50% OF OUR PROFITS GO BACK TO FREE SOFTWARE

FREE DVD!

# LINUXVOICE

115 PAGES OF LINUX LEARNING

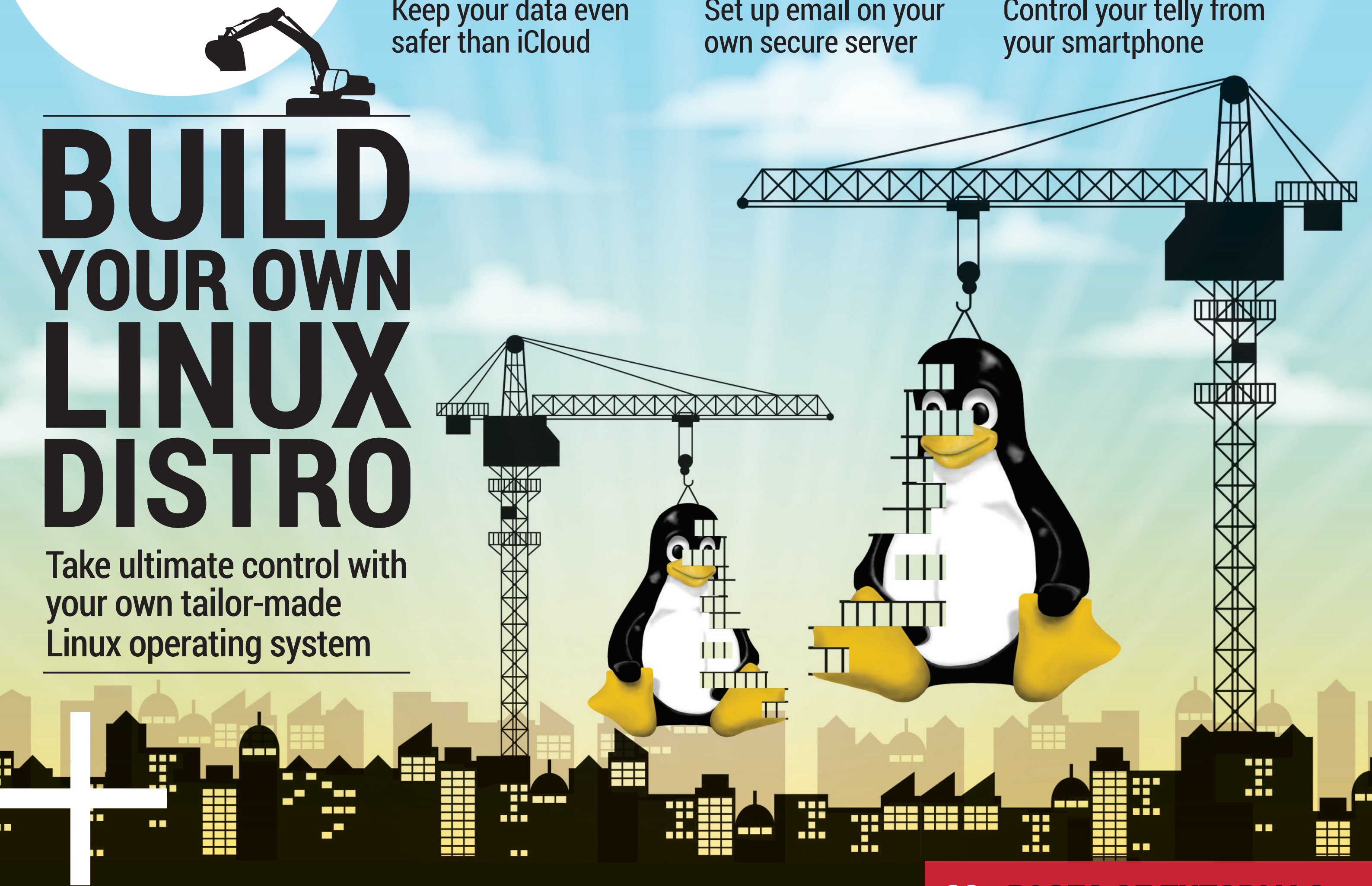
November 2014

FREE SOFTWARE | FREE SPEECH

PASSWORD CRACKING
## SECURITY
Keep your data even safer than iCloud

CYRUS
## EMAIL
Set up email on your own secure server

ANDROID
## TELEVISION
Control your telly from your smartphone



## BUILD YOUR OWN LINUX DISTRO
Take ultimate control with your own tailor-made Linux operating system

32+ PAGES OF TUTORIALS

**RASPBERRY PI** Operating system group test
**ANIMATION** Create a stop-motion movie masterpiece
**DIASPORA** The social network that's not trying to sell your soul

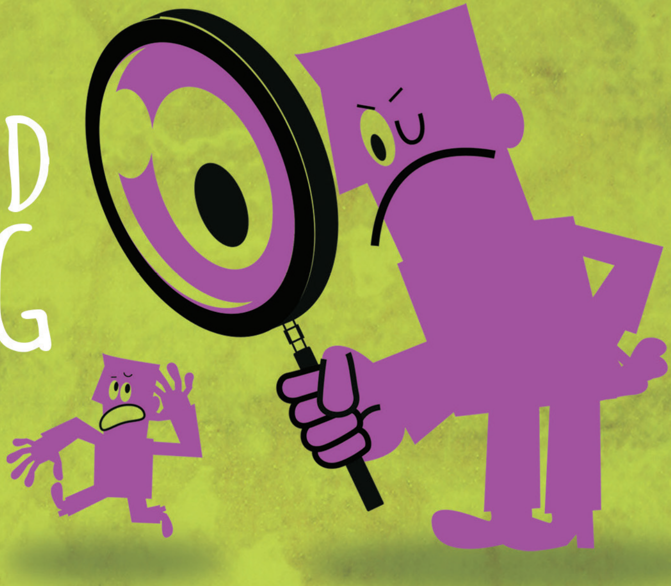
CAPITALISM
## INDIE TECH
The rise of the (open) machines

HISTORY
## LINUX GAMES
How gaming on Linux got to where it is today

ISSN 2054-3778
November 2014 £5.99 Printed in the UK

Andrews & Arnold Ltd

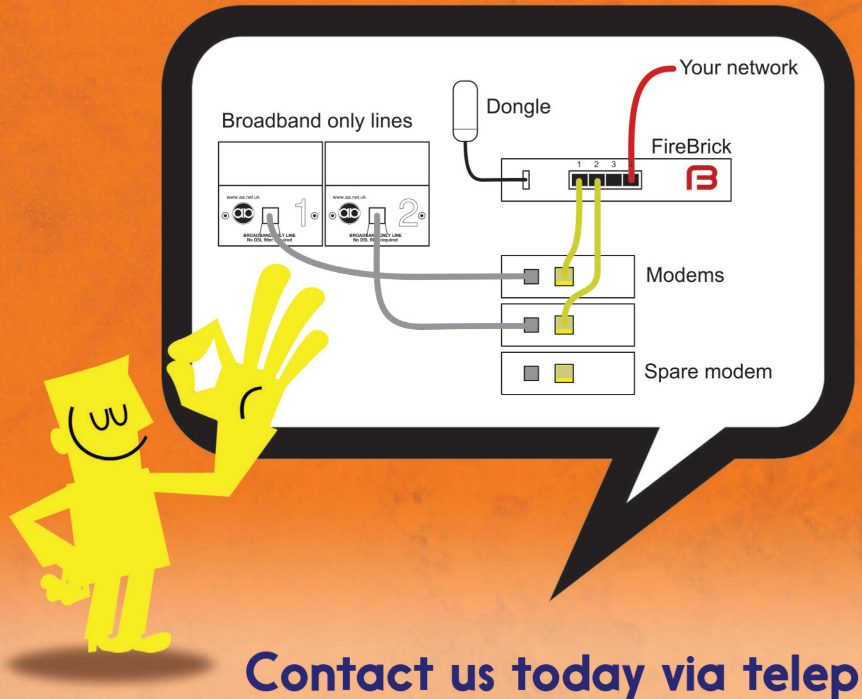
## HOME::1 BROADBAND F*CK FILTERING



Home::1 from AAISP is broadband that comes without unpleasant 'extras' like filtering, carrier grade NAT and offshore call centres. The service comes with one static real IPv4 address and a block of real IPv6 address space. In short, Internet the way it was meant to be.

FROM £25 A MONTH ★ NO CENSORSHIP ★ REAL IPv4 & IPv6
GREAT UK BASED SUPPORT BY LINUX USERS ★ CLEAR DRIPA POLICY

## OFFICE::1 BROADBAND BUSINESS GRADE CONNECTIVITY



Two dedicated broadband lines; two modems plus a spare, a FireBrick bonding appliance/router and a 3G dongle for failover are the ingredients that make up Office::1 Broadband from Andrews & Arnold.

- **Highly redundant connectivity**
- **No censorship**
- **Real IPv4 and IPv6**
- **Pro-active line monitoring**
- **Staff who use Linux daily**

**Contact us today via telephone, email, SMS or IRC!**

www.aa.net.uk 03333 400 220 sales@aa.net.uk

# Join us now and share the software

## The November 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, Juliet Kemp, John Lane, Vincent Mealing, Travis Mooney, Simon Phipps, Les Pounder, Mayank Sharma, Valentine Sinitsyn, Richard Smedley.

**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.

Many technical people have always argued that 'The Cloud' is no different to lots and lots of servers connected to lots of storage. This is true, but what's becoming more evident is that it's not the physical infrastructure that defines what the cloud is, but its seamless ubiquity. And that's where the real danger lies. How many iPhone and Android users, for example, really understand what it means when their images and videos are 'backed up' to the cloud and what the implications may be for their security, or how their rights may be affected by where that data is stored?

Education is obviously crucial. But we also need an alternative to show that the ubiquity and convenience of cloud services don't need to go hand-in-hand with a loss of privacy. The only possible source for such an alternative that I can see is Linux and Free Software, and there are projects doing exactly that (we look at two this issue; Diaspora on p40 and Indie/phone on p28). Richard Stallman's famous song may have been written in a pre-cloud 1993, but it's just as true today: "Join us now and share the software; You'll be free, hackers, you'll be free."

**Graham Morrison**
Editor, Linux Voice

SUBSCRIBE ON PAGE 62

## What's hot in LV#008

**ANDREW GREGORY**

Just like Bruce Willis in *Armageddon*, Linux is being launched into space to help save the entire human race **p32**

**BEN EVERARD**

Become the next Nick Park with nothing more than some LEGO and a Raspberry Pi with our animation studio **p78**

**MIKE SAUNDERS**

Learn how easy passwords are to crack, and how to best protect yours, with Ben's fantastic guide to breaking their encryption **p86**

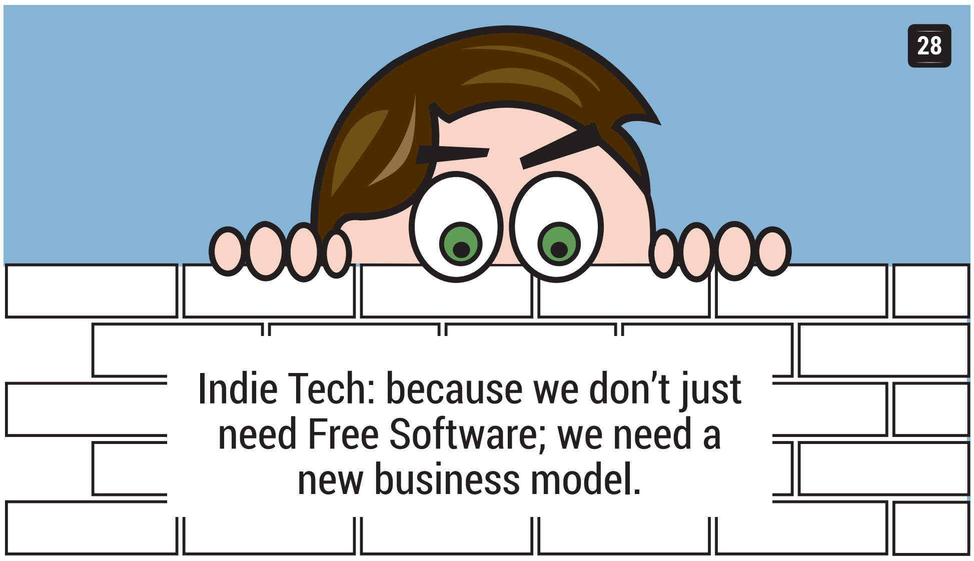
# CONTENTS

Novemberr LV008

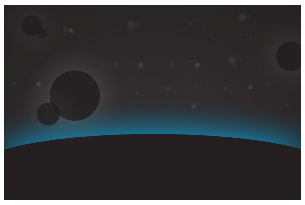
Season of mists and mellow apt-get dist-updates.

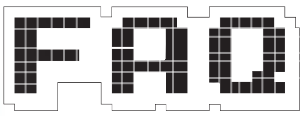
SUBSCRIBE ON PAGE 62

Build a Linux distro and share it with the world.

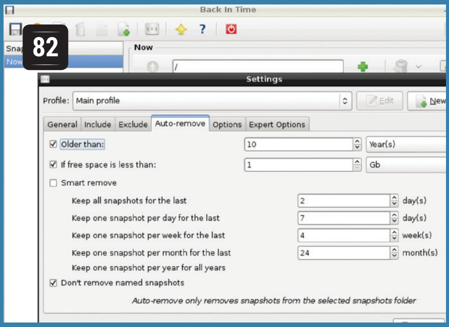


Indie Tech: because we don't just need Free Software; we need a new business model.



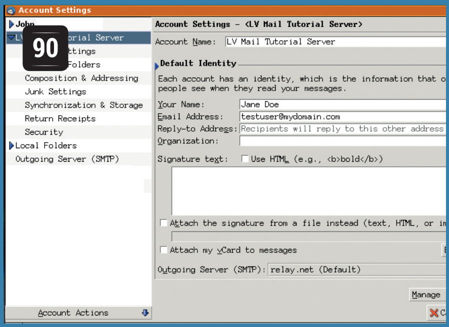


## TUTORIALS

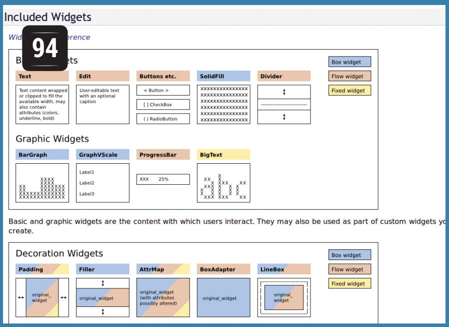


Back In Time
Now
Settings
Profile: Main profile
Edit
New
General | Include | Exclude | Auto-remove | Options | Expert Options
☑ Older than: 10 Year(s)
☑ If free space is less than: 1 Gb
☐ Smart remove
Keep all snapshots for the last 2 day(s)
Keep one snapshot per day for the last 7 day(s)
Keep one snapshot per week for the last 4 week(s)
Keep one snapshot per month for the last 24 month(s)
Keep one snapshot per year for all years
☑ Don't remove named snapshots
*Auto-remove only removes snapshots from the selected snapshots folder*

CloudCracker
An online password cracking service for penetration testers and network auditors who need to check the security of WPA protected wireless networks, crack password hashes, or break document encryption.
Start Cracking
File Type: SHA-512 (Unix)
Hash File: Browse... No file selected.
Next
Big. Fast. Cheap. Run your network handshake against 300,000,000 words in 20 minutes for $17.
Save Money. Save Time.
Comprehensive Dictionaries.

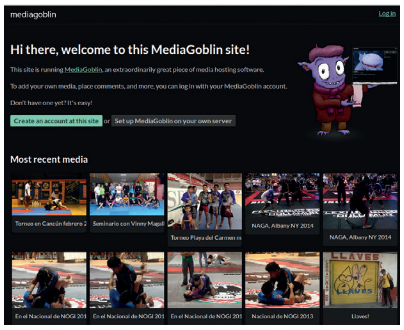


Account Settings
Account Settings - <LV Mail Tutorial Server>
Account Name: LV Mail Tutorial Server
**Default Identity**
Each account has an identity, which is the information that o
people see when they read your messages.
Your Name: Jane Doe
Email Address: testuser@mydomain.com
Reply-to Address: Recipients will reply to this other address
Organization:
Signature text: ☐ Use HTML (e.g., <b>bold</b>)
☐ Attach the signature from a file instead (text, HTML, or i
☐ Attach my vCard to messages
Outgoing Server (SMTP): relay.net (Default)
Manage
Composition & Addressing
Junk Settings
Synchronization & Storage
Return Receipts
Security
Local Folders
Outgoing Server (SMTP)
Account Actions



Included Widgets
Graphic Widgets
Basic and graphic widgets are the content with which users interact. They may also be used as part of custom widgets y
create.
Decoration Widgets

## REVIEWS



mediagoblin
Hi there, welcome to this MediaGoblin site!
This site is running MediaGoblin, an extraordinarily great piece of media hosting software.
To add your own media, place comments, and more, you can log in with your MediaGoblin account.
Don't have one yet? It's easy!
Create an account at this site or Set up MediaGoblin on your own server
Most recent media

# NEWSANALYSIS

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

Opinion

## The erosion of software patents

Reform is coming in the form of evolution, not revolution. Praise be for that…

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

I've long been a critic of patents on software, even if the boundary between them and other patents is hazy. Patents may work in other industries, where the cost of innovation is so high that a temporary, state-sanctioned monopoly provides just enough time to gain a return on the investment. That's the big justification of patents in pharmaceuticals, for example. But that investment–return ratio has a completely different value for software. It turns out that software patents have little bearing on encouraging innovation.

No programmer I've ever met refers to software patents, for two reasons. First, they aren't written for programmers to learn from -- they're written for patent lawyers to sue against. You'll find software patent filings that contain no sample code and few technically-oriented descriptions. When I worked at IBM, I asked a patent lawyer at the company what was needed to file a patent. I was told "a rough idea – we can fill in the details for you – and then all the ways you can think of how we could tell if someone else was using the idea."

The second reason programmers never refer to software patents is that they're told not to do so. US law permits much greater penalties if patent infringement is found to be wilful. Many people regard looking at patents as good a proof of wilfulness as you can get. Every company I've ever visited has told its programmers to stay well clear of reading patents.

### Safety in numbers

While corporations can usually find a way to defend themselves – in the extreme via patent licensing – open source communities would probably not be able to do so. There's often no legal entity to protect open source programmers. When there is an entity, it's likely to be a non-profit with few resources. Anything that stands in the way of software patents is good news for open source.

Given that the market for technology is global, like the internet, what happens in the USA is very significant in setting trends for all of us. So it's good to pay attention to US legal decisions, even if we live in Europe where we think the situation is different.

At last there's some good news. Recently, the US Supreme Court made a landmark decision when it declared software patents belonging to Alice Corporation to be invalid – the Alice Corporation vs CLS Bank decision. It looks like that decision is already making a difference in reversing the tide of software patents. It has now showed up several times in the US Court of Appeals for the Federal Circuit (CAFC), most notably in a major software patent troll case but also in individual cases. That's the court that usually handles appeals of patent cases in the USA.

**"Software patents aren't written for programmers to learn from – they're written for patent lawyers."**

Previously this court struggled to understand what it took to invalidate a software patent, but in decisions delivered recently the Supreme Court's clarifications in Alice vs CLS showed up several times. In a significant case, they helpfully clarified the decision making process in a case involving prolific patent plaintiff Digitech Image Technologies. The case related to a core part of digital imaging – colour profiles.

It had appealed a finding by the District Court for the Central District of California that the patents it was using to attack a veritable Who's-Who of the digital imaging market were invalid. Legal scholar Mark Lemley led a team representing camera manufacturers including Mamiya, Leica, Pentax and Hasselblad, computer makers such as Toshiba and Asus and major US retailers B&H, Newegg and Buy.Com.

The most important use of the Alice vs CLS decision came when the CAFC decided against accepting "a device which…" as a way to make an abstract idea patentable:

### A change in interpretation

This is a good sign for the software industry. Previously, CAFC had a tendency to accept the validity of such patents, but it seems the SCOTUS finding could reverse that tendency and in time discourage use of software patents. Let's hope they can resist the temptation to act on their slightly curious interpretation of the SCOTUS explanation of what it would take for a software patent to remain valid.

This is not the major reform some of us have hoped for – which may yet appear – but the steady drip drip drip of the Alice vs CLS decision on the existing mountain of bad software patents looks like it will level the landscape much sooner than would otherwise have been the case.

Desktop Linux • Munich migration • Gnome Foundation • Firefox • XBMC = Kodi

# CATCHUP

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

## 1 Linus Torvalds: "I still want the desktop"

Although Linux's market share on the desktop has hovered around a few percent for many years, it's no reason to give up. The kernel head honcho has said he still wants Linux to conquer home machines, stressing that the problems come from infrastructure and packaging. It should be easier for application developers to build binaries that run across all distributions, instead of needing separate packages for every distro and release, Torvalds believes. And we agree with the man.

## 2 Don't panic: Munich isn't switching back to Windows. Yet...

Over the last decade, Munich City Council has moved 15,000 PCs to Linux. Recently the new mayor claimed that the transition was a mistake and should be reverted, leading so-called "news" sites on the web to say the whole thing was a disaster. Not true: the Council has dismissed the mayor's remarks as "irrelevant personal opinions", and while alternatives will be considered, there's currently no plan to move away from Linux.

## 3 Gnome Foundation publishes its Annual Report for 2013

You've got to hand it to Gnome, even if you disagreed with the design choices for Gnome 3. The Foundation does a great deal to bring developers together with hackfests and conferences. Read the full report (in PDF format) at **http://tinyurl.com/gnome2013**

GNOME™

## 4 Firefox to get sponsored tiles in upcoming release

The Mozilla Foundation has received a lot of flak for this, but funding for *Firefox* development doesn't grow on trees. Future versions of the browser will have sponsored tiles on the new tab page – that is, tiles from Mozilla partners that "may be of interest" to users. As you visit more and more sites, however, the sponsored tiles will gradually be replaced by your most visited pages, so this is generally something that will only affect brand new *Firefox* installations.

## 5 China to launch new OS in September, probably based on Linux

The government of the world's most populous country is still largely running Windows XP, and has banned upgrades to Windows 8. Now the Communist Party has started work on its own OS, likely based on Linux, to move the country away from dependence on Western companies. It will have its own app store, and eventually run on tablets and smartphones. We can't help but feel that the whole NSA spying antics may have played a part in this...

## 6 Freshmeat (aka Freecode) reborn as Freshcode

For many years, Freecode (formerly known as Freshmeat) was the number one source for tracking free software releases. Unfortunately it died a few months ago, and we were stuck without a replacement... until now. A new website at **www.freshcode.club** provides the spiritual successor to the old sites, with a very similar look and feel. Currently it's at version 0.7.0 and lacking some features, but plenty of developers are submitting their wares and it's getting busier with each day.

## 7 XBMC gets new name: say hello to "Kodi"

It's probably the most popular media centre software for Linux, but its name was getting a bit dated: the *Xbox Media Centre* barely runs on the original console, and doesn't have ports for the successor machines. It also does more than just playing media – it has games too. So the team behind it has decided to give it a shiny new name, *Kodi*, along with a new logo. *Kodi 14* is undergoing development as we speak, with alpha releases coming thick and fast.

## 8 Kernel git repository gets two-factor authentication

Previously, any developer committing code to the main Linux kernel tree used their SSH private key as a means of identification. This works OK until the key is stolen – so a new system has been put in place. All hackers with access to the main tree now have USB gizmos (YubiKeys), which provides an extra level of security. For us end users, it's another safeguard against crackers masquerading as real kernel developers and sneaking dodgy code into the source tree.

# DISTROHOPPER

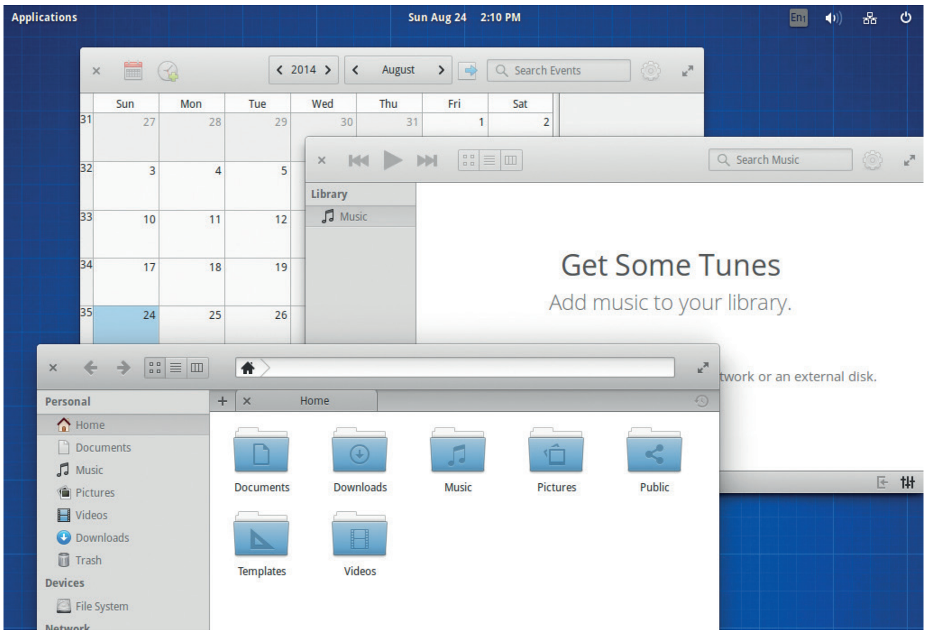
Our pick of the latest releases will slake your thirst for new distros.

## Elementary OS

### 2014's most anticipated distro.

Elementary is such a famous distro that it's hard to believe that we're only just testing out the beta of the third release (named Freya). For those of you who don't know, Elementary is built on top of Ubuntu with the addition of the Pantheon desktop environment, which is known for its focus on styling and simplicity.

Freya comes with an unusual set of applications. For example, *Midori* is the standard web browser and *Geary* fulfils email duties. Some of the software is written from scratch to fit in with the Elementary look. For example, it has its own music player, calendar, text editor and terminal. Most of these use *GTK 3* top bars that let you pack in icons and widgets where most desktop environments place the application's menus. This works well for providing easy access to the key functions, but can leave you wondering where to find the advanced features of the software. A little too often, the answer is that the software doesn't have any advanced features. The default software all has a very consistent look and feel. Of course, there is loads more software in the repositories, but the further you venture from the standard apps, the more you're likely to lose this consistent feel.



Elementary gets top marks for style, but power users may be better served elsewhere.

## Tanglu

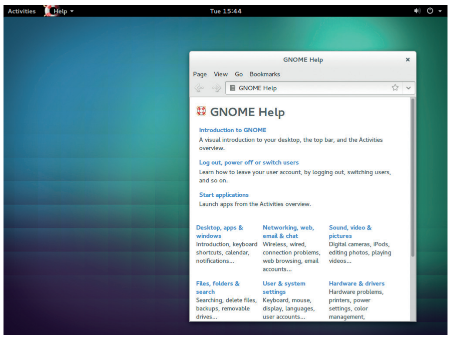
### Debian for desktops.

Tanglu is a project designed to polish up Debian to make it a little easier for end users. This doesn't mean adding a few packages and making the desktop environment a little prettier; it means locking the distro into a predictable release cycle, and making sure that the latest software is always available.

This isn't the first time an organisation has tried to provide a tamed Debian for desktop users – it's exactly how Ubuntu got started. However, unlike Canonical's distro, Tanglu is committed to working with Debian and upstream sources rather than pushing home-grown software and its own agenda. Most common desktop environments are available, but downloads come in flavours for Gnome and KDE. Both of which are in their vanilla states without any customisation. Outside of the desktop environments, you shouldn't expect any surprises. The first Alpha version of Tanglu 2 comes with *Libre Office 4.3* as a productivity suite and Firefox 30 as a web browser (as well as the native tools for the desktop environment).

It's a new distro (the first version came out in February 2014, and version two is due in October 2014), so it's too early to say if this approach will gain it the popularity of Ubuntu. Tanglu does have a slightly weaker policy on non-free software than Debain, so more firmware will be included on the install DVD. This is another sign of Tanglu's focus on home users rather than servers.

Overall, there's a lot to like about Tanglu, but we'd be tempted to wait a little while and see how well it's supported before switching any important machines over.



Tanglu: a Debian-based system with a fixed release cycle and unadulterated components.

## Qubes

### The ultimate secure distro.

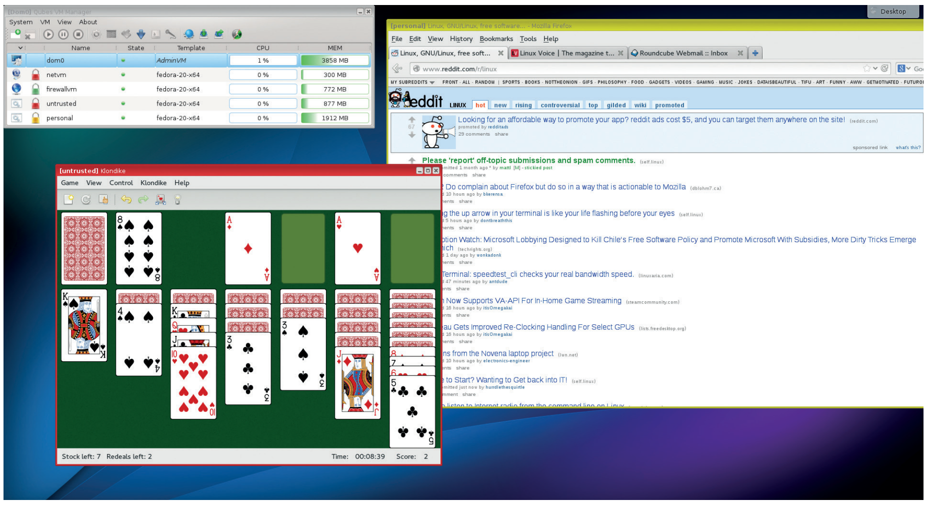
Qubes works on the principal of security by isolation. It's based on the Xen hypervisor with a series of virtual machines running on top of it. One runs the desktop environment, whereas others are AppVMs that run the applications. By default, there are AppVMs for work, banking, personal use and untrusted use, though this setup could be adjusted for other uses. The principal is that if an attacker compromises any individual VM, they still can't access applications running in the others. So, if you accidentally install some malware in the untrusted VM, it can't penetrate the banking VM.

Despite applications running on different virtual machines, they all appear on the same desktop, and the colour of the window border lets you know which AppVM it's running in. In version 2, you can now run Windows AppVMs inside Qubes in the same way you run Linux VMs.

You may have read all this and thought that the security offered by Qubes is just the same as running various virtual machines in *VirtualBox* or *Qemu* in a regular desktop Linux. This isn't the case. Qubes is, in theory at least, more secure because of its architecture. It runs the Xen Hypervisor on bare metal, then on top of this it runs various virtual machines. Some of these virtual machines are there to handle hardware, networking, etc. Others are there to run applications. However, they all run on top of the Xen hypervisor. This means that in order to break out of one virtual machine and get into another, an attacker has to break through the Xen hypervisor.

Using a more common desktop visualisation method, one Linux kernel is running on the bare metal, and then other Linux (or other OS) kernels run on top of this. To break from one virtual machine to the next, an attacker has to break through the Linux kernel. In Linux containers, all applications are running in sandboxes on top of a single Linux kernel, so again, and attacker has to break through the kernel.

The Linux kernel is quite secure. However, it's also massive. It's somewhere around a hundred times as many lines of code as the Xen hypervisor. That means that in order to have the same number of bugs overall, the Linux kernel would have to have 100 times fewer bugs per line as Xen.



Qubes VM Manager
[untrusted] Klondike
[personal] Linux, GNU/Linux, free software... - Mozilla Firefox

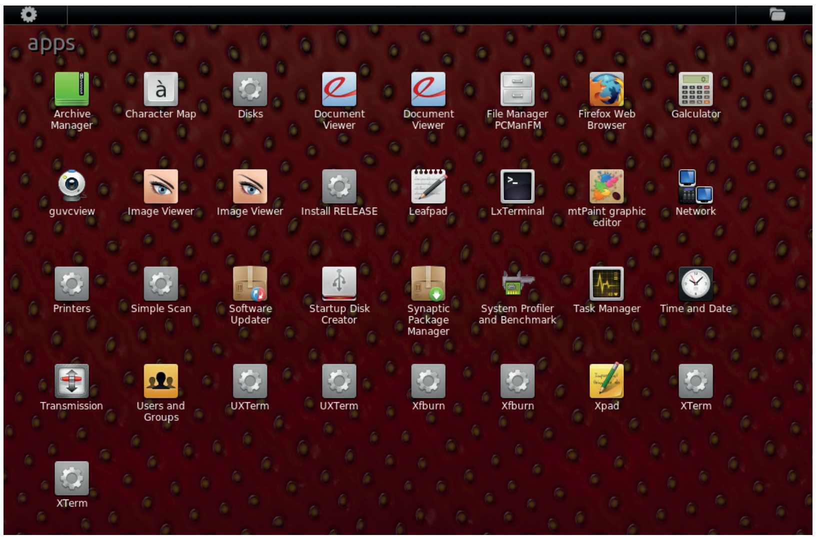
In this screenshot the web browser is running in the Personal AppVM (in the window with the yellow border), while Solitaire is running in Untrusted (red).

### Symphony A new approach to user-friendliness

Symphony is built around the Mezzo desktop environment, which is designed to simplify the graphical user experience. This simplification is built around the principal that hierarchical menus are confusing, but users find it easy to put the mouse in the corners of the screen. As a result, there are buttons in each corner of the desktop (clockwise from top-left: Settings, Places, Logout, Applications). Clicking on any of these brings up a screen that's a bit like a simplified version of Gnome's Dash.

There are also restrictions on how you can move windows, supposedly to stop users moving them in such a way that important information disappears off screen. The end result of this is a desktop environment that feels like a cross between Gnome Shell and Android.

It's always good to see experiments that hope to make computers more user-friendly, and Mezzo has some interesting ideas. However, at this stage, it seems like it's only ready as a proof-of-concept for people interested in user-interface design. It's still quite rough around the edges, and there isn't any specialised software; instead, it uses mostly *GTK* programs from LXDE and Gnome, so the applications follow a completely different design philosophy.



The Apps menu (from the bottom-left button) brings up a full screen selection menu.

# GAMING ON LINUX

The tastiest brain candy to relax those tired neurons

## MISTY AND MELLOW

**Liam Dawe is our Games Editor and the founder of gamingonlinux.com, the home of Tux gaming on the web.**

Something of a hot topic recently in the world of Linux is OpenGL. The reason behind this is that seemingly out of nowhere a number of developers have started doing big blog posts on the poor state of the OpenGL graphics API.

For those who don't know, OpenGL is an API that enables developers to hook into your graphics chips, and it's comparable to DirectX from Microsoft platforms. OpenGL is of course an open system where a consortium of people and companies have come together for a common goal.

A lot of complaints about OpenGL are based on the way it performs across different graphics chips from Nvidia, AMD and Intel, and quite rightly so as they differ massively from one vendor to the next.

There is also the fact that OpenGL support across different platforms is patchy, with Linux, Mac and Windows all supporting different versions, and more so in the case of Windows as it doesn't come with it as standard.

AMD came out with its own API, named Mantle, aimed at increased performance, but the problem with Mantle is that currently it's pretty well closed off, and AMD still hasn't given a clear indication of whether it will come to Linux or not.

The question we pose to you is this: Do we need a brand new graphics API to compete with DirectX and alleviate developers' woes surrounding OpenGL? A new API with a new name could offer a fresh start.

Let us know what you think: **http://forums.linuxvoice.com**

## Borderlands: The Pre-sequel

**Get ready to shoot 'n' Loot!**

Do not adjust your reading glasses – you did read that correctly. A *Borderlands* game is really coming to Linux and promises to excite the many fans of the franchise as it lights up the first person shooter genre on Linux.

Previously the CEO of Gearbox Software (the developer) told Linux gamers to not get their hopes up about *Borderlands 2*, but a recent public financial document from the publisher and later confirmation thanks to IGN showed that *Borderlands: The Pre-sequel* has plans for a Linux version.

The *Borderlands* series is well known for the excellent and frantic action that mixes first person shooting with random loot generation, and some fun graphics added into the mix make it something that serious Linux gamers are going to go nuts over.

There isn't currently any word on when it will be available to buy, but we do know that it will retail for around the £30 mark.
**http://store.steampowered.com/app/261640/**

One of the many random guns!

## Cities Skylines

**We don't need no Sim City!**

*Cities Skylines* has been announced by publisher Paradox Interactive and developer Colossal Order, promising an excellent city builder experience for Linux gamers. Since we don't have a game like *Sim City* this should help fill a rather big gap left wide open for Linux gamers. It can be played offline as opposed to the horribly DRM-crippled *Sim City* fiasco

Cities will be full of the usual features you would expect like building roads and different zones for buildings. One of the best features of *Cities* is that it will have support modding it, so you can expect many weird and wonderful buildings to download from the community. *Cities* will also have water flow simulation to bring some more strategy in for water-based services. There's no word yet on pricing or a release date, but we will be sure to update you on its progress closer to the release.
**www.paradoxplaza.com/cities-skylines**

That's an actual bridge you can build...

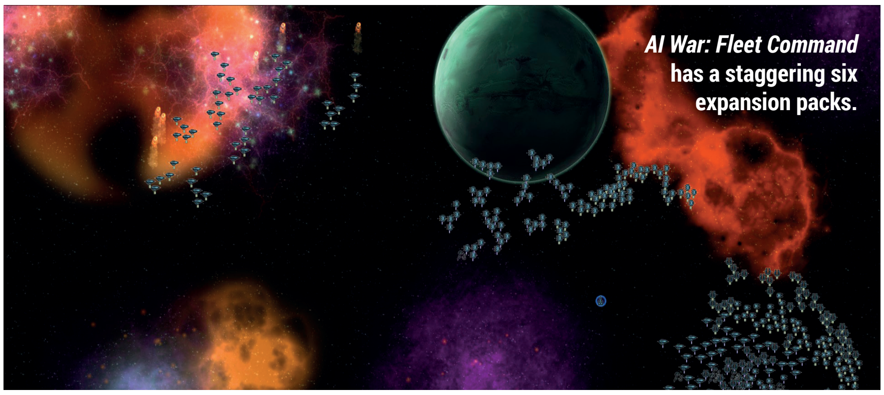
## AI War: Fleet Command

**Breaking the RTS mould wide open.**

What's that? Another from Arcen Games? You read that right folks: *AI War: Fleet Command* is another excellent game from the back catalogue of developer Arcen Games, and it brings some real time strategy to the table this time around.

*AI War: Fleet Command* is a very different kind of real time strategy and it seems to be inspired by some of the classics of the genre with certain features. The choices in *AI War* really matter, as you aren't just fighting in one battleground, but across many. You can, for example, bring a couple of hundred spaceships from one system to another, so your early game can matter just as much as the late game.

It has become quite the favourite here due to its massive amount of gameplay. You can grab it from Steam for £6.99. **http://store.steampowered.com/app/40400**


*AI War: Fleet Command* has a staggering six expansion packs.

## Sanctum 2

**A blend of FPS & Tower Defence.**

Ryan "Icculus" Gordon, formerly of Loki Software, sure is a busy man – and his most recent work is a new Linux port of *Sanctum 2*! *Sanctum 2* is a hybrid of the tower defence game genre combined with first person shooting, and it's really rather good. It's the sequel to the world's first tower defence/FPS hybrid game which sadly we don't have on Linux, but a sequel is the next best thing.

You don't have to go it alone either, as the game offers up to four players to play together in co-op mode. There's even an in-game visual novel to keep you busy too.

You can grab it for £10.99 from Steam right now and join the fun. **http://store.steampowered.com/app/210770/**

## Darksiders 2

**From War to Death**

Hot on the heels of last issue's announcement that *Darksiders* will come to Linux, *Darksiders 2* is also planned for Linux to complete the series!

We haven't even seen the release of the first instalment of the series yet, but the developer has enough confidence in Linux to announce that the second is on its way.

This crazy hack and slash RPG will have you embark on a quest to restore mankind even though your name is Death. An odd name for a hero don't you think? As if being one of the legendary Four Horsemen and having a brother named War wasn't odd enough. It usually retails for around £24.99 on Steam. **http://store.steampowered.com/app/50650**

## ALSO RELEASED...

### Football Manager 2015

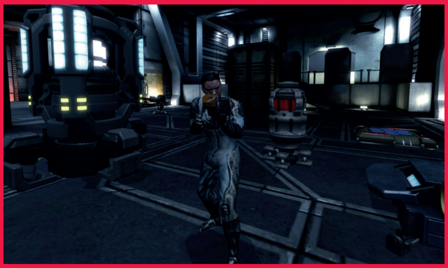
**Big news sports fans! *Football Manager* is the highly popular simulation game that first arrived on Linux last year, and now we will be graced with another!**

**Probably not one to pick if you don't like football, though it must have quite a few fans on Linux for the developer and publisher to bring *FM15* for us too. www.footballmanager.com**

### Dungeons 2

**Do you feel like doing a bit of digging? Do you dream of being an evil overlord? If those boxes are ticked then you'll love *Dungeons 2*! Kalypso Media is starting to push out more Linux game announcements and *Dungeons 2* is among them!**

**It promises a very *Dungeon Keeper*-like experience for Linux gamers, and that's not a bad thing. www.dungeons-game.com/en/index.php**



### Unvanquished

***Unvanquished* continues to push for a beautiful and fun open source first person shooter with the latest alpha release. The new release has new building models, optimisations, weapon inertia to be more realistic and much more!**

**It is not without bugs as it's still early days, so you have been warned, and it's free under the GPL v3 licence. www.unvanquished.net**

# LINUX VOICE YOUR LETTERS

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

## LINUX VOICE STAR LETTER

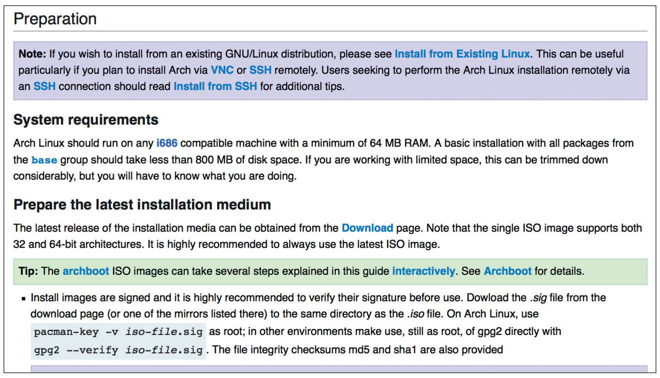
### WE'RE DOING SOMETHING RIGHT

I have been an Ubuntu user since 2009, but earlier this year when I heard that UbuntuOne was dropped I decided to change distros as nothing was keeping me loyal to Ubuntu anymore. What distro? Well in Issue One of Linux Voice I found your Arch tutorial; it took a full weekend of tinkering, but I have not looked back since. I wanted to try the mystical world of UEFI rather than BIOS and so your tutorial in Issue Two was required reading.

Ubuntu had been using Gnome when I started using it, but it subsequently switched to Unity, which I grew to know and love. I thought this would be the hardest thing to part with, but again Issue Two's KDE article came to the rescue. KDE is very user friendly and I enjoy it, but I have sited the system tray vertically on the left-hand edge.

I am continually dipping in to Linux Voice and I am glad I backed it last winter for six months. I have found the articles on subjects such as Vim and sockets both useful and informative. My six months is now complete, but I have resubscribed for a further year (make sure you put your subscription number on the order!). Keep up the great work and I look forward to more outstanding articles.

**Dom Walden**



Preparation

**Note:** If you wish to install from an existing GNU/Linux distribution, please see Install from Existing Linux. This can be useful particularly if you plan to install Arch via VNC or SSH remotely. Users seeking to perform the Arch Linux installation remotely via an SSH connection should read Install from SSH for additional tips.

**System requirements**

Arch Linux should run on any i686 compatible machine with a minimum of 64 MB RAM. A basic installation with all packages from the base group should take less than 800 MB of disk space. If you are working with limited space, this can be trimmed down considerably, but you will have to know what you are doing.

**Prepare the latest installation medium**

The latest release of the installation media can be obtained from the Download page. Note that the single ISO image supports both 32 and 64-bit architectures. It is highly recommended to always use the latest ISO image.

**Tip:** The archboot ISO images can take several steps explained in this guide interactively. See Archboot for details.

- Install images are signed and it is highly recommended to verify their signature before use. Dowload the *.sig* file from the download page (or one of the mirrors listed there) to the same directory as the *.iso* file. On Arch Linux, use `pacman-key -v iso-file.sig` as root; in other environments make use, still as root, of gpg2 directly with `gpg2 --verify iso-file.sig` . The file integrity checksums md5 and sha1 are also provided

Arch Linux is one of our favourites for its blend of speed and features.

**Andrew says:** That's music to our ears Dom, thanks for writing. You weren't the only one taken by surprise when Ubuntu dropped its Ubuntu One cloud service – we didn't see it coming either, but Canonical has many fish to fry and many pies in which to dip fingers, so there's bound to be something else in the pipeline from them soon.

It sounds like you've made the right choice in Arch Linux. Once you're over the hump of installing it it's fast, it lends itself to being customised, and it forces you to learn more about Linux. Plus, it won the Best Linux Distro 2014 accolade in last issue's epic distro battle. Congratulations on being ahead of the curve!

### YOU'VE GOT MAIL

I love the magazine and what it stands for, and also quite timely as I read the article about SMART disks health checking, having literally just logged out of my raid5 NAS and seen the ominous 'You have mail' message.

I checked **/mail/var/root** and found a bunch of mails which made it apparent that mdadm had been mailing me for the last two months trying to tell me that I had a drive failure. Luckily I had a spare in place which seems to have silently taken its place, and I now have a replacement disk and a replacement spare on the way.

My question is : how can I get mdadm (and SMART and other tools) to mail me somewhere more useful than **/var/mail/root**?

Here's what system I'm using:

Ubuntu 12.04.5 LTS (GNU/Linux 3.2.0-67-generic x86_64); mdadm - v3.2.5 - 18th May 2012; Heirloom mailx version 12.5 6/20/10.

Thanks!

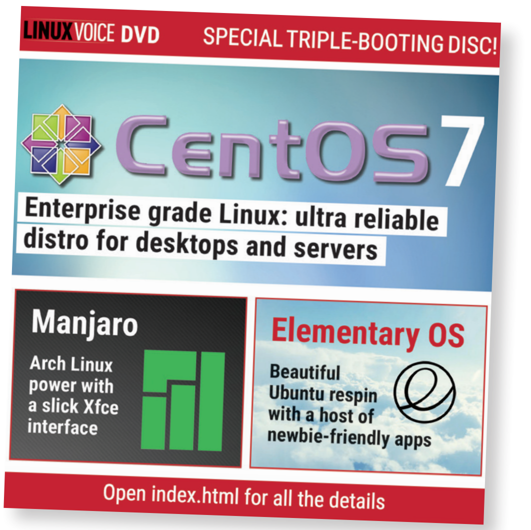
**John, London**

**Graham says:** This is a good question and one we hope to be able to answer more fully as we expand upon our mailserver tutorial series starting this issue (p90). However, after a little research it appears that the default email account for mdadm can be changed by editing or adding the MAILADDR field in /etc/mdadm.conf. Other services are likely to have a similar option in their configuration files, or you make want to look into filtering your root email automatically, depending on the sender, for example.

## THE DVD

Just a quick note to say that I appreciate your policy of only including a DVD when there's something worth having on it. Broadband is getting better, so even out here in the sticks I get decent connectivity. It's still not blazing fast, but unless you're downloading a 4GB ISO image every couple of hours, how fast do you really need your internet to be anyway? I've got plenty of books to read, a river to fish and countryside to explore, so Netflix doesn't appeal too much to me. A distro takes about an hour to download, which I find perfectly acceptable for a task I only perform about once or twice a year. And they do tend to get a little bashed on their way to the colonies, which lessens the appeal somewhat.
**David Kelly, Australia**

**Mike says:** The plan with the DVDs is to only have them when there's something really special. Readers – let us know if we've got the right idea. Do you get a lot of use from the DVD?


LINUXVOICE DVD
SPECIAL TRIPLE-BOOTING DISC!
CentOS 7
Enterprise grade Linux: ultra reliable distro for desktops and servers
Manjaro
Arch Linux power with a slick Xfce interface
Elementary OS
Beautiful Ubuntu respin with a host of newbie-friendly apps
Open index.html for all the details

Those unlucky enough not to have riparian pleasures can instead spend time playing with Manjaro Linux.

## SOMEBODY'S WATCHING ME

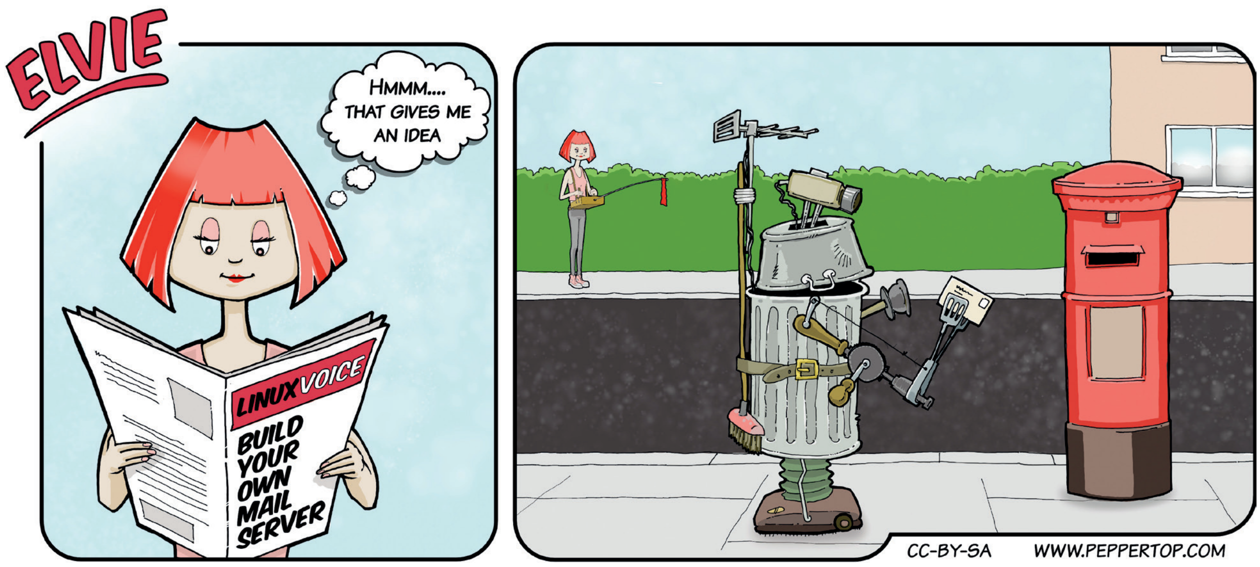
The iCloud leak got me thinking. For years I've had a vague mistrust of the cloud; giving your data for someone else to look after feels like asking for trouble. I don't think the internet would be interested in my private photographs, but if anyone did want to splash them over the web, they'd have to break into my house, find my external hard drive (good luck to them – it was in the cupboard under the stairs the last time I saw it) and physically take it away. At least with an open source cloud solution you know what security processes are in place – with iCloud it seems like the hackers were able to keep guessing passwords multiple times. I can't see an open system being so lax with its security. The moral here then, is to trust no-one with your data, but if you do have to trust someone, trust Free and Open source software.
**Rob Smith, Guildford**

**Andrew says:** Well, quite. When you hand over data to a company they will spend only as much effort on security as they think is worth it – and if they can plausibly blame someone else for a leak, they will. Accountability is key – that and a decent password.

## MONEY

Thanks for sponsoring OggCamp – I note with interest that 'other' publications don't seem to be offering the same level of support to this fantastic event (I plan to go, though I haven't booked my ticket yet) despite having been around for longer and having the benefit of a big parent company to fund community involvement.
**Laura, Workington**

**Graham says:** Aw, shucks. We want to get involved with events like OggCamp (and PiWars, and SpanConf) because we want Free Software to win, and because we'd be there anyway having a nice time. It's as simple as that. It's nice to get good karma, but it's more important to us that as many people as possible get to see the huge benefits that Linux can bring.



## TWO POINTS FOR A CONVERSION

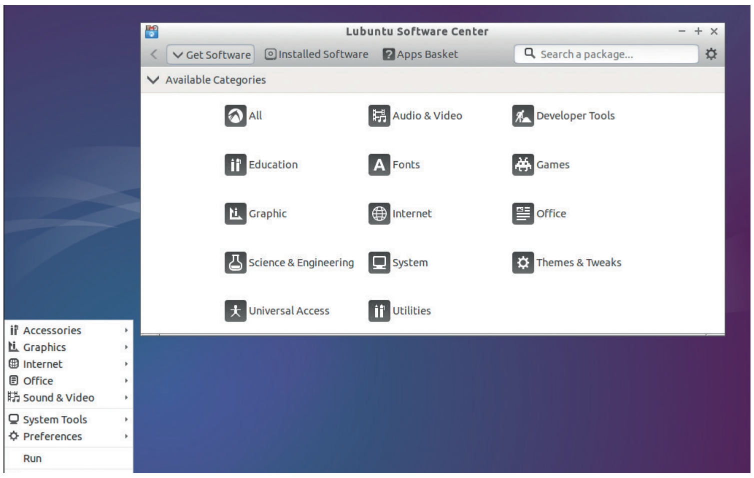
Thought I'd tell you about a recent (small) win for Linux. My father in law is brilliant at fixing broken fan belts, but rubbish with computers, so asked my advice when choosing a new laptop, as he wanted to try out one of the new touchscreen ones that are advertised on the television . Apparently the old one had got so slow it was unusable.

Thinking I could save him a few quid (and myself a few hours of unpaid tech support explaining how to use Windows 8 – no thanks!) I asked to see the old one, backed up the hard drive, then installed Lubuntu.

Result: massive success, brownie points and £400 saved. He can email, browse the web, open spreadsheets to do his accounts, and he's perfectly happy. He's even started to explore the joys of apt-get. He's still baffled by the concept of Free Software. He's always looking for the catch, or expecting an advert to start flashing and prompt him to pay to upgrade, but once he gets over that I'm convinced hat he's a Linux user for the long haul. It's still an old machine – the DVD drive won't last for ever and the battery lasts under an hour – but if it stays out of the landfill site for a year of two I think I've done a good deed. And I'll be there when he buys a new one ready with my Ubuntu disc!

**John, Kilmarnock**

**Ben says:** Fantastic work John, saving the planet one lump of copper at a time. Helping out a relative using Windows 8 feels like such a massive waste of time when you know that Linux us so easy to use. And don't forget to remind him of the money you've saved him on antivurus software the next time you're in the pub with him!



Lubuntu is perfect for resurrecting old machines, as it's light, user-friendly and has all of Ubuntu/Debian's software repositories to plunder.

*Open Source technologists support enterprise grade, mission critical systems*

***Get some for your team***

LINUXJOBS .CO.UK

## GNU'S NOT LINUX

Gnu stands for Gnu's Not Unix, which is both entirely accurate and uninformative.

Are you aware that when you talk about Linux on the desktop, or give advice on the best Linux distro for a certain use case, or even in the naming of your magazine 'Linux Voice', you're doing the GNU/Linux community a great disservice?

Yes, that's right: I wrote GNU/Linux, not just Linux. You may not know or care, but the Linux kernel is only a tiny part of the average user's system. The tools that make Debian GNU/Linux or Red Hat GNU/Linux possible were created long before the Linux kernel was even thought of. You could replace Linux with another kernel and the end user wouldn't notice anything different – the same can't be said about the GNU tools.

The GNU project is developing its own kernel to do just this, and offer a technically superior alternative to Linux. There is already a Debian GNU/HURD distribution, which has nothing to do with Linux, and yet you ignore it.

With the GNU project, the Free Software Foundation and his tireless advocacy work Richard Stallman has done more for Free Software (NB – not 'open source') than Linux, yes his efforts go unappreciated by the wider public. I'd expect that from the BBC (the Biased Broadcasting Corporation), but as purported experts you should really be doing a better job of spreading the truth about Free Software and the ethical points that it entails – and that includes giving proper credit to the people who made it happen

**David Walker, London**

**Andrew says:** Thanks for writing David. I think there are two issues here that have been conflated – the GNU/Linux name and the promotion of Free Software. First, the name: GNU/Linux Voice is longer than 'Linux Voice', so if we printed it on the masthead it would have to go smaller. It looks silly – English orthography is a mess, but it has not yet reached the point where a forward slash in the middle of a word is readable. The Hurd kernel is so far away from being usable that there is no ambiguity then we talk about Linux distros, because there is no workable alternative kernel. So, it's clumsy, it's harder to say, and imparts no information.

The logic that GNU has to be given credit every time we mention a system that used one of its tools is wildly impractical. If it were carried to its logical conclusion, we'd have to say something like GNU/X/Apache/MySQL/KDE/Linux. This would be silly.

The idea that we're not promoting Free Software because we don't use an approved nomenclature doesn't really stand up either. Every month we produce a huge amount of content aimed at helping users get more out of Free Software (and sometimes Open Source too – we prefer Free, but if Open gets the job done, there's nothing wrong with it).

We agree with the FSF's aims. We want to see a world where no money is wasted on software licence fees; where innovation is open and fluid; and where everyone has the freedom to use their computer as they see fit. We're just working towards it from a different angle.

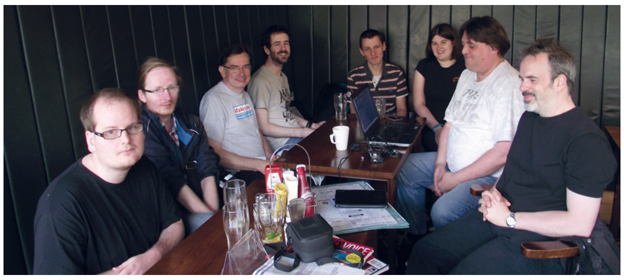
## CALLING ALL LINUX USERS

There may be lots of Linux users in West Lancashire and nearby areas who would welcome a regular meet-up with fellow-enthusiasts. And there must be many people who would simply like to know a bit more about Linux.

I'm suggesting that we could easily arrange a monthly date in a local pub in Ormskirk, without the formality of calling ourselves a Linux Users' Group. We needn't even have an agenda, just an invite to come along and chat. If you're interested, just email me at mauricegeorge71 AT gmail.com

**Maurice George**

**Graham says:** Simple, direct, to the point – we like your style, Maurice! This is how to start a LUG. It doesn't have to be anything fancy, and the most important thing you can bring is an open mind.



Most Linux User Groups in Britain (such as Bristol and Bath LUG) meet in one of our lovely public houses.

# LUGS ON TOUR

## ISACA: Government, risk and compliance

**Neil Curran** president of the ISACA Ireland chapter, writes:

Croke Park will play host to an impressive line up of thought leaders and practitioners in the fields of governance risk compliance (GRC), information systems audit, assurance, privacy and cyber security from Ireland and across the globe.

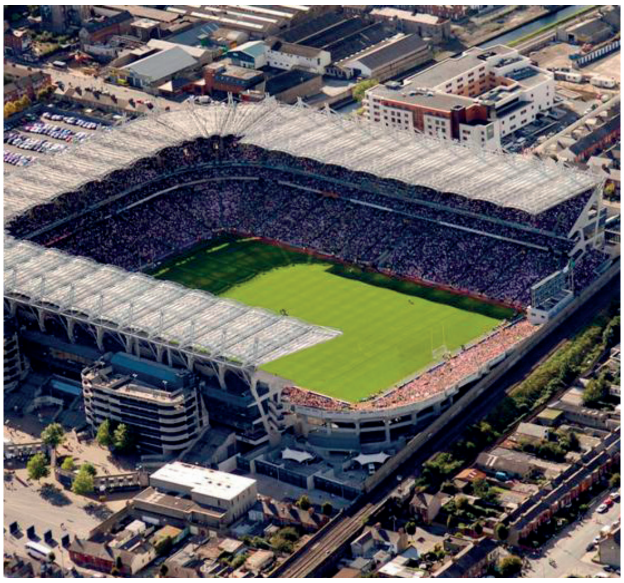
Keynote speeches will be given by renowned cyber security expert and CTO of Cytelligence, Professor John Walker, Patrick Curry, Director of MACCSA (Multinational Alliance for Collaborative Cyber Situational Awareness), the chair of ISACA London's Security Advisory Group, Amar Singh, John O'Dwyer, Deputy Data Protection Commissioner, as well as independent computer security analyst and prolific blogger, Graham Cluley.

We are extremely pleased to announce the launch of our annual conference, which will build upon the success of previous years' events as we aim to provide value to our members and our fellow industry professionals.

After receiving an overwhelming response to our call for papers, we have produced a fantastic programme from speakers all over the world covering risk, GRC and the new COSO framework, detecting malware, harmonising privacy compliance, measuring control effectiveness, application security, securing the supply chain, insider threats and much, much more. We are excited to bring the Irish information systems community together for this educational and great networking event.

The one-day event is open to ISACA members as well as non-members and talks given at the conference will be under six themes including: Audit Management, Cybersecurity, Risk Management, Privacy Management, Application Security and Enterprise Governance.

To find out more and to register for the event, please visit the registration page (**www.eventbrite.co.uk/e/annual-conference-grc-20-breaking-down-the-silos-tickets-11611613649?aff=eorg**). To receive €50 off a non-member registration, please use the code GRCDublin2014.


If you ever get the chance to watch some hurling at Croke Park, do so. It's baffling, but brilliant.

Join ISACA Renew Feedback Shopping Cart Sign In

ISACA® Trust in, and value from, information systems
Ireland Chapter

CHAPTER HOME | ABOUT OUR CHAPTER | BENEFITS | CERTIFICATION | EVENTS | NEWS | CONFERENCE

Home > Ireland Chapter

**Please Sign In**

Members, please sign in to access chapter information, including essential members-only content. If you are not sure you are a member, please sign in, and then click to My ISACA on the main ISACA web site to view your membership status and your chapter.Non-members are welcome to visit the public pages of this site.

SIGN IN

Find this event on Eventbrite

27
DAYS REMAINING UNTIL

**Conference: GRC 2.0 - Breaking Down The Silos**

Croke Park will play host to an impressive line up of thought leaders and practitioners in the fields of governance risk compliance (GRC), information systems audit, assurance, privacy and cyber security from Ireland and across the globe.

Keynote speeches will be given by renowned cyber security expert and CTO of Cytelligence, Professor John Walker, Patrick Curry, Director of MACCSA (Multinational Alliance for Collaborative Cyber Situational Awareness), the chair of ISACA London's Security Advisory Group, Amar Singh, John O'Dwyer Deputy Data Protection Commissioner, as well as independent computer security analyst and prolific blogger, Graham Cluley.

After receiving an overwhelming response to our call for papers, we have produced a fantastic programme from speakers all over the world covering risk, GRC and the new COSO framework, detecting malware, harmonising privacy compliance, measuring control effectiveness, application security, securing the supply chain, insider threats and much, much more. We are excited to bring the Irish information systems community together for this educational and great networking event.

The one day event, is open to ISACA members as well as non-members and talks given at the conference will be under six themes including: Audit Management, Cybersecurity, Risk Management, Privacy Management, Application Security and Enterprise Governance.

Click here to book now

News

Quick Links

I want to... | My Bookmarks | Saved Searches

- Contact Ireland Chapter
- ISACA Journal
- Join ISACA
- Learn about certification opportunities
- View chapter events

Events

- Demolition Derby: A crash course in IS Governance - Belfast
- AGM and Awards Ceremony
- Annual Conference: GRC 2.0 - Breaking Down The Silos
- COBIT 5: Using or abusing it! | Belfast
- COBIT 5: Using or abusing it! | Dublin

A nonprofit, independent membership association, ISACA helps business and IT leaders maximise value and manage risk related to information and technology.

**TELL US ABOUT YOUR LUG!**

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

## Coder Dojo Ham

**Andrés Muñiz Piniella**, writes:

You will have probably heard about Coder Dojo: **https://zen.coderdojo.com**. well, there is a new one starting in Ham, Richmond Upon Thames:

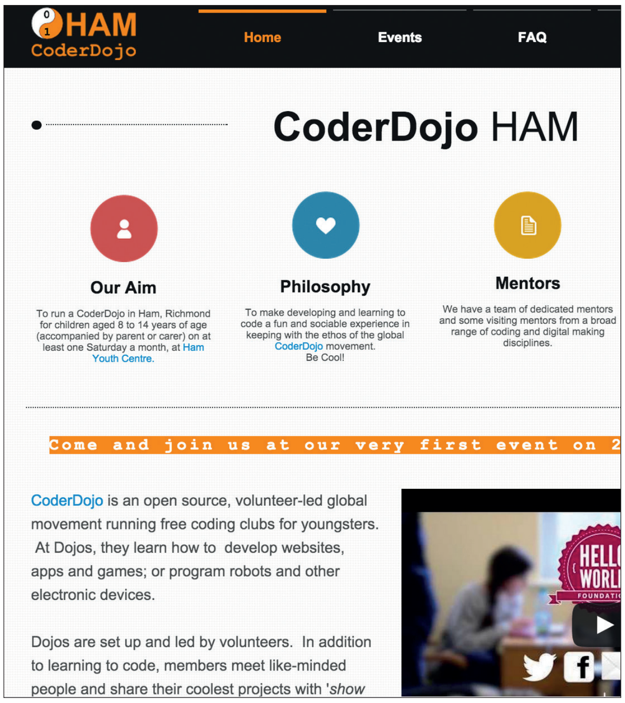
http://www.coderdojoham.org/

This is the local CoderDojo for kids aged 8 to 14 in Ham, Richmond, North Kingston and surrounding areas. We aim to run our Dojo at least monthly from September 2014, so join our mailing list by emailing hamrichmond.uk@coderdojo.com and follow us on Twitter @CoderDojoHam.

The sessions are free but numbers are limited so reserve places to avoid disappointment! Our inaugural session was on 20 September. Bring a laptop if you have one. If not, there are Windows PCs available – please make sure you select the correct ticket type. Be cool!

A parent or carer must accompany their child/children throughout the session and take responsibility for them and for their belongings while on Ham & Petersham Youth Centre premises. Being cool means no bullying, lying or wasting people's time. Please show respect for the Centre's equipment and building; and have consideration for others at all times.

If you're interested in becoming a mentor, get in contact with the team to join the mailing list: hamrichmond.uk@coderdojo.com. If you want to do one-to-one sessions with children you will probably need to have Disclosure and Barring Service (DBS) checks (previously known as CRB checks) but you shouldn't let that stop you if you don't have this: there are other things you can help with – such as lending us your Arduino!



CoderDojo HAM

**Our Aim**

To run a CoderDojo in Ham, Richmond for children aged 8 to 14 years of age (accompanied by parent or carer) on at least one Saturday a month, at Ham Youth Centre.

**Philosophy**

To make developing and learning to code a fun and sociable experience in keeping with the ethos of the global CoderDojo movement.
Be Cool!

**Mentors**

We have a team of dedicated mentors and some visiting mentors from a broad range of coding and digital making disciplines.

Come and join us at our very first event on 2

CoderDojo is an open source, volunteer-led global movement running free coding clubs for youngsters. At Dojos, they learn how to develop websites, apps and games; or program robots and other electronic devices.

Dojos are set up and led by volunteers. In addition to learning to code, members meet like-minded people and share their coolest projects with '*show*

Coder Dojo Ham grew out of the Kings of Hack hacker group in Kingston Upon Thames.

**LISTEN TO THE PODCAST**

**LINUX VOICE**

**WWW.LINUXVOICE.COM**

The conference, held in Chicago, was within walking distance of three kinds of pizza, one jazz and two blues clubs and a 120-tap beer bar.

# LinuxCon and CloudOpen

**Travis Mooney** stalks Linus Torvalds all the way to Chicago to report on the Linux Foundation's flagship conference

The last time I saw Linus Torvalds, we were eating chilli dogs in San Jose. I was a bit star-struck, and I'm sure he doesn't remember me. It was probably 1998, but many years and glasses of single malt have passed since those days. We were (both, separately together) at LinuxWorld, and three interesting things came out of the show keynotes:

- Linus wanted a great Linux desktop.
- Linux was looking at getting into smaller devices.
- IBM announced a major move to get Linux on open-reference Power systems (Longtrail CHRP PowerPC).

LinuxCon (and CloudOpen) 2014 is my first Linux-focused conference since. This time, instead of eating a chilli dog near Linus, I was drinking a beer near Linus. Again, I was a bit star-struck, and there is no chance he remembers me. And again, there were three interesting -- some might say recurring -- themes that came out of the show keynotes:

- Linus wants a great Linux desktop.
- Linux is moving into smaller devices.
- IBM is making a major move to get Linux on open-reference Power systems (OpenPower).

As one of my friends says: the more things change, the more they stay the same. Another one says: everything old is new, again. They both abuse aphorisms terribly. But the truth is, a lot of us have been waiting for nearly two decades for a proper Linux desktop (even though we thought we had it at least twice), Linux is going into smaller devices (soon to power all the untrendy but reliable bacteria), and IBM really wants to sell Power systems to people who don't run AIX (really, they don't care if you just keep them as a large-ish paperweight, as long as the cheque clears).

Unlike LinuxWorld of old, LinuxCon is a travelling show, and this year's North American edition settled in Chicago, Illinois. One of the themes this year was the push towards standardised and accessible training and certification. This centres around two things: a Massive Online Open Course, 'Introduction to Linux', offered through edX, an online learning destination founded by Harvard and MIT; and a new pair of Linux Foundation Certifications (Certified SysAdmin and Certified Engineer), which are both available online, and are backed with optional training programmes. Introduction to Linux, launched this Summer, has already had more than 200,000 student registrations. The Linux Foundation certifications are the first online multi-distribution -- SUSE, Ubuntu, or CentOS -- certifications available..

Driving Linux into smaller devices led to discussions of the Internet of Things (IoT) and whether it calls for a completely new kind of application to data 'Fog Computing', or whether it is all a marketing ploy, as we have always had a bunch of computer and other 'things' attached to the Internet, and hence there has always been an 'Internet of Things'. Represented in one keynote by Cisco, and another by Intel, both sides came out swinging, and it is fair to say that maybe they're both right.

## Linux is everywhere!

We all know that Linux is no longer a hobbyist OS, and Linux as underlying technology was the point of many of the keynotes, including those by Jay Rogers of Local Motors, who is using 3D printing technology and crowd-sourced design to make next-generation cars. Anthony Moschella of MakerBot Linux also talked about the power of Linux as a platform and the creation of an iterative free open-source thingiverse that will change design and manufacturing. Linux is now the platform that powers automobiles, 3D printers, mobile phones and servers.

Contrary to what you might think, the star of LinuxCon wasn't Linus Torvalds. It was 13 year-old Zachary DuPont, who proclaimed Linus his hero in a 6th-grade class assignment. Since Linus (wisely) doesn't disclose his home address, Zachary sent the letter to the Linux Foundation, which arranged for the two to meet at LinuxCon.

CloudOpen sessions included a strong series of presentations on the various ways that Docker is being used to push the 'cattle instead of pets' method of horizontal redundancy, along with OpenStack setup, storage backends, and a number of cloud security issues. Detailed sessions on the way that Google uses containers -- everywhere -- and container security -- current best practice is to run your containers on a hypervisor -- were particularly salient. Clearly, the FOSS industry is currently betting on OpenStack and Docker as the big thing when it comes to virtualisation and deployment.

Data storage, retention, timed deletion, and security in the cloud was another topic that ran through multiple sessions. Encryption as an end-user tool, legal requirements for data retention -- and timely deletion once they have been met -- as well as practical ways to deal with mobile data, were all hot topics.

Many events were co-located with LinuxCon, including the Annual Linux Kernel Summit, the Linux Security Summit, the Xen Project Developer Summit, #MesosCon, the OpenDaylight Mini Summit and the UEFI Mini-Summit. Attendee events included the First-Time Attendee meet-up, the Attendee Welcome Event @ Museum of Science and Industry, and the LinuxCon + CloudOpen Onsite Attendee Reception & Booth Crawl. The convention also took a break from being too serious with Superhero Costume Day, and the Linux Trivia Quiz.

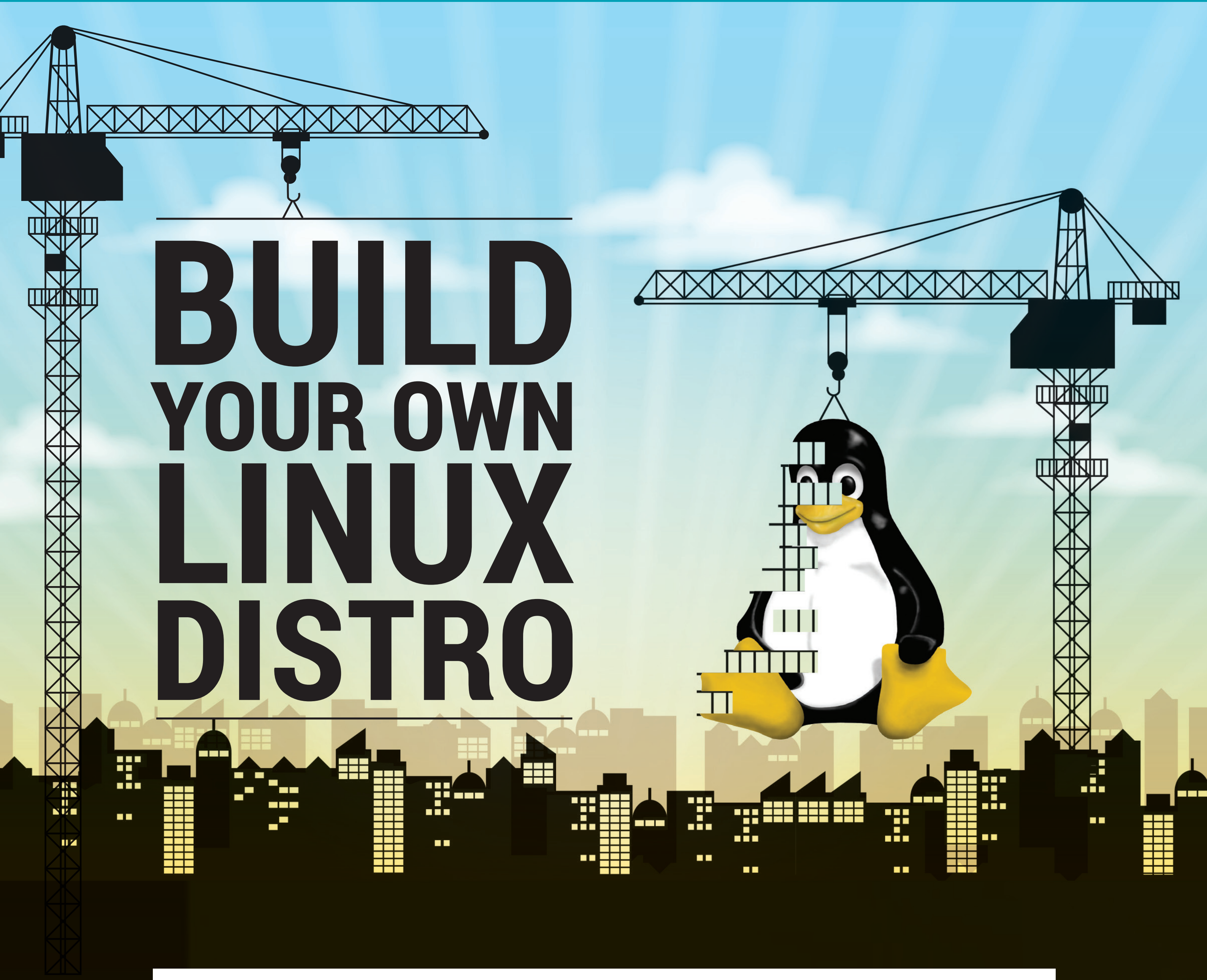
For those who couldn't make it to Chicago, there are videos of each of the keynotes, and many of the session slide shows, available at the LinuxCon website (**http://events.linuxfoundation.org/events/linuxcon-north-america**). The next LinuxCon event is LinuxCon Europe in Düsseldorf, Germany, 13–15 October (**http://events.linuxfoundation.org/events/linuxcon-europe**). LinuxCon Europe will also include CloudOpen, the Embedded Linux Conference Europe, and the KVM Forum.

The convention also took a break from being too serious with Superhero Costume Day, and the Linux Trivia Quiz.

To try to address the gender imbalance in FOSS, workshops were offered both to help women improve their CVs and to help men support women in FOSS roles.

**Travis 'TT' Mooney is COO of Talia Limited, a telecoms and technology company specialising in bringing FOSS solutions and cloud services to the developing world.**

# BUILD YOUR OWN LINUX DISTRO



Do you have a favourite distro that you've spent hours customising? Mayank Sharma shows you how you can spin it into a live distro that you can pass to friends, family, or even on to DistroWatch!

There are hundreds of actively maintained Linux distributions. They come in all shapes, sizes and configurations. Yet there's none like the one you're currently running on your computer. That's because you've probably customised it to the hilt – you've spent numerous hours adding and removing apps and tweaking aspects of the distro to suit your workflow.

Wouldn't it be great if you could convert your perfectly set up system into a live distro? You could carry it with you on a flash drive or even install it on other computers you use.

**"Wouldn't it be great if you could convert your perfectly set up system into a live Linux distro?"**

Besides satisfying your personal itch, there are several other uses for a custom distro. You can spin one with apps that you use in school and pass it around to everyone in class, stuffed with class notes and other study aids. You can do something similar within a professional organisation as well that uses a defined set of apps.

There are various tools for creating a custom distro. We'll start with the ones that are simple to use but offer limited customisation options and move on to more complex ones that enable you to customise every aspect of your distro.

# Quickly create your own Ubuntu

Perfect for mumbuntu and dadbuntu too.

**Difficulty:** Easy
**Level of Customisation:** Basic

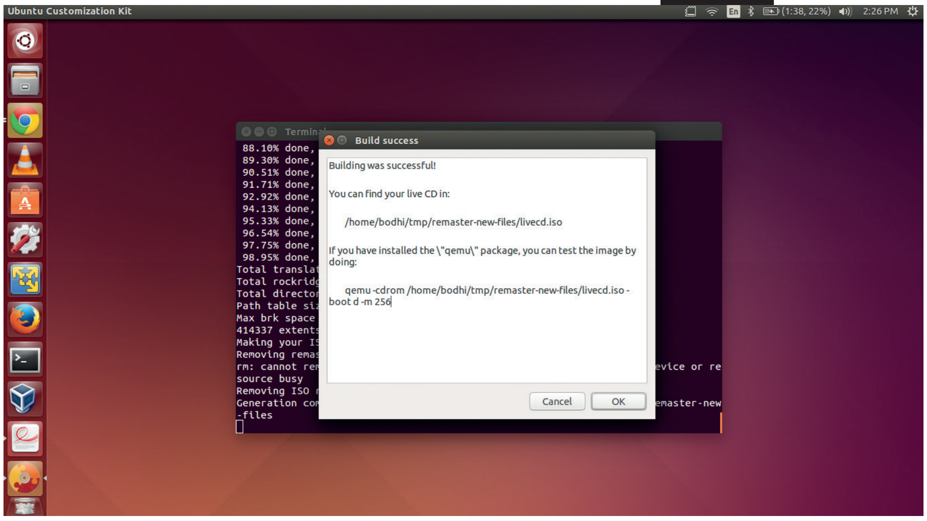
Over the years there have been many tools that help you create a customised version of Ubuntu, which is one reason why there are so many Ubuntu respins out there. While most have fallen through the cracks, the Ubuntu Customisation Kit (*UCK*) lives on.

You can install *UCK* on top of Ubuntu or a derivative distro such as Linux Mint. The tool is in the official repositories and you can install it from the package manager. Additionally, you'll also need the ISO image of the Ubuntu flavour you wish to customise. To simplify the build process, make sure you use the ISO image of the Ubuntu flavour which includes the desktop you want in your customised distro. For example, if you wish to include a localised Gnome desktop in your custom distro, use the Ubuntu Gnome spin instead of the default Ubuntu image. If you're on a 32-bit machine, you'll need the i386 image and not the x86-64 one. However, users of 64-bit OSes can also customise a 32-bit image.

When you launch *UCK*, the app will take you through a wizard after displaying a welcome message with information about its space requirements. In the first couple of steps you'll be asked to select the language packs that you want in your distro along with the boot language. (Make sure the Ubuntu flavour you're customising supports the languages you are building in.)

After you've selected a default language for the distro from the languages you're building in, you'll need to select the desktop environment for your distro. *UCK* will download the localised strings for the desktop in your distro based on the option you select on this screen. You'll then be asked to point to the ISO image of the Ubuntu distro you wish to customise.

*UCK* will then prompt you for a name for your distro before asking if you wish to manually customise the distro. If you choose to do so, *UCK* will launch a terminal window *chroot*ed into the build environment. In the final stages of the wizard *UCK* gives you the option to delete all Windows-related files from your distro and generate a hybrid ISO image that you can burn onto a CD or copy to a USB. Once it's run through these steps, *UCK* will unpack the ISO and then download the selected language packs. You'll then get the option to manually customise the distro, if you selected this option earlier. The Run Console Application option will launch a terminal window and drop you to the root shell of the mounted image.



```
 88.10% done,
 89.30% done,
 90.51% done,
 91.71% done,
 92.92% done,
 94.13% done,
 95.33% done,
 96.54% done,
 97.75% done,
 98.95% done,
Total translat
Total rockridg
Total director
Path table siz
Max brk space
414337 extents
Making your IS
Removing remas
rm: cannot rem                                                  evice or re
source busy
Removing ISO r
Generation com                                                  emaster-new
-files
```

```
Building was successful!

You can find your live CD in:

    /home/bodhi/tmp/remaster-new-files/livecd.iso

If you have installed the \"qemu\" package, you can test the image by
doing:

    qemu -cdrom /home/bodhi/tmp/remaster-new-files/livecd.iso -
boot d -m 256
```

*UCK* lets you customise your distro to the hilt if you know your way around the Ubuntu filesystem.

## Advanced configurations

From this window you can use the **apt-get** package manager to remove default packages and add new ones. For example, you can use **apt-get install ubuntu-restricted-extras** to install plugins to handle multimedia in various formats. If you're creating a distro for low-end machines you can uninstall *LibreOffice* with

```
apt-get remove --purge libreoffice* /
```

and replace it with *AbiWord* using

```
apt-get install abiword
```

If you want to put application shortcuts on the desktop, first create the **Desktop** directory under your custom distro with

```
mkdir -p /etc/skel/Desktop
```

You can now copy the application shortcuts for any installed apps, such as

```
cp /usr/share/applications/firefox.desktop /etc/skel/Desktop
```

and make them executable with

```
chmod +x firefox.desktop
```

If you want to change the default wallpaper, open the **/usr/share/glib-2.0/schemas/10_ubuntu-settings.gschema.override** file in a text editor and change the **picture-uri** parameter to point to the image you wish to use as the background, such as:

```
picture-uri='file:///usr/share/backgrounds/Partitura_by_Vincijun.jpg' /
```

Similarly, you can change the theme and icons by editing the respective parameters in this file. For example, if you wish to change the Ambiance theme to Radiance and use the HighContrast icon set, make sure the file reads as below:

```
[org.gnome.desktop.interface]
gtk-theme="Radiance"
icone-theme="HighContrast"
...
[org.gnome.desktop.wm.preferences]
theme="Ambiance"
```

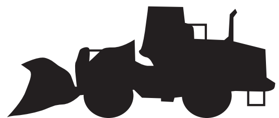
Once you've edited this file, make sure you compile the modified schemas with

```
glib-compile-schemas /usr/share/glib-2.0/schemas
```

You can also copy files into the live CD you are customising. To do this, launch another terminal and **cd** to **~/tmp/remaster-root/**, which is the root of the customised live CD. You can copy files into their appropriate folders under the **remaster-root** and *UCK* will include them in the live CD. For example, you can copy custom shortcuts and folders to **Desktop** with

```
sudo cp -r ~/Documents/README.txt ~/remaster-root/etc/skel/Desktop
```

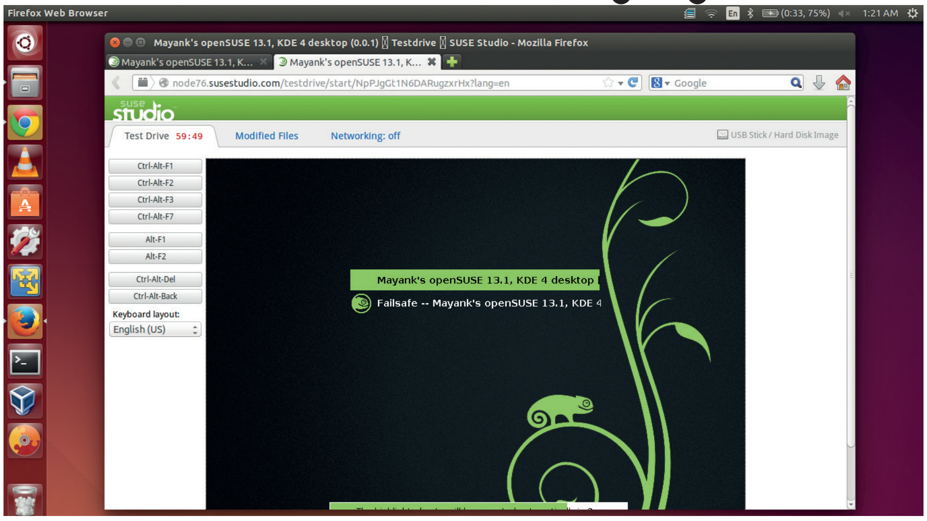
Once you're done, close the **chroot** terminal window and select the Continue Building option in the *UCK* wizard. The tool will now build your new localised Ubuntu distro and point you to the freshly baked customised ISO image.



# Point-and-click distros

Use SUSE Studio to assemble a distro using the web browser.

**Difficulty:** Straightforward
**Level of Customisation:** Moderate


Firefox Web Browser
Mayank's openSUSE 13.1, KDE 4 desktop (0.0.1) | Testdrive | SUSE Studio - Mozilla Firefox
node76.susestudio.com/testdrive/start/NpPJgGt1N6DARugzxrHx?lang=en
SUSE studio
Test Drive 59:49
Modified Files
Networking: off
USB Stick / Hard Disk Image
Ctrl-Alt-F1
Ctrl-Alt-F2
Ctrl-Alt-F3
Ctrl-Alt-F7
Alt-F1
Alt-F2
Ctrl-Alt-Del
Ctrl-Alt-Back
Keyboard layout:
English (US)
Mayank's openSUSE 13.1, KDE 4 desktop
Failsafe -- Mayank's openSUSE 13.1, KDE 4

You can test your images in SUSE Studio's web-based TestDrive before downloading them.

SUSE Studio is perhaps the easiest tool for creating custom distros. The app is graphical and works inside a web browser. It needs only a web browser and an internet connection, and while it creates OpenSUSE-based images you can operate SUSE Studio from any distro. With SUSE Studio you can create full-fledged desktop distros, minimal dedicated servers, and targeted virtual appliances. You can use the web interface to add users, customise the list of apps and even add files and customise the artwork.

Point your web browser to the SUSE Studio website at **www.susestudio.com** and create an account. Alternatively, you can sign into the service using any OpenID provider, such as Google, Yahoo, Twitter, Facebook, etc. Once you've signed in, click on Create New Appliance on the Dashboard. SUSE Studio refers to the custom distros as an appliance irrespective of whether it's designed for physical hardware or a virtual machine.

Before you can begin building your distro, you need to select a base template from one of the predefined ones. The templates help infuse the custom distro with essential packages for your distro. There are templates for the latest and the previous OpenSUSE release, OpenSUSE 13.1, OpenSUSE 12.3, as well as for the SUSE Linux Enterprise distro. Unless you have a licence for SLES, you'll want to base your distro on one of the OpenSUSE templates.

The Just enough OS (JeOS) template is ideal for building a minimalistic system. Then there's the Server template, which helps you build text-only server distros. Finally there are templates that help customise a Gnome 3 or KDE 4-based desktop distro. Once you've selected a base template, scroll down the page and select the processor architecture for the distro.

**"SUSE Studio can be used by virtually anyone, regardless of their level of Linux expertise."**

Click on the Create Appliance button to build the base image, on which you can build your customised Linux distro.

## Rolling start

You're now at the at the main screen of your appliance, which has a set of tabs to help you customise different aspects of your distro. The first tab, labelled Software, is where you choose software packages. Under this tab, you've got a list of the enabled repositories and the list of software that's already installed in your distro. Both of these are based on the template you selected earlier.

To install additional software, use the Find box on the page to look for packages in the repositories. When you find what you're looking for, just hit the corresponding +add button to include it in your distro. SUSE studio will automatically check for and add any dependencies. If the package you've just added conflicts with an existing one, you'll get options to resolve the issue by removing one of the two conflicting packages. If you have some custom apps you can also add their RPMs from this page.

In case the software you wish to add isn't in the default repositories, you can also add additional repos with the Add Repositories option. This brings up a page that's similar to the one for adding software. Once the repositories have been added, SUSE Studio will list them under the Software tab and allow you to search for packages inside them as well.

## Make it your own

The bulk of the configuration is handled from under the Configuration tab. This tab is further divided into seven different sections for configuring different aspects if your distro. From the General section you can localise the distro and select the default language and keyboard layout along with the time zone. You can also select how you want your distro to configure the network (DHCP is usually a safe bet) and enable the firewall and open ports for remote access. This is also where you add any users and groups. The Personalise section is where you choose the custom artwork for your distro. You can either select one of the listed ones or upload your own.

You can avoid visiting the Server tab, which only has options to add data to either a *PostgreSQL* or a *MySQL* server. Similarly, if you're setting up your distro for a virtual machine, head to the Appliance tab to configure related settings. However, most desktop users should just head to the Desktop tab, from where you can automatically log in any added user and define any apps that you want to autostart.

If you consider yourself an advanced user, you can take a look at the Scripts sections,

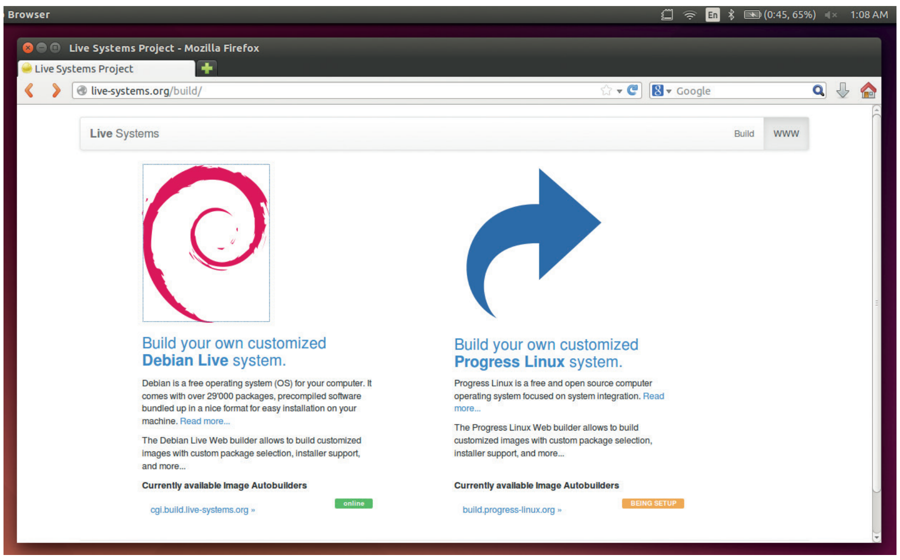
## Other online distro builders

SUSE Studio isn't the only web-based service for creating Linux distros. The Debian Builder (**http://live-build-cgi.debian.net/cgi-bin/live-build**) is hosted by the Live Systems project, which produces the tools that are used for producing official Debian live images. The service can create basic netboot images without the X server as well as hybrid ISO images that boot from USB disks.

You can create a basic distro by selecting a handful of options including the Debian branch you want the image to be based on (Wheezy, Jessie, Sid) and the predefined group of packages (Gnome Desktop, KDE Desktop, Mate Desktop, Rescue, etc).

Advanced users can also tweak additional advanced options. You get options to choose the architecture of the build, the filesystem of the **chroot** environment, the bootloader, whether it should include the Debian installer, and a lot more. The service will email you once your customised Debian Live system is ready to be downloaded.

Then there's the Porteus Wizard (**http://build.porteus.org**). Porteus is a small portable distro that's based on Slackware. Using its straightforward but feature-rich web interface you can build a customised version of Porteus with your choice of desktop environment (KDE4, Mate, LXDE, Xfce) and a host of popular software including web browsers (*Firefox*, *Chrome*, *Opera*), word processors (*LibreOffice*, *AbiWord*), VoIP client (*Skype*), graphics drivers for Nvidia and AMD Radeon, and more. You can also customise advanced boot parameters such as setting a custom size for a tmpfs partition and enabling the zram kernel module.



Build your own customized **Debian Live** system.

Debian is a free operating system (OS) for your computer. It comes with over 29'000 packages, precompiled software bundled up in a nice format for easy installation on your machine. Read more...

The Debian Live Web builder allows to build customized images with custom package selection, installer support, and more...

**Currently available Image Autobuilders**

cgi.build.live-systems.org » online

Build your own customized **Progress Linux** system.

Progress Linux is a free and open source computer operating system focused on system integration. Read more...

The Progress Linux Web builder allows to build customized images with custom package selection, installer support, and more...

**Currently available Image Autobuilders**

build.progress-linux.org » BEING SETUP

Go to **http://live.debian.net/manual/stable/html/live-manual.en.html** for more information.

from where you can run custom scripts. This section lets you define scripts that run at the end of the build as well as those that run every time you boot the custom distro.

Once you're done with the sections under the Configuration tab, move on to the Files section to add either single files or an archive of files to the custom distro. All files are added to the **/** directory. However, once they have been uploaded you can select the files and move them into other locations. For example, if you wish to include a file on the Desktop it should be placed under **/etc/skel/Desktop.**

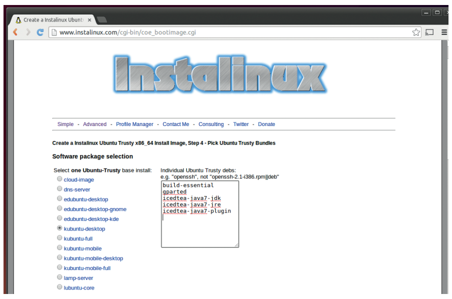
Now that you've customised your distro it's time to ask SUSE Studio to convert it into a usable distro. Head to the Build tab, which lists options to transform the distro into various formats. You can, for example, create a Live ISO image of your distro meant for optical drives as well as live images for USB and images for virtually every virtualisation software available, including *KVM*, *VirtualBox*, *VMware*, *Xen* and more. In order to create a traditional installation image, select the Preload ISO (.iso) option.

When you've select the format, hit the Build button to create your distro, which will only take a few minutes. If you've selected additional formats as well, click on the Build Additional button to get images in the other formats. SUSE Studio also assigns a version number to your distro. Every time you modify the distro, it will increment the version number and automatically generate a changelog that'll list all the changes since the last version.

## Create a customised Ubuntu install image

If you want to roll out Ubuntu on a bunch of identical machines with similar configurations and the same software, like in a lab or office, you can save yourself some time by creating automated installer images. The **www.instalinux.com** service is an online service like SUSE Studio, but instead of full-fledged OpenSUSE-based distros, it churns out small ISOs that are designed to prepare ready-to-use Linux machines by automatically fetching packages and installing them.

The web service is powered by the SystemDesigner CGI scripts from the Linux Common Operating Environment project (**http://linuxcoe.sourceforge.net**). The interface takes you through the steps involved in installing a distro, such as selecting a keyboard layout, timezone, password for the root user, package selection and the disk partitioning scheme. Once you've answered the questions, it creates a preseed installer and puts it on a small (about 30MB) CD.


Create a Instalinux Ubuntu
www.instalinux.com/cgi-bin/coe_bootimage.cgi
Instalinux
Simple - Advanced - Profile Manager - Contact Me - Consulting - Twitter - Donate

**Create a Instalinux Ubuntu Trusty x86_64 Install Image, Step 4 - Pick Ubuntu Trusty Bundles**

**Software package selection**

Select **one Ubuntu-Trusty** base install:

- ○ cloud-image
- ○ dns-server
- ○ edubuntu-desktop
- ○ edubuntu-desktop-gnome
- ○ edubuntu-desktop-kde
- ● kubuntu-desktop
- ○ kubuntu-full
- ○ kubuntu-mobile
- ○ kubuntu-mobile-desktop
- ○ kubuntu-mobile-full
- ○ lamp-server
- ○ lubuntu-core

Individual Ubuntu Trusty debs:
e.g. "openssh", not "openssh-2.1-i386.rpm||deb"

```
build-essential
gparted
icedtea-java7-jdk
icedtea-java7-jre
icedtea-java7-plugin
```

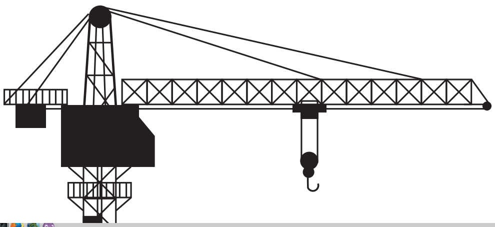
In addition to software bundles, Instalinux can also install individual applications.

### Take it for a spin

After the image has been built, you can test it from within your browser with the Testdrive option. Once you're satisfied, use the Download option to grab the image of your custom distro. You can also share your distro with other SUSE Studio users by heading to the Share tab, where you get textboxes to describe your distro. Once you have the image you can use it as you would any other distro image.

SUSE Studio has a very low threshold of entry and can be used by virtually anyone regardless of their level of Linux expertise. Most of the time-consuming and heavy-duty tasks, like fetching packages and assembling the distro, happen at the remote SUSE servers. You can also test the images remotely and only grab them once you're satisfied with your creation. The system also preserves your build system, and you can tweak it and make changes without much fuss. It's a great place to start.

# Wear a different hat



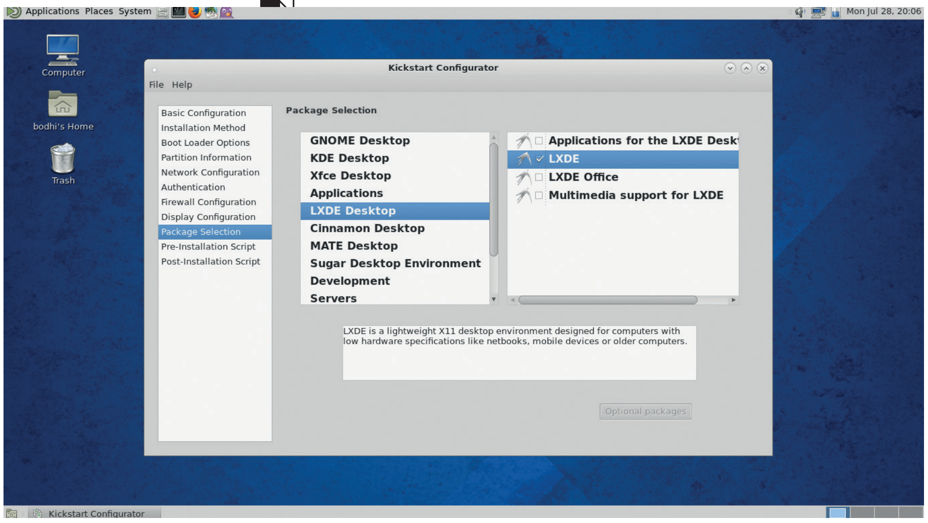
Create distros based on Fedora Linux.

**Difficulty:** High
**Level of Customisation:** Moderate

If you live in RPM-land and are more adept with Fedora, you can put together a customised distro using its *livecd-creator* tool. This is a set of scripts that are available in the official Fedora repositories. Unlike *UCK*, *livecd-creator* works solely on the command line, and instead of an ISO image of a Fedora release, you can grab all the packages you need in your custom Fedora distro from the internet.

The scripts use the powerful Kickstart files to set up your customised Fedora-based distro. If you haven't heard of them before, a Kickstart file is a simple text file containing a list of actions such as package names. The *livecd-creator* tool compiles your distro as per the instructions in this file.

To help you get started, you can download the Kickstart files for several Fedora spins by grabbing the **spin-kickstarts** package from the repositories. Once this is installed, you'll have a bunch of Kickstart files under the **/usr/share/spin-kickstarts** directory. You can customise any of these Kickstart files by editing them in any text editor.



LXDE is a lightweight X11 desktop environment designed for computers with low hardware specifications like netbooks, mobile devices or older computers.

New users will be well advised to use the graphical *Kickstart Configuration* tool (found in the system tools) for selecting software for their custom Fedora-based distro.

> **"Unlike Ubuntu Customisation Kit, Fedora's livecd-creator works solely on the command line."**

Although they are fairly straightforward and well documented, you can browse the Fedora wiki (**http://fedoraproject.org/wiki/Anaconda/Kickstart**) to get a hang of the various options.

You'll also save yourself some time by grabbing the *Kickstart Configurator* tool with

```
yum install system-config-kickstart
```

This tool has an easy-to-navigate graphical interface for creating a Kickstart file.

## Kick the tires

You can specify the packages you want inside your custom distro by listing them under the **%packages** section. Here, in addition to individual packages, you can also specify groups of packages such as **gnome-desktop**. You can also pull in packages from another Kickstart file by specifying its name and location with the **%include** parameter, such as

```
%include /usr/share/spin-kickfedora-live-base.ks
```

## Post installation

If you need to run commands after the live environment is up and running, such as for configuring the network, you need to specify them under the **%post** section. So if you wish to automatically launch *Firefox* you can place a shortcut to the app in the **~/.config/autostart** folder, and your **%post** section should have the following lines:

```
%post
# autolaunch Firefox
mkdir -p /etc/skel/.config/autostart
cp /usr/share/applications/firefox.desktop /etc/skel/.
config/autostart/
%end
```

Make sure that the **%packages** and **%post** sections are closed with **%end**. If you wish to run any commands outside the build environment, such as to copy files from the host distro to the custom distro, you can add the **--nochroot** parameter to **%post** like so:

```
%post --nochroot
#copy resolv.conf from host to the custom distro
cp /etc/resolv.conf $LIVE_ROOT/etc/
%end
```

The **$LIVE_ROOT** is a variable that points to the live environment. You can similarly copy any file from the host system to the live environment, for example:

```
cp -r /home/bodhi/Music $LIVE_ROOT/
```

The one important line you'll have to add manually to the Kickstart file if you use the graphical tool is the repository definition. This line points to the list of mirrors for the Fedora repository (along with the version and architecture information) from where the tool will pull in packages. So if you wish to grab packages from Fedora 21's repository for the 64-bit architecture, enter

```
repo --name=fedora --mirrorlist=http://mirrors.
fedoraproject.org/mirrorlist?repo=fedora-
21&arch=x86_64
```

Once your Kickstart file is all set up you can feed it to the *livecd-creator* tool for creating the custom distro. Assuming it's saved as **~/custom-kickstarts/Custom-Fedora.ks**, you can create your custom distro with the command:

```
sudo livecd-creator
--config=/home/bodhi/custom-kickstarts/
Custom-Fedora.ks
--fslabel=FedoraUltimate
--cache=/var/cache/live
--verbose
```

The **--fslabel** switch specifies the name for your custom distro. When the tool has run through all the instructions in the Kickstart file, it'll assemble the ISO image for your distro and place it in your home directory ready for you to **dd** it to a USB stick.

# Bake your own pie

Create your own Raspberry Pi distro.

**Difficulty:** Average
**Level of Customisation:** High

The *New Out Of the Box Software*, or *NOOBS* is the Raspberry Pi's official installer. It has simplified and standardised the procedure for installing a distro on the Raspberry Pi. While the main purpose of *NOOBS* is to simplify the installation of an operating system on to the Pi, the tool can also be used to create a custom distro.

To get started, grab the *NOOBS* installer from the website and install any of the supported distributions that you want to customise. We'd advise you to use the Raspbian distribution, which is also recommended by the *NOOBS* installer.

After you've installed Raspbian, boot the distro and make whatever changes you want. You can change the default wallpaper and also switch themes by running the **obconf** command from the command line, and you can install additional themes with:

```
sudo apt-get install openbox-themes
```

You can also install and remove apps either directly via **apt-get** or by first installing the graphical *Synaptic* package manager.

You can copy over any files into this Raspbian installation. *NOOBS* lets you create a 512MB partition that you can use to store files. Or, you can use the

```
raspi-config
```

command to expand the root partition to fill the SD card. Also make sure you set up the distro to work with your network hardware straight out of the box. So for example, you can configure the wireless adapter to connect to your Wi-Fi access point and access network services such as the directory server, or change the default browser page to point to your intranet landing page.

When you're done setting up the distro, it's time to package it into an archive. Change to the root directory with **cd /** and enter the following command:

Xpdf: /home/pi/Desktop/Linux Voice articles/robot.pdf

You'll also need a Linux machine to do some development on, and an Android phone (other smart

(other controll
The most imp
the brains of t
Python (almos
on the board's

**The build**
The chassis c
like. Specialise
robust and ca
don't need this
can use anyth
on it and it wil
Finally, we u
to power the F
quite a lot of h

We made a custom version of Raspbian for LV006's cover DVD – with NOOBS, you can too.

```
sudo tar -cvpf root.tar /* --exclude=proc/*
--exclude=sys/* --exclude=dev/pts/*
```

This command can take up to half an hour to complete depending on the number of changes you've made to Raspbian.

When it's done, you'll have a file called **root.tar** in the root directory. Similarly now roll up the boot files. First, move into the **boot** directory with

```
cd /boot
```

and then create the archive with the

```
tar -cvpf boot.tar
```

command. This will not take much time, and when it's done you'll have a file called **boot.tar** in the **boot** directory.

*NOOBS* requires compressed versions of these files. But the Raspberry Pi doesn't have the resources to squeeze these files. So you'll have to move them out to a regular desktop PC where you can compress them with the **xz -9 -e boot.tar** and **xz -9 -e root.tar** commands. This will replace the files with their compressed versions, namely **boot.tar.xz** and **root.tar.xz**.

Now format the SD card and extract a fresh copy of *NOOBS* into it. Use the file manager to navigate to the **os** directory under the newly extracted files. This directory further contains a number of directories, each of which containing the files for a supported distro including Arch, Pidora, Raspbian and others. Since our custom distro is based on Raspbian, we can remove all the other directories from under the **os** folder. Rename the **Raspbian** folder to the name for your custom distribution.

Head inside this folder and open the file named **os.json** in a text editor. In the file, replace the text beside the **name** and **description** fields from that of the original Raspbian distribution to your custom one. Also, make sure you remove the file named **flavours.json**. You can also optionally change the artwork of the distribution.

Finally, remove the existing **root.tar.xz** and **boot.tar.xz** files from under this folder and replace them with the ones you've just created. That's it! Now boot the Pi with this card. The *NOOBS* menu will now list your unique, customised Linux distro.

**"The main purpose of NOOBS is to simplify the installation of an operating system."**

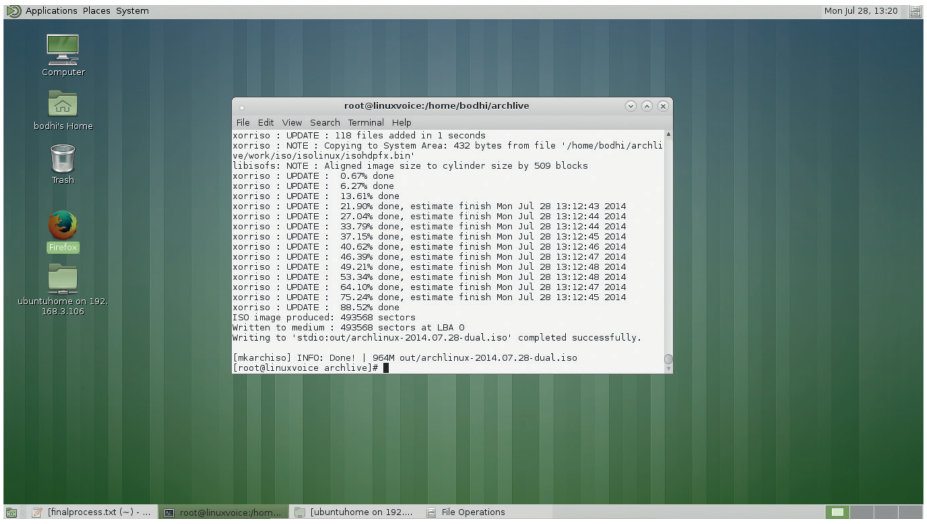
# Made-to-order distros

Build your Arch-based custom distro from the ground up.

**Difficulty:** Extreme
**Level of Customisation:** High

If you have the patience to hand craft a custom distro from scratch, you should build one on top of Arch Linux. The distro's approach to allow the user to craft their installation from the ground up makes it an ideal platform for cultivating a custom distro without the code bloat and package proliferation that afflicts so many other popular distros.

You can create a custom Arch-based distro with the command-line *Archiso* utility. The utility is a collection of Bash scripts, and although it has a steep learning curve it gives you a lot of control over the final result.


Applications Places System
Mon Jul 28, 13:20
Computer
bodhi's Home
Trash
Firefox
ubuntuhome on 192.168.3.106
root@linuxvoice:/home/bodhi/archlive
File Edit View Search Terminal Help
[finalprocess.txt (~) - ...
root@linuxvoice:/hom...
[ubuntuhome on 192....
File Operations

```
xorriso : UPDATE : 118 files added in 1 seconds
xorriso : NOTE : Copying to System Area: 432 bytes from file '/home/bodhi/archli
ve/work/iso/isolinux/isohdpfx.bin'
libisofs: NOTE : Aligned image size to cylinder size by 509 blocks
xorriso : UPDATE :  0.67% done
xorriso : UPDATE :  6.27% done
xorriso : UPDATE :  13.61% done
xorriso : UPDATE :  21.90% done, estimate finish Mon Jul 28 13:12:43 2014
xorriso : UPDATE :  27.04% done, estimate finish Mon Jul 28 13:12:44 2014
xorriso : UPDATE :  33.79% done, estimate finish Mon Jul 28 13:12:44 2014
xorriso : UPDATE :  37.15% done, estimate finish Mon Jul 28 13:12:45 2014
xorriso : UPDATE :  40.62% done, estimate finish Mon Jul 28 13:12:46 2014
xorriso : UPDATE :  46.39% done, estimate finish Mon Jul 28 13:12:47 2014
xorriso : UPDATE :  49.21% done, estimate finish Mon Jul 28 13:12:48 2014
xorriso : UPDATE :  53.34% done, estimate finish Mon Jul 28 13:12:48 2014
xorriso : UPDATE :  64.10% done, estimate finish Mon Jul 28 13:12:47 2014
xorriso : UPDATE :  75.24% done, estimate finish Mon Jul 28 13:12:45 2014
xorriso : UPDATE :  88.52% done
ISO image produced: 493568 sectors
Written to medium : 493568 sectors at LBA 0
Writing to 'stdio:out/archlinux-2014.07.28-dual.iso' completed successfully.

[mkarchiso] INFO: Done! | 964M out/archlinux-2014.07.28-dual.iso
[root@linuxvoice archlive]#
```

Arch Linux is already pretty snappy, but once you've mastered *Archiso* you can use the tool to create streamlined distros that can outperform all others.

## Setup the build

The first thing you need before you can use *Archiso* is an Arch Linux installation. If you don't already use Arch, follow Graham's tutorial in LV001 and also available on the LV website (**www.linuxvoice.com/arch-linux**) to setup a working Arch Linux system.

Once you've installed Arch on your computer, the next step is to customise it to your liking. That includes installing more packages, swapping out the default themes and artwork of your desktop environment and configuring other aspects of the systems such as the network. Later on, we'll copy these customisations and configurations from the installed instance of Arch over to the custom distro we're building.

When you're done customising the Arch installation, fire up a terminal and install the dependencies for *Archiso* with:

```
pacman -S make  squashfs-tools  libisoburn
dosfstools  patch  lynx devtools git
```

Now fetch the latest version of the **archiso** package from its Git repository with

```
git clone git://projects.archlinux.org/archiso.git
```

This will fetch the files inside the **~/archiso** directory. Move into the directory and install the tool with **make install**. Once it's installed, you can safely remove the **~/archiso** directory. Next, we'll create a directory where we'll tweak the files for our custom distro with

```
mkdir ~/archlive
```

Make sure you have enough free disc space to accommodate all the apps you wish to install, along with any files you want to copy over to the custom distro.

Now you need to copy over one of the two *Archiso* profiles. The **baseline** profile is useful for creating a basic live system with no pre-installed packages. However, we'll use the **releng** profile, which lets you create a fully customised Arch Linux with pre-installed apps. To use these scripts, simply copy them over to the **~/archlive** directory, like so:

```
cp  r /usr/share/archiso/configs/releng/ ~/archlive/
```

"Once you've installed Arch on your computer, the next step is to customise it to your liking."

## Add packages

Telling *Archiso* which packages to put on the custom ISO is as simple as adding them to a text file, one package name per line. Under the **~/archlive** directory you'll have three files: **packages.i686**, **packages.x86_64**, and **packages.both**. You can open these files in a text editor and include the names of the packages you want in your distro. *Archiso* will read the files for the respective architecture and include them packaged in the final ISO, which will be a dual-boot ISO that'll work on both 32-bit and 64-bit machines. However, for consistency we recommend you add the app names to the **packages.both** file so that they are available on both the architectures.

The **packages.both** file already lists a bunch of packages. You should leave them in there and append your own at the end of the file. Use the

```
pacman -Qqe
```

command to list all the packages installed on your machine, and then copy the ones you need. You can create a barebones system with the Mate desktop, the Simple Login manager and the *Firefox* web browser by adding the following packages in the packages.both file:

```
xorg-server
xorg-xinit
xorg-server-utils
xf86-video-vesa
slim
mate
firefox
```

If you're feeling adventurous you can copy all the packages installed on your machine over to the **packages.both** file with

```
pacman -Qqe >> ~/archlive/packages.both
```

## Configure root

The **airootfs** directory inside **~/archlive/** acts as an overlay for what will be the **/** directory of your new distribution. Any files

you add to this directory will be added to your distro's filesystem, so if you're using the *Slim* login manager, copy over its configuration file with

```
cp /etc/slim.conf ~/archlive/airootfs/etc/
```

Similarly you should also copy the **/etc/systemd/system/display-manager.service** file from the host machine to its corresponding location under **~/archlive/airootfs/**, along with directories that house custom artwork, namely **/usr/share/backgrounds**, **/usr/share/icons**, and **/usr/share/themes**.

If you want your custom distro to have the same users as your host machine, copy over the relevant files with

```
cp /etc/{shadow,passwd,group} ~/archlive/airootfs/etc/
```

Before you can copy over any files that you want within the user's /home directory, you need to create the skel directory with

```
mkdir ~/archlive/airootfs/etc/skel
```

This directory represents the home directory of the user inside the system under development. You can now copy files inside the user's home directory, such as

```
cp ~/.bashrc ~/archlive/airootfs/etc/skel/
```

Similarly you can copy over any files and directories from under your home directory to the **skel** directory, including **~/.xinitrc** and **~/.config**.

To log in automatically as your user instead of the default root user, open the **~/archlive/airootfs/etc/systemd/system/getty@tty1.service.d/autologin.conf** file in a text editor and modify the following line to swap the auto login user:

```
ExecStart=-/sbin/agetty --autologin bodhi --noclear
%I 38400 linux
```

Replace **bodhi** with the name of your user.

## Final configurations

Inside root's home folder (**~/archlive/airootfs/root**) there's a file named **customize-root-image.sh**. Any administrative task that you would normally do after an Arch install can be scripted into this file. Remember that the instructions within the file have to be written from the perspective of the new environment, which is to say that **/** in the script represents the root of the distro that's being assembled.

Open the file in a text editor, find the line that sets **/etc/localtime** and change it to your timezone, eg:

```
ln -sf /usr/share/zoneinfo/Europe/London /etc/localtime
```

Also make sure that the shell is set to *Bash* by changing the **usermod** line to read

```
usermod -s /usr/bin/bash root
```

Then copy the contents of the **skel** directory into your user's home directory with

```
cp -aT /etc/skel/ /home/bodhi/
```

and set proper ownership with

```
chown bodhi:users /home/bodhi -R
```

In both these commands, replace **bodhi** with the name of your user.

Finally, scroll down to the end of the file and comment out all the **systemctl** commands by appending a **#** symbol before them. To boot into the graphical desktop, make sure the correct services are started by adding the following:

```
systemctl enable pacman-init.service choose-mirror.service
systemctl set-default graphical.target
systemctl enable graphical.target
```

That's it. You're now all set to build the ISO for your custom distro. Enter the **~/archlive** directory and run

```
./build.sh -v -N EduArch -V 1.0 -L EduArch_1.0
```

to initiate the build process. The **-v** switch enables the verbose mode, the **-N** switch sets the name of the ISO image, **-V** sets the version number and **-L** appends a label to the generated ISO image.

Note that the build process is slow and can take several hours depending on the available resources of your computer. When it's done it'll place the ISO under the **~/archlive/out** directory.

## Generate updated images

You can now copy the ISO out of the build system and share it with anyone. After a while though, you'll want to update the system. Maybe the included apps have had a newer release since you last created the ISO image, or maybe you need to change any of the other files that you've manually copied into the distro.

To do so, head to the **~/archlive/work** directory. The **i686** and **x86_64** directories under the **work** folder house the filesystems for the corresponding architecture. You can **chroot** into either of them with

```
arch-chroot ~/archlive/work/x86_64/root-image
```

or

```
arch-chroot ~/archlive/work/i686/root-image
```

Once inside, you can perform any updates or changes to the system. If you wish to update the apps, first update the package manager's key database and package list:

```
pacman-key --init
followed by
pacman-key --populate
. Once that's done, you can update the system with
pacman -Syu
```

After you've made the changes, type

```
exit
```

to get out of the **chroot** environment. Remember to make the changes for both the architectures. You're now all set to recreate the ISO image. However, the **build.sh** script will fail to execute, as there's already a **work** folder. To force it to generate a new ISO file, open the **build.sh** file in a text editor. Scroll down to the very bottom of the file and remove the **run_once** parameter from the beginning of the **make_prepare** and **make_iso** commands, so that it reads:

```
for arch in i686 x86_64; do
  make_prepare
done
make_iso
```

Save the file and run the script with

```
./build.sh -v -N EduArch -V 2.0 -L EduArch_2.0
```

to generate the updated iteration of your custom distro.

### Build an embedded Linux distro

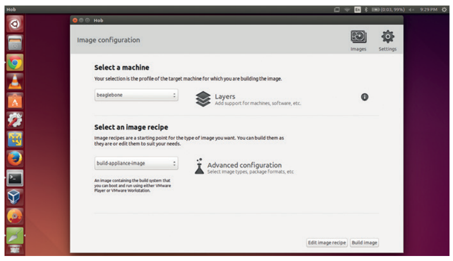
Linux is a popular choice in the embedded space. However the field is saturated with different embedded Linux distributions. To curb this proliferation, the Linux Foundation along with industry leaders such as Intel, AMD, Freescale, Texas Instruments, Wind River and others have created the Yocto Project.

The main aim of the project is to create and make available the build environment and tools for creating an embedded Linux distro. The project supports various 32- and 64-bit embedded architectures such as ARM, PPC, and MIPS. Using these tools developers can build a complete Linux system for an embedded device.

To aid developers the project offers the *Hob* tool, which is a graphical front-end for the project's build engine called *BitBake*. *Hob* reads recipes and follows them by fetching packages, building them, and incorporating the results into bootable images. You can install it on all the popular Linux distros including Fedora 20, Ubuntu 14.04, Debian 7.4, OpenSUSE 13.1 and CentOS 6.5.

To get started download the build system from the project's website using **git** with

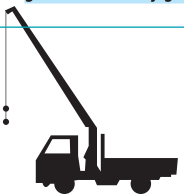
```
git clone -b daisy git://git.yoctoproject.org/poky.git
```

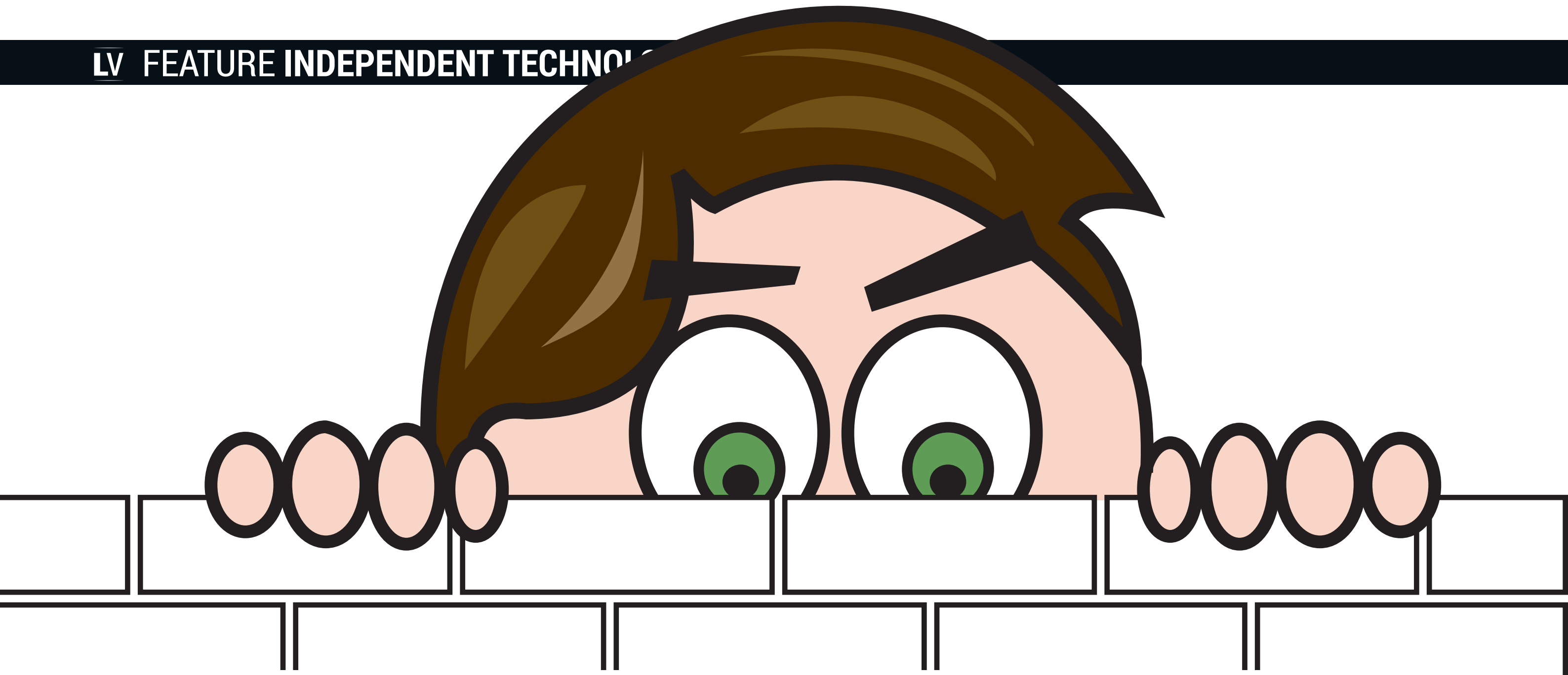


Select a machine
Your selection is the profile of the target machine for which you are building the image.
beaglebone

Select an image recipe
Image recipes are a starting point for the type of image you want. You can build them as they are or edit them to suit your needs.
build-appliance-image
An image containing the build system that you can boot and run using either VMware Player or VMware Workstation.

Unless you are a math geek you wouldn't know that Yocto is the smallest SI prefix.





# INDEPENDENT TECHNOLOGY

## WHEN FREE AND OPEN ISN'T ENOUGH TO PROTECT YOUR PRIVACY.

**Richard Smedley** reports on a new way of building the world that should respect your privacy.

Even in the technology world, events can take time to filter into actions. It's now more than a year since Edward Snowden, a private intelligence contractor for the US National Security Agency (NSA), disclosed thousands of classified documents to the media showing the extraordinary reach of global surveillance programs run by governments - in particular the UK, USA, New Zealand, Canada and Australia.

In December an unusual co-operation between the largest tech companies – Apple, Google, Microsoft, Yahoo, LinkedIn, Facebook, Twitter and AOL, which normally only co-operate on technical issues – produced a joint letter to governments, principally Washington, outlining a series of principles to limit surveillance on the internet. One Microsoft VP, Brad Smith, said at the time: "People won't use technology they don't trust. Governments have put this trust at risk and governments need to help restore it."

While this concern for citizens' data privacy from Silicon Valley isn't the final nail in the coffin of irony, it would be hard to find any collection of companies holding more data on the activities of people than these eight. To a greater or lesser extent it is at the centre of their business. And we have handed it over fairly willingly, as the price of using 'free' internet services. Usually not thinking too deeply about the cumulative power moving from citizens to large internet corporations. "They trust me — dumb f***s," said Mark Zuckerberg of early Facebook users handing over so much information.

### Unequal equity

Perhaps you try to avoid behemoths like Facebook, and use social media and services from smaller internet start-ups, particularly in the Free Software world. The problem here is that software freedom and open data – the place in the stack where geeks look to fix the problem – are still no guarantee of data privacy, as Indie Foundation founder Aral Balkan told Linux Voice: "The problem is a societal problem. And the problem is outside of the stack. The problem starts at the business model."

Idealistic young startup companies are trying to grow quickly, and their greatest asset as they grow, is their data. "Regardless of what kind of company you have," says Balkan, "if you're taking equity investment, if you have venture capital, then you have to have an exit. Either that's being bought by some other company, or you exit to the public with an IPO. Those are the only two possible routes if you take venture capital, or equity." And all of that data ends up with those tech companies with billions in the bank, and petabytes of data on all of us.

To successfully offer a service that respects your privacy and data ownership, Aral Balkan set up the Indie Foundation (which they style as ind.ie/foundation) social enterprise, proposed the Indie Phone (see below), and co-authored the Indie Tech Manifesto to support the creation of organisations that are "independent, sustainable, design-led, and diverse" (**https://ind.ie/manifesto**). Independent because organisations that respect ownership of data must reject equity investment - choosing "bootstrapping, non-equity-based crowdfunding, revenue-based investment." And these organisations will "create a new category of consumer products that are beautiful, free, social, accessible, secure, and distributed."

Sounds head-in-the-clouds idealistic? Not to the organisations and individuals from all over Europe and beyond gathered this summer in Brighton, on England's south coast, to discuss practical solutions at the Indie Tech Summit. From Dutch MEP Marietje Schaake and Danish privacy campaigner Pernille Tranberg to OpenStreetMap's Tom Morris and Global Head of Brand Design at Philips, Thomas Marzano, speakers included not just those concerned with privacy and software freedom but there was also a strong emphasis on good design.

Good design and software freedom, sad to say, do not often coincide, which is something of a bugbear to Balkan: "With Free and Open Software... we have mostly terrible experiences, in the short term, and we say 'Don't worry about that; work around that – have the terrible experience now, because we're protecting you in the long term. We're protecting your fundamental freedoms.' We can't do that: that's arrogant. People deserve great experiences in the here and now. They also need to have the tools in order to protect themselves long-term. So we need to design not just for the short-term, not just for the long-term, but both. I call this whole-term design. [This is] what Indie Tech is about: designing for the whole term."

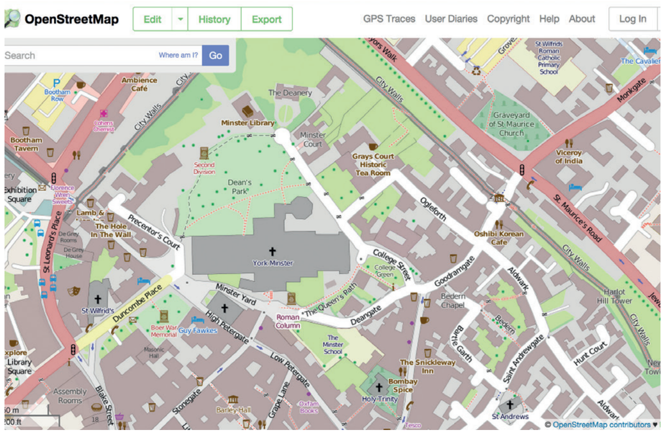
Fittingly for an organisation that wants to pronounce an independence day for the web, the Indie Tech Summit was held on 4 July.

## OpenStreetMap

It's not easy to leave behind the proprietary silos that we rely on, with plug-and-play Indie Tech replacements for services like Gmail not yet ready for the mainstream. But some "free" services, like Google Maps, do have free and open source replacements. OpenStreetMap was set up as the "Wikipedia of mapping", being built entirely from user-contributed data over a bare layer of public domain mapping information.

Like Wikipedia, this makes it better in many cases than single-source, proprietary rivals, as enthusiastic users add new streets as quickly as they appear. Some locations – notably the Netherlands and Cameroon, for example – are more densely mapped than others, but contribution is easy.

With Google tracking the correlation between map searches and where you go through opt-in location services in potentially half a billion Android phones, and FourSquare gamifying its large-scale location data collection, there are privacy concerns with any location or mapping service from corporations. Beyond that, though, is a more fundamental question of ownership of such important data: Google and others decide what businesses and services to show up in searches on their maps – decisions not open to outside governance. OpenStreetMaps gives transparency to the process of mapping, and puts ownership and control of the data in the hands of the commons.



Many people have some sort of GPS device, and local information can easily be added to OpenStreetMap though the web interface.

GNU and FSF founder Richard Stallman addressed the faithful. The Indie Foundation shares many of the FSF's goals.

Lots of us will have accessibility problems on a temporary basis long before old age – through accidents or illness. And usability goals make it easy for everyone to see, read and focus on content. The passion to make technology for everyone explains how Indie brought such a broad range of speakers to the Summit. Lena Reinhard, a writer and community manager involved with Hood.ie and Apache CouchDB, was there to speak on diversity: "For the future of the web, diversity is non-negotiable."

Diversity, Balkan told us, is "a cornerstone for what we're doing... if we don't have that we can't design the right systems." People design first and foremost for themselves, with designers and developers "a very small and too uniform crowd. In comparison," Reinhard told us, "the world population is a highly diverse group of people. When we want to build the future of the web, we'll have to build it not only for us, but for everyone. This is an act of representation. And it means: we'll have to build it with the highly diverse group of people in mind. And we have to be a diverse crowd ourselves. Without diversity, it won't be able to build the future of the web. ...This is why diversity is essential to good design and engineering on a very fundamental level." With the next billion people to connect to the Web predicted to do so through mobile devices, and to be in very different circumstances from the first billion, this is good advice.

**"If we were to take ourselves off Google et al we'd be removing ourselves from modern life."**

## FOSS for all

Indeed, Balkan sees diversity and accessibility as essential to get around one of the main problems with Open Source: "You just learn this really hard-to-learn thing, and then everything else is simple... it's an accessibility problem that we're facing, really. It really is. How do we make free and open accessible to people, and that's where design comes in." It also involves control of the whole experience: "So, Apple and Google: how do you compete with that? You do the same thing. If you do not have control over hardware, software, the services – at a minimum – you can't compete. That's what we're doing... we don't need a carrier, we have full control over the end user experience. We go to the public, we say: 'You know what, if you believe in our vision, that's what we're going to do, at the end of this year...' that's what we're going to announce at the summit. We say 'Hey, if you trust what we're trying to do. If you want an actual alternative: support us.'"

That whole experience means developing indieOS to make it "as invisible as possible. Because when something just works, you can simply forget about it," and getting Indie Cloud to integrate seamlessly, but leave you in ownership and control of all of your data. There have been many starts at Independent Technology in cloud services, but this is the first to look at the whole user experience - and thus perhaps the first to stand a chance at take-up beyond the privacy-concerned, tech community.

Even in that community, we all use the products of these behemoths – because they are, as Indie's Aral Balkan reminded us, consumer essentials: "When we talk about Facebook, when we talk about Google... we're talking about products that are essential to modern life. If we were to take ourselves off of Google, and Facebook, and Yahoo, and LinkedIn, we'd be removing ourselves from modern life."

As the manifesto puts it: "We do not cut people off from their existing networks, we wean them off by making the canonical location of their data a place that they own." The Indie Tech Summit hosted many organisations building solutions to return control of data to the user, including decentralised cloud services like Cloudfleet and Cozycloud, and Linux-based self-hosting cloud OS, arkOS. MailPile gives you a webmail client and service as flexible and simple as Gmail, with speedy search and powerful tagging, yet entirely Free and Open – and you can run your own server anywhere you wish. It makes PGP-signed email easy for non-tech users on all platforms, and is getting better with each alpha release (get to **www.mailpile.is** and try for yourself).

The Indie Foundation's own proposal for seamless services to host your data, Indie Cloud, is – like MailPile – not tied to the organisation. As the ind.ie

Aral Balkan opens up your letters in a parody of what Google and Facebook do with your private emails.

site says: "You can install and run Indie Cloud on your own machine if you want to and we will work hard to make migrating your data from one machine to another as easy as possible. All this means that we could not become another Google even if we wanted to (and it's really the last thing we want to do)."

## Unveiling ind.ie/phone

Putting all of these compelling services, with great design, into a single package to give everyone a device that respects their freedom and privacy is the forthcoming Indie Phone, which aims to make "freedom accessible to all", and to empower everyone to control their own data. "What's an OS? Why should you care? Our thoughts exactly," proclaims the **ind.ie/phone** website. Yes, the phone will be totally Free and Open Source. And yes, Linux enthusiasts will be able to get to a terminal, and write software to the well-documented APIs. But the point is that most users won't have to worry about that. The defaults will be great. Indie has the team together and has started on this, despite the scale of the task.

"Working on something that hasn't been done before, where you are going up against not just conventional wisdom in one of the most successful industries of our age (if you measure success by revenue or profit — which I don't) is definitely not a walk in the park." admitted Balkan. "It has its ups and downs. As much as we're making lots of amazing new friends and getting an increasing amount of support from people who are fed up with the status quo, we are also pissing some people off. I like to think we're pissing the right people off but it just so happens that those are some powerful people. And it's scary to think what someone with lots of billions in the bank can do to you in today's world if you piss them off a bit too much. Of course, that's not stopping us and if we thought that way we'd probably not be doing this in the first place."

Aral Balkan: "We're going to ask people on 8 November to give us the push we need to gain the momentum to make a meaningful dent in the world."

## A fairer phone

The Fairphone project started in 2010, to raise awareness about conflict minerals in electronics funding wars in the Democratic Republic of Congo (DRC). In 2013, an independent social enterprise was set up to design and produce a smartphone which would "open up supply chains, solve problems and use transparency to start a debate about what's truly fair."

For example, instead of avoiding conflict zones like DRC, Fairphone sources conflict-free minerals from within the conflict zone, to ensure an income for people there. They work closely with manufacturers who want to invest in employee wellbeing, and consider the whole life-cycle of the phone.

## Follow the crowd(fund)

In order to get there it will need success in crowdfunding: ind.ie will be running a Thunderclap (**ind.ie/phone/thunderclap**) – starting on 24 October and ending on 8 November (birthday of the late Aaron Swartz, coder, writer, political organiser and internet Hacktivist), when crowdfunding starts – trying to get people signed up for the newsletter, and pledging support for the Thunderclap and crowdfunding. A Thunderclap, for those who avoid social media trends, is a crowdspeaking platform, helping amplify a message by getting users to sign up and agree to share a key message on Twitter, Facebook or Tumblr – in this case, crowdfunding for the first Indie Phone.

Balkan believes they will get support because "there isn't a true alternative right now. And I believe that the world deserves better than this business model that treats people as natural resources to be mined, and to be farmed, and to be surveilled." Like all successful people, Balkan is undaunted by the prospect of failure: "We are going to be working on Indie 20 years from now and others are going to be working on it in 40 years time. The crowdfunding is just the beginning. It's not about 'Hey, fund this phone' ... it's about 'Hey, help us create an organisation that can meet the challenges of our time. Oh, and you'll get an awesome phone that's the first example of this need breed of technology as an amazing bonus!'"

# PLANET LABS: PUTTING LINUX IN SPACE

**Graham Morrison** reports on a pioneer at the heart of a revolution on the final frontier.

Space. It's big. And the costs associated with getting large chunks of human engineered debris accelerated to escape velocity are on a similar scale. The 2010-adjusted costs of the Apollo programme, between 1959 to 1973, for example, come to approximately $109 billion dollars. And it's astronomical costs like these that have undoubtedly helped push investment in space exploration back in various political manifestos. Our current age of austerity must surely be the final nail in the coffin for the kind of governmental sponsorship that helped get mankind to the moon.

This has had a perhaps unsurprising side-effect – the democratisation of space, whereby individuals and companies have been able to take up some of the slack and send create their own space-bound projects, or help space agencies deliver far better value for their more limited money. This is something that would have been unimaginable without the great technological leaps we've made over the last 50 years. To commemorate 40 years since the Apollo 11 mission in 2009, for instance, Google published the original code for the command module and the lunar module for the Apollo Guidance Computer. It's less than 2,000 lines of assembly language.

Choosing Linux isn't about cost. It's about choosing the best solution for the job and not re-inventing the wheel. And this is why Linux is having a profound effect on science and space – it's why the International Space Station switched, with the United Space Alliance being quoted as saying, "We migrated key functions from Windows to Linux because we needed an operating system that was stable and reliable" in the original article on ExtremeTech (**bit.ly/1bD0UWD**), and it's why Linux is such a common component at institutions such as CERN.

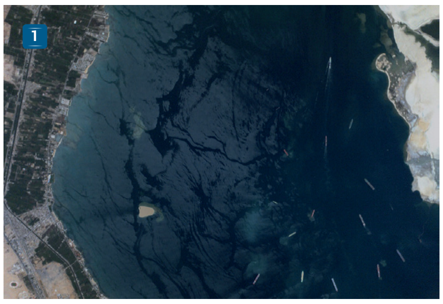
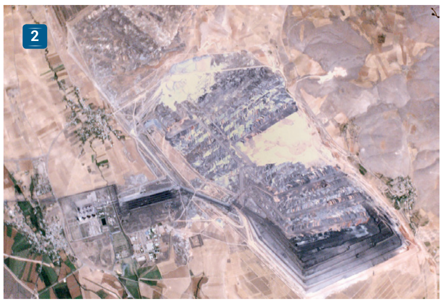
**"Choosing Linux isn't about cost. It's about choosing the best solution for the job."**

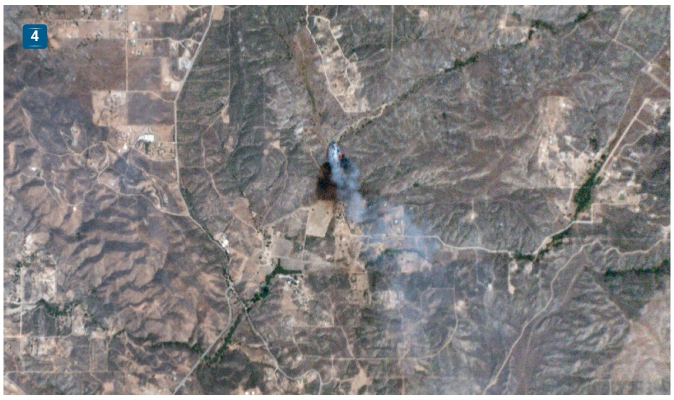
## Planet Labs

But the most recent space-bound use for which we've seen Linux mentioned is as the operating system within an unbelievably small satellite that's (almost) launched by astronauts throwing boxes out of the back of the International Space Station. Yes, as it hurtles across the planet some 330km above us. The project is being run by a private company called Planet Labs. It's still not clear how this company is going to monetise its assets or its innovation but there's obviously a well thought-out business case for all of this. It's still too early to tell. But with that caveat out of the way, what we've seen so far from Planet Labs does genuinely get us excited because not only is Linux and open source at the heart of its technology, it's also attempting to change the world for the better.

The idea is simple enough to visualise; create a large ring of satellites that stay fixed in respect to the sun while the Earth rotates beneath them. Each satellite then takes a picture of every position on the

**Image Gallery** Imagine what might be possible with an API that allows any of us to access daily images of any location



1 With a resolution capable of seeing ships, how about an API that attempts to track their progress around the globe?
2 Even with daily images, such as this one of a coal mine in Turkey, there's often a demonstrable change.
3 The system promises to do things like track crop yield for every single field every single day.
4 This wildfire in Sabina, California, was imaged just 10 minutes after being reported.
5 Or how about tracking the insane amount of development currently in progress in China?

(Images credit: Planet Labs. **https://www.planet.com**. CC-BY-SA)

Earth every 24 hours a day. It's a procedure that Planet Labs CEO, Will Marshall, likens to a line scanner for the planet. The satellites then beam back those images, which are processed and made accessible to everyone through an API, and the resolution is so good that you can make out individual trees. With access to data like this you can easily imagine monitoring deforestation or the shrinking ice caps, the crop yield for different forms of agriculture, or even the size and scale of opencast mining output.

## Eyes in the sky

To get the kind of ubiquitous coverage needed to complete a photo cycle every 24 hours, Planet Labs is going to need more than 100 satellites in orbit. Fortunately, it's well on its way. With the first launch of 28 satellites from the International Space Station in February 2014, it became the largest constellation of earth orbiting satellites in human history, and this was followed by more launches from the ISS and even the Russian Dnepr rocket.

We spoke to one of the founders of Planet Labs (and its CEO), Will Marshall, after he gave an excellent presentation on this very subject at this year's OSCON in Portland. Considering the huge potential for both business and humanitarian efforts, our first question was whether both aspects to the image data would take equal precedents.

"Yes, absolutely," he replied, "I don't know if they take equal precedents – I would say our overriding goal is to help humanity with the data, but it's great to have a solid business case to help to boost that."

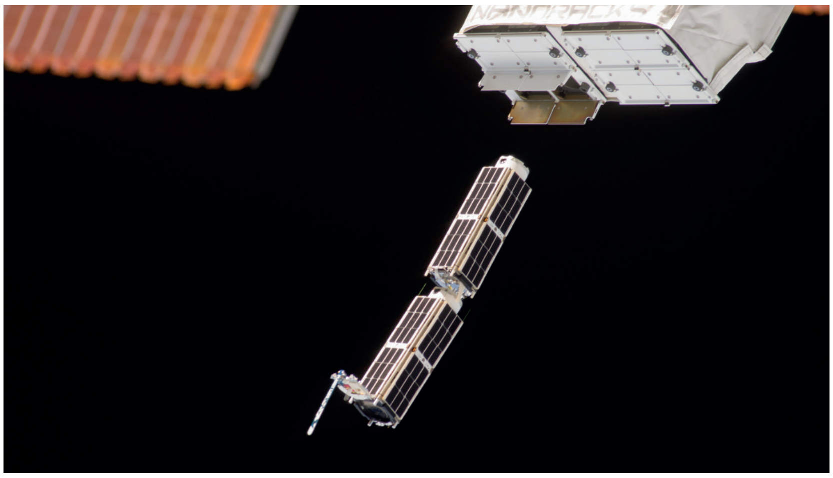
Planet Labs is perhaps not dissimilar to Canonical in trying to create a commercial business with an altruistic side, and Will started to tell us how the ideas behind Planet Labs began to take shape. He told us that while he'd been working at NASA, they'd been experimenting with what they now call 'PhoneSats'. These were literally smartphones that they were putting in orbit to see if they could work. And they worked just fine.

## Money makes the satellites go round

"I worked on a couple of what NASA considered small satellites with 10–200 million dollars of cost, roughly." Will told us. "They're not necessarily physically small, but they're small in cost because normal satellites cost half a billion or billions of dollars."

With the PhoneSat, the aim was to "break down psychological barriers. It's not as hard all that. Now there's a lot of systems complexity into putting satellites together and working with all of the ground stations and stuff, so it's not trivial. But nevertheless, it doesn't need to be a billion dollars."

Like computers in the 1950s and 1960s, satellites are traditionally huge and heavy. A typical payload is


The International Space Station using its nanoracks deployer to launch Planet Labs satellites, shown here from 11 February 2014 (Photo: NASA).

6,000kg, and that kind of weight needs the entire fairing of a rocket to make it into orbit. Not only is that expensive, it adds many different layers of complexity and organisation, which is why you find countries rather than companies sponsoring and managing their deployment. Part of the solution for Planet Labs is to borrow from the philosophy of agile development, – that's releasing early and releasing often, taking advantage of the latest consumer technology.

So why hasn't this methodology been adopted before? "Because technology wasn't ready and because if was a different philosophical approach to satellites and a higher risk one in a way," Marshall says. "We hadn't guaranteed that the technology was going to work. It was a radically different approach. We started Planet Labs because we realised that we wanted to explore the humanitarian and commercial uses of taking imagery of the earth's surface."

The satellites being built at Planet Labs are tiny by comparison (only 10 x 10 x 30cm, and weighing a mere 4kg), like ants beneath the feet of elephants, which is perhaps why they could build them from the garage. The main section is an elongated rectangle containing a small telescope pointing down to a camera at the back. What's even better is that it's stuffed full of the latest technology, and amazingly, an x86 PC running Ubuntu. Marshall says that they chose Linux and open source because Planet Labs wanted to be able to rapidly reconfigure its OS to do the things it needed to do. We're left guessing as to whether it's a long-term release, but the lifespan of one of these satellites is only 1–2 years, depending on their altitude, so it might not even matter.

But what's just as impressive is that alongside its x86 Linux PC, Planet Labs is also using copious amounts of open source both for its onboard processing and for its image processing closer to home. "Most of the image processing stack is on the ground," Will told us, "but there is some processing on board. Most of the image processing stack on the ground uses open source software built in libraries like GRASS and GDAL and things like this – open source libraries that our employees are helping to develop."

**"The satellites are stuffed full of the latest technology, including an x86 PC running Ubuntu."**

So does that mean that any of Planet Labs' changes are making their way back upstream? "Absolutely. That's our goal."

## Get a job building satellites Fancy studying something that can take you to space?

Until relatively recently, none of us might have thought about the viability of a career building satellites. But with the advent of companies like Planet Labs building cool things from a garage and hanging out in the Californian sunshine, it looks like we're on the cusp of a revolution in the space industry revolution. And if that sounds like your thing, the bad news is that it's probably too late. You need to make sure you study the best possible subjects from the very start, which is why we asked Will Marshall exactly what it takes to build a career launching satellites. And yes, Planet Labs is hiring!

**LV How did you become the CEO of PlanetLabs?**
**Will Marshall:** I've been interested in space since I was yea high. I built a telescope when I was a kid, and got interested in astronomy. Then I found myself looking for ways in which I could use science to help people, basically. I did a degree in astrophysics and a PhD in quantum physics and then went to work at NASA building small satellites for planetary science purposes, primarily.

**LV So it's probably too late to get into the field by the time you're 16 or 17!**
**MW:** It depends on which education system you're in! Study computer science. Study aerospace engineering. Study physics or astrophysics. These are the kinds of things that would be useful in this area. I would alway encourage people to start on the most abstract mathematical end, because you can always go more engineering. So if you're not sure whether to go into physics or into engineering, or maths or physics, stick towards the left of that axis of abstraction because if you study mathematics you can always go to physics, if you study physics you can always do more engineering, but the reverse is less simple. If you want to get into space, you could start with aerospace engineering but you could also start with physics and maths.

"Firstly, I would say study hardcore science or engineering." Will Marshall and Graham Morrison discuss career prospects.

"We want to push out whatever useful things that we do to process imagery in a massive way… [we have] a compositor that takes deep stacks of imagery, looks for ones with cloud, rejects those, takes some of the images and pulls them into something that is a coherent composite image that is the highest quality from that stack. So that's the kind of thing that will be useful for lots of other people, that gets stuff out there and enable other people to work on it too."

## Agile aerospace

This software is the Pixel Lapse Compositor, and its lead developer, Frank Warmerdam, is already maintaining the project on GitHub (**https://github.com/planetlabs/plcompositor**). Frank developed and is still one of the lead maintainers of the aforementioned GDAL – the Geospatial Data Abstraction Library, a major project used by many different projects to read and write to lots of different kinds of raster geospatial data formats typically used in tracking data. If you've ever tracked yourself with a GPS and put the file on your Linux box, you'll have come across one of the formats and realised that despite them all being called 'GIS', it's never simple to make sense of the data that these files contain. Other open source projects used by the team include *PostGIS*, *NGINX* and *OpenVC*, and another team member, Jesse Andrews, is one of the lead developers of OpenStack.

This is just the beginning of the deployment and testing phase, and the crux of the project's success, at least from our perspective, depends on how the team licence their data and how freely projects will be able to access that data.

"We will enable anyone to access the data via the developer API, says Marshall. "We'll talk more about the product when we get ready to launch it, but we intend it to be in that spirit." That's great news, and it means that hackers will be able to get their hands on some dramatically up-to-date earth imagery.

On 19 August 19 2014, Planet Labs licensed its early imagery under Creative Commons Attribution Share Alike 4.0, and while this only includes the images that can currently be found within the company's hosted gallery, it would be wonderful if a licence like this could eventually used for the image data obtained through the eventual API. It's obvious that there are hundreds of applications for this data and even with the inclusion of commercial interest, there will always far more potential with an open interface. The thought of an open source project being able to run its own algorithms against the data set – whether it's someone tracking queueing traffic on the Suez canal or the water levels in reservoirs, or anything else that the collective imagination can come up with, is a wonderful one.

Unlike the predatory naming convention used by many other satellites, these are called 'Doves' (Photo: NASA).

Planet Labs even has an artist in residence who designs the artwork for both the satellites and for the outside of the ground stations.

# A HISTORY OF LINUX GAMING

**Liam Dawe** peeks into the belly of an unstoppable beast.

In the first ever issue of Linux Voice we briefly touched down on the colourful history of Linux gaming. Now we're here again to give you a better picture of how we went from being an operating system that was mostly ignored by every major developer possible, to having major publishers on board. Let that just sink in for a moment, as two years ago we didn't have anything looking as bright as it is now. That's an insanely short amount of time for such a big turnaround.

## The dark ages

We start our look in the early 90s, before most popular Linux distro even existed.

Back in the 90s, people would most likely laugh at you for telling them you used Linux on the desktop. It was around this time that Id Software was creating the game *Doom*, which actually helped push Windows as a gaming platform. Ironically it was Id that threw us our first bone. A man named Dave Taylor ported *Doom* to Linux the year after the original release, and he only did it because he loved Linux.

In the **README.Linux** file Dave gave his reasons for the port:

"I did this 'cause Linux gives me a woody. It doesn't generate revenue. Please don't call or write us with bug reports. They cost us money, and I get sorta ragged on for wasting my time on Unix ports anyway."

*Doom* wasn't quite the polished 3D FPS that we have now, but it blew away most other games that came before it, and was fantastic for Linux. *Doom*, then, has the honour of being the origin myth in our history of gaming on Linux.

There were unsupported executables of a number of later Id games , such as *Quake 4* and *Doom 3*, which you could download after purchasing the Windows version to run them natively on Linux.

One of the first big name games to ever grace our platform, *Doom* has left quite a legacy.

Sadly though, Id Software no longer supports Linux with unofficial binaries to run their games like it did in the past, and comments like this from John Carmack (formerly of Id Software) don't help:

"Improving *Wine* for Linux gaming seems like a better plan than lobbying individual game developers for native ports. Why the hate?"

Luckily Timothée Besset – the chap responsible for a number of those unsupported Id Software Linux ports – was more positive towards Linux-native affairs: "I don't think running games on *Wine* is going to get much easier… it's pretty much as good as it's going to be. It's such a complex piece of engineering that it'll always remain a rather frustrating barrier. Native is where it's at."

After the release of *Doom* we didn't exactly have much else going for us, but luckily Open Source picked up some of the slack, as it always does on Linux when we're missing something.

Some of our older readers may remember *Freeciv*, which is a clone of the original *Civilization* turn-based strategy game, but it wasn't until near the end of the 90s that *Freeciv* even gained computer-controlled opposition players, so you were stuck finding people to play with. It's a great game though, and if you haven't checked it out yet you really should, as it's still actively developed today. You can even play *Freeciv* in your browser: **http://play.freeciv.org**.

Neither Gnome nor KDE came with a decent set of games until the end of the 90s either, so things were looking a bit drab.

We had none of the really excellent open source games that we have now, as even well known time-wasting games like *Frozen Bubble* and the wonderfully crafted *Battle for Wesnoth* didn't come out until the 2000s. It was a dark time, but it was early days. The day would come – we just didn;t know it yet.

## Wine

*Wine* and *CodeWeavers CrossOver* seemed like they may have been the only hope for Linux gamers. *Wine* enables us to run Windows games on Linux without needing any kind of Windows install, and that's pretty enticing.

The problem with *Wine* is that it comes with a whole host of drawbacks such as performance loss, and bugs that we may never be able to be solve due to *Wine* being a layer on top of the game itself – there are many different technical aspects to *Wine* due to its replicating Windows.

*Wine* can be a bit hit and miss, as for some games it may give you an almost native feel, but with others it may flat-out not work.

Some actually feel that *Wine* disincentivises developers from bringing out native ports, and seeing things like "We've been told you can try *Wine*" can be very disheartening to Linux gamers.

## The light at the end of the tunnel

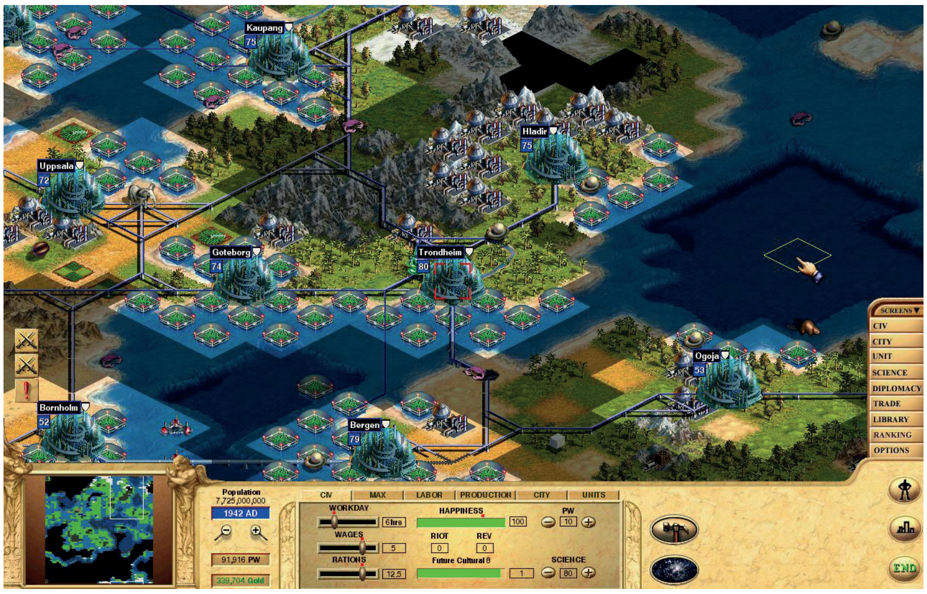
### At the end of the 90's there was a spark – Loki Software.

Loki Software came up with what seemed like a great idea – it approached major game developers and offered to port their games to Linux.

Loki was responsible for giving Linux *Civilization: Call to Power*, getting a game like that at the time was almost unheard of for Linux. A Loki Software employee named Sam Lantinga (who now works for Valve) created the extremely useful library Simple Direct Media Layer (SDL), which is used by many games and companies today. Even *Freeciv*, mentioned earlier, has a version that uses SDL.

Luckily for us a man named Ryan Gordon, who worked for Loki Software, carried on porting a number of games and quickly rose to fame as a name in Linux gaming (he still works on ports today). Ryan has been responsible for some high-profile games as well as a number of indie games such as *Serious Sam*, *Psychonauts*, *Aquaria*, *Goat Simulator* (that game is utterly hilarious) and many more.

A little sore spot for Linux gamers is *Unreal Tournament 3*, which Ryan ported to Linux, but it never saw the light of day for unknown reasons (most likely middleware licensing issues). Luckily that hasn't stopped Ryan from working with the Epic Games community with Linux-related *Unreal Engine* tasks. Then along came Linux Game Publishing (LGP) near the end of 2001 – another porting house that rose from the ashes of Loki Software. Linux Game Publishing  was originally run by Michael Simms, and was based on the same idea as Loki Software. LGP seemed to be a good deal for Linux gamers, as not only would you be supporting Linux, but you'd get your games in a shiny box.



*Civilisation: Call to Power* was one of the early Linux ports courtesy of Loki Software.

LGP based its business on the same model, as it spoke to developers to port their games to Linux, and offer high-priced boxed versions of games that had already hit the bargain bin for other platforms. This was a common complaint among gamers – having to pay $40 for a game that was $5 on Windows.

Sadly LGP suffered a different kind of problem, as its CEO at the time stepped down due to what seemed like a burnout. After no new ports were done for some time he handed the reins to a new CEO who made a small push into digital stores before the company finally fell silent.

If companies like LGP and Loki Software came about nowadays they would probably have a lot more success. The porting house gap has seemingly been filled by Aspyr Media and Feral Interactive, which are currently porting some pretty high-profile games to Linux.

## The indie revolution

2008 and beyond saw an ever so 'umble development.

After the demise of Loki & LGP, Linux gaming seemed hit a bit of a sore spot, but from 2008 onwards everything changed, and the change was fast. In 2008 a real time strategy title named *0 A.D.* popped up on the Linux radar. The developers noted on their forum that they had planned to release the game as open source and it was a matter of months away.

This turned out to be true, as in 2009 it did release the source code, and two months later they had released the first Alpha version of the now open source RTS game. This is another big project using SDL (which was created at Loki Software, remember), so six years after Loki closed its doors its software was still in use.

This was huge news, as *0 A.D.* was the first open source and completely free RTS game on Linux that was being built to a commercial standard, and now at Alpha 16 it looks incredible.

Around this time a number of slightly higher-profile indie games started to release Linux versions of their games. We had the highly anticipated *Amnesia: The Dark Descent*, which was released in 2010 and promised Linux gamers their first proper experience of a horror game in first person. Then in May of 2010 came the Humble Bundle, which promised DRM-free, pay-what-you-want games. The beauty of the Humble Bundle was the fact that all games in it had to be cross-platform. The first bundle was such a runaway success that it has been repeated over and over and now does much more than just game bundles – although much to our dismay not all bundles feature Linux games now, as they have started to add DRM-filled bundles.

Thanks to the second Humble Bundle we were graced with the award-winning puzzler *Braid*.

Linux gamers came out in force for the Humble Bundle, and have repeatedly smashed the average buying price compared with Windows and Mac gamers, proving that Linux gamers do in fact pay for games. If we look at the history of the Humble Bundles we can see just how far Linux gamers go for games now, thanks to the Humble Bundle Visualisations website (**http://cheesetalks.twolofbees.com/humble**) created by Josh Bush.

When you look into the above you can see (at the time of writing) the overall averages for purchases of bundles:

- **Average payment** $5.83
- **Linux average** $9.36
- **Mac OS average** $7.46
- **Windows average** $5.56

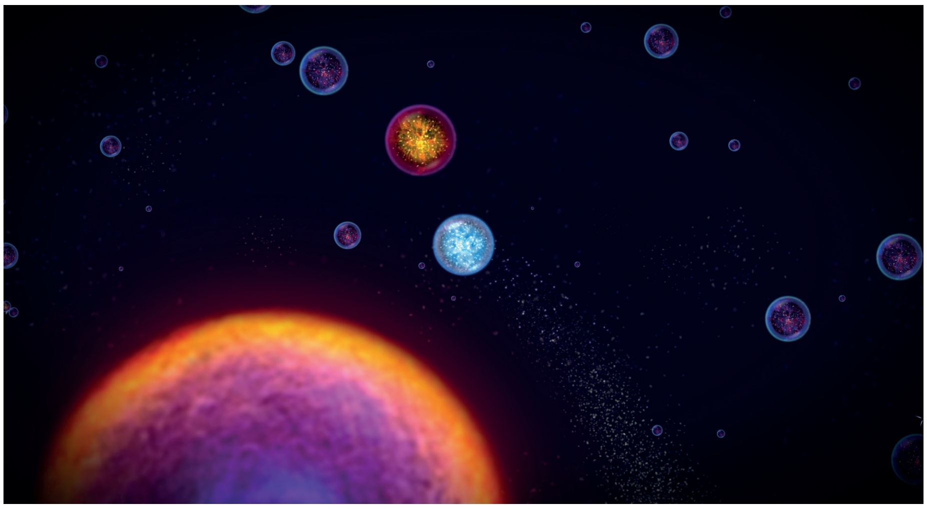
As you can see, Windows gamers pay the least, but it's not actually that surprising when you think that many of the games that Humble Bundle put out have probably been on Windows for some time, so their value is less.

The obstacle that Linux gaming faced at this moment in time was the lack of a decent web store. Most big stores ignored Linux until Desura came along. Desura is a small (in comparison to Steam) online store that has its own Steam-like client that keeps games up to date for you.

Desura decided not only to support Linux games on its store, but also to ported its Desura client to the Linux desktop. The client, though buggy, was functional enough, and it was even open sourced down the line to appease the Linux community in the hope of gaining more community developers.

The big buzz around Desura was short-lived however, as it hasn't pushed out an officially updated Linux client in quite a long time. This is partly due to the newer owners of Desura coming in and forcing a restrictive contributor agreement for anyone who wanted to write code for the open source client.

By this time we'd proved that there was no technical reason that games couldn't exist on Linux, and more importantly, that there was plenty of money to made. The scene was set for Valve!


Osmos may not be the biggest or most innovative game around, but it came at the start of the Linux indie craze, and had an awesomely mellow soundtrack.

## The rise of Steam for Linux

### If people are making money out of us, we're doing something right.

When the news came that Valve was bringing its Steam games client to Linux, it caused some controversy. Some gamers feeling that the Steam client is merely a storefront that allows the use of DRM; but considering you need to have the store installed to download and play the games, some see that as DRM. It's a fair argument, and one best not to get involved in as it can get a little heated.

Even Richard Stallman himself had a good point to make about Steam coming to Linux with this comment:

"If you're going to use these games, you're better off using them on GNU/Linux rather than on Microsoft Windows. At least you avoid the harm to your freedom that Windows would do."

Even for users who shun services like Steam it's hard to deny all the good that it does to boost Linux's popularity. It's not just about increasing popularity, but Steam announcing lots of Linux-related projects increases the overall awareness of Linux too. Steam coming to Linux by itself wouldn't have been as big a deal as it was without Valve pushing its own games onto Linux as well, and even stating in blog posts how well they ran with OpenGL on Linux.

When talking about their work with *Left 4 Dead 2* and talking with driver developers directly the Valve devs actually talked up how good OpenGL and Linux are:

I know what you're thinking: "Ahhh, sports!", but *Football Manager* deserves all the acclaim it gets. It's a massively popular title, and one of the first from SEGA to support Linux.

"After this work, *Left 4 Dead 2* is running at 315 FPS on Linux. That the Linux version runs faster than the Windows version (270.6) seems a little counter-intuitive, given the greater amount of time we have spent on the Windows version. However, it does speak to the underlying efficiency of the kernel and OpenGL."

One other store that's quite the favourite among gamers is **GOG.com**, formerly Good Old Games. This is thanks to its stance against DRM, and offering good policies like refunds if you cannot get a game working (which is quite the opposite to Steam), but it seemed originally quite against bringing its store to Linux.

We will also get support from GOG's soon-to-be-released desktop client named GOG Galaxy, which can be seen as being Steam-like for its ability to auto-update your games for you along with adding in extras like online matchmaking. It will be useful, but GOG being as gracefully as ever has noted that the client will be 100% optional and that its standalone downloads will exist alongside the new client.

## The future

### The future's bright. The future's steamy.

In the past two years the Linux gaming scene has exploded. We've gone from people outright laughing at our platform to regularly seeing Linux users make statement like "I have too many games, what do I play?!".

We have an insane number of games to look forward to this year alone, and who knows what 2015 and beyond will bring us? We still have SteamOS, which is Valve's custom Linux distribution aimed at gaming, and primarily meant to help the company push its Steam Machine consoles.

Then we'll get to see the famed Steam Controller, which will work natively on Linux as any other USB device would. This controller ditches the traditional controller sticks in favour of touchpads, and it looks set to change the way we think of gamepads. As amazing as Linux gaming is right now it's pretty safe to say we would have never gotten as far as we have without Valve and Steam. To think one company has changed things so dramatically for us in such a short space of time is crazy, but that's what happens when a major player in the gaming space moves into our territory.

Who knows – maybe even Electronic Arts and its Origin client will announce Linux support next. The future for Linux is stronger than it has ever been, and we are witnessing a change in the PC gaming sphere with front-row tickets to the show.

Linux gaming isn't just a sideshow: it's where the innovation is happening.

FAQ

# DIASPORA*

The one social media system to bring them all, and in the openness, bind them.

BEN EVERARD

**Q OK, let's start simple. What is Diaspora*?**

**A** It's a source social network. From a user's perspective, it's quite similar to Facebook or Google+ in that you add people you want to be in contact with, then it brings all their updates into a stream for you to view. You can assign people to different groups depending on how you know them and tailor with whom you share information. You can follow hashtags, and posts that mention these hashtags get added to your activity stream.

**Q I've checked the bottom of the page, and I can't see any footnotes that reference that asterisk. Why do you keep using it?**

**A** The software's called Diaspora*. The asterisk is part of the name, not a reference to a footnote. In fact, it's silly, so let's get rid of it.

**Q Oh, OK. So Diaspora is a kind of a mashup of Facebook, Google+ and Twitter. Why on earth do I want another social network sucking up my free time?**

**A** Well, I wouldn't quite call it a mashup, but it certainly appears to have taken some inspiration from those other social networks, and some of those other social networks may have borrowed ideas from Diaspora. For example, Google+'s circles seem remarkably similar to Diaspora's aspects (which appeared first).

The real advantage of Diaspora isn't the software features though, it's the philosophy behind it. Diaspora is open source and federated, so it's not under the control of any one organisation.

"The real advantage of Diaspora isn't the software features, it's the philosophy."

**Q Federation? What does that mean, and how does it benefit a social network?**

**A** Federation means that the network is open and anyone can create a new server. For example, anyone can set up a new email server. All you need is a computer connected to the internet with an externally route-able IP address. This means that no company can monopolise the medium, and no one can be banned. You can use whichever email provider best suits your needs, or run your own server if you want to keep control.

The same is true of Diaspora. The network is decentralised, and consists of a number of servers (known as pods) that connect to each other. Each pod can handle many users (depending on the hardware hosting it), so you don't have to host your own; you can join a pre-existing pod.

The pods are independently operated, and anyone can set one up and connect it to the Diaspora network. Pods can be private to a particular group, or open and allow anyone to join.

**Q So, since it's open source and federated, does that mean Diaspora is more secure than commercial social networks?**

**A** That depends on how you set it up. Whenever you upload something to a website – any website – you're giving up control of that data. Whether it's a 140-character tweet that you want to share with the world, or a picture that you only want your Facebook friends to see.

All social networks specify with whom they will share the data, but it's up to them to make sure they follow these rules. There's no technical reason to stop Facebook sharing all your private data with the world. This is also the case with Diaspora. When you upload data, it's stored on the pod, and you have to trust the pod's admins to respect the terms under which you uploaded it. What's more, if you share it with a member on another pod, the data will be transmitted to that pod, so you'll have to trust that pod's administrators as well.

The thing that makes Diaspora different is that you choose what pod you use. If you're worried about security, you can set up your own pod, and invite people to join you on it. Then, anything that you only share with people on your pod will only be shared with that group. There won't be any other admins that could poke around in your data.

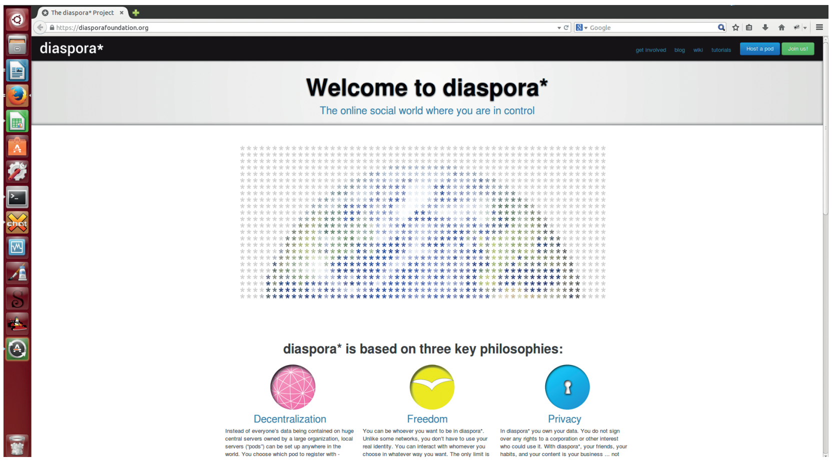
**Q Are you saying that unless I run my own pod, Diaspora is not necessarily any more secure than the commercial networks?**

**A** To a certain extent, the answer to that question depends on from whom you wish to keep your data secure. If you wish to keep it more secure from rogue sysadmins or hackers, then there's a risk with any social network. On the other hand, if you want to keep your data away from advertisers who are trying to build a profile of you, then there's a slight risk that a malicious Diaspora pod may do this, but we know that Google and Facebook do this – after all, they're both advertising companies that run social networks to get more people to look at their adverts.

What's more, since Diaspora is run by the community, the security decisions taken are the ones in the best interests of the community, not what's in the best interests of the advertising company running the website. This means that we don't expect to see sudden changes to privacy settings that lead to once-private data being shared with the world. You can also download all of the information Diaspora holds about you, or delete it at any time.

**Q If it's all open source, who's running it?**

**A** The project was started by a group of four students (Dan Grippi, Maxwell Salzberg, Raphael Sofaer and Ilya Zhitomirskiy) at Courant Institute of Mathematical Sciences in New York, and they turned to Kickstarter to crowdfund the development. On 24 April 2010, they launched a campaign aiming to raise $10,000, they received just over 20 times that much, making it the most successful Kickstarter project at that point. The first pod (an invitation-only alpha) went live on 23 November 2010. However, before the software reached a stable state, tragedy struck and Ilya Zhitomirskiy killed himself.


The diaspora* Project
https://diasporafoundation.org
diaspora*
Welcome to diaspora*
The online social world where you are in control
diaspora* is based on three key philosophies:
Decentralization
Freedom
Privacy

You can find out more about the project at **https://diasporafoundation.org**, but this isn't a pod, so to join the network, you'll need to head to **http://podupti.me** to find one.

Two years after receiving the funding – with the software still in beta – the main developers shifted their focus to a new project, and announced that they'd let the community take ownership of the project. The project is now run under the umbrella of the Free Software Support Network.

Of course, this is just the development of the software. Because of the federation, the actual hosting of the pods is done by other organisations and individuals around the world.

**Q Wow, it sounds like there's no downside. Should I close down all my other social media accounts and switch to Diaspora?**

**A** There is a sort of chicken-and-egg problem with new social networks. No one wants to join until there are enough of their friends on to make it worth while. At the moment, the Diaspora community is tiny in comparison to the big social networks, and so it's unlikely that you're going to be able to connect with all your old school friends.

There is a slight mitigation for this in the way Diaspora can link to the other networks. That means that you can push your posts from Diaspora to Facebook, Twitter, Tumblr and Wordpress. However, this doesn't solve the problem, as you still have to log into these services to interact with them.

The federated nature of Dispora also means that there is no one in overall control of the network. For the most part, this is a good thing. However, there are cases when this means that there is no one there to enforce good policies. For example, IS (the organisation formerly known as ISIS) has been banned from Facebook and Twitter, and has now moved to Diaspora. Since no one is in overall charge of the network, it becomes the responsibility of individual podmins to remove the accounts of people sending messages of hate or using the network to organise malicious activity. The core team have worked with podmins to remove inappropriate accounts, and the situation is ongoing at the time of writing.

**Q OK, I'm convinced. Where do I sign up?**

**A** There's a list of existing pods at **http://podupti.me**. You can only join ones that have open signups. It's also a good idea to look for a pod with high uptime, and a recent software version. Hashtags aren't federated, so you'll only receive the public posts for the pod that you're a member of (you can get posts from friends on any pod). So, it's a good idea to join a pod with a large community in an area of interest to you. This could be people from a particular locality, or who follow a particular technology.

Alternatively, you could set up your own pod. To do this, you'll need a computer with an IP address that's routeable from the internet, and ideally you should have an SSL certificate (not self-signed). Once this is set up, you can follow the install guide at **https://wiki.diasporafoundation.org/Installation**. LV

# THE MAN IN THE RED HAT: FREEDOM BEYOND FOSS

Jan Wildeboer, Red Hat's EMEA open source evangelist, has some big ideas about freedom in our everyday lives...

You probably know Red Hat as the company that makes a Linux distribution targeted at enterprises, and also backs the Fedora project. But Red Hat has its fingers in many pies, contributing code back to a large number of free software projects, and many of its developers and managers are passionate advocates of FOSS and freedom.

Back at the FOSDEM conference earlier this year, we bumped into Jan Wildeboer when doing the rounds, and he had a lot to say about the importance of freedom and identity in the digital world. Jan is responsible for open source evangelism at Red Hat, visiting large enterprises and espousing the benefits that free software can bring. We met up with him again for a proper chat at Red Hat's offices in Munich, and learned more about his intriguing ideas. This interview isn't so much about Red Hat and Linux; we also talked about CentOS and its relationship to the company, so we'll have more on that it in a future issue of Linux Voice.

**LV One thing we wanted to start with, even though it's not Linuxy per se: when we met at FOSDEM you were talking about how you microwaved your passport. Could you tell that story again because that was awesome?**

**Jan Wildeboer:** Well it's quite simple. The new European passport has this embedded RFID chip, which hands out data on two levels. There's a sort of public access that everyone can read, which gives you some unique identifier but no real information. And there's a

"Why can't we take what we've learned from machines and transport that to human relationships?"

"When you buy stuff in high-price fashion stores they have RFID chips and they track you when you walk through the shop."

second level, where you have to authenticate against the RFID chip and then it will spit out more information, going up to your fingerprints, biological data, picture and that kind of stuff.

I don't like this first level of access on the passport, because it has been used in shopping malls etc to collect tracking data. You can't identify the people behind it, but you can at least see the flow of people – who is standing where for how long.

**LV In a lot of airports, they ask you to show your boarding card when you buy something. I tried to buy a packet of chewing gum in Manchester airport and they asked to see my passport, so I said "I just won't have the chewing gum then". I don't want to be tracked all the time. I guess it's for marketing purposes.**

**JW:** No, it's tax reasons mainly. If they sell it to you at the airport and you're flying out of your tax zone.

**LV Oh! I didn't realise that. I was being so cynical.**

**JW:** I always have online check-ins to be careful, but they're tracking me in the airport anyway with the passport. So I have this risk of data leakage, which I don't like. So I went to the Dutch Embassy to get a new passport and asked this guy what happens if the chip isn't working anymore? Is the passport is still valid? He said yes, because there is the document of the passport and the chip stuff is an additional feature. If the chip doesn't work anymore, that doesn't render the passport invalid.

Now, the way an RFID chip works is quite simple: it's an antenna that collects data. The chip is completely without any power source. It gets powered from the electromagnetic field that is used to read it; that's a coil, and this coil collects energy and sends the data out etc.

Once you have a coil, and you put too much power on it, it's overloaded. So what you do is you put it in the microwave for two or three seconds, because a microwave is very concentrated electromagnetic fields, and then the chip goes 'poof' immediately. So the chip doesn't work anymore but the passport is still valid. And if you do it in the right timing, you won't even see it – there will be no black spot or something like that.

**LV There was talk that people have them to scan things next to you. So say you're on the London Underground or something with your passport in your pocket and someone wants this information and they have a reader there. I think the American version of these chips actually have lead-lined covers.**

**JW:** I don't know how the American passports work, and honestly, I don't want to know. But with this whole RFID stuff here, it's quite fascinating. When you buy stuff in stores like high-price fashion stores, they also have RFID chips and they use that to track you when you walk through the shop.

**LV What – when you pick something up in the shop?**

**JW:** Yeah, and then you are carrying it around and they have readers and can quite exactly divine where the people are going, so they use this stuff, which is marketing. So they see, for example, that somebody takes a dress and walks over to the jeans department, and that happens quite often, so they put them closer together. All this is cool stuff, don't get me wrong. But the moment I buy the dress, each of the RFID chips has a unique identifier, so the moment I buy the dress, then it becomes something that identifies me.

But when I go into the shop the next time and I haven't removed it (it's often on a sticker they put on it but sometimes it's also woven into the object itself)... there is a very simple rule within the EU about data protection and privacy etc. You can ask them to remove it and effectively they should do it voluntarily anyway. You can ask them to destroy the chip the moment the transaction is closed. I mean, I've paid for this stuff and it's mine, now we have an ownership transfer, and at that moment I can destroy the chip myself of course, but I can also ask them to remove the chip because the official purpose of the chip is now gone, I have paid for it.

**LV Are they under any obligation to tell you about the chip?**

**JW:** No. Well, yeah, sortish, but who really cares, it's a symbol. It's one of these standard questions that are just asked at the checkout, "Can you please destroy whatever chips are in there?" Typically, the reply you will get is "Uh? What chip? ". They also have these special stickers where it's already prepared and written through because the coil has an antenna.

**LV I think I remember stickers like that on CDs and DVDs back in the day. It does get invasive.**

**JW:** But it's also cool just to – this is one of my hobbies, which I call social

Jan travels a lot in the EMEA region, convincing businesses that open source and standards are vitally important.

hacking or social engineering, to collect these, take them out of the stuff and put them in pockets to confuse people.

**Yeah, give spurious data. So when Facebook asks for your SMS contacts and you can get these hacks for Android that just feed back random data.**

**JW:** Yes, there are a lot of fun things you can do with your online identity. One of my favourites is to set up the cookie exchange network. So with just a little bit of shell scripting and stuff, we can swap cookies. So at a given moment in time, it starts swapping cookies, so I get your cookies and you get my cookies, and it travels around, which totally kills my online profile because all of a sudden the advertisement would look totally weird. I mean, I would get an insight into what you might be interested in. And if you do that a bit randomly, if this were between a network of friends, then you could really destroy this data.

**I've been getting Indonesian car adverts on YouTube before videos, because I realise that adverts help to fund a lot of websites in general so I don't block them. But I don't like the obsessive tracking, so I turn on 'Do not track me' and stuff like that. I'm not in the market for a car, I don't speak Indonesian, and these adverts are coming up so I think yes, I've confused Google enough, it simply has no idea about me.**

**JW:** You know it's being tracked anyway, every little thing we do. Using these little methods, it's not so much I don't care about being tracked – I can't do anything to avoid it anyway. What I do care about is, however, pissing in their pool of data to make the quality bad. Because the moment the quality of my profile gets bad, I get filtered away immediately because I'm not relevant.

I think that's a fair way to take revenge, but it's also a way of saying that I understand what's happening in the background, and because I understand and I'm allowed to tinker with it, you cannot stop me from tinkering with stuff. Deleting cookies is what some people do all the time. That's not enough. Nowadays, you have lots of methods of tracking stuff. It's cookies, it's browser identifiers, it's persistent flash cookies, and all that kind of shit. So I don't trust anything in that regard.

**Tell me about the other thing you were talking about – United Transnational Republics ...**

**JW:** It's a political idea of a better way to identity yourself. If you think about identity, really the fundamental concept of identity, of me being able to identify myself to you, so you know who I am. Identity is, in daily practice, quite easy to us. You don't need the Dutch state to understand that I'm Jan Wildeboer, even though I'm a Dutch citizen and I have a Dutch passport. But this level of identity is very centralised and there are lots of authorities involved. As a Dutch citizen, I have a Dutch passport, which is given to me by the Dutch government, effectively by the Dutch state, but there's no obligation for them to give me a passport. They can take it away at any time. They can renounce my citizenship. Ask Edward Snowden about that.

**But isn't it a UN human right that everyone has the right to citizenship?**

**JW:** Interestingly not. That's exactly the point of the whole concept behind the Transnational Republics: we want to have that as a fundamental human right. The right to own your identity, and the right to define your own identity. The reason we came to that idea was quite simple. The contract of Geneva about the fundamental human rights say very clearly that the fundamental human rights are granted to every citizen of a

Fact: the original creator of Red Hat Linux, Marc Ewing, used to wear a red hat at university.

member state of the United Nations. Which means, the other way around, if you're not a citizen of a member state of the United Nations, you do not have access to fundamental human rights.

**But pretty much every country is in the UN now isn't it, apart from North Korea...**
**JW:** But some are not citizens. Refugees, who officially give up their citizenship because they want to escape the country. In the UK it's, like, when they have British citizens who are from, let's say, originally from Pakistan and they go back to Pakistan to whatever, so-called terror camps and the US finds a drone to deliver an explosive package, then there is a problem because when the US drones kill, technically, a UK citizen, that would be an act of war. So what you have to do, to be able to kill them, is to take their UK citizenship away. And that's exactly what's happening. There's a whole process behind it where the US, or the NSA or CIA or whoever, calls the UK authorities and says "So we're going to send a drone to this and that place, and that or that people, and there might be UK citizens, can you check?". And then within hours they take their citizenships away, and then there is no violation of human rights.

**"We think the United Nations sucks at democracy, and we can do better."**

Red Hat's EMEA headquarters is located in Grassbrunn, on the outskirts of Munich.

**So they're just killing someone that barely exists in their eyes.**
**JW:** Very over simplified. The whole legal stuff is extremely complex. And this sort of shows you how identity that you don't own yourself is a privilege and not a right. So we think it's better to have a right to identity. And the reason for that is because we want to build a global democracy. We think the United Nations sucks at democracy, and we can do better, so we created this idea of the United Transnational Republics to give a better system for global democracy. You know, don't take it all too seriously, it's just a way of thinking. And to have democratic votes, you need to be sure that nobody double votes and that everybody who votes is allowed to vote, and that's why you need identity.

Now, with machines we have this level of assurance and de-centralised checks and balances. You know, with TCP/IP, we have a relatively anarchistic self-organising network that is de-centralised at its core. With TLS and SSL we have certificate authorities etc, so that with these certificates that offer validity, we have quite a high level of assurance when machines talk to each other that they really are the intended machines. That's why you need man-in-the middle attacks to compromise a system, but then with certificate pinning you can make that secure again, so these problems are solved now. I think, for philosophical reasons, I find it very interesting that machines enjoy more security than human beings. So why can't we take what we've learnt from machines and transport that into human relationships? So instead of TCP/IP, why can't we not have TCP/ID? That's a sort of way to look at it. And that's at the core of the Transnational Republics and transnational identity. Decentralised, self-owned and self-regulated system of identity where everybody technically becomes his own identity. That's the theory, that's a bold plan. A crazy ridiculous plan, but it's interesting to think about identity in that way.

Once you start looking at this decentralised way, then you're looking at open source communities and how they organise themselves. Who's allowed to commit to open source repositories. Again, all about identity. It's all about trust, relationships and somehow making those relationships work on a global level without being able to physically interact with each other. It's an interesting thought, especially now that we have all of these privacy discussions, and all of this security and snooping... At the core of any kind of democracy lies identity.

Write once,
deploy everywhere.

ubuntu

**Back issues are now available at**
**http://shop.linuxvoice.com/products/single-issues**

# LINUX VOICE REVIEWS

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

**Andrew Gregory**
**Keeps his special photos on Ubuntu One, where they'll be safe forever...**

This month I've treated myself to a new guitar amplifier. It uses valves instead of microchips – the same sort of valves that were used in the first computers of the 1940s. I read a magazine on paper, first mashed out of dead trees thousands of years ago. And I used a distro installer that used a text-mode front-end, which seem to have been around forever but really hit their peak in the 80s.

It may seem like the march of technology has left me behind, but it really hasn't. Technology lives on after its original uses have become obsolete. In the 80s, a text UI was the easiest way available for the user to interact with the machine. That's not true any more, but it does have certain advantages – low system requirements being top of the list. Paper is still useful for its low glare and unlimited battery life, factors that weren't relevant in Han dynasty China.

## Babies and bathwater

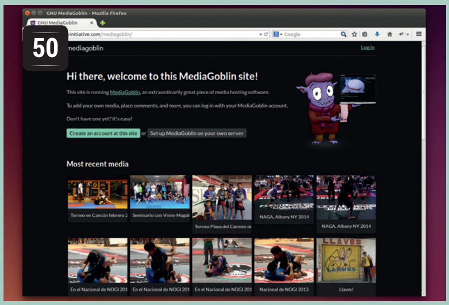
Even if you run a constantly updating Arch system, there are components of distro that are decades old. Mike mentions the **tar** command elsewhere in the magazine, and there are loads of others that date back from the 1970s, 80s and 90s. We should embrace the new, but not reject the old.
**andrew@linuxvoice.com**

## On test this issue...

48

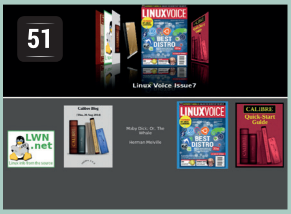
### Wacom Intuos Pro

**Graham Morrison** always wanted to be an artist, expressing himself through the medium of colour and shape. Now he is!


50

### Mediagoblin 0.7

This Gnu project aims to become a free alternative to YouTube. **Ben 'cat videos' Everard** isn't convinced.


51

### Calibre 2.0

One day **Mike Saunders** is going to write the Great Cumbrian Novel. He'll probably use this brilliant piece of editing software to help him organise it.

52

### Energine sockets

**Ben Everard** uses 240V to control his projects, so he needs something to sit between the Raspberry Pi and the mains power supply. Like this!

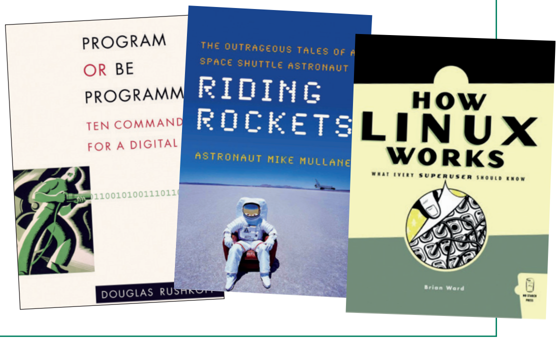
53

### Android x86

It's great on phones, but will the touch-friendly version of Linux work as a desktop OS? **Mike Saunders** finds out, and leaves finger marks all over the screen.

## BOOKS AND GROUP TEST

We dearly love Raspbian. It's optimised for the Raspberry Pi, is based on the fantastic Debian Linux distro, and has tons of documentation. But it's not the only fruit. The standard Noobs installer offers five other choices, all of which are brilliant in their own way, and you owe it to yourself to try them. In books, there's a range of Linuxy and non-Linuxy titles to get your teeth into, the pick of which is an oldie but still a goodie – *The Cathedral and the Bazaar*, a founding tome of the Free Software ethos. Cosy up with a copy as the nights draw in...


PROGRAM OR BE PROGRAMMED
TEN COMMANDS FOR A DIGITAL AGE
DOUGLAS RUSHKOFF
THE OUTRAGEOUS TALES OF A SPACE SHUTTLE ASTRONAUT
RIDING ROCKETS
ASTRONAUT MIKE MULLANE
HOW LINUX WORKS
WHAT EVERY SUPERUSER SHOULD KNOW
Brian Ward

# Wacom Intuos Pro

**Graham Morrison** continues his journey to becoming an old master by getting his hands on a tasty graphics tablet.

**DATA**

**Web**
wacom.com
**Developer**
Wacom/Community drivers
**Price**
£299

We've become smitten by drawing things on our Linux desktops, and we've had quite a few emails from readers to say they feel the same. We could probably form some kind of clandestine art club for geeks held in the candlelit cellar of The White Hart every other Thursday. But until now, we've only really been dabbling with applications like *Gimp* and *Krita* while clicking around with a regular mouse.

This isn't bad. It's how lots of great digital artists have created lots of great art. But mice feel neither especially creative nor particularly precise. Which is perhaps why nearly every designer we've ever met favours the graphics tablet. These things have been around for a long time, and come in two parts. The first is the stylus, which acts as your virtual pen. It feels and weighs the same as one, only it's made of plastic and doesn't leave a mark on paper.

The second part is the tablet itself, which acts as your writing surface. For artist, this surface is important, because they want to duplicate the feel of the surfaces they'd typically draw upon, so they need to offer just the right amount of friction to give the artist enough control and feedback over their movement.

**"Nearly every designer we've met favours the graphics tablet over the mouse."**

Esoteric hardware like this is exactly the kind that doesn't normally have good support for Linux, especially when Apple's OS X is the traditional domain for those artistic types. But the primary reason for this review is that we're happy to report that there are some excellent drivers for many of the devices from the market leader, Wacom. The drivers themselves are developed by the Linux users themselves, rather than Wacom, which is an important distinction. But these devices have been in development for over 10 years and the developers are able to keep up with any major development in tablets. They're considered some of the best tablets with the best compatibility you can get. Wacom was also more than happy to send us a device for review, knowing we were only going to consider the Linux compatibility, so it sounds like there's a good relationship between the two. But it's probably worth remembering that Linux support is always going to be limited to the community, rather than the official channels (who do at least link to the Linux drivers).

Along with the stylus stand you get 10 nibs – five standard, one flex, one stroke nib, and three felt.

The Intuos Pro can work both wired and wirelessly with the bundled battery, expansion and USB dongle.

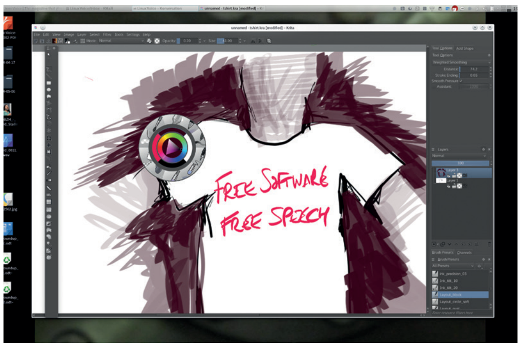
## Rococoagogo

We've been sent the medium model in the range – there's one smaller and one larger, but they all perform identically. The tablet itself is relatively large, occupying about the same footprint as a 15-inch laptop (it's 380 x 251 x 12mm) and weighs just under a kilogram, making it relatively travel friendly. We don't think a larger one would be a benefit. Size is important, because you need the space to place the tablet almost directly in front of your screen, and it's far less likely to give you RSI than a mouse. There are buttons and a touch-sensitive dial on the left,and a USB connector on the right, although you can change the orientation to suit whatever works for you. You can also operate this model wirelessly by adding the bundled rechargeable battery and wireless extension with the tiny wireless dongle plugged into your PC. We had no problem getting this to work, but neither did the USB cable bother us too much.

If you've not used a tablet before, they take a little getting used to – the main difference being that movement is now absolute rather than relative. Place the stylus in the same place and the cursor will appear in the same place, although this can be changed. You can move the cursor around the screen by hovering

## Tablet apps

We used the Intuos with as many Linux desktop applications as we could find, and they generally all worked well. Our favourite was *Krita*, purely because it was the only application to really make the most of all the tablet's features, and its brush and pencil models can trick the mind into think you're really drawing. *Gimp* is also great to use with the tablet and doesn't tax your CPU in the same way. You do need to customise how it interprets the tablet though, which can be done from the Edit > Input Devices menu option by enabling 'Screen' control for the pen, pad, eraser and stylus input. *Inkscape* is the same and also has some great drawing modes, plus the advantage that you're creating vectors rather than bitmap images. We also had very little problem using the tablet as our default input for a desktop, although we couldn't find a way to enable the touchpad feature which works on other operating systems.



Look out for our Autumn/Winter collection coming to an online store near you.

the stylus a few millimetres above the touch surface, just as you when sketching with a pencil. The package includes a variety of other nibs, removed using a chrome tweezer, which attempt to emulate the characteristics of a flexible brush, or a softer pencil, although they have no effect on the data. Prodding the surface with the stylus is the equivalent to a left click of the mouse and two further buttons on the stylus have a default configuration of right and middle mouse clicks.

### Prometheus unbound

Our next step was to try the tablet with some drawing software, and given our recent experience, the first we wanted to try was *Krita*. However, we were surprised to discover it crashes immediately, spitting out an error; "Rel Vert Wheel 11 -> 6" to the standard output. What our system was missing was the all essential **xf86-input-wacom** for the windowing system, which is a standard package installation for every distribution we looked at. With that installed, *Krita* launched without any issue and we were immediately able to start drawing.

The texture of both the nib on the stylus and the tablet itself contrive to create a feeling akin to a pencil on paper. The more you increase the pressure, the darker the impression on the virtual canvas (although this is entirely governed by your software), and the defaults have you pressing quite hard to get the darkest lines.

*Krita* has a tablet configuration setting that enables you to adjust the curve of the pressure you apply, so you could make more of a mark by pressing lightly, for example, or less of a mark by pressing harder. The resolution of the tablet means you can create incredibly fine lines, smaller even than a cross-hatched Escher drawn in Indian ink. The tilt function also worked perfectly within *Krita* by changing the shape of the brushes that support the feature. Using the 2B pencil brush gave results almost identical to sketching with a pencil, and we wasted a long time playing with this. By default, *Krita* has also configured the quick brush and colour palette to appear when you press the first button on the stylus, and the Canvas Move mode for the second button. The eraser wasn't set to erase by default, but this can easily be changed, and some people prefer to use a keyboard shortcut anyway.

We wanted to experiment more with the options provided by the drivers to the operating system, and to access those we installed a package called **kcm-wacomtablet**. This is a setting panel for KDE and there's an equivalent for Gnome, but you can equally perform all the same options using the **xsetwacom** command installed with the driver. The KDE settings panel lets you adjust all the various options and apply them to profiles, which can be easily switched. This lets you create a configuration for *Krita*, for example, and switch easily to a different profile for working with *Gimp*. You can adjust the pressure threshold curves, re-assign any of the buttons surrounding the surface and change the functions of both the buttons on the side of the stylus and the nib and eraser, which is useful as you may want to use the eraser as a different kind of brush, for example. All of this worked without any problems, and you can easily see how a tablet like this could maximise your productivity when you've got everything configured to your liking. In fact, we loved the whole experience so much that even without really having that much interest in art, we'd seriously consider buying a Wacom tablet for the joy of just doodling with it.

**"We'd seriously consider buying a Wacom tablet for the joy of just doodling with it."**

### LINUX VOICE VERDICT

It's expensive but it's professional. And for once, the Linux drivers are a joy to use. Highly recommended if you have any artistic leanings.

★★★★☆

# Gnu MediaGoblin 0.7

**Ben Everard** investigates Gnu's ambitious project to conquer the world of web-based media sharing.

**DATA**

**Web**
www.mediagoblin.org
**Developer**
Gnu
**Price**
AGPL

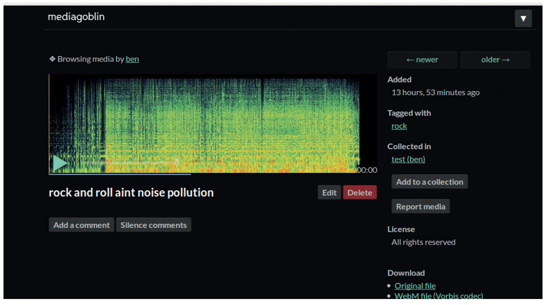
*MediaGoblin* aims to provide a free software alternative to media hosting sites like YouTube, Flickr and SoundCloud. Instead of focussing on a single media type, it allows users to upload and share many different types of media through a plugin system. To achieve this, the project launched a crowdfunding campaign that's raised over $60,000 to fund development (the campaign is ongoing – visit **http://mediagoblin.org/pages/campaign.html** to see the current status).

The team have come a long way towards realising their goal. You can now upload images, videos, sound files, documents and 3D models, which that should be enough to cover most people's media needs. *MediaGoblin* will re-encode them to the appropriate format, and make them available on the web page. You can also group them together into collections of various types, and viewers can add comments.

**"For now, Mediagoblin is a strong foundation for a useful project."**

Blogs are now supported as an additional media type, but it's currently considered experimental, so use it at your own risk. Throughout the application, markdown is supported for adding formatting to comments and descriptions.

The Pump API allows users to share their content with (and upload content from) other applications. This means that *MediaGoblin* doesn't have to live in isolation, but can become part of an ecosystem. This feature is new in 0.7, so as yet, not much supports it, but hopefully that will start to change soon. In future releases, the developers hope to include full federation, which will enable users to seamlessly move across different *MediaGoblin* sites.


mediagoblin

❖ Browsing media by ben

rock and roll aint noise pollution

Added: 13 hours, 53 minutes ago
Tagged with: rock
Collected in: test (ben)
License: All rights reserved
Download:
- Original file
- WebM file (Vorbis codec)

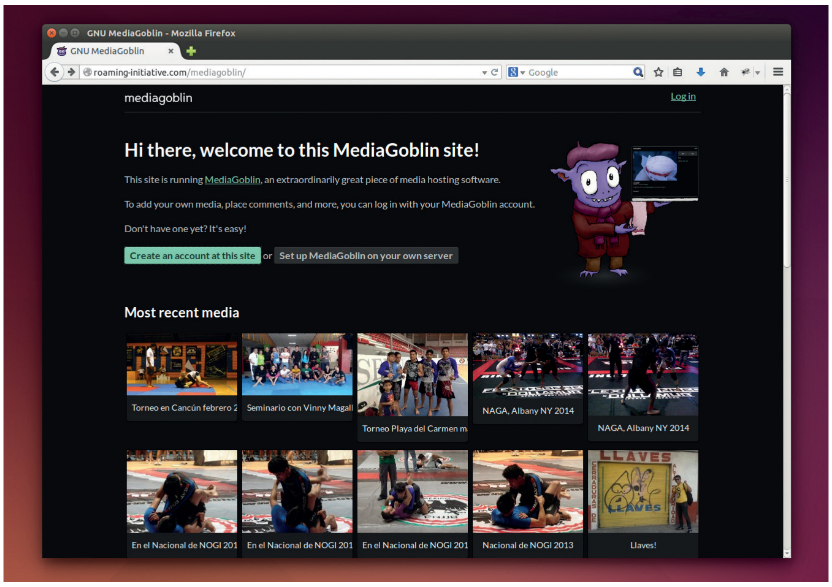
*MediaGoblin* enables you to tag media with a licence from all rights reserved to public domain.

To us, the biggest problem with the current version is the layout. The web pages look good, but they don't always show what you might want them to. For example, the home page shows a feed of what's been uploaded, but not popular collections, or particular users, and there's no way to search or browse by tag. Once you've found the particular user you're interested in, you can look through their collections, but you can't just click through from the front page unless they've recently uploaded media. It makes it quite hard to find things when you know where they are, and almost impossible to browse for media. Discoverability is absolutely critical to media servers, and for *MediaGoblin* to become a viable alternative to commercial services, it needs to get a lot better.

## Limited wardrobe

*MediaGoblin* is themeable, but it's new and theme developers haven't yet caught up. Other than the two themes that come pre-installed, we couldn't find another one that worked with the latest version. Hopefully this will start to change as *MediaGoblin* gets more popular. Another important feature that's not yet available is the ability to embed media in other web pages. The popularity of YouTube videos around the web is a testament to just how effective this can be in spreading content. This is currently being worked on, so should be available soon.

For now, *MediaGoblin* is a strong foundation for a useful project. However, there is still quite a bit of work to do before the project offers a real alternative to closed-source media hosting. Given the current pace of development, we don't expect it to take too long.


GNU MediaGoblin - Mozilla Firefox
roaming-initiative.com/mediagoblin/

mediagoblin

Hi there, welcome to this MediaGoblin site!

This site is running MediaGoblin, an extraordinarily great piece of media hosting software.

To add your own media, place comments, and more, you can log in with your MediaGoblin account.

Don't have one yet? It's easy!

Most recent media

Torneo en Cancún febrero 2 · Seminario con Vinny Magall · Torneo Playa del Carmen m · NAGA, Albany NY 2014 · NAGA, Albany NY 2014 · En el Nacional de NOGI 201 · En el Nacional de NOGI 201 · En el Nacional de NOGI 201 · Nacional de NOGI 2013 · Llaves!

GNU MediaGoblin
Google
Log in
Create an account at this site
or
Set up MediaGoblin on your own server

There are a few public *MediaGoblin*, servers like this one provided by the Roaming Initiative (**www.roaming-initiative.com/mediagoblin**).

**LINUX VOICE VERDICT**

A good basis, but there's a long way to go before it starts to worry the incumbents.

★★★☆☆

# Calibre 2.0

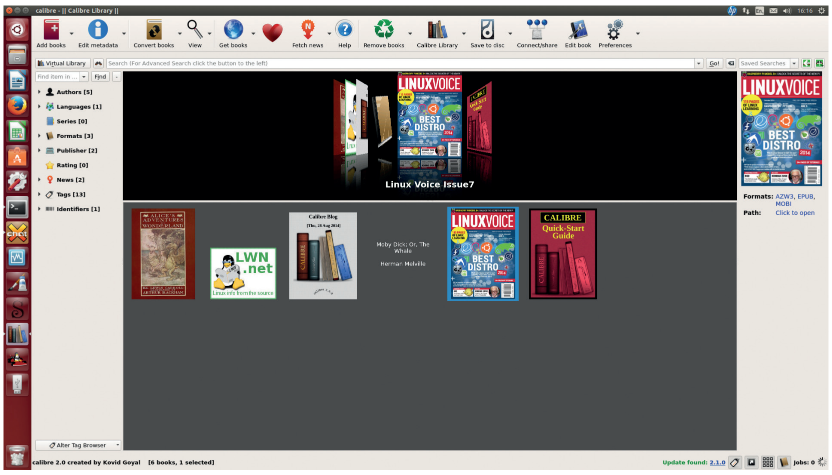
**Ben Everard** saves the trees and casts his paper books aside in favour of environmentally friendly electronic ones.

Calibre is one of the top ebook utilities on Linux. In its basic use, you can use it to download and (if necessary) manipulate ebooks to create your own book shelf, then upload particular books to your e-reader devices (a wide variety are supported, including Android MTP phones and tablets). It can also display ebooks, but desktop and laptop screens are rarely good for reading from. Perhaps, as tablets get more powerful, *Calibre* will see more installs on reader hardware (*Calibre* supports touchscreen controls for Windows tablets, but not yet for Linux ones).

The biggest change in version 2.0 is that it's shifted from the *Qt 4* toolkit for its graphical interface to *Qt 5*. This has cleared a lot of problems that were the result of *Qt 4*. However, it does mean that the project no longer supports Windows XP. We won't take any marks off for that though – *Calibre* has supported XP later than Microsoft, and it's high time you switched any remaining XP machines to Linux anyway.

If you've got enough books to make managing them difficult, *Calibre* lets you sort and filter them by author, tag, language and various other parameters. You can also convert between most popular ebook formats, so you can manage books across a range of devices. This all works well, but the interface is a little lacklustre. The icon theme is inconsistent (some are flat, some aren't, one's animated and the save icon is like nothing we've seen before), the window feels cluttered even though it's actually quite a simple layout, and it's not always obvious where particular options are. None of this is bad enough to put us off using it, but the software would really benefit from a little more attention to design.

*Calibre* can get books from a wide range of sources including free (both as in beer and speech) and paid-for stores. The list of sources is exhaustive, so it's a great way to quickly find the cheapest store for a particular book, and it highlights which sources include DRM. In addition to grabbing ebooks, *Calibre* can also be configured to download RSS feeds allowing you to create a sort of eNewspaper to be read offline. The RSS is automatically converted to ePub for upload to an e-reader.



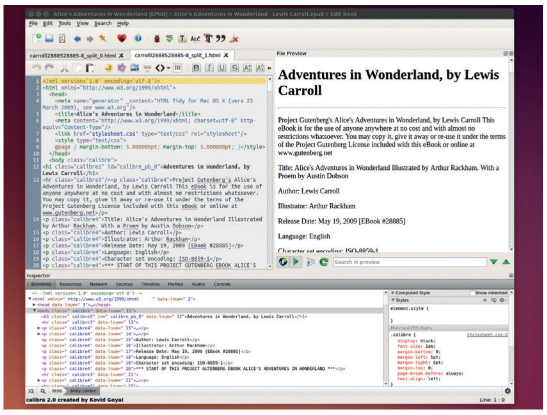
All subscribers can now get ePub copies of Linux Voice, which work well with *Calibre* and e-readers.

## Ebook editor

*Calibre* isn't just for reading and managing books. It also includes quite a capable ebook editor (for ePub and Kindle formats). This includes a side-by-side HTML editor and preview, CSS tools, inspector and an ePub validator. Of course, all these tools are available separately, but *Calibre* brings them together into a sort of ebook integrated development environment.

Although editing ebooks isn't *Calibre*'s main function, it's one it performs very well. Options like pretty-printing the HTML, removing unused CSS and smartening the punctuation are useful for working with automatically generated HTML files, which can be something of a mess.

It is, perhaps, let down a little by the lack of a WYSISYG editor, so all changes have to be made directly to the raw HTML. This means it isn't great for writers (especially non-technical ones) looking to create a book, but it is the best open source tool we know of for editors compiling and tidying up the book. With version 2, *Calibre* continues its domination of the open-source ebook scene. LV


Alice's Adventures in Wonderland [EPUB] :: Alice's Adventures in Wonderland - Lewis Carroll.epub :: Edit Book
File Edit Tools View Search Help
carrolll2888528885-8_split_0.html
carrolll2888528885-8_split_1.html
File Preview
Inspector
Elements Resources Network Sources Timeline Profiles Audits Console
Computed Style
Show inherited
Styles
Matched CSS Rules
Search in preview
calibre 2.0 created by Kovid Goyal
Line: 1 : 0

```
<?xml version='1.0' encoding='utf-8'?>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta name="generator" content="HTML Tidy for Mac OS X (vers 25
March 2009), see www.w3.org"/>
    <title>Alice's Adventures in Wonderland</title>
    <meta content="http://www.w3.org/1999/xhtml; charset=utf-8" http-
equiv="Content-Type"/>
    <link href="stylesheet.css" type="text/css" rel="stylesheet"/>
    <style type="text/css">
    @page { margin-bottom: 5.000000pt; margin-top: 5.000000pt; }</style>
  </head>
  <body class="calibre">
<h1 class="calibre2" id="calibre_pb_0">Adventures in Wonderland, by
Lewis Carroll</h1>
<hr class="calibre3"/><p class="calibre4">Project Gutenberg's Alice's
Adventures in Wonderland, by Lewis Carroll This eBook is for the use of
anyone anywhere at no cost and with almost no restrictions whatsoever.
You may copy it, give it away or re-use it under the terms of the
Project Gutenberg License included with this eBook or online at
www.gutenberg.net</p>
<p class="calibre4">Title: Alice's Adventures in Wonderland Illustrated
by Arthur Rackham. With a Proem by Austin Dobson</p>
<p class="calibre4">Author: Lewis Carroll</p>
<p class="calibre4">Illustrator: Arthur Rackham</p>
<p class="calibre4">Release Date: May 19, 2009 [EBook #28885]</p>
<p class="calibre4">Language: English</p>
<p class="calibre4">Character set encoding: ISO-8859-1</p>
<p class="calibre4">*** START OF THIS PROJECT GUTENBERG EBOOK ALICE'S
```

**Adventures in Wonderland, by Lewis Carroll**

Project Gutenberg's Alice's Adventures in Wonderland, by Lewis Carroll This eBook is for the use of anyone anywhere at no cost and with almost no restrictions whatsoever. You may copy it, give it away or re-use it under the terms of the Project Gutenberg License included with this eBook or online at www.gutenberg.net

Title: Alice's Adventures in Wonderland Illustrated by Arthur Rackham. With a Proem by Austin Dobson

Author: Lewis Carroll

Illustrator: Arthur Rackham

Release Date: May 19, 2009 [EBook #28885]

Language: English

Character set encoding: ISO-8859-1

```
<!--?xml version='1.0' encoding='utf-8'?-->
▼<html xmlns=" http://www.w3.org/1999/xhtml    " data-lnum=" 2">
 ▶<head data-lnum=" 3">…</head>
 ▼<body class=" calibre" data-lnum=" 11">
    <h1 class=" calibre2" id=" calibre_pb_0" data-lnum=" 12">Adventures in Wonderland, by Lewis Carroll</h1>
    <hr class=" calibre3" data-lnum=" 13">
   ▶<p class=" calibre4" data-lnum=" 13">…</p>
   ▶<p class=" calibre4" data-lnum=" 14">…</p>
    <p class=" calibre4" data-lnum=" 15">Author: Lewis Carroll</p>
    <p class=" calibre4" data-lnum=" 16">Illustrator: Arthur Rackham</p>
    <p class=" calibre4" data-lnum=" 17">Release Date: May 19, 2009 [EBook #28885]</p>
    <p class=" calibre4" data-lnum=" 18">Language: English</p>
    <p class=" calibre4" data-lnum=" 19">Character set encoding: ISO-8859-1</p>
    <p class=" calibre4" data-lnum=" 20">*** START OF THIS PROJECT GUTENBERG EBOOK ALICE'S ADVENTURES IN WONDERLAND ***</p>
    <hr class=" calibre3" data-lnum=" 21">
   ▶<p class=" calibre4" data-lnum=" 21">…</p>
```

```
element.style {
}
.calibre {                          stylesheet.css:2
  display: block;
  font-size: 1em;
  margin-bottom: 0;
  margin-left: 5pt;
  margin-right: 5pt;
  margin-top: 0;
  page-break-before: always;
  text-align: left;
}
```

The side-by-side HTML editor and preview make it easy to sort out any display problems with an ebook.

## LINUX VOICE VERDICT

Essential software for everyone using e-readers. *Calibre* is only let down by an untidy interface.

★★★★☆

## DATA

**Web**
www.calibre-ebook.com
**Developer**
Kovid Goyal
**Licence**
GPLv3

# Energenie Radio controlled sockets

**Ben Everard** decided against powering his latest project from a lightning rod attached to the clock tower; instead he's using one of these.

**DATA**

**Web**
https://energenie4u.co.uk
**Developer**
Energenie
**Price**
£19.99

It's easy to use the GPIO pins on the Raspberry pi to switch low voltage devices on and off. Even components that need more current than the pins can provide can be handled using a motor driver, optical isolator or relay. This is easy to set up and not likely to damage your Pi. However, switching mains voltage is a different case entirely.

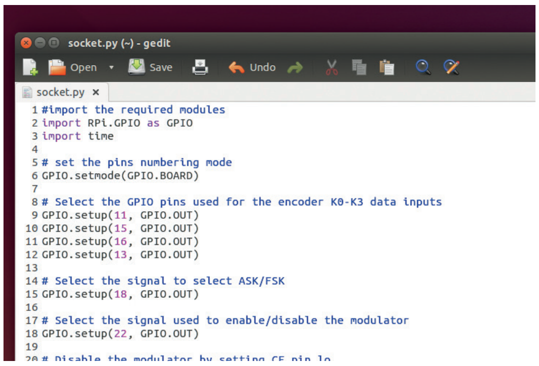
When you're dealing with mains voltage at 240V (OK, fine, 230V with a tolerance of +10% or -6%. Thanks EU!), things begin to get a little more tricky. Not only do you need more capable components to switch this level of voltage, they need to be driven by the low voltages that the Pi can supply. It also becomes more dangerous, as any mis-wiring could lead to hardware damage or worse.

These radio controlled sockets are a great solution to the problem. There's no wiring, so it's no more dangerous to use than using electrical appliances normally, and the Pi is air-gapped so there's no risk to that hardware. The manufacturer claims they can handle 13A, and while we had no problem switching high wattage devices, we weren't able to test them at the top of that range.

**"When you're dealing with mains voltage at 240V, things begin to get a little more tricky."**

The boxed set is a single Pi expansion board controller, and two radio controlled sockets (additional sockets are sold separately, and it's possible to use up to four sockets with a single expansion board). The expansion board works with every current model of the Pi (A, B and B+).

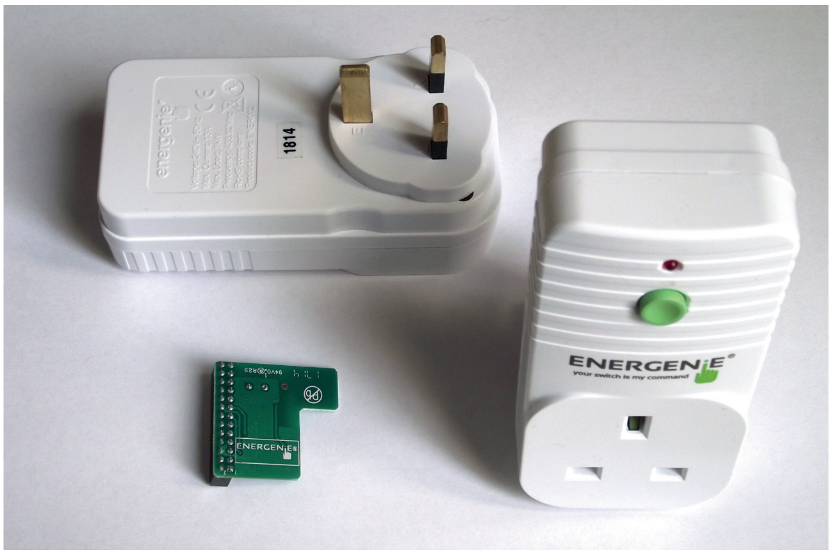
The protocol for controlling the expansion board is explained on the project's website (**https://energenie4u.co.uk/index.php/catalogue/product/ENER002-2PI**). It doesn't require any specialist software other than what's needed for controlling the GPIO pins. There is some example code in Python using the RPi.GPIO module, but it should be trivial to port this to any other language. For that matter, we wouldn't envisage any problems controlling the expansion board from any 3.3V controller, but we haven't tried it with anything other than the Pi.



```
#import the required modules
import RPi.GPIO as GPIO
import time

# set the pins numbering mode
GPIO.setmode(GPIO.BOARD)

# Select the GPIO pins used for the encoder K0-K3 data inputs
GPIO.setup(11, GPIO.OUT)
GPIO.setup(15, GPIO.OUT)
GPIO.setup(16, GPIO.OUT)
GPIO.setup(13, GPIO.OUT)

# Select the signal to select ASK/FSK
GPIO.setup(18, GPIO.OUT)

# Select the signal used to enable/disable the modulator
GPIO.setup(22, GPIO.OUT)

# Disable the modulator by setting CE pin lo
```

The example code is well commented, so it's easy to see what all the GPIO operations are for.

The board and sockets are also sold separately for £9.99 and £12.99 respectively.



## Programmers only need apply

There isn't any specific software (either graphical or command line) provided other than a simple example, so it's only suitable for programmers. That said, you could simply copy and paste bits from the example program, so you don't need much programming experience to make it work. Setting the hardware up was simply a case of pressing a button (the only button) on the socket before sending a command.

The listed range is 30m in open space. Some people have reported being able to extend the range by soldering an additional antenna on, but this is not officially supported.

These sockets really couldn't be easier to use for programmers, and while it would be nice to have a graphical application to make it really simple to get started, it's hard to see how this would be useful beyond demonstrating the capabilities. The hardware is really designed for letting you control things with scripts. At the simplest level, this could be turning lamps on or off, but really, it could be anything.

**LINUX VOICE VERDICT**

The easiest way of switching mains voltage from a computer, but only for those of us in the UK.

★★★★☆

# Android x86 4.4

Can a mobile OS work well on the desktop, or is this just the first step on the path to madness? **Mike Saunders** pops in a USB key and finds out…

It's a funny old world. The Linux kernel was born because its creator wanted to run a Unix-like OS on his home PC. And while Linux hasn't yet conquered the desktop, it has made big inroads into the server market over the last decade or so. More recently, Linux has been gigantically popular in the mobile space thanks to Google's Android, and now we've come full circle: running Android on your desktop. Back in issue 2's FOSSpicks section we had a brief look at the 4.4 release candidate from a "curious dabbler" perspective, and now we want to see if it's really usable as a day-to-day OS.

Android x86 is available as an ISO image, but you can easily convert it to run from a USB key with *UNetbootin*. Both formats let you install it to your hard drive. We dug out our trusty old ASUS K52F laptop, which served as a reliable workhorse for many years, to see how well Android supports common PC hardware. The ASUS machine has never had any major problems running normal Linux distributions – but we didn't know what to expect from a significantly different platform.

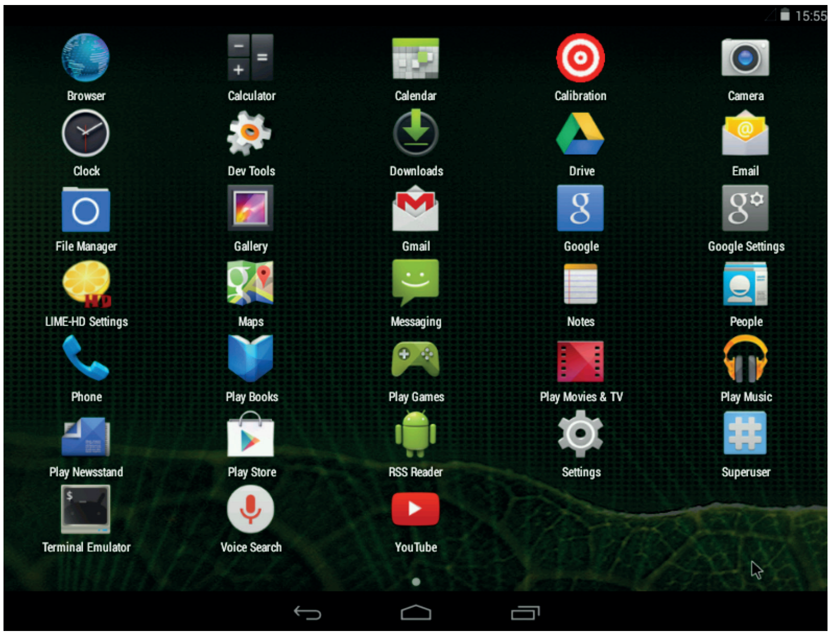
Well, the results were good. Video performance was great, sound and the webcam worked out of the box, and the Fn keys for controlling audio levels and screen brightness also did their job without manual intervention. The touchpad responded well to multi-touch gestures, and in terms of power management, Android didn't use significantly more battery juice than the previous Xubuntu 13.04 installation.

## Familiar territory

Although Android x86 isn't an official product from Google, and is developed and maintained as an unofficial port, it comes with the usual host of Google programs: Maps, YouTube, Drive, Gmail, Play Music/Games/Newsstand and so forth. Handily, a terminal emulator is installed – but don't expect much in the way of a typical GNU/Linux userland. It's essentially *BusyBox* with a smattering of tools such as OpenSSH.

So, what's Android x86 like as a desktop OS? If you're familiar with it on a mobile phone or tablet, you'll pick it up in seconds: it's almost exactly the same. (Indeed, many of the dialogs refer to "your tablet" during configuration.) Clicking and swiping to bring up the System and Action bars feels a bit strange at first, as does switching applications, but fortunately Alt+Tab is still available for those of us who prefer a more traditional approach. The biggest potential obstacle is the inability to resize windows – or show anything side-by-side. It's not a huge deal on smaller screens, but it doesn't make much sense if you have a 27" monitor.

Ultimately, Android x86 is too limiting for regular desktop Linux users, but it's a great Windows alternative for non-technical types. If you've got friends or relatives desperately trying to get off XP, and all they do is some light browsing, email and watching YouTube videos, this is exactly what they need. It doesn't have the richness of a full Linux installation, but it has fewer moving parts to break, and almost anyone can pick it up quickly. LV



A host of apps is included, including the usual suspects from Google.

**DATA**

**Web**
www.android-x86.org
**Developer**
Google and community
**Price**
Free under OSS licences

15:58
DESCRIPTION
SUBSCRIBE
SUGGESTIONS
Super Mario 64 120 star Speedrun 1:49:06 (1:48:06 in SDA timing) by Siglemic
1,060,310 views
Super Mario 64 - Speed Run in 1:47:48 (120-Star)...
SpeedDemosArchiveSDA
1 year ago · 320,541 views
2:24:44
1:49:26
Super Mario 64 120 star

It's no replacement for Arch Linux, but Android x86 does a decent job for light browsing and communication tasks.

**LINUX VOICE VERDICT**

Surprisingly good, and a great "my first" Linux distro for non-savvy users who do everything on the web.

★★★★☆

## Program or Be Programmed: Ten Commandments for the Digital Age

**Ben Everard** finds out how to ensure computers are shaping our lives for the better.

*Program or Be Programmed* is a book about how the internet is shaping our society, and what we need to do in order to make sure the changes that it's bringing are beneficial. It's split up into 10 chapters, each of which deals with one commandment that is supposed to help ease one factor of technology. Program or Be Programmed is the title of the final chapter, and it's the only one about programming. The rest deal with how we interact with our machines, other people on line and new forms of media.

Rushkoff doesn't dive into the technicalities of how to follow his advice (he leaves that up to the reader). It's quite a jargon-heavy book, so non-geeks may struggle to follow parts of it.

It's quite a short book – just 144 small form-factor pages – but the information is quite dense. This is probably the best and worst fact about the book. It felt good to be able to read it all in a few hours, but at the same time the rush of information meant we couldn't fully digest it in a single sitting.

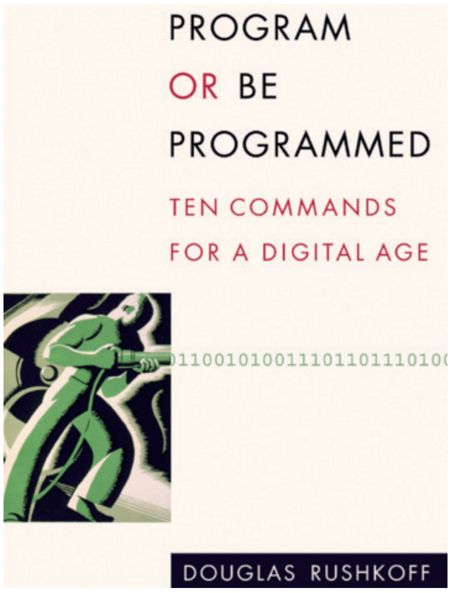
While we don't completely agree with the proposed 10 commandments, the thorough reasoning he provides for each one make thought-provoking reading, which, we suspect, is really the point of the book. Maybe we should start our own book club so we can all discuss our thoughts on IRC?

**LINUX VOICE VERDICT**

**Author** Douglas Rushkoff
**Publisher** OR Books
**ISBN** 978-1935928157
**Price** £11

Even if you don't fully agree with Rushkoff, this book provokes a debate we should be having.

★★★★☆



Will we sleepwalk into dystopia, or harness technology for the benefit of humanity?

## Riding Rockets: The Outrageous Tales of a Space Shuttle Astronaut

**Ben Everard** learns about the good, the bad and... er... monkey faeces.

The space shuttle era was the golden era for astronauts. The large capacity of the craft meant that many more people went up in each mission than do in the smaller and often unmanned rockets that make most missions today.

The reader is introduced a world where applicants lie and cheat to get onto the astronaut program, then compete against each other for that greatest of prizes: a trip into space. If you're used to seeing astronauts as staid professionals, *Riding Rockets* could be quite a shock for you.

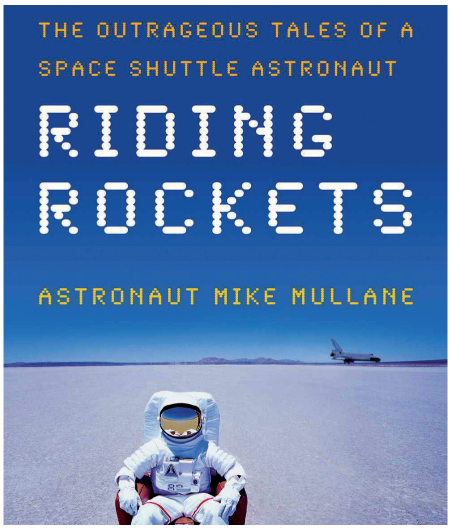
This slightly sordid version of events comes first hand from Mike Mullane, one of "The F.....g New Guys" brought into NASA at the start of the shuttle era. Mullane takes us through the edge-of-your-seat excitement of going into space, the sexism and the clashes between military aviators and scientists. As the shuttle program goes on, Mullane becomes more and more disenfranchised by hubris and mismanagement at NASA, which he says led to the tragic deaths of the crews of Challenger and Columbia, but he never loses his infectious enthusiasm for all things related to space travel. *Riding Rockets* is laugh-out-loud funny, exciting and sad.

**LINUX VOICE VERDICT**

**Author** Mike Mullane
**Publisher** Simon & Schuster Ltd
**ISBN** 978-0743276832
**Price** £10.99

It's equal parts cautionary tale, exposé and hilarious memoir.

★★★★☆



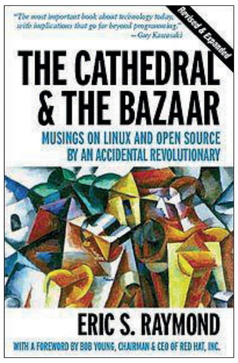
The book that proves that astronauts are mere humans like the rest of us.

## The Cathedral and the Bazaar

**Ben Everard** revisits the work of an accidental revolutionary.

Prior to Linux, software (even free software) tended to be developed in a closed space with a trusted team of developers, then pushed out into the world as a completed product. This was the best practise of the time. According to most theories of software development prior to about 2000, the Linux kernel shouldn't work. Hundreds of people just wouldn't be able to collaborate on code in active development – or so it was thought. At the time, the dominant theories said that the complexity of managing people on that scale would be overwhelming and the project would languish in a bug-ridden stupor. However, there's no denying that – from a software development perspective – the Linux kernel has been an overwhelming success.

The book is a little dated now, but just about the only thing that's changed in the past decade is that the author's views have become mainstream. Projects like GitHub are based on the principals Linus developed, and Eric S Raymond encapsulated. *The Cathedral and the Bazaar* remains a must read.



The *Cathedral and the Bazaar* has inspired many people, including Jimmy Wales, co-founder of Wikipedia.

### LINUX VOICE VERDICT

**Author** Eric S Raymond
**Publisher** O'Reilly Media
**ISBN** 978-0596001087
**Price** £10.99

*The Cathedral and the Bazaar* is the definitive book on Linux-style open software.

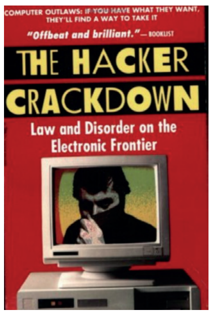
★★★★★

## The Hacker Crackdown

Even tech history repeats itself, discovers **Graham Morrison**

This is a book from 1992 about the subversion of a technology that dominated the previous decade – dial-up bulletin board systems and the misuse of old telephone systems. Subsequently, it's also about the rise of a hacker culture in a pre-internet world. It's the first instance we can think of where geek culture clashes with authority after techniques that start as cool hacks becomes exploited by wider communities, eventually leading to Operation Sundevil, possibly the first crackdown on hackers by a governmental institution.

Operation Sundevil and similar initiatives led to the creation of the Electronic Frontier Foundation in an attempt to bridge the misunderstanding between law enforcement agencies and the technology they believed was being misused. What's most fascinating about reading this book over 20 years later (it's free), is that so little has essentially changed. There's still this divide and the hacker subculture remains, naturally adapted to life on the internet and the web.



Originally published in 1992, *The Hacker Crackdown* has been in the public domain since 1994.

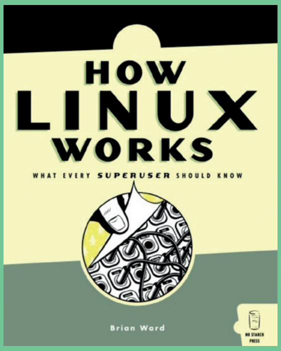
### LINUX VOICE VERDICT

**Author** Brice Sterling
**Publisher** Bantam Books
**ISBN** 0-553-56370-X
**Price** Free ePub or second hand on paper

A fascinating slice of history and antiquated tech systems that remains relevant today.

★★★★★

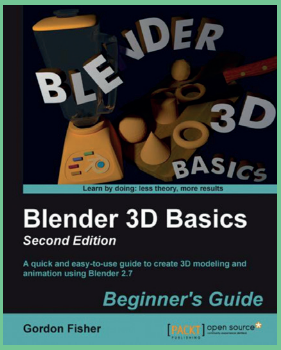
## ALSO RELEASED…



Brian Ward is the author of The Linux Kernel HOWTO.

### How Linux Works

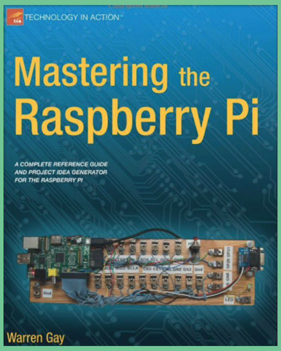
We all need at least one book that covers broadly how Linux works. Ours has always been *Linux in a Nutshell*, but that is perhaps getting a little dated now. This new edition covers similar territory – there's no desktops – but it has a different, less formal style.



*Blender*'s capabilities are getting so impressive, it's slightly scary.

### Blender 3D Basics Beginner's Guide

The open source 3D rendering engine, *Blender*, has become an industry changing application. But it is difficult to get into, so we're very happy to see more books trying to ease people into the *Blender* way of doing things. They're all helping to strengthen a great system.



Unlike the many beginner titles, this is a book that promises schematics and details.

### Mastering the Raspberry Pi

There are many Raspberry Pi books to choose from, but this is a title pitched at 'experienced electronics engineers, Linux admins and users' which should put it on the more technical side. That's an area that hasn't been covered adequately, so this could be a good fit for people looking for more advanced projects.

LINUX VOICE

# GROUP TEST

RASPBERRY PI DISTROS

**Graham Morrison** burns a hole in his SD card after installing every raspberry flavoured distro he can get his hands on.

## On Test

### Raspbian

**URL** www.raspbian.org
**VERSION** 20/06/2014
**LICENCE** Mostly GPL
*Most consider this the official operating system of the Pi, and the one to beat.*

### Pidora

**URL** http://pidora.ca
**VERSION** 20
**LICENCE** Open source
*Love the Pi? Love Fedora? See what they've done here with the name?*

### OpenElec

**URL** http://openelec.tv
**VERSION** 4.07
LICENCE **GPLv2**
*An ultra-minimal distro built from the kernel up to play your media.*

### OSMC/Raspbmc

**URL** www.raspbmc.com
**VERSION** June 2014
**LICENCE** Open Source
*Unlike OpenElec, this is a media distro paired down from Debian.*

### RISC OS

**URL** https://www.riscosopen.org
**VERSION** RC12a
**LICENCE** Non Open Source
*An ARM operating system from the 1990s can't still be any good can it?*

### Arch Linux

**URL** http://archlinuxarm.org
**VERSION** June 2014
**LICENCE** Open Source
*This is the same Arch you know and love, only built for ARM.*

## Raspberry Pi distros

The Raspberry Pi needs no introduction. It's a credit card sized Linux computer that can be used for everything from brewing beer to playing arcade games. And it's usually found running its default Raspbian distribution. But this being Linux, Raspbian isn't the only fruit for your Pi. And because your Pi is supposed to be played with, subverted, coerced and occasionally broken, you owe it to yourself to try something else.

Not only will a different distribution give you a different perspective on such familiar hardware, you may well find a different distribution suits your requirements better than the default options, or learn something about what you need or don't need.

Raspbian is a great all-rounder, for example, but how does it perform if you require only a minimal installation, or you want your Pi to stick to the back of your television and be used purely for media playback? Is it better to install the media software you need onto a new installation, or use a distro created for a single purpose?

We've looked at the six different distributions you can install through the Noobs installer, which means you can avoid the **dd** roulette of copying a distribution image across from your Linux machine and onto the SD card. It's also important to remember that these distributions aren't really competing directly against one another. Two are designed specifically for media playback, for example, which is why we pit them against one another, and while RISC OS is fun to use, it's not a realistic replacement for something like Raspbian.

We should give the same caveat for the table of statistics we present at the end. A value like free memory can be pernicious because the kernel uses memory in strange and dynamic ways, and in the case of the distributions running *XBMC*, the amount of free memory fluctuated from one second to another without us performing any actions.

**"Raspbian is a great all-rounder, but what if you require only a minimal installation?"**

### HARDWARE

For our testing, we used an old Raspberry Pi model B with a 4GB class 4 SD card connected to wired networking rather than using a wireless dongle. As ever, the most important hardware requirement is a decent powered USB hub, as the early models are renowned for their lack of USB power while doing more than one thing at the same time (this has improved with the B+). We left the amount of RAM assigned to the GPU at its default value and didn't overclock any installation other than with OpenELEC and Raspbmc, but this is something you should look into if you're using your Pi as a regular desktop.

## Installing distros

### Don't take any risks with dd – Noobs packages all the distros in into an easy to use installer.

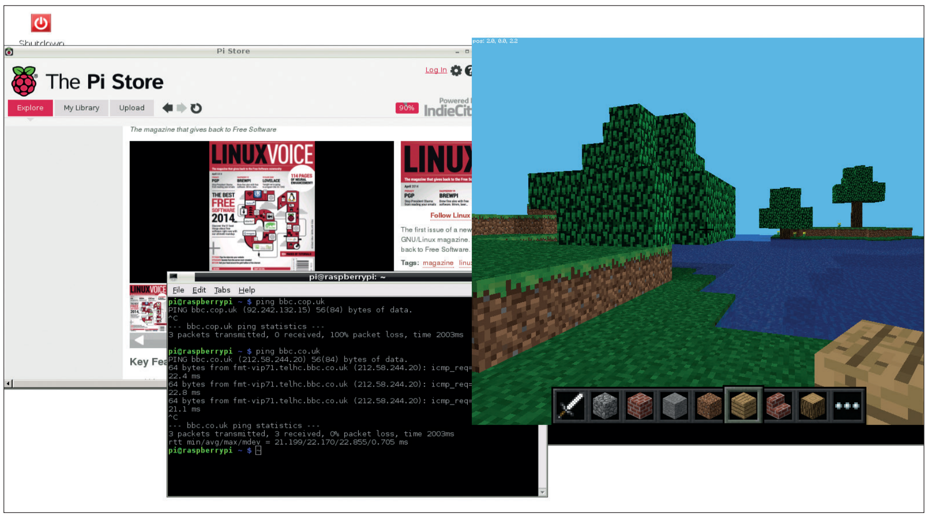
Many users will simply copy the raw image of their downloaded distribution using the **dd** command or one of its GUI equivalents. But this is potentially dangerous, as it requires you to enter the device ID of your SD card. Get this wrong, and you may overwrite valuable data. A better alternative is the *Noobs* installer. This can be either a 20MB network install download, or 1.5GB file that doesn't require network access. When either is downloaded, installation is as simple as copying the contents of the unarchived Zip folder into the root of your SD card and booting your Pi with it after safely unmounting the device. When the Pi boots, you'll see menu pop up inviting you to install all the distros on test here plus a tool to add a 512MB data partition. Depending on space, you'll also be able to install more than one at the same time!

## Raspbian

### Ra Ra Raspbian, the Foundation's greatest tech machine.

This is the distribution to beat. Raspbian is the distro recommended by the Raspberry Pi Foundation. It's the distribution used by nearly all tutorials and much of the official documentation. It's a distribution that's funded by the Foundation and it's the first to take advantage of much of its investment. There are Wayland and Weston patches funded and built specifically for the Raspberry Pi, for example, that came to Raspbian first. It's also the only distribution that will work perfectly with the Foundation's expansions and peripherals from day one. And it's probably got the best name. When combined Raspian's Debian foundations and its huge software repository, it's almost unbeatable. We say almost because there are still four pages to go.

This experience starts with its ease of configuration, although we spare a thought for those new to Linux. The grey, blue and black of the *Curses* configuration tool has all the charm of a 1993 MS-DOS game's audio configuration panel, untouched by the touchscreen revolution. But it is functional and fast. From this simple menu, you can expand the filesystem to use your entire SD Card, overclock the hardware, enable the camera module and tell Raspbian you want the desktop booted by default.



```
pi@raspberrypi ~ $ ping bbc.cop.uk
PING bbc.cop.uk (92.242.132.15) 56(84) bytes of data.
^C
--- bbc.cop.uk ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2003ms

pi@raspberrypi ~ $ ping bbc.co.uk
PING bbc.co.uk (212.58.244.20) 56(84) bytes of data.
64 bytes from fmt-vip71.telhc.bbc.co.uk (212.58.244.20): icmp_req=
22.4 ms
64 bytes from fmt-vip71.telhc.bbc.co.uk (212.58.244.20): icmp_req=
22.8 ms
64 bytes from fmt-vip71.telhc.bbc.co.uk (212.58.244.20): icmp_req=
21.1 ms
^C
--- bbc.co.uk ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 21.199/22.170/22.855/0.705 ms
pi@raspberrypi ~ $ 
```

The desktop is packed with dozens of points from which you can launch your Pi Adventure.

### startx

Postponing this choice, rather than booting to the desktop first, is a particularly good idea, as many Pi users are going to want to stick with the command line, and if not, the desktop is only an option away. So too is the SSH server that's already running, meaning you can remotely configure and install packages from the very first boot, and the pre-installed build environment, making this a perfect distribution for just getting on with what you want to do.

The Raspberry Pi's limited performance and memory does restrict the desktop, making Raspbian's default LXDE a perfect choice. It's quick, functional and low on resources. The default configuration looks a little like Windows 98 running with a dark theme, but at least it's a nine-year advance on the MS-DOS configuration panel, and LXDE does everything you need. The desktop is littered with links to great starting points, such as a Python games launcher or the Scratch launcher – perfect for classes and tutorials, although we missed a more obvious link to package installation. There's also some proprietary software in the shape of Pi versions of *Mathematica 10* and its associated Wolfram language (see our review of the £195 latest release in LV007). Both are incredibly powerful, but the former runs its loading off a Sinclair ZX Microdrive, so we're not quite sure how useful it's going to be. Far more successful is the Pi edition of *Minecraft*, which need to be installed manually but runs perfectly and will help the Pi win teenage kudos whenever it's installed.

We experienced an update hitch with the 233MB **wolfram-engine** package stalling at 98%. The only options are to remove the package beforehand or remove it from the update (**aptitude hold wolfram-engine**), and we wonder why things like *Qjackctl* are included when there's no instantly workable Jack configuration, but these are both tiny blemishes on an excellent Linux distribution.

**VERDICT**

Some weird proprietary choices, but an unrivalled foundation for all other Pi experimentation

★★★★★

## Pidora

### The closest you'll get to a full distro experience on your Pi.

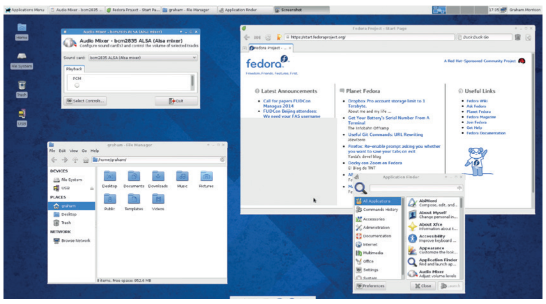
Fedora is the cutting-edge RPM based distribution that's a direct descendant to the old Red Hat releases, and it's brilliant to see a version that's been built for the Raspberry Pi. Like Debian, its creators are fortunate in that the root distribution is available for many different platforms and has been around long enough that the diminutive ARM chip of the Pi shouldn't pose too much of a challenge. Pidora also wins with its boot visuals as it smoothly scrolls and flips a large logo across where other distributions present the boot log – we half expected some chiptunes to be played alongside!

This is also the only distro we looked at that had anything like an installer. When you first boot Pidora, you are asked to accept a licence, choose a keyboard, create a user and a root account, set the time (the default is 31/12/1969!) and whether or not to boot to a graphical desktop. This all means your passwords and accounts are unique from the first boot, unlike nearly every other Pi distro, which is good for the SSH server that's already running. We also liked the way you're asked about overscan, as most of us connecting the Pi to a monitor don't have to worry about this (and Raspbian defaults to overscan being enabled).

As a desktop distro, Pidora looks fantastic. The Fedora theme is the most polished of all the distributions we've looked at, and Xfce helps make it all feel like a modern computer.

Performance is an issue, however. Even opening the *Thunar* file manager takes seconds, which doesn't bode well for the all the regular desktop applications that have been installed alongside. For this reason, it's easier to consider Pidora as a CLI-based distribution that can take advantage of Fedora's huge package repository alongside the same excellent package management and system configuration.

"The Fedora theme is the most polished of all the distros we've looked at."


It's a great desktop, but it's slow. The white square in the middle of the screen is the frame update lag when we ran the screenshot utility.

**VERDICT**
We love the way this is unadulterated Fedora, but the desktop is going to frustrate some people.
★★★★☆

## Risc OS

### Oh the horror. This isn't Linux!

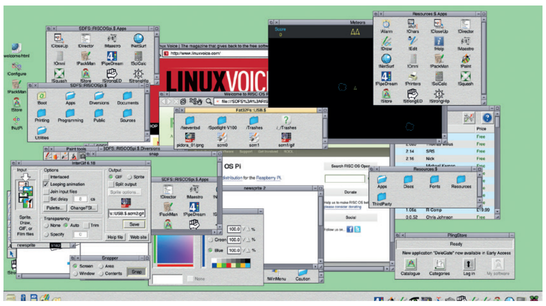
Once upon a time, there was a furious debate about which kind of CPU architecture was superior; RISC versus CISC. It doesn't really matter now, but there's some history here. Acorn, the creators of the BBC and whose naming convention inspired the Raspberry Pi, was rather fond of RISC and developed the first commercial RISC processor which they promptly put into the first RISC-based home computer, the Archimedes. And the Archimedes begat the first version of RISC OS (see page 104 for more of this back story).

The relevant part is that various departments of Acorn became ARM Holdings, the company now responsible for creating the most widely used CPU architecture ever, and the one used by the Raspberry Pi. Which is why porting RISC OS to the Pi has a certain karmic symmetry to it.

Running RISC OS in 2014 is part nostalgia, because it still looks and behaves in a way that will feel familiar to Archimedes veterans, and part practical. Even the Pi's ARM6 is way faster than the old ARM3, the chip for which Risc OS was first written. This makes it lightning fast for things like text editing and file management, as long as you're happy using applications that feel like they're from the mid 90s. The web browser, for example, is very quick, but it also feels like you're running *iBrowse* on an Amiga from the 20th century.

#### Back to the future

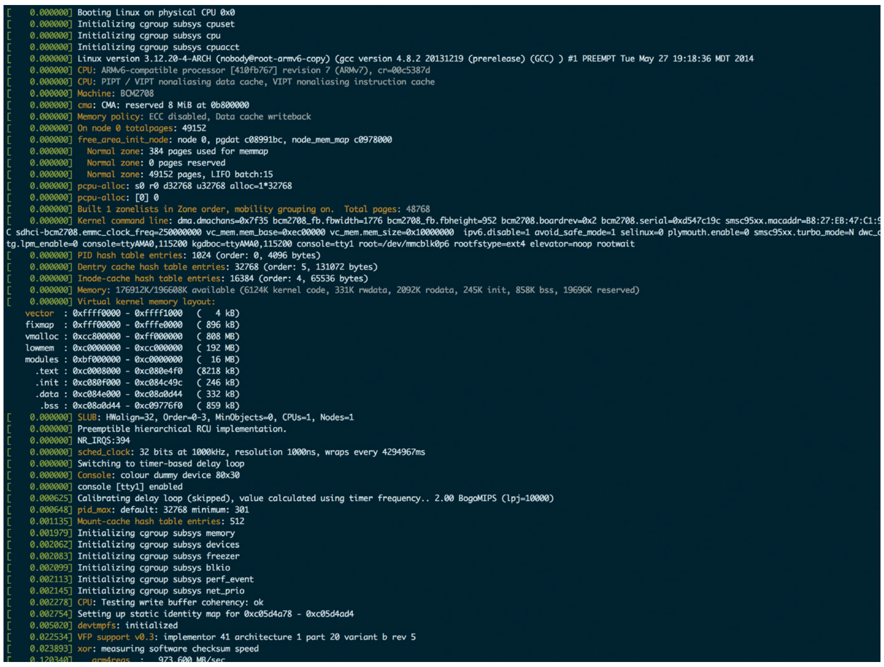
There are modern concessions – you can mount MS-DOS formatted drives and USB sticks, and networking works out of the box, and there's even an app store. But for most of us, RISC OS feels like landing on an alien planet. Which is an excellent learning experience, because there's a refreshing world of modal window constraints, dynamic resizing, saving files, filenames and file management to learn about. And while there are too few Linux/open source apps, there are plenty of other things to discover, and you'll find yourself rebooting to Linux and wondering where all that performance has gone.


We never did find out how to close the windows on the applications we were opening.

**VERDICT**
By far the fastest OS, but ultimately more a curiosity than an alternative to Linux.
★★★☆☆

# Arch Linux Arm

## Who needs audio, graphics or a configuration tool?



```
[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Initializing cgroup subsys cpuset
[    0.000000] Initializing cgroup subsys cpu
[    0.000000] Initializing cgroup subsys cpuacct
[    0.000000] Linux version 3.12.20-4-ARCH (nobody@root-armv6-copy) (gcc version 4.8.2 20131219 (prerelease) (GCC) ) #1 PREEMPT Tue May 27 19:18:36 MDT 2014
[    0.000000] CPU: ARMv6-compatible processor [410fb767] revision 7 (ARMv7), cr=00c5387d
[    0.000000] CPU: PIPT / VIPT nonaliasing data cache, VIPT nonaliasing instruction cache
[    0.000000] Machine: BCM2708
[    0.000000] cma: CMA: reserved 8 MiB at 0b800000
[    0.000000] Memory policy: ECC disabled, Data cache writeback
[    0.000000] On node 0 totalpages: 49152
[    0.000000] free_area_init_node: node 0, pgdat c08991bc, node_mem_map c0978000
[    0.000000]   Normal zone: 384 pages used for memmap
[    0.000000]   Normal zone: 0 pages reserved
[    0.000000]   Normal zone: 49152 pages, LIFO batch:15
[    0.000000] pcpu-alloc: s0 r0 d32768 u32768 alloc=1*32768
[    0.000000] pcpu-alloc: [0] 0
[    0.000000] Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 48768
[    0.000000] Kernel command line: dma.dmachans=0x7f35 bcm2708_fb.fbwidth=1776 bcm2708_fb.fbheight=952 bcm2708.boardrev=0x2 bcm2708.serial=0xd547c19c smsc95xx.macaddr=B8:27:EB:47:C1:9
C sdhci-bcm2708.emmc_clock_freq=250000000 vc_mem.mem_base=0xec00000 vc_mem.mem_size=0x10000000  ipv6.disable=1 avoid_safe_mode=1 selinux=0 plymouth.enable=0 smsc95xx.turbo_mode=N dwc_o
tg.lpm_enable=0 console=ttyAMA0,115200 kgdboc=ttyAMA0,115200 console=tty1 root=/dev/mmcblk0p6 rootfstype=ext4 elevator=noop rootwait
[    0.000000] PID hash table entries: 1024 (order: 0, 4096 bytes)
[    0.000000] Dentry cache hash table entries: 32768 (order: 5, 131072 bytes)
[    0.000000] Inode-cache hash table entries: 16384 (order: 4, 65536 bytes)
[    0.000000] Memory: 176912K/196608K available (6124K kernel code, 331K rwdata, 2092K rodata, 245K init, 858K bss, 19696K reserved)
[    0.000000] Virtual kernel memory layout:
    vector  : 0xffff0000 - 0xffff1000   (   4 kB)
    fixmap  : 0xfff00000 - 0xfffe0000   ( 896 kB)
    vmalloc : 0xcc800000 - 0xff000000   ( 808 MB)
    lowmem  : 0xc0000000 - 0xcc000000   ( 192 MB)
    modules : 0xbf000000 - 0xc0000000   (  16 MB)
      .text : 0xc0008000 - 0xc080e4f0   (8218 kB)
      .init : 0xc080f000 - 0xc084c49c   ( 246 kB)
      .data : 0xc084e000 - 0xc08a0d44   ( 332 kB)
       .bss : 0xc08a0d44 - 0xc09776f0   ( 859 kB)
[    0.000000] SLUB: HWalign=32, Order=0-3, MinObjects=0, CPUs=1, Nodes=1
[    0.000000] Preemptible hierarchical RCU implementation.
[    0.000000] NR_IRQS:394
[    0.000000] sched_clock: 32 bits at 1000kHz, resolution 1000ns, wraps every 4294967ms
[    0.000000] Switching to timer-based delay loop
[    0.000000] Console: colour dummy device 80x30
[    0.000000] console [tty1] enabled
[    0.000625] Calibrating delay loop (skipped), value calculated using timer frequency.. 2.00 BogoMIPS (lpj=10000)
[    0.000648] pid_max: default: 32768 minimum: 301
[    0.001135] Mount-cache hash table entries: 512
[    0.001979] Initializing cgroup subsys memory
[    0.002062] Initializing cgroup subsys devices
[    0.002083] Initializing cgroup subsys freezer
[    0.002099] Initializing cgroup subsys blkio
[    0.002113] Initializing cgroup subsys perf_event
[    0.002145] Initializing cgroup subsys net_prio
[    0.002278] CPU: Testing write buffer coherency: ok
[    0.002754] Setting up static identity map for 0xc05d4a78 - 0xc05d4ad4
[    0.005020] devtmpfs: initialized
[    0.022534] VFP support v0.3: implementor 41 architecture 1 part 20 variant b rev 5
[    0.023893] xor: measuring software checksum speed
[    0.120340]    arm4regs  :   973.600 MB/sec
```

It would be nice to have a working desktop out of the box, but that's not the Arch way.

Despite its reputation for being difficult (and the Arch chattering classes will hate us for saying that), when someone else has gone to the trouble of tidily packaging the operating system up for your specific hardware, it's almost as easy to use Arch as it is to use Raspbian. Which is exactly what's happened with Arch for the Pi. With nothing more than a simple copy to your SD card, you've got a fully functional Arch installation ready and waiting for anything you want to throw at it.

Arch is a blank slate for your own projects or for building your own perfect environment, and you'll need to install everything else yourself. It's good and proper that the Pi version takes the same principle.

### The font of learning

One concession to usability we were pleasantly surprised to find was the SSH server up and running, which means you can continue to configure your Raspberry Pi remotely. This being Arch, the amount of stuff that can be installed via the **pacman -S** command is colossal, although it can't compete with Raspbian unless you add the Arch User Repository. We've also found that installing the AUR build environment (which is required when creating packages for the the ARM architecture) is the best way to install and keep up to date with the latest package developments for any of the distributions we looked at. The way you can pull packages out of the build system, make your own modifications or patches, and then run the binaries without worrying about dependencies is a significant time saver, especially when the whole system boots so quickly.

This makes Arch perhaps perfect for those developing their own embedded projects, or who need the greatest possible breadth of potential packages to install from. Most of this is covered in Arch's wiki page for the Pi. Reading the wiki and making these changes yourself forces you to learn about the system you're creating. Doing that from a Raspberry Pi is a natural progression from the open nature of the hardware, just at a lower level. You end up understanding exactly how the system is running, and that's something you can't easily achieve from any other distribution.

**VERDICT**

This is the perfect way to get started with Arch, and a great way to learn about the Linux OS.

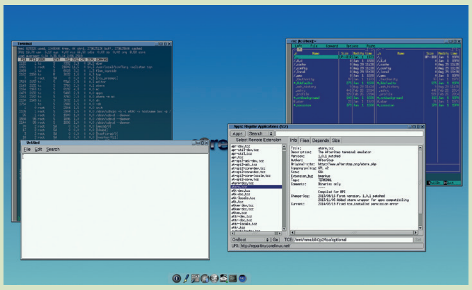
★★★★☆

## More obscure alternatives

In this group test, we've looked at the six distributions you can easily install from the *Noobs* installer. These are the same distros you can download from the main site and install manually. But by our reckoning, there are over 40 to choose from. Arch is a great minimal distribution, for example, but it still weighs in at around 500MB, and that's before you install any of the further packages you're likely to need. If you want to stick with Raspbian, one alternative is Minibian. It's close to being a 200MB download and uses the same servers and packages as Raspbian so it can easily be augmented with whatever additional software you need.

By far the smallest we've found is PiCore, a version of Tiny Core Linux built for ARM. The download image with SSH running for headless installations is a mere 18.6 MB, and adding a graphical environment only adds 14.6MB – just less than 40MB when uncompressed. It also leaves you with an impressive 114MB of RAM, but you'll need to install everything else, as the default installation doesn't even include a web browser (although it does include a package manager of sorts).

You may also want to keep an eye on the Kano OS project, which promises an Elementary OS-style makeover to the Raspberry Pi desktop. But our favourite, though sadly a touch impractical, is the Commodore Pi Project. This turns your Pi into a Commodore 64 by using the *Comeback64* emulator as its kernel, albeit a Commodore 64 with Ethernet and access to more RAM, USB and the GPIO pins. At the time of writing, the only video output working is the composite, which makes it truly old school (and already out of date if you've got a Model B+).



PiCore doesn't have much functionality, but it's perfect if you've only got a small SD card.

# OpenELEC vs OSMC/Raspbmc

Video may have killed the radio star, but it's the making of these two great systems.

Almost by complete surprise, one area of great success for the Raspberry Pi has been in the realm of media playback. Its CPU isn't powerful, but it is optimised for audio and video, making it punch far above its weight when it comes to playback. It comes with HDMI by default and includes the audio within the HDMI connection. And it's also cheap and almost completely open.

All of which perhaps explains why there are two excellent distributions for the Raspberry Pi designed to make it work as a media centre – OpenELEC and Raspbmc. Because *XBMC* is changing its name, so too is Raspbmc, with its new project name being OSMC. But because nearly all references within the distribution and online still use Raspbmc, we're going to use this too. We installed OpenELEC first and were impressed by the way it automatically expanded the filesystem when first booted to take advantage of as much space on your SD card as possible. It then reboots and launches *XBMC 13.1* with the OpenELEC setup wizard. This asks you a few simple questions, such as for a hostname, sets up networking and enables both SSH and Samba, which is incredibly useful.

We were able to test the update between versions 4.05 and 4.07, which downloaded, extracted and applied itself perfectly, although we'd appreciate a little more feedback to stop us restarting the system while an update is being applied.

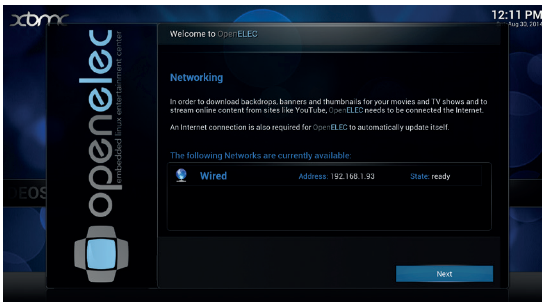
Raspbmc/OSMC is quite different. You have a choice of installing from either a minimal network image or a fatter (1.2GB) image that requires less to be downloaded. Because we're professionals, we tried them both, and as a result we'd recommend going with the network install (unless your Pi is connected to a 28.8 baud modem). The package download that's part of the installation takes only 5–10 minutes, but the entire installation takes a lot longer. Both versions still download, unpack and install new kernels, resize partition tables, extract updates and post chirpy updates to the blue and grey display.

"When connected to local storage, playback from both systems is excellent."

Raspmc also has its own settings add-on, and while not as polished in appearance, it offers much more detailed control over your system than the default in OpenELEC. You can overclock various parts of your Pi, for example, or enable the Pi camera module to take intermittent photos. You can also configure a GPIO infrared receiver for a remote control, allow updates (even from a nightly build) and there's a web browser too. By default, *XBMC 13.2 Git* was installed, which perhaps explains the slightly smoother performance in Raspmc.

When connected to local storage, playback from both systems is excellent. The system info pages report that OpenELEC ran 1–2 frames per second slower, which we wouldn't worry about, but Raspmc had the edge when it came to accessing media, starting playback and screen updates. This surprised us, considering the more bespoke and minimal strategy taken by the OpenELEC team. However, both had problems when we connected *XBMC* to our *TVheadend* back-end and attempted to watch live or recorded television. The answer for both is to enable overclocking, which we ran without adding any instability, and keeping things up to date.

OpenELEC is proudly built from the kernel up to only include what is required, unlike Raspbmc, which is a minimal build on top of Debian. But a Debian foundation could also be an advantage, as it enables you to **apt-get install** anything else. These two are so very close otherwise. OpenELEC is more polished, and would be our choice for an installation where you're not perhaps local to fix things. But for us, Raspbmc wins the comparison thanks to its geeky settings add-on and marginally better performance.



Networking

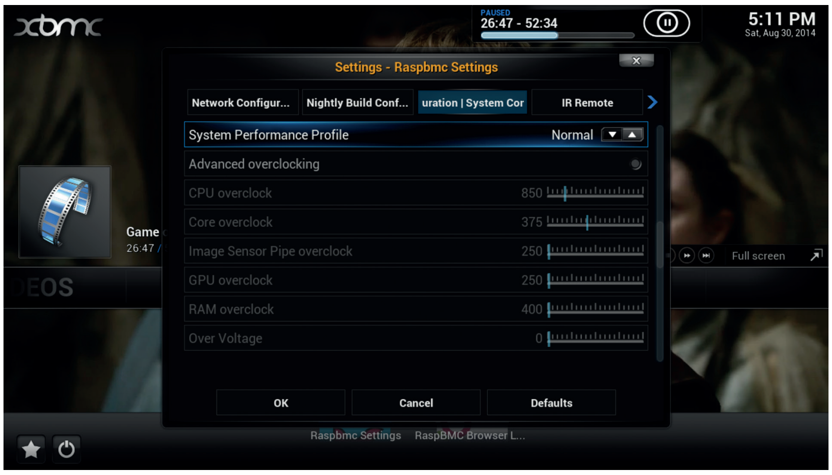
In order to download backdrops, banners and thumbnails for your movies and TV shows and to stream online content from sites like YouTube, OpenELEC needs to be connected the Internet.

An Internet connection is also required for OpenELEC to automatically update itself.

The following Networks are currently available:

Wired Address: 192.168.1.93 State: ready

OpenELEC is a wonderfully polished media centre that's a great choice for friends and family.



Settings - Raspbmc Settings

System Performance Profile Normal
Advanced overclocking
CPU overclock 850
Core overclock 375
Image Sensor Pipe overclock 250
GPU overclock 250
RAM overclock 400
Over Voltage 0

Raspbmc, now called OSMC, enables you to overclock your Pi from within *XBMC*, which we'd recommend as it vastly improved performance.

**VERDICT**

**Raspbmc/OSMC:** A little rough around the edges, but the hacker's choice.
★★★★★

**OpenELEC:** Without doubt our media player setup of choice for non-Linux users.
★★★★☆

# OUR VERDICT

## Raspberry Pi distributions

The task of choosing a winner in a group test where many of the distributions are trying to do different things is a tough one. We'd rather recommend that you install them all, because that's what the Raspberry Pi is about. And because SD cards are becoming increasingly affordable, there's no reason why you can't have more than one installed and switch between them for whatever task is at hand – use one distribution for watching a film or listening to music and another for building your next hardware project, for example.

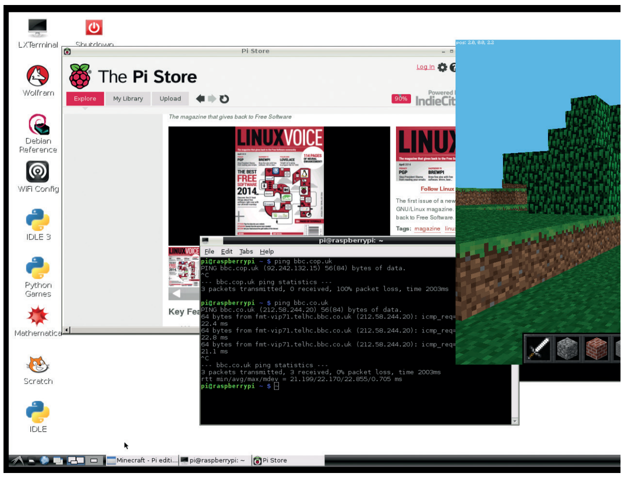
But the distribution we've chosen is perhaps no surprise. It's the best at providing the broadest range of possibilities and starting points and it's as close to being a standard operating system for the Pi as you can get. And that's Raspbian. In this case, we think it's important that there's a standard because it enables new users in particular to get the most from the widest range of tutorials and support, and thanks to Raspbian being built on Debian, they're acquiring some excellent general Linux knowledge at the same time.

For the sake of media playback, we chose the closely related OSMC/Raspbmc, but it could just have easily been OpenELEC, and we're going to switch between the two over the next few months to see how they both progress. Arch's Pi incarnation also surprised us, and it's our chosen platform for any new projects we embark upon. It's also the distribution we'd choose if you want to use low-latency audio, for example, or run an emulator. This is followed by Pidora, another excellent choice and worth trying purely because it's Fedora, and RISC OS, which is itself a fascinating operating system. And there are many, many more to try.

So really, even though we've chosen one winner, this should just be the beginning of the adventure. Go forth, and make the most of your SD card's spare capacity to broaden your Pi horizons. LV

"Raspbian provides the broadest range of possibilities and starting points."

| Distro Name | Boot time (s) | Root size (GB) | Free memory (MB) | Packages |
|---|---|---|---|---|
| Arch CLI (no AUR) | 11 | 0.456 | 105 | 11000 |
| OpenELEC | 43 | 0.995 | 24 | 0 |
| Pidora CLI | 16 | 2.3 | 72 | 31706 |
| Pidora XFCE | 63 | 2.3 | 53 | 31706 |
| Raspbian CLI | 30 | 2.4 | 137 | 37246 |
| Raspbian LXDE | 56 | 2.4 | 64 | 37246 |
| Raspbmc/OSMC | 56 | 0.890 | 10 | 37294 |
| RISC OS | 17 | 0.277 | 205 | 150 |



```
pi@raspberrypi ~ $ ping bbc.cop.uk
PING bbc.cop.uk (92.242.132.15) 56(84) bytes of data.
^C
--- bbc.cop.uk ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2003ms

pi@raspberrypi ~ $ ping bbc.co.uk
PING bbc.co.uk (212.58.244.20) 56(84) bytes of data.
64 bytes from fmt-vip71.telhc.bbc.co.uk (212.58.244.20): icmp_req=
22.4 ms
64 bytes from fmt-vip71.telhc.bbc.co.uk (212.58.244.20): icmp_req=
22.8 ms
64 bytes from fmt-vip71.telhc.bbc.co.uk (212.58.244.20): icmp_req=
21.1 ms
^C
--- bbc.co.uk ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 21.199/22.170/22.855/0.705 ms
pi@raspberrypi ~ $
```

Raspbian's two killer features are its support (it's the Pi Foundation's official distro) and Debian's software repositories.

### 1st Raspbian

**Licence** Mostly GPL **Version** 20/06/2014

**www.raspbian.org**
It's the sensible choice, and also the easiest to use and the best for any potential project.

### 2nd OSMC/Raspbmc

**Licence** Open Source **Version** June 2014

**www.raspbmc.com**
The Pi is perfect for media playback, and OSMC is the best distro we're found for media.

### 3rd Arch Linux

**Licence** Open Source **Version** June 2014

**http://archlinuxarm.org**
Everyone should give Arch Linux a go at least once, and this is the best way to get started with it.

### 4th OpenElec

**Licence** GPLv2 **Version** 4.07

**http://openelec.tv**
It's only going to take one update, and OpenELEC could easily leapfrog into position two.

### 5th Pidora

**Licence** Open Source **Version** 20

**http://pidora.ca**
It's a little unfair this comes fifth, as it's still an excellent option, and the only one if you love RPMs.

### 6th RISC OS

**Licence** Non Open Source **Version** RC12a

**www.riscosopen.org**
The fastest OS in last position? This is mainly because of the licence and the lack of free software.

# SUBSCRIBE

shop.linuxvoice.com

## Introducing Linux Voice, the magazine that:

- Gives 50% of its profits back to Free Software
- 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 and we will refund you for all unmailed issues.

# NEXT MONTH IN LINUX VOICE

ON SALE THURSDAY 23 OCTOBER

THE BEST FREE SOFTWARE KNOWN TO HUMANITY

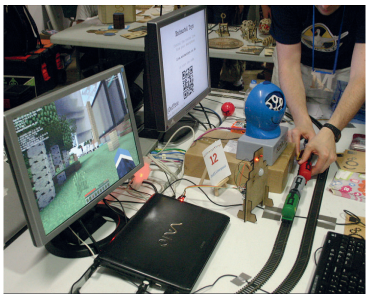
## KILLER APPS

If you only ever use the standard software in your distro's menus, you're missing out big time – join us on a journey through the finest free software known to humanity.

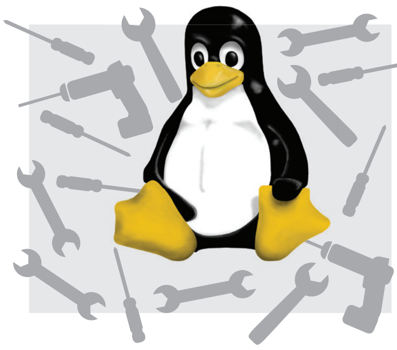
## EVEN MORE AWESOME!

LaTeX

**Latex**
The layout tool of choice for scientists, boffins and other brainy characters, *Latex* used to be dauntingly difficult – until we got hold of it!



**Get your hack on**
Be inspired by the clever, creative things that people just like you are doing right now with free software, exactly like we promised last issue.



**Tune your kernel**
Delve into the workings of your Linux kernel to unlock features and performance known only to the elite. And us, obviously.

## LINUX VOICE IS BROUGHT TO YOU BY

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

**Editorial consultant** Nick Veitch
*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

ISSN 2054-3778

Subscribe: **shop.linuxvoice.com**
*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

Dive under the skin of your Linux system to find out what really makes it tick.

## Filesystem: what's going on?

Take a programmer's-eye view of the Linux filesystem.

Over the last three months our look at core Linux technology has focussed mostly on inter-process communications – pipes and sockets. This month we're going to turn our attention to the filesystem. My interest here is not about how to access and manage files from the command line (**ls**, **mv**, **rm**, **cp**, **chmod**... that kind of thing). I'm assuming you know all that. Rather, I want to take you behind the scenes of the filesystem and view it through the eyes of a programmer.

The lowest level at which you can read and write files is by using the four system calls **open()**, **read()**, **write()** and **close()**. Let's dive straight in with an example. This simple file copy program is written in C:

```
1. #include <fcntl.h>
2. #define BSIZE 1024
3.
4. void main()
5. {
6.   int fin, fout;
7.   char buf[BSIZE];
8.   int count;
9.
10.   fin  = open("foo", O_RDONLY);
11.   fout = open("bar", O_WRONLY | O_CREAT, 0644);
12.
13.   while ((count = read(fin, buf, BSIZE)) > 0)
14.     write(fout, buf, count);
15.
16.   close(fin);
17.   close(fout);
18. }
```

Down at the system call level, file descriptors (or file handles – call them what you will) are plain integers. We declare two of them (one for input, one for output) at line 6. We allocate a modest buffer at line 7; this will be used to store the data as it is being copied across. At lines 10 and 11 we open our input and output files. In each case we get back descriptors that refer to the open files. For simplicity we've just hard-coded the filenames here; more realistically, you'd take them from the command line. The parameters passed at line 11 say that we want to write to the file and that we want to create it if it doesn't exist. The mysterious octal value **0644** specifies the permissions that will be assigned to the file as it is created. You may recognise them more easily written as **rw-r--r--**. Notice that you don't get to specify the owner of the file – it will be owned by whoever runs the program. You don't get a choice.

### Coding back to front

All the real work happens in the loop at lines 13 and 14, and there's a lot packed into these two lines of code. Line 13 needs reading 'inside-out'; it goes something like this: Read the next BSIZE bytes from the input file into the buffer. Record the number of bytes you read in the variable **count**. Test the value of **count**: if it's greater than zero, write however many bytes you got back out to the output file (line 14). To illustrate how this works, suppose the input file was 2500 bytes long. Then line 13 would execute 4 times, returning count values of 1024, 1024, 452 and 0. The zero means we've reached the end of the file. This 'perform an action, capture the result, and test it' is a common idiom in C; indeed, any C programmer worth his salt hides all the really important parts of his programs inside the test predicates for **if()** and **while()** loops in this way.

After we fall out of the loop (line 15) we are careful to close both file descriptors. This will ensure that any data buffered by the kernel is actually written to the disk. In this example the progam terminates immediately afterwards and any open descriptors will be implicitly closed. But if the program went on to process lots of other files we would eventually run out of file descriptors if we failed to close the ones we'd finished with.

Now I realise that some of you may think that this system-level code looks like awfully hard work. Well, maybe it's because I was weaned on a diet of assembly languages as a youngster, but I actually quite enjoy programming at this level. Short of micro-miniaturising yourself and crawling out over

"Short of crawling over the disk with a tiny magnet, this is as close as you can get to the metal."

### A Ken Thompson quote

There was originally a system call named **creat()** that created a new file. Indeed there still is, but it's seldom used since you don't usually want to create a file unless you're about to write to it, and files can be created by the **open()** call, as our file copy example shows. But there's a nice story about **creat**. Apparently Ken Thompson was once asked what he would do differently if he were redesigning the Unix system. His reply: "I'd spell **creat** with an e". (See *The Unix programming environment* by Kernighan and Pike, p204). The implication being, of course, that he'd got everything else right.

the disk's surface with a tiny magnet, this is the closest you can get to the metal when it comes to file I/O.

## Moving up a level

Let's move up a level and re-write the program using the standard I/O library instead of direct system calls:

```
#include <stdio.h>
#define BSIZE 1024

void main()
{
 FILE *fin, *fout; /* Input and output handles */
 char buf[BSIZE];
 int count;

 fin = fopen("foo", "r");
 fout = fopen("bar", "w");

 while ((count = fread(buf, 1, BSIZE, fin)) > 0)
  fwrite(buf, 1, count, fout);

 fclose(fin);
 fclose(fout);
}
```

It doesn't look too much different, does it? File descriptors are now of type **FILE *** instead of just integers, and the calls are renamed – **open()** becomes **fopen()** and so on. But there's an important distinction. The first program used Unix-specific calls; the second uses routines from the standard I/O library, so it should run anywhere that C is supported.

The I/O calls we've just seen – **read()**, **write()**, **fread()** and **fwrite()** – just do binary I/O. There's no sort of format conversion; they just shovel bytes between a file and an in-memory buffer. In contrast, **fprintf()** does formatted output of strings and numeric data, something like this:

```
fprintf(fout, "Answer is %f\n", 22.0/7.0);
```

## Random access

By default, the contents of a file are read sequentially. There's a "file position pointer" maintained for each open file, which points to a specific byte offset within the file and determines where the next read or write will start. If I read 1024 bytes, the pointer advances by that much so that the next read continues where the last left off. Our file copy program relies on this behaviour for both the input and output files.

However, it's possible to explicitly manage this file position pointer, moving it to any desired position within the file. This gives us 'random access', as opposed to 'sequential access', into the data. (The use of the word 'random' here has always struck me as rather odd. It shows up again in the common abbreviation RAM – Random Access Memory – and seems to suggest that we have no control over which piece of the data we actually get. But I digress.)Here's an example that swaps the first and last lines in a text file. I confess it's slightly contrived; in particular it assumes that the first and last lines are the same length. But it illustrates random access quite well. This example is in PHP, though since PHP is just providing its own language binding to the same standard I/O library, the code would not look that much different in C:

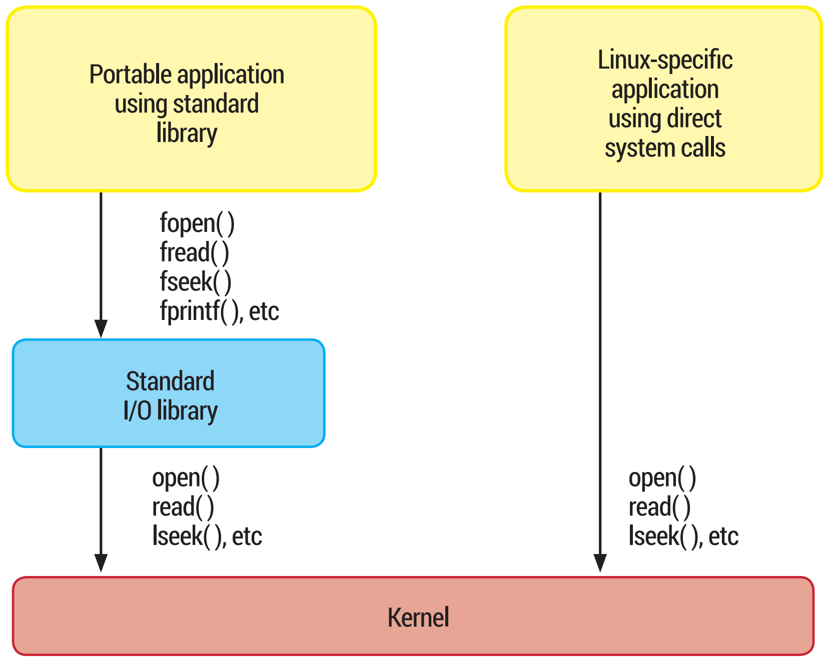
```
1. #!/usr/bin/php
2. <?php
3.   $f = fopen("foo", "r+");
4.   /* walk to the first newline */
5.   while (fread($f, 1) != "\n") ;
6.
7.  /* get current file position */
8.   $n = ftell($f);
9.
10.   /* Read and save the first line */
11.   rewind($f);
12.   $alpha = fread($f, $n);
13.
14.   /* Read and save the last line */
15.   fseek($f, -$n, SEEK_END);
16.   $omega = fread($f, $n);
17.
18.   /* Replace the first line */
19.   fseek($f, 0, SEEK_SET);
20.   fwrite($f, $omega, $n);
21.
22.   /* Replace the last line */
23.   fseek($f, -$n, SEEK_END);
24.   fwrite($f, $alpha, $n);
25.   fclose($f);
26. ?>
```

Here's the scoop. We open the file at line 3. The parameter **r+** is important – it says that we want to both read and write the file. The loop at line 5 (with an empty body) just walks along the file a byte at a time until we reach the first newline character. We are trying to figure out how long the line is. The **ftell()** call at line 8 gets the current file pointer position; this gives us the line length. Line 11 resets the file position pointer to the beginning. The call

```
fseek($f, 0, SEEK_SET)
```

would do the same. Then at line 12 we re-read that first line all in one go, saving it for later. Line 15 is interesting. It positions the file pointer one line before the end of the file. (This is where our assumption that the first and last lines are the same length kicks in.) At line 16 we read in that last line. At line 19 we rewind to the beginning of the file again then overwrite the first line of text.

### Portability and the standard I/O library



Applications can choose to access files through the Standard I/O library, or use direct system calls .

### mmap

The **mmap()** system call provides a very different approach to random access into a file's data. It allows a file's contents to be mapped into the address space of a process and accessed like an array. Random access is achieved simply by indexing into the array. The **mmap** call itself is a little complicated, but if you're looking for an efficient way to dive into a file, **mmap** may be worth a look.

Finally, at lines 23 and 24 we scoot along to the start of the last line of the file and overwrite that, too.

Well, that's a little tricky to follow, so I've drawn a diagram that might help (see below). And if you want to explore this in more detail, the man page for **fseek** will show you the C language bindings for these functions, or browse to **http://php.net/manual/en/function.fseek.php to see the PHP bindings**.

## Listing directories, deleting files

So far we've concentrated on accessing the data within a file, with code that does things broadly equivalent to commands like **cat** and **cp**. Let's shift focus a little and look at the management of the filesystem itself; something more analogous to commands like **cd**, **ls**, and **rm**. Here's a program that will delete all the files in a directory (passed as a command line argument). To add variety, this one's in Perl; it even has some error checking built in!

```
1. #!/usr/bin/perl
2.
3. if (@ARGV != 1) {
4.   warn "usage: empty dirname\n";
5.   exit(1);
6. }
7.
8. if (!chdir($ARGV[0])) {
9.   warn "$ARGV[0]: $!\n";
10.   exit(1);
11. }
12.
13. opendir($d, ".");
14.
15. foreach $info (readdir($d)) {
16.   if ($info ne "." && $info ne "..") {
17.     print "removing $info\n";
18.     if (unlink($info) != 1) {
19.       warn "$info: $!\n";
20.       exit(2);
21.     }
22.   }
23. }
```

Let's talk through this. Lines 3–6 verify that the user provided a command-line argument, printing an error message and bailing out if not. Lines 8–11 change into the directory specified on the command line (equivalent to **cd** in a shell script), printing an error if this fails. Line 13 opens the directory; the handle is returned in **$d**. Line 15 is the start of a loop, calling **readdir()** repeatedly to enumerate the files in the directory. There is an explicit check at line 16 to ignore the entries **.** and **..**; otherwise the file is deleted (unlinked) at line 18. Notice that the program will fail ungracefully if there's a subdirectory in the directory you're emptying. Do be careful if you run this example – it really will remove all the files in the directory you specify, so beware!

My reason for providing examples in different languages is not just to add variety, but to make the point that although different languages have different syntax, they are all providing language bindings to the same library routines – in this case **chdir()**, **opendir()**, **readdir()** and **unlink()**.

## Everything looks like a file

As we reach the end of this discussion we're in a good position to answer the question "what is a file?" Well, the traditional answer is that it's information stored on a disk, referenced by a name. But there's a broader view... anything that responds to the classic system calls such as **open()**, **read()** and **write()** in the appropriate way is going to look like a file, and can be accessed by the usual command line tools like **cat** or **cp**. This perhaps makes a little more sense of the 'files' in the procfs and sysfs virtual filesystems, usually mounted onto **/proc** and **/sys**. These files are purely a figment of the kernel's imagination, providing a view from userspace into internal kernel data. For example, the following command:

```
$ cat /proc/cpuinfo
```

will provide details of the kernel's view of the processor on which it's running. Most parts of these filesystems are read-only – you can't upgrade your processor by writing to **/proc/cpuinfo** or get more memory by writing to **/proc/meminfo**. But some parameters can be tweaked by writing to the appropriate 'file'. A classic example is **/proc/sys/net/ipv4/ip_forward**, which determines whether the Linux kernel will forward (route) IP traffic. By default this is disabled, (zero) as you'll see if you examine the file:

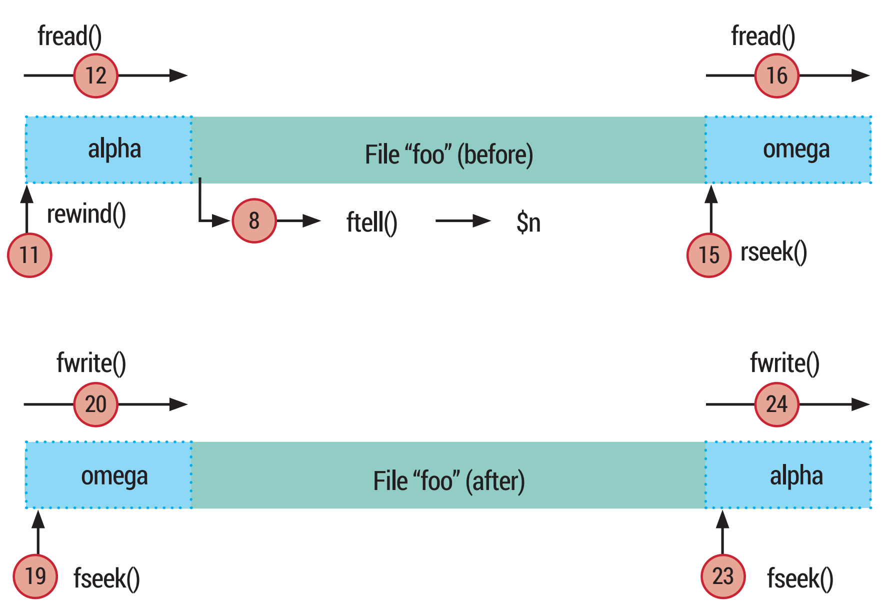
```
$ cat /proc/sys/net/ipv4/ip_forward
0
```

but you can enable it by writing to the 'file' (you'll need to do this as root):

```
# echo 1 > /proc/sys/net/ipv4/ip_forward
```

There are lots more parameters you can interrogate and adjust in this way; my purpose here is not to survey them all but simply to point out that we are able to treat these things like files because they respond to the file I/O system calls in the usual way.

### Swapping records in a file



Linux supports random access into a file's data. Numbered circles correspond to numbered code lines within the main text.

## How to become invisible

Would you like to learn how to write to a file that has no name from a program that doesn't exist? Here's how! There's a well-known (but slightly weird) feature of Linux that if a program opens a file then deletes it (keeping it open) the file will continue to exist. It will have a valid inode but no entry in the filesystem. Here's a program that does exactly that (this one's in C again):

```
1. #include <fcntl.h>
2.
3. main()
4. {
5.   int fout;
6.   char buf[10];
7.   fout = open("/tmp/topsecret", O_WRONLY | O_
CREAT, 0600);
8.   unlink("/tmp/topsecret");
9.   write(fout, "attack at dawn\n", 16);
10.   pause();
11. }
```

The **pause()** at line 10 is there simply to keep the process alive.

To compile this program, place the code into a file called **secret.c** and compile it with:

```
$ gcc -o secret secret.c
```

If we run this program with the **unlink()** call at line 8 commented out, we can of course list and examine the output file in the usual way:

```
$ ./secret &
$ ls -l /tmp/topsecret
-rw------- 1 chris chris 16 Aug  6 15:06 /tmp/topsecret
$ cat /tmp/topsecret
attack at dawn
```

But if we re-run it with line 8 in place, things get more interesting. There will be no entry in the filesystem for **/tmp/topsecret**. It won't show up on the output of **ls** and you certainly can't examine it with **cat**.

```
$ ls -l /tmp/topsecret
ls: cannot access /tmp/topsecret: No such file or
directory
```

We can even delete the executable:

```
$ rm secret
```

Now, neither the file we're writing nor the program that's writing it has an entry in the filesystem. Is this weird or what? And why do we care? Well, let's pin on our "Paranoid About Security" badges and imagine that a hacker of evil intent has managed to plant a program on our machine that is collecting important information in a file that it later intends to transmit back to the bad guy. Using this trick, our villain remains pretty well hidden. But not entirely. We can ask **lsof** (my command of the month in LV005) to show unlinked files like this:

```
$ sudo lsof +L1
secret   8632 chris   3w   REG   8,1     16    0
1573121 /tmp/topsecret (deleted)
```

The option **+L1** tells **lsof** to only show files that have a link count less than 1. If you run this command you will almost certainly see lines of output in addition to the one shown here from programs like **init** (among others).

OK, so we have some evidence that the file still exists. From this output we know its size (16 bytes) and we know the PID of the process that has it open (8632). But given that it has no name, can we see its contents? It turns out we can! You may be aware that **/proc** contains directories named after each process ID, and within each of these is a subdirectory called **fd**. Here you'll find symbolic links (named after the file descriptor) to each file that the process has open. In this case, file descriptor 3 is the one we're interested in:

```
$ cd /proc/8632/fd
$ ls
0  1  2  3
$ cat 3
attack at dawn
```

and – hey presto! – we see the contents of our invisible file.

Similarly, most of the things in **/dev** present a file-like view to userspace. Pseudo-devices like **/dev/null**, **/dev/random**, and **/dev/zero** deliver data streams (or not, in the case of **/dev/null**). Disk partitions have names like **/dev/sda3** (these are linked to more complex names in modern linux kernels) and can be written to like a file, so that a command like:

```
$ echo "Kilroy was here" > /dev/sda3
```

is perfectly legal, though probably not at all a good idea if there is a filesystem on **sda3**.

This "everything looks like a file" view of things, which is such a fundamental part of Linux, provides a very consistent picture of the world, with disk partitions having owners, timestamps and access permissions just like regular files. The only things that aren't part of this world (for reasons I have never really understood) are the network interfaces. There's no **/dev/eth0** for example.

Next month I'm planning to look at the system calls that examine and modify a file's attributes, and to examine the **inotify()** API, which lets you monitor the filesystem for changes. See you then! LV

## Command of the month: dd

My command of the month is **dd**. It's basically a file copy program. A simple invocation is:

```
$ dd if=foo of=bar
```

which copies the file **foo** to **bar**. Of course you could do it more easily with **cp**.

But **dd** supports various conversions that will be applied to the file as it is copied. For example,

```
$ dd if=foo of=bar conv=ucase
```

will convert the file to upper case. Or:

```
$ dd if=foo of=bar conv=swab
```

will swap each pair of bytes in the file (historically useful if you were moving data between "little-endian" and "big-endian" machines).

The **dd** command also lets you control how much data is copied, and in what size chunks. For example:

```
$ dd if=/dev/zero of=zeros bs=1MB count=10
```

copies the pseudo-device **/dev/zero** (an endless source of zeros) into the file **zeros**, copying 1MB (1 million bytes) at a time, and continuing for 10 records. So we end up with a file exactly 10,000,000 bytes long.

Occasionally **dd** is used to image disk partitions. For example,

```
# dd if=/dev/sda3 of=sda3copy
```

will make a direct bit-for-bit copy of a complete disk partition into the file **sda3copy**. Or you can restore a partition by doing it the other way round:

```
# dd if=sda3copy of=/dev/sda3
```

though please don't try this at home, folks, unless you know what you are doing! Also beware that copying disk partitions in this way may not be the most efficient approach, because **dd** will blindly copy the partition byte by byte, whereas tools like *Partimage* and *Clonezilla*, which understand the filesystem structure, will only copy the blocks that are actually in use. This can result in a much smaller image if the file system isn't very full.

The name **dd**, and to some extent its command syntax (which is decidedly not Unix-like) are a reference to an old job control language used on IBM mainframes. Nowadays we take the ease and elegance of the Unix command line for granted. If you think it's arcane, please believe an old-timer: the job control language needed to persuade an IBM mainframe to to anything at all was breathtaking in its obscurity.

# 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...

Programming language

## nuBASIC 1.18

We're spoilt for choice with programming languages on Linux, with every paradigm under the sun represented, and returning to the clumsy spaghetti code of 80s home computers seems bonkers. So we're not advocating that people write large-scale programs in BASIC today. But nuBASIC still fills a niche: for those who fancy a trip down memory lane, for programmers who want to see how a language is implemented (the interpreter is written in C++), and for children looking for an easy path into the world of programming. You could argue that kids are better off learning Python, but the BASIC implementation here actually has elements of structured programming, and it makes it easy to handle keyboard input, graphics and so forth.

nuBASIC is provided in RPM and Deb formats, the latter of which worked perfectly on our Ubuntu 13.10 test box. Source code is also available of course – the main dependency when you're building it is SDL v2. Annoyingly, the packaged version doesn't come with a manual page, nor with any examples, so you have to grab those via **examples_1.13.tar.bz2** from the project's website. And then you might get stuck when trying to run a program; it turns out that you need to use the **-e** flag, otherwise you're dropped into an interactive session. So, run a program like so:

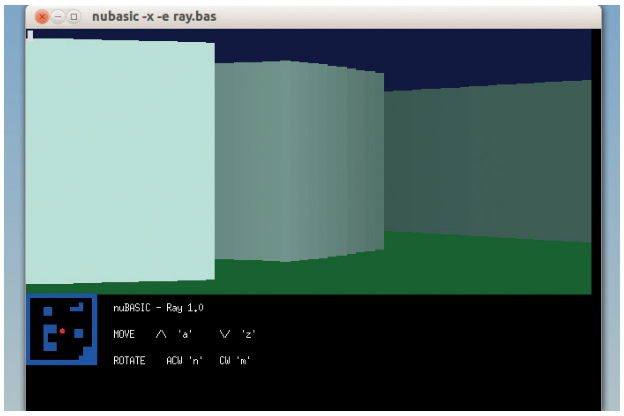
```
nubasic -e breakout3.bas
```

But! There's another hitch: the default window size is too small for many of the supplied examples, so you'll have to resize it before you use the programs.

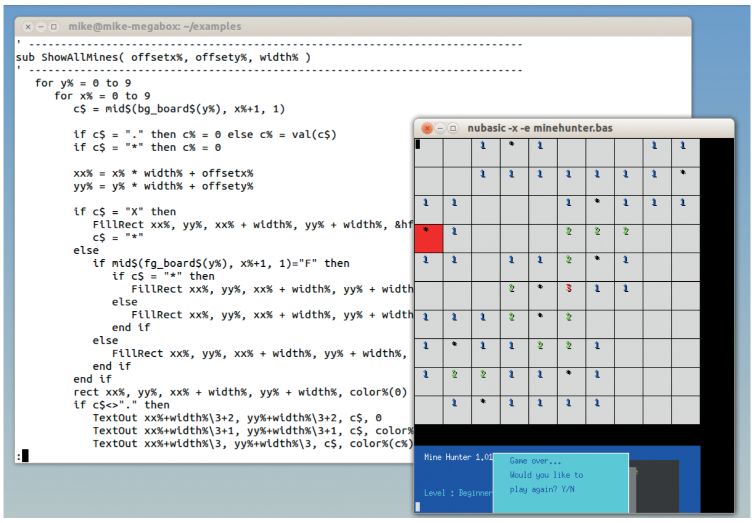
Take a look at the examples to see what the language can do. The examples bundle includes three versions of the classic block-bashing *Breakout* game; the first uses the hideous GOTO-laden line-numbered programming approach of Speccy-era machines, while **breakout2.bas** and **breakout3.bas** demonstrate the interpreter's ability to use more advanced programming methods.

"The BASIC implementation here makes it easy to handle keyboard input, graphics and so forth."

Other examples include **ray.bas**, an impressive (albeit slow) 3D-esque raytracing demo, along with **minehunter.bas**, a clone of the classic *Minesweeper*. The examples show many aspects of the language, from reading keyboard and mouse input to plotting pixels and working with files. nuBASIC is well documented, with an extensive programming guide and language reference explaining the interpreter's capabilities using copious examples.



You can write simple 3D games, as the raytracing demo shows.



```
' ------------------------------------------------------------
sub ShowAllMines( offsetx%, offsety%, width% )
' ------------------------------------------------------------
   for y% = 0 to 9
      for x% = 0 to 9
         c$ = mid$(bg_board$(y%), x%+1, 1)

         if c$ = "." then c% = 0 else c% = val(c$)
         if c$ = "*" then c% = 0

         xx% = x% * width% + offsetx%
         yy% = y% * width% + offsety%

         if c$ = "X" then
            FillRect xx%, yy%, xx% + width%, yy% + width%, &hf
            c$ = "*"
         else
            if mid$(fg_board$(y%), x%+1, 1)="F" then
               if c$ = "*" then
                  FillRect xx%, yy%, xx% + width%, yy% + width
               else
                  FillRect xx%, yy%, xx% + width%, yy% + width
               end if
            else
               FillRect xx%, yy%, xx% + width%, yy% + width%,
            end if
         end if
         rect xx%, yy%, xx% + width%, yy% + width%, color%(0)
         if c$<>"." then
            TextOut xx%+width%\3+2, yy%+width%\3+2, c$, 0
            TextOut xx%+width%\3+1, yy%+width%\3+1, c$, color%
            TextOut xx%+width%\3, yy%+width%\3, c$, color%(c%)
:
```

```
Game over...
Would you like to
play again? Y/N
```

Here's *Minehunter* in action, along with one of the more complicated snippets from its source code.

**PROJECT WEBSITE**
**https://sites.google.com/site/nubasiclanguageinterpreter**

Operating system

## Haiku OS 2014-08-31

Linux on the desktop is a curious beast: there's no single team in charge of it all. We have the kernel hackers working in one group, X being developed by another, the Gnome and KDE coders busy elsewhere, and so forth. Distribution vendors fit it all together, and the end result is a hugely versatile desktop OS.

Now, imagine an OS created from the ground up that focuses entirely on the desktop. Unlike Linux, it doesn't have an interest in also working on big-iron mainframes or postage stamp-sized embedded devices. Everything is developed in unison – the kernel, the graphical layer, the toolkit, the desktop and the core applications. This is Haiku OS, an open source implementation of BeOS, a scorchingly fast multimedia OS that gained some small scale popularity in the late 90s (and became defunct in 2001).

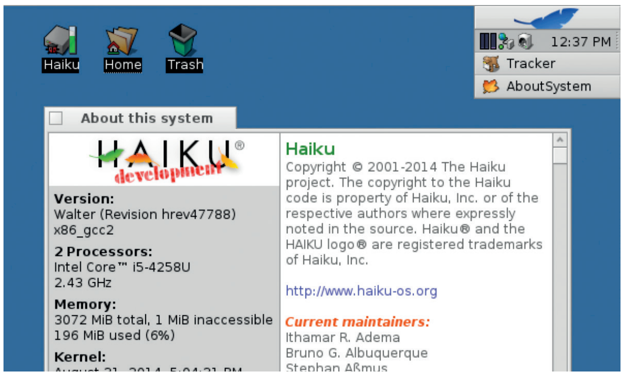
It's been a while since the last alpha release, so we fired up a nightly development snapshot, which is available as a **.vmdk** virtual hard drive file, ready to use in *VirtualBox* or *VMware*. Download the Zip file, extract it and in *VirtualBox*, go to Settings > Storage and choose it as the drive image for your virtual SATA controller. (It's also available in other formats, eg for writing to a USB key – see the bottom of **http://download.haiku-os.org** for more details.) Haiku boots impressively quickly, even inside a virtual machine, and displays a bare desktop that harks back to the days of Windows 98. There's little visual glitz here, as the Haiku team is focused on usability and performance. Click on the leaf icon in the top-right to open the main menu; this includes a number of submenus, such as Applications and Demos, where you can play around with the included software.

### What's in the box?

If you're running in *VirtualBox*, networking should be enabled automatically. *WebPositive* is a *WebKit*-based browser that runs at a decent lick, while additional apps are included for accessing mail (IMAP and POP3) and playing media files. You'll even find a terminal running *Bash*, but note that this is not a Unix-like system. Switch into the **/boot/system** directory and run **ls**, for instance, and you'll see that the filesystem layout is completely different.

Haiku aims to be compatible with the last release of BeOS, although this has meant sticking with *GCC 2* until now – and that version of the compiler is 13 years old. *GCC 4* is available though, for those not interested in backward compatibility. The API is well documented, and if you're a dab hand at C++, it doesn't take much effort to knock together a quick Hello World app. Various third-party applications are available at **www.haikuware.com**, although the selection is very small when compared to the big-name distros.

Haiku's progress has been slow in recent years, but we still cheer it on as an alternative to Linux, especially on older PCs. There's room in the market for a svelte low-latency OS with a razor-sharp focus on desktop computing – especially if it can bring new features to the table.



**Version:**
Walter (Revision hrev47788)
x86_gcc2

**2 Processors:**
Intel Core™ i5-4258U
2.43 GHz

**Memory:**
3072 MiB total, 1 MiB inaccessible
196 MiB used (6%)

**Kernel:**

Haiku
Copyright © 2001-2014 The Haiku project. The copyright to the Haiku code is property of Haiku, Inc. or of the respective authors where expressly noted in the source. Haiku® and the HAIKU logo® are registered trademarks of Haiku, Inc.

http://www.haiku-os.org

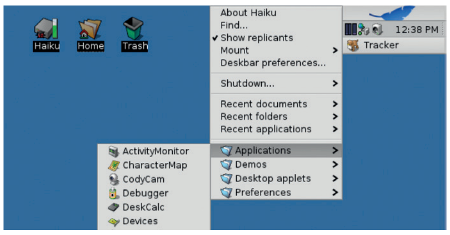
*Current maintainers:*
Ithamar R. Adema
Bruno G. Albuquerque
Stephan Aßmus

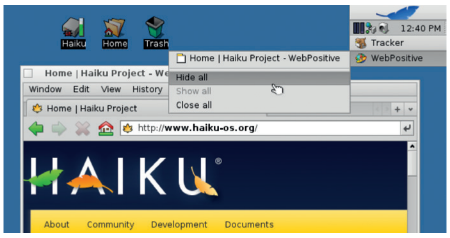
Haiku doesn't sport wobbly windows or fancy drop shadows, but it runs at a blistering pace.

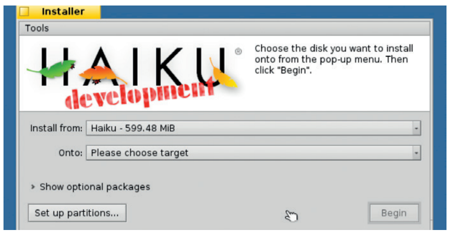
"Haiku is a svelte low-latency operating system with a razor-sharp focus on desktop computing."

**PROJECT WEBSITE**
**www.haiku-os.org**

## How it works: The Haiku desktop







Choose the disk you want to install onto from the pop-up menu. Then click "Begin".

Install from: Haiku - 599.48 MiB

Onto: Please choose target

**1 Boot**
Boot the hard drive image in *VirtualBox* (or the alternative image from a USB key on a real PC) and you'll arrive at the desktop. Click on the leaf icon to explore software.

**2 Run programs**
When you start each program, it will be added underneath the leaf button and system tray in the top-right. This is like a taskbar – click on buttons to close apps (or use the buttons in their titlebars).

**3 Install**
You can perform a native hard drive installation under Applications > Installer. Note that this is still alpha software, so back up important data and don't install it on a production machine!

Video downloader

## youtube-dl 2014.08.29

YouTube might go down in history as the biggest time-waster ever created. Sure, there are some genuinely useful videos on there, but in all honesty we spend 99% of our time there watching cat videos and people playing games that we used to play (but can't be bothered loading up now). It's possible to download videos from YouTube, but some of the browser extensions that do this are rather dodgy, possibly sending your browsing history to unknown third parties.

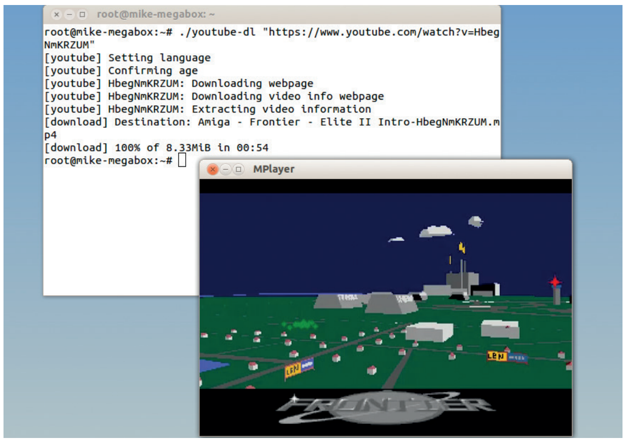
We Linux users have a better solution in the form of **youtube-dl**. This is a (large!) single Python script that takes a URL and spits out a video file. It's remarkably easy to install; just grab the file from the Downloads section of the website, make it executable (eg **chmod +x youtube-dl**") and run it from your home directory like so:

```
./youtube-dl "<URL>"
```

Replace **<URL>** here with the full address of the YouTube video, as displayed in your browser. As you can see in the screenshot, **youtube-dl** grabs the page and parses it for the video content, before downloading the media. In many cases this will be a Flash (**.flv**) file – but some videos are provided in MP4 format. A decent media player like *MPlayer* or *VLC* should be able to handle both formats.

But **youtube-dl** can do a lot more: it can extract the audio from a video and convert it into a different format (providing you have the right tools installed), which is great if you've found a music video and want to keep the song on your MP3/Ogg player. You can ask it to embed subtitles into video files, log in to YouTube using a username and password, and even download adverts, if you feel guilty about not giving enough money to Google.

The program also works with video sites such as Vimeo, Vine and LiveLeak, and because these sites often change their underlying HTML (causing **youtube-dl** to break), you can always upgrade to the latest version in-place with the **-U** flag.



```
root@mike-megabox:~# ./youtube-dl "https://www.youtube.com/watch?v=Hbeg
NmKRZUM"
[youtube] Setting language
[youtube] Confirming age
[youtube] HbegNmKRZUM: Downloading webpage
[youtube] HbegNmKRZUM: Downloading video info webpage
[youtube] HbegNmKRZUM: Extracting video information
[download] Destination: Amiga - Frontier - Elite II Intro-HbegNmKRZUM.m
p4
[download] 100% of 8.33MiB in 00:54
root@mike-megabox:~# 
```

Store videos locally (and avoid dodgy browser plugins) with this handy script.

**PROJECT WEBSITE**
**http://rg3.github.io/youtube-dl**

Lightweight static content web server

## Filed 1.8

Picture the scene: you've resurrected an old PC to see what it's still capable of. You want to share some files over your home network from it via HTTP, so you install *Apache* and... it crawls. You try another web server from the repositories, but it's equally sluggish on such limited hardware. You try yet another, and this time you end up getting bamboozled by its configuration files.

In these cases, you want the simplest, fastest, no-nonsensest HTTP server possible, and *Filed* is just that. It's a single 64k binary, with no configuration file – everything is set at the command line.

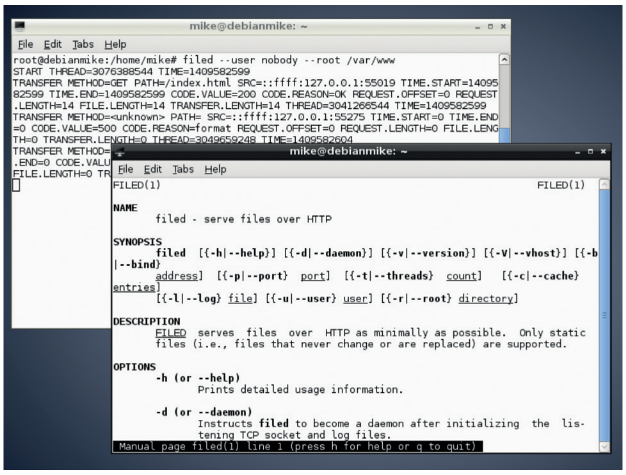
To build it, you'll need *Tcl* installed, and when you run **make** you might see an error message about a missing **mime.types** file. In this case, open the **Makefile** in a text editor and change the **MIMETYPES** line to point to **/etc/mime.types** instead of the default location. Run **make** again, followed by **sudo make install**, and you're ready to go.

By default, *Filed* should be run as the root user, and it serves up your root (**/**) directory. Obviously this isn't very useful, and potentially dangerous; to change the user (via **chroot**) and directory that's served up, run it like so:

```
filed --user nobody --root /var/www
```

*Filed* doesn't generate directory listings and instead attempts to serve up **index.html** by default. To boost performance, *Filed* is multithreaded with every thread serving a single concurrent client. Various extra options are available to bind to a different address or operate on another port, and instead of logging to the terminal you can redirect the output to a file.

"Filed is the simplest, fastest HTTP server possible."



```
root@debianmike:/home/mike# filed --user nobody --root /var/www
START THREAD=3076388544 TIME=1409582599
TRANSFER METHOD=GET PATH=/index.html SRC=::ffff:127.0.0.1:55019 TIME.START=14095
82599 TIME.END=1409582599 CODE.VALUE=200 CODE.REASON=OK REQUEST.OFFSET=0 REQUEST
.LENGTH=14 FILE.LENGTH=14 TRANSFER.LENGTH=14 THREAD=3041266544 TIME=1409582599
TRANSFER METHOD=<unknown> PATH= SRC=::ffff:127.0.0.1:55275 TIME.START=0 TIME.END
=0 CODE.VALUE=500 CODE.REASON=format REQUEST.OFFSET=0 REQUEST.LENGTH=0 FILE.LENG
TH=0 TRANSFER.LENGTH=0 THREAD=3049659248 TIME=1409582604
TRANSFER METHOD=
.END=0 CODE.VALU
FILE.LENGTH=0 TR
```

```
FILED(1)                                                         FILED(1)

NAME
       filed - serve files over HTTP

SYNOPSIS
       filed  [{-h|--help}] [{-d|--daemon}] [{-v|--version}] [{-V|--vhost}] [{-b
|--bind}
       address]  [{-p|--port}  port]  [{-t|--threads}  count]   [{-c|--cache}
entries]
       [{-l|--log} file] [{-u|--user} user] [{-r|--root} directory]

DESCRIPTION
       FILED  serves  files  over  HTTP as minimally as possible.  Only static
       files (i.e., files that never change or are replaced) are supported.

OPTIONS
       -h (or --help)
              Prints detailed usage information.

       -d (or --daemon)
              Instructs filed to become a daemon after initializing  the  lis-
              tening TCP socket and log files.
 Manual page filed(1) line 1 (press h for help or q to quit)
```

Filed's all-caps log format (background terminal) is a bit painful on the eyes, but at least there's plenty of info.

**PROJECT WEBSITE**
**http://filed.rkeene.org/fossil/index**

Book writing assistant

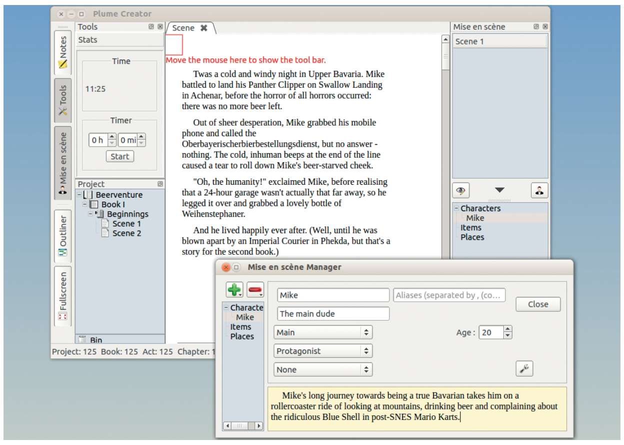
## Plume Creator 0.67

They say that everyone has a book in them, but have you ever tried writing one? It's all fun and games at the start, when you're concocting plots, scenes and characters, but as the story progresses, managing everything can become a nightmare. You can try to structure things in a word processor, but a better solution is to use a dedicated novel writing tool like *Plume Creator*.

*Plume*'s website is pretty rubbish, with little documentation on using the program. But it does show you how to install it: 32-bit and 64-bit packages for Ubuntu and Mageia are available, along with the source code. You'll need version 4 of the *Qt* libraries to install it, as the interface is built with that toolkit.

Start *Plume* and you'll be prompted to create a new project. You're asked for the type of book (eg a short novel), and you can choose how many chapters and scenes it should contain here – but don't worry if you're not sure, as you can modify them later. From here onwards, *Plume* works a lot like a regular editor, except it helps you to manage different scenes and chapters. A tree list down the left-hand side lets you quickly switch between different parts of the book, while additional tools are available such as a note-taking panel and a timer.

The mise-en-scène panel is especially useful, letting you keep track of characters, items and places. You can note here where a character was at a certain time, and what items he/she had, to avoid continuity errors. Once you're happy with your work, you can export it in a variety of formats, including ODT (as used by *LibreOffice*), HTML and plain text. There are still plenty of unfinished bits in *Plume*, but by version 1.0 it should be a great app for aspiring writers.



Project: 125 Book: 125 Act: 125 Chapter: 1

Twas a cold and windy night in Upper Bavaria. Mike battled to land his Panther Clipper on Swallow Landing in Achenar, before the horror of all horrors occurred: there was no more beer left.

Out of sheer desperation, Mike grabbed his mobile phone and called the Oberbayerischerbierbestellungsdienst, but no answer - nothing. The cold, inhuman beeps at the end of the line caused a tear to roll down Mike's beer-starved cheek.

"Oh, the humanity!" exclaimed Mike, before realising that a 24-hour garage wasn't actually that far away, so he legged it over and grabbed a lovely bottle of Weihenstephaner.

And he lived happily ever after. (Well, until he was blown apart by an Imperial Courier in Phekda, but that's a story for the second book.)

Mike
Aliases (separated by , (co...
The main dude
Main
Protagonist
None

Mike's long journey towards being a true Bavarian takes him on a rollercoaster ride of looking at mountains, drinking beer and complaining about the ridiculous Blue Shell in post-SNES Mario Karts.

*Plume*'s interface could do with some refinement, but after 10 minutes of exploring you'll get the hang of it.

**"Plume helps you manage different scenes and chapters of your book."**

**PROJECT WEBSITE**
**http://www.plume-creator.eu/site/index.php/en**

Convert ANSI codes to readable text

## Ansifilter 1.9

Here's something interesting to try: in a terminal window, in a directory with various files and folders, enter **ls --color > list.txt**. This redirects the output of the **ls** command (with all its colour goodness) to the file **list.txt**. Now open that file in a text editor, or view it with **less list.txt**. Notice something strange? The colours aren't there – just some weird characters like:

```
ESC[01;34mfolderESC[0m
```

Ugh. What's happening here? Well, colours and effects (like bold text) are created in the terminal via ANSI codes, which involve the escape character and numbers. Any good terminal can interpret these in command output and display them properly, but when you redirect the output to a file, it just becomes plain text.

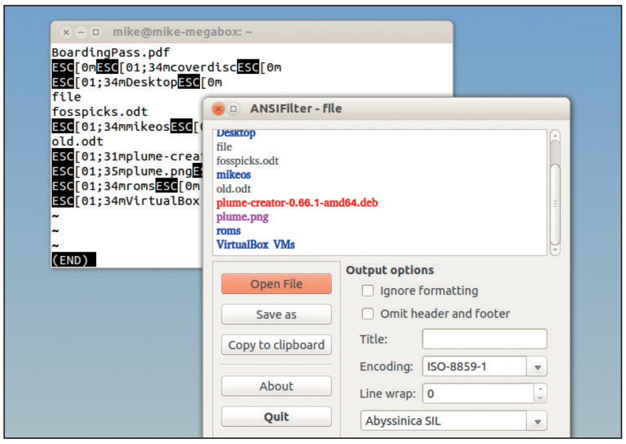
Now, say you have an important file containing these codes, eg from a log, and you want to make the information human-readable. *Ansifilter* is a godsend here: it converts the file into a better format, such as plain text, HTML, Latex, RTF or even BBCode (very useful if you want to paste the output of a command into a forum post). It's supplied as two programs, the first of which runs at the command line, and the second of which uses *Qt* to produce a simple but pleasant little GUI app.

To convert **file.log** into a HTML version called **file.html**, you'd run:

```
ansifilter -T file.log > file.html
```

Alternatively, run **ansifilter-gui file.log** to get a preview of the output, then click Save As to choose one of the formats mentioned previously. You can even change the text encoding, along with the line wrap settings and font that should be used.

*Ansifilter* isn't a tool you'll use on a daily basis, but it can save your life if you have a log file peppered with control codes and you desperately need to get information out of it.



```
BoardingPass.pdf
ESC[0mESC[01;34mcoverdiscESC[0m
ESC[01;34mDesktopESC[0m
file
fosspicks.odt
ESC[01;34mmikeosESC[
old.odt
ESC[01;31mplume-crea
ESC[01;35mplume.pngE
ESC[01;34mromsESC[0m
ESC[01;34mVirtualBox
~
~
~
(END)
```

```
Desktop
file
fosspicks.odt
mikeos
old.odt
plume-creator-0.66.1-amd64.deb
plume.png
roms
VirtualBox VMs
```

Output options
Ignore formatting [ ]
Omit header and footer [ ]
Title:
Encoding: ISO-8859-1
Line wrap: 0
Abyssinica SIL

ANSI codes in their raw format, and how *Ansifilter* interprets them.

**PROJECT WEBSITE**
**www.andre-simon.de/doku/ansifilter/en/ansifilter.php**

Spreadsheet app

## mtCellEdit 2.4

The flagship spreadsheet program for Linux and other FOSSy systems is *LibreOffice Calc*. We already have a lighter alternative in the form of *Gnumeric*, which is darn good by the way – but *mtCellEdit* is even smaller. It's a very basic spreadsheet program, lacking many of the features and frills you'll find in the bigger tools, but for basic calculation jobs it's great.

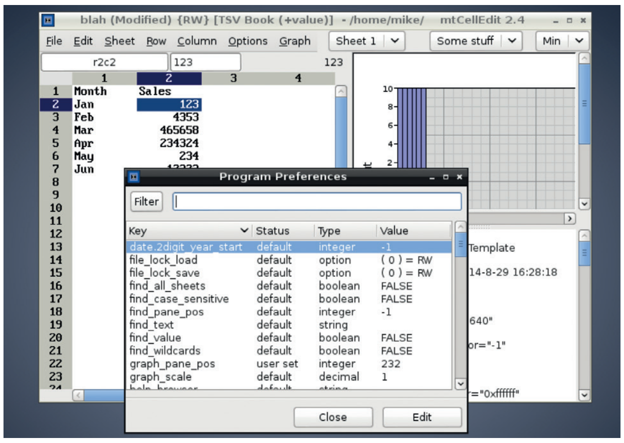
What's not so great, though, is the compilation process. When you extract the tarball you're faced with a bunch of directories containing different parts of the program – and a **README.txt** that doesn't provide much info. It does, however, point you at the project's HTML documentation, which explains the order in which you need to compile the components. The *GTK 2* toolkit is the main dependency.

Start the program and you'll see that *mtCellEdit*'s interface is as bare as they come: you have a grid for entering data, a drop-down list to switch between different sheets, and a handy list in the top-right showing values for selected cells (eg sum, maximum, average).

*mtCellEdit* refers to individual cells by their row and column numbers, so if you want to display the sum of columns 1 and 2 in row 1, you'll use this command:

```
=r1c1+r1c2
```

That's rather different to the A1, B2 etc system used by other spreadsheets, and takes a while to get used to. It's possible to generate bar charts in the program, although we found this cumbersome, requiring copying and pasting chunks of data into a text file, and having to do a lot of manual fiddling to get it right. *mtCellEdit* can open and save CSV and TSV (comma and tab separated value) files, though, so it's easy to share data with other apps.


blah (Modified) {RW} [TSV Book (+value)] - /home/mike/ mtCellEdit 2.4
File Edit Sheet Row Column Options Graph
Sheet 1
Some stuff
Min
Program Preferences
Filter
Close
Edit

| | 1 | 2 |
|---|---|---|
| 1 | Month | Sales |
| 2 | Jan | 123 |
| 3 | Feb | 4353 |
| 4 | Mar | 465658 |
| 5 | Apr | 234324 |
| 6 | May | 234 |
| 7 | Jun | |

| Key | Status | Type | Value |
|---|---|---|---|
| date.2digit_year_start | default | integer | -1 |
| file_lock_load | default | option | ( 0 ) = RW |
| file_lock_save | default | option | ( 0 ) = RW |
| find_all_sheets | default | boolean | FALSE |
| find_case_sensitive | default | boolean | FALSE |
| find_pane_pos | default | integer | -1 |
| find_text | default | string | |
| find_value | default | boolean | FALSE |
| find_wildcards | default | boolean | FALSE |
| graph_pane_pos | user set | integer | 232 |
| graph_scale | default | decimal | 1 |

There are plenty of options to tweak, but they're not presented in the most human-friendly fashion.

"For basic calculation jobs, mtCellEdit is great."

**PROJECT WEBSITE**
**http://code.google.com/p/mtcelledit/**

Scripting language

## PHP 5.6.0

PHP gets a lot of flak from many developers; they regard it as a toy language that has become ugly and bloated over the years, lacking logical design and consistency. Even Rasmus Lerdorf, the creator of PHP, said that he had "absolutely no idea how to write a programming language" at the beginning. On the other hand, it's useful for cooking up quick websites on a LAMP stack, and many well-known web apps such as *WordPress* are built with it.

Anyway: PHP 5.6.0 was released at the end of August, and it brings a bunch of improvements, many of which have been in discussion for a while. High up on the list is support for constant scalar expressions, where you can use expressions in which PHP previously expected static values. For instance, you can now do this:

```
const ONE = 1;
const TWO = ONE * 2;
```

You can use them in other places like default function arguments too – the idea is to make code easier to read and more expressive. Then there's better handling of variable length argument lists for functions, so instead of messing around with **func_num_args()** and the like, you can start a function like so:

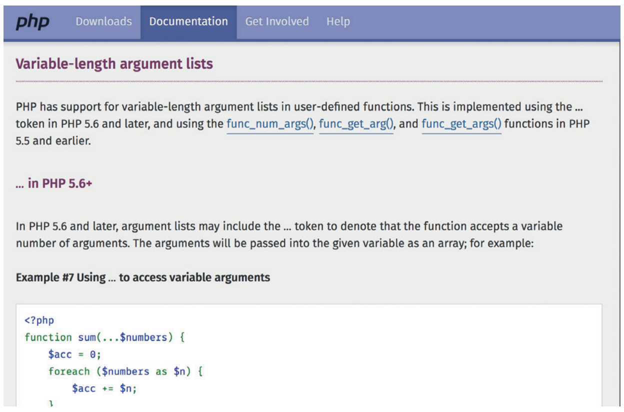
```
function sum(...$numbers) {
```

Thanks to the **...** token, this places all the arguments into an array called **$numbers**, over which you can iterate using **foreach**.

Exponation using the ** operator is now supported in PHP 5.6.0, which means you can do this:

```
$c = $a ** $b;
```

Where **$c** contains the result of raising **$a** to the **$b**'th power.



Variable-length argument lists

PHP has support for variable-length argument lists in user-defined functions. This is implemented using the ... token in PHP 5.6 and later, and using the func_num_args(), func_get_arg(), and func_get_args() functions in PHP 5.5 and earlier.

... in PHP 5.6+

In PHP 5.6 and later, argument lists may include the ... token to denote that the function accepts a variable number of arguments. The arguments will be passed into the given variable as an array; for example:

Example #7 Using ... to access variable arguments

```
<?php
function sum(...$numbers) {
    $acc = 0;
    foreach ($numbers as $n) {
        $acc += $n;
```

As usual, PHP's new features are well documented, with examples showing how you can incorporate them into your own code.

Many other improvements and tweaks have been made around the codebase too: the **phpdbg** debugger has been integrated into the core function and constant importing is now possible with the **use** keyword; and file uploads of larger than 2GB are now supported. This release might not win over all the naysayers, but it's a solid job.

**PROJECT WEBSITE**
**www.php.net**

## FOSSPICKS Brain Relaxers

Space trading/combat game

## Oolite 1.80

We at Linux Voice HQ all have misspent youths thanks to David Braben and Ian Bell. While other kids were being cool, playing sports and chasing girls, we were perfecting docking sequences and selling robots on the black market in Sol. Yes, we loved *Elite* (and its sequel *Frontier*), and as *Elite: Dangerous* is getting tantalisingly close to release, we've been playing some open source *Elite*-ish games too.

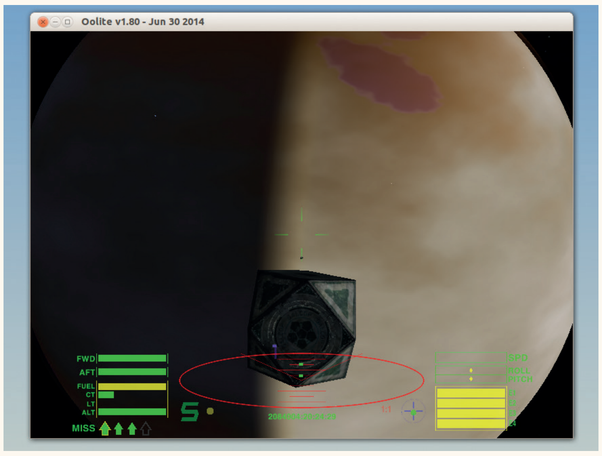
*Oolite* is the arguably the best, and recently received a major update, bringing it to version 1.80. You can grab it in 32-bit or 64-bit versions from the game's website – we did the latter, and installed it like so:

```
tar xfv oolite-1.80.linux-x86_64.tgz
./oolite-1.80.linux-x86_64.run
```

We asked for the game to be installed in our home directory, and a menu icon was created under Games. (The installer also tells you how to run it manually.)

There are three main modes to *Oolite*: Normal is the full game, taking the core gameplay of *Elite* and adding lots of extra goodies. There's a tutorial mode for new players, along with a Strict mode, which aims to ape the original as closely as possible.

Version 1.80 brings about more variety in the galaxy maps, and more combinations of non-player characters, such as packs of pirates working together. You, as the player, now have a reputation, so if you're a skilled bounty hunter then many pirates will stay out of your way. It's also now easier to install expansion packs – a darn good thing, given that there are over 500 of them...



The HUD is almost identical to *Elite*'s, but the planets and spacecraft look a jillion times better.

**PROJECT WEBSITE**
**www.oolite.org**

Board/puzzle game

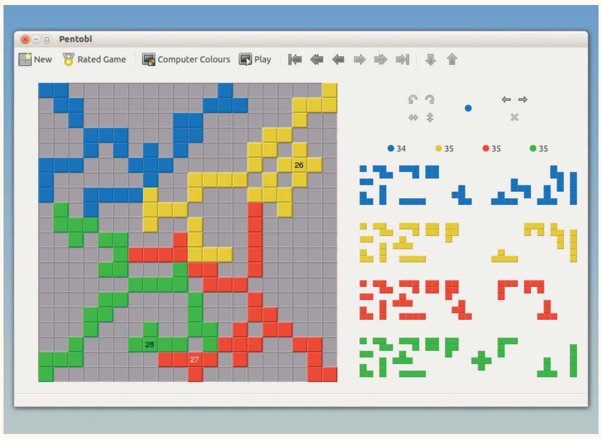
## Pentobi 8.1

Looking at the screenshot, you might be tempted to think that *Pentobi* is yet another *Tetris* rip-off, and fair enough - the formula has been done to death. But although *Pentobi* uses similar shaped pieces, it's a very different game. For starters, it's based on a board game called *Blokus* that was invented in 2000, and it's great fun in multi-player mode.

*Pentobi*'s main dependency is *Qt*; usefully, it can be built with version 4.x or 5.x of the toolkit. When you start the game, you're presented with a blank board, and by default it's you vs three CPU-controlled opponents. (Click on the Computer Colours button at the top of the window to replace CPU players with real-life human ones.)

The rules are like so: each colour takes it in turns to place a piece on the board, starting with the blue player. On the right-hand side is a palette of pieces from which you can choose – ranging from single-block pieces to five-block ones – and you can only use each piece once. You place your first piece in your designated starting corner, and subsequent pieces have to touch the same colour on the corners, but not directly on the edges. So you end up building a construction out of your corner.

However – as the other players build their constructions, there's less and less space on the board. You have to plan ahead to place as many of your pieces as possible. The game ends when nobody can place anything else, and a score is totalled based on how many pieces you didn't place. It's challenging, addictive, and gets the brain ticking over...



It's early days, but blue is getting trapped here, thanks to sneaky CPU opponents...

**PROJECT WEBSITE**
**http://pentobi.sourceforge.net**

# PCI EXPRESS LTD

## LINUX BASED DIGITAL TV TUNERS

## Matrix ARM MINI PC

MATRIX IS A SINGLE BOARD MINI COMPUTER BASED ON ARM WITH A WIDE RANGE OF INTERFACE, EQUIPPED WITH A POWERFUL I.MX6 FREESCALE PROCESSOR

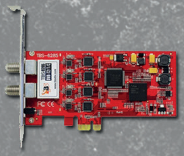
ONLY £119 - FREE UK DELIVERY

## NEW PRODUCT - £49.95

5220 USB DVB-T2 / T / C TUNER TV STICK

### TBS6285 DVB-T2 QUAD TUNER

THE ONLY 4 TUNER FREEVIEW CARD

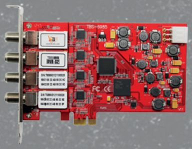


£179

LOW PROFILE DESIGN
QUAD CHANNEL RECEIVING SIMULTANEOUSLY
BOTH DVB-T2 AND DVB-T ARE SUPPORTED
HIGH-DEFINITION VIDEO
HIGH SENSITIVITY DVB-T2 QUAD TUNER
WINDOWS AND LINUX DRIVER READY

### TBS6285 DVB-T2 QUAD TUNER

THE ONLY 4 TUNER FREEVIEW CARD



£209

LOW PROFILE DESIGN
QUAD CHANNEL RECEIVING SIMULTANEOUSLY
BOTH DVB-S2 AND DVB-S ARE SUPPORTED
HIGH-DEFINITION VIDEO
HIGH SENSITIVITY DVB-S2 QUAD TUNER
WINDOWS AND LINUX DRIVER READY

### OTHER TUNERS AVAILABLE

| | | |
|---|---|---|
| 6220 | DVB-T2 TV TUNER | £79 |
| 6281 | DVB-T2 DUAL TUNER | £109 |
| 6925 | DVB-S2 PROFESSIONAL TV TUNER | £189 |
| 6982 | DVB-S2 DUAL TUNER | £99 |
| 6991 | DVB-S2 DUAL TUNER DUAL CI | £159 |
| 6922 | DVB-S2 TV TUNER | £79 |
| 6928 | DVB-S2 TV TUNER CI | £99 |

### LOOKING TO BECOME A TBS RESELLER ?

TO BE ELIGIBLE FOR A WHOLESALE TRADING ACCOUNT WITH PCI EXPRESS, AND THUS PURCHASE TBS PRODUCTS AT TRADE/WHOLESALE PRICES, WE SIMPLY REQUIRE YOU TO COMPLETE OUR TRADE WHOLESALE APPLICATION FORM.

PLEASE EMAIL MIKE@PCIEX.CO.UK FOR THE APPLICATION FORM

WHOLESALE / BULK / TRADE / DROPSHIPPING

OFFICIAL RESELLERS

REDAPPLEDIGITAL.CO.UK | QUIETPC.COM | CCLONLINE.COM | VALUEAV.CO.UK | DRSTRADING.CO.UK

OFFICIAL UK DISTRIBUTOR

## PCI EXPRESS LTD

TEL: 01372 377850 | WWW.TBSCARDS.CO.UK

UNIT 3 BAYTREE AVENUE, KINGSTON ROAD, LEATHERHEAD, SURREY, KT22 7UE

LINUX VOICE

# TUTORIALS

Dip your toe into a pool full of Linux knowledge with eight tutorials lovingly crafted to expand your Linux consciousness

**Ben Everard**
**is glad there's no IT department to stop him poking about the internals of his PC.**

All of us at Linux Voice would like to send our congratulations to Limor Fried and the team at Adafruit for their **inc.com** ranking as the 11th fastest growing manufacturing company in the USA. In case you've never heard of Adafruit, it designs and builds electronics stuff for hobbyists (we'd like to be more precise than 'stuff', but it really does sell everything that an amateur circuit builder might need). What's more, all of the things it designs are released open source under creative commons licences.

While most of the things they make are fairly straightforward (at least when compared to computer components), it shows that you can build a company that respects people's freedoms. This, of course, isn't news in the software world, where companies have been working with free software for quite a long time. However, it is quite new in the manufacturing world.

There's a real energy and buzz around the hobby electronic scene that's driving open source hardware at the moment. If physical computing is something that interests you, now's a really good time to get into it. If you're looking for some hardware to help you get started, well, I know a place that stocks some great stuff and respects your freedoms.
**ben@linuxvoice.com**

## In this issue...

LINUX VOICE TUTORIAL

# HDR: CREATE AWESOME PHOTOGRAPHS

GRAHAM MORRISON

Harness the power of open source to capture light and shade in stunning photo composites.

**WHY DO THIS?**

- Use open source firmware on your camera.
- Turn photography into a geeky hour of parameter tweaking.
- Impress your friends and relatives.

Photos with a high dynamic range (HDR) have a quality and detail that can't be matched by ordinary photos. This is because an HDR image is a combination of both the underexposed and overexposed details within more than one photo – the parts that are usually lost when your camera attempts to set a single exposure value for a single shot. The most popular solution, and the one commonly referred to as HDR, involves taking the same photo at different exposure settings and then combining the various images with a clever piece of software that can then export the final HDR image. And that's exactly what we're going to show you to do now.

Turn an old French château into a vibrant explosion of colour and detail.

## Image composition with Magic Lantern and Luminance

### 1 Steady as she goes

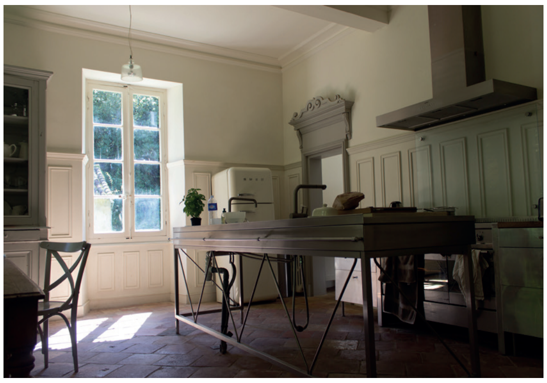
You'll need a camera that enables you to control the exposure settings, because you'll need to adjust these between each of the shots we're going to take. And because the final generated image is going to be a clever composite of all these shots, it's absolutely essential that your camera remains in exactly the same position between each shot. If not, the hassle of aligning your images or compensating for even a small movement can take much of the enjoyment out of creating the images.

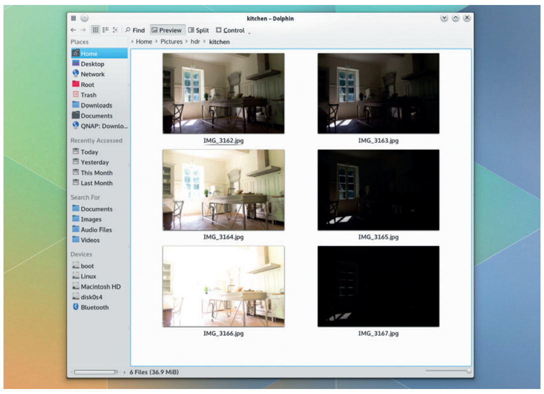
For this reason, you should try to use a tripod, or at the very least, find a stable place to put your camera and use its timer delay function. This will help to remove any wobble added by your finger prodding the shutter button. In the below image you can see that HDR would be able to bring out the details in the dark parts of the image without overexposing the bright part shining through the window.

### 2 Use a camera with bracketing

Some cameras can now do this automatically with a function called 'bracketing' – ramping up the exposure in a scene from underexposed (dark) to overexposed (light). Canon's DSLRs are our option purely because they can run the *Magic Lantern* open source firmware. This brilliant third-party firmware is worth a tutorial in itself, as it adds a host of excellent features not enabled by Canon.

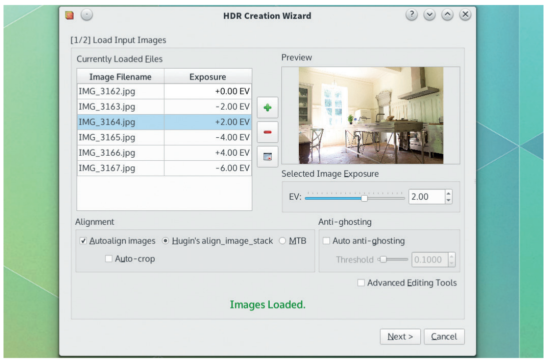
With the firmware installed, for example, HDR Bracketing is the first option in the custom menu, and when this is enabled you simply press the shutter. *Magic Lantern* calculates how many different exposures are needed and takes the shots as required. If you need to do this manually, make sure your camera is in its aperture value mode, set manual focus, use the timer and change the aperture/exposure values – typically six times – -3,-2,-1,+1,+2,+3.





### 3 Luminance HDR

The software that's going to perform most of the magic is called *Luminance HDR*, and we used version 2.4.0. You should be able to find it from your distribution's package manager. You should also install the beta version of *hugin*. This is the awesome panorama stitching tool, and its **align-image-stack** command is used by *Luminance HDR* to ensure each image is perfectly aligned. With that out of the way, launch *Luminance HDR* and click on the 'New HDR Image' button. This will open a requester where you should add your set of images with the **+** icon. Your camera should include the exposure metadata, which will be listed to the right of the images, and you should check that these correspond with the preview. Unless you've ensured your images are aligned, check the Autoalign Images option and click Next. This can take a while with autoalign enabled.



HDR Creation Wizard

[1/2] Load Input Images

Currently Loaded Files

| Image Filename | Exposure |
|---|---|
| IMG_3162.jpg | +0.00 EV |
| IMG_3163.jpg | -2.00 EV |
| IMG_3164.jpg | +2.00 EV |
| IMG_3165.jpg | -4.00 EV |
| IMG_3166.jpg | +4.00 EV |
| IMG_3167.jpg | -6.00 EV |

Preview

Selected Image Exposure

EV: 2.00

Alignment

✓ Autoalign images ● Hugin's align_image_stack ○ MTB

☐ Auto-crop

Anti-ghosting

☐ Auto anti-ghosting

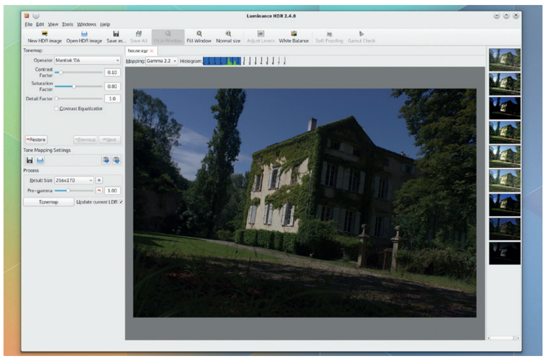
Threshold 0.1000

☐ Advanced Editing Tools

Images Loaded.

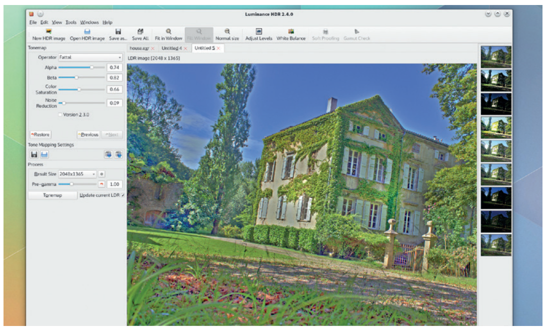
### 4 Tonemapping

You can click Next to skip through the creation profile wizard. After a little more processing, you should be dropped back to *Luminance HDR*'s main window with a single tabbed image showing the results of your composition. It will probably look dark and terrible, but this is because we have yet to map the depth of image data to the screen. This is done by configuring a tonemap, and there are variety on offer. The quickest and easiest to use is called 'Mantiuk '06', and this should be selected from the drop-down menu in the tonemap panel. Below this, expand the result's size resolution so you can get a better feel for the result – size will affect the processing, but not as much as the tonemap algorithm. We suggest saving the *Luminance HDR* project here, as we experienced a few stability problems. Now click on the 'Tonemap' button. This will generate a new tab with your first HDR image.



### 5 Playing with the options

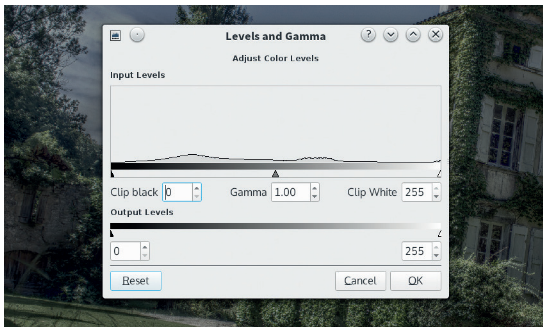
It takes a bit of time between each preview, so you now need to make small changes to the tonemap parameters until you get the HDR look you want. With 'Mantiuk '06', we'd suggest ramping up the contrast and saturation factors and only sparingly adding to the detail factor. You'll see what's happening much easier than us possibly trying to explain it, but the detail slider adds that crazy haunting look that lots of HDR images use. If you find a combination you like, it's worth saving it as a preset before moving on to another tonemapping algorithm. Each has a different style; 'Mantiuk '08' is a more subtle version of the one we've been playing with, for example, whereas 'Fattal' really does add lots of noise and colour to an image – especially if you disable the 'Version 2.3.0' checkbox. The best thing to do is experiment and find a result you like before moving on to the final step.



### 6 Final output

When you've got a result you like, we'd suggest opening the levels window and dragging the black arrow on the left and the white arrow on the right inward slightly to improve the contrast. You can turn on a real-time preview for this from the Tools menu to make your adjustments easier. You might also want to click on the White Balance button. Finally, save your creation just as you did the settings, only this time make sure the extension is **.jpg**.

Before sharing the file, we'd highly recommend making a few final changes using something like *Gimp*. This is because there are usually a few artefacts, and you can adjust the hues and contrast a little more intuitively in *Gimp* than you can within *Luminance HDR*. We also use The *Gimp* for a adding a slight blur and noise removal, before a final alignment and crop of the image before saving it. LV



Levels and Gamma

Adjust Color Levels

Input Levels

Clip black 0 Gamma 1.00 Clip White 255

Output Levels

0 255

LINUX VOICE TUTORIAL

# RASPBERRY PI: LET'S GET ANIMATED!

LES POUNDER

Start your own rival to Aardman Studios with a bit of stop motion animation, a tiny Linux machine and the magic of Python.

**WHY DO THIS?**

- Create your own mini movies using Lego and toys.
- Learn about the official Raspberry Pi camera and its Python Library.
- Expand the possibilities of the Pibrella add-on board.

**TOOLS REQUIRED**

- A Raspberry Pi.
- Raspbian operating system.
- Pibrella £10 from pimoroni.com.
- Official Raspberry Pi camera £15 from pimoroni.com.
- A light source.
- A white background.
- Modelling clay or Lego figures.
- Lego, Meccano, Blu Tack and anything that can be used to build a rig for the camera.

Wallace and Gromit, the classic British animated characters, started life as a very simple, but effective project using modelling clay. To create the illusion of animation a technique called stop motion photography was used. Stop motion is nothing new, but it is an effective tool and has been used in films such as *The Terminator* and *Aliens*. Stop motion photography is where a picture is taken of a model, and then the modeller will make a tiny adjustment to the model and take another picture; this is repeated many times to create a sequence of individual frames. Once these pictures are stitched together it looks as though the model is moving. Stop motion is a very labour intensive task, with twenty four frames making just one second of video (to create just one minute of video would take 1,440 frames!).

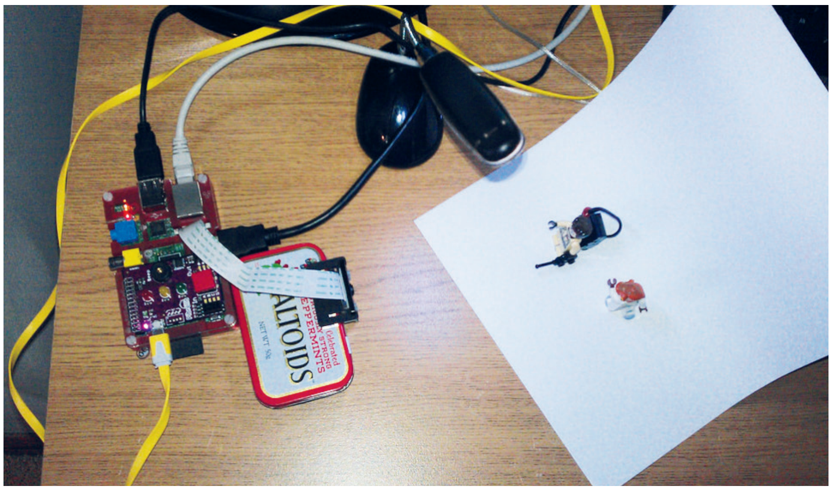
With the advancement of technology the animation process has become easier, and with the cost of hardware also dropping, anyone can enjoy making their own animation. The Raspberry Pi has become the go-to board for many projects and this month we will use it to create our own animation studio – though you could follow these steps on any Linux box.

Using a combination of Python code and a Bash script we will have all the software that we need to create animations. We're going to use two pieces of hardware in this project: the official Raspberry Pi camera and the fantastic Pibrella board, which we're going to use as a simple interface device thanks to its rather lovely big red button.

The Raspberry Pi Camera is the first component to be attached to our Raspberry Pi. With your Pi turned off, locate the CSI connector on your Pi. It is placed between the HDMI and the Ethernet port. At either end of the connector there are small lips that you need to gently lift from the Raspberry Pi. They're quite fragile so be careful, and once they are fully extended the CSI connector will be open and ready for you to insert the camera. The official camera has a very thin ribbon cable, another fragile component to be careful with. Insert the camera ribbon cable with the silver tips facing the HDMI port. With the ribbon cable in place press the lips down until the ribbon cable is locked in place. Installation of the camera hardware is complete, but we will need to make a few adjustments to the software later in this guide.

To install the Pibrella you just have to push the board down onto the GPIO pins. If you're lucky enough to own the new Raspberry Pi B+ board the Pibrella board works exactly the same, and should be connected to the first 26 pins of the GPIO. One little snag is that the board will be a little loose on the B+, as a capacitor that used to balance the Pibrella on previous models has been removed on the B+. The best remedy for this is to use something non-conductive between the Pibrella and B+ – Lego would work well.

*Ghostbusters* meets *Return of the Jedi*'s Admiral Ackbar in our cinematic opus. Still better than *Attack of the Clones*.


You don't need to spend a fortune to build a studio – some white paper, Blu-Tack and Lego figures can produce a simple film.

## Now set up the software

For this tutorial we used the latest version of the NOOBS installer to install an up-to-date version of Raspbian, as it comes with all the latest software and firmware for use with the camera. To download NOOBS and for instructions on how to set up your SD card head over to **www.raspberrypi.org/downloads**.

With NOOBS successfully installed on your SD card, now is the time to plug in all of the various peripherals such as keyboard, screen and Ethernet/wireless dongle. With that done, power up your Raspberry Pi and on first boot it will launch into the *raspi-config* setup tool. Using this tool we will expand the filesystem to ensure that we have the maximum amount of space that we need (option 1 in the list), and then enable the Pi Camera (option 5).

With that complete, exit *raspi-config* and reboot your Raspberry Pi, then when the Pi has fully rebooted, log back in and type:

```
startx
```

to start a new desktop session.

## Install Pibrella & Pygame

Pibrella from Cyntech and Pimoroni is a £10 add on board that enables anyone to quickly use electronics in their project. It comes with many different inputs and outputs for use in class and in LV005 we used it to control traffic lights and a dice game using Scratch and Python. For this tutorial we will use the lovely big red button to control taking a picture with the camera.

To install Pibrella, double-click on the *LXTerminal* desktop icon. In the terminal, type the following, remembering to press Enter at the end of each line.

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install python-pip
sudo pip install pibrella
sudo apt-get install vlc
sudo apt-get install mencoder
```

These commands will update the software installed and use the Python package manager **pip** to install the software needed for Pibrella to work. It will also install the *VLC* video player so that we can later view our completed project. To encode our pictures into a video we install the *Mencoder* tool– more on this later.

## Coding the animation studio

We're going to use the *Idle* development environment running Python 2.7, both of which come already installed in Raspbian. *Idle* is the ideal development environment for Python on the Pi. It's light, simple and helpful. Because we will be using the Raspberry Pi GPIO (General Purpose Input Output) pins we need to open *Idle* as root, as only the root user can use the GPIO. To do that, double-click on the *LXTerminal* icon to open a terminal window, and type

```
sudo idle
```

*Idle* will open with a shell window, which is an interactive session where you can test our code before writing a full program. To create a new project use File > New to open a blank document ready for our code. We first tell Python what libraries we would like to use, and we do that using the **import** command.

```
import pibrella
import picamera
import time
import datetime
import pygame
```

We have imported five Python libraries:

- **pibrella** to work with the Pibrella add-on board.
- **picamera** to work with the Raspberry Pi camera.
- **time** to enable us to delay and control the speed of the project.
- **datetime** enables our code to work with dates and times.
- **pygame** brings the **pygame** library of functions for audio, video and gaming to our code.

With the imports complete we now move to starting up **pygame** using

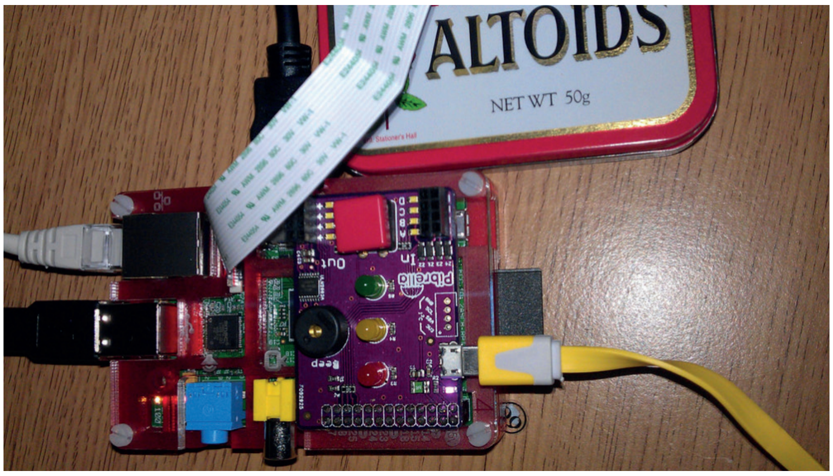
```
pygame.init()
```

Without doing this **pygame** will not work, and will create a lot of errors in the Python shell.

Our focus now moves to two variables, **w** and **h**, and a tuple that stores the values of both **w** and **h**. Variables can store individual values, but a tuple can store many more values, all separated by commas. Tuples can be used to create a readily updated set of values, such as GPS co-ordinates, or in our case the size of the window used by **pygame**.

```
w = 640
h = 480
size = (w,h)
```

The next stage of the project is a function that will be called when the big red button on the Pibrella is

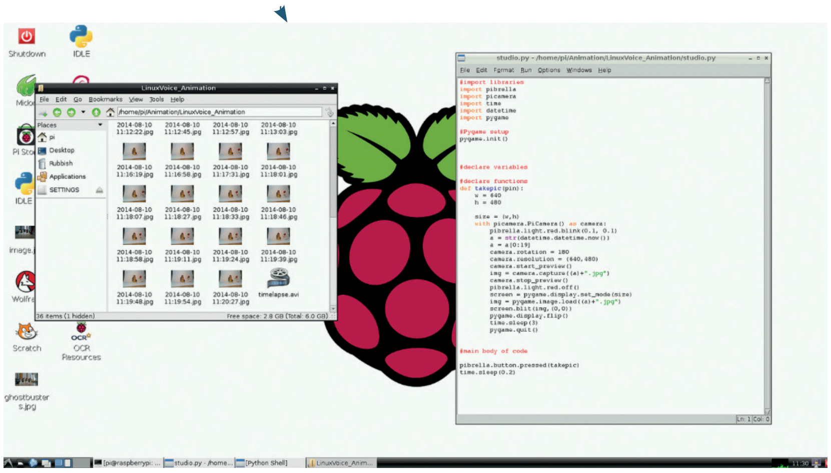

Pibrella simply slots on to the Raspberry Pi GPIO and works with all models of the Raspberry Pi.

### Boilerplate

Starting anything from scratch can be hard, and programming is no different. Python code is quite free and easy with how things are done, but a little structure can help you get started quicker. The term boilerplate comes from the web development community and it translates as a structured template to start from. I like to use comments to create sections in my Python code:

```
#Import any libraries
#Create any variables
#Create any functions
#Main body of code
```

In these sections I create the structure of my code, and by setting a formal structure I can easily locate and debug any issues that may occur. By using comments we also clearly show the order and logic of our code so that others can use and learn from the code in the future.



studio.py - /home/pi/Animation/LinuxVoice_Animation/studio.py

```
#import libraries
import pibrella
import picamera
import time
import datetime
import pygame

#Pygame setup
pygame.init()



#declare variables

#declare functions
def takepic(pin):
    w = 640
    h = 480

    size = (w,h)
    with picamera.PiCamera() as camera:
        pibrella.light.red.blink(0.1, 0.1)
        a = str(datetime.datetime.now())
        a = a[0:19]
        camera.rotation = 180
        camera.resolution = (640,480)
        camera.start_preview()
        img = camera.capture((a)+".jpg")
        camera.stop_preview()
        pibrella.light.red.off()
        screen = pygame.display.set_mode(size)
        img = pygame.image.load((a)+".jpg")
        screen.blit(img, (0,0))
        pygame.display.flip()
        time.sleep(3)
        pygame.quit()


#main body of code

pibrella.button.pressed(takepic)
time.sleep(0.2)
```

The Python code for this project will save a series of image files into the same directory as the location of the code.

pressed. When the function is called it will run through its code line by line.

As this function is rather large, let's break it down into chunks.

```
def takepic(pin):
  with picamera.PiCamera() as camera:
    pibrella.light.red.blink(0.1, 0.1)
    a = str(datetime.datetime.now())
    a = a[0:19]
```

First we define the name of our function; in this case, that's **takepic**. You will also see from the **(pin)** part of the function name this is a function takes an argument, or an extra bit of information. In this case the argument is a reference to the button present on the Pibrella board.

The second line is a handy method of renaming the rather long **picamera.PiCamera()** library as **camera**, making it much easier to work with.

The third line uses a function in the **pibrella** library to blink the red light on and off every 0.1 of a second. This blink is optional, but we added it to indicate that the button has been successfully pressed, and everyone loves a blinking LED.

The fourth line is a variable that we only create when the button is pressed. The variable **a** contains the output of **datetime.datetime.now()**, which is the current date and time. The sharp-eyed among you will have noticed the **str()** function also on this line. This rather helpful function converts any numerical data in to a string, in other words, text. We need to do this so that we can create the filename for the image later in the code.

The fifth and final line for this chunk of code is another variable... called a. But this time we are using a tool called string slicing to remove any unwanted text from the variable.

## The code

```
a = str(datetime.datetime.now())
```

produces the following output

```
2014-08-09 22:56:36.577712
```

**datetime** very helpfully gives us the exact time, but it's rather long, so using string slicing we can chop that down to a more manageable time to the second.

```
a = a[0:19]
```

produces the following output

```
2014-08-09 22:56:36
```

The second chunk of the function looks like

```
camera.rotation = 180
    camera.resolution = (640,480)
    camera.start_preview()
    img = camera.capture((a)+".jpg")
    camera.stop_preview()
    pibrella.light.red.off()
```

In this second chunk of code, the first line controls the rotation of the Pi camera. I rotated the camera 180 degrees, effectively turning the image upside down. Why do this, you might ask? Well I have a mount to protect the camera but it makes it a little unwieldy to position, and I found flipping the image provided me with the best position.

The second line:

```
    camera.resolution = (640,480)
```

sets the resolution of the picture taken, in this case to a rather small 640 pixels wide by 480 pixels high. This resolution is a compromise, as the camera is capable of creating pictures with a resolution of 2592px by 1944px. I chose 640 x 480 as it is a small file for the Pi to render into a video, which we will do later in this tutorial.

The third line:

```
    camera.start_preview()
```

instructs the camera to turn on and show a preview of the intended shot.

For the fourth line:

```
    img = camera.capture((a)+".jpg")
```

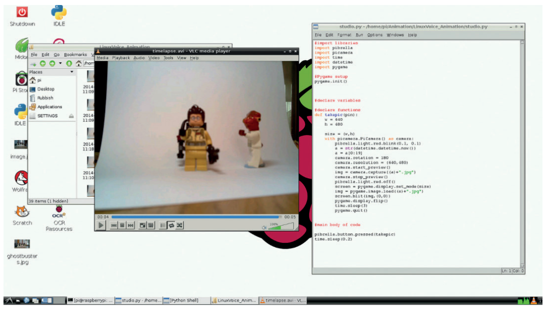
we capture the picture and then create a new variable called **img**; in this variable we store the filename created for the picture. Remember the variable **a** that we created earlier using **datetime**? Well, here we will use the contents of **a** and use a concept called concatenation to join the contents of **a** to the string "**.jpg**", effectively creating a complete filename.

The fourth line stops the camera preview window and quits the active window.

For the fifth and last line in this chunk the Pibrella red LED is reset by turning it off ready for the next shot to be taken.

Here is the last section of code that makes up the function.

```
  screen = pygame.display.set_mode(size)
```



Raspbian, the Raspberry Pi's default distro, has a built-in image viewer that can be used to review your images.

```
img = pygame.image.load((a)+".jpg")
screen.blit(img,(0,0))
pygame.display.flip()
time.sleep(3)
pygame.quit()
```

First in this chunk of code is a new variable called **screen**, which stores the values of setting the **pygame** display and uses the values stored in the tuple we created earlier.

The second line of code is another variable, which we use to call the function **pygame.image.load** and load the image that we have just taken, ready for display.

To display the image on the screen we use line three and something called **blit** (short for blitter). A blitter is a portion of memory dedicated to holding a bitmap image and is commonly used for sprites in video games – think Mario or Sonic running around in a game. We tell the blitter to open the picture, **img**, that we have just taken and position it at 0,0 on the screen. That means dead centre of the screen, using x and y co-ordinates.

To ensure that the display has been updated correctly the fourth line, **pygame.display.flip()**, is used to ensure that the correct image is displayed.

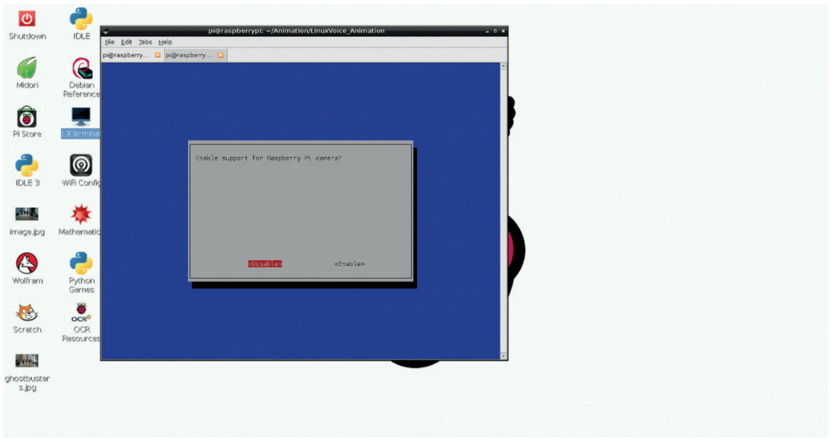
To give the user just enough time to see the picture we use line five to stop the code for three seconds by using the **sleep** function from the **time** library. The last line of code for the function closes the **pygame** window and cleans up ready to be used again.

With the function created our focus now shifts to the last two lines of code that make up the main body.

```
pibrella.button.pressed(takepic)
time.sleep(0.2)
```

Rather than use a **while True** loop to constantly check the status of the Pibrella button, we use an event. Events are commonly used in video games – for example, when a player presses the jump button, this instructs the game to make the sprite jump. So when the big red button is pressed, an event is triggered and this calls the function that we created earlier. The last line of code in this project is another **sleep** to delay the code by 0.2 seconds; this reduces the chance of the button being accidentally triggered twice, commonly known as a debounce.

With everything in place we are now ready to use the code for our studio. Go to the Run menu and select Run Module. The code will take a few seconds to load, you can use this time to arrange your shot. Lego and Blu Tack are great tools to help build a camera rig and studio. For your pictures you will need



```
Enable support for Raspberry Pi camera?
```

The Raspberry Pi camera is enabled using the **raspi-config** command in a terminal window.

a consistent light source and a bare background colour such as white. Arrange your Lego figures or modelling clay actors for the shot that you want. When you're ready, press the red button on the Pibrella to activate the code. You should see the red light flash, a preview picture appear on the screen, then a few seconds later the actual picture will appear.

All you need to do now is move your actors a little, take another picture and then repeat the process until complete. To make it a little easier on yourself aim for 6 pictures per second, so for a 10 second clip you will need 60 pictures. A top tip from Simon Walters (on Twitter know as @cymplecy, the eager maintainer of Scratch GPIO and its compatibility with many different add-on boards) is to record two seconds worth of images before and after the sequence that you wish to film, so the viewer settles in with the video.

## Encoding the video

Earlier we installed the *Mencoder* tool, which is a handy media converter. To make it even easier to use I have written a quick Bash script that will:

- List all the images in the same folder as the script.
- Save the list as a text file, which *Mencoder* will use to find the source files.
- Run the *Mencoder* tool to stitch the pictures together at six pictures per second, and save the video as **timelapse.avi**.

When you are ready to encode, open *LXTerminal* via the desktop icon and navigate to where you extracted the Animation Station code. In the terminal, type

```
./encode.sh
```

The script will launch and depending on the number of pictures in your movie, it will take a few minutes to encode the video. Once the encoding is complete, the script will launch *VLC* and your new movie.

Videos created using this technique can be imported into video editing applications such as *OpenShot* or *Kdenlive* on your main computer, mixed with audio and other videos to create the next *Toy Story* and amaze your friends. LV

**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.**

## Where can I find the completed code?

I've made the code for this project publicly available via GitHub. For those who are familiar with GitHub you can clone the repository at **https://github.com/lesp/LinuxVoice_Animation**, of you can download the archive as a Zip file from **https://github.com/lesp/LinuxVoice_Animation/archive/master.zip**.

LINUX VOICE TUTORIAL

# LINUX 101: BACK UP YOUR DATA

MIKE SAUNDERS

**Data loss can be agonising, whether it involves business documents or family photos. Never lose a file again with our guide!**

**WHY DO THIS?**
- Understand common Linux/Unix archiving tools.
- Save time with incremental backups.
- Encrypt your data for maximum security.

Linus Torvalds has made some classic quips over the years. Back in 1996, when announcing the release of Linux kernel 2.0.8, he noted that his hard drive was close to buying the farm, and added: "Only wimps use tape backup; real men just upload their important stuff on FTP, and let the rest of the world mirror it."

And it's a good point, especially today. If you're an open source software developer, you probably don't keep backups of your code, as it'll already be on SourceForge, or GitHub, or a million other repositories and mirror sites. But what about personal files? What about your music collection, letters, financial documents, family snaps and so forth?

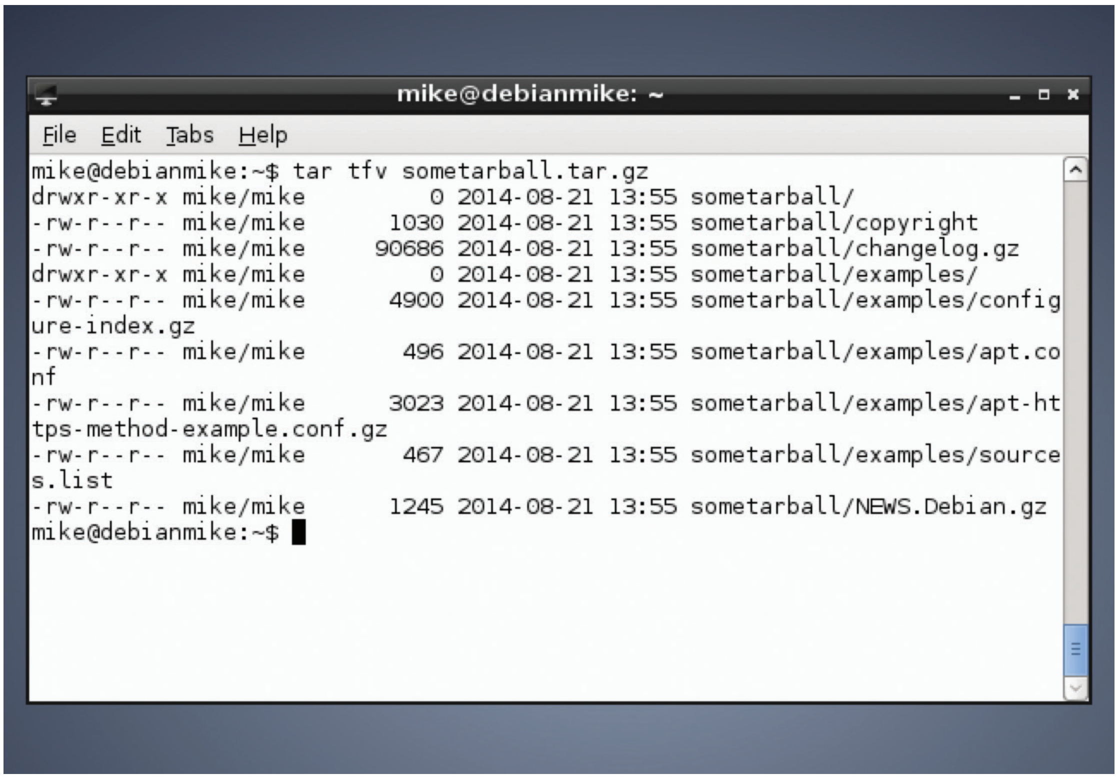
You can upload them onto a cloud storage service such as Dropbox, but there's no guarantee that the service will be around in the future, nor that government spooks aren't poking around inside your data. Ultimately, the best way to keep your data safe and secure is to make your own backups and maintain full control – and that's what we'll focus on now. We'll start off looking at the basic archiving tools included with every Linux distro, then examine more advanced options for incremental backups and encryption.

## 1 ROLLING UP A TARBALL

Many Linux and Unix commands have intriguing names that hark back to the early days of computing. For instance, the tool that's used to join a bunch of files together into a single file is called **tar**, which is a contraction of "tape archiver". Yes, it's a program that was originally designed for data tapes (we last used one in 2004), which aren't so much in common use today, but its job is still important.

You see, the Unix philosophy is all about small and distinct tools doing individual jobs, so that users can plug them together. (In contrast to giant megalithic applications that do a million things ineptly.) So when you create a compressed archive of some files in Linux, you actually end up using two programs. Take this command, for instance:

```
tar cfvz mybackup.tar.gz folder1/ folder2/
```



```
mike@debianmike:~$ tar tfv sometarball.tar.gz
drwxr-xr-x mike/mike         0 2014-08-21 13:55 sometarball/
-rw-r--r-- mike/mike      1030 2014-08-21 13:55 sometarball/copyright
-rw-r--r-- mike/mike     90686 2014-08-21 13:55 sometarball/changelog.gz
drwxr-xr-x mike/mike         0 2014-08-21 13:55 sometarball/examples/
-rw-r--r-- mike/mike      4900 2014-08-21 13:55 sometarball/examples/config
ure-index.gz
-rw-r--r-- mike/mike       496 2014-08-21 13:55 sometarball/examples/apt.co
nf
-rw-r--r-- mike/mike      3023 2014-08-21 13:55 sometarball/examples/apt-ht
tps-method-example.conf.gz
-rw-r--r-- mike/mike       467 2014-08-21 13:55 sometarball/examples/source
s.list
-rw-r--r-- mike/mike      1245 2014-08-21 13:55 sometarball/NEWS.Debian.gz
mike@debianmike:~$
```

Have a peek inside a tarball without extracting it using the **tar tfv** command.

This creates a single, compressed file (a tarball) called **mybackup.tar.gz**, containing **folder1** and **folder2** – you can add as many files or directories as you want onto the end. Now, we're using **tar** here to create the tar archive (a single file), hence the **.tar** part of the filename. But the **z** option to the command says that we want to run it through the **gzip** compression program as well, so we end up with **.tar.gz**. (The **c** option means create an archive, **f** means to create a file (instead of spitting the output to the terminal), and **v** means verbose, so it shows each file as it's being added.)

You can change the compression program that's used. For instance:

```
tar cfvj mybackup.tar.bz2 folder1/ folder2/
tar cfvJ mybackup.tar.xz folder1/ folder2/
```

Here we've replaced the **z** (gzip) option with **j** and **J**, which means **bzip2** and **xz** respectively. These programs use different algorithms to compress data, and the results can vary widely. The table below shows the time required to compress a 700MB folder containing a mixture of executable files, along with the resulting file size:

## Compression performance

| Program | Time | Size |
|---|---|---|
| gzip | 48.9s | 231MB |
| bzip2 | 2m34s | 208MB |
| xz | 10m1s | 164MB |

So you can see that **xz** is much, much slower than **gzip**, but it's also considerably better at compression. Different compression tools work better with different file types (eg some are more suited to audio data), so for your own backups, it's worth trying them all and seeing what results you get. You also need to consider the trade-off between speed and size: if your backup media has plenty of space and you want to archive files quickly, **gzip** is the way to go. If you need to be more economical with space but can leave the archiving process running overnight, **xz** is better.

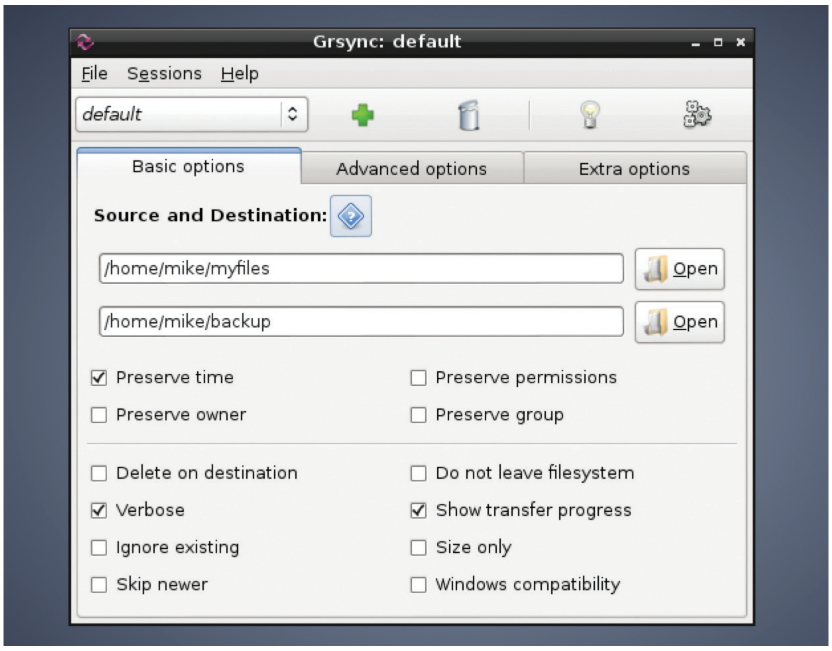
Extracting a compressed file is easy:

```
tar xfv mybackup.tar.gz
```

The same command works for files compressed with **bzip2** and **xz**. If you want to peek inside an archive to see what files are contained therein, without actually expanding it, use:

```
tar tfv mybackup.tar.gz
```

Again, this works for the other formats too. And if you have an archive without a useful filename extension – so you don't know what format it's in – just run the ever-useful **file** tool on it, eg **file mybackup.xxx**.



Source and Destination:
/home/mike/myfiles
/home/mike/backup

- [✓] Preserve time
- [ ] Preserve permissions
- [ ] Preserve owner
- [ ] Preserve group
- [ ] Delete on destination
- [ ] Do not leave filesystem
- [✓] Verbose
- [✓] Show transfer progress
- [ ] Ignore existing
- [ ] Size only
- [ ] Skip newer
- [ ] Windows compatibility

If you're not overly familiar with the command line, the Grsync GUI front-end to rsync (www.opbyte.it/grsync/) can make life easier.

**LV PRO TIP**

You can open **.tar.gz**, **.tar.bz2** and **.tar.xz** files on almost any Linux system, but what about backups that need to be opened on Windows machines? You can get third-party software to handle these formats, but it's often simpler to just use the de-facto standard Zip format. To create an archive: **zip -ry file.zip folder/**, and to extract use **unzip file.zip**. When creating, you can also add the **-1** option for the fastest compression (but larger resulting files), or **-9** for slower compression (but smaller files).

## 2 THE MIGHTY POWER OF RSYNC

So we've seen how to make simple compressed backups of data, but it's time to delve a bit deeper with the hugely versatile **rsync** tool. As its core, **rsync** helps you to synchronise data between a source and a destination directory, but various features make it especially useful for backup purposes. Another plus point is that it's ubiquitous – you can find it in virtually every Linux distribution, and it's also installed by default in Mac OS X and available for Windows.

Let's say you have a folder called **myfiles** with a few items in it, and an empty folder called **backup**. To copy the files from the former to the latter:

```
rsync -avh myfiles/ backup/
```

The **-a** option here means **archive** mode, so that metadata such as timestamps and permissions are preserved, while **-v** means **verbose** (providing extra information) and **-h** presents the information in a more human-readable form. When you execute the command, you'll see a list of files being copied, along with the total amount of data that was transferred.

Now, you're probably thinking: "Big wow! I can do that with a normal **cp** operation, right?" That's true, but try running the same command again – and notice the amount of data that's copied. Just a few bytes. Helpfully, **rsync** is cleverer than **cp** and checks to see if files already exist before copying them. And here's where it's great for backup purposes: it makes incremental backups, and doesn't shift data around unnecessarily.

For example: say you've been using a USB key to back up important files each month. The last backup of **/home/you** was 10GB. Since the last backup, you've only created a few extra files and your home directory contains 11GB. If you use **rsync** to perform the backup, it will only transfer the 1GB that has changed in the meantime, and not copy the whole 11GB over mindlessly. This saves a lot of time (and makes flash media last longer!).

## Media and location

Once you have the perfect backup system in place, you'll need to choose the right kind of media to store your data. On the low end, recordable DVDs are cheap and cheerful, and decent brands have guarantees for longevity (providing you keep the discs in the right environment). Blu-ray is becoming increasingly affordable as well – an external USB writer costs around £65, and for a spindle of 50 TDK discs (holding 25GB each) you'll pay a smidgen under £30.

Then there are external USB hard drives, which are reaching impressive capacities (2TB for around the £75 mark), along with tape drives that many businesses still swear by. In any case, if your data is incredibly important and you're making multiple backups, it's a good idea to use a variety of media. Imagine using three hard drives from the same vendor for your backups, only to find that a design defect makes them all break after six months…

Then there's the question of where to store your backup media. Where possible, it's a good idea to use different physical locations, to prevent everything from being lost in the case of robbery, fire or natural disaster. If you use Linux at home, you could always tightly encrypt your data using the guides in this article and ask a friend or neighbour to put a DVD or USB hard drive in a safe place. Most banks in the UK have stopped offering safety deposit box services now, although you can find independent companies that claim to store physical items securely.

### LV PRO TIP

Somtimes you'll see **.tar.gz** and **.tar.bz2** filenames written in a slightly shorter form: **.tgz** and **.tbz2**. This can help when files are being transmitted to older versions of certain operating systems that could get confused by multiple full-stop characters (naming no names…).

By default, **rsync** won't delete files from the destination directory if they have been removed from the source, but you can change that with:

```
rsync -avh --delete myfiles/ backup/
```

This is useful if you want your backups to be simple snapshots from certain points in time, and you don't want old and unwanted files lingering around forever.

Another great feature of **rsync** is the ability to narrow down the range of files to be stored. Try this:

```
rsync -avh --include="*.jpg" --exclude="*" myfiles/ backup/
```

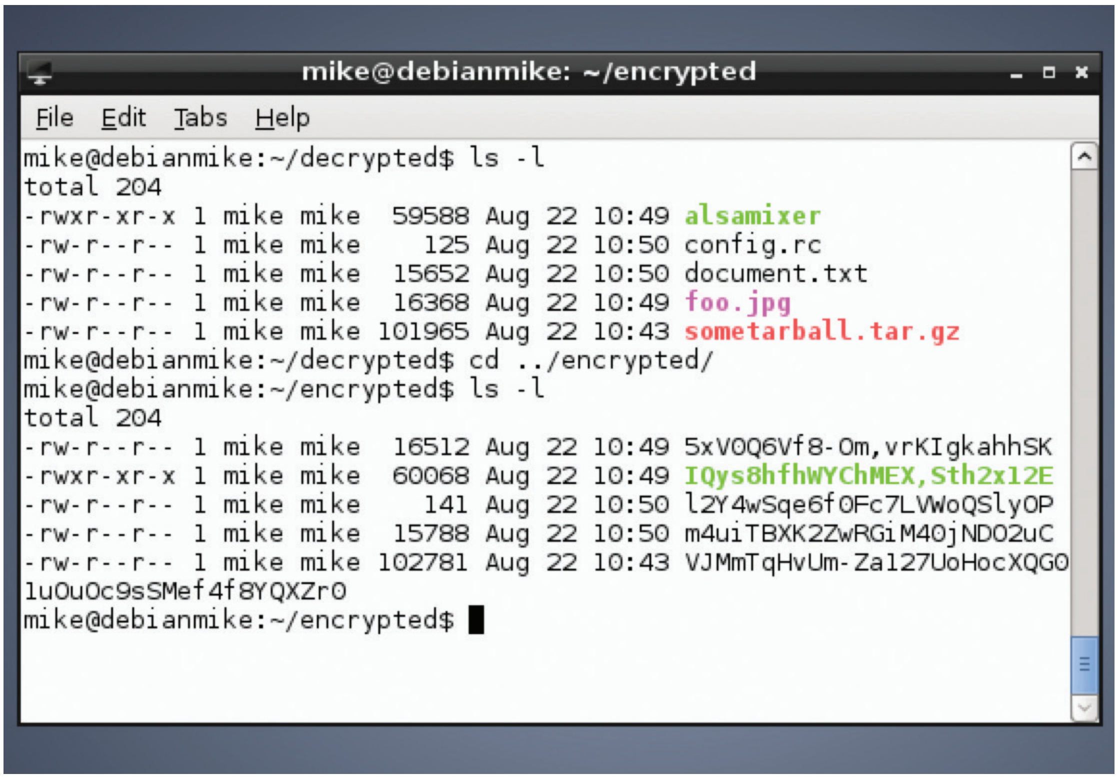
In this case, we're using wildcards to tell **rsync** to copy all files that end in **.jpg**, and exclude everything else (the asterisk means "all text" – ie any filename). This is handy when your home directory is a jumble of stuff, and you just want to back up your MP3, Ogg or FLAC files. (Use multiple **--include** options if you want to copy several types of file.)

Finally in this section, **rsync** also works a treat when copying files to remote servers. This helps if you have a NAS box somewhere on your home network, for instance, and you want to back up your desktop or laptop files to it. The simplest way to do this is via SSH, so if you have an SSH server running on the remote machine, you can do:

```
rsync -avhze ssh myfiles/ user@remote.box:backups/
```

The two options we've added here are **z** (to compress the data going across the network), and **e** followed by **ssh** to tell **rsync** which protocol we're using. Then we specify the local folder as usual, followed by a user and hostname combination, and then the folder in that user's home directory where the backup should be created.

Oh, and a last bit of efficiency awesomeness: when large files have been modified, **rsync** can detect which bits have changed, so it doesn't have to transmit entire files each time. If you take a large file and tack an extra byte on the end (eg **echo x >> file**), and then run **rsync** again, you'll see that it only sends the chunk that has changed. This really cuts down on bandwidth usage.



```
mike@debianmike:~/decrypted$ ls -l
total 204
-rwxr-xr-x 1 mike mike  59588 Aug 22 10:49 alsamixer
-rw-r--r-- 1 mike mike    125 Aug 22 10:50 config.rc
-rw-r--r-- 1 mike mike  15652 Aug 22 10:50 document.txt
-rw-r--r-- 1 mike mike  16368 Aug 22 10:49 foo.jpg
-rw-r--r-- 1 mike mike 101965 Aug 22 10:43 sometarball.tar.gz
mike@debianmike:~/decrypted$ cd ../encrypted/
mike@debianmike:~/encrypted$ ls -l
total 204
-rw-r--r-- 1 mike mike  16512 Aug 22 10:49 5xV0Q6Vf8-0m,vrKIgkahhSK
-rwxr-xr-x 1 mike mike  60068 Aug 22 10:49 IQys8hfhWYChMEX,Sth2x12E
-rw-r--r-- 1 mike mike    141 Aug 22 10:50 l2Y4wSqe6f0Fc7LVWoQSlyOP
-rw-r--r-- 1 mike mike  15788 Aug 22 10:50 m4uiTBXK2ZwRGiM40jND02uC
-rw-r--r-- 1 mike mike 102781 Aug 22 10:43 VJMmTqHvUm-Za127UoHocXQGO
1uOuOc9sSMef4f8YQXZr0
mike@debianmike:~/encrypted$ 
```

EncFS in action: the first directory shows the regular files, while the second is the encrypted versions with funny filenames.

## 3 ENCRYPTING YOUR DATA

And here we come to arguably the most important step in a backup procedure: encrypting your data. Obviously, this is essential if you're going to store your files in a cloud-based service such as Dropbox, but it's also well worth considering for locally stored backups as well. If someone gets physical access to your machines and nabs the drives, at least they won't get their mitts on your critical data.

If you've looked online for encryption tutorials before, you might've been overwhelmed by all of the options available. That's not a bad thing per se – it's good that there are so many methods and algorithms in widespread usage. Monocultures are normally bad, and if everyone were using the same encryption system and a fatal flaw in it were discovered, we'd all be doomed. So here are a couple of possibilities.

The simplest method is to use *GnuPG* like so:

```
gpg -c --cipher-algo AES256 filename
```

You'll be asked to enter a password (twice, to prevent typos from encrypting your file with the wrong password). The file will then be encrypted using a symmetric cypher, AES-256, which is strong enough for general usage, and the resulting file will be given a **.gpg** extension. To decrypt it, simply enter:

```
gpg filename.gpg
```

And that's it. It's also possible to encrypt using public/private key combinations, although that's a more complicated process and beyond the scope of this tutorial. But if you're interested, see **http://serverfault.com/a/489148**.

### Extra security with EncFS

Instead of encrypting individual files or tarballs, you can also add a layer of encryption onto your filesystem. So you can work with files normally, but when you shut down your machine, they're automatically stored in an encrypted format. To do this, install EncFS; it's a userspace filesystem that's available in most distros, and in Debian/Ubuntu it's just an **apt-get install encfs** away.

Firstly, create two directories in your home directory like so:

```
mkdir ~/encrypted ~/decrypted
```

(If you're not too familiar with the shell, **~** is a shortcut for your home directory.)

Now, the first directory here will be used as a permanent store for your data (in encrypted format), while the latter will be used on a temporary basis when you want to access the files. Enter this:

```
encfs ~/encrypted ~/decrypted
```

When prompted, hit **p** for 'paranoid' mode, and then enter a password (preferably long) that will be used to secure your data. The **encrypted** directory will now be mounted in **decrypted**, so try copying some files into the latter. Everything looks normal at this stage – you can work with your files just like in any other directory. Switch into the encrypted directory, however, and run **ls** – you'll see that there is the same number of files as in decrypted, but they all have bizarre names like **XEfn2,34CC-Bu3hs**.

These are the encrypted versions, in which the data permanently lives. So once you're finished doing your work in the **decrypted** directory, enter:

```
cd ~
fusermount -u ~/decrypted
```

This unmounts the encrypted drive from **decrypted**, so the latter is now empty; as mentioned, it's just a temporary place for working with the readable data. The permanent store is in **encrypted**, and you can access it at any point by repeating the previous **encfs ~/encrypted ~/decrypted** command and entering your password.

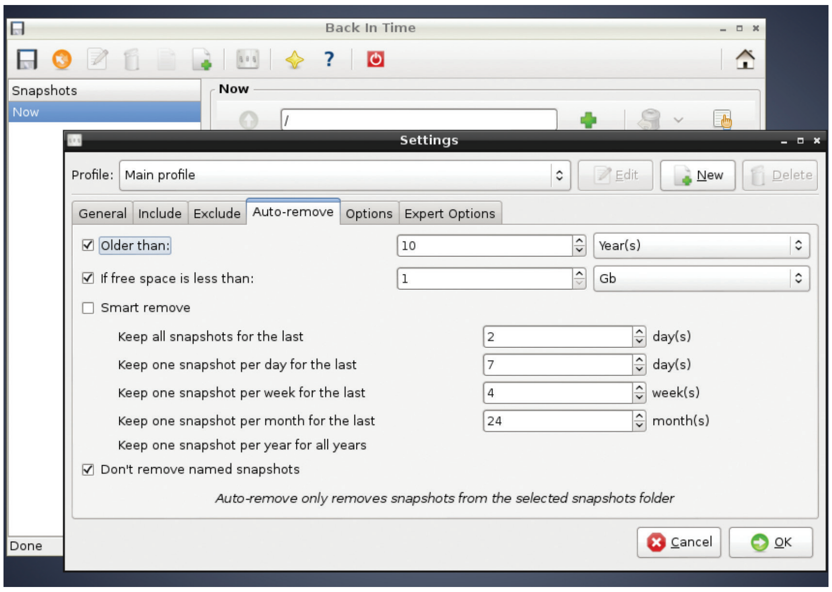
**Mike Saunders stores his data by printing out hex dumps and laminating the sheets. His cellar holds a whopping 30MB!**

### Alternative tools

We've focused on a core set of Linux tools in this article, but you can find more specialised open source backup solutions as well. *Bacula* (**www.bacula.org**) is a notable example that focuses on enterprises and backing up data over the network. To give you an example of its target users, it lets you print out special barcodes to stick on data tapes that can be then chosen in a tape drive auto-changer.

*BackupPC* (**http://backuppc.sf.net**), meanwhile, uses a client/server model, where the server organises backup schedules for multiple clients on the network. It's a complicated program, but thanks to its web-based administration panel, you don't have to faff around too much at the command line to set it up.

For home desktop users, *Areca Backup* (**www.areca-backup.org**) is a mature and well-designed app written in Java, while *Back In Time* (**http://backintime.le-web.org**) strives to provide a snapshot-based alternative to Apple's *Time Machine* system.



☑ Older than: 10 Year(s)
☑ If free space is less than: 1 Gb
☐ Smart remove
Keep all snapshots for the last 2 day(s)
Keep one snapshot per day for the last 7 day(s)
Keep one snapshot per week for the last 4 week(s)
Keep one snapshot per month for the last 24 month(s)
Keep one snapshot per year for all years
☑ Don't remove named snapshots
*Auto-remove only removes snapshots from the selected snapshots folder*

*Back In Time* clones some features of Apple's Time Machine, and has both Gnome and KDE-based front-ends.

**LV PRO TIP**

Complex **rsync** operations can do potential damage, such as overriding important data, so it's often worth adding the **--dry-run** option when you first run the command. This will show you exactly what **rsync** intends to do, without actually doing it. Once you're satisfied that everything is in order, re-run the command without it.

LINUX VOICE TUTORIAL

# JOHN THE RIPPER: CRACK PASSWORDS

BEN EVERARD

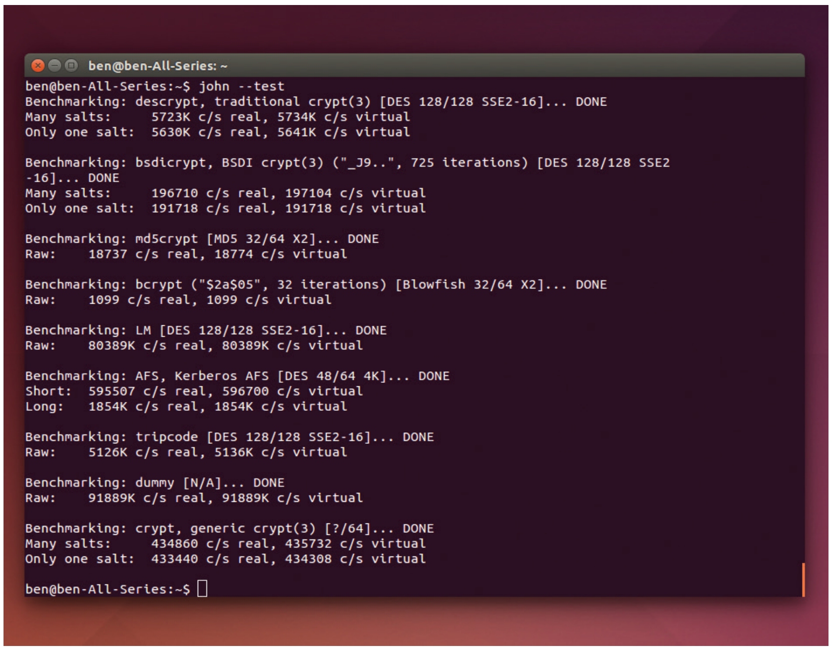
How secure are your passwords? Find out (and learn to stay safer online) by trying to crack them.

**WHY DO THIS?**

- Check the strength of password hashes.
- Understand the options when creating a secure system.
- Learn how password crackers work so you can create secure passwords.

Most people use passwords many times a day. They're the keys that unlock digital doors and give us access to our computers, our email, our data and sometimes even our money. As more and more things move online, passwords secure an ever growing part of our lives. We're told to add capital letters, numbers and punctuation to these passwords to make them more secure, but just what difference do these have? What does a really secure password look like?

In order to answer these questions, we're going to turn attacker and look at the methods used to crack passwords. There are a few password-cracking tools available for Linux, but we're going to use *John The Ripper*, because it's open source and is in most distros' repositories (usually, the package is just called **john**). In order to use it, we need something to try to crack. We've created a file with a set of MD5-hashed passwords. They're all real passwords that were stolen from a website and posted on the internet. MD5 is quite an old hashing method, and we're using it because it should be relatively quick to crack on most hardware. To make matters easier, all the hashes use the same salt (see boxout for details). Although we've chosen a setup that's quick to crack, this same setup is quite common in organisations that don't focus on security. You can download the files from **www.linuxvoice.com/passwords**.



```
ben@ben-All-Series: ~
ben@ben-All-Series:~$ john --test
Benchmarking: descrypt, traditional crypt(3) [DES 128/128 SSE2-16]... DONE
Many salts:     5723K c/s real, 5734K c/s virtual
Only one salt:  5630K c/s real, 5641K c/s virtual

Benchmarking: bsdicrypt, BSDI crypt(3) ("_J9..", 725 iterations) [DES 128/128 SSE2
-16]... DONE
Many salts:     196710 c/s real, 197104 c/s virtual
Only one salt:  191718 c/s real, 191718 c/s virtual

Benchmarking: md5crypt [MD5 32/64 X2]... DONE
Raw:    18737 c/s real, 18774 c/s virtual

Benchmarking: bcrypt ("$2a$05", 32 iterations) [Blowfish 32/64 X2]... DONE
Raw:    1099 c/s real, 1099 c/s virtual

Benchmarking: LM [DES 128/128 SSE2-16]... DONE
Raw:    80389K c/s real, 80389K c/s virtual

Benchmarking: AFS, Kerberos AFS [DES 48/64 4K]... DONE
Short:  595507 c/s real, 596700 c/s virtual
Long:   1854K c/s real, 1854K c/s virtual

Benchmarking: tripcode [DES 128/128 SSE2-16]... DONE
Raw:    5126K c/s real, 5136K c/s virtual

Benchmarking: dummy [N/A]... DONE
Raw:    91889K c/s real, 91889K c/s virtual

Benchmarking: crypt, generic crypt(3) [?/64]... DONE
Many salts:     434860 c/s real, 435732 c/s virtual
Only one salt:  433440 c/s real, 434308 c/s virtual

ben@ben-All-Series:~$
```

The speed at which *John* can crack hashes varies dramatically depending on the hashing algorithm. Slow algorithms (such as *bcrypt*) can be tens of thousands of times slower than quick ones like *DES*.

CloudCracker :: Online Hash Cracker - Mozilla Firefox

https://www.cloudcracker.com

CloudCracker

An online password cracking service for penetration testers and network auditors who need to check the security of WPA protected wireless networks, crack password hashes, or break document encryption.

Start Cracking

File Type: SHA-512 (Unix)

Hash File: Browse... No file selected.

Next

Handshake Dictionary Delivery

Save Money. Save Time.

Whether it's a WPA2 network, NTLM hashes, Unix hashes, or an encrypted PDF file, one thing's for certain. By specializing in optimized cracking solutions and by fine-tuning dictionaries from iteration to iteration, we can provide a solution that's more effective, faster, and cheaper than anything else.

Comprehensive Dictionaries.

We have a range of dictionaries, fine-tuned for the format at hand. By extrapolating from our successes and iterating over our failures, we've been able to converge on the most effective wordlists for the money, every time.

Big. Fast. Cheap. Run your network handshake against 300,000,000 words in 20 minutes for $17.

"Welcome to the future: cloud-based WPA cracking is here!" -- TechRepublic

"Low cost service cracks wireless passwords from the cloud..." -- TheRegister

"This really is a great idea." -- Hacker News

There are online services (like **www.cloudcracker.com**) that will try to crack passwords for a small fee.

After downloading that file, you can try and crack the passwords with:

```
john md5s-short
```

The passwords in this file are all quite simple, and you should crack them all very quickly. Not all password hashes will surrender their secrets this easily.

When you run **john** like this, it tries increasingly more complex sequences until it finds the password. If there are complex passwords, it may continue running for months or years unless you press Ctrl+C to terminate it.

Once this has finished running you can see what passwords it found with:

```
john --show md5s-short
```

That's the simplest way of cracking passwords – and you've just seen that it can be quite effective – now lets take a closer look at what just happened.

*John The Ripper* works by taking words from a dictionary, hashing them, and comparing these hashes with the ones you're trying to crack. If the two hashes match, that's the password you're looking for. A crucial point in password cracking is how quickly you can perform these checks. You can see how fast **john** can run on your computer by entering:

```
john --test
```

This will benchmark a few different hashing algorithms and give their speeds in checks per second (c/s).

By default, *John* will run in single-threaded mode, but if you want to take full advantage of a multi-threaded approach, you can add the **--fork=N** option to the command where **N** is the number of processes. Typically, this is best where **N** is the number of CPU cores you want to dedicate to the task.

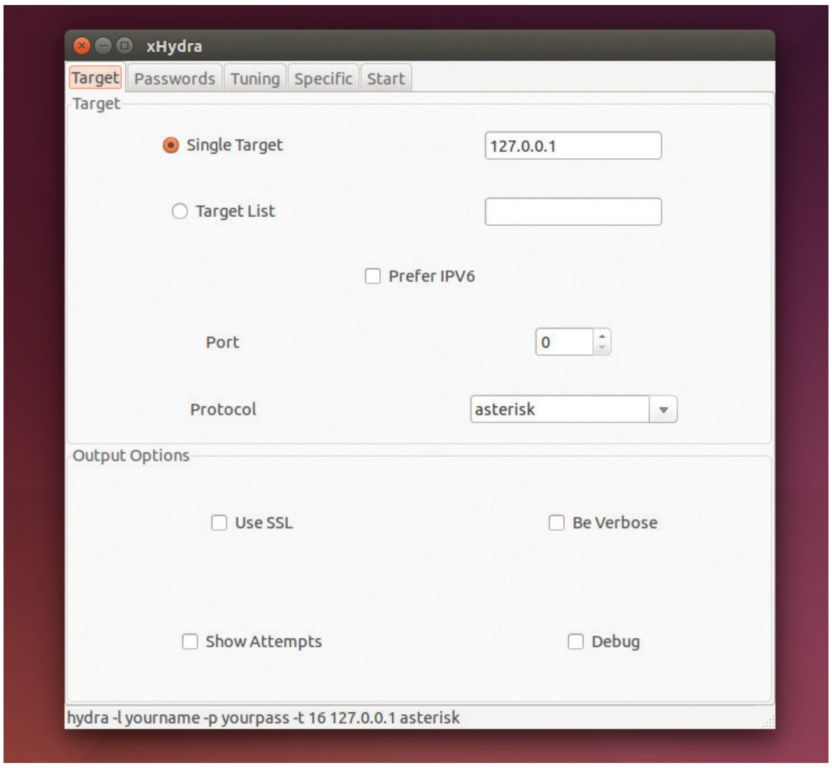
## Processing power

The faster your computer can hash passwords, the more you can try in a given amount of time, and therefore the better chance you have of cracking the password. In this article, we've used *John The Ripper* because it's an open source tool that's available on almost all Linux platforms. However, it's not always the best option. *John* runs on the CPU, but password hashing can be run really efficiently on graphics cards.

*Hashcat* is password cracking program that runs on graphics cards, and on the right hardware can perform much better than *John*. Specialised password cracking computers usually have several high-performance GPUs and rely on these for their speed.

You probably won't find *Hashcat* in your distro's repositories, but you can download it from **www.hashcat.net** (it's free as in zero cost, but not free as in free software). It comes in two flavours: **ocl-Hashcat** for OpenCL cards (AMD), and **cuda-Hashcat** for Nvidia cards.

Raw performance, of course, means very little without finesse, so fancy hardware with GPU crackers means very little if you don't have a good set of words and rules.



```
hydra -l yourname -p yourpass -t 16 127.0.0.1 asterisk
```

Hydra can be used to try and guess passwords on network services, although this is much slower than cracking hashes locally.

In the previous example, you probably found *John* cracked most of the passwords very quickly. This is because they were all common passwords. Since *John* works by checking a dictionary of words, common passwords are very easy to find.

John comes with a word list that it uses by default. This is quite good, but to crack more and more secure passwords, you then need a word list with more words. People who crack passwords regularly often build their own word lists over years, and they can come from many sources. General dictionaries are good places to start (which languages you pick will depend on your target demographic), but these don't usually contain names, slang or other terms.

Crackers regularly steal passwords from organisations (often websites) and post them online. These password leaks may contain thousands or even millions of passwords, so these are a great source of extra words. To search out even more elusive words, crackers turn to web scrapers and other tools to find sequences of characters that are used. There are some good sources of words at **https://wiki.skullsecurity.org/Passwords**, while good word lists are often sold (such as **https://crackstation.net/buy-crackstation-wordlist-password-cracking-dictionary.htm**, which is pay-what-you-want). The latter has about 1.5 billion words. Larger word lists are available, but often for a fee.

With John, you can use a custom word list with the **--wordlist=<filename>** option. For example, to check passwords using your system's dictionary, use:

```
rm ~/.john/john.pot
john --wordlist=/usr/share/dict/words md5s-short
```

This should work on most Debian-based systems, but on other distros, the **words** file may be in a different place. The first line deletes the file that contains the cracked passwords. If you don't run this, it won't bother trying to crack anything, as it already has all the passwords. The regular dictionary isn't as good as *John The Ripper*'s dictionary, so this won't get all the passwords.

## Mangling words

Secure services often place rules on what passwords are allowed. For example, they might insist on upper and lower case letters as well as numbers or punctuation. In general, people won't add these randomly, but put them in words in specific ways. For example, they might add a number to the end of a word, or replace letters in a word with punctuation that looks similar (such as **a** with **@**).

*John The Ripper* provides the tools to mangle words in this way, so that we can check these combinations from a normal word list.

For this example, we'll use the password file from **www.linuxvoice.com/passwords**, which contains the passwords: password, Password, PASSWORD, password1, p@ssword, P@ssword, Pa55w0rd, p@55w0rd. First, create a new text file called **passwordlist** containing just:

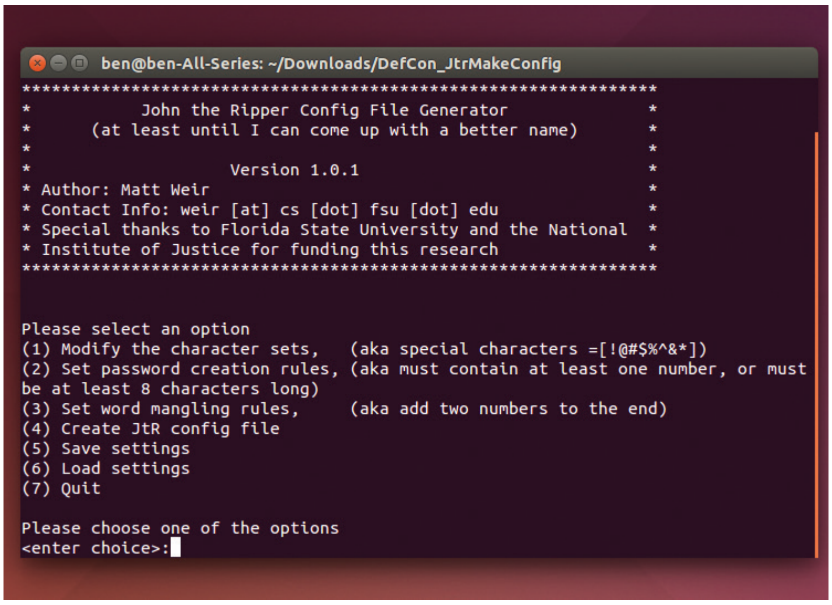
```
password
```

This will be the dictionary, and we'll create rules that crack all the passwords based of this one root word.

Rules are specified in the **john.conf** file. By default, **john** uses the configuration files in **~/.john**, so you'll need to create that file in a text editor. We'll start by adding the lines:

```
[List.Rules:Wordlist]
:
c
```

The first line tells **john** what mode you want to use the rules for, end every line below that is a rule (we'll



```
ben@ben-All-Series: ~/Downloads/DefCon_JtrMakeConfig
******************************************************************
*           John the Ripper Config File Generator                *
*      (at least until I can come up with a better name)         *
*                                                                *
*                     Version 1.0.1                              *
* Author: Matt Weir                                              *
* Contact Info: weir [at] cs [dot] fsu [dot] edu                 *
* Special thanks to Florida State University and the National   *
* Institute of Justice for funding this research                 *
******************************************************************


Please select an option
(1) Modify the character sets,   (aka special characters =[!@#$%^&*])
(2) Set password creation rules, (aka must contain at least one number, or must
be at least 8 characters long)
(3) Set word mangling rules,     (aka add two numbers to the end)
(4) Create JtR config file
(5) Save settings
(6) Load settings
(7) Quit

Please choose one of the options
<enter choice>:
```

A text-menu driven tool for creating *John The Ripper* config files is available from **https://sites.google.com/site/reusablesec2/jtrconfiggenerator**.

add more in a minute). The **:** just tells *John* to try the word as it is, no alterations, while **c** stands for capitalise, which makes the first character of the word upper case. You can try this out with:

```
john passwords.md5 --wordlist=passwordlist --rules
```

You should now crack two of the passwords despite there only being one word in the dictionary. Let's try and get a few more now. Add the following to the config file:

```
u
$[0-9]
```

The first line here makes the whole word upper case.

## How passwords work

Passwords present something of a computing conundrum. When people enter their password, the computer has to be able to check that they've entered the right password. At the same time though, it's a bad idea to store passwords anywhere on the computer, since that would mean that any hacker or malware might be able to get the passwords file and then compromise every user account.

Hashing (AKA one-way encryption) is the solution to this problem. Hashing is a mathematical process that scrambles the password so that it's impossible to unscramble it (hence one-way encryption).

When you set the password, the computer hashes it and stores the hash (but not the password). When you enter the password, the computer then hashes it and compares this hash to the stored hash. If they're the same, then the computer assumes that the passwords are the same and therefore lets you log in.

There are a few things make a good hashing algorithm. Obviously, it should be impossible to reverse (otherwise it's not a hashing algorithm), but other than this, it should minimise the number of collisions. This is where two different things produce the same hash, and the computer would therefore accept both as valid. It was a collision in the MD5 hashing algorithm that allowed the *Flame* malware to infiltrate the Iranian Oil Ministry and many other government organisations in the Middle East.

Another important thing about good hashing algorithms is that they're slow. That might sound a little odd, since generally algorithms are designed to be fast, but the slower a hash is, the harder it is to crack. For normal use, it doesn't make much difference if the hash takes 0.000001 seconds or 0.001 seconds, but the latter takes 1,000 times longer to crack.

You can get a reasonable idea of how fast or slow an algorithm is by running **john --test** to benchmark the different algorithms on your computer. The fewer checks per second, the slower it will be for an attacker to break any hashes using that algorithm.

On the second line, the **$** symbol means append the following character to the password. In this case, it's not a single character, but a class of characters (digits), so it tries ten different words (password0, password1… password9).

To get the remaining passwords, you need to add the following rules to the config file:

```
csa@
sa@so0ss5
css5so0
```

The rule **s<character1><character2>** replaces all occurrences of **character1** with **character2**. In the above rules, this is used to switch **a** for **@** (**sa@**), **o** for **0** (**so0**) and **s** for **5** (**ss5**). All of these are combination rules that build up the final word through more than one alteration.

## Limitations of cracking rules

The language for creating rules isn't very expressive. For example, you can't say: 'try every combination of the following rules'. The reason for that is speed. The rules engine has to be able to run thousands or even millions of times per second while not significantly slowing down the hashing.

You've probably guessed by now that creating a good set of rules is quite a time-consuming process. It involves a detailed knowledge of what patterns are commonly used to create passwords, and an understanding of the archaic syntax used in the rules engines. It's good to have an understanding of how they work, but unless you're a professional penetration tester, it's usually best to use a pre-created rule list.

The default rules with *John* are quite good, but there are some more complex ones available. One of the best public ones comes from a DefCon contest in 2010. You can grab the ruleset from the website: **http://contest-2010.korelogic.com/rules.html**.

You'll get a file called **rules.txt**, which is a *John The Ripper* configuration file, and there are some usage examples on the above website. However, it's not designed to work with the default version of *John The Ripper*, but a patched version (sometimes called **-jumbo**). This isn't usually available in distro repositories, but it can be worth compiling it because it has more features than the default build. To get it, you'll need to clone it from GitHub with:

```
git clone https://github.com/magnumripper/JohnTheRipper
cd JohnTheRipper/
```

There are a few options in the install procedure, and these are documented in **JohnTheRipper/doc/Install**. We compiled it on an Ubuntu 14.04 system with:

```
cd JohnTheRipper/src
./configure && make -s clean && make -sj4
```

This will leave the binary **JohnTheRipper/run/john** that you can execute. It will expect the **john.conf** file (which can be the file downloaded from KoreLogic) in the same directory.

If you don't want to compile the **-jumbo** version of *John*, you can still use the rules from KoreLogic, you'll just have to integrate them into a **john.conf** file by

## Salting

For hashing to work, every time a password is hashed, it has to produce the same result. This plays into the hands of crackers because it means that if they have a list of password hashes they've stolen, they can check every word from their word list against all of them at the same time. It also means that they could create lookup tables with the hashed value of common words to speed up the process of cracking passwords (these are sometimes known as rainbow tables).

To stop this, salts are sometimes used. Salts are small amounts of additional data that are added to the plain text before hashing. They're stored alongside the hash so that the same salt is used on the same password. Crackers who get access to the hashes will also usually get access to the salts, but it means they have to crack every password individually rather than working against the whole lot simultaneously.

At the very least, salting will slow an attacker down by the factor of the number of hashes they have. If a cracker steals a thousand password hashes, it will be at least a thousand times slower to crack them if they are salted (though it could be less if they can use rainbow tables to speed up the crack).

To be secure, salts have to be randomly generated. In WPA Wi-Fi security, the network name (SSID) is used as a salt for the password. This is useful because it's automatically known to both parties. However, SSIDs aren't unique, and many are quite common. It's possible to download lookup tables for many of the most common SSIDs against many passwords. A traditional crack against the hashing in WPA is quite slow, because WPA uses 4,096 rounds of SHA1. The lookup tables sidestep this because the hashing has already been done.

It's important to use a random salt to stop this sort of attack, and it's important to use an obscure SSID on your Wi-Fi network to avoid falling victim.

You can download the lookup tables and a list of SSIDs from **www.renderlab.net/projects/WPA-tables**.

hand first. There are a lot of rules, so you'll probably want to pick out a few, and copy them into the **john.conf** file in the same way you did when creating the rules earlier, and omit the lines with square brackets.

As you've seen, cracking passwords is part art and part science. Although it's often thought of as a malicious practice, there are some real positive benefits of it. For example, if you run an organisation, you can use cracking tools like *John* to audit the passwords people have chosen. If they can be cracked, then it's time to talk to people about computer security. Some companies run periodic checks and offer a small reward for any employee whose password isn't cracked. Obviously, all of these should be done with appropriate authorisation, and you should never use a password cracker to attack someone else's password except when you have explicit permission.

*John The Ripper* is an incredibly powerful tool whose functionality we've only just touched on. Unfortunately, its more powerful features (such as its rule engine) aren't well documented. If you're interested in learning more about it, the best way of doing this is by generating hashes and seeing how to crack them. It's easy to generate hashes by simply creating new users in your Linux system and giving them a password; then you can copy the **/etc/shadow** file to your home directory and change the owner with:

```
sudo cp /etc/shadow ~
sudo chown <username> ~/shadow
```

Where **<username>** is your username. You can then run *John* on the shadow file. If you've got a friend who's interested in cracking as well, you could create challenges for each other (remember to delete the lines for real users from the shadow file though!). Alternatively, you can try our shadow file for the latest in our illustrious series of competitions.

So, what does a secure password look like? Well, it shouldn't be based on a dictionary word. As you've seen, word mangling rules can find these even if you've obscured it with numbers or punctuation. It should also be long enough to make brute force attacks impossible (at least 10 characters). Beyond that, it's best to use your own method, because any method that becomes popular can be exploited by attackers to create better word lists and rules.

**Ben Everard is the co-author of the best-selling *Learn Python With Raspberry Pi*, and is working on a best-selling follow-up called *Learning Computer Architecture With Raspberry Pi*.**

## COMPETITION

### Put your skills to the test with the Linux Voice password cracking competition

We've created 100 users on our Linux box using a range of passwords. Linux distros store the password hashes in the **/etc/shadow** file, and you can get ours from **www.linuxvoice.com/passwords**.

Some are easy, some are hard. Some are real passwords we've extracted from dumps, some we've generated using password generators, others we created by hand (that might be a clue). Oh, and incidentally, we like the XKCD web comic.

Your task is to crack as many passwords as possible. They're in the standard SHA512 format (*John The Ripper* – and most other password crackers – will detect this automatically). This is quite a slow algorithm, and some of the passwords are quite complex, so we don't expect anyone to guess all of them. The prize will go to the person who manages to crack the most. If two people crack the same number, the prize will go to whoever sends in their entry first.

To enter, just send a plain text file with a list of unhashed passwords that you've cracked from the **competition-shadow** file to **ben@linuxvoice.com**. The deadline for entries is 25 October 2014.

Happy cracking!

LINUX VOICE TUTORIAL

# CYRUS: BUILD YOUR OWN EMAIL SERVER

JOHN LANE

Don't trust Google? We'll help you navigate the sea of acronyms to build your own mailserver.

**WHY DO THIS?**

- Take control of your email provision.
- Stop outside agencies from scanning the content of your emails.
- Get webmail without advertising.

You can't beat the convenience and ease of use offered by Gmail. But unfortunately, all that free storage comes at a price: your privacy. Spam, intrusive adverts and snooping from unnamed government agencies are the inevitable downside of using someone else's service for free. So why not build your own email server including anti-spam, anti-virus and webmail?

You can use your own server to retrieve messages from other mailservers, such as those provided by internet service providers, or other services like those from Google and Yahoo. But you don't need to rely on others if you have your own server. If you have a domain name that you control, and if you can give your server a static public IP address then you can receive email directly.

We're going to implement a sealed server, which means that users cannot log in to it. They have email accounts that are only accessible using client applications that connect to the server using IMAP, the Internet Message Access Protocol (we could, but won't, also use the older Post Office Protocol, POP).

"Why not build your own email server, including anti-spam, anti-virus and webmail?"

At the heart of the system is the IMAP server, *Cyrus*. This accepts messages using a protocol called the Local Mail Transfer Protocol, or LMTP, and stores them in mailboxes – it's a mail delivery agent. Users can access their mail by connecting to the server using any IMAP-capable email client application.

You will need a, preferably new, server for this project and you'll need root access to it. Our examples use Arch Linux, and we created a new virtual server.

Begin by installing *Cyrus* (build the Arch User Repository package first – see the boxout below-right):

```
$ pacman -U ~build/cyrus-imapd/cyrus-imapd-2.4.17-5-x86_64.pkg.tar.xz
```

The default configuration writes data to **/var/imap** and user mailboxes to **/var/spool/imap**. You can change this if you prefer another location; we'll configure our server to use **/srv/mail/cyrus** to illustrate this. If you follow suit, you can also delete the default locations:

```
rm -r /var/spool/imap /var/imap
```

Some command line tools are installed to **/usr/lib/cyrus/bin** so it's worth extending your **PATH** (do it in **/etc/profile** to make this permanent):

```
export PATH="$PATH":/usr/lib/cyrus/bin
```

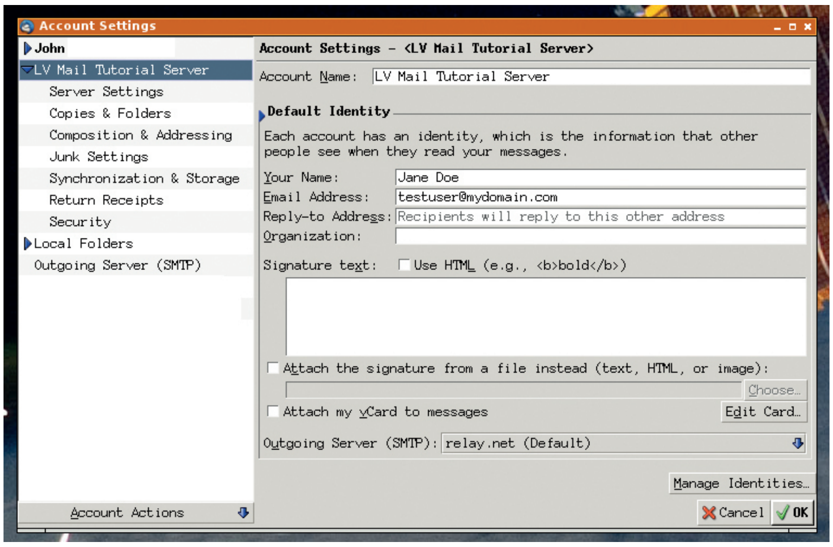
There are two configuration files, and the first of these is **/etc/cyrus/cyrus.conf**. It defines the services that the server will offer, and the default file is generally acceptable unless, like us, you want to change the data path. This requires one entry in the file to be altered:

```
lmtpunix cmd="lmtpd" listen="/srv/mail/cyrus/socket/lmtp" preform=0
```

The **listen** argument points to the Unix domain socket where the server accepts LMTP protocol connections. We change this to be in a subdirectory of our chosen data path. You can also take this opportunity to disable unwanted services; we commented out **pop3** and **pop3s** because we plan to offer IMAP-only access.

The second file, **/etc/cyrus/imapd.conf**, configures the IMAP server and needs to be written from scratch. The following example will get you started, but you may want to read the documentation and configure it to meet your needs.

```
configdirectory: /srv/mail/cyrus
partition-default: /srv/mail/cyrus/mail
admins: cyrus
sasl_pwcheck_method: saslauthd
sasl_saslauthd_path: /var/run/saslauthd/mux
sasl_mech_list: PLAIN
allowplaintext: yes
altnamespace: yes
unixhierarchysep: yes
virtdomains: userid
```



Account Settings - <LV Mail Tutorial Server>

Account Name: LV Mail Tutorial Server

Default Identity

Each account has an identity, which is the information that other people see when they read your messages.

Your Name: Jane Doe
Email Address: testuser@mydomain.com
Reply-to Address: Recipients will reply to this other address
Organization:
Signature text: ☐ Use HTML (e.g., <b>bold</b>)

☐ Attach the signature from a file instead (text, HTML, or image):

☐ Attach my vCard to messages

Outgoing Server (SMTP): relay.net (Default)

You can give your test account a meaningful name and enter your own name in the identity section.

```
defaultdomain: mydomain.com
hashimapspool: true
sieve_admins: cyrus
sievedir: /srv/mail/cyrus/sieve
```

This tells *Cyrus* to use **/srv/mail/cyrus** for its configuration and, within that, a **mail** subdirectory where it should store mail. Virtual domains allows domain-specific mailboxes – you can have accounts for **alice@example-one.com** and **alice@example-two.com**. The **defaultdomain** is the domain that unqualified user accounts, like "alice", belong to.

To improve the end-user experience, we set **altnamespace** so that users' email folders appear alongside, rather than within, their inbox, and **unixhierarchysep** delimits mail folders with slashes instead of the default, which is to use a period.

## SASL

Our configuration uses SASL for authentication. This is the Simple Authentication and Security Layer, and was automatically installed as a dependency of the IMAP server. We just use the default configuration here, which passes plain-text passwords to the **saslauthd** daemon that, in the default configuration on Arch Linux, uses PAM for authentication. This is acceptable for a test system, but you should consider configuring SASL to use more secure methods that satisfy your own security requirements.

So, create a test account for testing and verify that SASL can authenticate it. The default SASL configuration authenticates system users so we use a nobody account that can be  authenticated but cannot be used to log in to the server.

```
$ useradd -c 'Test email account' -u 99 -o -g nobody -d /dev/null -s /bin/false testuser
$ echo testuser:testpass | chpasswd
```

Start **saslauthd** (also enable it so that it starts on boot) and test that SASL authentication works for the new test user:

```
$ systemctl enable saslauthd
$ systemctl start saslauthd
$ testsaslauthd -u testuser -p testpass
0: OK "Success."
```

The installation also created a **cyrus** user, and the server's processes run as this user. We can also use it for administrative tasks if we set its home directory, shell and password:

```
$ usermod -s /bin/bash -d /srv/mail/cyrus cyrus
$ echo cyrus:cyrus | chpasswd
```

To complete the configuration, make the required directories and build the IMAP folders:

```
$ mkdir -p -m 750 /srv/mail/cyrus/mail
$ chown -R cyrus:mail /srv/mail/cyrus
$ su cyrus -c 'mkimap /etc/cyrus/imapd.conf'
```

Now start the server

```
$ systemctl enable cyrus-master
$ systemctl start cyrus-master
```

Test IMAP access for the test user

```
$ telnet localhost imap
. login testuser testpass
. logout
```

If everything went well, the server responses will begin with *** OK**. You can now set up your email client to connect to the IMAP account, but it doesn't have any folders yet. The **cyradm** tool is used to create mailboxes, and the minimum is an inbox:

```
$ su cyrus -c 'cyradm -u cyrus -w cyrus localhost
localhost.localdomain> cm user/testuser
```

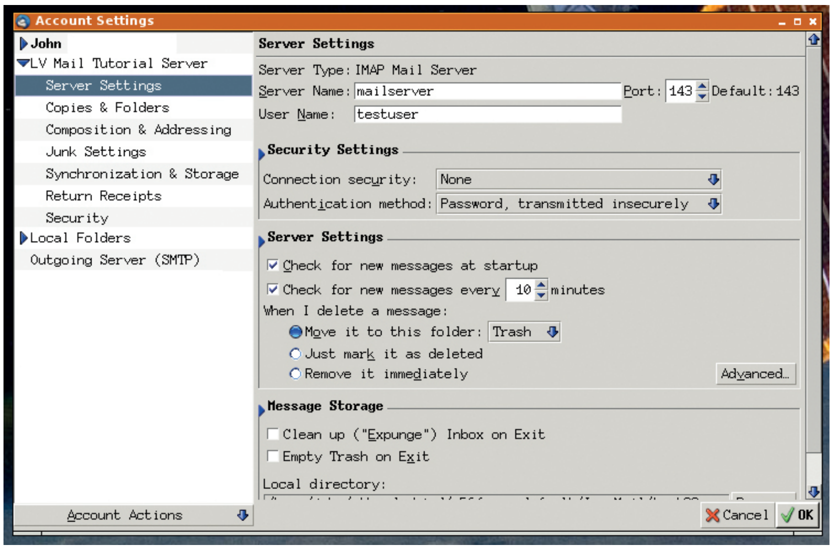
You can then use your email client to create subfolders, or you can use **cyradm** – **cm** creates mailboxes (folders) and **lm** lists them:

```
localhost.localdomain> cm user/testuser/Sent
localhost.localdomain> lm
user/testuser (\HasChildren)
user/testuser/Sent (\HasNoChildren)
user/testuser/Trash (\HasNoChildren)
```

You can now send a message to the test user. Create a test message in a file (call it **testmessage**) with the following contents (the empty line is required – it marks the beginning of the message body).

```
From: Test Message <test@example.com>
Subject: This is a test message
This is a basic test e-mail message
```

To send the message into *Cyrus*, use the **deliver** tool



Server Settings
Server Type: IMAP Mail Server
Server Name: mailserver Port: 143 Default: 143
User Name: testuser

Security Settings
Connection security: None
Authentication method: Password, transmitted insecurely

Server Settings
[✓] Check for new messages at startup
[✓] Check for new messages every 10 minutes
When I delete a message:
[✓ Move it to this folder: Trash]
Just mark it as deleted
Remove it immediately

Message Storage
Clean up ("Expunge") Inbox on Exit
Empty Trash on Exit
Local directory:

You can specify the server by its host name or IP address. The username is the IMAP "testuser" account that we set up on the server.

**LV PRO TIP**

*Cyrus* documentation is available at **http://cyrusimap.org/docs/cyrus-imapd**.

## A virtual mailserver

We used Linux Containers to create a virtual server to implement our mailserver on. Here's what we did. As root, on any host machine (ours runs Arch Linux):

```
lxc-create -n mailserver -t archlinux -- -P dhcpcd,openssh,wget --ewnable_units dhcpcd,sshd.socket -r mysecret
lxc-start -n mailserver
```

You can then log in with **ssh root@mailserver** using **mysecret** as the password.

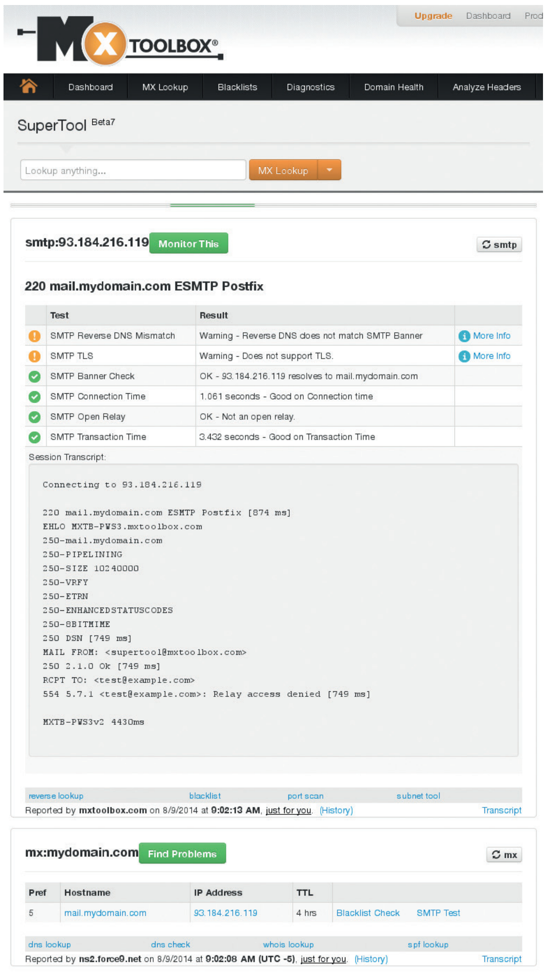
Some of the packages that we will use aren't in the repositories, but they can be built from the Arch User Repository, AUR. We created a build account on our new server for building these packages.

```
$ pacman -S base-devel devtools
$ useradd -c 'Build Account' -m -g users -d /home/build -s /bin/bash build
$ echo build:build | chpasswd
$ echo 'build ALL=(ALL) NOPASSWD: ALL' >> /etc/sudoers
```

To build a package, log on as the "build" user, download and extract the package's AUR tarball and use **makepkg** to build it. Further instructions are available on the Arch Linux website. Here is an example:

```
$ wget https://aur.archlinux.org/packages/cy/cyrus-imapd/cyrus-imapd.tar.gz
$ tar xf cyrus-imapd.tar.gz
$ cd cyrus-imapd
$ makepkg -s
```

MXToolbox.com can test your server from outside…



smtp:93.184.216.119

220 mail.mydomain.com ESMTP Postfix

| | Test | Result | |
|---|---|---|---|
| ! | SMTP Reverse DNS Mismatch | Warning - Reverse DNS does not match SMTP Banner | More Info |
| ! | SMTP TLS | Warning - Does not support TLS. | More Info |
| ✓ | SMTP Banner Check | OK - 93.184.216.119 resolves to mail.mydomain.com | |
| ✓ | SMTP Connection Time | 1.061 seconds - Good on Connection time | |
| ✓ | SMTP Open Relay | OK - Not an open relay. | |
| ✓ | SMTP Transaction Time | 3.432 seconds - Good on Transaction Time | |

Session Transcript:

```
Connecting to 93.184.216.119

220 mail.mydomain.com ESMTP Postfix [874 ms]
EHLO MXTB-PWS3.mxtoolbox.com
250-mail.mydomain.com
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 DSN [749 ms]
MAIL FROM: <supertool@mxtoolbox.com>
250 2.1.0 Ok [749 ms]
RCPT TO: <test@example.com>
554 5.7.1 <test@example.com>: Relay access denied [749 ms]

MXTB-PWS3v2 4430ms
```

reverse lookup blacklist port scan subnet tool

Reported by **mxtoolbox.com** on 8/9/2014 at **9:02:13 AM**, just for you. (History) Transcript

mx:mydomain.com

| Pref | Hostname | IP Address | TTL | |
|---|---|---|---|---|
| 5 | mail.mydomain.com | 93.184.216.119 | 4 hrs | Blacklist Check SMTP Test |

dns lookup dns check whois lookup spf lookup

Reported by **ns2.force9.net** on 8/9/2014 at **9:02:08 AM (UTC -5)**, just for you. (History) Transcript

and then check your email client for the message.

```
deliver testuser < testmessage
```

That completes the configuration of the IMAP server. It's ready to receive mail and can serve it to users' email clients, but nothing is yet being sent to it.

The simplest way to get mail into your server is to fetch it from another one. A daemon known as a Mail Retrieval Agent (MRA) can fetch mail from remote IMAP or POP mailboxes such as your Gmail account. The MRA that we'll use is called *Fetchmail*:

```
$ pacman -S fetchmail
```

*Fetchmail* takes instructions from **/etc/fetchmailrc**, which must be set with **0700** permissions. The file begins with global settings and defaults and it's here that we tell *Fetchmail* to deliver all mail to our server's LMTP socket.

```
defaults
 smtphost "/srv/mail/cyrus/socket/lmtp"
 smtpaddress mydomain.com
```

Specify the same domain here as the **defaultdomain** in **/etc/cyrus/imapd.conf**. Without this, any unqualified usernames will have **localhost** appended and the mailserver won't recognise them.

With the defaults configured, what remains is to provide blocks for each remote server that we wish to fetch from. You can fetch messages from many remote accounts and deliver them to any configured local email account. Here is an example that fetches from Gmail:

```
poll poll imap.gmail.com protocol imap
 user alice@gmail.com there pass abc123 is alice here
 user alice_other@gmail.com there pass secretword is alice here
 user jane.doe@gmail.com there pass secretword is jane here
```

and similar examples for Yahoo and Microsoft mail accounts:

```
poll pop.mail.yahoo.com protocol pop3
 user johndoe there pass mypassword is john here ssl
poll pop3.live.com protocol pop3
 user bob@hotmail.com there pass 123abc is bob here ssl
```

You can fetch mail on demand (the optional **-v** makes it verbose):

```
$ fetchmail -v -f /etc/fetchmailrc
```

Or, what you will most likely want to do is start it as a daemon that regularly polls for available messages. The daemon on Arch Linux runs as the **fetchmail** user and requires that it owns the **/etc/fetchmail** file. We can start the daemon:

```
$ chown fetchmail /etc/fetchmailrc
$ systemctl enable fetchmail
$ systemctl start fetchmail
```

*Fetchmail* will poll at an interval defined by its **systemd** unit. On Arch Linux this is 900 seconds (15 minutes). You can use the **SIGHUP** signal to instruct the daemon to poll on demand.

```
$ pkill -USR1 fetchmail
```

We now have a working email server that fetches email from other external mailservers. We can improve upon that by having mail sent to us.

## Join the Postal Union

Email is sent across the internet by Mail Transfer Agents. These aren't trench-coated sleuths but network services that converse using the Simple Mail Transfer Protocol, or SMTP. We need to join in this conversation so that we can receive email – we need our own Mail Transfer Agent, and we'll use *Postfix*; it's a straightforward installation from the repository:

```
$ pacman -S postfix
```

*Postfix* is controlled by a configuration file called **main.cf**, and you'll find it in **/etc/postfix**. It contains a large number of options but most of the defaults are acceptable for our needs.

Our mailserver supports mail accounts for multiple domains, so we'll configure *Postfix* to recognise these Virtual Mailbox Domains and deliver any mail received for them into our mailserver's LMTP interface.

```
virtual_mailbox_domains = mydomain.com myotherdomain.
co.uk
virtual_transport = lmtp:unix:/srv/mail/cyrus/socket/lmtp
```

Start the *Postfix* server and tail its journal so that you can see what it does:

```
$ systemctl enable postfix
$ systemctl start postfix
$ journalctl -f -u postfix &
```

You can use Telnet to send a test message. You should be able to see it in your email client as soon as you've sent it.

```
$ telnet localhost smtp
```

**LV PRO TIP**

All mail users created with **useradd** can have the same UID.

```
EHLO example.com
MAIL FROM:bob@example.com
RCPT TO:testuser@mydomain.com
DATA
From: Bob <bob@example.com>
Subject: This is a test message

This is a test SMTP message
.
QUIT
```

The test confirms that our server can deliver emails received for our domains over SMTP but, before anything can be sent to it, it needs a static public IP address and the domains' DNS records need to be updated with that address so that other Mail Transfer Agents can find it.

## Speak to me

Your internet service provider allocates you a public IP address for your connection. You will need to ensure this is static. If in any doubt, contact your ISP. We'll use the public address of **example.com** in our examples, which is **93.184.216.119**.

You'll need to open the SMTP port (25) on your perimeter firewall and configure a NAT translation to connect that port to your mailserver. How you do this will depend on what networking hardware you have. The following examples assume that **93.184.216.119:25** reaches your *Postfix* SMTP interface. Once you have a static IP address that connects to your server, you should configure your domains' DNS records. How you do this depends on the tools provided by your DNS provider, usually the registrar of your domains.

You need to configure two records: an address record (**A** record) that points to your static public IP address, and a mail exchange record (**MX** record) that points to the **A** record. DNS records have four fields but each record only uses three of them. Configure the **A** record like this:

```
Left field: mail
Type: A
Priority: <blank>
Right field: 93.184.216.119
```

and the **MX** record like this:

```
Left field: <blank>
Type: MX
Priority: 5
Right field: mail
```

The **MX** record references the **A** record by name (we imaginatively chose to call ours "mail"). The **A** record gives the IP address of the server. Both records are required – the **MX** record cannot contain an IP address. Remember that DNS updates can take up to 48 hours to take effect.

You can define multiple **MX** records and use the **priority** field to order them. If you do this then delivery is attempted using each **MX** record in ascending priority order until one succeeds. If delivery fails then the message is returned to the sender (it's bounced).

You could use multiple **MX** records to have mail delivered to a mailbox at your ISP if your own server is offline. Your server's Mail Retrieval Agent, *Fetchmail*, could then retrieve any such mail when it comes back online.

You can perform various tests to ensure that your server can accept mail. You can probe your port (**https://www.grc.com/x/portprobe=25**) and test your **MX** records, either online with **http://mxtoolbox.com** or on the command line with **dig**:

```
$ dig +short MX mydomain.com
5 mail.mydomain.com.
$ dig +short A mail.mydomain.com
93.184.216.119
```

Now that your SMTP server is on the internet you need to make sure it's properly configured, otherwise it won't be long before spammers find it and start using it to distribute their wares. You can use **http://mxtoolbox.com/SuperTool.aspx** to check how your server responds to the outside world and confirm that you aren't offering an open relay to spammers; **https://www.wormly.com/test_smtp_server** lets you send test emails into your server.

We've configured enough to receive, store and serve email to multiple users over IMAP. Next time, we'll start filtering out unwanted messages, like anything containing spam or viruses or even just mails from people we just don't like. We'll also let our users send email, because it's good to talk. LV

**John Lane is a technology consultant with a penchant for Linux. He helps new businesses and start-ups make the most of open source software.**

## The right protocol

There are quite a few protocols involved in the transmission of email.

- **SMTP** is what drives email. The mailserver's MTA makes connections using SMTP: it listens on port 25 for incoming messages and sends messages to port 25 on other MTAs. SMTP was originally specified by RFC821 back in 1982.
- **LMTP** is the Local Mail Transfer Protocol defined by RFC2033 used for local mail delivery within the same network. Our MDA, *Cyrus-IMAP*, accepts mail using LMTP through a Unix domain socket.
- **ESMTP**, Extended or Enhanced SMTP, defined by RFC5321, is a set of extensions to SMTP. They include STARTTLS, which is used to establish transport layer security. Because of this, it's common to see ESMTP used to describe SMTP over TLS.

Next month we will add a Message Submission Agent to our system that listens on port 587 for ESMTP connections. Message submission to this port is known as SMTP-MSA.

There used to be a secured form of SMTP called SMTPS or SMTP-Secured, that MTAs supported on port 465 but it was deprecated in favour of STARTTLS because this allows both insecure and secure connections over the same port.

Mail User Agents use POP, the Post Office Protocol (RFC1939) and IMAP, the Internet Message Access Protocol (RFC3501). They send email, ideally to the MSA on port 587, but more often to the MTA on port 25.

You can read the RFC specifications at **http://tools.ietf.org** if you want to understand more about these protocols.

### Common Ports

- **25** is for message transfer (SMTP-MTA).
- **110** is for POP.
- **143** is for IMAP.
- **465** was for SMTP-Secured (deprecated).
- **587** is for message submission (SMTP-MSA).
- **993** is for IMAP over SSL.

These assignments are specified by the Internet Assigned Numbers Authority (IANA). Although some MUAs and MTAs support the deprecated SMTP-Secured on port 465, this port has been reassigned to the URL Rendezvous Directory for SSM, which has nothing to do with email whatsoever.

### LV PRO TIP

You'll need an SASL back-end that can support fully qualified user names like **bob@example.com** to host accounts for domains other than the "defaultdomain".

LINUX VOICE TUTORIAL

# URWID: CREATE TEXT MODE INTERFACES

VALENTINE SINITSYN

Text-mode user interfaces do not belong to museums yet – find out why and craft one yourself.

**WHY DO THIS?**

- Create easy to use, lightweight interfaces.
- Rewrite dialog(1)-based shell scripts in Python.
- Learn Linux beyond the desktop.

Today, one can hardly imagine the PC without a graphical desktop. Even the smallest computers such as the Raspberry Pi have an HDMI port and a CPU powerful enough for a graphical environment. Text (or console) user interfaces (TUI) may feel like a weird artefact from ye olden days that fit a museum stand better than your monitor. Sure, you are unlikely to use a terminal to chat on Facebook (although you can surf the web with the *Links* browser if you wish), or write a report (*Latex* can award you with state-of-the-art documents). Nevertheless, console-based programs come in handy where you don't have graphics configured (in installers or setup tools) or work on slow connections (say, you SSH into your Raspberry Pi-based sensor somewhere in countryside available over a 2.75G cellular network only). Text interfaces are also often preferable for specialised applications, like point-of-sale terminals.

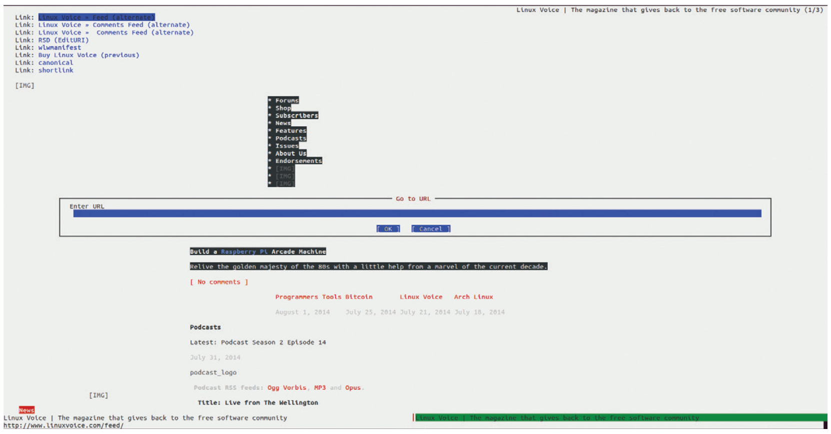
This tutorial is about making console interfaces in Python with the *Urwid* library. If you've ever done any programming with *Qt*, *GTK* or any other toolkit, you will find many concepts similar, but not the same. That's because *Urwid* is, strictly speaking, not a widget toolkit. It's a widget construction toolkit, and this subtle difference sometimes matters. It provides the elements of a user interface that you'd expect, like buttons or text input boxes. But many advanced widgets, say dialogs or drop-down menus, are missing (you do them yourself, and we'll show you how in a minute). There is also no straightforward way to set the "tab order" (ie how the focus moves with Tab key). This doesn't mean that *Urwid* is limited or primitive – it's a full-fledged library with mouse support, third-party IO loop integration and other services that you might expect from a mature toolkit – but it's a peculiarity to keep in mind when you program with it.

## Widget types

One task that a widget toolkit performs is calculating positions and screen space for widgets. This is not as simple as it may sound, and there's no one-size-fits-all recipe either. Some older libraries tended to avoid this job altogether, so if a label was too long to display, it was simply cut off.

*Urwid*'s approach is to introduce three types of widgets. The first one, "box", takes as much space as its container allocates; a top-level widget in *Urwid* application is always a box one. Flow widgets are given a number of columns to occupy, and are responsible for calculating the number of screen rows they need (as we are working in text mode, units are characters, and widget size is measured in rows and columns, not pixels). Fixed widgets are, er, fixed: they always occupy the same screen space regardless what is available, and they decide on their size themselves. A typical example of a flow widget is Text; common boxed widget is SolidFill, which fills an area with the given character and is useful for backgrounds. Fixed widgets are rare, and we won't discuss them.

There are also "decoration widgets" that wrap other widgets and alter their appearance or behaviour. In this way, flow widgets can be made boxed (for



```
                                       Linux Voice | The magazine that gives back to the free software community (1/3)
Link: Linux Voice » Feed (alternate)
Link: Linux Voice » Comments Feed (alternate)
Link: Linux Voice »  Comments Feed (alternate)
Link: RSD (EditURI)
Link: wlwmanifest
Link: Buy Linux Voice (previous)
Link: canonical
Link: shortlink

[IMG]

                                   * Forums
                                   * Shop
                                   * Subscribers
                                   * News
                                   * Features
                                   * Podcasts
                                   * Issues
                                   * About Us
                                   * Endorsements
                                   * [IMG]
                                   * [IMG]
                                   * [IMG]
      ┌──────────────────────────────────── Go to URL ────────────────────────────────────┐
      │ Enter URL                                                                          │
      │                                                                                    │
      │                                       [ OK ]  [ Cancel ]                           │
      └────────────────────────────────────────────────────────────────────────────────────┘
                          Build a Raspberry Pi Arcade Machine

                          Relive the golden majesty of the 80s with a little help from a marvel of the current decade.

                          [ No comments ]

                                     Programmers Tools Bitcoin       Linux Voice    Arch Linux

                                     August 1, 2014     July 25, 2014 July 21, 2014 July 18, 2014

                          Podcasts

                          Latest: Podcast Season 2 Episode 14

                          July 31, 2014

                          podcast_logo

                           Podcast RSS feeds: Ogg Vorbis, MP3 and Opus.

               [IMG]        Title: Live from The Wellington
  News
Linux Voice | The magazine that gives back to the free software community        Linux Voice | The magazine that gives back to the free software community
http://www.linuxvoice.com/feed/
```

There are TUI eqiuvalents for many graphical programs, including browsers.

## In a timely manner

The main loop is not only the dispatcher of events, but also a timer. These two roles may seem distant, but they are closely related if you descend to the system calls level.

We won't go that deep here, but instead will see how to use timers in *Urwid*. Actually, it's quite simple, and the API resembles JavaScript's **window.setTimeout()**:

```
def callback(main_loop, user_data):
            # I'm to be called in 10 seconds
handle = main_loop.set_alarm_in(10,
     callback, user_data=[])
```

**user_data** is for passing arbitrary values to your callback; if you don't need it, simply omit the argument. There is also **set_alarm_at()**, which schedules an alarm at the given moment. If you don't need an alarm anymore, you can remove it with:

```
main_loop.remove_alarm(handle)
```

Alarms in *Urwid* are not periodic, so there is no need to remove the alarm that was already triggered.

instance, with Filler, which fills rows left unused by its child) or vice versa (see BoxAdapter). All of these types are visually summarised in the "Included Widgets" section of the *Urwid* manual (**http://urwid.org/manual**).

Sometimes you misuse widgets and put a box one where a flow widget is expected, or whatever. *Urwid* is not very friendly in this case, and all you get is a cryptic **ValueError** exception:

```
... Few other calls here ...
      File "/path/to/urwid/widget.py", line 1004, in
render
  (maxcol,) = size
ValueError: too many values to unpack
```

It originates from the way widgets are rendered. You don't need to dig into details of this backtrace, just remember that if you see it, you've probably missed a decoration widget.

## Hello, Urwid world!

It's time to write some code. Like many other (if not all) UI frameworks, *Urwid* is built around the main loop, represented by the MainLoop class. This loop dispatches events such as key presses or mouse clicks to the widget hierarchy rooted at the topmost box widget, passed as the first argument to the MainLoop constructor (and available later as a 'widget' attribute on the main loop object). In this way, a simplest *Urwid* program might look like this:

```
from urwid import MainLoop, SolidFill
mainloop = MainLoop(SolidFill('#'))
mainloop.run()
```

This will fill the screen with hashmarks. The **run()** method is where the main loop starts. To terminate it, raise the **ExitMainLoop** exception:

```
def callback(key):
    raise ExitMainLoop()
mainloop = MainLoop(SolidFill('#'),
    unhandled_input=callback)
```

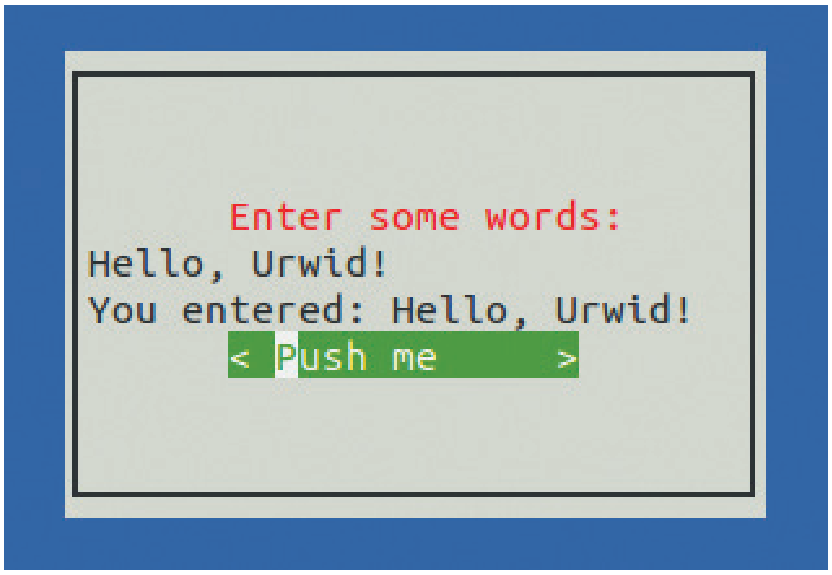
**unhandled_input** callback is executed for any event that is not handled by the topmost widget (or its descendants). Since **SolidFill()** doesn't respond to keypresses, any key will stop the program. You can check this yourself – just make sure you have installed *Urwid* with your package manager (it's called **python-urwid** or similar).

## Add some colour

Black and white text is boring. *Urwid* can paint colours, but it needs a palette first:

```
single_color = [('basic', 'yellow', 'dark blue')]
mainloop = MainLoop(AttrMap(SolidFill('#'),
    'basic'), palette=single_color)
```

Here, the palette contains a single colour: yellow text on a blue background. You can define a palette with as many colours as you want, but keep in mind that not all colours (and attributes) are supported by all terminals. If you don't target a specific environment, it is better to stick to "safe" colours, as defined in the "Display Attributes" section of the *Urwid* manual.



```
    Enter some words:
Hello, Urwid!
You entered: Hello, Urwid!
    < Push me        >
```

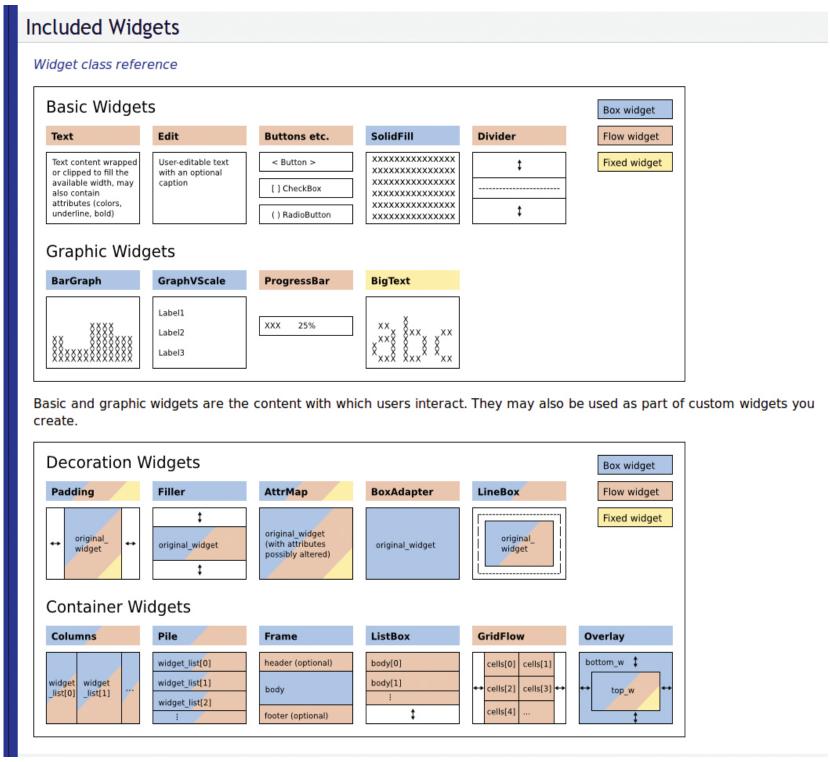
Our first *Urwid* program: basic, but fully functional.

The **palette = keyword** argument installs the palette for your application, but the **AttrMap** decoration widget is where the colour is actually applied. **'basic'** serves as an identifier, and can be anything you want.

## Let's open windows

Programs usually interface with users via some dialog windows. In text mode, they look like framed rectangular areas, so let's create one. To make things more interesting, we'll also include a few basic widgets. A blue background can be created with **SolidFill(' ')** the usual way (let's creatively call this widget 'background'). To create a framed area, we can use the **LineBox()** decoration widget (don't forget to import widgets from the **urwid** package as they appear in the text):

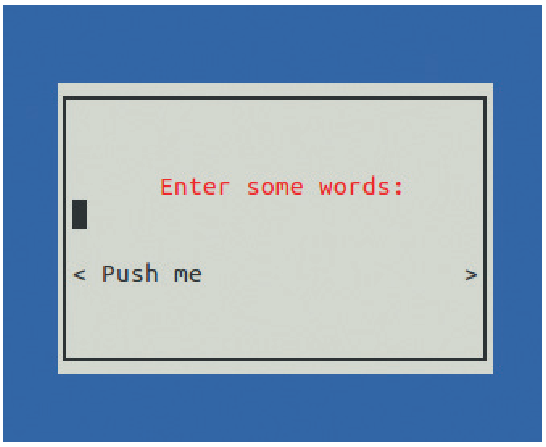
The *Urwid* manual has a neat refresher for widget types and more.



Included Widgets

*Widget class reference*

Basic and graphic widgets are the content with which users interact. They may also be used as part of custom widgets you create.



```
Enter some words:

< Push me                >
```

By default, **Pile** stretches widgets to the whole parent's width.

```
window = LineBox(interior)
```

By default, **LineBox** draws a single line around the supplied widget; however, you can configure every aspect of the frame using Unicode box drawing characters (**http://unicode-table.com/en/#box-drawing**). Forget about the 'interior' widget for now – we'll get to it shortly. But for now, how do we put the dialog over the background? *Urwid* provides the **Overlay()** widget for that:

```
topw = Overlay(window, background,
    'center', 30, 'middle', 10)
main_loop = MainLoop(topw,
          palette=some_palette)
main_loop.run()
```

This lays out a 30x10 window centred on the background and starts the main loop. Note that we've used **Overlay** as the topmost widget. Should we need to change the view, the **main_loop.widget** is to be set to something different.

Now, back to the 'interior'. We want some labels (**Text**), an input (**Edit**), and a push button (**Button**) stacked vertically one over another. The way to do it in *Urwid* is to use a Pile container:

```
caption = Text(('caption', 'Enter some words:'),
    align='center')
input = Edit(multiline=False)
# Will be set from the code
scratchpad = Text('')
button = Button('Push me')
button_wrap = Padding(AttrMap(button,
          'button.normal', 'button.focus'),
          align='center', width=15)
interior = Filler(Pile([caption, input,
          scratchpad, button_wrap])
```

Here, we see two new ways to apply attributes (colours). The **Text** widget can accept a markup (a tuple or a list of tuples), and **AttrMap** can assign different attributes to focused and unfocused widgets. As we create widgets, we store them in variables for further reference.

If you try to run this code now, you'll see it fails with the **ValueError** we've already discussed. This is because the **Pile** widget's type is determined by its children, and **Text**, **Edit** and **Button** are flow widgets. **LineBox** works the same way, so finally 'window' is a flow widget in our program. However, the way we use **Overlay** implies that the top widget is a box one (since we allocated both the width and height for it ourselves), and this is the problem. We need to wrap 'interior' into something to make it boxed. The natural choice is **Filler**: we'll let flowed interior widget decide how many rows it needs, and **Filler** will take the rest. By default, **Filler** centres its contents vertically, and this is also what we want:

```
interior = Filler(Pile([...]))
```

Now the program runs; however, the button is wider than needed. That's because **Pile** makes all children equal width, so the button needs some padding:

```
button_wrap = Padding(AttrMap(...),
    align='center', width=15)
```

By default, **Padding** makes contents left-aligned, so we explicitly tell it we need them centred. Width can be an integer (the exact number of columns for the contents), 'pack' (try to find optimal width, which may not work out), or ('relative', percentage) if you want the contents to scale with the container.

Now, the interface looks as needed, however, it still does nothing. Let's change the scratchpad's contents when the button is clicked (either with the Enter key or with the mouse):

```
from urwid import connect_signal
def button_clicked(button, user_data):
        input, scratchpad = user_data
        scratchpad.set_text('You entered: %s' %\
                input.edit_text)
connect_signal(button, 'click', button_clicked,
    [input, scratchpad])
```

We pass references to **input** and **scratchpad** in **user_data**; in real-world code they will likely be some object's attributes. If you no longer want the button to work, you can disconnect the signal with the **disconnect_signal()** function. For **Button**, you can achieve the same results with the **on_press=** and **user_data=** constructor arguments, however the approach we just saw works for any event and widget (for example, Edit emits a 'change' signal when the text is changed).

Our simple program is now fully functional, except that there's no way to exit from it. We can reuse the **unhandled_input** trick, but this time, let's exit only if the user presses the F10 key:

```
def unhandled_input(key):
    if key == 'f10':
```

## Walking through the lists

**ListBox** doesn't dictate how the contents (including focused widgets) are stored: it simply manages them using the **ListWalker** interface. The latter is quite simple, and there are some stock *Urwid* classes that already implement it (like the **SimpleFocusListWalker** we saw), but you can always create your own. This is reasonable when **ListBox** contents are unsuitable to store in a Python list as a whole: they are large, take a long time to receive or whatever else. *ListWalker* solves the problem by providing the way to get (or set) the current (focused) item, and to retrieve siblings for any position in the list. This is enough to display the currently visible part of the contents. For more details, look at the **fib.py** and **edit.py** examples that ship with **Urwid**.

```
            raise ExitMainLoop()
```

If you want to, you can also add another button to close the application.

## A secret weapon

As we've already learned, *Urwid* is missing many advanced widgets. However, it includes one very powerful one: **ListBox**. You might imagine a box with a few lines of text and a highlighting bar, but *Urwid*'s **ListBox** is different (although it can look and behave this way as well). It's a scrollable list (or even tree) of arbitrary widgets that's generated dynamically, and it can serve various purposes, including creating menus, sequence editors and almost anything else (except coffee makers, you know).

**ListBox** is a bit like **Pile** in that it takes a list of widgets and stacks them vertically. However, there are many discrepancies, and they are quite important. First, passing **ListBox** a list of widgets is the most simple, limited and somewhat discouraged way to set its contents. Second, **ListBox** is always a box widget that contains flow widgets; in other words, it decides what part of the contents will be shown at given time. To make this decision, **ListBox** manages focus: if, for instance, you press the Down key, the focus will be shifted to the next child, and its contents will be scrolled accordingly.

While **ListBox** is a real Swiss Army knife, we'll use it to create a simple menu. Let's start with the **MenuItem** class. A simple menu item is just a text label that's highlighted when it has focus and responds in some way to activation (like pressing the Enter key). This means the Text widget is a perfect base class for it. We need to register a signal (let's call it 'activate'), intercept the Enter key and make the widget selectable (that's a basic property of all widgets in *Urwid*; only selectable widgets receive focus from the **ListBox** container).

```
from urwid import register_signal, emit_signal
class MenuItem(Text):
    def __init__(self, caption):
        Text.__init__(self, caption)
        register_signal(self.__class__, ['activate'])
    def keypress(self, size, key):
        if key == 'enter':
            emit_signal(self, 'activate')
        else:
            return key
    def selectable(self):
        return True
```

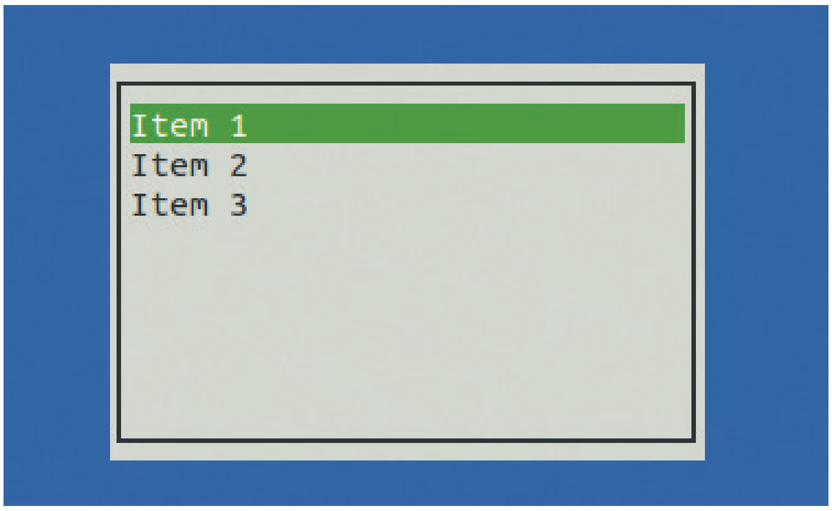
Signals are registered per-class with **register_signal()** and emitted with **emit_signal()** later. The **keypress()** method is defined in the base **Widget** class and overridden by all widgets that want to respond to the keyboard (its size is the current widget's size). If the widget successfully handled the key it returns **none**, or **key** otherwise. There is a similar **mouse_event()** method, but we won't discuss it here.

Next, we need to pack **MenuItem** objects into **ListBox**. To make current focus visible, we'll use an **AttrMap** the same way we did it for the button earlier:

```
def exit_app():
        raise ExitMainLoop()
contents = []
for caption in ['Item 1', 'Item 2', 'Item 3']:
        item = MenuItem(caption)
          connect_signal(item, 'activate', exit_app)
          contents.append(AttrMap(item,
                    'item.normal', 'item.focus'))
interior = ListBox(SimpleFocusListWalker(contents))
```

This assumes that the overall program layout is the same as in the previous example; however, since **ListBox** is box widget, there is no need to wrap 'interior' with **Filler**. We connect the 'activate' signal to the **exit_app()** function that simply terminates the program.

The **SimpleFocusListWorker** class is a basic adapter to make **ListBox** work on top of a static widget list. It derives from **ListWalker**, and you can use its other subclasses here, including the ones you create yourself, as well. The primary reason to do this is to make the contents of **ListBox** dynamic, for example, read lines from a file only when the user scrolls down to them. This is where **ListBox** comes to its full powers.



```
Item 1
Item 2
Item 3
```

**ListBox** is a natural choice for, er, a list box widget.

## Where to go next?

That's basically all for the introduction. There are some concepts, like text layout or canvas cache, that we haven't discussed, and there are others we've touched only briefly. However what you've learned today will hopefully help you to master more advanced concepts quickly. Should you need to create a sophisticated *Urwid* UI, bundled examples and existing applications (**http://excess.org/urwid/wiki/ApplicationList**) are great resources for *Urwid* programming ideas and techniques. Just don't forget to post your *Urwid* toolbox to some code hosting site for community's benefit, too!

**Dr Valentine Sinitsyn has committer rights in KDE but prefers to spend his time mastering virtualisation and doing clever things with Python.**

LINUX VOICE TUTORIAL

# XBMC: BUILD A REMOTE CONTROL

BEN EVERARD

Take control of your home media player with a custom remote control running on your Android phone.

**WHY DO THIS?**

- Learn to use the *Bottle* framework for quickly developing web applications.
- Design a remote control for your media centre that fits with your ways of working.
- Show off to friends and family with a music player that's better than theirs.

*XBMC* is a great piece of software, and can turn almost can computer into a media centre. It can play music and videos, display pictures, and even fetch a weather forecast. To make it easy to use in a home theatre setup, you can control it via mobile phone apps that access a server running on the *XBMC* machine via Wi-Fi. There are loads of these available for almost all smartphone systems.

We've recently set up an *XBMC* system for playing music, and none of the *XBMC* remotes we found really excel at this task, especially when the TV attached to the media centre is turned off. They were all a bit too complex, as they packed too much functionality into small screens. We wanted a system designed from the ground up to just access a music library and a radio addon, so we decided to build one ourselves. It didn't need to be able to access the full capabilities of *XBMC*, because for tasks other than music, we'd simply switch back to a general-purpose *XBMC* remote control. Our test system was a Raspberry Pi running the RaspBMC distribution, but nothing here is specific to either the Pi or that distro, and it should work on any Linux-based *XBMC* system provided the appropriate packages are available.

The first thing a remote control needs is a user interface. Many *XBMC* remote controls are written as standalone apps. However, this is just for our music, and we want to be accessible to guests without them having to install anything. The obvious solution is to make a web interface. *XBMC* does have a built-in web server, but to give us more control, we decided to use a separate web framework. There's no problem running more than one web server on a computer at a time, but they can't run on the same port.

There are quite a few web frameworks available. We've used *Bottle* because it's a simple, fast framework, and we don't need any complex functions. *Bottle* is a Python module, so that's the language in which we'll write the server.

You'll probably find Bottle in your package manager. In Debian-based systems (including Raspbmc), you can grab it with:

```
sudo apt-get install python-bottle
```

A remote control is really just a layer that connects the user to a system. *Bottle* provides what we need to interact with the user, and we'll interact with *XBMC* using its JSON API. This enables us to control the media player by sending JSON-encoded information.

We're going to use a simple wrapper around the XBMC JSON API called *xbmcjson*. It's just enough to allow you send requests without having to worry about the actual JSON formatting or any of the banalities of communicating with a server. It's not included in the *PIP* package manager, so you need to install it straight from *GitHub*:

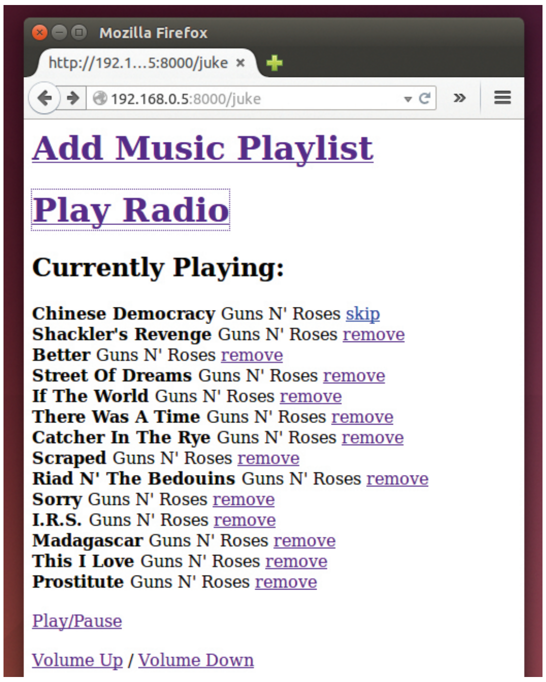
```
git clone https://github.com/jcsaaddupuy/python-xbmc.git
cd python-xbmc
sudo python setup.py install
```

This is everything you need, so let's get coding.

## Get started with Bottle

The basic structure of our program is:

```
from xbmcjson import XBMC
from bottle import route, run, template, redirect, static_file, request
```



Add Music Playlist

Play Radio

Currently Playing:

**Chinese Democracy** Guns N' Roses skip
**Shackler's Revenge** Guns N' Roses remove
**Better** Guns N' Roses remove
**Street Of Dreams** Guns N' Roses remove
**If The World** Guns N' Roses remove
**There Was A Time** Guns N' Roses remove
**Catcher In The Rye** Guns N' Roses remove
**Scraped** Guns N' Roses remove
**Riad N' The Bedouins** Guns N' Roses remove
**Sorry** Guns N' Roses remove
**I.R.S.** Guns N' Roses remove
**Madagascar** Guns N' Roses remove
**This I Love** Guns N' Roses remove
**Prostitute** Guns N' Roses remove

Play/Pause

Volume Up / Volume Down

The UI still needs a bit of attention, but at least it's working.

## Setting up

Once you've developed your remote control, you'll need a way of ensuring that it starts every time you turn on your media centre. There are a few ways of doing this, but the easiest is just to add a command launching it to **/etc/rc.local**. We installed our file to **/opt/xbmc-remote/remote.py** with all the other files alongside it. We then added the following line to **/etc/rc.local** before the final **exit 0** line.

```
cd /opt/xbmc-remote && python remote.py &
```

```
import os
xbmc = XBMC("http://192.168.0.5/jsonrpc", "xbmc", "xbmc")
@route('/hello/<name>')
def index(name):
        return template('<h1>Hello {{name}}!</h1>',
name=name)
run(host="0.0.0.0", port=8000)
```

This connects to *XBMC* (though doesn't actually use it); then *Bottle* starts serving up the website. In this case, it listens on host 0.0.0.0 (which is every hostname), and port 8000. It only has one site, which is **/hello/XXXX** where **XXXX** can be anything. Whatever **XXXX** is gets passed to **index()** as the parameter name. This then passes it to the template, which substitutes it into the HTML.

You can try this out by entering the above into a file (we've called it **remote.py**), and starting it with:

```
python remote.py
```

You can then point your browser to **localhost:8000/hello/world** to see the template in action.

**@route()** sets up a path in the web server, and the function **index()** returns the data for that path. Usually, this means returning HTML that's generated via a template, but it doesn't have to be (as we'll see later).

As we go on, we'll add more routes to the application to make it a fully-featured *XBMC* remote control, but it will still be structured in the same way.

The *XBMC* JSON API can be accessed by any computer on the same network as the *XBMC* machine. This means that you can develop it on your desktop, then deploy it to your media centre rather than fiddle round uploading every change to your home theatre PC.

Templates – like the simple one in the previous example – are a way of combining Python and HTML to control the output. In principal, they can do quite a bit of processing, but they can get messy. We'll use them just to format the data correctly. Before we can do that, though, we have to have some data.

## Getting data from XBMC

The *XBMC* JSON API is split up into 14 namespaces: JSONRPC, Player, Playlist, Files, AudioLibrary, VideoLibrary, Input, Application, System, Favourites, Profiles, Settings, Textures and XBMC. Each of these is available from an *XBMC* object in Python (apart from Favourites, in an apparent oversight). In each of these namespaces there are methods that you can use to control the application. For example, **Playlist.GetItems()** can be used to get the items on a particular playlist. The server returns data to us in JSON, but the *xbmcjson* module converts it to a Python dictionary for us.

There are two items in *XBMC* that we need to use to control playback: players and playlists. Players hold a playlist and move through it item by item as each song finishes. In order to see what's currently playing, we need to get the ID of the active player, and through that find out the ID of the current playlist. We've done this with the following function:

```
def get_playlistid():
    player = xbmc.Player.GetActivePlayers()
    if len(player['result']) > 0:
            playlist_data = xbmc.Player.
GetProperties({"playerid":0, "properties":["playlistid"]})
            if len(playlist_data['result']) > 0 and "playlistid" in
playlist_data['result'].keys():
                    return playlist_data['result']['playlistid']
    return -1
```

If there isn't a currently active player (that is, if the length of the **results** section in the returned data is 0), or if the current player has no playlist, this will return **-1**. Otherwise, it will return the numeric ID of the current playlist.

Once we've got the ID of the current playlist, we can get the details of it. For our purposes, two things are important: the list of items in the playlist, and the position we are in the playlist (items aren't removed from the playlist after they've been played; the current position just marches on).

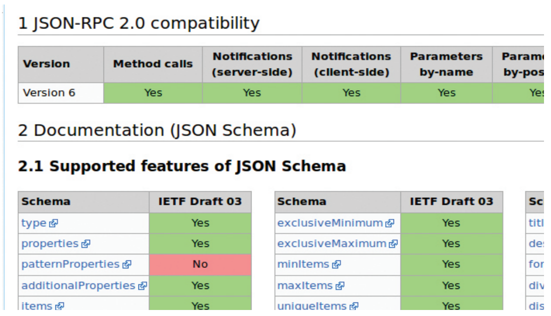
```
def get_playlist():
        playlistid = get_playlistid()
        if playlistid >= 0:
                data = xbmc.Playlist.GetItems({"playlisti
d":playlistid, "properties": ["title", "album", "artist", "file"]})
                position_data = xbmc.Player.
GetProperties({"playerid":0, 'properties':["position"]})
```

### Logging

It's not always clear how to do something using the *XBMC* JSON API, and the documentation is sometimes a little opaque. One way of finding out how to do something is seeing how other remote controls do it. If you turn on logging, you can see what API calls are being performed as you use another remote control, then incorporate these into your code.

To turn on logging, hook your *XBMC* media centre up to a display and go to Settings > System > Debugging, and turn on Enable Debug Logging. With logging turned on, you need to access the *XBMC* machine (eg via SSH), then you can view the log. Its location should be displayed in the top-left corner of the *XBMC* display. In RaspBMC, it's at **/home/pi/.xbmc/temp/xbmc.log**. You can then keep an eye on what API calls are being performed in real time using:

```
cd /home/pi/.xbmc/temp
tail -f xbmc.log | grep "JSON"
```



1 JSON-RPC 2.0 compatibility

| Version | Method calls | Notifications (server-side) | Notifications (client-side) | Parameters by-name | Param by-pos |
|---|---|---|---|---|---|
| Version 6 | Yes | Yes | Yes | Yes | Ye |

2 Documentation (JSON Schema)

2.1 Supported features of JSON Schema

| Schema | IETF Draft 03 |
|---|---|
| type | Yes |
| properties | Yes |
| patternProperties | No |
| additionalProperties | Yes |
| items | Yes |

| Schema | IETF Draft 03 |
|---|---|
| exclusiveMinimum | Yes |
| exclusiveMaximum | Yes |
| minItems | Yes |
| maxItems | Yes |
| uniqueItems | Yes |

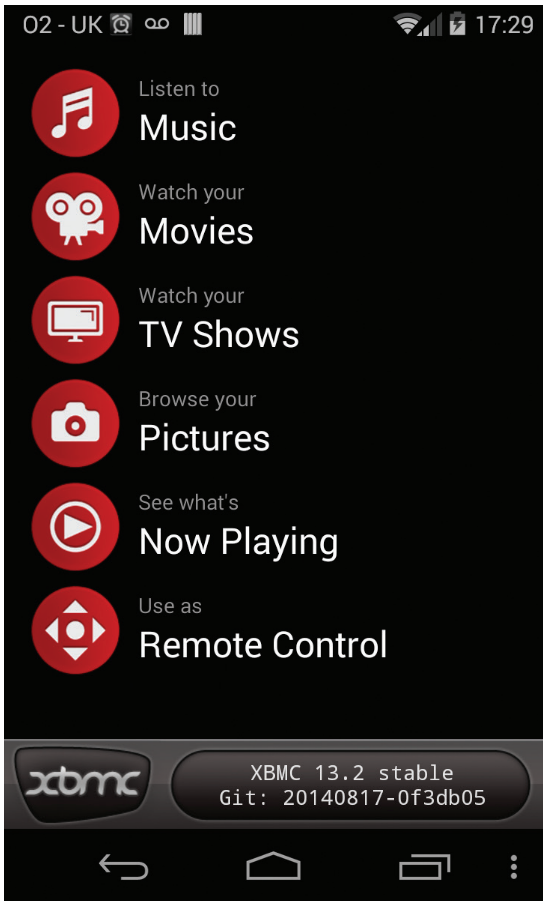
| Sc |
|---|
| titl |
| de |
| for |
| div |
| dis |

The API is documented at **http://wiki.xbmc.org/?title=JSON-RPC_API/v6**. It lists all the available functions, but it a little short on details of how to use them.

### Kodi

By the time you read this, *XBMC* may be no more. The project team have decided to rename it *Kodi* for legal reasons (and because *XBMC*, or *X-Box Media Centre*, refers to older hardware that is no longer supported). Other than the name, though, nothing has changed. Or at least nothing other than the usual raft of improvements you'd expect from a new release. This shouldn't affect the remote software though, and it should work on both existing *XBMC* systems, and newer *Kodi* systems.

The official Android remote can still control our media player when we need more complex functions.



```
        position = int(position_data['result']['position'])
        return data['result']['items'][position:], position
    return [], -1
```

This returns the current playlist starting with the item that's currently playing (since we don't care about stuff that's finished), and it also includes the position as this is needed for removing items from the playlist.

## Bringing them together

The code to link the previous functions to a HTML page is simply:

```
@route('/juke')
def index():
        current_playlist, position = get_playlist()
        return template('list', playlist=current_playlist,
offset = position)
```

This only has to grab the playlist (using the function we defined above), and pass it to a template that handles the display.

The main part of the template that handles the display of this data is:

```
<h2>Currently Playing:</h2>
% if playlist is not None:
% position = offset
% for song in playlist:
<strong> {{song['title']}} </strong>
% if song['type'] == 'unknown':
Radio
% else:
{{song['artist'][0]}}
% end
% if position != offset:
<a href="/remove/{{position}}">remove</a>
% else:
<a href="/skip/{{position}}">skip</a>
% end
<br>
% position += 1
% end
```

As you can see, templates are mostly written in HTML, but with a few extra bits to control output. Variables enclosed by double parenthesise are output in place (as we saw in the first 'hello world' example). You can also include Python code on lines starting with a percentage sign. Since indents aren't used, you need a **% end** to close any code block (such as a loop or if statement).

This template first checks that the playlist isn't empty, then loops through every item on the playlist. Each item is displayed as the song title in bold, then the name of the artist, then a link to either skip it (if it's the currently playing song), or remove it from the playlist. All songs have a type of 'song', so if the type is 'unknown', then it isn't a song, but a radio station.

The **/remove/** and **/skip/** routes are simple wrappers around *XBMC* controls that reload **/juke** after the change has taken effect:

```
@route('/skip/<position>')
def index(position):
        print xbmc.Player.GoTo({'playerid':0, 'to':'next'})
        redirect("/juke")
@route('/remove/<position>')
def index(position):
        playlistid = get_playlistid()
        if playlistid >= 0:
                xbmc.Playlist.Remove({'playlistid':int(pla
ylistid), 'position':int(position)})
        redirect("/juke")
```

Of course, it's no good being able to manage your playlist if you can't add music to it.

This is complicated slightly by the fact that once a playlist finishes, it disappears, so you need to create a new one. Rather confusingly, playlists are created by calling the **Playlist.Clear()** method. This can also be used to kill a playlist that is currently playing a radio station (where the type is unknown). The other

## JSON

JSON stands for JavaScript Object Notation, and was originally designed as a way of serialising JavaScript Objects. It still is used for that, but it's also a useful way of encoding all sorts of data.

JSON objects always have the form:

```
{property1:value1, property2:value2,
property3:value3}
```

For an arbitrary number of property/value pairs. To Python programmers, this all looks suspiciously similar to dictionaries, and the two are very similar.

As with dictionaries, the value can itself be another JSON object, or a list, so the following is perfectly valid:

```
{"name":"Ben", "jobs":["cook", "bottle-washer"],
"appearance": {"height":195, "skin":"fair"}}
```

JSON is often used in web services to send data back and fourth, and it's well supported by most programming languages, so if Python's not your thing, you should easily be able to use the same functions to control *XBMC* from software written in the language of your choice.

complication is that radio streams sit in the playlist and never leave, so if there's currently a radio station playing, we need to clear the playlist as well.

These pages include a link to play the songs, which points to **/play/<songid>**. This page is handled by:

```
@route('/play/<id>')
def index(id):
    playlistid = get_playlistid()
    playlist, not_needed= get_playlist()
    if playlistid < 0 or playlist[0]['type'] == 'unknown':
        xbmc.Playlist.Clear({"playlistid":0})
        xbmc.Playlist.Add({"playlistid":0,
"item":{"songid":int(id)}})
        xbmc.Player.open({"item":{"playlistid":0}})
        playlistid = 0
    else:
        xbmc.Playlist.Add({"playlistid":playlistid,
"item":{"songid":int(id)}})
    remove_duplicates(playlistid)
    redirect("/juke")
```

The final thing here is a call to **remove_duplicates**. This isn't essential – and some people may not like it – but it makes sure that no song appears in the playlist more than once.

We also have pages that list all the artists in the collection, and list the songs and albums by particular artists. These are quite straightforward, and work in the same basic way as **/juke**.

## Adding functionality

The above code all works with songs in the *XBMC* library, but we also wanted to be able to play radio stations. Addons each have their own plugin URL that can be used to pull information out of them using the usual *XBMC* JSON commands. For example, to get the selected stations from the **radio** plugin, we use:

```
@route('/radio/')
def index():
        my_stations = xbmc.Files.GetDirectory({"directory":"
plugin://plugin.audio.radio_de/stations/my/", "properties":
["title","thumbnail","playcount","artist","album","episode","season"
,"showtitle"]})

if 'result' in my_stations.keys():
                    return template('radio', stations=my_
stations['result']['files'])
            else:
                    return template('error', error='radio')
```

This includes a file that can be added to a playlist just as any song can be. However, these files never finish playing, so (as we saw before) you need to recreate the playlist before adding any songs to it.

## Sharing songs

As well as serving up templates, *Bottle* can serve static files. These are useful whenever you need things that don't change based on the user input. That could be a CSS file, an image or an MP3. In our simple controller there's not (yet) any CSS or images to make things look pretty, but we have added a way to download the songs. This lets the media centre act as a sort of NAS box for songs. If you're transferring large amounts of data, it's probably best to use something like Samba, but serving static files is a good way of grabbing a couple of tunes on your phone.

The *Bottle* code to download a song by its ID is :

```
@route('/download/<id>')
def index(id):
            data = xbmc.AudioLibrary.GetSongDetails({"songid":i
nt(id), "properties":["file"]})
            full_filename = data['result']['songdetails']['file']
            path, filename = os.path.split(full_filename)
            return static_file(filename, root=path,
download=True)
```

To use this, we just put a link to the appropriate ID in the **/songsby/** page.

We've gone through all the mechanics of the code, but there are a few more bits that just tie it all together. You can see for yourself at the *GitHub* page: **https://github.com/ben-ev/xbmc-remote**. LV

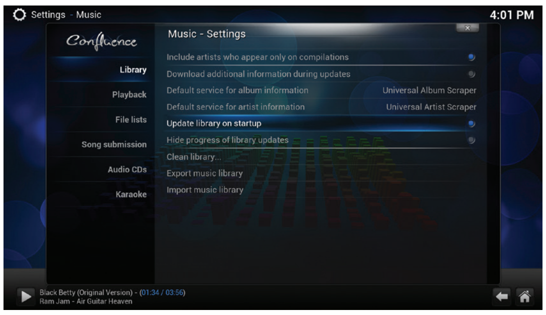
**For fun, Ben Everard hacks hardware projects held together with a big dollop of Linux and Free Software glue.**

### Paste

*Bottle* includes its own web server, which is what we've been using for testing the remote control. However, we found that it didn't always perform well. When we put the remote into action, we wanted something that could deliver pages a bit quicker. *Bottle* can work with quite a few different web servers, and we found *Paste* worked quite well. In order to use this, just install it (in the package **python-paste** on Debian), and change the run call to:

```
run(host=hostname, port=hostport,
server="paste")
```

You can see details of how to use other servers at **http://bottlepy.org/docs/dev/deployment.html**.



Include artists who appear only on compilations ●
Download additional information during updates ●
Default service for album information Universal Album Scraper
Default service for artist information Universal Artist Scraper
Update library on startup ●
Hide progress of library updates ●
Clean library...
Export music library
Import music library

Black Betty (Original Version) - (01:34 / 03:56)
Ram Jam - Air Guitar Heaven

By editing the settings in System > Music Library, you can set *XBMC* to scan for new music on startup, so the most current music gets added without manual intervention.

### GitHub

This project is quite bare-bones at the moment, but – the business of running a magazine means we don't have as much time as we'd like to program. However, we've set up a *GitHub* project where we hope to keep working on it, and if you think you'd benefit from the project as well, we'd love your input.

To see what's going on, head over to **https://github.com/ben-ev/xbmc-remote** and take a look at what state it's in. You can get a copy of the latest code from that web page, or clone it from the command line.

If you want to improve it, you can fork the project to develop in your own branch, and then send a pull request when your features are working. For more information on working with *GitHub*, head to **https://github.com/features**.

LINUX VOICE TUTORIAL

# CODE NINJA: LAMBDA FUNCTIONS

BEN EVERARD

Anonymous functions aren't just 4Chan meetups – they're also a way to create cleaner code.

**WHY DO THIS?**

- Write cleaner code.
- Understand one of the formal underpinnings of computation.
- Sound clever in conversations with other programmers.

If we were trying to come up with a name to make something sound excessively mathematical, we couldn't do better than lambda calculus. The phrase conjures up a picture of a stern-faced maths teacher peering over his glasses while wearing a tweed jacket with leather-patched elbows.

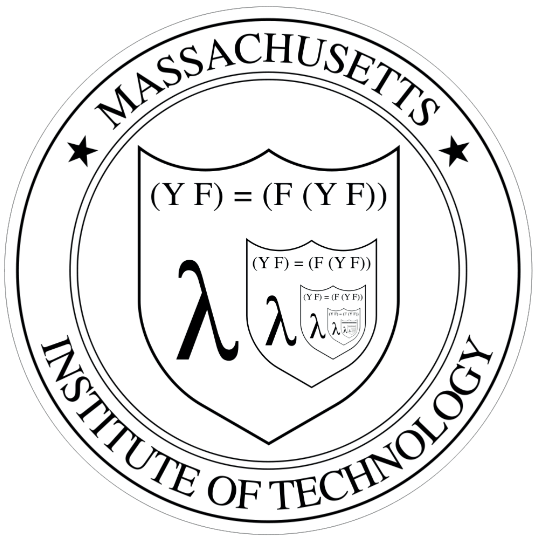
The reason it sounds so confusing is because it hails from a time when computing was little more than an obscure branch of maths that only interested academics and the military.

Lambda calculus was created as mathematicians were struggling to understand computation, and what its limits were. It's a very simple way of specifying programs, and because it's simple, it's easy to reason about mathematically.

Basically, lambda calculus is a way of forming computer programs out of functions with two restrictions. Firstly, the functions don't have a name, and secondly they can only take one argument. Functions that follow these rules are known as Lambda functions. Let's take a look at this in Python, which supports lambda functions with the lambda statement:

```
>>> add2 = lambda x: x+2
```

This creates a function that takes one argument (**x**) and returns the number **x+2**. Python imposes additional restrictions on lambda functions: they can only contain one statement and that statement must return a value (which not all statements do in Python).



The badge of the Knights of the Lambda Calculus – a band of Lisp programmers who wait for the day when a well-placed anonymous function will save the world.

Because they only contain one statement, they don't need the return keyword to specify what they return. Whatever is after the colon is the statement, and the function will return whatever it evaluates to.

In this case, we've assigned the function to a variable called **add2**. You don't have to assign the function to a variable and most of the time it's more useful not to (remember that we said functions don't

## Church-Turing thesis

We've looked at lambda functions in Python where they're a convenient shorthand for creating functions to be used only once. However, the basic purpose of Lambda calculus wasn't to add syntactical simplicity to high level languages. It was to help understand computation.

One of the big problems in early computer science was working out what could be computed and what couldn't. Alonzo Church worked with lambda calculus as Alan Turing worked with Turing machines.

It's possible to show that anything computable using a Turing Machine is computable using lambda calculus and vice versa. It's also possible to prove that some things can't be computed using Turing Machines or lambda calculus. For example, the halting problem can't be computed. This means that it's impossible to write a program that takes another program as input and works out whether or not it will finish running, or not (eg whether it will get stuck in an infinite loop).

The Church-Turing thesis states that anything that can be computed by a computer can be computed using lambda calculus or a Turing Machine. However, this problem remains stubbornly a thesis and has never been formally proven. Since lambda calculus can implement anything that a Turing machine can, lambda calculus is known as Turing-complete. If the Church-Turing thesis is correct, any language that is Turing complete can compute anything that is computable. All general-purpose languages are Turing complete – as you would expect – but so are some languages that are quite restrictive. For example, **sed** is Turing complete (see **www.robertkotcher.com/sed.html** for proof). Some more powerful markup languages are also Turing complete, such as HTML5 + CSS3 (**https://github.com/elitheeli/stupid-machines**) and C++ templates (**http://ubietylab.net/ubigraph/content/Papers/pdf/CppTuring.pdf**).

The creativity of geeks knows no bounds, and it's become a challenge to prove ever more obscure things are Turing complete. *Minecraft* is Turing complete (**www.youtube.com/watch?v=1X21HQphy6I**) and so is an infinite version of *Minesweeper* (**http://web.mat.bham.ac.uk/R.W.Kaye/minesw/infmsw.pdf**), but the most bizarre thing we could find that is Turing complete is the *Magic: The Gathering* card game (**www.toothycat.net/~hologram/Turing/HowItWorks.html**). If the Church-Turing thesis is correct, this means that it's possible to port any computer program to run on the *Magic: The Gathering* card game. Weird, huh?

have names?), but we'll get onto that in a bit. You can run the function with:

```
>>> add2(1)
3
```

So far, this just looks like a slightly awkward way of creating functions. You could be forgiven for wondering why Python includes this slightly odd theoretical concept. One of the advantages of lambda functions in Python is that they can be a very convenient way of specifying a function that will only be used once. Typically, this when a function is needed as a parameter.

For example, take a look at the following function from the *XBMC* remote elsewhere in this issue's coding section:

```
def get_artists():
        data = xbmc.AudioLibrary.GetArtists()
        return sorted(data['result']['artists'], key=lambda k:
k['label'])
```

Here, the Python function **sorted()** can take an argument called **key** which specifies a function that is called on each element to be sorted that returns the value that the items should be sorted on. In this case, key is a lambda function that takes a dictionary as its parameter and outputs the particular item from that dictionary that we want to sort on. We could define a function in the usual Python way (by using **def** and giving it a name). However the lambda notation is clearer and simpler.

## Hello again, Mr Turing!

Lambda calculus wasn't created as a convenient shorthand. It was created as a method of defining computation. Like Turing machines, lambda calculus is a computationally complete language. That means that anything that can be computed, can be defined using lambda calculus (not necessarily in Python's restricted version of it though).

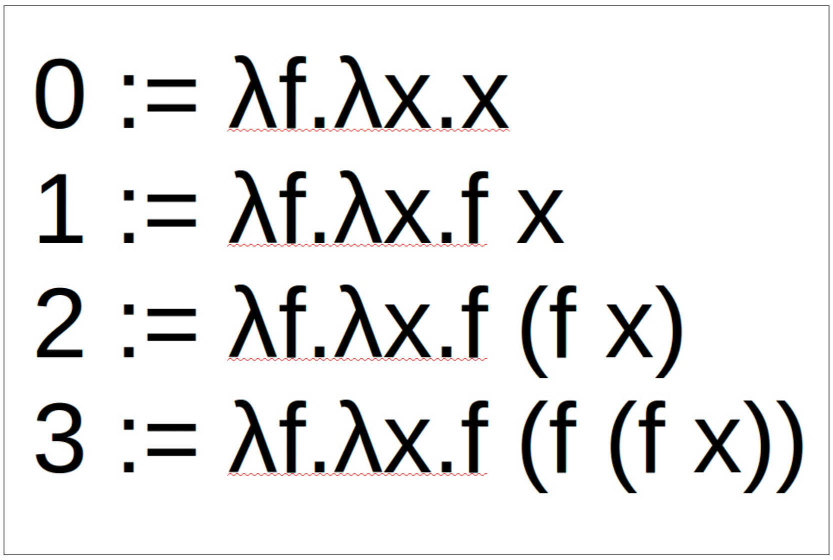
Obviously this isn't possible if each function can only operate on a single value. Lambda calculus also allows chaining of functions to build up more complex operations. For example, you could create a function to add two values together with:

```
>>> add = lambda x: lambda y: y+x
>>> add(3)(2)
5
```

### Beyond Python

Most programming languages allow anonymous functions (you can argue about whether an anonymous function with more than one argument is really a lambda function). The only commonly used general purpose languages without them are C (though they are supported in Clang) and Fortran. No other common language has the single statement restriction of Python.

The syntax and terminology varies from language to language, but they're usually used for cases similar to those we've looked at here when functions need passing as arguments in other functions, particularly in callbacks (which we looked at in LV007).



$$0 := \lambda f.\lambda x.x$$
$$1 := \lambda f.\lambda x.f\ x$$
$$2 := \lambda f.\lambda x.f\ (f\ x)$$
$$3 := \lambda f.\lambda x.f\ (f\ (f\ x))$$

Lambda calculus gets its name from the lower-case Greek letter lambda, which is used to denote anonymous functions. It's shown here calculating the Church numerals.

This chaining – also known as currying – enables you to build up functions of arbitrary complexity. It also enables you to build functions by fixing particular parameters in other lambda functions. For example (following on from the previous session):

```
>>> add10 = add(10)
>>> add10(1)
11
```

This is rarely used in Python, but it can be used in a few ways. For example, we could use it to create logging functions for system and application errors in Python 3:

```
>>> p_log = lambda er: lambda msg: print(er, msg)
>>> p_sys_err = p_log("System error:")
>>> p_app_err = p_log("Application error:")
>>> p_sys_err("operating system problem")
System error: operating system problem
>>> p_app_err("the application has crashed")
Application error: the application has crashed
```

You need to use Python3 because in previous versions of Python, **print()** didn't return a value, and so couldn't be used as a lambda statement (in Python3, **print()** is a function that returns **None**).

In Python, the restriction to only one statement means you can't loop through data, since there can't be any code blocks. However, you can still use **if** statements using a slightly different format:

```
x if <conditions> else y
```

For example, you could use this to return the lowest number in a pair using:

```
>>> min = lambda x: x[0] if x[0]<x[1] else x[1]
>>> min([3,5])
>>> 3
```

Python doesn't need lambda functions. Everything you do with them could also be achieved without them. However, there are several places where they can be used to make your code more readable. This is usually in places where a function object is passed (like in the **sort** example above).

LINUX VOICE TUTORIAL

# SOPHIE WILSON, ACORN AND THE DEVELOPMENT OF ARM

JULIET KEMP

ARM chips – via Android and smartphones – are taking Linux to the masses. Here's what makes them so special.

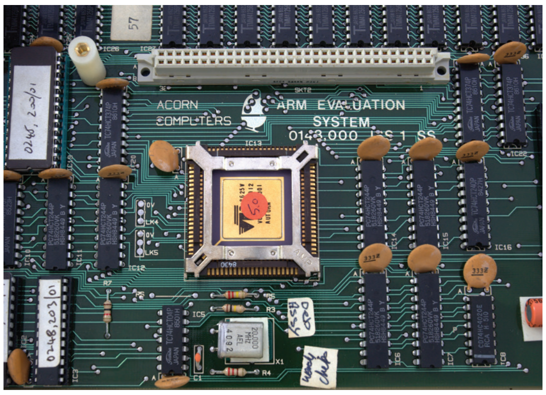
By 2014, over 50 billion ARM processor cores had been shipped since the first ARM chip was created by Sophie Wilson in the mid-1980s. Ten billion of those were produced in 2013, so by the time you read this, the figure is probably coming up on 60 billion. This meteoric rise from a mere 10 billion ever shipped in 2008 mirrors the rise of mobile computing. Nearly 60% of mobile devices, and 95% of smartphones, contain an ARM-based chip. You've probably got one in your pocket right now. I certainly have. So where did they start out?

Sophie Wilson was born in Leeds in 1957, and studied maths at Cambridge. In 1978, during the big microprocessor boom (see the BASIC article in LV005), she was working with Hermann Hauser to solve a problem for a fruit machine manufacturer. Someone had developed a hack which used a cigarette lighter to shock (literally!) the new electronic machines into disgorging cash. Wilson created a radio receiver to detect the cigarette lighter spark, solving that problem; whereupon Hauser challenged her to create a working PC by the end of the summer. Wilson succeeded, and six months later, Hauser's company, now relaunched as Acorn Computers, started offering the Acorn System One, with a princely 512B of RAM, for £70. Everything was built in-house: logic circuits, assemblers, BASIC interpreters – the lot. By mid-1981, the UK PC market was dominated by the ZX81 (by Clive Sinclair, and available in WHSmith shops) and the Acorn Atom (more expensive, and only available as a kit from Acorn).

In 1981, Wilson improved and extended the Acorn's version of BASIC into the Acorn Proton, which then became the BBC Micro and had its BASIC developed into BBC BASIC. The Proton was built in a week after Chris Curry, co-founder of Acorn, promised the BBC that they would have a machine to demonstrate within the week. They made it – just.

Wilson ported the OS across to the Proton's raw hardware, and installed BASIC, in the two hours between the hardware working and the BBC arriving for the demo.

However, what we're looking at in this article is ARM, the Acorn RISC Machine, one of the first RISC processors, which later became one of the most successful IP cores of the 1990s and 2000s, in particular for use in mobile devices.



The first ARMv1 in an evaluation system. ARM chips have begun to show up in servers for the first time, though they'll have a long way to go to challenge Intel's dominance.

## Creating ARM

The ARM chip was a specific instance of a RISC processor. Reduced Instruction Set Computing (RISC) originated at IBM. It meant that instead of the increasingly complex instructions that processors were using in the early 1980s, a RISC processor would use a limited set of simple instructions. However, IBM hadn't really got anywhere with the idea – they'd created a RISC processor after months of work simulating instructions on a mainframe, but it was a commercial flop. Meanwhile, working on the BBC machines, Acorn were becoming frustrated by the limitations of the BBC's microprocessor. The main problem was the memory interface: how fast a chip could fetch, and thus execute, transactions. Wilson found it frustratingly slow, and it was restricting what they could do with their secondary processors.

After reading one of the first papers about RISC,

### IP cores

A semiconductor IP (intellectual property) core is a chunk of chip or logic design that is the intellectual property of a particular party, usually a company. The chunks can be used as building blocks for larger chip or logic designs. They may be used only by that company or may be licensed out. The ability to license designs like this means that chip makers can use a standard set of processors and internal functions, and then focus on specific features or innovations of their particular chip. This has sped up development significantly since it became common in the 1990s. IP cores can be soft cores, described in a 'high level' hardware description language (and thus modifiable by the chip maker), or hard cores, described as a physical description (and thus not modifiable). ARM architectures are soft designs and are licensed and used in a huge range of systems. A major advantage of being an IP core company is that you don't have to pay for the (very expensive) kit to fabricate your own chips.

Wilson and Acorn started investigating their options. A visit to the huge facilities at National Semiconductors in Israel was depressing; Acorn couldn't afford anything like that. Then they visited the much smaller but very successful Western Design Centre in Arizona, which consisted of only a couple of bungalows and a small team of engineers and students. Reassured that you didn't need a huge operation to design processors, Wilson got stuck into designing the ARM instruction set back at her desk at Acorn (and in the local pub over lunches with colleagues!). Steve Furber was then responsible for turning Wilson's instruction set into something that could be produced at a factory. Eighteen months later, they had the first working ARM.

It's odd that what is now the major selling point of ARM processors, their low power consumption, was only a side effect. What Acorn were interested in was low cost, and low cost meant plastic. Plastic is a good insulator, which is bad news on a high-power chip as the heat takes longer to dissipate and your chances of frying the chip increase. So that in turn meant keeping the ARM power consumption under 1W.

However, when they got the first test chips back and plugged them into a development board, the chip worked – but seemed to be consuming no power at all. It turned out that there was a fault in the board, and the power supply line wasn't working. The chip was, as Wilson explains, "running on leakage from the logic circuits". The chip consumed an incredibly low 0.1 watts. Wilson's ARM, it turned out, was a particularly efficient version of RISC.

Wilson rewrote BBC BASIC in ARM assembler very efficiently, but the first complete ARM computer was the Acorn Archimedes in 1987. It and its successors were among the most powerful home computers at the time. Of more long-term importance, Apple had realised that the ARM processor needed only a small amount of chip real estate – making it possible to squeeze further processing power onto the same chip. Apple invested heavily in ARM for the Newton (the first ever tablet, which flopped); but the investment paid off later in the iPhone, iPod, and iPad.

## ARM architecture and instruction set

When Wilson and the other Acorn folk were designing ARM, they weren't dedicated to sticking exactly to the model set by Berkeley RISC. They kept the load/store architecture, the fixed length instructions, and the three-address instruction format (destination, operator 1, operator 2). They rejected register windows, branch delay slots, and universal single-cycle instructions (most ARM instructions are single-cycle, but not all of them). ARM also initially lacked multiply and co-processor support. It had a 32-bit data bus, 26-bit (later 32 bit) address space, and 27 32-bit registers.

Since ARMv4T, ARMs have a second instruction set: the 16-bit Thumb set. This increases compiled code density by reducing the available functionality. The shorter opcodes also improve performance, especially on embedded hardware with limited memory bandwidth. If you're interested in the details of the registers (37 of them), processor modes, exception handling, and so on of current ARM chips, there's a great lecture online at **http://www.ee.ncu.edu.tw/~jfli/soc/lecture/ARM_Instr_Set.pdf** from Jin-Fu Li, National Central University, Taiwan. You can also get extensive documentation for various chips from the ARM website.

I wasn't able to find an instruction set for ARM v1, but 1987 documentation for ARM v3 should have largely the same instructions (with a larger address space). They divide into five basic groups:

- Data manipulation (**ADD**, **AND**, **MOV**, **SUB**, **CMP** etc).
- Load and store (**LDR** to load a register and **STR** to save one).
- Multiple load and store (**LDM**, **STM**).
- Branch – conveniently jump between instructions.
- Software interrupt (**SWI**, but there are many different expressions that can be passed to it to determine what it does, including keyboard output and input).

Let's take a look at some ARM assembler code. This example from an ARM handbook multiplies a

### RISC

The basic idea behind Reduced Instruction Set Computing (RISC) is that you can get better performance (compared to a complex, specialised instruction set) out of a simplified instruction set running on a microprocessor which needs as few as possible cycles per instruction. The 'reduced' refers not necessarily to the number of instructions, but to the amount of work that an instruction does – each instruction should use a single clock cycle (often achieved by using a technique called pipelining). A precise definition is hard to pin down, but two common RISC traits are a small, highly optimised set of instructions; and load/store architecture, where memory must be accessed through specific instructions, rather than as part of other instructions.

RISC is inherently more power-efficient than, say, x86, because a RISC instruction is always four bytes long. That means that the chip doesn't have to expend any processor power in parsing the length of the instruction and separating instructions. So (put very simply) a RISC instruction takes less energy to handle, and can be understood by a smaller chip.

The two projects most associated with RISC are Stanford's, which emerged into the commercial world as the MIPS architecture, and Berkeley's RISC, which eventually became SPARC. IBM's efforts (after their initial commercial flop) eventually led to the Power Architecture. And of course ARM has been incredibly successful, as have other RISC architectures.

Acorn Archimedes setup in 1987

Sophie Wilson was made a Fellow of the Royal Society in 2013, for having made "a substantial contribution to the improvement of natural knowledge".

value by 6:

```
ADD  Ra,Ra,Ra,LSL #1     ; multiply by 3
MOV  Ra,Ra,LSL #1        ; and then by 2.
ADD
```

takes three arguments: one destination and two operands. So

```
ADD Ra,Rb,Rc
```

means

```
Ra := Rb + Rc
```

(where **Rn** is register n). However, the line here seems to have a third operand, **LS#1**.In fact, the second operand isn't Ra, but

```
Ra,LSL #1
```

**LSL #n** means Logical Shift Left n places, which effectively multiplies the number stored in Ra by $2^n$. (Similarly, if using logical shift right (LSR),

```
Ra,LSR#n
```

divides Ra by $2^n$.) So here, **Ra,LSR#1** multiplies Ra by $2^1 = 2$. Thus,

```
ADD Ra,Ra,Ra,LSL #1
```

means

```
Ra := Ra + (Ra * 2)
```

ie

```
Ra := Ra * 3.
```

To add an absolute value, you could write it like this:

```
ADD Ra, Ra, #1
```

This would add 1 (the absolute value 1) to **Ra**, and store the result back into **Ra** – acting as an increment line. **MOV** transfers its operand to the destination register:

```
MOV destination, operand
```

So here:

```
MOV  Ra,Ra,LSL #1
```

means that **Ra,LSL#1**, that is, **Ra * $2^1$ = Ra** is transferred into the **Ra** register. So this line just multiplies **Ra** by 2. Since the previous line multiplied **Ra** by 3, the total effect is to multiply the contents of **Ra** by 6 and store the result back in the **Ra** register.

You may have noticed that multiplying by 8 would have been rather easier:

```
MOV  Ra,Ra,LSL #3
```

And, of course, there are many ways to achieve the same result. The left-hand operand must always be a single register, but the right-hand operand can, as here, contain other operations. This versatility is helpful when maximising code efficiency.

Here's a slightly more complicated example. I'll use the code from the Grace Hopper article from LV002, which instructed UNIVAC to add a series of numbers stored in memory addresses 100–999. Memory in UNIVAC was a series of registers from 0-999, whereas memory in ARMv1 used a 26-bit address value, with a 4 byte (32 bit) word length. This means that ARM word addresses start at 0 and go up in 4s: 0, 4, 8, ... 64M. I've translated UNIVAC addresses 100-999 as ARM memory addresses &1000-&1E0C (in hexadecimal). A semi colon denotes that the rest of the line is a comment. This is theoretical code, not tested, but should give you an idea of how ARM assembler works.

```
MOV  R0,#0        ; Zero the running total
MOV  R1,#0        ; Zero the number that holds the next value
MOV  R2,#1000  ; Store memory address 1000 into R2
 .LOOP ADD  R0,R0,R1  ; Label loop, and R0 := R0 + R1
 LDR R1,[R2],#4  ; Load the contents of R2 address and
increment it
 TEQ R2,#1E10   ; test which address we're at
 BNE  LOOP        ; carry on unless we're done
SWI WriteI+R0    ; output the running total with SWI (pseudo-
code)
```

Let's look at that in more detail:

- **MOV R0,#0** this loads the literal value 0 into R0. The next two lines work similarly, initialising R1 and R2.
- **LOOP** this is a label for the first line of the loop.
- **ADD R0,R0,R1** as above, **R0 := R0 + R1**. Note that the first time around, this translates as **R0 := 0 + 0**, ie nothing happens.
- **LDR R1,[R2], #4** Load contents of address held in R2 into R1, then increment R2 by 1 word. Note that this requires the numbers you're adding to be single-word length. The first time through the loop, this will load the contents of memory address 1000 into R1 (so the next time through the loop, the **ADD** line will add it to R0), and increment the memory address stored in R2 ready for the next time through the loop.
- **TEQ R2,#1E10** – **TEQ** compares its two operands, here the value of R1, and the address 1E0C (the address after the final memory address we want. The **Z** result flag is set to 1 if they are equal, 0 if not.
- **BNE LOOP** – **B** is the simple branch instruction, and send us back to the **LOOP** label. The conditional suffix **NE** stands for Not Equal. If **Z** is not set, then a

**BNE** instruction will run. If it is set, then **BNE** is not true, and will not run. The opposite of this is **EQ**. **BEQ** would run if **Z** is set, and not if not. This instruction stops the loop if we've passed the final memory address, ie we have run out of numbers to add.

- **SWI WriteI+R0** – **SWI** offers a call-out to other instructions, and the instructions available will depend on the details of the architecture. Input/output are usually available, and this pseudocode outputs **R0**.

If you want to delve further into ARM Assembly language programming, I strongly recommend the web-based version of Pete Cockerell's 1987 book, *ARM Assembly Language Programming*, at **www.peter-cockerell.net/aalp/html/frames.html**. This covers specifically ARMv3, but I found it to be a useful reference for the basics of ARM programming (and an interesting document!). An ARM quick reference card is available from ARM at **http://infocenter.arm.com/help/topic/com.arm.doc.qrc0001m/QRC0001_UAL.pdf**.

## RISC OS

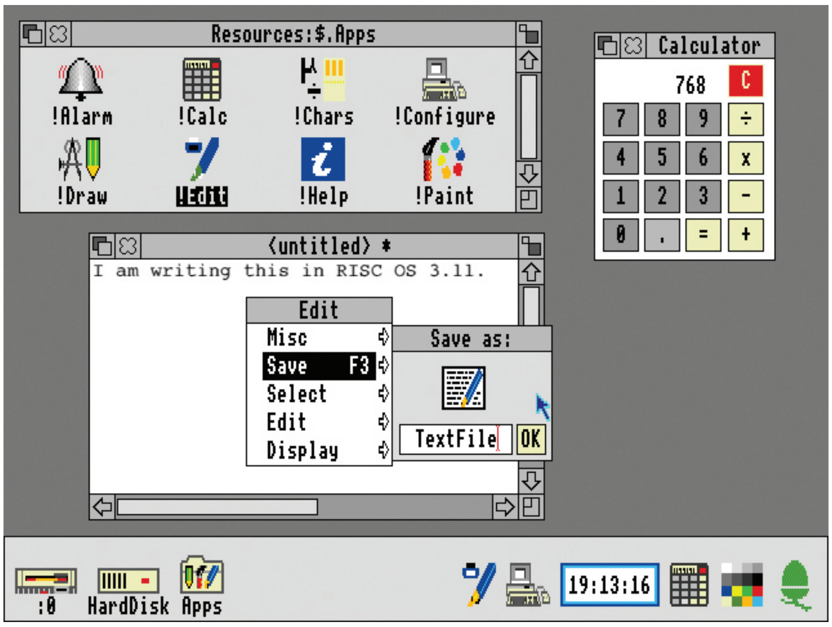
Acorn's other big achievement was RISC OS. After some financial problems, in 1985 Olivetti took a controlling stake in Acorn, but the company continued to operate independently. During this time, Acorn was developing RISC OS for the Archimedes, and released it in 1987 as Arthur 1.20. The original aim was to develop something similar to the functionality of the BBC Micro/Master OS, while waiting for the more complicated ARX system to be ready for release. However, Arthur's small size, constant delays of the ARX project, and the realisation that Arthur could be extended to provide a window manager and desktop environment, meant that ARX was eventually dropped and Arthur/RISC became Acorn's main OS. It had a primitive GUI, but could only run one application at a time, and most work was done via the command line.

Arthur 2 became RISC OS 2 and was released in 1989. The GUI was now the main way of interacting with the OS, and it had added some co-operative multitasking. Graphics and sound were also a big improvement. (For comparison, Apple's colour UI OS, System 7, was released in 1991.) Further developments were made in RISC 3.x versions, including a bunch of useful built-in applications and improved font support.

Acorn released the new RiscPC in 1994, with 16 million colour display and the ability to handle up to 256MB of memory (rather than the 16MB of previous machines). RISC OS 3.5 was released to handle these improvements but otherwise was pretty similar to previous releases. Further updates were similarly hardware driven.

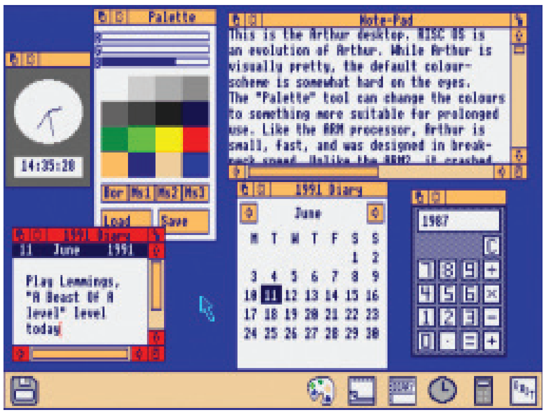
In 1999, following further financial problems, Acorn was renamed as Element 14 Ltd, after which it was bought out. ARM Ltd had been spun off in 1990, and was doing very well, so this move allowed Acorn shareholders to cash out their much more lucrative ARM stock. Element 14 carried on with DSL technology, and a new company, RISCOS Ltd, licensed RISC OS from its eventual new owners. RISC OS 4 was released shortly after, and RISC OS 6 in 2006. RISC OS remains under development. (RISC OS 5 is a separate fork by Castle Technology.) If you fancy giving it a go, you can buy a RISC OS emulator USB stick for Windows, Mac, or Linux, from **www.riscosopen.org**, or RISC OS is also available for the Raspberry Pi.

Meanwhile, Sophie Wilson is still working for Broadcom (who bought out Element 14) and was the chief architect of their Firepath processor. She was awarded the Fellow Award by the Computer History Museum, California, in 2012, was elected as a Fellow of the Royal Society in 2013, and is considered one of the most important women in tech history. Think of her the next time you check your phone. LV



I am writing this in RISC OS 3.11.

RISC OS 3 – an OS that lives on in a version for the Raspberry Pi.



This is the Arthur desktop. RISC OS is an evolution of Arthur. While Arthur is visually pretty, the default colour-scheme is somewhat hard on the eyes. The "Palette" tool can change the colours to something more suitable for prolonged use. Like the ARM processor, Arthur is small, fast, and was designed in break-neck speed. Unlike the ARM2, it crashed

Play Lemmings, "A Beast Of A level" level today

Arthur 1 was tiny – you could run it on a 512K machine with a floppy disk – but full of functionality.

**Juliet Kemp is a scary polymath, and is the author of O'Reilly's *Linux System Administration Recipes*.**

LINUX VOICE MASTERCLASS

Essential Linux tools explained – this month, SSL, the tech that enables secure connections over the web.

# SECURE YOUR WEBSITE WITH SSL ENCRYPTION

SSL Secures the web. Understand what that means with a practical example.

JOHN LANE

SSL is the Secure Sockets Layer. It's the technology that secures the web, and just about everyone who has used a web browser will have heard of it and (especially after the recent Heartbleed incident) its widely-used open source implementation: OpenSSL.

SSL provides a secure communications channel over an insecure network. Its best-known use is to secure the connection between a web server and browser but it also has other uses, such as securing the transmission of email.

OpenSSL is both a toolkit and library that implements SSL. The library is also used by other tools that use cryptography such as SSH. Most distros will install OpenSSL by default or as a dependency of another application like your web browser. Check that you have it:

```
$ openssl version
OpenSSL 1.0.1h 5 Jun 2014
```

You should expect to see at least version 1.0.1g, because this is the one that fixed the Heartbleed bug. If you need to install or update, you should find it in your distro's repository.

SSL is a cryptographic protocol that enables two parties such as a web server and a browser to exchange information securely by encrypting it before sending and decrypting it upon receipt.

Encrypting and decrypting requires a secret, like a password, which is known as a key. A symmetric key can both encrypt and decrypt, whereas an asymmetric key can only do one or the other and therefore requires a key-pair; one for encryption, which can be given to anyone (a public key), and another for decryption that must be kept secret (a private key).

Asymmetric ciphers are more complex than symmetric ones, and therefore have a higher computational overhead. This makes a symmetric cipher preferable for data transmission, but presents the challenge of sharing a symmetric key between two parties previously unknown to each other.

SSL solves this key exchange problem by using an asymmetric cypher to encrypt the symmetric key. Here's what happens when you access a website secured with SSL.

- The client (web browser) connects to the server (website) using a URL that begins with **https:**.
- The Server sends its SSL Certificate to the client.
- The client validates the certificate.
- The client generates a random symmetric key and encrypts it using the public key contained in the certificate.
- The client sends the encrypted symmetric key to the server.
- The server uses its private key to decrypt the symmetric key.
- The server and client encrypt all further communication sent in either direction using the symmetric key.
- At the end of the session, the symmetric key is discarded.
- The process is repeated for further sessions.

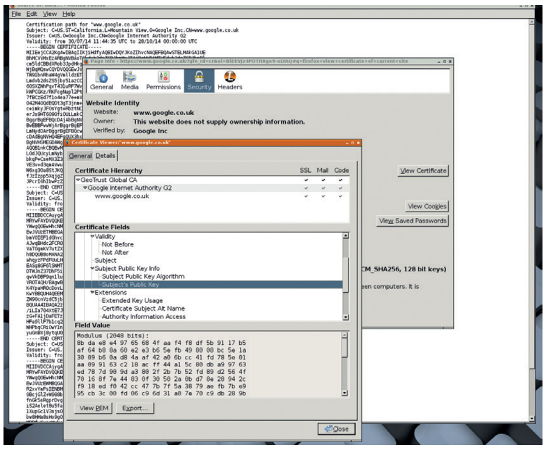
An SSL certificate is like an envelope with the public

## Protocol variations

There are several versions of the SSL protocol, the latest being 3.0, after which it was renamed Transport Layer Security (TLS) and has since seen several revisions. The current TLS version is 1.2. The terms SSL and TLS are often used interchangeably despite their differences (for example, TLS 1.0 is also known as SSL 3.1) and SSL has now become a generic term for describing secure websites.

When a connection is established between a client and server, the protocol negotiates and uses the latest version that they both support.

TLS added the ability for a client to connect to a server's standard port and then negotiate a secure connection. Prior to this, SSL required a dedicated secure port. To be used this way, TLS uses a protocol-specific method to negotiate the switch, and not all protocols include one. StartTLS is the protocol-specific method supported by email protocols. There is an HTTP Upgrade header that allows an HTTP connection to negotiate TLS as specified by RFC2817 but it isn't widely implemented. HTTP continues to use separate ports: 80 for unsecured HTTP and 443 for HTTP over SSL/TLS.


```
Certification path for "www.google.co.uk"
Subject: C=US,ST=California,L=Mountain View,O=Google Inc,CN=www.google.co.uk
Issuer: C=US,O=Google Inc,CN=Google Internet Authority G2
Validity: from 30/07/14 11:44:35 UTC to 28/10/14 00:00:00 UTC
-----BEGIN CERTIFICATE-----
MIIEejCCA2KgAwIBAgIIKjiHdfyAQBIwDQYJKoZIhvcNAQEFBQAwSTELMAkGA1UE
```
Website Identity
Website: www.google.co.uk
Owner: This website does not supply ownership information.
Verified by: Google Inc

Certificate Hierarchy
- GeoTrust Global CA
  - Google Internet Authority G2
    - www.google.co.uk

```
Modulus (2048 bits):
8b da e8 e4 97 65 68 4f aa f4 f8 df 5b 91 17 b5
af 64 b8 8a 60 e2 e3 b6 5e fb 49 80 08 bc 5e 1a
30 09 b6 0a d8 4a af 42 a0 6b cc 41 fd 78 5e 01
aa 09 91 63 c2 18 ac ff 44 a1 5c 80 db a9 97 63
ed 78 7d 90 9d a3 80 2f 2b 7b 52 fd 89 d2 56 4f
70 16 8f 7e 44 83 0f 30 50 2a 0b d7 8e 28 94 2c
f9 18 ed f0 42 cc 47 7b 7f 5a 38 79 ae fb 7b e9
95 cb 3c 00 fd 06 c9 6d 31 a0 7e 70 c9 db 28 9b
```

*Firefox* and the other major web browsers enable you to look inside a server's certificate.

## Is that really you?

As well as transporting the public key, the certificate also permits identity validation, which enables a web browser to confirm that the web server that it is communicating with is the one that it thinks it should be communicating with.

This works because a signed certificate cannot be modified, and the certificate authority (CA) is trusted to perform identity validation before issuing a certificate.

It is, however, up to the CA to decide how it verifies an entity's identity, and this can vary from basic domain validation using **whois** data to extended validation that mandates specific checks that require submission of physical documents like passports or company accounts. EV certificates can only be issued by CAs who pass an independent audit as required by the CA/Browser Forum (see **www.cabforum.org**), making them expensive and difficult to obtain. Domain validated certification may suffice for some applications – low-cost and fee-free certificates are offered by **cacert.org**, **startssl.com** and **comodo.com**.

There is no technical difference between a normal and an EV certificate – both offer the same level of encryption. The EV certificate carries a Certificate Policy Object Identifier and browsers are hard-coded to recognise them (the latest browsers add a green highlight to the address bar).

key inside. It is signed so that the recipient can be confident that the contents have not been altered and can be trusted. This is done by a certificate authority (CA) using its own certificate that is also signed, either by another CA or self-signed.

A CA's certificate that is self-signed is a root certificate and those that are pre-installed in web browsers are trusted implicitly. Web browsers include the root certificates for the major certificate authorities that provide the certificates used by most websites.

A certificate is trusted if its signings can be traced back to a trusted root certificate. This Public Key Infrastructure underpins SSL and is defined by a standard called X.509.

## Get the key

OpenSSL supports the X.509 standard, and you can use it to prepare a certificate signing request that you need to send to a CA to get a new certificate. If you have a certificate, you can use it to sign other certificates. You can even create your own self-signed certificate and be your own certificate authority. But, before you begin, you need your own private key:

```
$ openssl genpkey -algorithm rsa -out private.key
$ chmod 400 private.key
```

You can choose the key generation algorithm, but the usual choice for SSL is "RSA", because it can generate larger keys (up to 4,096) bits. Remember to change the access permissions of the key file to keep it secret. You can then extract the corresponding public key:

```
$ openssl pkey -in private.key -pubout -out public.pem
```

**pem** means Privacy Enhanced Mail, and is a file format that uses base64 encoding. You can specify other formats, such as **der**, which is a binary equivalent of **pem**.

You can further secure a private key by encrypting it with a triple-DES symmetric key. Add **-des3** when generating the private key or encrypt an existing private key with

```
$ openssl pkey -in private.key -des3 -out private-enc.key
```

You will need to enter the passphrase for an encrypted key whenever it is used, making them less useful on servers. A passphrase can be removed:

```
$ openssl pkey -in private-env.key -out private.key
```

You can use PEM format keys with X.509, and you can use OpenSSL to create the certificate signing request (CSR):

```
$ openssl req -new -key private.key -out request.csr
```

This will request some data from you, but the most important field is the Common Name. This must match the domain that the certificate is for. The remaining fields can be completed as desired, or as mandated by the CA. Enter a period **.** for a blank field. Once you have the certificate signing request, you'll need to submit it to a certificate authority using their own procedures.

## Be your own certificate authority

For testing or internal use, a self-signed certificate may be all you need, and creating one is similar:

```
$ openssl req -new -key private.key -x509 -out mycert.crt
```

The **-x509** option is what causes a certificate to be written instead of a CSR. The information required for a CSR applies here too, and you will be prompted to enter it. You can add further parameters such as **-days**, which changes the certificate's validity from the 30 day default.

Self-signed certificates are useful for development and testing and other internal purposes but have otherwise limited use because they lack trust. To get a trusted certificate, you will need to send a certificate signing request to a trusted certificate authority.

You can use your own certificate (whether signed by a trusted CA or self-signed) to sign new certificates.

```
$ openssl x509 -req -in request.csr -CA mycert.crt -CAkey
private.key -out cert.crt
```

You'll need to add **-CAcreateserial** the first time you do this so that OpenSSL creates a serial number file (it's then used automatically for subsequent certificates). Alternatively, you can use **-set_serial** to supply a specific serial number.

We've explained how SSL works and how you can use OpenSSL to create certificates. Next, we'll use a

### LV PRO TIP

You can see the root certificates included in *Firefox* at **mzl.la/1mpp0cV**.

real certificate authority to get a certificate and use it to set up a secure SSL website. SSL gives visitors to a website confidence that it is genuine and that the information supplied to it is safe. If you run a website, you can increase your users' confidence by supporting SSL and you can do this without costing the earth. In fact, you can do it for free.

StartSSL is a certificate authority with trusted root certificates in most major web browsers that offers free one-year domain-verified SSL certificates.

All you need is a domain that you can receive administrative email for – they send a verification email to either the 'postmaster', 'hostmaster' or 'webmaster' address for the domain. There are no additional checks (such as verifying domain ownership) made for these free certificates, but you can pay a fee for extended validation.

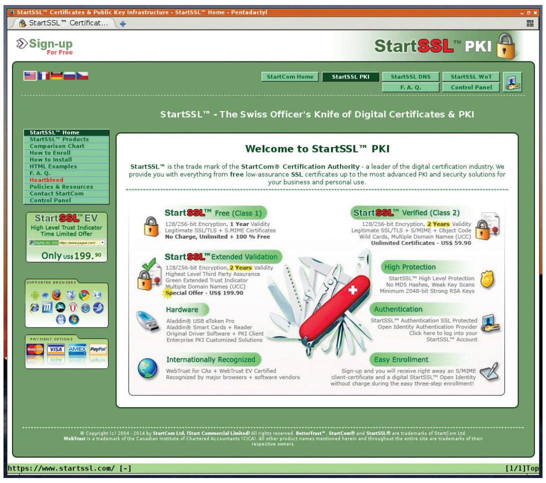
We'll use StartSSL to create a basic, fee-free domain-validated certificate. These are good for one domain (eg **example.com**) and one subdomain (eg **shop.example.com**) which means that one certificate could theoretically be used for two sites. However, given that you can create as many certificates as you wish, there isn't really any limitation on what you can do (you can also get wild-card multiple domain certificates, but they are not free).

The first thing to do is to sign up for an account at **www.startssl.com**. You have to enter your personal details including address and phone number, and these may be used depending on the level of validation that you require.

You will be sent a verification email containing a code that you need to enter into the website. It then sends a second email containing a link and another verification code. Clicking that link and entering the code takes you to a Generate Private Key page.

The private key is for a new client certificate that will be installed in your browser and will be used to authenticate you with StartSSL instead of a username and password (using an SSL certificate to authenticate onesself is a little-used capability of web browsers that few people are aware of).

Leave the drop-down with 'High Grade' selected and click on Continue to generate the key. Next, press



StartSSL™ Certificates & Public Key Infrastructure - StartSSL™ Home - Pentadactyl

StartSSL™ Certificat...

Sign-up For Free

StartSSL™ PKI

StartCom Home | StartSSL PKI | StartSSL DNS | StartSSL WoT | F. A. Q. | Control Panel

StartSSL™ - The Swiss Officer's Knife of Digital Certificates & PKI

StartSSL™ Home
StartSSL™ Products
Comparison Chart
How to Enroll
How to Install
HTML Examples
F. A. Q.
Heartbleed
Policies & Resources
Contact StartCom
Control Panel

StartSSL™ EV
High Level Trust Indicator
Time Limited Offer
Only US$199.90

SUPPORTED BROWSERS

PAYMENT OPTIONS

Welcome to StartSSL™ PKI

**StartSSL™** is the trade mark of the **StartCom® Certification Authority** - a leader of the digital certification industry. We provide you with everything from **free** low-assurance **SSL** certificates up to the most advanced PKI and security solutions for your business and personal use.

StartSSL™ Free (Class 1)
128/256-bit Encryption, **1 Year** Validity
Legitimate SSL/TLS + S.MIME Certificates
**No Charge, Unlimited + 100 % Free**

StartSSL™ Verified (Class 2)
128/256-bit Encryption, **2 Years** Validity
Legitimate SSL/TLS + S/MIME + Object Code
Wild Cards, Multiple Domain Names (UCC)
**Unlimited Certificates - US$ 59.90**

StartSSL™ Extended Validation
128/256-bit Encryption, **2 Years** Validity
Highest Level Third Party Assurance
Green Extended Trust Indicator
Multiple Domain Names (UCC)
**Special Offer - US$ 199.90**

High Protection
StartSSL™ High Level Protection
No MD5 Hashes, Weak Key Scans
Minimum 2048-bit Strong RSA Keys

Hardware
Aladdin® USB eToken Pro
Aladdin® Smart Cards + Reader
Original Driver Software + PKI Client
Enterprise PKI Customized Solutions

Authentication
StartSSL™ Authentication SSL Protected
Open Identity Authentication Provider
Click here to log into your
StartSSL™ Account

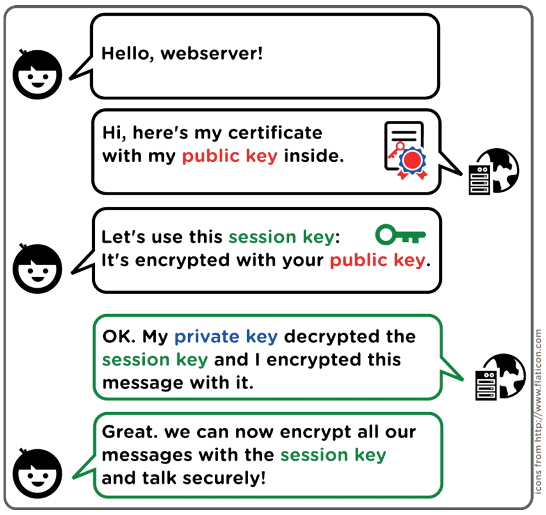
Internationally Recognized
WebTrust for CAs + WebTrust EV Certified
Recognized by major browsers + software vendors

Easy Enrollment
Sign-up and you will receive right away an S/MIME client-certificate and a digital StartSSL™ Open Identity without charge during the easy three-step enrollment!

© Copyright (c) 2004 - 2014 by **StartCom Ltd. (Start Commercial Limited)** All rights reserved. **BetterTrust™**, **StartCom®** and **StartSSL®** are trademarks of StartCom Ltd.
**WebTrust** is a trademark of the Canadian Institute of Chartered Accountants (CICA). All other product names mentioned herein and throughout the entire site are trademarks of their respective owners.

```
https://www.startssl.com/ [-]                                   [1/1]Top
```

StartSSL offers SSL certificates ranging from fee-free domain-validated certificates through to the extended validation certificates necessary to turn your browser's address bar green.


Hello, webserver!
Hi, here's my certificate with my public key inside.
Let's use this session key: It's encrypted with your public key.
OK. My private key decrypted the session key and I encrypted this message with it.
Great. we can now encrypt all our messages with the session key and talk securely!
icons from http://www.flaticon.com

The beginnings of a typical SSL conversation.

Install to install it into in your browser, which should respond with a pop-up confirming the certificate installation. The web page then displays links explaining how to back up the key that was just installed. Do that, then click the Finish button.

With your client certificate installed, you can click on the Control Panel button. The Authenticate button there uses your client certificate to authenticate you, and is how you log in to the StartSSL website on return visits.

Once authenticated, you can use the control panel's tool box, certificates wizard or validations wizard.

## Domain Validation

Before you can create a certificate, you must perform the domain validation, and you can validate as many domains as you want using the Validations wizard.

You enter a domain and it sends an email to an administrative address for the domain (your choice of either 'postmaster', 'hostmaster' or 'webmaster') containing a validation code that needs to be entered on the website to complete the validation.

The validation lasts for 30 days, but you can re-validate whenever you need to.

You use the Certificates wizard to create certificates for validated domains. You can supply a Certificate Signing Request (CSR) or have StartSSL generate one, including a private key for you. While this convenience might sound nice, and StartSSL states that no copies of generated private keys are kept at any stage, it's a really bad idea for anyone but you to have access to your private key. For this reason we recommend that you use a CSR! It's easy to create a CSR using OpenSSL on your own machine:

```
$ openssl req -new -key private.key -out request.csr
```

The **-key** option specifies the private key to use. If it's omitted, a new private key will be generated and you will be prompted to supply the required information.

StartSSL only uses the public key embedded in the CSR and ignores any applicant data so, when creating the CSR, you can just accept the defaults or enter meaningful detail; it doesn't matter.

In the StartSSL Certificates wizard, choose 'Web Server SSL/TLS Certificate' and, to use a CSR, press the skip button to bypass the private key generation.

Gather the text of the CSR (eg 'cat request.csr') and paste it into the box in the Wizard. The response indicates success and reminds you that all content of the certificate signing request is ignored except its public key. Press 'Continue'.

You are then presented with your validated domains; select the relevant one. You are then presented with a box to enter one subdomain (you'll need to pay if you need a certificate for multiple domains or sub-domains). Enter a subdomain (like 'www') and press Continue.

After a final confirmation of the domain and sub-domain, press Continue once more. The certificate is displayed on the screen. Copy and paste it into a local file. It's customary to use a **.crt** file extension, like **server.crt**. Once the certificate has been obtained, the CSR can be discarded. You now need to install the certificate and associated private key on your web server.

## Webserver configuration

Assuming that you have an *Apache* webserver already installed and working without SSL, we'll now configure a new SSL virtual host, and afterwards another one for the subdomain.

Now we need the private key and the new site certificate file from StartSSL. The exact location for the *Apache* configuration depends on your Linux distribution. On Arch Linux it's at **/etc/httpd/conf**. Copy the **private.key** and **server.crt**, for example:

```
$ scp private.key server.crt root@webserver:/etc/httpd/conf
```

Then edit the default *Apache* SSL configuration, which may be found in the **extra** subdirectory and called **httpd-ssl.conf**. Make the following changes, all within the **<VirtualHost _default_:443>** block:

- Set **DocumentRoot** to the directory where this virtual host's files will reside (eg **/srv/https**)
- Set **ServerName** to the domain covered by the server certificate and the **https** port (443) (eg **mydomain.com:443**).
- Set **SSLCertificateFile** to the path of the server certificate (eg **/etc/httpd/conf/server.crt**)
- Set **SSLCertificateKeyFile** to the path of the private key (eg **/etc/httpd/conf/private.key**).

If your certificate has intermediate certificate authoritity certificates, concatenate them into a single file and set **SSLCertificateChainFile** to its path (you don't need to do this for StartSSL but may need to if you get your certificate elsewhere). If you have set **DocumentRoot** to a new directory path, a **<Directory>** entry may be required to make it accessible:

```
<Directory "/srv/https">
  Order allow,deny
  Allow from all
</Directory>
```

Now, edit the main *Apache* configuration file, **httpd.conf**, to uncomment the line that includes the SSL configuration:

```
# Secure (SSL/TLS) connections
Include conf/extra/httpd-ssl.conf
```

And, finally, restart *Apache*, in the appropriate manner for your system. Systemd users can use

```
$ systemctl restart httpd
```

Assuming there is content at the document root, pointing a browser to the new site should work with no security warnings and the browser should display its padlock icon to show that the connection is secure.

*Apache*'s Name-based virtual hosts support enables you to configure further SSL virtual hosts in a similar way. Find the **Listen 443** stanza in **httpd.conf** and add another to enable it on port 443:

```
  NameVirtualHost *:443
```

This relies on Server Name Indication (SNI) to resolve hosts by name, as described in the boxout, left. With name-based virtual hosts configured, you can add further blocks for additional virtual hosts:

```
<VirtualHost *:443>
 DocumentRoot "/srv/https/subdomain.mydomain.com"
 ServerName servername.mydomain.com:443
 SSLCertificateFile /etc/httpd/conf/private.key
 SSLCertificateKeyFile /etc/httpd/conf/server.crt
</VirtualHost>
```

A server re-start is required for the configuration changes to take effect. Restart the server and point your browser to the virtual host's URL.

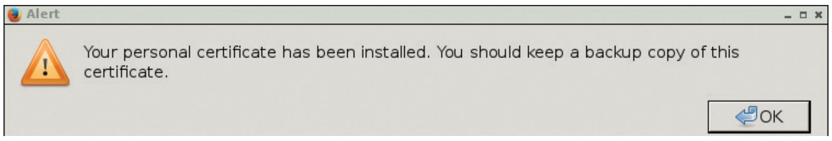
John Lane is a technology consultant with a penchant for Linux. He helps new business start-ups make the most of open source.

### SSL, virtual hosts and SNI

Historically, it wasn't possible to host multiple SSL hosts on a single IP address and port because the web server needs to know the host name to choose the correct certificate, but this information is wrapped up in the encrypted content and, therefore, can't be accessed until the encryption is established. This has been solved by Server Name Indication (SNI), an extension to the **https** protocol that presents the host name during the pre-encryption handshake.

Support isn't universal, however, and it's likely that a browser that doesn't support it will be offered the incorrect certificate, because the SNI is missing. If a web server receives a request without SNI, it will fall back to a default certificate. *Apache* uses the first virtual SSL host's certificate when this happens. This may result in an unexpected certificate being returned to the browser, which may trigger a security warning. You can change this behaviour by enabling **SSLStrictSNIVHostCheck** so it returns a 403 error page instead.

This won't be an issue if all the virtual hosts share the same certificate (perhaps they are subdomains or you have a certificate that covers multiple domains).



Your personal certificate has been installed. You should keep a backup copy of this certificate.

StartSSL installs a client certificate into your browser to authenticate you.

### PRO TIP

You can access your certificates at Toolbox > Retrieve Certificate on the StartSSL website.

# LINUXVOICE DVD 008

Distros, videos, podcasts – get the latest Linux goodness today!

## SOMETHING FOR EVERYONE

Welcome to the DVD! We spent a lot of time umming and ahhing about which distro should take centre stage this month, and ultimately we went for CentOS. It's true that it's not the most cutting-edge distro out there, but in its Red Hat Enterprise Linux form it has brought Linux and open source to tens of thousands of businesses around the world. Kudos to Red Hat for supporting the CentOS community, and effectively giving away its flagship product for free. (Of course, many CentOS dabblers will go on to buy RHEL support subscriptions, so the company benefits in the end.)

But that's just the start: we also have the latest release of Manjaro, one of the hottest up-and-coming distros, which makes Arch Linux more accessible. Arch won our distro group test last issue, so if you've been dying to try it but a bit daunted by the installation process, here's your chance.

Then there's the snazzy Elementary OS, videos and pocasts. Enjoy exploring!

Mike Saunders, Disc Editor
mike@linuxvoice.com

Ultra-reliable desktop and server distro

## CentOS 7 (64-bit)

Red Hat Enterprise Linux, rebuilt for the community.

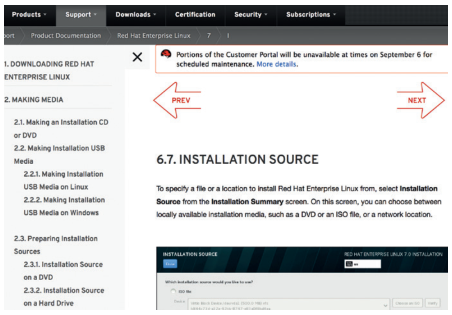
We love playing with bleeding-edge software at Linux Voice HQ. Trying the latest apps, poking around in new window managers, fixing breakage when init systems change – it's all part and parcel of being an ever-inquisitive Linux user who loves to explore under the surface. We know you love tweaking and customising too, which is why Arch has become so popular.

But sometimes you need more stability and consistency – especially on servers, or in businesses. You want your distro to be supported for years, to be well tested, and to not suddenly break with the next round of updates. CentOS is exactly one such distro, built from the sources of Red Hat Enterprise Linux. It will be supported until at least 2020, so if you install it now, you can still be rocking CentOS 7 at the end of the decade. It'll just keep chugging on and on, so it's perfect for production machines where reliability is paramount, and you just want things to keep working day-in, day-out.

### Booting it up

On the Linux Voice DVD you'll find the 64-bit version of CentOS 7, including the Gnome desktop and various productivity apps, directly bootable from the disc. Just pop it in your drive and reboot, and you should be able to select it from the menu. (If you need to change the boot order in your BIOS, consult your PC's documentation.) You'll arrive at the desktop in live mode, where you can double-click the installer icon on the desktop to copy the distro to your hard drive.

System requirements are 512MB RAM and 10GB hard drive space, but that's mainly for server usage, where you don't need graphics – on the desktop, it's better to have at least 1GB of RAM. If you need any help with CentOS, or just want to learn more, there are heaps of resources on the distribution's website at **www.centos.org**.



1. DOWNLOADING RED HAT ENTERPRISE LINUX
2. MAKING MEDIA
2.1. Making an Installation CD or DVD
2.2. Making Installation USB Media
2.2.1. Making Installation USB Media on Linux
2.2.2. Making Installation USB Media on Windows
2.3. Preparing Installation Sources
2.3.1. Installation Source on a DVD
2.3.2. Installation Source on a Hard Drive

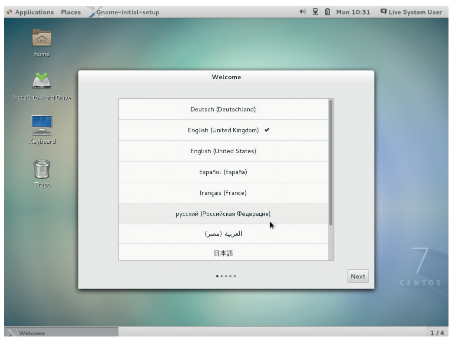
Portions of the Customer Portal will be unavailable at times on September 6 for scheduled maintenance. More details.

6.7. INSTALLATION SOURCE

To specify a file or a location to install Red Hat Enterprise Linux from, select **Installation Source** from the **Installation Summary** screen. On this screen, you can choose between locally available installation media, such as a DVD or an ISO file, or a network location.

INSTALLATION SOURCE
RED HAT ENTERPRISE LINUX 7.0 INSTALLATION
Which installation source would you like to use?
ISO file

CentOS isn't officially supported by Red Hat, but RHEL docs are still applicable.



CentOS sports the "classic" version of Gnome 3, so it's more like the previous desktop release.

LINUXVOICE DVD
SPECIAL TRIPLE-BOOTING DISC!
CentOS 7
Enterprise grade Linux: ultra reliable distro for desktops and servers
Manjaro
Arch Linux power with a slick Xfce interface
Elementary OS
Beautiful Ubuntu respin with a host of newbie-friendly apps
Open index.html for all the details

User-friendly installer for Arch

## Manjaro 0.8.10 (32-bit)

**The raw power of Arch Linux, with a slick front-end.**

If you read our distro grout test in the previous issue of Linux Voice, you'll have seen that Arch Linux won in various categories, including packages and documentation. Thanks to the Arch User Repository, almost every piece of free and open source software under the sun is available in Arch – and often in a much more up-to-date form than in other distros. The documentation on the wiki, meanwhile, is second to none, and it's often supremely useful even if you run an entirely different version of Linux.

So what's the catch? Why isn't the whole world running Arch? Well, there are a few reasons, but perhaps the most notable is its learning curve. Arch Linux doesn't hold your hand, and it's certainly not targeted at completely new Linux users. It expects you to read the documentation thoroughly, and keep track of major changes to the underlying system. Because Arch is a rolling release distro, which means you get the latest software all the time (yay!), there are occasional breakages as well (not so yay). But if you follow the right mailing lists, you should be able to fix any glitches that occur pretty swiftly.

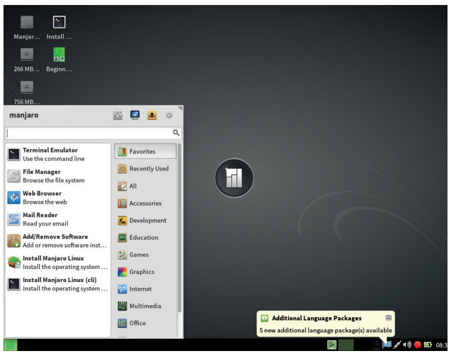
Anyway, if you've been using Linux for a while, you're familiar with the command line and you're tempted to try Arch, but you've always been put off by the lengthy beginner's installation guide (see **http://tinyurl.com/archnewbs**), Manjaro is exactly what you need. It's essentially a flavour of Arch Linux designed to get you up and running as quickly as possible, providing you with an attractive desktop environment and all of the usual goodness of Arch (such as the mighty *Pacman* package management system). Even if you're a long-time Arch user, Manjaro is still great for those times when you want to set up a new box with Arch and don't have much time to spare. It sports a graphical installer based on Ubuntu's, so getting it onto your hard drive is a familiar process.

### Pretty as a picture

The 32-bit version of Manjaro 0.8.10 is directly bootable from the Linux Voice DVD, so you don't need to burn anything to a disc or use a USB key. Just select it from the boot menu and you're ready to go. After it loads, you'll land at a neatly polished Xfce desktop, which is accompanied by a bunch of familiar desktop applications.

This is running in live mode – ie straight from the DVD – so it won't touch anything on your hard drive until you tell it to. Live distros are always useful for testing a machine's Linux compatibility before committing to an install, or for those times when you're forced to use a machine that only has Windows installed, but you need a quick Linux fix.

If you like what you see and you're ready to install Manjaro to your hard drive, double-click the appropriate installer icon on the desktop. There are two installers:



Manjaro has KDE and Openbox editions, but we've gone for the speedy Xfce on the DVD.

we recommend that most users go with the graphical one, as it's based on Ubuntu's installer and is therefore well tested, but a text-mode alternative is also available should you prefer it. The graphical installer gets you set up with just a few mouse clicks, and you can of course install the distro alongside another operating system if you want a multi-boot machine. When the installation is done, shut down the live distro and remove the disc from your drive. You can now boot Manjaro directly from your hard drive. Enjoy!

If this is your first time in an Arch-based distro, we strongly recommend spending some time on the wiki at **https://wiki.archlinux.org**. In particular, it's worth reading the Arch Way, FAQ and *Pacman* pages. It might seem like a lot to go through, but you'll really grasp the design decisions behind Arch, and once you've mastered *Pacman*, you'll find it difficult to ever go back to another package management system. Arch can be a demanding beast, but it's totally worth it in the end.

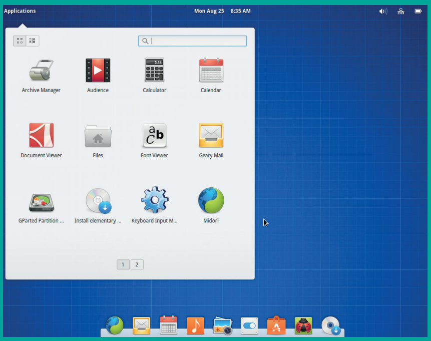
## Elementary OS (32-bit)

**Freya Beta 1 is here for your testing pleasure.**

**We've been following the progress of Elementary OS for a while: it's one of the most attractive and well presented distros we've ever seen, with a special focus on usability and having a core set of software for day-to-day computing. Many have argued that it apes Mac OS X too closely – but then, whether you like Apple or not, it's hard to deny that the company doesn't have a knack for spit-shine.**

**You can try the Beta 1 snapshot of Freya, the upcoming Elementary OS release, by booting your PC from the Linux Voice DVD and selecting it from the menu. After a few moments you'll arrive at the desktop, where you can explore the included software range (see especially the dock at the bottom of the screen). Elementary showcases some of the best FOSS programs out there, but has some home-brewed tools too.**

**It's important to note that this release is for testing and curious onlookers, and shouldn't be installed on production machines. It will have bugs and glitches – but it's still fascinating to see in action.** LV



Visit **www.elementaryos.org** for the full lowdown on this frill-laden distro.

# /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!

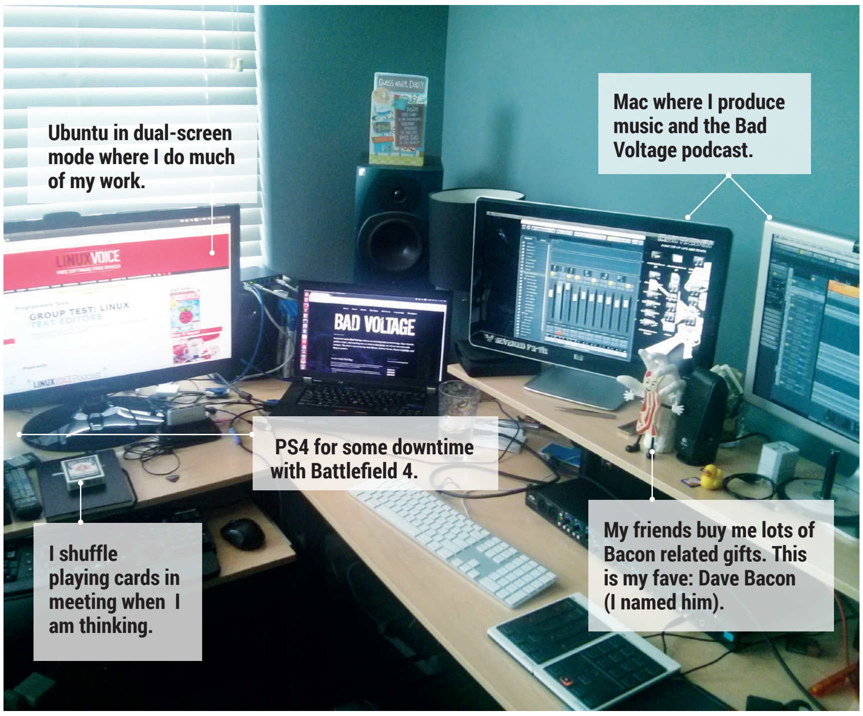
Developing software can be hazardous. I don't mean the risks of RSI or health problems associated with over-caffeination. I mean receiving death threats.

There has been a lot of media attention focussed on the threats to various people involved in the games scene recently. From the coverage it would be easy to deduce that the internet is full of unpleasant teenage boys with disturbing attitudes towards women.

However, the problem isn't limited to games or boys. There has been a growing incidence of this sort of threatening behaviour, or at least of people who have decided they are not going to put up with it. Most recently, Seth Vargo, who worked at cloud enabling software company Chef (**www.getchef.com**) has quit, citing unwanted death threats from the community as one of the motivators. You can read his blog here: (**https://sethvargo.com/leaving-chef**).

In my day things were more personal – I got death threats in the mail. These days threats can be delivered in moments by hastily opened dummy accounts. One problem with such behaviour is that it shuts down any reasonable discussion. An *agent provocateur* need only lob a molotov of threats from within an otherwise sensible protest and everything becomes all CS gas and water cannon. When everyone is shouting, nobody is listening.

Either people are going to need to grow up and realise that threatening behaviour never does their cause any good (unlikely) or at the very least, communities are going to need to be managed better to make this sort of thing have consequences. It seems impossible to do that without some restrictions on web anonymity, which seems like a high price to pay, but we also can't expect developers to put up with threats.


Ubuntu in dual-screen mode where I do much of my work.
Mac where I produce music and the Bad Voltage podcast.
PS4 for some downtime with Battlefield 4.
I shuffle playing cards in meeting when I am thinking.
My friends buy me lots of Bacon related gifts. This is my fave: Dave Bacon (I named him).
LINUX VOICE
BAD VOLTAGE

## My Linux setup Jono Bacon

The man at the helm of the Bad Voltage podcast, Xprize Foundation community chap and formerly Ubuntu person.

**Q What version of Linux are you using at the moment?**

**A** On my laptop I am running Ubuntu and on the desktop machine, which I use for producing Bad Voltage as well as recording music, I'm running Mac OS X. I also run Ubuntu on my servers.

**Q What desktop do you prefer (as if we can't guess)?**

**A** My desktop of choice is Unity. I like how it just gets out of my way and lets me focus on my work.

**Q What was the first Linux setup you ever used?**

**A** I started out with Slackware 96 back in 1998. I then moved over to using Red Hat, then Mandrake, a quick flirt with Corel Linux, then to Debian, and finally Ubuntu. I have never considered anything else since Ubuntu.

**Q What Free Software/open source can't you live without?**

**A** A few things; *Firefox*, *Chromium*, *Gimp*, *Inkscape*, and *XChat* on my laptop. On my servers I couldn't live without Wordpress and Discourse (and their associated servers/databases).

**Q What do other people love but you can't get on with?**

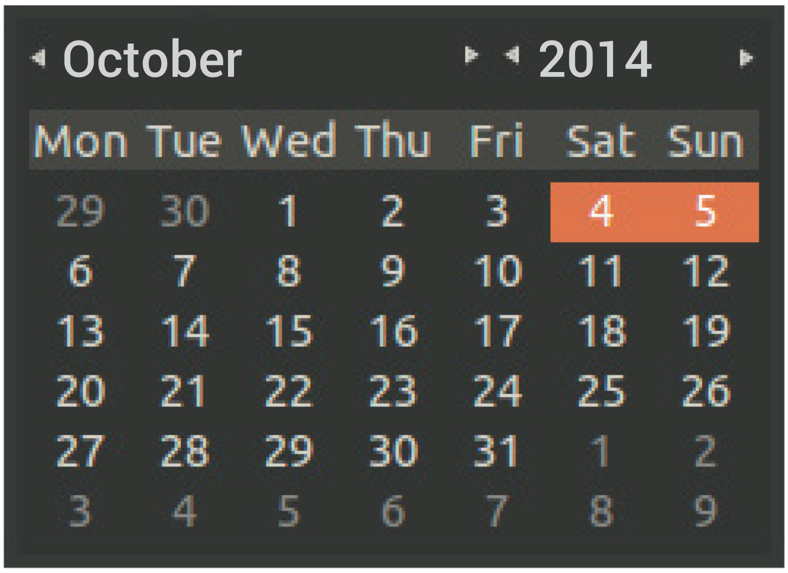
**A** A bunch of people use KDE, and I have tried, but it just doesn't work with my brain. This isn't KDE's fault, my brain is stupid.

# OGGCAMP

EST. 2009

LEARN / TEACH / PLAY

A Free Culture Unconference



October 2014

| Mon | Tue | Wed | Thu | Fri | Sat | Sun |
|---|---|---|---|---|---|---|
| 29 | 30 | 1 | 2 | 3 | 4 | 5 |
| 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| 20 | 21 | 22 | 23 | 24 | 25 | 26 |
| 27 | 28 | 29 | 30 | 31 | 1 | 2 |
| 3 | 4 | 5 | 6 | 7 | 8 | 9 |

SCO
United Kingdom
Republic of Ireland
London
Guernsey
Amst
Lux

Presentations from attendees welcome, bring your idea or project!

FREE ENTRY

Tickets and info at oggcamp.org

Supported by:

The OGGCAMP Community

OggCamp is seeking sponsors! Tweet @oggcamp or check the website for details

LINUXVOICE
FREE SOFTWARE | FREE SPEECH

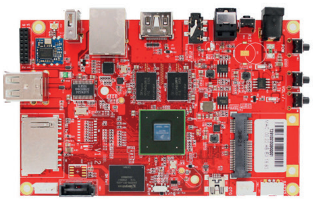
ubuntu
Supported by Canonical

# Matrix
## Arm Mini PC

- TBS TUNER SUPPORT
- FREESCALE QUAD CORE
- 100% OPEN SOURCE
- XBMC, VDR, TVHEADEND

NOW WITH openelec
EMBEDDED LINUX ENTERTAINMENT CENTER

NOW ONLY
£119
FREE UK MAINLAND DELIVERY



| | |
|---|---|
| System-on-a-chip (SoC) | Freescal MCIMX6Q5EYM10AC |
| CPU | Quad ARM Cortex-A9 at 1.0GHz |
| GPU | Vivante GC2000, Quad core GPU, Quad IPU |
| RAM | 2GB DDR |
| USB 2.0 ports | 3x USB 2.0 |
| Audio & Video interface | HDMI port |
| | 3.5 mm jack |
| Storage | eMMC 16GB |
| | 1x SD card slot |
| | 1x TF card slot |
| Network | 10/100/1000 wired Ethernet |
| | WIFI IEEE 802.11n/b/g |
| Power input | 5V, 3A |

RATED 4/5
BY YOUR VERY OWN: BEN EVERARD

SOUNDS TOO GOOD TO BE TRUE? IT'S NOT!
UNLIKE OTHER XBMC SOLUTIONS, OPENELEC IS NOT BASED ON UBUNTU.
IN FACT, IT'S NOT BASED ON ANY LINUX DISTRIBUTION;
OPENELEC HAS BEEN BUILT FROM SCRATCH SPECIFICALLY TO ACT AS A MEDIA CENTER.

"XBMC performance is fantastic and could make the Matrix one of the best frontends you could buy"

"The hardware looks good, feels solidly made and works well"

xbmc ubuntu

WWW.TBSCARDS.CO.UK

PCI EXPRESS LTD, UNIT 3 BAYTREE AVENUE, KINGSTON ROAD, LEATHERHEAD, SURREY, KT22 7UE