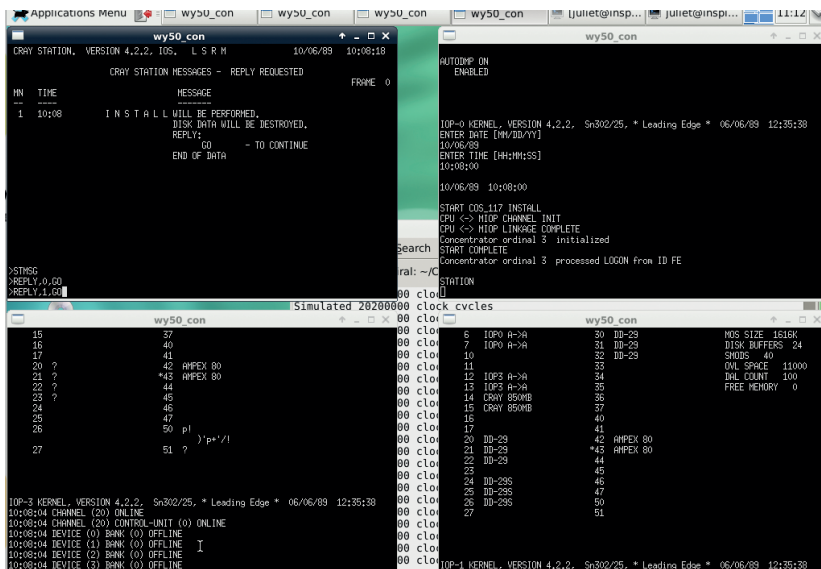
Applications menu  wy50.com  wy50.com  wy50.com  wy50.com  juiet@inspiral: ~
Cray Station, VERSION 4.2.2.2, IOS, L S R M 10/06/89 10:08:06
CRAY STATION MESSAGES - REPLY REQUESTED
IN TIME MESSAGE FRAME 0
1 10:08 INSTALL WILL BE PERFORMED,
  17 41 DISK DATA WILL BE DESTROYED,
  20 42 REPLY: - TO CONTINUE
  21 43
  22 44
  23 45
  24 46
  25 47
  26 50 p1
  27 51 ? )p*/1

TOP-3 KERNEL, VERSION 4.2.2., S#302/25, * Leading Edge * 06/06/89 12:35:38
ENTER DATE (MM/DD/YY) 10/06/89
ENTER TIME (HH:MM:SS) 10:08:00
10/06/89 10:08:00
START COS.117 INSTALL
CPU -> MISC CHANNEL UNIT

juiet@inspiral: ~/Cray/bin
File Edit View Search Preferences Tabs Help
1. juiet@inspiral: ~/Cray/bin
Simulated 16200000 clock cycles
Simulated 16300000 clock cycles
Simulated 16400000 clock cycles
Simulated 16500000 clock cycles
Simulated 16600000 clock cycles
Simulated 16700000 clock cycles
Simulated 16800000 clock cycles
Simulated 16900000 clock cycles
Simulated 17000000 clock cycles
Simulated 17100000 clock cycles
Simulated 17200000 clock cycles
Simulated 17300000 clock cycles
Simulated 17400000 clock cycles
Simulated 17500000 clock cycles
Simulated 17600000 clock cycles
Simulated 17700000 clock cycles
Simulated 17800000 clock cycles
Simulated 17900000 clock cycles
Simulated 18000000 clock cycles
Simulated 18100000 clock cycles
Simulated 18200000 clock cycles
Simulated 18300000 clock cycles
Simulated 18400000 clock cycles

10:08:04 CHANNEL (20) ONLINE
10:08:04 CHANNEL (20) CONTROL-UNIT (0) ONLINE
10:08:04 DEVICE (0) BANK (0) OFFLINE
10:08:04 DEVICE (1) BANK (0) OFFLINE
10:08:04 DEVICE (2) BANK (0) OFFLINE
10:08:04 DEVICE (3) BANK (0) OFFLINE

```



The emulator installing the system. The top-right window shows the 'start' commands, and the top-left is the station window.

**PRO TIP**

Cray's design notes were all written in hand using Boolean notation; the folk assembling the system worked directly from these, without creating schematic diagrams first. See a picture at [www.computerhistory.org/revolution/supercomputers/107/915](http://www.computerhistory.org/revolution/supercomputers/107/915).

loop), the programmer could use a single instruction and apply it to the entire dataset. So instead of fetching a million instructions, the machine only fetches one. But more importantly, it also means that the CPU can use an 'instruction pipeline', queuing instructions up to be sent through the CPU while the previous one is still processing, rather than waiting for it to be fully completed.

Instruction pipelining wasn't new – the 6600 did something rather like this, as did Atlas – but vector processors are able to fine-tune the pipelines because the data layout is known already to be a set of numbers arranged in order in a specific memory location. The Cray-1 took this a step further, using registers to load in a piece of data once and then apply, say, three operations to it, rather than processing the memory three times to operate on it three times. (Specifically, this was an improvement on the STAR.) This reduced flexibility, as registers were expensive to produce and therefore limited; the Cray-1 would have to read in a vector in portions of a particular size. However, the overhead was worth it in terms of the speed increase payoff. The Cray-1 hardware was optimised specifically to get these operations as fast as possible. Cray called this whole process "chaining", as programmers could "chain" together instructions for performance improvements.

The Cray-1 was also the first Cray design to use integrated circuits (silicon chip style circuits) rather than wiring together a bunch of independent transistors. ICs were developed in the 1960s but initially were low-performance. By this time they had become fast enough to be worthwhile. However, they also ran very hot, especially in the huge stacks that were wired together for the Cray-1. (It had 1,662 modules each with one or two boards of up to 144 ICs per board.) The wiring was arranged so as to balance the load on the power supply very neatly. The circuit boards were paired with a copper sheet between them; the copper drew the heat to the outside where liquid Freon drew it away to the cooling unit. This

system took some time to perfect as the lubricant and Freon mix kept leaking out of the seals.

The Cray-1 featured some other nifty tricks to get maximum speed – such as the shape of the chassis. The iconic C-shape was actually created to get shorter wires on the inside of the C, so that the most speed-dependent parts of the machine could be placed there, and signals between them speeded up slightly as they had less wire to get down. Overall throughput was around 4.5 times faster than the CDC 7600.

It was a 64 bit system (the 6600 and 7600 were 60-bit), with 24-bit addressing, and 1 megaword (ie 1M of 64-bit words) of main memory. It had eight 64-bit scalar registers and eight 24-bit address registers, each backed with 64 registers of temporary storage. In addition, the vector system had its own eight 64-element by 64-bit vector registers. There was also a clock register and 4 instruction buffers. Its fastest speed was around 250 MFLOPS, although in general it ran at around 160 MFLOPS.

Also it looked very cool, in a 1970s kind of way, with a central column, in orange and black, containing the processing unit, and a padded bench around it covering the power supplies. It weighed an impressive 5.5 tons (just shy of 5 metric tonnes), and used about 115kW of power (plus cooling and storage).

**Emulators**

Two awesome geeks, Chris Fenton and Andras Tantos, have been working on emulating the Cray-1. Chris's project is to create a desktop size version, and the first part was (fairly) straightforward: build a model of the thing, and find a modern circuit board to hide inside it. Amazing though the Cray-1 was at the time, these days a tablet has more computing power.

But what Chris wanted was real Cray-1 software: specifically, COS. Turns out, no one has it. He managed to track down a couple of disk packs (vast 10lb ones), but then had to get something to read them... in the end he used an impressive home-brew robot solution to map the information, but that still left deciphering it. A Norwegian coder, Yngve Ådlandsvik,

**Other emulators**

There are a few other emulators online, though we haven't tried them ourselves:

- 1 Verilog implementation of Cray-1 for FPGAs <https://code.google.com/p/cray-1x/source/browse/#svn%2Ftrunk%2FSoftware>.
- 2 Desktop CYBER emulator, which emulates various CDC machines but not the Cray-1 or others. It emulates the CDC 6400 but you need a disk image of your own to run an OS on it. <http://members.iinet.net.au/~tom-hunter>. You can also log onto a real live Cray machine online, courtesy of the folks at [www.cray-cyber.org](http://www.cray-cyber.org). Unfortunately at time of writing this service was offline (while they're moving their machines), but it is due back up soon. Many of the machines are only available on Saturdays as they cost so much to run (power bill donations are gratefully accepted). Sadly they don't have a Cray-1 or Cray X-MP; all their Cray machines are later ones that run NOS.

managed to play with the data set enough to figure out the data format and other bits and pieces, and wrote a data recovery script. Unfortunately that disk was just a maintenance image, but another disk was located which did indeed contain an OS image.

This is where Tantos came in; he found that the images were full of faults, so worked on a better recovery tool to reconstruct the boot disk. He's been working on it since (his website has lots of detailed information and links to some of the disk images) and now has an emulator of sorts. In fact, it's not strictly a Cray-1 but a Cray X-MP emulator. The Cray X-MP was an improvement on the Cray-1 design, released in 1982. Andras Tantos has lots of detailed information on it (<http://modularcircuits.tantosonline.com/blog/articles/the-cray-files/the-return-of-the-cray>). The other design path taken by Cray Research at the time led to the Cray-2, a full redesign which wasn't particularly successful. As with all the Cray machines since, the instruction set of the Cray X-MP derives directly from the instruction set of the Cray-1. More pertinently, those were the disks that have been recovered, so that's what we've got.

You can download the most recent zipfile from Tantos' webpage (<http://modularcircuits.tantosonline.com/blog/articles/the-cray-files/downloads>). It has Windows binaries, but for Linux you'll have to compile it yourself, as per these steps (on Debian stable up-to-date at time of writing):

### Compile steps:

Install the Boost library (exists as Debian package **libboost1.49-all-dev**), **GCC** (**gcc-4.7**, **g++-4.7**, **make**, and **libncurses5-dev**).

Unzip the file into its own directory and **cd** into that directory, then add a line to **sw/common.mak** in the downloaded file, after the **SYSTEM** line (line 26):

```
SYSTEM=linux
```

```
SHELL=/bin/bash
```

Type **make** from the **sw** directory, and wait for a bit. (Tip: if you get an internal G++ compiler error, try increasing your swap.)

Copy the **sw/\_bin/linux\_release** files into **bin/**, then add **~/Cray/bin** to your **\$PATH** in **.bashrc**.

(With thanks to Jonathan Chin for assistance in fixing my compile problems. See also this page in French (<http://framboiseipi.fr/installation-dun-simulateur-cray>) and Tantos' instructions.)

You should now be ready to go. To start it up, **cd bin** and type **cray\_xmp\_sim xmp\_sim.cfg**, and then follow the steps on Andras Tantos' blog to get the system installed. Here's a quick summary:

1 Enter a date and time (before 1999! Year-2000 error here...).

2 Type **START COS\_117 INSTALL**.

3 Type **STATION** once you see the line

```
Concentrator ordinal 3 processed LOGON from ID FE
```

You'll get a new window up: this is your main console (Cray Station) window. Type **LOGON**, then **HELP** to see the available commands.

The screenshot shows a terminal window titled 'wy50\_con' with the following content:

```

CRAY STATION, VERSION 4.2.2, IOS, L S R 01/01/89 12:09:18
CRAY SYSTEM STATUS
CSDN = DEFAULT
QUEUES E I O R S
-----
JSQ DC DATASET CLASS STATUS PRI USED TIME FIELD ID TID
-----
5 IN JGENCAT SYSTEM WAIT-EVT 7,0 0 ***** 56 AP OPERATOR
END OF DATA

>SUBMIT, JTEST30
>SUBMIT, JGENCAT
  
```

Below this, a table of jobs is visible:

Job ID	Job Name	Status	Time	Field	ID	TID
6	IOPO A->A	30	DD-29			
7	IOPO A->A	31	DD-29			
10		32	DD-29			
11		33				
12	IOP3 A->A	34				
13	IOP3 A->A	35				
14	CRAY 850MB	36				
15	CRAY 850MB	37				
16		40				
17		41				
20	DD-29	42	AMPEX 80			
21	DD-29	*43	AMPEX 80			
22	DD-29	44				
23		45				
24	DD-29S	46				
25	DD-29S	47				

Another window shows the following text:

```

AUTODMP ON
ENABLED
IOP-0 KERNEL, VERSION 4.2.2, Sn3
ENTER DATE [MM/DD/YY]
01/01/89
ENTER TIME [HH:MM:SS]
10:00:00
  
```

4 To continue booting, type **STMSG** to see the system messages, then **REPLY,0,GO** to reply to message 0 and tell the system to GO.

5 When the next message pops up, warning that install is about to start, type **REPLY,1, GO**.


6 Installation takes 10–15 minutes on a fast machine. Go make a cup of tea. (Or try out some other commands, like **STMSG,I** to see the info messages.)

7 It's done when **FIXME** gets this working!

When you've installed it once, you can use the deadstart process another time; please see Tantos' blog for details.

Unfortunately, test jobs is all that you can do at the moment; there's still no compiler, libraries, or any of the other parts of the system that would mean actually being able to write and run proper software. Tantos is still hopeful that something may show up, but sadly it is entirely possible that those disks are lost forever. Keep watching the project if you're interested (and get in touch with him if anyone reading this happens to have a Cray disk in their loft!).

### What happened next

Cray-related companies have gone through a multitude of mergers and separations over the years, and Seymour Cray died in a traffic accident in 1996. A company called Cray Inc does still exist, and as of Sept 2014 has just launched the XC40 and CS400 supercomputer and cluster supercomputer systems. These include SSD-based buffering with DataWarp, hoping to solve the current problem that compute power is increasing faster than regular disk-based IO can handle. It's still very much in the game of designing whole system speed that Seymour Cray was so enthusiastic about. 

Juliet Kemp is a scary polymath, and is the author of Apress's *Linux System Administration Recipes*.

The Cray simulator running a job. The top-left window shows the status as the job is processed.

# CREATE A FIREFOX ADD-ON FOR FUN AND PROFIT

Give your web surfing extra power by creating new features for your browser – and get your work seen by millions!

## WHY DO THIS?

- Create more features for *Firefox* to improve your web browsing.
- Find out who's tracking your browsing habits.
- Package your software for a platform that's used by more than 450 million people around the world.

**F**irefox is a great web browser, but there are times when you want a little more functionality than it provides by itself. For these occasions, you can use add-ons to power-up the browser

Like most web technology, *Firefox*'s addons are written in JavaScript, CSS and HTML. So, if you know how to create a web page, you know how to create a *Firefox* add-on. All you have to do is package it in the right way.

Mozilla provides all the tools you need for this in the add-on SDK. You can grab this from <https://add-ons.mozilla.org/en-US/developers/builder>.

Unzip this and move into the directory in creates:

```
unzip add-on-sdk-1.17.zip
```

```
cd add-on-sdk-1.17
```

You should then be able to start the SDK with:

```
source bin/activate
```

This expects Python to default to Python 2, which is the case in most Linux distros. However, if you're using a bleeding-edge distro such as Arch, it may point to Python 3 instead. If you get a Python error, this is probably what caused it. You can solve this by changing the first line of the `cfx` file in the `bin` subfolder to:

```
#!/usr/bin/env python2
```

It will still output a warning, but should work fine.

We also require the *Bash* shell. Again, this is the default in most Linux distros, but if you're using a different shell, switch to *Bash* for this session.

Once you've run this command, you should notice that your command line prompt has changed.

This tells you that the SDK is now running. It isn't permanent, so every time you start a new shell, you'll have to re-run `source bin/activate` to start the SDK

```
main.js
1 var self = require("sdk/self");
2 var buttons = require("sdk/ui/button/action");
3 var tabs = require("sdk/tabs");
4
5 var button = buttons.ActionButton({
6   id: "track",
7   label: "tracker",
8   icon: {
9     "16": "./icon-16.png",
10    "32": "./icon-32.png",
11    "64": "./icon-64.png"
12  },
13  onClick: handleClick
14});
15
16 function handleClick(state) {
17   tabs.activeTab.attach({
18     contentScriptFile: self.data.url("tracking.js");
19   });
20 }
```

The `main.js` file is where you define how your add-on will work, but in our case, most of the processing is done in `trackers.js`.

again. If you're going to be doing a lot of add-on development, you could add this to your *Bash* profile to run it automatically when you start.

The SDK works on a directory basis, so you'll need to create a new directory for your add-on. This can be anywhere. Once you've created a new directory, `cd` into it using your shell with the SDK active and run:

```
cfx init
```

This will create the directory structure and files you need for an add-on. There should now be subdirectories called `data`, `lib` and `main`, and a file called `package.json`. From this directory, you can also use the `cfx` tool that's part of the SDK to launch *Firefox* with your add-on enabled:

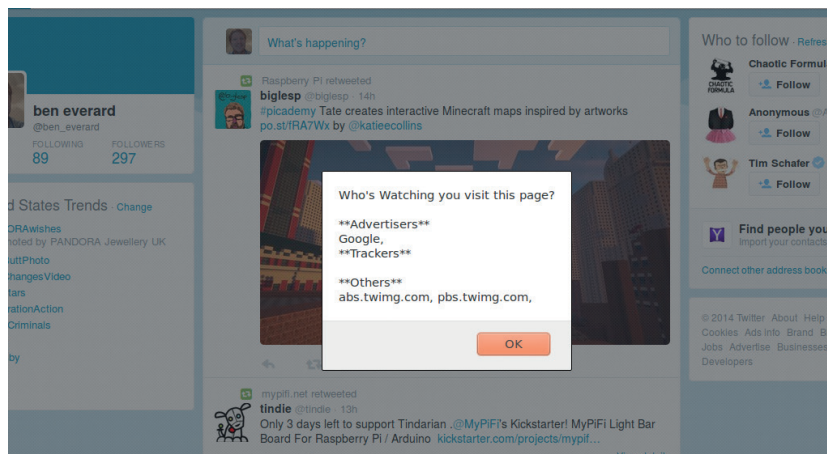
```
cfx run
```

This won't interfere with your normal *Firefox* session, so you can still use that to browse normally while using this second *Firefox* instance for testing your plugin.

## Start coding

Let's now start making an add-on. Our simple test add-on will track which companies are tracking you as you browse the web. Every time you load a website, you download the text, scripts and images separately, often from different servers. With every server you download one of these items from, you reveal your browser's location. Lots of companies exploit this to try and track people as they move around the web then use this data to target advertising. We'll turn this around, and get data about who's tracking us.

Twitter lets Google track users as they view tweets through Google's Analytics.



The first JavaScript file that *Firefox* loads is **lib/main.js**. The SDK creates this when it initialises the directory, but it leaves it blank. Typically, this file is just used to load the appropriate parts of the SDK, then hand off to other JavaScript files that are in the data directory.

For our add-on, we'll need the **tabs** section of the SDK, which enables us to interact with the web pages the user is viewing, and the **self** section, which just lets us load additional scripts. Add the following to **lib/main.js**:

```
var tabs = require("sdk/tabs").on("load", runTracker);
var self = require("sdk/self");

function runTracker(tab) {
    tab.attach({contentScriptFile: self.data.url("tracking.js")});
}
```

This attaches **data/tracking.js** to pages as they are loaded. The terminology here is a little confusing. The **tabs** part of the SDK is used for interacting with web pages rather than the actual tabs on the browser.

In this case, attaching a script means injecting it into the page and running it. This script could do anything that a script in the page could do. That includes things like manipulating the page, inserting or removing elements from the page and sending data to remote servers.

It's in the **tracking.js** file that we'll do all the work, so create this as an empty text file in the data directory. The first thing we need to do is create an area where we can display information to the user. There are many ways to do this, but we'll use a JavaScript alert. This is a simple pop-up that will spring up every time a page is loaded. Our **tracking.js** code for this is:

```
alertText="Who's Watching you visit this page?\n"
// add details of who's watching
```

## Our top five add-ons

- **Ghostery** is like a super-charged version of the add-on created in this tutorial. It also gives you the ability to block trackers, and gives you more details about some of them. However, it doesn't give details on all servers that can see your web traffic like ours does (<https://add-ons.mozilla.org/en-US/firefox/add-on/ghostery>).
- **Firebug** is the most popular add-on for web developers. It adds a host of features to *Firefox*'s already impressive developer's toolset (<https://add-ons.mozilla.org/en-US/firefox/add-on/firebug>).
- **NoScript Security Suite** gives you fine-grained control over what type of scripts which sites can run. This can increase your security, privacy and browsing speed (<https://add-ons.mozilla.org/en-US/firefox/add-on/noscript>).
- **Leech Block** If you're anything like us, you're easily distracted by the web, and can lose hours of productivity on some sites. Leechblock is an add-on to force you to get off those sites and back to work. (<https://add-ons.mozilla.org/en-US/firefox/add-on/leechblock>).
- **LastPass** helps you remember secure passwords to your online accounts. (<https://add-ons.mozilla.org/en-US/firefox/add-on/lastpass-password-manager>).

## Google Chrome

Most major web browsers allow users to extend their functionality in one way or another. In Google's popular *Chrome* browser, this is done through extensions. In many ways these are very similar to *Firefox* add-ons. They're also written in HTML, CSS and JavaScript, and also defined by a JSON file. However, there are different methods available to you to interact with the browser.

If you want to develop for *Chrome*, you'll find all the information you need to get started at <https://developer.chrome.com/extensions>.

The *Opera* web browser is now based on *Chromium*, so extensions work in much the same way as with Google's browser. *Opera* has one additional API for interacting with the Speed Dial (<https://dev.opera.com/extensions/speedial.html>).

### alert(alertText);

We just need to replace the comment line with code that actually locates the various people tracking us.

To do this, you need to get the URL of every element on the page, then loop through these and extract just the domains, then assemble a list of the domains. You can do this with the following code:

```
// add details of who's watching
var spyElements = document.querySelectorAll('img, script')
var domains = [];

for(var i = 0; i < spyElements.length; i++) {
    try {
        var domain = new URL(spyElements[i].src).hostname;
    }
    catch(err) {
        domain = null;
    }

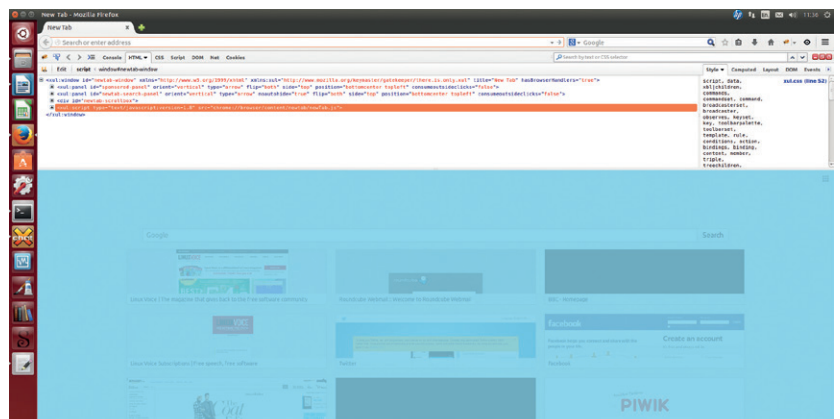
    if(domain && domains.indexOf(domain)==-1 ) {
        domains.push( domain );
        alertText += domain + "\n";
    }
}
```

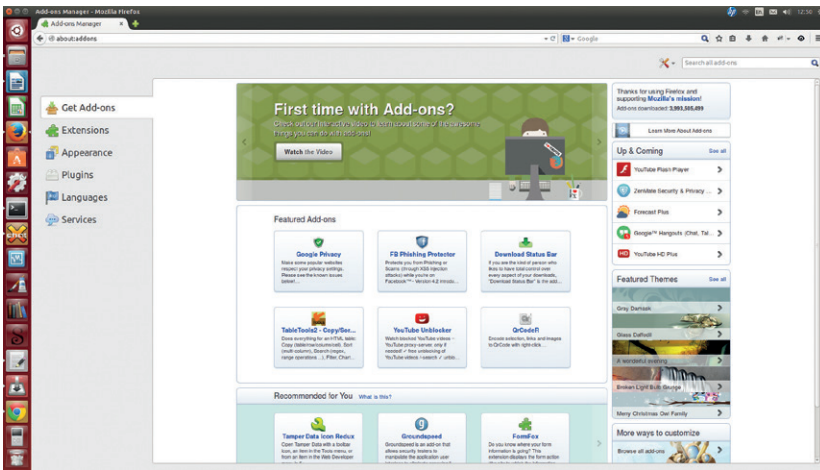
### //sort by type

The **document.querySelectorAll()** function is used to get an array of all the images and scripts on the page. We can get the source (**src**) of these, and from this we need to get the hostname.

There are a few ways of extracting the hostname from a string in JavaScript, but the easiest is just to try and make a URL out of it. If the **src** doesn't have a hostname, this will throw an error, so we put this

The popular *Firebug* add-on gives web developers extra power for debugging web pages.





You can find loads of other add-ons by pointing your browser to **about:add-ons**. See boxout for our recommendations.

line in a **try** block to catch the error. If it does throw an error, we can safely ignore that element because it will be on the same host as the main file and therefore can't be used to track us.

The **if** statement checks that the variable domain isn't null, and that it isn't already in the **domains** array (we don't want to report each domain more than once). If the domain passes this test, we add it to the **domains** array, and add it to the string that is displayed on the screen.

You can test this out by saving the above to **tracking.js**, then running **cfx run** in the root directory of the add-on. You should now be able to browse the web and see who's spying on you.

**Let's tidy it up a bit**

This works, but it's not a very friendly way of outputting the information. After all, domain names may not mean all that much to you. Unless you

happen to know who a particular domain belongs to, and what they use it for, you can't know if there's a problem or not.

The next part we'll add will sort out some of the most popular domains. We'll split them up into Advertisers (companies that make money out of selling advertising), Trackers (companies that make money out of profiling people's browsing habits), and Others (domains we haven't been able to classify).

First we need to set up the data:

```
var advertisers = "";
var others = "";
var trackers = "";
var found = false;
var knownDomains = [['google', 'Google', 'Ad'],
                    ['doubleclick', 'DoubleClick(Google)', 'Ad'],
                    ['facebook', 'Facebook', 'Ad'],
                    ['adnxs', 'AppNexus', 'Tr']];
```

Here, we've just included four domains to keep the code short, but we could easily include as many as we know about.

The next thing we need to do is loop through the domains we've found, and for each one, we'll put an entry in either **advertisers**, **others** or **trackers**. Put the following code at the bottom of **tracking.js**.

```
for(var i=0; i<domains.length; i++) {
    found = false;
    for (var j=0; j<knownDomains.length; j++) {
        if (domains[j].indexOf(knownDomains[j][0]) > -1) {
            found = true;
            if (knownDomains[j][2] == 'Ad') {
                if (advertisers.indexOf(knownDomains[j][1]) == -1) {
                    advertisers += knownDomains[j][1] + ", "
                }
            }
        }
    }
    else {
        if (trackers.indexOf(knownDomains[j][i]) == -1) {
            trackers += knownDomains[j][1] + ", "
        }
    }
}

if (found == false) {
    others += domains[i] + ", ";
}

alertText += "\n**Advertisers**\n" + advertisers +
"\n**Trackers**\n" + trackers + "\n**Others**\n" + others;

alert(alertText);
```

This does everything we need, so you can test it with **cfx run**.

**Getting interactive**

At this stage, we have a working add-on that lets you know who's watching you as you browse online. It could use a little more data on which domains are trackers and advertisers, but otherwise it works. However, it is a little invasive, and when surfing you don't always want a pop-up on every new page.

Rather than automatically injecting the code into every page, you can add a button that just runs this script on the current page whenever it's pressed. This

**"We have a working add-on that lets you know who's watching you as you browse."**

**Themes**

Themes are another form of *Firefox* add-on. They don't add any functionality, but they do make the interface look better (or at least different). The simplest way of creating a theme is with a lightweight theme. These don't have as much scope as full themes, but they also don't require any coding. You just select the images you want and fill in a few details. There's information on getting started with this at <https://add-ons.mozilla.org/en-US/developers/docs/themes>.

Complete themes allow you to interact with the interface XUL (an XML grammar that controls the layout) using CSS. This way you can modify the GUI in far more ways than adding simple images, however, it is more involved than a lightweight theme. You can find out more at [https://developer.mozilla.org/en-US/docs/Building\\_a\\_Theme](https://developer.mozilla.org/en-US/docs/Building_a_Theme).

enables the user to leave the add-on running, but only see who's tracking their browsing when they want to.

To do this, we'll need another API: **ui/ActionButton**. This puts an icon button in the main bar that you can use. The first thing you need for this is an icon for the button. There are loads of sources of icons around under different licences. We like the eyeball icon from [https://www.iconfinder.com/icons/126581/eye\\_eyeball\\_view\\_icon](https://www.iconfinder.com/icons/126581/eye_eyeball_view_icon). You'll need it in sizes 16x16, 32x32 and 64x64 in PNG format. These can all be downloaded from the above website. The icons are by Timothy Miller and released under the Creative Commons Attribution Share Alike licence, so you're free to use them as long as you credit the creator and release any changes under the same licence.

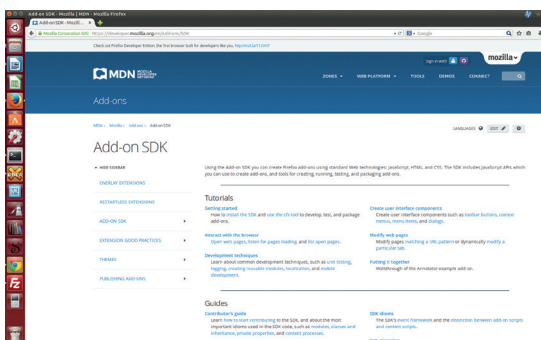
Grab the images, and save them as **icon-16.png**, **icon-32.png** and **icon64.png** in the **data** folder of your add-on. The button is added in **main.js**. Change this to the following:

```
var self = require("sdk/self");
var buttons = require("sdk/ui/button/action");
var tabs = require("sdk/tabs");

var button = buttons.ActionButton({
  id: "tracker",
  label: "tracker",
  icon: {
    "16": ".icon-16.png",
    "32": ".icon-32.png",
    "64": ".icon-64.png"
  },
  onClick: handleClick
});
```

To add an action button, all you need to do is bring in the appropriate part of the SDK, then set the button up with an **id**, **label** and **icon** set. The line **onclick: handleClick** tells the add-on which function we want to run when the user clicks on the icon. This function also needs to be added to the **main.js** file as well with:

```
function handleClick(state) {
  tabs.activeTab.attach({
    contentScriptFile: self.data.url("tracking.js");
  });
}
```



The add-on SDK is well documented at <https://developer.mozilla.org/en/Add-ons/SDK>.

## Useful parts of the SDK API

In our sample add-on, we've only interacted with a few parts of the SDK. The full API is far more complete. Here are some of the most useful parts.

- **add-on-page**: This is where you can create an about page for your add-on.
- **panel**: These are like the JavaScript alerts that we used in the sample add-on, but let you include HTML so they are more useful if you've got complex information to display, or want to include pictures.
- **notifications**: Another type of pop-up. These are desktop notifications that pop up to alert the user to some event. In Linux, they use libnotify, so the appearance will depend on your desktop environment.
- **request**: This is the API for making HTTP requests. It allows you more control over

the request than you would have if you used JavaScript to simply fetch the resource.

- **simple storage**: If you need to store data between browser sessions, then this is the API you need.
- **tabs**: You've seen how it can be used to inject scripts into web pages, but this API can also pull information from pages, or interact with them in other ways.
- **ui**: In our example, we used this to add a button to the user interface, but it can also be used to add other things such as a toggle button, a frame, a toolbar or even a whole sidebar.
- **io/file**: As you can probably guess from the name, this is the API for interacting with the filesystem. It allows you far more control than would usually be possible.

This uses the same **attach()** method that we used earlier, but instead of doing it to each tab when they loaded, we do it to the active tab when the button is pressed. With this added, you can use **cfx run** to start *Firefox* with the new version, and you should be able to inspect who's spying you in the currently active tab by clicking on the eye icon.

## Packaging our add-on

We've now finished coding and the only thing left to do is package the add-on so we can distribute it. All the information about the add-on is in the **packages.json** file in the **add-on** directory. You can edit this to add the appropriate information like this:

```
{
  "name": "LVPrivacy",
  "title": "Linux Voice Privacy",
  "id": "jid1-jBER4uLTx3qzfQ",
  "description": "See who's spying on your web browsing",
  "author": "Ben Everard",
  "license": "MPL 2.0",
  "version": "0.1"
}
```

You can change these for your own add-on. There are full details about what can go in this file at [https://developer.mozilla.org/en-US/Add-ons/SDK/Tools/package\\_json](https://developer.mozilla.org/en-US/Add-ons/SDK/Tools/package_json).

The final thing is to use **cfx** to package your file as an XPI that can then be installed in *Firefox* just like any other add-on. This is done with:

### cfx xpi

This will create a file with the XPI extension which you can then install in *Firefox* by going to Tools > Add-Ons > The settings menu in the top right corner > Install Add-On From file.

**Ben Everard is the best-selling author of *Learning Python With Raspberry Pi*. He hacks robots for fun.**

# CODE NINJA: NOSQL

**BEN EVERARD**

When data gets big, get NoSQL – it'll future-proof your project and enhance your job prospects too!

## WHY DO THIS?

- Understand how huge databases handle billions of transactions.
- Gain the flexibility of not having a schema.
- Use the trendiest database in town.

Before we take a look at NoSQL databases, let's first consider databases in general. Broadly speaking, a database is anything that can store and retrieve data. Most of the common databases use Structured Query Language (SQL) to access and manipulate this data. SQL databases are in the relational class of database. In relational databases, everything is stored in tables, and there are links between these tables known as keys. Each table has a series of columns, and each column has a data type associated with it.

As a quick example, a shop may have a database with tables for customers, orders, invoices, and stock. If you needed to know what items a particular customer had bought, you'd need to link the relevant rows from all the tables to build up a picture of what was going on. SQL makes this linking of the tables very easy. Splitting data up this way means that data isn't duplicated, and so can be easily updated. For example, in this example, a customer's address can

be stored in one table and automatically linked to all orders. This means you can easily find out the current address for a customer on an old order.

**“There are some areas in which relational databases are struggling to keep up.”**

Relational databases

have served the computing world well for a few decades; however, the computing world is changing and there are some areas in which these old-fashioned databases are struggling to keep up.

- **Size** Once a database gets too big to store on a single machine, it becomes complex to store it in a relational database.
- **Performance** At very high transactional levels, the overhead of linking tables together can slow down the database.
- **Flexibility** Changing the table structure can be a complex procedure.

At this point, we should say that the above points are only relevant in the most extreme cases. Relational databases can be very big, very fast (and a little flexible). Unless you're trying to push the boundaries of what your hardware can do, a relational database will probably serve your needs well.

However, if you're starting a new company and hope to be the next Google or Facebook, how do you ensure that your database will scale to a few billion users? The answer is NoSQL. Technically speaking,

```

ben@ben-laptop:~$ mongo
MongoDB shell version: 2.4.9
connecting to: test
> for(i=0;i<10;i++){print("hello world")}
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
>
  
```

MongoDB's JavaScript Shell enables you to create simple programs that interact with the database.

NoSQL can refer to any database that doesn't use SQL, but it's generally used to refer to schema-less databases. These basically consist of one big pot into which you can put any data you want regardless of its format. These are sometimes known as document stores.

## Introducing MongoDB

We'll look at one of the most popular of the document stores, *MongoDB*. In this database, data is stored in JSON-style documents. Each document can be put in the store regardless of what format it's in.

To try this out, you'll first need to install *MongoDB*. In Ubuntu and derivatives, it's in a package called **mongodb**, so you can grab it with:

```
sudo apt-get install mongodb
```

Once that's finished, you can run it with:

```
mongo
```

This will drop you into the *MongoDB* shell. It uses a stripped-down version of JavaScript that you can use to build software, but we won't deal too much with that. As a quick example, we'll add and retrieve a couple of items in completely different formats:

```

db.test.save({writer: "Ben", title: "Code Ninja: MongoDB"})
db.test.save({issue: "11", mag: "Linux Voice"})
db.test.find();
  
```

This automatically creates a database called **test**, then puts two entries in it. The final line retrieves them from the database.



You can also grab particular entries by adding parameters to the **find** function. For example:

```
db.test.find({writer: "Ben"});
```

As you can see, the fact that it is schema-less means you can store and retrieve any information. This means that if your requirements change, you can just put different information in. On the other hand, it means that you can't always be sure what format the data coming out will be. This can have advantages for all sorts of projects. It means you can just start coding your hobby project, and not have to worry about the overhead of changing the design should you wish to, and it also means that a multi-million dollar Internet of Things project won't be rendered obsolete in six months when a new device comes out and needs to store different data.

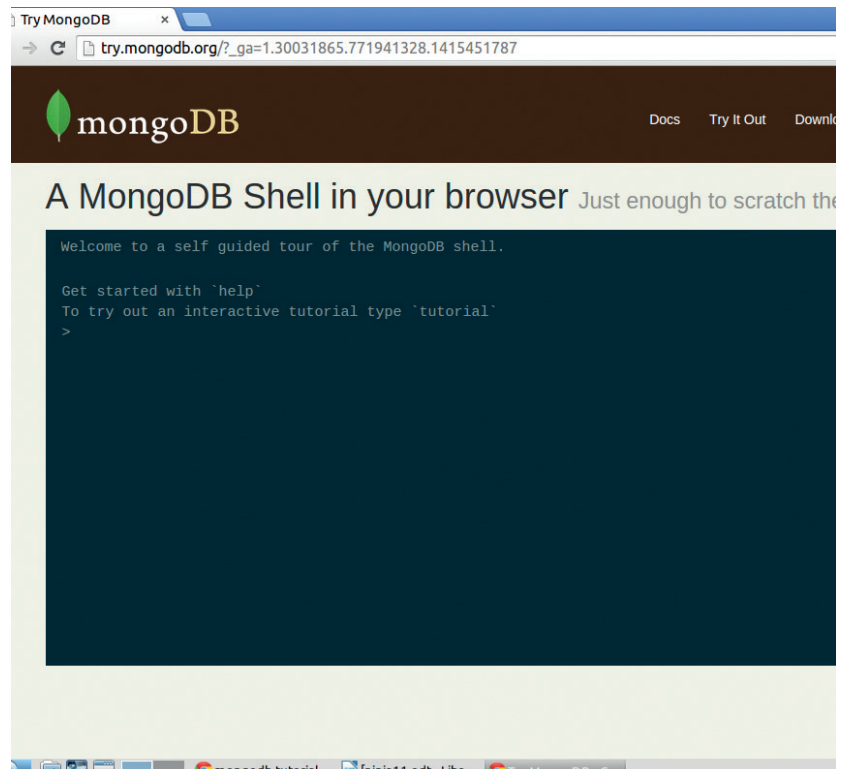
## Linking documents

The concept of linking tables isn't completely gone. Even though there are no tables, you can still reference other bits of data in *MongoDB*. Unlike in relational databases, *MongoDB* gives you a choice. You can do this by linking documents or embedding them. Linking documents works in a fairly similar way to linking tables in relational databases. You just include the document index of one document in another. Embedding is different. When you embed one document in another, you make a copy of the first document inside the second one. This means that the database takes up more space, and it also means that if you update the first document, the second document won't get updates as well. However, it also means that any queries returning the document will be quicker because they only have to find a single document in the store. The actual time it takes to perform the query isn't usually critical on its own, however, it means that you can perform more queries per second on the database server than you otherwise could. If you're serving billions of people who each do hundreds of queries a day, this can make a huge difference.

```
mag = db.test.findOne({issue: "11"});
db.test.save({writer: "Ben", issue: mag, title: "DistroHopper"});
db.test.save({writer: "Ben", issue_id: mag._id, title: "DistroHopper"});
```

This will create two different documents for DistroHopper, one which is linked with a reference (a manual reference in *MongoDB* speak), and the other is embedded. You can see the differences between them by finding them:

```
> db.test.find();
{ "_id" : ObjectId("545e4058035f3795183110d2"), "issue" : "11", "mag" : "Linux Voice" }
{ "_id" : ObjectId("545e44b9e4d4d3f214c587f7"), "writer" : "Ben", "issue_id" : ObjectId("545e4058035f3795183110d2"), "title" : "DistroHopper" }
{ "_id" : ObjectId("545e44e2e4d4d3f214c587f7"), "writer" : "Ben", "issue" : { "_id" : ObjectId("545e4058035f3795183110d2"), "issue" : "11", "mag" : "Linux Voice" }, "title" : "DistroHopper" }
```




The screenshot shows a web browser window with the URL [try.mongodb.org/?\\_ga=1.30031865.771941328.1415451787](http://try.mongodb.org/?_ga=1.30031865.771941328.1415451787). The page features the MongoDB logo and navigation links for 'Docs', 'Try It Out', and 'Download'. The main heading is 'A MongoDB Shell in your browser' with the subtitle 'Just enough to scratch the...'. Below this is a dark-themed terminal window with the text: 'Welcome to a self guided tour of the MongoDB shell.', 'Get started with `help`', 'To try out an interactive tutorial type `tutorial`', and a prompt '>'. The browser's taskbar at the bottom shows several open windows, including 'mongodb tutorial...', 'linia11.odt - Libr...', and 'Try MongoDB - G...'.

You don't have to install *MongoDB* to try it out. There's a web-based version (with a tutorial) available at <http://try.mongodb.org>.

Not only can *MongoDB* perform a query faster on a particular piece of hardware, but it can also spread the load across hardware better. Typically, getting better performance out of an SQL database means buying a faster computer to run it on. This is known as scaling up. You can get better performance out of a NoSQL database by running it across more computers (scaling out). This is too complex a topic to get into in just two pages, but briefly, scaling out makes it easier to manage your database as load increases (especially as it scales up to huge transaction volumes). Again, this is something most users never need to worry about because a good server is powerful enough to run a large *MySQL* database.

Any readers well versed in SQL are probably reeling at some aspects of *MongoDB*, like embedding documents. For anyone indoctrinated with the importance of normalising data, this looks like a terrible violation of all things that are important in a database. In many ways, it is. However, in return for violating these important principals, you get speed and scalability. It's not a tradeoff that always makes sense, but there are occasions when this flexibility can be important.

Perhaps the most compelling reason to learn NoSQL though is the job market. Currently [jobs.com](http://jobs.com) lists NoSQL as the second best trending skill in the jobs market. What's more, if you're still near the start of your career, you won't be competing for jobs with anyone with huge amounts of experience. There are very few people with more than three years NoSQL experience, so it's relatively easy to enter the field. 

# FEEL THE TASTE OF GPU PROGRAMMING

Use your videocard for non-graphics tasks, and discover a whole new programming paradigm.

VALENTINE SINITSYN

**WHY DO THIS?**

- Make your programs run faster.
- Discover new tools for day-to-day tasks.
- Get prepared for the computing way of tomorrow.

People like faster computers. Faster computers means more numbers to crunch per second, and more importantly, fancier user interfaces and eye candy. For the last decade, many PCs came with videocards delivering decent FPS rates in 3D shooters and enough sides on the *Compiz* cube. Wouldn't it be cool to have a supercomputer at home? Perhaps you'd be surprised to learn that you already do (almost). Graphics Processing Units (GPUs) on videocards have many (up to thousands) computing cores, come with fast memory, are optimised for number crunching, and are parallel from the ground up. Sounds like a supercomputer to us!

**Early days**

In the early 2000s, researchers realised that massively parallel GPU architecture works perfect for some scientific problems (eg molecular dynamics). In those days the only available interface to a GPU was OpenGL (or DirectX, for Windows folks). So you needed to express the solution in terms of pixel shaders and texture coordinates. This became known as GP (General Purpose) GPU computing, and this term is sometimes applied to later technologies as well. GPGPU was clever, but neither versatile nor convenient, so Nvidia's CUDA was born in 2007.

CUDA stands for Compute Unified Device Architecture, and it is meant to provide uniform access to Nvidia GPUs (or "devices") for general purpose computations. CUDA programs (by convention, they carry a **.cu** suffix) are written in CUDA C/C++, which is essentially a C language with extensions. You can use other languages as well, but functions to be executed on the GPU (called "kernels" in CUDA parlance; don't confuse them with

Linux kernel) always use CUDA C (unless you're from Fortran camp, but we won't discuss that here). Besides the language and compiler for it (LLVM-based **nvcc**), the CUDA Toolkit (<https://developer.nvidia.com/cuda-toolkit>) contains some other tools and a set of libraries, including accelerated BLAS and Sparse BLAS implementations (the *de-facto* standard in scientific computing).

CUDA is non-free (as in speech). There is also OpenCL – an open heterogeneous (another term to describe CPU+GPU code) computing specification baked by the Khronos Group. They also maintain OpenGL, and there are certain similarities between these two technologies. Where CUDA relies on language extensions, OpenCL is more like a conventional library with API calls. It's also vendor-neutral: OpenCL is available for AMD, Intel, Nvidia and some others, and supports not only GPUs but also multi-core CPUs and specialised hardware. However, OpenCL implementations aren't necessarily open: say, Beignet (for Intel integrated graphics) is free, while AMD's APP SDK for AMD/ATI videocards isn't.

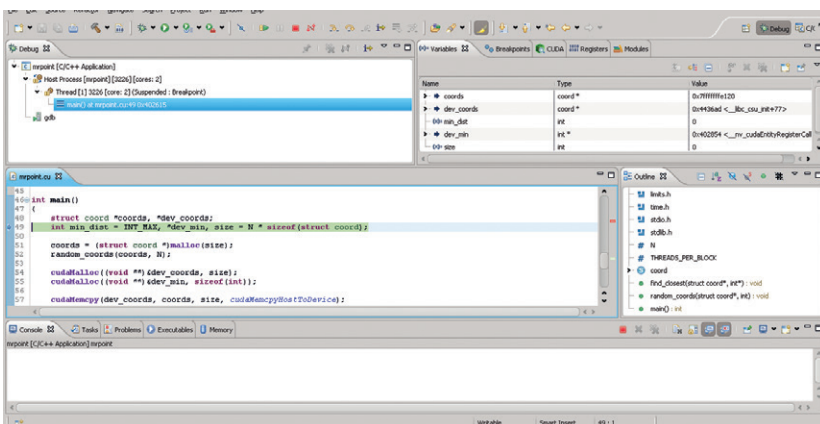
OpenCL and CUDA coexist peacefully: the former is a way to make ideas behind the latter (historically the first) a formal standard. Nvidia supports it, and many applications come both in CUDA and OpenCL editions.

CUDA and OpenCL provide a two-level hierarchical view on the GPU's computational resources. At the lowest level is the basic entity that executes a kernel – called a "thread" in CUDA or a "worker item" in OpenCL. These are combined in three-dimensional "blocks" or two-dimensional "worker groups". There is an upper limit on the number of threads per block (512 or 1024 in CUDA, depending on your card's age). Finally, blocks and worker groups are combined in a "grid" (CUDA) or "index space" (OpenCL). Many programs use one-dimensional blocks and 1x1 grids, which makes up for a simpler geometry.

**A silly example**

CUDA and OpenCL make GPUs accessible for general computing, but this approach also has some limitations. First, the amount of RAM available on many videocards is not very large (somewhere in the region of 4GB is pretty common) or easily extensible. Second, early GPUs (prior to CUDA 1.3, or around 2009) lacked support for double-precision floating point arithmetics. Even where available, it's much slower than single-precision. And there are some

Nsight provides complete Eclipse-based IDE for CUDA programming, including interactive debugger.



tasks that suit the CPU better: GPU computing isn't meant to replace the CPU, but to supplement it.

Say hello to Mister Point. He lives in an unrestricted two-dimensional plane (the poor guy). One day, he comes to his pointy kitchen and spots fire on the curtains. He promptly calls 999 for the fire brigade. As Mr Point resides in highly urbanised area, there are loads of them around, but which one is the closest and quickest to come? This is the question that an emergency phone operator has to answer instantly. As Mr Point is a proud resident of the large city with many blocks, a straight distance between him and a firehouse means next to nothing, and Manhattan norm (the pattern of city blocks) is what we need to consider (ignoring traffic jams for now).

This toy problem demonstrates a typical task that is easy to parallelise. On input, we have a (presumably long) list of fire brigade coordinates (both are pairs of integers to keep things simple). Poor Mr Point is assumed to live at (0, 0). The output is a single integer (the distance to the closest brigade).

Developing a heterogeneous computing program is always about writing kernels and code that launches them, and waiting for the result (besides other things, of course). Let's implement ours for both CUDA and OpenCL, using different host-side languages to feel the difference.

We start with CUDA kernel. For convenience, we declare struct coord (not shown here) which is a pair of integers.

```
#define N (1024 * 1024)
#define THREADS_PER_BLOCK 512

__global__ void find_closest(struct coord *coords, int *closest)
{
    int i = blockDim.x * blockIdx.x + threadIdx.x, t = threadIdx.x;
    __shared__ int dist[THREADS_PER_BLOCK];

    /* Calculate the distance */
    dist[t] = abs(coords[i].x) + abs(coords[i].y);

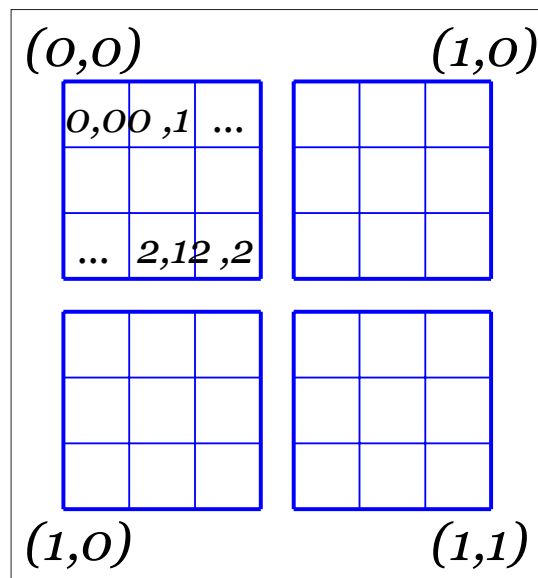
    __syncthreads();
```

## Obtaining the tools

CUDA is (unfortunately) proprietary, but Nvidia maintains repositories for major distros, including Red Hat and family, SUSE, and Debian/Ubuntu. Download and install the repository configuration files (available as Deb or RPM) then add the software through your package manager.

OpenCL is available from different vendors, and installation methods differ as well. With some, you may be lucky enough to find the required libraries in your distro's package repositories. For others, you may need to download a tarball or an unofficial package.

Either way, pay attention to system requirements. CUDA and OpenCL integrate tightly with host-side tools (**gcc** and alike). Although you may be able to run them on a system that isn't officially supported (I do), I'd recommend you stick to the vendor-approved list for production. Red Hat, SUSE, Debian and friends are usually on it. You're also likely need to install a proprietary graphics driver.



An example of a two-dimensional 3x3 CUDA block aligned in a 2x2 grid.

```
/* Put minimum distance for this block in dist[0] */
for (int j = blockDim.x / 2; j > 0; j /= 2) {
    if (t < j && dists[t] > dist[t + j])
        dist[t] = dist[t + j];
    __syncthreads();
}

/* Update global minimum distance, if ours is smaller */
if (t == 0)
    atomicMin(closest, dist[0]);
}
```

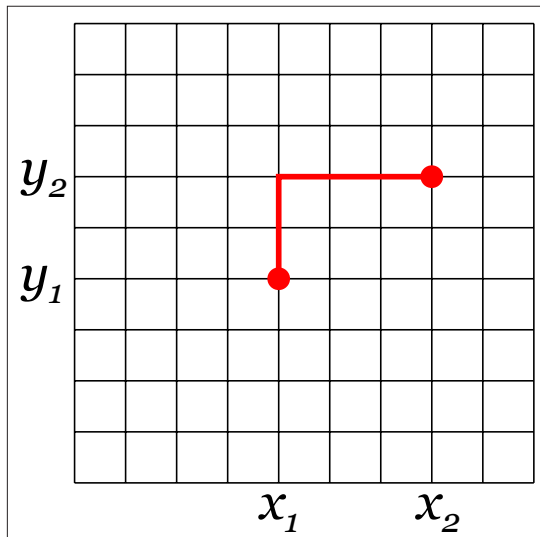
`__global__` designates the function as a kernel that is callable from the host; there is also the `__device__` attribute for GPU-only visible functions, and `__host__`, which is implicit and is used for the host-only code.

A built-in `threadIdx.[xyz]` variable is the block-local thread identifier. Similarly, `blockIdx.[xyz]` locates that block in a grid, and `blockDim.[xyz]` stores the block size in each dimension. For one-dimensional geometries, only the `.x` part of these variables is used.

Each thread calculates the distance for a given fire brigade and stores the result in the `dist[]` array, which is shared between all threads in a block (hence the `__shared` keyword). Then the array is "reduced" to find the block-local minimum (ie the smallest distance among computed distances in the current block). Before the reduction starts, we must ensure that all threads in a block have calculated the distance and `dist[]` is fully filled. This is what `__syncthreads()` (known as a barrier) is for. We also need a barrier at each reduction iteration to guarantee the array is in consistent state.

The way we find a block-local minimum isn't straightforward. We are iteratively taking pairs of values and putting the smaller one at a lower index, until the minimal value is written to `dist[0]`. The reason for this complexity is that GPU is a variant of SIMD (Single Instruction – Multiple Data) architecture, and multiple threads (32 in current CUDA devices) are

The Manhattan (or taxicab) norm is an easy way to measure distance in a rectangular street grid.



running the same instruction but on different input data. If two threads “diverge”, that is, need to run two different instructions (as with many native reduction implementations), it is done in two passes and negatively affects the performance. The algorithm above is the standard way to minimise thread divergence. This being said, the example’s code aims at expressiveness, not the speed.

Finally, if block-local minimum is smaller than current ‘closest’ value, the latter is updated. This way, the smallest per-block distance becomes the result. The first thread in a block (whose `threadIdx.x` is 0) does this, however we can’t simply use `if (dist[0] < *closest) *closest = dist[0]` here. The reason is that thread #0 in another block can interleave between the check and the assignment. To prevent the race, one should use atomic operations, like `atomicMin()` above.

### Launching kernels

Now, let’s turn to the host-side code that launches the kernel. To feel the full taste of CUDA, we’ll do it in native C:

```
int main()
{
    struct coord *coords, *dev_coords;
    int min_dist = INT_MAX, *dev_min, size = N * sizeof(struct coord);

    coords = (struct coord *)malloc(size);
    random_coords(coords, N);

    cudaMalloc((void **)&dev_coords, size);
    cudaMalloc((void **)&dev_min, sizeof(int));

    cudaMemcpy(dev_coords, coords, size, cudaMemcpyHostToDevice);
    cudaMemcpy(dev_min, &min_dist, sizeof(int), cudaMemcpyHostToDevice);

    find_closest<<<N/THREADS_PER_BLOCK, THREADS_PER_BLOCK>>>(dev_coords, dev_min);
}
```

```
cudaMemcpy(&min_dist, dev_min, sizeof(int), cudaMemcpyDeviceToHost);
printf("Mr. Point is at (%d, %d) and the closest brigade is %d units away\n", 0, 0, min_dist);

free(coords);
cudaFree(dev_coords);
cudaFree(dev_min);

return 0;
}
```

Launching kernels with CUDA is a three-stage process. First, input data is copied from the host (main RAM) to device (GPU global memory). They are separate memories, and the cost of the copying (although relatively small) should always be kept in mind. Then, the kernel is launched. Finally, the results are copied back from device to host memory.

`func<<<N, M>>>(args)` is a special syntax for kernel launch. It schedules the kernel on **N** one-dimensional blocks **M** threads each (1x1 grid assumed). There is also an advanced syntax to run kernels on multidimensional blocks and larger grids – consult the CUDA Toolkit Documentation for details. Here, the **TM coords** array is equally split between 2048 blocks 512 threads each. Both values have architecture-defined limits, and you should play with them to see how it affects the performance.

As we are working with C, memory management is manual, and you shouldn’t forget to allocate buffers for input and output data and free them when they are no longer needed.

Now, you can compile and run the program with:

```
nvcc -arch sm_20 mrpoint.cu
./mrpoint
Mr. Point is at (0, 0) and the closest brigade is 8 units away
```

It is recommended that you explicitly set the device architecture to match your card’s capabilities (CUDA 2.0 here), otherwise you may encounter weird bugs.

### The OpenCL way

Let’s now see how the same example can be rewritten the OpenCL way. To make things more interesting, we also switch from C to Python for the host-side.

#### OpenCL vs CUDA

Choosing between OpenCL and CUDA is much like deciding on OpenGL vs DirectX. CUDA is somewhat simpler but Nvidia-only. OpenCL requires more work, but enjoys wider vendor support. The downside of this diversity is that it is harder to optimise your code for each particular device, but you should be able to achieve the same performance with CUDA and OpenCL on the same hardware. There are some discrepancies in feature set (mostly minor), and CUDA has somewhat more advanced tools.

For in-house application targeting Nvidia hardware, we’d probably choose CUDA because of its features and consciousness of API. For a less biased comparison, visit Andreas Klöckner’s (the maintainer for both PyCUDA and PyOpenCL) wiki page at <http://wiki.tiker.net/CudaVsOpenCL>.

For OpenCL, the kernel looks almost the same:

```
__kernel void find_closest(__global struct coord *coords, __
global int *closest)
{
    int i = get_global_id(0), t = get_local_id(0);
    __local int dist[THREADS_PER_BLOCK];
    dist[t] = abs(coords[i].x) + abs(coords[i].y);

    barrier(CLK_LOCAL_MEM_FENCE);

    for (int j = get_local_size(0) / 2; j > 0; j /= 2) {
        if (t < j && dist[t] > dist[t + j])
            dist[t] = dist[t + j];
        barrier(CLK_LOCAL_MEM_FENCE);
    }

    if (t == 0)
        atom_min(closest, dist[0]);
}
```

Underscored markers look a bit different, and we need to explicitly say that the arguments come from global memory. Instead of built-in variables, functions are used to get indices (also note that OpenCL provides a direct way for this with no maths involved).

`__local` declares worker group shared memory, and `barrier()` creates a barrier (we synchronise local memory access only, as it is where `dist[]` is). Otherwise, the kernel stays pretty the same.

Launching it, however, is more involved process, although PyOpenCL hides some complexity. Compared to CUDA, OpenCL provides no high-level API – that’s the price to be paid for flexibility and multivendor support.

```
import numpy as np
import pyopencl as cl

N = (1024 * 1024)
THREADS_PER_BLOCK = 256
ctx = cl.create_some_context()
queue = cl.CommandQueue(ctx)
program = cl.Program(ctx, kernel_src).build()
coords = np.random.randint(-8192, 8192, size=(N, 2)).
astype(np.int32)
min_dist = np.array([2147483647]).astype(np.int32)
mf = cl.mem_flags
dev_coords = cl.Buffer(ctx, mf.READ_ONLY | mf.COPY_HOST_
PTR, hostbuf=coords)
dev_min = cl.Buffer(ctx, mf.READ_WRITE | mf.COPY_HOST_PTR,
hostbuf=min_dist)
program.find_closest(queue, (N,), (THREADS_PER_BLOCK,),
dev_coords, dev_min)
cl.enqueue_copy(queue, min_dist, dev_min)
print "Mr. Point is at (%d, %d) and the closest brigade is %d units
away" % (0, 0, min_dist)
```

First, we obtain a context encompassing all OpenCL devices in the system. PyOpenCL provides a convenient wrapper for this. We also need a queue to push commands to the OpenCL driver. Then the program is built; `kernel_src` is a string containing its source (you could use string formatting to pass



`THREADS_PER_BLOCK` in). There were no stringified sources in CUDA thanks to `nvcc`, but with PyCUDA it would look similar – consider reading external files if it doesn’t look neat. PyOpenCL integrates with NumPy, and we use `numpy.array` for data exchange.

To execute a kernel, you call a method on the `program` object. The parameter list contains the global and local sizes and the kernel’s arguments. Note that the global size is the total number of worker items in OpenCL (not blocks, as in CUDA), and we could choose to pass `None` later if we wanted the runtime to choose the appropriate local size for us. To get the result, we explicitly enqueue the copy operation. Mr Point’s trouble was, of course, a simple example. However, with some generalisations it may form a building block for a more complex task like classification or character recognition.

### Everyone’s covered

At this point you may think: “GPU’s benefits for scientific computing are clear, but I’m not into it, so why should I care?” Glad you asked. While the APIs certainly target writers of high-performance code, there are tools ready that are useful for non-programmers as well.

Administrators can secure their networks with *Suricata* IDS/IPS (<http://suricata-ids.org>), which uses CUDA to speed up protocol, file etc detection in network traffic. You still need a decent network card to capture packets quickly, but GPU processing will help you to discover potential threats faster. There are also many WPA/ZIP file/whatever else password recovery utilities: a tool like *Hashcat* (<http://hashcat.net>) would certainly have improved your chances of winning the LV’s Password Cracking Challenge. There are other legitimate uses for these tools but keep in mind that are in the bad guys’ arsenal as well, and don’t forget to use strong passwords (run some of these password crackers on your password file to see if you are already in danger).

GPU computing has many other applications in medicine, engineering and even finance, and we’ll certainly see more in the future. Stay tuned! 📺

**Dr Valentine Sinitsyn spends half of the day in university where he teaches students physics and diagonalizes large matrices.**

There are specialised massively parallel accelerator boards, like Intel Xeon Phi, or Nvidia Tesla found in this author’s new server. They are fully supported by CUDA and/or OpenCL.

You wouldn't want other people opening your letters and your data is no different. Encrypt it today!

# CIPHERSHED: ENCRYPTION FOR EVERYONE

TrueCrypt lives on as CipherShed, so it's still really easy to protect your valuable data.

JOHN LANE

Everybody has something to hide. It might be a little more mundane than what our governments get up to but, to each of us, that something is important and valuable enough to protect. It could be your personal finances, or perhaps that new app or book you've been working on. If your laptop were stolen, it would be pretty useless if your precious data were encrypted. The good news is that it's easy and, this month, we show you how.

One of the best freely available encryption tools over the past decade was *TrueCrypt*. It provided on-the-fly filesystem encryption and was a cross-platform solution that worked, not only on Linux, but on Windows and Mac OS X too.

Back in May, *TrueCrypt* as we know it ceased to exist. Its SourceForge site was replaced with some basic pages claiming that it is "insecure and may contain unfixed security issues". It now only provides guidance for migrating away, and the only download available is for version 7.2, a limited functionality version that can only decrypt. However, general opinion is that these claims are unfounded and the original developers just asserted their right to kill the product. But the free software community is making sure that the story doesn't end there.

The latest news is, of course, that *TrueCrypt* has been forked and is moving forward as *CipherShed*; you

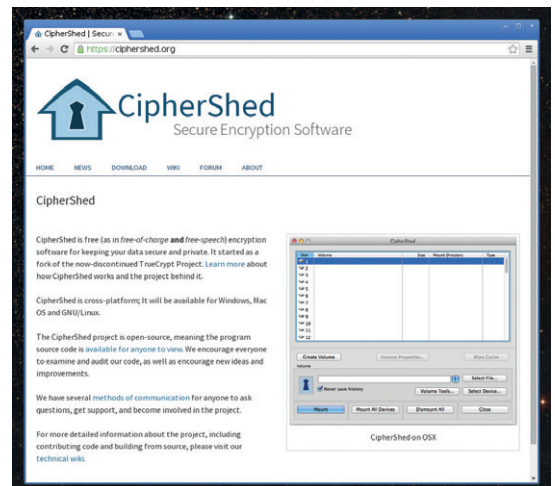
## AES new instructions

If your computer has a recent Intel or AMD processor then it may support AES-NI. This is a set of new x86 CPU instructions that provide hardware-accelerated AES encryption, allowing encryption tasks to be performed four to eight times faster.

Support for AES-NI was introduced with *TrueCrypt* version 7.0. You can check whether your system supports it by looking at Settings > Preferences > Performance. If you have support but prefer not to use a proprietary encryption mechanism then you can disable it on the same screen.

## LV PRO TIP

Truecrypt / Cyphershed requires root privileges. If you can "sudo" then you'll be ok.

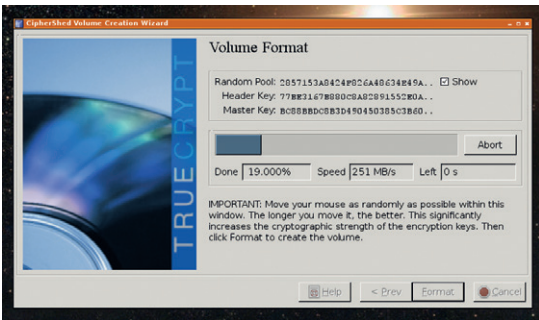


*CipherShed's* goals include a secure audited codebase that is released under an OSI-approved licence.

can install the current development version from its GitHub repository. It's based upon and named similarly to *TrueCrypt* version 7.1a. There are instructions for installing on Debian-based distros, and Arch Linux users can build it from a package in the Arch User Repository. Here's what you need to do if you're on Ubuntu or another Debian-based distro:

```
$ sudo apt-get git build-essential
$ sudo apt-get install libwxgtk2.8-dev nasm libfuse-dev
$ git clone https://github.com/CipherShed/CipherShed.git
$ cd CipherShed/src
$ LIBS="-ldl" make
$ sudo install -m755 {Main,/usr/bin}/ciphershed
```

So, what's it all about? Well, you get very straightforward encryption tools that you can configure using a GUI or command line interface. They make encrypted filesystems either on real disks or partitions, or as virtual disks contained within a file and mounted as a real disk. It claims to offer plausible deniability by creating volumes hidden undetectably



Moving your mouse rapidly provides entropy, which is used to generate encryption keys.

within others and can even boot operating systems hidden in this way.

This is transparent encryption. Once you set up and mount an encrypted volume, you use it just like any other. Copy files there, work on them, edit them, delete them. Do whatever you would do with unencrypted files. All the while, the encryption happens in the background. Once you unmount the volume, the data inside is secure.

### Your first encrypted volume

Begin by typing `truecrypt` or `ciphershed` at a command prompt to start the GUI. Press the Create Volume button to launch the Volume Creation Wizard. This offers two choices – you can either create an encrypted file container (a virtual encrypted disk within a file), or you can create a volume on a partition/drive – essentially any valid block device that you can create a filesystem on.

The first option is best to experiment with; select it and press “Next” to proceed. Now choose between a standard or hidden volume. Choose a standard volume and, on the following page, a location for it.

You’re then offered the choice of several encryption and hashing algorithms, but the defaults offer an appropriate balance between speed and security. If you’re paranoid, choose a stacked scheme like “AES-Twofish-Serpent” – these apply multiple algorithms one after the other but result in slower read/write times.

### Also consider...

There are other *TrueCrypt* derivatives besides *CipherShed* that you may also like to try.

*VeraCrypt* contains enhanced security algorithms that, the developers claim, make it immune to new developments in brute-force attacks and solves vulnerabilities found in *TrueCrypt*. These enhancements, however, mean its storage format is incompatible with *TrueCrypt*. Read more on their website at <http://sourceforge.net/projects/veracrypt>.

*Realcrypt* is essentially *TrueCrypt* with the branding changed. It’s available for Fedora users in the RPM Fusion repository <http://rpmfusion.org/Package/realcrypt>.

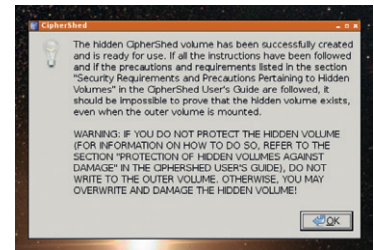
*Tcplay* is a free BSD-licensed command-line *TrueCrypt* implementation based on the Linux kernel’s `dm-crypt` device mapper (<https://github.com/bwalex/tc-play>). It is compatible with *TrueCrypt* volumes.

### Plausible deniability

People keen on privacy and encryption are well aware that the weak link is the person. You can have the strongest keys but they won’t protect you from being forced to reveal them. Being able to reveal a fake key in such scenarios is attractive and a *TrueCrypt* hidden volume enables you to do just that.

A hidden volume is created within the free space of a normal volume in such a way that it cannot be detected. Each has different passphrases and the volume that gets mounted is selected by the given passphrase. This feature would allow someone under duress to give out the normal volume passphrase, allowing access to whatever seemingly important files were placed there while leaving the true secrets

protected within the hidden volume, affording plausible deniability in the event that it should be necessary.



Hidden volumes offer an added layer of protection – just be careful not to overwrite it.

The final pages set the volume’s size and password (this can be a passphrase – using single words is insecure). You can also use keyfiles to enhance security. These are just normal files that can be on your hard drive or removable media. The first 1024 kilobytes of each key file is considered as part of the passphrase that is required to unlock the volume (you should therefore only choose files that won’t change). You can leave the password field empty if you use keyfiles, although it’s less secure if you do. The final choice you have is the encrypted volume’s filesystem and this can be Windows FAT format or Linux ext2–4.

You then land on the Volume Format screen, which will compute a volume key before formatting. It invites you to move your mouse around as a way of gathering entropy for the key.

### Favourite mounts

You need to mount devices before you can use them. The Volume section of the main window is where you select a file or device and use the Mount button to mount it. This is when you need to supply the passphrase and any required key files. Once the device is mounted, it’s accessible as a subdirectory of `/media` and you use it like any other filesystem.

You can optionally cache the passphrases and key files in memory to avoid having to re-enter them on successive mounts. The cache only persists while the encryption driver is running. You close the *CipherShed* GUI by pressing its Exit button. If you have mounted volumes, *CipherShed* goes into the background and presents itself as a taskbar icon that you can use to re-open the main window or quickly mount/unmount favourite volumes via a right-click pop-up menu. *CipherShed* terminates if you exit when there are no mounted volumes in place.

*TrueCrypt* has, for a long time, been one of the easiest ways to use some of the most secure methods available for encrypting sensitive data. *CipherShed* aims to continue that legacy and should mean that we’ll be able to continue securing our data with an easy-to-use GUI desktop application.

### PRO TIP

You can download the *TrueCrypt 7.1a User’s Guide* PDF <http://bit.ly/tc71a Ug>.

# KEEP YOUR DATA SAFE WITH ENCRYPTION

Linux has baked-in encryption capabilities. Use them or regret it when your laptop gets stolen.

JOHN LANE

The Linux kernel has a feature called a device mapper. It allows virtual block devices to be created that are based on other block devices, and there's a device mapper module called **dm-crypt** that we can use to create encrypted block devices.

The device mapper allows devices to be stacked. You can, for example, create RAID or LVM devices and then encrypt them. You can also do it the other way around. You need userspace tools to work with the **dm-crypt** kernel module. The primary one, **cryptsetup**, is used to administer encrypted volumes and requires root privileges. The other tool is **cryptmount**; it allows unprivileged users to mount encrypted volumes.

Encrypted volumes can either be formatted or raw. Formatted volumes contain metadata that describes the encrypted payload, whereas raw volumes are just encrypted disk blocks. Use of raw volumes requires things like ciphers, keys, etc, to be provided as command line parameters; they should be considered as an expert-level option.

The standard volume format on Linux is called LUKS – the Linux Unified Key Setup format.

**Cryptsetup** also supports the *TrueCrypt* format. The LUKS format uses a header at the start of the volume that contains metadata including cipher details and eight key-slots. You can have up to eight different salted, hashed and changeable pass phrases that decrypt a master key to unlock the data payload. LUKS automatically configures non-default **dm-crypt** parameters to make it more secure. The format occupies a header that can consume between 1 and 2MB of the volume's capacity.

Begin by installing the userspace tools; your distro should carry them in its package repository:

```
$ sudo apt-get install cryptsetup cryptmount
```

We'll begin by using **cryptsetup** to format a block device with LUKS.

## A block device in a file

You aren't restricted to real block devices – you can create an encrypted volume in a regular file. To do this, just create a file of whatever size device you want:

```
$ head -c 100M /dev/urandom > /path/to/myvolume
```

You can then use the file's path wherever **cryptsetup** expects a device.

## \$ cryptsetup luksFormat /dev/mydevice

The default cipher that you get depends on the version of **cryptsetup** that you have. Since version 1.6.0, this is **aes-xts-plain64**, where **aes** is the cipher and **xts** is the chaining mode that affects how the cipher is applied to subsequent blocks of data. **xts** is an improvement over the **cbc** mode used by prior versions. Go with the defaults unless you have reason to change them; you can specify an alternative with the **--cipher** command line argument.

So far, we have an encrypted block device but it needs to be opened (you'll be asked for the pass phrase). You can then put a filesystem onto it and mount it:

```
$ cryptsetup open /dev/mydevice myvolume
```

```
$ mkfs.ext4 /dev/mapper/myvolume
```

```
$ mount /dev/mapper/myvolume /mnt
```

**myvolume** is how the device mapper will identify the unlocked device; you can use any meaningful label. To take an encrypted filesystem offline, unmount and then close it:

```
$ umount /mnt
```

```
$ cryptsetup close myvolume
```

## Boot configuration

You can automatically unlock encrypted devices when your system boots. The encrypted device table is a file called **/etc/crypttab** that is similar to the **/etc/fstab** used for mounts. You specify four things: a device mapper name, the device path, an optional key file (or just "none") and options. Use the "luks" option to specify the format:

```
myvolume /dev/mydevice none luks
```

The listed volumes will be opened at boot time and this will require entry of the pass phrases. An alternative to passphrase entry is to store it in an appropriately secured key file:

```
$ sudo echo -n 'my secret passphrase' > /root/keyfile
```

```
$ sudo chmod 0400 /root/keyfile
```

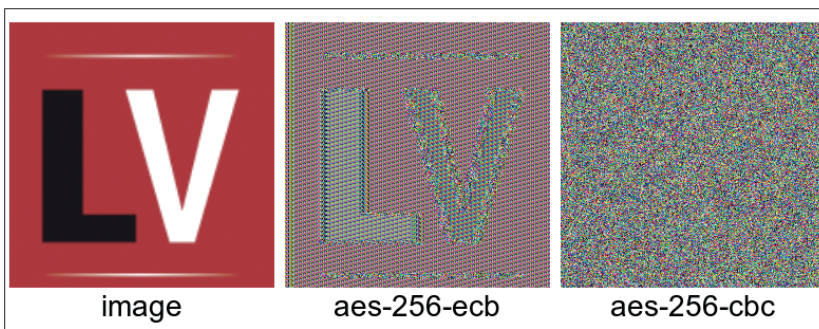
You can, if you want, use a more complex key now that it won't need to be manually entered. It's

### LV PRO TIP

The LUKS default header size of 2MB maintains sector alignment of the LUKS volume with the underlying device. LUKS uses 512-byte blocks.

### LV PRO TIP

Earlier **cryptsetup** versions required **luksOpen** and **luksClose** instead of **open** and **close**.



Choose an appropriate cipher: an encrypted bitmap can reveal cryptographic weaknesses.



customary to use a large blob of random data:

```
$ head -c 4096 /dev/urandom > /root/keyfile
```

Specify the key file in `/etc/crypttab` instead of “none” and add it to the LUKS header – you’ll need to enter an existing passphrase to unlock it before the new one can be added:

```
$ cryptsetup luksAddKey /dev/myvolume /root/keyfile
```

Once a volume is opened, it can be mounted in the usual way with an entry in the `/etc/fstab` file.

You can use `/etc/crypttab` for all filesystems and swap devices. However, if you want to encrypt your root partition, then your system’s `initrd` will need cryptography support. You should refer to your distro’s docs for more information about this because boot configurations vary. The system’s BIOS needs to read the boot partition, so that cannot be encrypted.

### Encrypted filesystems

Another approach to encryption is to use Encrypted Filesystems. These are virtual filesystems stacked on top of existing ones. They provide cleartext read/write access to encrypted files stored in the underlying filesystem. Encrypted filesystems work at the filesystem level, whereas `dm-crypt` operates at the block level, beneath the filesystem.

Two encrypted filesystems available for Linux are eCryptFS and EncFS; the latter runs entirely in userspace (it’s FUSE-based) and, therefore, doesn’t require elevated privileges to use it. Using EncFS is straightforward. You specify an encrypted directory where your files will be stored and an unencrypted directory where you’ll read and write them:

```
$ encfs ~/.mysecrets_encrypted ~/.mysecrets
```

You need to use absolute paths. Follow the instructions: select the “Pre-Configured Paranoia Mode” for suitable defaults or, for more control, use the advanced mode. When your shell prompt returns,

### cryptmount volumes for unprivileged users

The other userspace tool for `dm-crypt` is called `cryptmount`, which offers an easy way for unprivileged users to use encrypted volumes. Root privileges are required to create an encrypted volume for an unprivileged user but that user can mount and unmount it without any special privileges.

```
$ sudo cryptmount-setup
```

Follow the prompts – you need to enter a volume name, username and absolute paths to a mount point and the file that will contain the encrypted volume. Both are created automatically and an ext3 filesystem is created inside the file. The volume’s configuration is written into `/etc/cryptmount/cmtab` and the key is securely stored in a file in the `/etc/cryptmount` directory. You’ll also be asked for a passphrase to secure the key and the user will need to enter this when mounting their volume. They do that with `cryptmount`:

```
$ cryptmount myvolume
```

Unmounting is similar:

```
$ cryptmount --unmount myvolume
```

`Cryptmount` might be more appropriate for some applications. Unprivileged users could, for example, have encrypted volumes on USB sticks and be able to use them without help from the systems administrator.

### Use TrueCrypt volumes with cryptsetup

`Cryptsetup` has been able to open `TrueCrypt` volumes since version 1.6. You just need to specify the volume type:

```
$ cryptsetup open --type tcrypt /path/to/myvolume myvolume
```

```
$ mount /dev/mapper/myvolume /mnt
```

If you want to mount a hidden volume, add the `--tcrypt-hidden` command-line argument and use `--key-file` if you need to specify key files.

you’ll have an encrypted filesystem. A file written to `mysecrets` will be transparently encrypted and stored in `.mysecrets_encrypted`:

```
$ echo "This is my secret" > mysecrets/test
```

```
$ ls -l mysecrets
```

```
-rw-rw-r-- 1 myuser users 18 Oct 20 14:08 test
```

```
$ ls -la .mysecrets_encrypted
```

```
-rw-rw-r-- 1 myuser users 1092 Oct 20 13:58 .encfs6.xml
```

```
-rw-rw-r-- 1 myuser users 34 Oct 20 14:08
```

```
5gk8Df5Gk3eN0sJx1fiqPppA
```

Notice the encrypted file is larger and has an indecipherable name. There’s also a hidden file called `.encfs6.xml` containing the metadata required. You use the FUSE mount command to unmount your encrypted filesystem:

```
$ fusermount -u ~/.mysecrets
```

Mounting is performed using the same `encfs` command that was used above to create the filesystem – it only creates a configuration if it doesn’t already exist. A useful thing to know is that the encrypted and plaintext directories can be on different filesystems. One useful application for this is encrypting files in cloud-based storage like DropBox: you can do something like this:

```
$ encfs ~/Dropbox/Private ~/.mysecrets/Private
```

The other part to EncFS is `encfstcl`, an administrative tool that can display information about an encrypted filesystem but is mostly useful to change its password:

```
$ encfstcl passwd ~/.mysecrets_encrypted
```

You can also use `encfstcl` to access an encrypted directory without mounting it.


```
$ encfstcl ls ~/.mysecrets_encrypted
```

We’ve covered the two main ways to perform transparent encryption but neither suit if you just want to secure a single file. You can do this quickly with nothing more than OpenSSL – you can encrypt a file like this:

```
$ openssl aes-256-cbc < plaintext > encrypted
```

and decrypt it

```
$ openssl aes-256-cbc -d < encrypted > plaintext
```

The `aes-256-cbc` cipher gives good protection but, while this method achieves its objective, it’s more practical to use a public key infrastructure to share encrypted files. OpenPGP is an example of this that we’ll explore next month. 

#### PRO TIP

Ubuntu users can do `cryptdisks_start myvolume` to open a volume in `/etc/crypttab`. `cryptdisks_stop` closes it. `systemd` has `cryptsetup.target`.

#### PRO TIP

`cat /proc/crypto` lists the kernel’s available ciphers and supported key sizes.

#### PRO TIP

EncFS is used by [boxcryptor.com/classic](http://boxcryptor.com/classic), which may be handy if you need to access protected directories from other platforms.

John Lane provides technical solutions to business problems. He has yet to find something Linux can’t solve.

# /DEV/RANDOM/

## Final thoughts, musings and reflections



**Nick Veitch** was the original editor of *Linux Format*, a role he played until he got bored and went to work at Canonical instead. Splitter!

So, at this time of year, there is one burning question on everyone's mind. I don't mean whether Santa will be able to deliver a 3D printer on time, but... will 2015 finally be the year of Linux on the Desktop?

I'm sorry, it's just my seasonal joke. 2015 will come and go and still analysts will tell us that Linux is irrelevant, nobody uses it and even Mac OS X is more worthy of attention. Maybe, in some ways, they are right. I know I have long since tired of trying to convert friends and relatives to the way of Linux.

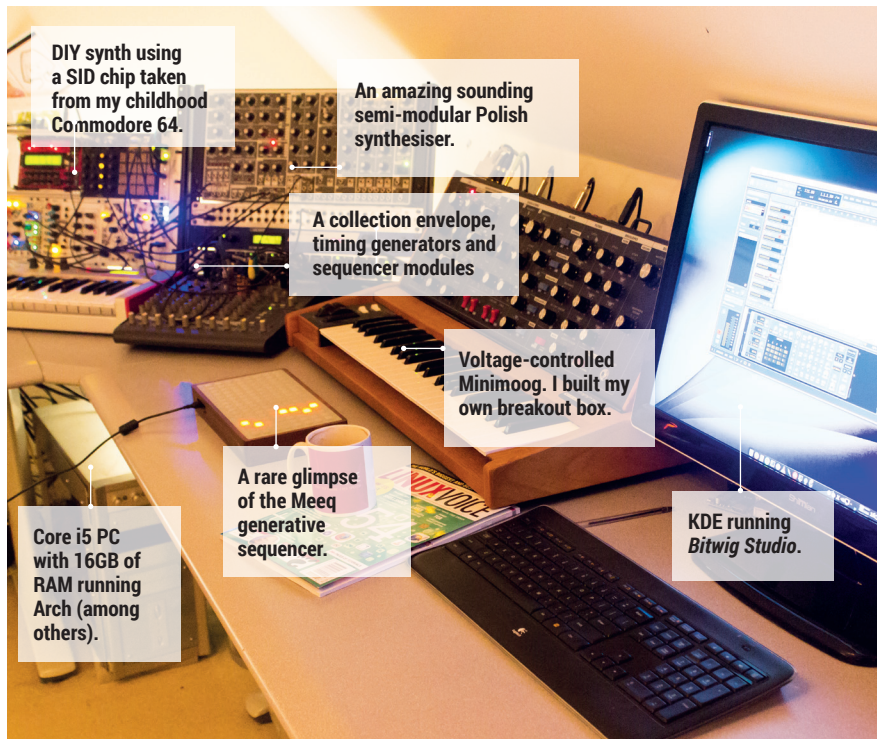
If anything, running Linux is a great get-out-of-jail-free card for me, because at family gatherings I can truthfully deny being able to help them with their IT problems, since I have never run Windows 8 or seen a version of OS X since they stopped naming them after cats.

The amusing thing is, that aside from the desktop, Linux is not only flourishing, but open source operating systems dominate.

Servers have long been the mainstay of Linux usage, but in emerging areas of computing, like clouds, Linux all but eclipses everything else. You may argue whether Android really counts as Linux (I would say not), but in the mobile and tablet space it also eats the competition.

But those are boring examples. A lot of the systems run by the European Space Agency are based on their own version of Linux – it forms part of the standard software deployment for remote experimentation. Even better, a lot of the Antarctic ice shelf was just measured by this ([www.who.edu/page.do?pid=21140](http://www.who.edu/page.do?pid=21140)) – an autonomous underwater vehicle that runs on Ubuntu. Because why not?

Let the "others" keep the desktop – everything exciting runs on Linux. And I won't have to answer so many support questions at dinner.



## My Linux Setup **Graham Morrison**

Editor of *Linux Voice* and creator of weird music noises.

**Q** What version of Linux are you using at the moment?

**A** I've been using Arch for a couple of years. But that's not really a badge of honour. I'm constantly breaking it and getting told off. But I do love the user repository and the package management.

**Q** What was the first Linux setup you ever used?

**A** I tried getting Red Hat 5.1 running on a Commodore Amiga and gave up when I couldn't get X to work. After that, it was Mandrake 6.0 on a PC a couple of years later.

**Q** What Free Software/open source can't you live without?

**A** There's just so much; Vim/X/KDE/Audacity/Bash/kernel/Kodi/Ardour/

Gimp. The list is endless. But to be a little more adventurous, I launch everything from KDE's *Kickoff*.

**Q** What do other people love but you can get on without?

**A** I really don't like most music players – such as *Amarok*. I listen to quite a lot of music and I just want quick access to audio and simplicity. I don't want Wikipedia entries, lyrics and VU meters.

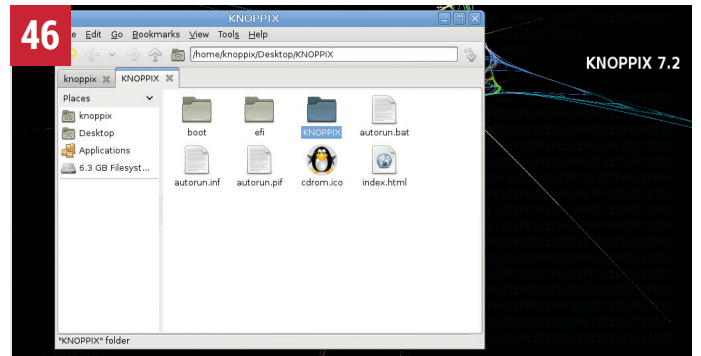
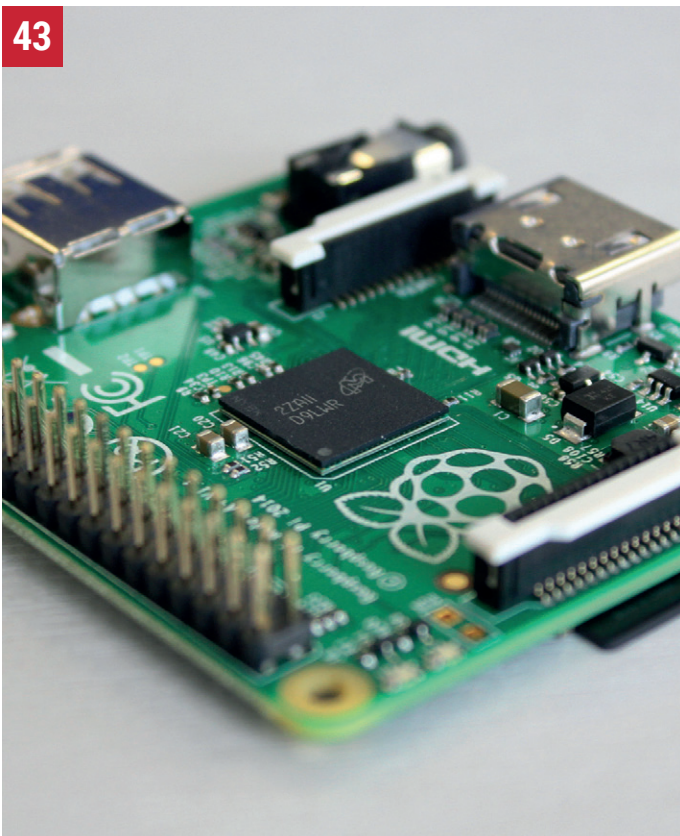
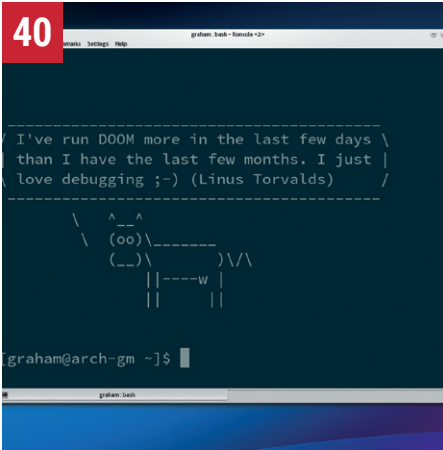
**Q** Is there one single piece of proprietary software you wish were open source?

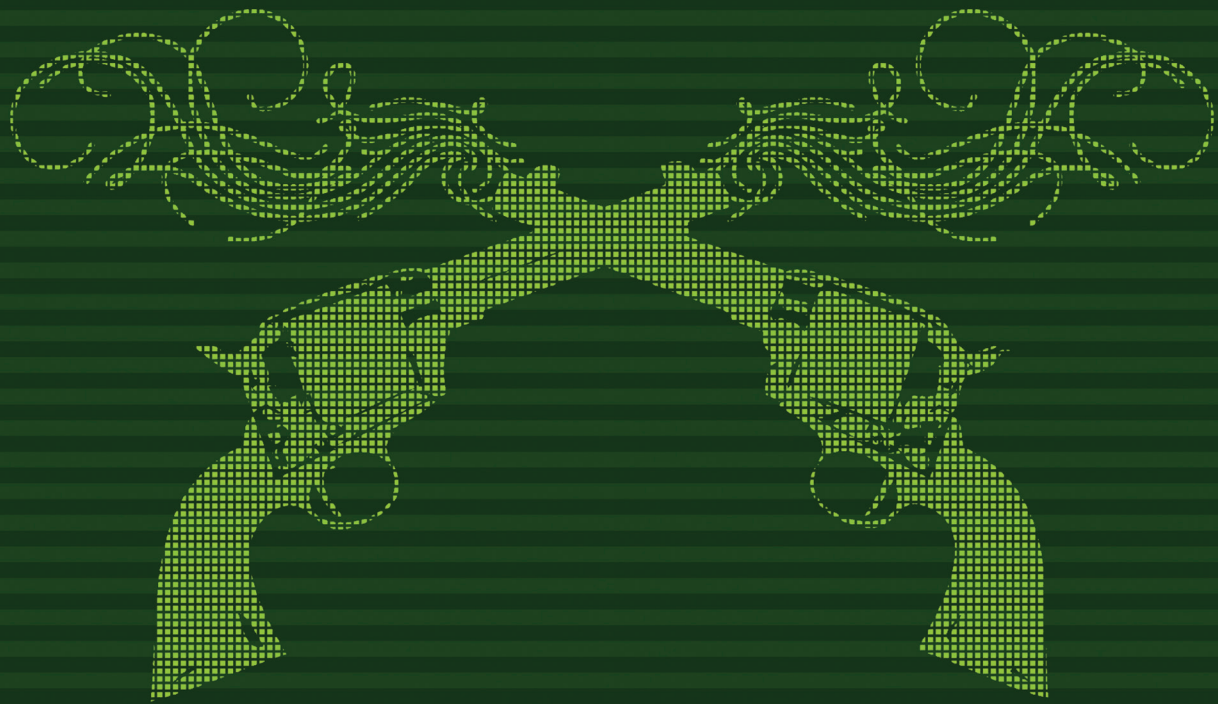
**A** Adobe's *InDesign*. That way I'd be completely free of any other operating system but Linux. ☑

# LINUXVOICE

## Pub Quiz Image Round

(See page 28)





"Thanks to the  
Linux Outlaws - it's  
been a wild ride."

EVERYONE AT BYTEMARK

**:BYTEMARK**

Proud to sponsor Linux Outlaws - right to the very end.