REVIEWED: AMD THREADRIPPER 1950X

LINUX FORMAT

Get into Linux today!

65 Pages of tutorials and features

- Get started with Ubuntu Snaps
- Improve Ubuntu swap files
- Better photography with Linux

Coding Academy: Write a SlackBot and try Kotlin

RASPBERRY Pi PROTECTION!

- Monitor hacker intrusion
- Scan for Wi-Fi weakness
- Build a honeypot PC
- Secure your USB storage

Oggcamp!

"People liked to stand up and talk. This is the best result I could've asked for"

Mark Johnson on organising Linux events

Plus: Pi User

Your Raspberry Pi companion

- Play Wolfenstein 3D
- Build RFID-based projects
- Physical computing with FLASK

Desktops
Discover KDE
20 years of development to build the best Linux desktop

Roundup
Encryption tools
How you can protect your data on- and off-line

Future

Did you know that European forests, which provide wood for making paper and many other products, have grown by 44,000km^2 over the past 10 years? That's more than 1,500 football pitches every day![†]

Love magazines? You'll love them even more knowing they're made from natural, renewable and recyclable wood

[†]UNFAO, Global Forest Resources Assessment 2005-2015.

Two Sides is a global initiative promoting the responsible use of print and paper which, when sourced from certified or sustainably managed forests, is a uniquely powerful and natural communications medium.

There are some great reasons to #LovePaper
Discover them now,
twosides.info

TWO SIDES

LINUX FORMAT
Get into Linux today!

What we do

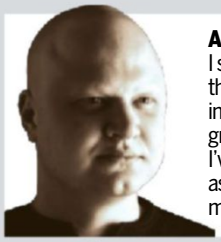
- We support the open source community by providing a resource of information, and a forum for debate.
- We help all readers get more from Linux with our tutorials section – we've something for everyone!
- We license all the source code we print in our tutorials section under the GNU GPL v3.
- We give you the most accurate, unbiased and up-to-date information on all things Linux.

Who we are

This issue we asked our experts: KDE is 20 years old, so what have you achieved in the past 20 years, other than getting a bit heavier...

Jonni Bidwell
I fulfilled my childhood dream of working for a magazine ending in "Format". And I read Proust (spoiler alert: nothing happens. Also: those sentences, full of subclauses, rambling meanderings and not a full stop till halfway down the page, wouldn't get past our production "team".)

Nate Drake
Twenty years ago I began my tenure at a Hogwarts-esque secondary school. Notable highlights of my colourful time since then include crashing a tank, sipping vodka in Red Square, studying modern languages (until I discovered Linux) and rediscovering the lost city of Ubar.



Adam Oxford
I still open up Facebook every now and then, so imagine my surprise when an invite to a 20th anniversary reunion of my graduate class popped up in my newsfeed. I've been in the workplace exactly as long as KDE, and I've been through about as many controversial reinventions too.

Les Pounder
I've been a Linux user for almost 20 years, and in that time I've grown from a Windows user, confused by the sheer choice Linux presented to him, to a confident Linux user who has learnt and shared his skills with others, who in turn have passed on those skills to others.

Shashank Sharma
Coincidentally, 2017 marks my 31st year on this spinning blue ball and is exactly 20 years since I discovered GPL and open source software. I used the latter to earn a living as an author and tech writer. And I was inspired by the former to pursue a career as a trial lawyer.

Securberry Pi

Security doesn't have to be all doom and gloom. Monitoring, protecting and defending your networks can be fun... if you take the right approach. Part of making it fun is – love it or loathe it, there's no denying it – the Raspberry Pi, which enables anyone to undertake a host of real-world complex projects for very little cost. The support this device now wields in the form of industry-leading names such as Kali Linux, Debian, Ubuntu and Fedora, means that this tiny board can be put to any use you can think of.

Helping to protect your networks this issue, we're creating a Raspberry Pi honeypot that you can deploy on any network to lure and ensnare unwitting hackers. To block malware we'll cover Pi-hole, now on version 3, to protect your entire network from ad-based annoyances; USB Sanitizer will scan and safely copy suspect storage devices, and finally we use Kali Linux to wardrive your networks for security holes. The Raspberry Pi ensures these solutions are easy to deploy, but with a few tweaks you should be able to use them on any Linux system.

Outside of the Raspberry Pi the Linux world keeps on ticking. KDE has hit 20 years of development (so has Gnome, while Debian has passed 24!) and we look at its growth and where it's heading. We report live from the floor of Oggcamp 2017, the greatest unconference in the world; explore Ubuntu Snaps, the new Ubuntu swap file system and encryption tools; and review the low-end AMD Ryzen 3 1300X and the monster 32-thread running AMD ThreadRipper 1950X processor–that can compile the entire Linux kernel in just 37 seconds!

Do please keep on writing in (**linuxformat@futurenet.com**) to let us know what you're doing with Linux, what you'd like us to cover and what we've done wrong in this issue!

Neil Mohr Editor
neil.mohr@futurenet.com

LINUX FORMAT

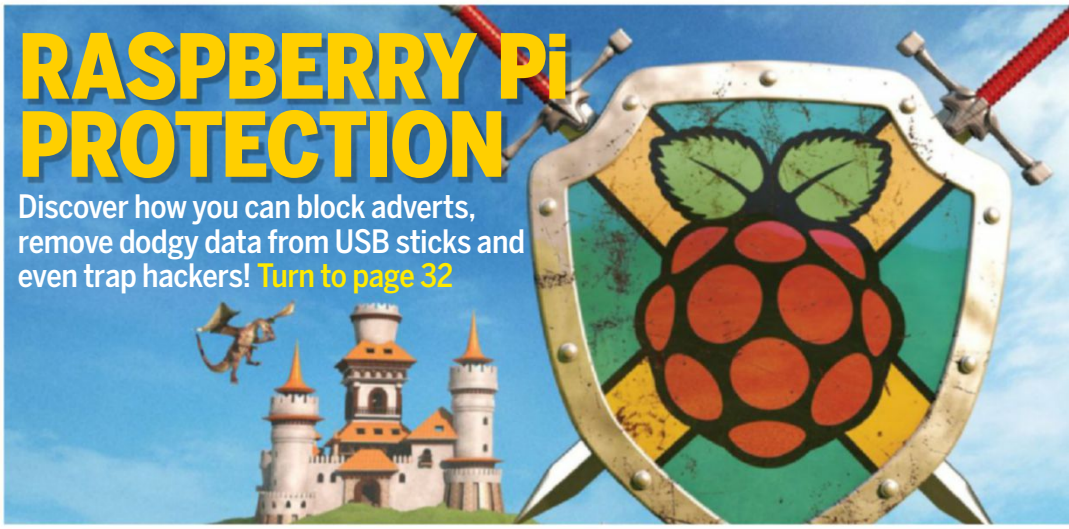
Contents

"Thank goodness I was never sent to school; it would have rubbed off the originality." – Beatrix Potter

Reviews

The best-value processor option we've seen in a long time.

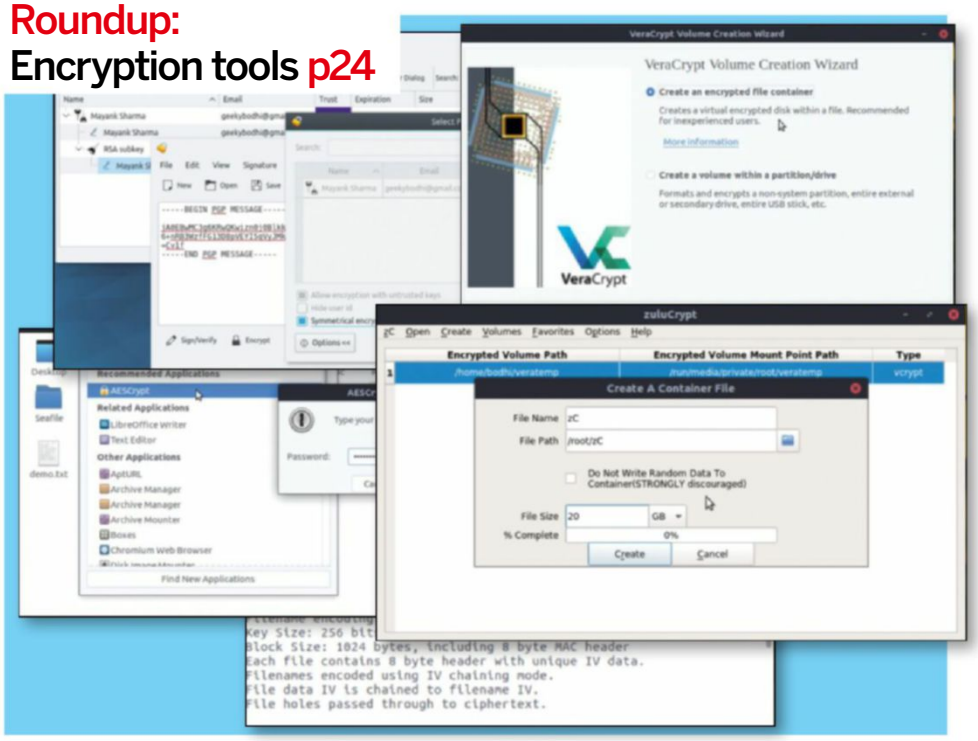
Living the apocalypse in the Canadian wilderness. It warms our cold hearts.



RASPBERRY Pi PROTECTION

Discover how you can block adverts, remove dodgy data from USB sticks and even trap hackers! Turn to page 32

Roundup: Encryption tools p24



Oggcamp 2017

"I didn't prepare for my first talk – I just did it on the spur of the moment!"

Rachel Wong gets into the unconference spirit p42

On your FREE DVD

Sparky Linux 5.0 64-bit

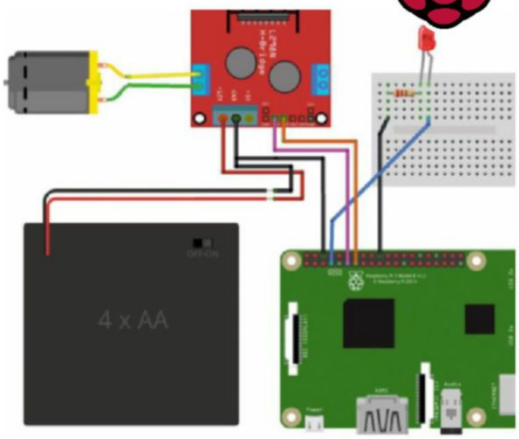
BackBox Linux 5 32-bit

Rescatux 0.41 32- and 64-bit

» Only the best distros every month p96

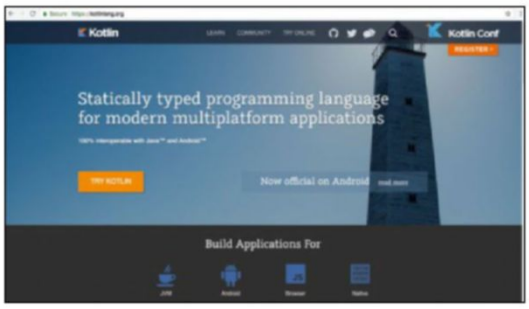
Subscribe & save! p30

Raspberry Pi User

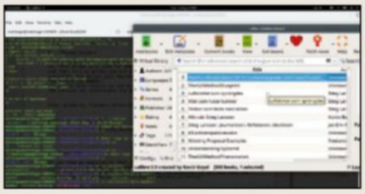


In-depth...

Coding Academy



Tutorials



› **Don't fight the change, embrace it.**

Regulars at a glance

› **Our subscription team is waiting to take your call.**

MOBILE

Who's for Android 8?

There's a new Android out, but if you're not a fan of Google's walled garden then a kickstarter has launched for an open source alternative.

Google has revealed the next version of its mobile operating system: Android 8.0. While the yearly update of a mobile OS isn't the most exciting news, with Android being the most popular mobile operating system in the world, it's likely that many readers own an Android device.

So what can we expect from this upcoming release, called Oreo? For starters, devices running Android 8.0 will benefit from Picture in Picture mode, enabling people to minimise apps, such as YouTube, to a corner of the screen while using another app.

Another feature restricts background apps and manages the functions they use while in the background, which should help improve battery life. Google is also claiming that it has introduced machine learning into its press-to-hold gestures, so when you select certain items by pressing and holding your finger on the touchscreen, context-sensitive options will appear.

A post on the Android Developers Blog (**http://bit.ly/2iIXYa1**) by Sami Tolvanen, senior software engineer, Android Security, goes into detail about some of the more advanced security-focused updates of Android 8.0. One such move is the further hardening of the kernel in an attempt to reduce the frequency and impact of security bugs within it.

In the blog post, Tolvanen highlights four new security features that have been backported from upstream Linux. These are hardened usercopy functions, which adds bounds checking to help developers spot malicious use or bugs, and Prevailed Access Never (PAN) emulation, which brings the feature found in ARM v8.1 devices to other hardware. According to the blog, hardened usercopy and PAN emulation has helped to find and fix bugs in four kernel drivers in Pixel devices.

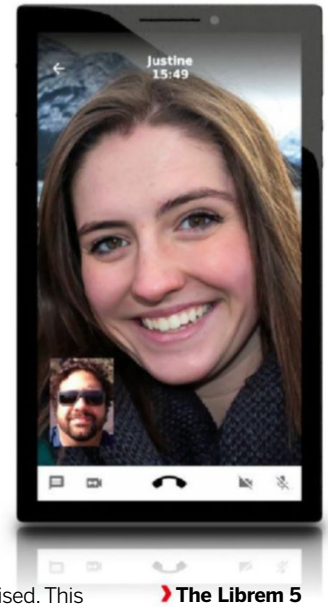
The other two new security features for the kernel is kernel address space layout randomisation (KASLR), which helps avoid kernel vulnerabilities by randomising the location where kernel code is loaded on each boot, and post-init read-only memory, which creates a memory region that becomes read-only after the kernel has been initialised. This protects data that needs to be written during initialisation, but then should not be modified afterwards.

Android 8.0 is out now, however, as with previous releases the newer Google devices, such as the Pixel, Pixel XL/C and Nexus 6P/5X, will get the update first, with the myriad of Android device makers implementing Oreo over (a long) time. For a full list of new features, see **http://bit.ly/2wy5ZEb**.

> **"The Librem 5 phone will be the world's first ever IP-native handset"**

If you're sick of waiting for these yearly updates to grace your smartphone, or you're no fan of Google's walled garden approach to Android, then there are plenty of truly open source alternatives. While Ubuntu Phone has bitten the dust (see **LXF224**), a new crowd-funded open source phone, called the Librem 5, is currently raising funds (**https://puri.sm/shop/librem-5**). This phone will run on free and open source software, with no hint of Android's restrictions. It runs on the Debian-based PureOS (or it can install any Linux distro where all the source code is available), and the team behind the phone, Purism, states that the Librem 5 phone will be the world's first ever IP-native mobile handset, using end-to-end encrypted decentralised communication.

The phone has a five-inch screen, camera, microphone, Wi-Fi and Bluetooth (plus hardware killswitches for instantly disabling them) and works with 2G/3G/4G, GSM, UMTS, and LTE networks. Pledging $599 will get you a handset, with an estimated delivery of January 2019. Some people on non-Google devices may even have received the Android O update by then.



The Librem 5 is a phone that promises to be a fully open alternative to Android devices.

FOSS GAME ENGINE

Godot 3.0 is here

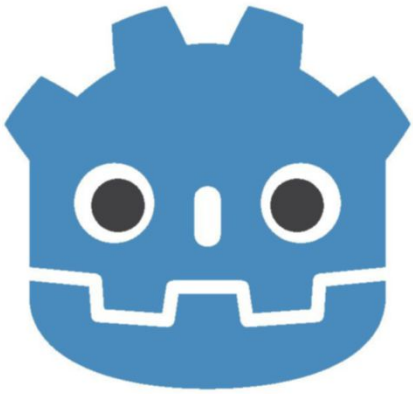
Vladimir and Estragon can finally get their 3D rendering done.

A new major release of Godot, a 2D and 3D cross-platform open source game engine that enables developers to create games for PC, consoles, mobile and web platforms, is nearing release.

A YouTube video, which can be viewed at **www.youtube.com/watch?v=XptIVErsL-o**, shows off some of the new features that will be included in this release, such as a brand new 3D renderer that features physics-based rendering, real-time global illumination and improved post-processing effects. Even with YouTube compression, the video does a great job of showing off how good these new features can make games created with Godot 3.0 look.

A new high-level network multiplayer API has been included to make creating online multiplayer games easier, plus a new audio engine built from the ground up should make games even more immersive. Support for C++ (GDNative), C# (Mono) and Visual Scripting means that Godot is an even more flexible alternative to Unity and Unreal Engine. It's also been announced (**http://bit.ly/2eNC6tg**) that Godot now supports glTF 2.0, a JSON-based and royalty-free format for transmitting and loading 3D scenes and models, created by the Khronos group, which is also behind the popular Vulkan API.

If you're interested in checking out the benefits of Godot over other game engines, there's a fantastic blog post (**http://bit.ly/2wKGTjj**) by a game developer that outlines the benefits of Godot. If you fancy giving it a try, then the alpha build of Godot 3.0 can be downloaded from **https://godotengine.org/download**. And if you'd like to help the team financially, there is a Patreon campaign (**www.patreon.com/godotengine**) fund-raising page that enables you to help Juan Linietsky, who wrote most of the code himself, to work on the project full time.



Godot, the open source game engine, is getting a new version with plenty of exciting features.

AD(UN)BLOCKING

Adblocking under attack

A domain is using DMCA takedown requests to beat ad filters.

While we always want to support our favourite websites, it's often essential to run an ad-blocker in your browser for a safer and more enjoyable online experience. Many websites don't want to appear on any filter lists, but one domain is sneakily removing itself from those lists – something that could pose serious problems for adblocking in the future.

It emerged that a commit had been added by a Github account to the popular EasyList filter list, that removed the **functionalclam.com** domain – which belongs to an ad server – with the comment that it was "Removed due to DMCA takedown request". The original DMCA (Digital Millennium Copyright Act) notice can be read at **http://bit.ly/2vPtdq5**. It threatens a DMCA takedown unless **functionalclam.com** is removed from the list and "not replace[d] with alternative circumvention attempts".

The domain is connected to Admiral (**https://getadmiral.com**), an anti-adblocking company. Admiral defended its move (**http://bit.ly/2wAt0Eb**), arguing that **functionalclam.com** is not an adserver, and that it followed GitHub's Guide to Submitting DMCA Takedown Notice for "Code [that]... is used to circumvent access controls."

Despite this, many people feel that a company has misused the threat of a DMCA takedown to remove a domain from an adblocking list. Comments by the filter maintainers on GitHub note that, "We had no option but to remove the filter without putting the Easylist repo in jeopardy."

If it worked for one company, what's to stop other adservers getting removed? As a comment on GitHub points out, could malware sites issue DMCA letters to anti-virus and security companies? That may be a little hysterical at this point, but we'll be keeping a close eye on how this unfolds...

Newsbytes

SUSE has reaffirmed its support for the Btrfs file system (based on the copy-on-write principle), of which it's the biggest contributor, with it remaining the default filesystem for SUSE Linux Enterprise. As a blog post (**www.suse.com/communities/blog/butter-bei-die-fische**) written by Matthias Eckermann, director product management of SUSE Linux Enterprise, explains, SUSE's investment in Btrfs has produced a number of features and innovations that are now essential to the distro – and of course, moving to a new filesystem for enterprise users would be a huge undertaking.

Google has announced the launch of Chrome Enterprise, a version of Chrome OS designed for enterprise users. With Chromebooks becoming so popular among business and academic users, we feel this is a smart move. Chrome Enterprise, which costs $50 a year per device, comes with a range of extra features, including an enterprise app store, advanced security controls, 24/7 support and integration with cloud management tools. For more information, check out the blog post at **www.blog.google/topics/connected-workspaces/introducing-chrome-enterprise**.

Devices such as the Asus Chromebook Flip are popular with businesses, which is why Chrome Enterprise is so welcome.

Imagine if Linus Torvalds had been headhunted by Apple and stopped working on Linux. In that alternate reality you'd be probably holding a copy of *Mac Format* in your hands right now, but according to an interview with *Wired*, Linus was offered a job at Apple by Steve Jobs in 2000, to work on the Unix-based kernel of Mac OS. While it would have meant working for a bigger (at the time) user base, and probably resulted in a significantly larger monthly pay cheque, it would have also meant that Linus could no longer work on Linux, so he declined. We think he definitely made the right move!

Comment

Boost your OSS skill set

If you want a secure and rewarding career, open source software is the field to be in according to this year's *Open Source Jobs Report*, an annual survey of the industry conducted by The Linux Foundation and Dice. Not only did 89 per cent of hiring managers surveyed say that they have difficulty recruiting enough OS (open source) talent, but 86 per cent of professionals believe that knowing OS has advanced their career.

Both hiring managers and OS pros agree that the top skill right now is cloud. Forty seven per cent of hiring managers are looking to hire professionals with OS cloud expertise, and 69 per cent of professionals believe cloud skills will gain in importance in the next year. Big Data is also in demand, with 43 per cent of hiring managers seeking employees with this expertise, and 57 per cent of pros seeing it growing in importance. Software development, DevOps, security and containers are similarly seen as key skills now and in the future.

The next question is how to go about acquiring these in-demand skills. Fifty per cent of hiring managers say they would be more likely to hire a certified professional, and 76 per cent of professionals with certifications say they've helped their career. And the percentage of hiring managers willing to pay for employees to get certified has jumped from 33 per cent in 2016 to 47 per cent this year. This is good news for existing professionals who want to learn new skills, and also gives those looking to break into the industry a starting place to demonstrate their knowledge. Don't be shy about asking your company to sponsor your training!

To learn more about what benefits of an open source career professionals enjoy most, and what skills and experience hiring managers are looking for, the *2017 Open Source Jobs Report* is available for free download at **http://bit.ly/2017OSSjobsreport**.

› Clyde Seepersad is general manager of training and certification at The Linux Foundation.

Distro watch

What's behind the free software sofa?

KAOS 2017.09

The latest snapshot for the Linux distro that focuses on providing sleek and user-friendly KDE and Qt-based software, is now available to download. KaOS 2017.09 comes with a hardened Linux kernel with address space layout randomization (ASLR) and *Nomad*, a Qt5-based firewall application, as well as the *Kooka* scanning program.

To find out more, visit the KaOS 2017.09 release announcement over at **https://kaosx.us/news/2017/kaos09**.

› KaOS is a stylish distro build for KDE and Qt-based apps.

REACTOS 0.4.6

If you want an open source operating system alternative to Windows, offering binary compatibility, then check out ReactOS. The latest version offers, as the release announcement states, "A major step towards real hardware support." This is great news for anyone who's been looking to make the switch from Windows to ReactOS. Other updates include dual-booting issues being resolved, safer partition management and other critical bug fixes. For a full rundown of changes, see the release announcement at **www.reactos.org/project-news/reactos-046-released**.

› ReactOS is a binary compatible alternative to Microsoft Windows.

LINUX LITE 3.6

Looking around for a beginner-friendly distro? Then consider Linux Lite, which has a new version out with some major changes. A new repository selector, Lite Sources, enables users to easily select a software repository nearest to their location, which will help improve download speeds. An online and offline search engine for the Linux Lite Help Manual is also included, which is an excellent addition for this easy-to-use distro to help people find answers to their questions quickly and easily. Go to **www.linuxliteos.com/forums/release-announcements/linux-lite-3-6-final-released** to learn more about this release.

› This friendly distro contains a decent help manual, for once.

BLACKARCH LINUX 2017.08.30

The Arch-based distro, BlackArch Linux, is a distribution designed for penetration testers and security researchers, and a new snapshot has been released. It includes the Linux kernel 4.12.8, as well as updates to a host of tools, system packages and menus, which will make this version easier to use than ever.

For more information on what's new, and to download the latest version, head over to **https://blackarch.org/blog.html**.

A suitably moody logo for the distro built for penetration testers.

BODHI LINUX 4.3.1

A new version of the lightweight Ubuntu-based disto that features the Moksha desktop has a new update out. The latest release includes a number of updated applications, including *EFL 1.19.1*, *Terminology 1.1.0*, *Ephoto 1.5* and Linux kernel 4.11. As with previous versions of Bodhi Linux in the 4 series, it's been built on the stable foundation of Ubuntu 16.04.

Find out more at **www.bodhilinux.com/2017/08/29/bodhi-linux-4-3-0-released**.

There was a bug in the Bodhi Linux 4.3.0 installation media, so the latest version you need is 4.3.1.

REDCORE 1708

A fresh snapshot of Redcore, a rolling release distro based on Gentoo, is now available. The latest version brings with it a number of new package upgrades and bug fixes. According to the release announcement (which can be read at **https://groups.google.com/forum/?hl=en#%21topic/redcorelinux/EqA-r822xpI**), "This release focuses on polishing the overall look'n'feel and out of the box experience of the distribution." Redcore 1708 includes a resync with Gentoo stable portage tree (27.08.2017) and features Linux kernel LTS 4.9.40, with the 4.12.x kernel available in the repository, if you'd rather have a newer version.

The Redcore 1708 snapshot brings a host of updates, so make sure you check out the release announcement for the full list.

Command substitutions

Keith Edmunds

Command substitution enables a command to include the output of another command. For example, we want the symbol `THIS_HOST` to hold the host name of the system, using the `hostname` command. There are two straightforward ways of assigning this to a symbol. The older, deprecated way is to enclose the command in backticks:

```
$ THIS_HOST=`hostname`
```

This is still in common use, but it has a number of disadvantages: it's too easy to misinterpret and nesting substitutions is very challenging.

The preferred modern syntax is to wrap the substituted command in a `$(...)` construction. Our host name example now becomes:

```
$ THIS_HOST=$(hostname)
```

Nesting is now simple. Take assigning the time since a file was last modified, in seconds to the symbol's age – let's use FILE to hold the path of the file in question. The last modified time in seconds since the Epoch is like this:

```
$ FILE=/tmp/mytestfile
$ touch $FILE
$ stat -c %Y ${FILE}
1499673894
```

Find the current time since the Epoch with:

```
$ date +%s
1499673946
```

For the age of the file in seconds:

```
$ age=$( ( $(date +%s) - $(stat -c %Y ${FILE}) ) )
$ echo $age
52
```

The two time-since-epoch commands are each wrapped in a `$(...)` so the above becomes:

```
$ age=$( (1499673946 - 1499673894) )
```

Bash does the calculations within a `$((...))` and yields the time since the last modification. Yes, it's confusing: single parentheses for command substitution, and double for expression evaluation. Backticks still work, but I'd suggest it's worth using the newer form in the future.

Keith Edmunds is the managing director at Tiger Computing Ltd, **https://tiger-computing.co.uk**.

United Linux!

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

Find and join a LUG

Alpinux, le LUG de Savoie
Meet on the first and third Thursday of the month at the Maison des Associations de Chambéry.
www.alpinux.org

Build Brighton Open night Thursday evenings.
www.buildbrighton.com

Sandbox Sandbox Digital 5 Brasenose Road, Liverpool L20 8HL open maker night is Tuesday 6pm-9pm; kids clubs are Monday (six to eight years) and Wednesday (eight to 12 years).
www.sandboxdigital.co.uk

Leeds Hackspace Open night every Tuesday, 7pm. Open day 2nd Saturday of the month, 11am-4pm.
www.leedshackspace.org.uk

Hull Raspberry Jam
Malet Lambert School, Hull. Every other month.
www.twitter.com/hullraspjam

rLab Reading Hackspace Unit C1, Weldale S, Reading. Open sessions Wednesday from 7pm.
www.rlab.org.uk

Huddersfield Raspberry Jam
Meets every month at Huddersfield Library, typically the fourth Saturday of each month.
www.huddersfieldraspberryjam.co.uk

Medway Makers
12 Dunlin Drive, St Mary's Island, Chatham ME2 3JE.
www.medwaymakers.com

Cornwall Tech Jam Second Saturday of the month, alternating between Bodmin and Camborne.
www.cornwalltechjam.uk

A weekend of choices

Take part in events up and down the country

The last weekend of October 2017 is one of many choices. Across the UK there will be three main events. First off, PyConUK (**http://2017.pyconuk.org**) returns to Cardiff for four days of Python-related content. This year we see the return of the Raspberry Jam, offering families the chance to get hands-on with Python and the Raspberry Pi, and work with Python developers, but new for 2017 we see the Raspberry Pi Foundation running its Picademy training course. Picademy at PyConUK is a great idea and we'll sure that it'll help teachers make contacts with Pythonistas!

The second event that weekend is Mozilla's Festival, known as MozFest (**www.mozillafestival.org**). MozFest is three days of talks, workshops and demonstrations that centre around the open web. **LXF** was there last year and it was great to see an entire section of the event dedicated, and run by children. These children curated their own content and ran a great programme of events that covered Raspberry Pi, virtual reality and radio broadcasts. Expect more this year!

The third and final event is Craft Council Make Shift Do event (**www.craftscouncil.org.uk/what-we-do/makeshiftdo**), which is taking place in Maker/Hack spaces around the UK. This two-day event offers attendees the chance to show what their space offers via a series of projects, which can be funded via the Craft Council. This is an important event to help bring new members to your space, something that will bring fresh ideas, views and much-needed funding.

It's going to be a busy weekend no matter where you are! Enjoy and don't forget to tell us about your events! LXF

MozFest was great in 2016! Lots to see and do – if only we could have spent more time there.

Community events news

Hack Manchester
The 24-hour party people of Manchester are back for another massive hack! Hack Manchester is, as you've no doubt guessed, a 24-hour hack session in which teams work to solve a challenge or problem set by the hosts. The event takes place in Manchester's Museum of Science and Industry on 28-29 October, and is sure to generate great ideas from the many coders and hackers around the UK. Who knows, perhaps Hack Manchester will solve a few of the real-world problems that we all face? You can find out more and secure your tickets by visiting **www.hac100.com/event/hack-mcr-17**.

#gashack
Here's something new and a little different. Anesthetists aren't something that we'd normally cover in **LXF**, but when they want to hack technology to help provide better services to patients and staff, we're happy to help. The event follows the recent NHS hack days that have been cropping up around the UK. Delegates pitch an idea and work as a team to realise the idea, culminating in a demonstration to a series of judges who'll award prizes to the winners. #gashack takes place on 21-22 October at The Royal College of Anesthetists, London, and you can sign up by heading over to the website: **http://gashack.rcoa.it/index.html**.

Mailserver

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

Old converters

As a member of an occupational pensioners' group, I receive a monthly magazine. It contains a regular Q&A column, where readers can ask for IT advice. But guess what? All the readers' problems are with various Microsoft products – nothing at all about Linux.

So I sent an email to the column writer, asking him why this was the case, and suggesting that he tell readers to ditch Microsoft and install a Linux distro instead. I finished my email by telling him that I experienced IT bliss (no, not the dreaded XP wallpaper) after I installed Ubuntu a couple of years ago. I wonder if my contribution will appear in print?

Pete Barrett, Northumberland

Neil says: Let's face it, the reason why everyone is writing in with Windows problems is that it's the only option you have when you buy a new PC. With Microsoft putting more and more adverts into Windows 10 and, it seems, locking out traditional .EXEs, too, I suspect we'll be seeing more people trying Linux out down the line.

Insecure verdict

I do very much disagree with your assessment and verdict on Linux Kodachi. First and foremost, there is one redditor who has pretty much tried to debunk some of its privacy claims, though maybe with some minor mistakes in the criticism.

The verdict of it to be in first place is a grave mistake on your part. Please do read the valid criticisms pointed out by other redditors, and I'll let you judge what they say for yourselves: http://bit.ly/LXF229reddit.

You also mentioned that Kodachi "removes traces of its use from the computer during shutdown". So does Tails! You have a responsibility in this regard – trying to talk to your audiences about privacy-oriented distros is praiseworthy, but this Kodachi should not be promoted at all.

I would also like to comment on your statement concerning Qubes OS: "While it is an interesting Linux distribution, it's more geared towards security-conscious users, rather than privacy advocates..." It's a bit odd to say that. Despite being a security-oriented OS, it is the most recommended OS in the subreddit r/Privacy when it comes to privacy.

It would be great if you added a brief explanation of different threat models when talking about privacy distros.

Zero, via email

Mayank says: It would be very irresponsible of us to disregard a project just because its developer isn't popular on Reddit. Have you heard Linus Torvalds vent? Richard Stallman? Miguel De Icaza? Nat Friedman? Just because someone is criticised on a public forum or refuses to engage with them doesn't necessarily reflect poorly on the software.

Privacy is a simply enormous topic. The *Roundup* **LXF222** was about the distributions themselves, and not a privacy primer, and it would have been the wrong place to explain all the nitty-gritty of the individual pieces of software or the different "threat models". I'll agree that I could have worded the DNS encryption statement better, but Kodachi does use VPN along with DNSCrypt. And, once again, just because */r/*Privacy recommends Qubes for privacy doesn't mean that everybody else should, too. Qubes is designed (and described by itself as "a security-oriented operating system") for security, which it delivers via isolated virtual machines. If you are concerned about privacy, the Qubes developers themselves recommend using Whonix alongside Qubes.

It's true! Everyone says Qubes is a reasonably secure OS.

Distro overload

Take a look at Distrowatch and you'll understand why Linux is a no-go zone for new computer users. How many distros are there now? Around 300 and counting. And every distribution has its own website. This is simply ridiculous.

The seasoned Linux user will tell a newcomer there's a version for every imaginable need. There are distros for general requirements, for servers, mathematics, art and creativity, forensic investigations, encryption, ultra-modern and ancient computers, embedded applications, science and so on.

Why are there so many replications of Linux? Who needs them? It's tempting to suggest that lots of them are the products of ego trips. Programmers take one of the foundation distros and tweak it into a 'new' edition, which differs only minutely from dozens or scores of others. Who would lament their loss?

Maurice George, Ormskirk

Jonni says: There are, indeed, a lot of distros – and, yes, a user

Helpdex

shane_collinge@yahoo.com

Letter of the month

Women writers

I greatly enjoyed reading Joanna and Nancy Mellor's letter concerning the lack of female contributors. As a regular contributor to the magazine, I'd like to share my thoughts.

As Neil rightly points out, articles are mostly submitted by male freelancers who pitch ideas to the editor. There is no deliberate exclusion of women and, indeed, several of the Linux-related magazines I write for have female editors. This is particularly true (and surprising) in my specialist field of cybersecurity as, in theory, there's nothing to stop a woman teaching herself to code privacy-related programs. She needn't even reveal her gender if she wishes, yet the field is almost entirely male. This said, the most secure OS I have ever seen to date is Qubes, the brainchild of Joanna Rutkowska.

At a time when women undertaking journalism courses at British universities outnumber men, and over 70% of freelance writers are female, clearly there's no shortage of women with the potential to contribute to Linux-related magazines. If there are any female readers who are new to Linux and would like to write about this awesome OS, I would be delighted to provide some pointers. Feel free to contact me in confidence at natewillhelp@gmx.com.

Nate Drake, Ireland

Neil says: With the recent Google Manifesto debacle as an example, there's still plenty of ridiculous nonsense around to put women off entering the profession. If there are any female readers out there wanting to contribute, please get in contact.

not familiar with the Linux landscape will be understandably daunted by their abundance. However, as is possibly not said enough in our mag, there are probably fewer than a dozen major distros worth considering as a beginner.

People and companies make new distributions for any number of reasons, and the nature of open source (it's a bazaar and not a cathedral) means that no one has any authority to tell people how to use their free time or to tell companies what and what not to develop – they should be free to do that without criticism.

It would certainly be simpler if there were one desktop distro, like there is one version of Windows now and one desktop version of Mac OS, but it would stifle innovation and variety, and ultimately be more boring. It's all about choice, and if you want a monoculture or a walled garden, then the proprietary OSes are a better fit.

Even Ubuntu lists nine official different releases of just Ubuntu.

Low-energy, fab!

I am trying to find a low-energy solution for a good performance machine I can use for my weather satellite monitoring and my weather stations. My machines run 24/7, so I am trying to reduce the number of watts I use, and hence my electricity bill.

I have been a Gentoo enthusiast since probably 1999, and a regular subscriber to *Linux Format* for many years. My present machine runs CentOS 7, with Intel Core 2 Quad CPU Q6600 2.40GHz, it has 16GB of memory and two NICS bonded, because I need plenty of Ethernet speed for moving the weather satellite images around. It has two 500GB drives with RAID 1 and one 1TB without. The VGA is Nvidia Corporation GT200 GeForce GTX 260 dual-head for two monitors. Yup, lots of real estate, so quite a bit of power needed. Is there any advice you can give me?

Francis, West Cork

Neil says: Yikes! That old Q6600 in itself will consume lots of power – its TDP is 105W. A modern desktop processor will likely halve this (taking into account lower-power DDR4 and motherboard chipset) and provide triple the CPU power, combined with integrated graphics able to run three displays and a motherboard with dual NICs.

The six-core AMD Ryzen 5 1600 is a good all-round choice – but you will need a graphics card on top – to build from scratch. On the Intel side, for similar speed, you'd want a Core i7 7700K, but it's £100 more just for the chip, though it does have integrated graphics. Do let us know how you got on! LXF

Write to us

Do you have a burning Linux-related issue you want to discuss? Need to tell us what we've done wrong this issue? Perhaps, what we have done correctly? Or just topics we should be covering? Write to us at *Linux Format*, Future Publishing, Quay House, The Ambury, Bath, BA1 1UA or **lxf.letters@futurenet.com**.

DISCOVER THE ALTERNATIVE TO APPLE AND WINDOWS

Take complete control of your computer by learning how to choose, install and use a version of Linux that's tailored just for you

ON SALE NOW

Future

Ordering is easy. Go online at:

www.myfavouritemagazines.co.uk

Or get it from selected supermarkets & newsagents

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

AMD Ryzen 3 1300X

There's a new budget king in town and he's wearing a crown made by AMD, **Kevin Lee** gives praise where it's due.

In brief...

- **Socket:** AM4
- **Type:** 64-bit
- **Process:** 14nm FinFET
- **Cores:** four
- **Threads:** four
- **Clock:** 3.5GHz
- **Turbo:** 3.7GHz
- **Cache:** L1 384kb, L2 2MB, L3 8MB
- **Mem:** DDR4-2667, two channels
- **TDP:** 65W
- **Virtual:** AMD-V, AMD-Vi

AMD's family of Ryzen processors has made a name for itself with high core and thread counts, but its most appealing facet has been always affordability. With the introduction of Ryzen 3, AMD's newest processors finally dip below the £100 mark.

The Ryzen 3 1300X sits at the top of this new range of processor with four cores, and it doesn't break the bank with its £120 price. It's a speedy chip, too, with a base frequency of 3.5GHz that cranks up to 3.7GHz. Typically, chips at this price point offer fewer cores or lower frequencies, but Ryzen 3 1300X stands up for users on a budget with strong performance.

For a fast quad-core processor the Ryzen 3 1300X makes an exceedingly good deal. Especially when you consider Intel's top-of-the-line Core i3 chip, the 7350K, is only dual-core and cost £150. That said, Intel has a leg up in frequency, with its part operating at up to 4.2GHz without overclocking. Users on a tighter budget can pick up the AMD Ryzen 3 1200 for just under £100. It offers the same quad-core capabilities as the Ryzen 3 1300X, even if it operates a tick slower at 3.1GHz to 3.4GHz. Meanwhile, if you want to pick up a processor with at least four cores from Intel's camp, you'll have to spend £155 on the Core i5-7400.

Features and chipset

Like AMD's Ryzen 5 platform, Ryzen 3 comes built on a 14nm FinFET architecture and optimised for its current AM4 platform. Users going for this budget CPU would likely go for a budget-friendly B350 chipset motherboard (starting at around £75) that supports six lanes of PCIe Gen 2 for solid-state drives (SSDs), two USB 3.1 Gen2, but lacking support for multiple graphics cards. More price-conscious builders can also opt for the even lower-priced (starting at around £50) A320 chipset, but this choice comes with some trade-offs. Namely, overclocking is locked off, and you'll only have one USB port with a throughput of 10Gbps and two fewer PCIe lanes for SSDs.

AMD's CPU is a bargain, however you look at it.

The Ryzen 3 1300X performs exactly as we expected it would up against the Intel Core i3-7350K. AMD's processor has double the cores over Intel's chip, but each die isn't quite as quick because of the lower clocks. Benchmarks show the AMD lagging on single-core loads but beating the 7350K on multi-threaded tests. Gaming performance goes to the Ryzen. It does lag – only just, mind – with some games like *Total War: Warhammer*, but soundly leads with other such as *Rise of the Tomb Raider*. Considering the lack of any optimisation for the Ryzen, and we can only see the gaming gap improving over time.

Where the extra cores make the real difference is media creation, especially for rendering objects and encoding video. In this case, the Ryzen 3 1300X blows away the Intel Core i3-7350K in Handbrake, with a 50 per cent faster encoding speed.

The Ryzen 3 1300X is clearly the best-performing processor on paper and it's an unquestionably great deal. For £30 less than the Intel Core i3-7350K, you're getting a processor that outperforms it on every important metric, offers two more real cores, bigger L3 cache and a thorough thrashing at media encoding and rendering on an affordable platform, too. LXF

LINUX FORMAT Verdict

AMD Ryzen 3 1300X

Developer: AMD
Web: www.amd.com/ryzen
Price: £120

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

AMD's brilliant quad-core processor is cheaper and faster than its main Intel competitor. What's not to like?

Rating 9/10

AMD ThreadRipper 1950X

Sometimes they let **Jonni Bidwell** play with fancy hardware. But the tantrums that occur when he has to return it are oh so tiresome…

Specs

- **Socket:** TR4
- **Clock:** 3.4GHz (unlocked)
- **Turbo:** 4.0GHz
- **Cores:** 16
- **Threads:** 32
- **Process:** 14nm FinFET
- **Cache:** L1 1.5MB, L2 8MB, L3 32MB
- **Memory:** 128GB DDR4, four channels, ECC support
- **PCIe:** 64 lanes
- **TDP:** 180W

We were impressed with AMD's erstwhile flagship processor, the Ryzen 1800X (see **LXF223**). Now AMD's back with a new beast, the 16-core, 32-thread ThreadRipper 1950X. Its predecessor gave Intel's i7 professional class 6950X (which costs twice as much) a run for its money in many tests, particularly multi-threaded workloads. For gamers though, a more suitable comparison was found in Intel's quad core 7700K, which cost less and performed significantly better at single-threaded workloads than the Ryzen. This time round it's a similar story: there are expensive Intel chips for it to compete against, but there are also cheaper, fewer-cored offerings from Intel if all you care about is gaming.

AMD's innovative Infinity Fabric enables its silicon to scale naturally. ThreadRipper can be thought of as two Ryzen 7s stuck together in a single package, wrapped in an attractive industrial orange bracket. Those Ryzen 7s can in turn be thought of as two four-core complexes (CCXes) Infinity Fabric'ed (*pretty sure that's not a verb – Ed*) together. Going the other way and sticking up to eight CCXes together, you get AMD's enterprise level Epyc CPUs, aimed at datacentres. Such magic glue (*you'll never make a marketing person – Ed*) enables 64 cores to fit on a dual socket motherboard. That's textbook journalistic oversimplification, of course, and there are subtleties between the mainstream (Ryzen 5 for example), ThreadRipper and Epyc components that we won't go into. What's worth making clear is that ThreadRipper can handle up to 64 PCIe lanes, as opposed to the paltry 24 (of which only 20 are usable) of the Ryzen 7. This makes it much more tempting for those looking to build multi-GPU setups or house lots of high speed SSDs.

Benchmark results

Benchmark	ThreadRipper 1950X	Ryzen 7 1800X
FFTW 1,024bit (Mflops)	22,573	21,004
John the Ripper – Blowfish (cracks/s)	12,096	12,996
GraphicsMagick – resize (it/min)	234	242
TTSIOD (fps)	429.75	315.29
C-Ray (s)	5.11	8.10
Kernel compilation (s)	37.33	77.37
FFmpeg (s)	7.62	13.47
LAME (s)	8.80	9.01
FLAC (s)	5.45	5.24
OpenSSL – 4,096-bit (signs/s)	2,196	1,149.23
Blender (s)	521.5	566.19

It's a big 'un

Size isn't everything, but ThreadRipper's dimensions are something to behold. It measures 72x55mm, which make it not that much smaller than a Raspberry Pi B+. It also has 4,094 pins and so doesn't fit in an AM4 socket: this one calls for the new TR4 socket. Installation is reasonably hardcore, there are three torx screws that clamp down a retention bracket. Once this is released a tray can be lifted up and then two covers removed. Then the great orange thinking unit can be slotted in place, and everything battened back down. The ThreadRipper chips have a maximum power draw of 180W, so things are going to get hot (AMD recommends liquid cooling).

Looking at the base clocks, the 1950X's 3.4GHz versus the 1800X's 3.6GHz, it's reasonable to expect that the 1950X won't improve on the single-threaded results. Both have cores that boost to 4.0GHz (although this is handled differently on ThreadRipper) so we'd expect performance to be similar. With 32 threads, though, we would expect to see some pretty awesome results for parallelisable workloads.

Our test bench featured the eminently covetable Asus ROG Extreme Zenith motherboard armed with 32GB of 2,400MHz DDR4 RAM, a nippy 512GB Samsung 960 Pro SSD and a GTX 1080. We found Fedora 26 booted okay on this, but working off a live USB was incredibly slow. Once installed and updated (to Kernel 4.12.5) we encountered no problems, not even with Nouveau. We put the chip through its paces with a selection of *Phoronix Test Suite* benchmarks.

As predicted, the single-threaded tests (*FFTW*, *GraphicsMagick*, *Lame*, *FLAC* and *Blender*) are very similar to Ryzen, which in turn was bested by Intel's 7700K. Comparing these to results that are available on **www.openbenchmarking.org**, we see that the i9-7900X does even better here, particularly at multimedia workloads. It's then reasonable to conjecture that the 7900X will be a better chip for games, which tend to not scale beyond around four cores. That said, we didn't test AMD's Gaming Mode (that would have required booting into Windows), so things may not be so clear cut.

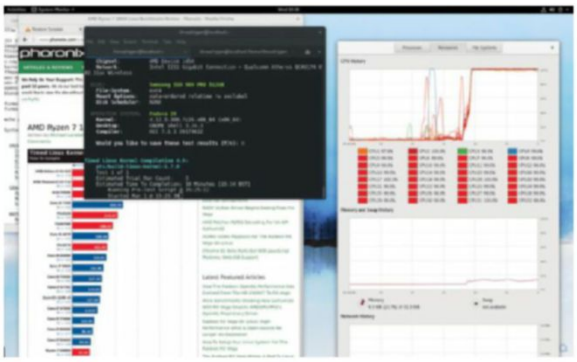
The multithreaded results are a mixed bag, but before we go into any analysis, it's worth remembering that this is a new CPU and any anomalies we noted may be addressed in upcoming firmware or kernel updates. There could be issues within the benchmarks too, the *FFmpeg* test for example uses `-threads` argument but actually passes in the number of cores. This may not matter for a quad core chip, but with a 16-core monster the difference will be marked. They could also be the result of errors, oversights or misconfigurations on our part. Maybe we should've tried Game Mode. With that aside, let our wild speculations begin.

John The Ripper ought to be doing about twice as much, er, ripping on the ThreadRipper as we saw. It uses OpenMP for distributing workloads across cores, so perhaps something is awry in our toolchain. Rendering with *TTSIOD* was promising: we wouldn't expect a straight doubling in performance here, since this is a complex job that won't always parallelise nicely. Likewise *C-ray*, and since this test runs for such a short time it's reasonable to expect more complex ray tracing jobs to exhibit an even greater differential.

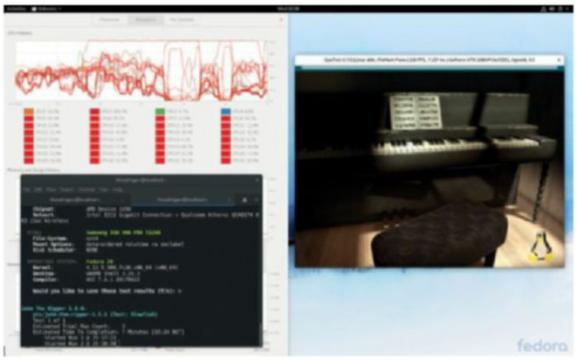
The tangled threads we weave

AMD is aware that not everyone needs 16 cores, and most games won't make effective use of more than eight. Some legacy titles even suffer when so many cores are presented to them. So ThreadRipper offers two modes of operation: Creator (the default) and Game Mode. In gaming mode half the cores/threads are disabled (PCIe lanes et al remain open for business) and the CPU is forced into non-uniform memory architecture (NUMA) mode, which means it favours memory physically closer to the die requesting it, reducing latency.

The Zen microarchitecture supports up to eight channels of memory, although the previous Ryzens only enabled dual-channel configurations. With ThreadRipper this is boosted to quad-channel, and the potential latency arises because each die is only directly connected to two channels. Note that, contrary to what you might read elsewhere, Game Mode doesn't boost clock speeds anyway: according to AMD, it boosts gaming fps by an average of four per cent. This isn't earth shattering, but ThreadRipper was never going to shine in the gaming arena, or at least would shine brighter elsewhere. Game Mode is just a Band Aid until gaming workloads learn to scale better, and is unlikely to be something Linux users are interested in, especially as it can only be enabled from AMD's Windows-only *Ryzen Master* software.



This is what compiling a kernel in under 40s looks like.



There's something funky about our John the Ripper results. Running it in tandem with GPUTest failed to improve them.

Compiling kernels

We were really impressed with the kernel compilation results, although the speedy SSD probably helped a lot here. This is a good real-world benchmark that tests raw processing power, I/O and concurrency. The kernel (4.9 in this case) is so sprawling and GCC so clever that it rarely has to wait for one compile job to finish before others can start, so it achieves awesome concurrency. We remember when compiling kernel's took several cups of tea. The *FFmpeg* test decodes an H.264-encoded HD stream to NTSC DV, and was rather disappointing with Ryzen. This time around we see an improvement in line with what we'd expect from twice as many cores, but this test seems to do better on Intel silicon. ThreadRipper excels at the *OpenSSL* test (4,096-bit RSA signatures), doing a little better than results reported for the 7900X.

Intel's soon-to-appear 12-, 14-, 16- and 18-core monsters will see the blue team better compete with Threadripper's multithreaded amazingness (and continue to trounce it on single-threaded workloads). But with the 10-core 7900X costing the same as the 1950X (£900, $999), its bigger siblings won't be competing on price. Intel's top-of-the-range i9-7980XE will be on sale by the time you read this for $1,999 (probably Brexit priced at £1,800). That's twice the price for two more cores.

ThreadRipper is a mighty powerful chip, but we can't help but wonder how big this mythical high-end desktop market segment is. It's great to see these numbers this side of £1,000, and there's a certain class of people that will be sold on numbers alone. There are also people that spend days compiling/rendering/transcoding – but these aren't mainstream users. So maybe this potent chip is solving a problem that, for the many, doesn't (yet) exist. LXF

Linux Format Verdict

ThreadRipper 1950X

Developer: AMD
Web: www.amd.com/threadripper
Price: £900

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

A behemoth of processing power, but such potency will be wasted on the masses. Gamers should look elsewhere.

Rating 8/10

VolksPC S905X

Fancy a $90 desktop? **Will Meister** takes a spin on an ARM-powered TV box that doubles up as a capable – if a little barebones – Linux PC.

In brief...

US startup VolksPC wants to see smart TV boxes repurposed as cheap Linux desktops. We tried out a preinstalled package that pairs its customised Debian with an otherwise unremarkable smart TV box.

Specs

- **CPU:** Amlogic ARM Cortex A53 quad-core 1.5GHz
- **RAM:** DDR3 2GB
- **ROM:** Onboard eMMC Flash 16GB (2.4GB for Linux)
- **Comms:** 100Mbps LAN, 802.11b/g/n, Bluetooth 4.0
- **Ports:** 2x USB2.0, 1x HDMI 2.0
- **Dimensions:** 110x110x17mm

Fanless smart TV boxes have been with us for several years. Built to run Android on ARM-based systems-on-chips (SOCs), they combine low cost with excellent specs. However, implementation issues have made them hard work for the few Linux developers who've tried to engage.

US startup VolksPC wants to see the boxes repurposed as low-cost PCs. The company has built a lightweight Xwindows alternative called MicroXwin, which it uses in a customised Linux that runs on top of the Android kernel. VolksPC is selling the software as a Debian image for Odroid's C2 developer board, and preinstalled on an Amlogic S905X smart TV box.

The one-line pitch is: full Linux on a cheap ARM SOC. How could we resist?

The VolksPC's $90 box crams Debian Linux into an Android app.

A boxful of ARMs

The S905X is a 2016-model quad-core TV box with 2GB of RAM soldered in place, HDMI, USB and Ethernet ports, a microSD slot, and built-in Bluetooth and Wi-Fi. It feels solid and well-built, but it's still light and small enough to knock off the desk with your mug of tea.

The unit ships with an international adaptor (but no UK plug), an HDMI cable and a remote control. The latter's rather handy given the box's lack of a hardware on-off switch. But only two USB ports meant we needed a robust, externally powered USB hub.

We connected it to a Samsung Syncmaster HD TV. The VolksPC synced up at 60Hz, cycling in a few seconds through a startup screen to a blue Debian desktop with a customised version of the XFCE 'rodent' theme.

It took a few seconds longer for the mouse to start responding. That's an indication of the VolksPC's main quirk: you'll need to switch the device into Android to apply settings like Wi-Fi passwords and date-and-time. Switching is easy and instantaneous, but we found it confusing to deal with two different interfaces. If you can cope, you might prefer to get your Netflix or *Skype* in VolksPC's Android mode.

VolksPC's Linux implementation is built on Debian Jessie. It resembles Lubuntu in its utilitarian setup. The system arrives preconfigured with Desktop and Root accounts, and a handy quick-start guide – although we'd have liked more on printer installation.

VolksPC's complex relationship with Android made us apprehensive about installing our own software, but *Synaptic* fired up as usual and we soon had *The GIMP*, *Filezilla* and a bunch of familiar tools up and running. Performance was consistently good, with even notorious slow coach *Inkscape* loading in seconds.

Installing *CUPS* was a different matter. *Aptitude* problems and *Synaptic* crashes that persisted even after a restart had us checking back with the developers. It turned out that VolksPC operates within a tight 2.4GB subdivision of the larger Android filesystem, which we'd managed to fill without noticing. The lack of any obvious means of keeping track of this storage is our single largest criticism of the system. Yet reinstalling Debian turned out to be as simple as logging out, downloading a disk image to microSD card and launching the company's app in Android. We managed a full, clean install in minutes.

After a second, more discriminating session with *Synaptic* – and a better-informed *CUPS* install – we had a working system with more-than-acceptable performance and near 100 per cent access to familiar peripherals. The only device that wouldn't play was our scanner, which remained invisible to *Simple-Scan*... although *Xsane* would probably have got it eventually. LXF

Features at a glance

Loadsapixels
VolksPC's MicroXwin is more efficient than Xwindows. A low-spec box can drive an HD monitor.

Fanless
The Amlogic base system runs on less that 10W, so requires no cooling hardware in place.

LINUX FORMAT Verdict

VolksPC 905X

Developer: VolksPC
Web: www.volkspc.org
Price: $90 plus shipping

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

The S905X is a well-implemented solution for third-world IT access, and should win over first-world fans, too.

Rating 7/10

SharkLinux 4.10

Jonni Bidwell at last finds a distro that caters to his selachimorphaphile tendencies. But does it bite back? Time to get into hot water and find out...

In brief...

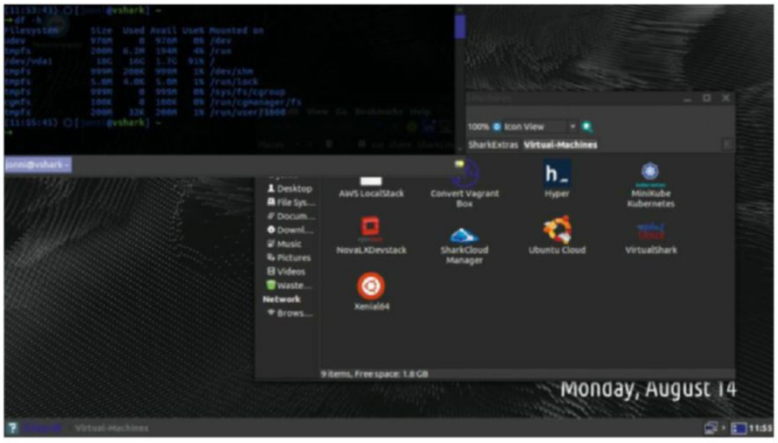
An Ubuntu 16.04 (jacked up with the latest HWE stack) distro with a difference. Actually several, but most obvious is its catering to users who want to run the desktop remotely, who want to run many flavours of VMs and containers, and who want easy access to upstream releases. See also: Debian, Fedora.

Shark Linux's motto, "Do Linux differently", isn't just a poke at Apple's famous "Think different". Scanning the website (*that's not how you review distros – Ed*) it's quick to point out its "100 per cent cloud compatible desktop" and "support for virtual environments of all shapes and sizes". Our interest was piqued.

Obviously, we've seen cloud-focused distros before, and we've played with remote desktops, but marrying the two together? That seemed wild and slightly crazy. Linux done differently indeed. It's hard work tricking a headless machine into sending graphics down the wire, so on the one hand it's great that this is all set up for us.

On the other hand, it's debatable how useful running a full-blown desktop remotely is, or if it's just a waste of precious resources. Shark Linux (no relation to SharkOS, the defunct Gentoo-based project) is certainly not just another Ubuntu-based distro.

On first boot (*bite? – Ed*) the user is greeted with a welcome screen that invites them to install or set up many things, including *DropBox*, *Thunderbird* or extra desktops. The base install is really a fairly threadbare Ubuntu 16.04 with a cloud compatible Mate 1.16 desktop and some slightly questionable theme-ing decisions. Through the extra desktops option, one can switch to the popular DeepIn desktop, or SharkLinux Edge that features Mate 1.18. Also on the welcome menu, the user is urged to install the SharkLinux Expansion. Without it, we're cautioned, one misses out on many of Shark's unique features.



We like Guake, the drop-down terminal, but this prompt spilling onto the next line upsets our sensitivities.

The expansion includes all the tooling and frameworkery to enable Shark to pull, build and install packages straight from their upstream sources. This provides the user with a single click path to software that would otherwise require some wrangling, including but not limited to *Grive* (the Google Drive client), *TeamViewer* (the proprietary remote access tool), and (shudder) Microsoft's *Powershell*.

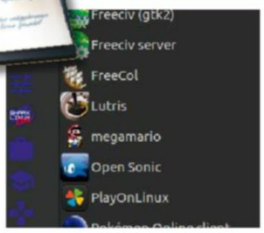
Bells and whistles

With the expansion pack enabled, a number of cloud and virtualisation tools also become available. Some of these seem terribly niche, such as the *DevStack* installer and a utility for setting up and administering *Vagrant* images. There are other tools for managing *Juju*, *Docker* and *LXD* images. Virtualisation gurus we are not, but we suspect many who are would rather set this up themselves, and not do so on a machine that's running a desktop environment with all sorts of other bells and whistles installed.

Besides the serious cloudy business, a number of games (*see left*) come with the expansion pack. This seems vaguely incongruous, but the inclusion of *Dope Wars*, the, er, *Business Simulator*, from the late nineties, made us chortle. A one-click install of the upstream *Wine* release is also available, for those wishing to do battle with Windows games or other software.

There's a lot to like about this release, the convenient passwordless `sudo` set up, the easy access to upstream software, the prefab shortcuts to common commands. But it all seems like a bit of an ad hoc collection, the entirety of which no one could possibly be interested in. Of course, the lone developer (Marcus 'rhymes with SharkOS' Petit) sees a use case here and we can't fault (and sincerely applaud) his efforts getting such an eclectic collection of tools in one place. LXF

Features at a glance



Games

The games selection, which includes *Open Sonic*, *Mega Mario* and *Freeciv*, betrays the developer's preference for (FOSS ports of) games of times past.

Kernel tool

A tool is provided that makes easy work of installing mainline kernel images, so new (and unsupported) features are just a click and a keypress away.

LINUX FORMAT Verdict

SharkLinux 4.10

Developer: Marcus Petit
Web: www.sharklinuxos.org
Licence: Mixed FOSS/proprietary

Features	8/10
Performance	8/10
Ease of use	6/10
Documentation	6/10

Certainly needs more documentation, perhaps needs to pare back and refocus the included software selection.

Rating 7/10

Netrunner Rolling

After a nice cup of tea and a few quality biscuits, **Jonni Bidwell** has finally got over his irrational fear of rolling release distros – and thinks you should, too.

In brief...

» A Manjaro-based distro with a highly polished and up-to-date Plasma 5 desktop (and named after a cyberpunk-themed card game). Excellent choice for people who want a fuss-free way to a fully loaded desktop, replete with the latest KDE offerings See also: Maui, KaOS, Antergos, KDE Neon, Netrunner.

Arch-based distros are becoming all the rage now. And it's easy to see why: it has repos bursting at the seams (*repositories don't have seams – Ed*) with the newest packages and it's highly customisable. It's also much more stable than people give it credit for, so once you throw in an installer (the popular Calamares in Netrunner's case) and a ready-to-roll desktop, then there's potential indeed for Linux greatness.

The Debian-based, fixed release edition Netrunner has been popular for a long time, but it's Manjaro-based, rolling release sibling hadn't seen any development since January 2016. Such was the length of this furlow that it was forked, giving rise to the Nurunner project. But Netrunner Rolling (NR) has recently been rebooted, and now Nurunner is no more. Such are the hatches and dispatches of Linux distros, (and may this serve as a warning to proponents of sensational spellings).

In January, Netrunner rebased its fixed release distro on Debian's testing branch. This assuaged users frustrated by old versions of software in Debian's stable repos (historically Netrunner was based on Ubuntu, but this was forked and became Maui Linux). This Manjaro-based rolling release will enable things to be even fresher, though not quite as fresh as Arch, since there's some paucity in getting packages from there to Manjaro to NR. At the time of writing Netrunner sports Kernel 4.9, whereas an updated NR gives you 4.11.12, and Arch has 4.12.6.

» **Marble, included in the install has some features you won't find in Google Earth. This is Behaim's Erdapfel (earth apple, literally). Something is missing.**

The vanilla Plasma setup that's on Arch is certainly inoffensive and intuitive, but we like to tweak it here and there for optimal satisfaction. Manjaro's KDE edition features some stylish themeing, but doesn't make any substantive changes beyond the Arch config. NR has opted for a striking dark theme, and the fullscreen application dashboard rather than the traditional menu.

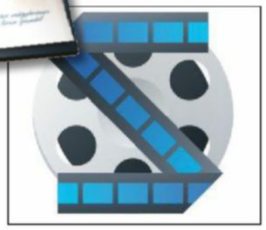
This is entirely reasonable, and will help Unity refugees feel a tiny bit more at home, but perhaps more could've been done to showcase Plasma's capabilities. Then again, maybe it's better to leave this fine tuning to your users – that is the Arch way, after all. A nice touch is the inclusion of the popular *KDE Connect* applet, so you can see phone notifications from the comfort of your desktop.

Programs galore

NR comes with lots of software that Arch users would have to invest effort to install and maintain: *Skype*, *Flash*, the *b43-fwcutter* package (for extracting firmware from Broadcom drivers), *Steam*. The latter adopts Arch's favoured approach of using a native runtime rather than *Steam's* bundled one (which is still based on Ubuntu 12.04). This may cause problems, but may equally well improve performance. Any *Steam* issues you encounter will likely have been encountered by gamers in the parent distributions' communities, so you shouldn't be stuck for too long.

Firefox is the default browser in NR, and it ships with some preloaded add-ons, namely the *Ant Video Downloader* and *uBlock Origin*. Plasma recently added support for icons on the desktop and NR comes with some (including the dreaded My Computer) to get you started. We can't condone this heathen practice, but each to their own. We did like that *Yakuake*, the drop-down terminal, was enabled by default, so any time you're feeling lost, F12 will bring you command-line solace. LXF

Features at a glance



Multimedia ready

SMPlayer, *Handbrake* and *KDEnlive* are all installed, as are *Gimp*, *Gwenview*, *Inkscape* and *Krita*.

Neon writer

The stylish disc/USB imaging utility from KDE Neon has been co-opted into Netrunner Rolling.

LINUX FORMAT Verdict

Netrunner Rolling 2017.07

Developer: The Netrunner team
Web: www.netrunner.com
Licence: Mixed FOSS/proprietary

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

» *Great for those who want Arch, but want to skip the hassle. Yet isn't the hassle all part of the fun? [NO! – Ed]*

Rating **8/10**

Antergos 17.8

A mightily relieved **Jonni Bidwell** discovers that Arch Linux and a convenient, out-of-the-box setup experience aren't mutually exclusive.

In brief...

An accessible, rolling release distribution based on Arch. A choice of desktops are available, and everything is ready to go post-install. Anyone intrigued but afraid of Arch should check this one. See also: Manjaro, Chakra, Netrunner Rolling.

Not one but two Arch-based distros this month. Arch purists may cry foul at their pure distro being corrupted with such fripperies as an installer (the awesome *Cnchi*) and a selection of desktop environments, it's not the Arch way, after all. But vanilla Arch is still there for those people, and Antergos, and a growing number of other distros, are there for those who would like to harness the power of Arch but would rather not spend three days setting it up.

Antergos is part of the second wave of Arch-based distros, first appearing as Cinnarch in 2012. Back then, Cinnamon was rather tricky to use anywhere other than its native Linux Mint, and it was especially difficult to make it play nice with the shiny new GTK libraries found in the Arch repos. So Cinnarch switched to Gnome rebranded itself Antergos (a Galician word meaning 'ancestors'), and the rest, as they say, is history.

Nowadays, Cinnamon is more portable and nicely packaged in the Arch repos and it, or Gnome (the default), KDE, Mate, Openbox or Xfce, can be chosen straight from the installer. A spartan 'Base' session, with no desktop, is also available, for those wanting something a little closer to the Arch experience.

Once you've chosen your desktop poison, *Cnchi* offers to set up Arch User Repository (AUR) support, Bluetooth, printers, *Flash* (*noooo – Ed*), LTS kernels and a few others. AUR support is a nice touch, because the process for bootstrapping an out-of-repo package manager, such as *Yaourt*, is convoluted and confusing to the uninitiated.

Cinnamon looks every bit as good on Antergos as it does on Mint, but we doubt that the world is ready for more wobbly windows.

Play time!

Steam and *PlayOnLinux* are bundled together in *Cnchi*, which is slightly confusing because they don't necessarily complement each other; games played via the latter may require the Windows version of the former. So users who are interested in a straightforward method of installing *Steam* are lumbered with a *Wine* installation they may never use. This aside, Antergos is a great choice for a gaming distro (**gamingonlinux.com**'s Liam Dawe says so), and as with NetRunner Rolling any gaming problems you encounter have probably been encountered and solved somewhere in the Arch ecosystem.

The partitioning utility enables LVM or ZFS to be set up in one click, in addition to offering an advanced option for those users who know what they want. There's also a handy checkbox to put **/home** on another partition. Whichever desktop environment you choose, you'll find it set up and looking delightful. There's a great selection of desktop wallpapers, and windows and icons use the bold and modern Numix Frost theme. Combined with all the shiny new packages you'll be the envy of your Ubuntu-using cohorts.

Despite everything being nicely presented and ready to go, Antergos is still very much Arch Linux under the hood, and uses Arch's repos under the hood. As soon as new packages hit the Arch repos, they're available in Antergos. There's a separate Antergos repository for its customised packages and high level additions. This includes some stuff prebuilt from the AUR too, such as Dropbox and the Widevine DRM plugin, so you can watch Netflix in *Chromium* without having to install *Chrome*.

Antergos' uses the *Pamac* frontend for managing packages, which is to Arch's *pacman* what *Synaptic* is to *Apt*. This makes all package-related business much less daunting, allowing one-click system upgrades so everything can be kept fresh. LXF

Features at a glance

Numix themes
These distinctive flat themes and icon sets give Antergos a slick and stylish look.

Just right
Antergos has everything you need to get started. Apps are chosen based on which desktop you use.

LINUX FORMAT Verdict

Antergos

Developer: Alexandre Filgueira & team
Web: www.antergos.com
Licence: Various

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

There may be no royal road to Arch, but this is certainly a pretty pathway that many people will want to take.

Rating 9/10

The Long Dark

Brutal, unforgiving, a survival of the fittest… these are just some of the words **Andy Kelly** uses to describe working at **Linux Format** Towers…

Specs…

Minimum

OS: SteamOS

CPU: Intel Core i5 dual-core 2GHz+

Mem: 4GB

GPU: Intel HD 4xxx, 512MB VRAM

A geomagnetic anomaly has plunged the world into darkness and rendered all technology useless, including the plane that you were flying over the vast, frozen wilds of Canada. You awake surrounded by flames and wreckage – injured and freezing to death – and find yourself in a battle to survive in one of the most inhospitable corners of the planet. It's a hell of a place to spend the apocalypse, and death lingers around every corner of this deadly, wintry expanse.

There are two distinct ways to play *The Long Dark*. There's Wintermute, an episodic story mode that follows bush pilot Will Mackenzie as he searches for his missing friend. Then there's Sandbox, which enables you to tell your own stories and explore at your leisure. The only objective here is surviving for as long as possible, and how you do that is left to you.

Wintermute is a good place to start. It begins with a series of tutorials designed to drip-feed the game's systems to you. Sometimes you'll meet survivors who need your help, forcing you to complete a series of thinly veiled fetch quests, which grind the story to a halt and feel a little too much like busywork at times.

But it's in Sandbox mode where *The Long Dark's* survival knife is sharpest. Having the freedom to explore and travel between its large, interconnected regions is more compelling than following a prescribed path. Choosing how you spend each day is more engaging than ticking off objectives. This freedom, along with the dynamic, unpredictable elements such as the weather, make every Sandbox game fertile ground for emergent storytelling.

Do spend some time admiring the beautiful art direction of The Long Dark. But not too long, else you may freeze to death by sundown.

Some of the most vivid memories of *The Long Dark* weren't created by the developers, but emerged naturally. Like the unbearable tension of being on the edge of starvation, one bullet in the rifle, and a skittish deer in our sights. Cowering in a cave at night, campfire about to burn out, listening to wolves howling outside. Limping half-dead and hypothermic through a blizzard, only to see the silhouette of a life-saving shelter through the wall of snow.

The weather is constantly in turmoil, which can change the mood of the game – and your fortunes – in an instant. One minute it's a crisp, clear day with piercing blue skies. The next a stormfront is rolling in, wind blowing the falling snow so hard it moves horizontally. Watercolour skies shift from a blanket of looming grey to the dusky pink of early evening, painting the snowfields around you in vivid colours. It's an incredibly atmospheric game, with a hand-painted art style that lends it a peculiar, ethereal beauty, despite how gruelling it is.

Like a lot of survival games, everything in *The Long Dark* boils down to managing a series of perpetually dwindling meters: hunger, thirst, tiredness and so on. But thanks to the elegant design of the simulation, and a slick, minimal UI, it's not a game where you feel like you spend half your time buried in menus. The abundance of progress bars is slightly disappointing, though. Many actions, such as breaking a branch down for firewood or cooking food, happen off-screen, illustrated by a slowly filling circle.

There are only a handful of really great survival games on PC, and this is one of them. The story mode has its moments, but it's when you're creating your own stories in the sandbox that *The Long Dark* is at its most absorbing. Beautiful art direction and rich, nuanced sound design bring the deep forests, frozen lakes and ragged mountains of the Canadian wilderness to vivid life. LXF

LINUX FORMAT Verdict

The Long Dark

Developer: Hinterland Studio
Web: www.thelongdark.com
Price: £27

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

Deep, brutal, and hauntingly atmospheric, The Long Dark is a survival game done right.

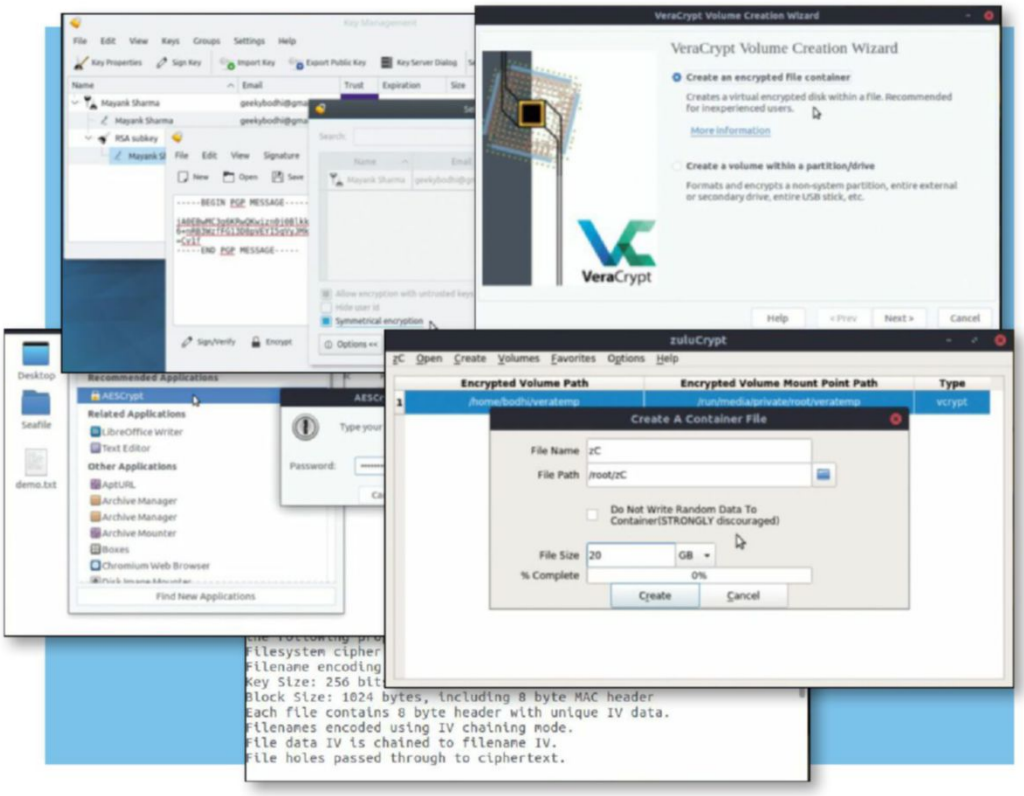
Rating 8/10

Roundup

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

Encryption tools

It's not paranoia but prudence that makes **Mayank Sharma** encrypt all his data. Twice. In a hermetically sealed room. While wearing a tin-foil hat.



How we tested...

All the tools were installed using their recommended installation mechanisms. Tools that were easier to install and use were rated higher. We also look at usability with respect to their feature set: a feature-rich command-line program may not be a better option than a simpler but intuitive graphical tool that gets the job done. We also looked at the help and documentation offered.

We didn't test the security provided by the tools, because they all use industry-standard ciphers to encrypt the data. We did, however, take note of the ones that enable their users to select the encryption cipher and its strength. We also rate them for their configurability options and tools that work across platforms are rated higher than those that cater to just one platform. The tools are all installed atop an Ubuntu Budgie 17.04 installation.

Our selection

- AES Crypt
- EncFS
- KGpg
- VeraCrypt
- zuluCrypt

There was a time when keeping your data under a username and password was considered ample protection. The computational overhead and the cumbersome programs relegated encryption into the realm of the paranoid. However, news of large-scale data snooping and overarching surveillance have reinvigorated interest in personal privacy. These days, basic file permissions and user accounts aren't enough to deter a determined intruder.

The only pragmatic approach to keep your personal data to yourself is to encrypt it. Working with encrypted data is an involved process, but it'll go a long way towards reinforcing your security and insulating your data from unwanted attention. There are a couple of strategies you can take for encryption. Most of the leading distributions now enable you to encrypt your entire disk while you're setting them up.

> **"Basic file permissions and user accounts won't deter intruders"**

Then there are programs that will help you create encrypted silos within your filesystem. The hallmark of these applications is that they can do on-the-fly encryption. This means they'll automatically encrypt your data before writing it to the disk and decrypt it when called for, assuming you have the right credentials. In this Roundup we'll look at some of the programs that assist with this kind of transparent encryption and can easily slot themselves into your daily interactions with any kind of data.

Ciphers

Do they conform to industry standards?

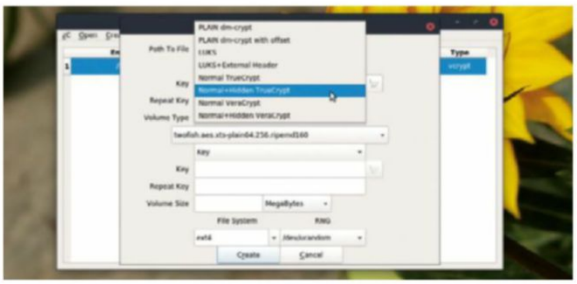
All encryption tools rely on a cipher to encrypt data. A cipher is an algorithm that does the computation for the encryption and decryption. Also important is the key size of the cipher that's used to encrypt. As the key size increases, so does the complexity of exhaustive search, to the point where it becomes impracticable to crack the encryption directly.

The most popular encryption cipher is the Advanced Encryption Standard (AES), which is based on the Rijndael cipher. AES with a key size of 256 bits is widely used because it offers the right balance of speed and security. Most of the tools in this month's *Roundup* default to this popular cipher and key size combination.

AES Crypt stands out from the rest because it only supports one cipher, which is a dead giveaway from its name. *Kgpg*, which is a frontend to *GPG*, is primarily used for key-based encryption but can also carry out symmetrical encryption that uses ciphers instead of keys. The `gpg --version` command will list all the available ciphers in your system. However, support for symmetric encryption in *KGpg* is rather limited, in that it's only available when encrypting text using the built-in editor and strangely not when encrypting files on the computer.

EncFS supports AES, Blowfish, Twofish, and any other ciphers available on the system. The tool also enables you to change the key length for ciphers that support variable key lengths. *VeraCrypt* defaults to AES, but you can choose from any of the five supported ciphers. It also supports cascade encryption, which is the process of encrypting an already encrypted message, either using the same or a different algorithm. It supports five algorithms for this purpose, including AES-Twofish and Serpent-Twofish-AES.

zuluCrypt is a front-end to the *cryptsetup* utility and by default sets up encrypted LUKS volumes. LUKS is the Linux Unified Key Setup, which is a disk-encryption specification designed specifically for Linux. It, too, can use any algorithm built into your kernel. You can double the key size to 512 bits or select one of the other two ciphers: Twofish and Serpent. These two are considered by the US National Institute of Standards and Technology to have a higher security tolerance than AES, but are computationally slower than AES.



Usually only the user can access the mounted volumes, but zuluCrypt can mirror its contents in a public folder.

Verdict

AES Crypt ★☆☆☆☆
EncFS ★★★☆☆
KGpg ★★☆☆☆
VeraCrypt ★★★★☆
zuluCrypt ★★★★☆

Besides AES Crypt, all other tools enable you to switch ciphers and key lengths.

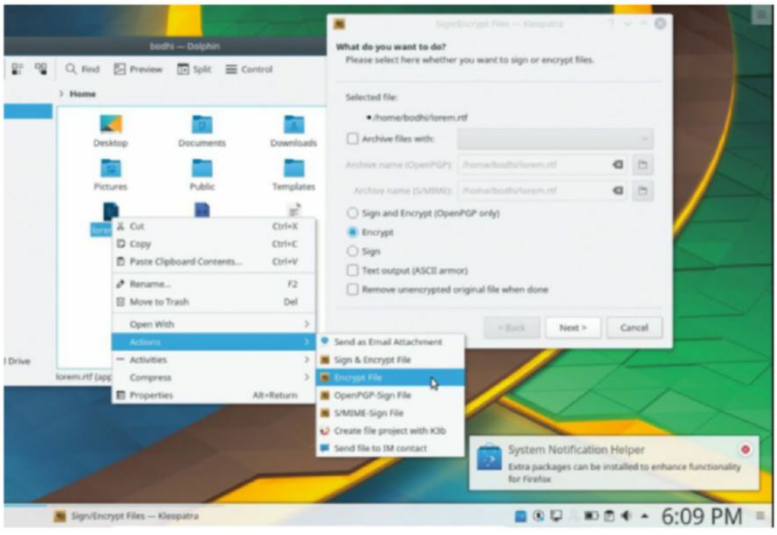
Integration and portability

How do they fit into the desktop?

You can mount the encrypted volumes using one of the several *EncFS* frontends. Some like *Gnome EncFS Manager* also have a tray icon and can even automatically mount and unmount encrypted folders on removable devices. KDE users can use the *kdeencfs* script to mount encfs encrypted folders using password stored in the KDE wallet.

On first use you'll have to manually select *AES Crypt* from a list of installed programs in your system. On subsequent use however, most file managers will list it in the right-click context-menu for quicker access. *KGpg* also integrates in the right-click context menu of KDE's file manager. There's also an icon in the system tray that can be used to launch the program's main window as well as for quickly performing certain actions such as encrypting and decrypting the clipboard contents and launching the editor.

VeraCrypt installs an icon in the taskbar that can be used to launch the program and even mount all encrypted volumes marked as favourite and open any mounted volumes in the file manager. *zuluCrypt* is currently available for Linux only. However, it can also create and open *TrueCrypt*, *VeraCrypt* and Plain volumes. *TrueCrypt* or *VeraCrypt* volumes are better alternatives if the encrypted volume is to be shared between Linux, Windows and OS X computers. You can also ask *zuluCrypt* to tie in with either the Gnome or KDE keyring for quicker access but you can't drag and drop files into the *zuluCrypt* interface to encrypt them.



All related options to encrypt and decrypt your sensitive files are housed under the Actions sub-menu for easier access.

Verdict

AES Crypt ★★★★☆
EncFS ★★★☆☆
KGpg ★★★★★
VeraCrypt ★★★★☆
zuluCrypt ★★★★☆

KGpg can't work with the file manager on non-KDE desktops.

Using the program

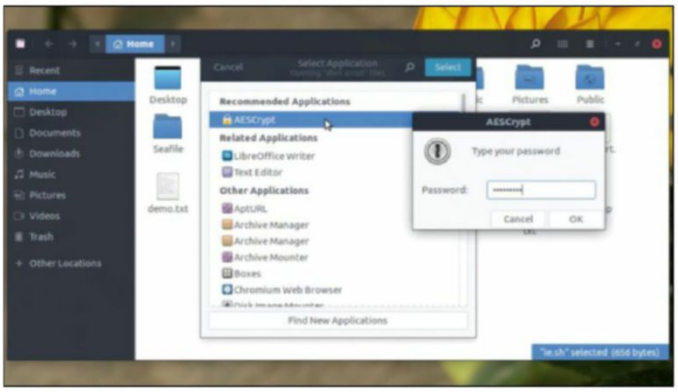
Are these encryption programs house trained?

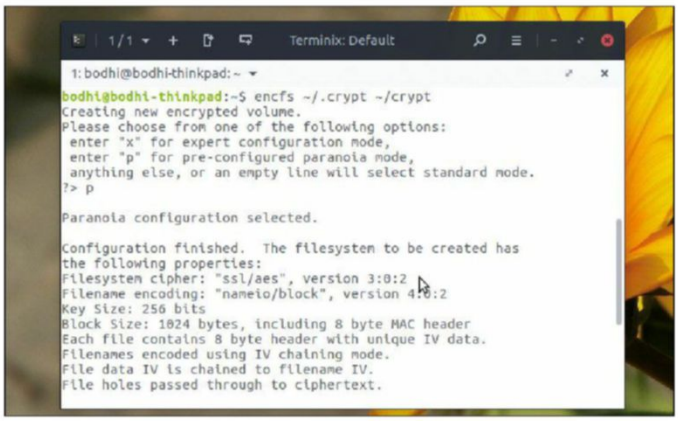
Tools that encrypt your data should be handled with care. The documentation of virtually every encryption tool has a section that warns the user about accidental data loss because of forgotten passwords, misplaced key files or damaged headers.

In essence, the same software that guards your data might also prevent you from accessing it ever again. This places even more responsibility on the developers of encryption software to make sure that users of their programs aren't overwhelmed by the tools at their disposal. In addition to being easy to install, a well-designed tool should display its features clearly and make it obvious which tools do what, so that its users aren't forced to look for the help file or support on the forums to get started.

AES Crypt ★★★★★

This is one of the simplest tools to install and use. Simply download and extract the installer from the tarball on its website. Then run the installer that prompts for superuser privileges before launching the graphical installer. To encrypt a file, simply right-click a file and select the relevant option in your file manager to open the file with another application. This will bring up a list of all the programs installed in your distribution. Select *AES Crypt*, which prompts you for a password to encrypt the file twice. That's all there's to it. The encrypted version of the file will be placed in the same folder with the same name but with an .aes extension. To decrypt the file you can open the .aes file with the *AES Crypt* program from the right-click context-menu. It'll prompt you for the password before decrypting the file in the same folder.





EncFS ★★★☆☆

EncFS is primarily a command-line tool that you'll find in the repositories of virtually every desktop distribution. While it doesn't have an official graphical front-end, Gnome users can use the *Gnome EncFS Manager* to manage and mount encrypted directories. However, you'll have to switch to the terminal to set up *EncFS* encrypted folders. The program requires two directories be used to keep encrypted and decrypted files. It's common practice to store the encrypted files inside a hidden directory. When you invoke *EncFS* it prompts you to select one of two configuration modes. There's the predefined paranoia mode that uses AES cipher with a key size of 256 bits. Advanced users can run *EncFS* in expert mode, which enables you to manually pick the various encryption settings. Similarly, when you mount a new encrypted directory, you get two pre-configured settings with different encryption settings.

Documentation & support

Need help? Online and offline assistance is usually available.

The primary source of usage information for *EncFS* is in the form of basic man pages. However, you'll also find detailed tutorials littered across the web. *AES Crypt* is a little better in that the project's website has basic usage information for both the graphical and the CLI versions of the program. There's also a link to download the more detailed User Guide in PDF format.

KGpg has a detailed user manual and you'll also find some tips and tricks in its page on the KDE UserBase wiki. You can ask for help on the KDE Community forums as well as on the kde-utils IRC channel. You can also post version-specific queries in your distribution's support infrastructure.

zuluCrypt's help infrastructure covers all the basics. The Help menu informs users about the importance of backing up volume headers. It also has a user guide that covers usage and also explains the pros and cons of the various supported volume types. The wiki on the project's Github has howtos on the CLI versions of the *zuluCrypt* and the zuluMount tools as well as a FAQ. Although it doesn't have a forum board of its own, its developer is quite vocal and active on other popular forum boards. *VeraCrypt* wins by featuring an illustrated tutorial for beginners as well as a detailed user guide that explains various things about the program's functionalities. There's also an in-depth FAQ and you can seek help on the official forums.

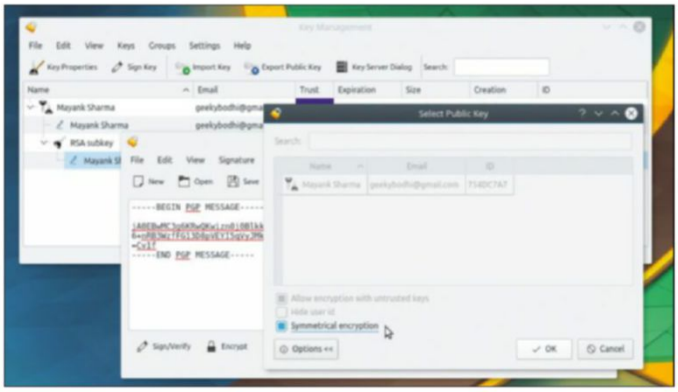
Verdict

AES Crypt ★★☆☆☆
EncFS ★☆☆☆☆
KGpg ★★★☆☆
VeraCrypt ★★★★★
zuluCrypt ★★★★☆

» *Barring EncFS all others tools have basic usage information to get you started.*

KGpg ★★★★☆

The first thing you need before using *KGpg* is a key and the program will automatically launch the key generation wizard at the first start-up. After creating the key you should also create a Revocation Certificate to void your key if your system or key is compromised. *KGpg* offers two methods to encrypt your data: symmetric and key-based. For the latter you'll need to exchange your public key with your friends and colleagues. You'll then have to encrypt the message with your colleague's public key, while they'll need their own secret key and passphrase to decrypt the data. You can also encrypt files by right-clicking the file you want to encrypt and choosing Actions > Encrypt File. Similarly, to decrypt a file bring up its right-click context menu and head to Actions > Decrypt File. You can also use the program to encrypt and decrypt the contents of the clipboard.

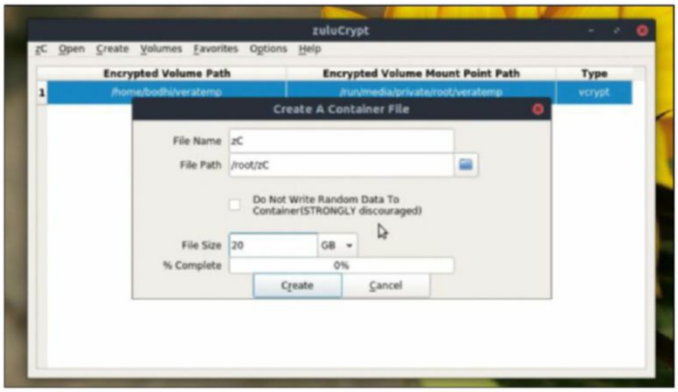


VeraCrypt ★★★★★

VeraCrypt is a fork of the now-defunct *TrueCrypt* project and has inherited most of its parent's functionality and also some quirks like its license, which is why it isn't available in any distro's software repository. The project also doesn't ship binaries for popular distros, but installing it is simple. The program has wizards for common tasks such as creating encrypted volumes and then mounting them, which makes it ideal for to new users. There's has an option to benchmark the speed for the encryption and decryption of various supported encryption ciphers. You can use the software to organise volumes and mount them with a single click. You can change the encryption password and add or remove key files to volumes. Conducting some tasks using its graphical interface is fairly straightforward, while others require some digging around.

zuluCrypt ★★★★★

Zulucrypt's website hosts binaries for Ubuntu, Debian, Fedora and OpenSUSE distributions. The program has a fairly intuitive interface. You get separate options to create an encrypted container in a file and in a partition. You can also create random keyfiles and use these to encrypt the containers. If you use the program to create a LUKS container, it'll remind you to back up its header immediately after creating the container. The tool also has options to encrypt and decrypt standalone files and securely erase a device by writing random data to it, as well as a graphical tool for mounting and managing volumes. The program can also perform block device encryption which means that it can encrypt everything written to a certain block device. The block device can be a whole disk, a partition or even a file mounted as a loopback device.



Mounting volumes

How easy is it to work with encrypted containers?

AES Crypt encrypts individual files, so there's a concept of volumes. Once you've linked a .aes file with *AES Crypt* you can double-click it in the file manager to decrypt it. Due to *KGpg's* architecture there's no concept of mounting volumes.

To mount an encrypted volume, in all other programs, you supply a password and/or keyfile. Once mounted, an encrypted volume behaves like any other disk. The biggest advantage with *EncFS* is that it can be used to protect existing filesystems without block device access, such as Samba shares or cloud storage folders. It also enables offline file-based backups of encrypted files. *VeraCrypt* can mount volumes via its graphical interface as well as from the CLI. While mounting you can optionally mount volumes as read-only and manually specify their mountpoint

zuluCrypt stands out from the rest and includes *zuluMount*, a mounting tool that can mount all encrypted volumes supported by *zuluCrypt*, including LUKS and TrueCrypt volumes. You can also mount volumes from the main *zuluCrypt* program, but *zuluMount* has a simpler interface and is designed just for mounting and unmounting filesystems. In fact, *zuluMount* can mount and unmount unencrypted volumes as well and can even manage plugged-in devices. Like *zuluCrypt*, the *zuluMount* tool has a CLI interface as well. *zuluMount* also makes it possible to make a mount point public and share it with other users.

Verdict

AES Crypt ★★★☆☆
EncFS ★★★★☆
KGpg ★★★☆☆
VeraCrypt ★★★★☆
zuluCrypt ★★★★★

» *zuluCrypt has more features and mounts more volumes than its peers.*

Configurability

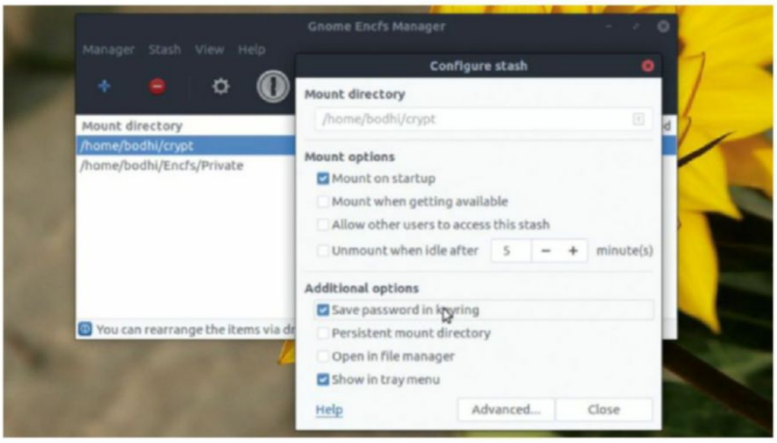
Is there something for the experienced campaigners?

While the programs ship with reasonable defaults, they should be malleable enough to enable advanced users to alter some parameters as per their requirements. There's not much you can tweak with *AES Crypt*. It's a very simple and straightforward program with hardly any configurable options. In contrast, its CLI version does make it possible to tweak a few parameters, such as using a keyfile. *EncFS*, when used in the expert configuration mode, prompts users for various parameters such as the cipher, its key size, type of filename encoding, and others.

The other three programs are all fairly equally matched in terms of the configurable parameters they offer. *KGpg* is like every KDE program and highly configurable. You can set default parameters for encryption, decryption, user interface and applet. Many of the encryption options are directly related to gpg. You can force all encryptions to be encrypted with a particular key in addition to the one manually selected. Similarly, you can also choose to encrypt files with untrusted keys.

You can instruct *VeraCrypt* to cache passwords in memory and also wipe the cache after exiting the program. You can also manually configure default mount options for mounting encrypted volumes and the program can use hardware acceleration to assist AES if the extensions are available in your processor.

While creating a new volume with *zuluCrypt* you have the option to encrypt with a keyfile instead of key and even use both for more security. You can also change the filesystem of the volume from the default ext4 to vfat, Ntfs, Btrfs and others. Besides English you can use the program in German and French, and also mark volumes as favourites for easier access.



Gnome Encfs Manager helps you to automount encrypted folders and then integrate them with the file manager.

Verdict

AES Crypt ★☆☆☆☆
EncFS ★★★☆☆
KGpg ★★★★☆
VeraCrypt ★★★★☆
zuluCrypt ★★★★☆

VeraCrypt, KGpg and zuluCrpyt offer varying degrees of customisation.

Encryption features

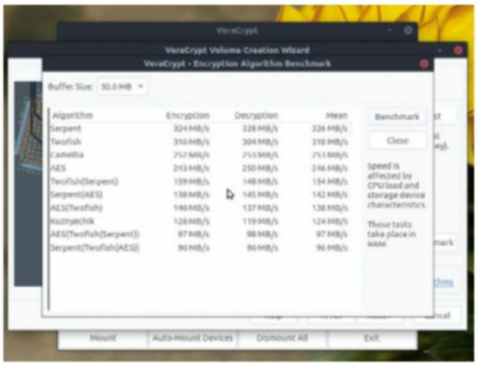
Which program manages to stand out from all the rest?

The graphical version of *AES Crypt* is rather rigid though the CLI version supports generating and encrypting using a keyfile. Unlike the other tools, *EncFS* has no volumes that occupy a fixed size: it encrypts directories that grow and shrink as more files are added or removed from the mount point. The encrypted files are portable in that they can be decrypted on another operating system.

EncFS's encrypted directory can also be located on a remote file server accessible via NFS, Sshfs and such. The tool has a special reverse mode that enables encrypted files to be backed up with normal file-system tools, such as *Rsync*. *KGpg* includes a simple text editor, where you can type or paste text to encrypt/decrypt it.

ZuluCrypt and *VeraCrypt* are fairly equally matched. Besides creating encrypted containers inside folders, both programs can also place the encrypted container inside a partition of its own or even on a USB drive. They also can't encrypt the boot partition or the boot drive. But you can use them both to encrypt a file with either a passphrase or a keyfile or both. Both also enable you to add a key or a keyfile to an already created volume and also offer the option of adding salt data to the encrypted container. The salt comprises values generated by their respective random number generators, and makes it difficult to pre-compute all the possible keys for a bruteforce attack.

Using *VeraCrypt* you can also benchmark and test all the supported ciphers while *zuluCrypt* can also be used to securely erase data in a removable device. *VeraCrypt* can also create hidden encrypted volumes for plausible deniability. However many experts have played down this feature's usefulness. *ZuluCrypt* can also create VeraCrypt's hidden volumes.



You can use VeraCrypt's benchmarking dialog to test the performance of the various supported ciphers on your computer.

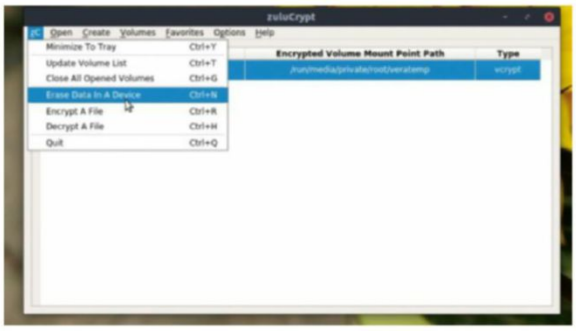
Verdict

AES Crypt ★★☆☆☆
EncFS ★★★☆☆
KGpg ★★★☆☆
VeraCrypt ★★★★☆
zuluCrypt ★★★★★

zuluCrypt and VeraCrypt reveal plenty of the features of their back-end tech.

Verdict: Encryption tools

The verdict



ZuluCrypt can create encrypted silos inside files and removable disks just as easily as it can encrypt files.

While all the tools in this *Roundup* work as advertised, we're on the lookout for the one that's intuitive to use. *EncFS* doesn't score well here because it's CLI-only, so you'll have to switch to the terminal to create folders to house your encrypted data. However, it isn't all that tricky to do so. Once you've created the encrypted folders, you do get graphical tools to manage interactions with them. While it'll grow on you, it loses out to the others because it lacks a graphical interface.

Its diametric opposite is *AES Crypt*. The program is simple to use and poses no learning curve whatsoever. If you want to reap the benefits of encryption with the least amount of time investment, then use *AES Crypt*. However, its simplicity and lack of configurability limits its appeal.

KGpg, being a KDE program, doesn't suffer from the lack of configurable options. But it comes in last because symmetric encryption isn't its strongest suit. Yes, *KGpg* is perhaps one of the best frontends to GPG. But GPG is mainly used for its implementation of public key cryptography that involves two keys and uses the recipient's public key for encryption.

It's a close contest between *VeraCrypt* and *zuluCrypt*. Both programs employ easy-to-use graphical interfaces and make light work of creating and using encrypted silos for securing any kind of data. *VeraCrypt* is a fork of the popular but controversial *TrueCrypt* program and retains its familiar interface. You can also install it on Windows and Mac OS X besides Linux which makes it cross-platform unlike the Linux-only *zuluCrypt*.

That said we'll award first place to *zuluCrypt* for a variety of reasons. First, it can work with multiple container types including those created with *TrueCrypt*, which makes it more versatile. Furthermore, *zuluCrypt* is more intuitive of the two. Its menus are better labelled and offer a clearer starting point that more than makes up for its weak support infrastructure.

> **"zuluCrypt's menus are better labelled that makes up for its weak support infrastructure"**

1st zuluCrypt ★★★★★

Web: http://bit.ly/2esjr5U **Licence:** GPLv2+ **Version:** 5.0.2

» Intuitiveness and list of features makes up for the lack of documentation.

2nd VeraCrypt ★★★★☆

Web: www.veracrypt.fr/en/Home.html **Licence:** Mixed **Version:** 1.21

» Cross-platform and feature-rich but not as intuitive as *zuluCrypt*.

3rd KGpg ★★★☆☆

Web: http://bit.ly/2wLTXaG **Licence:** GPLv2 **Version:** 17.04

» Good for KDE users, but symmetric encryption isn't its strongest suit.

4th AES Crypt ★★★☆☆

Web: www.aescrypt.com **Licence:** ISC License **Version:** 3.11

» Totally newbie-proof, but too simple for our tastes.

5th EncFS ★★☆☆☆

Web: https://vgough.github.io/encfs **Licence:** LGPL **Version:** 1.9.2

» Being CLI-only limits its appeal.

Over to you...

Do you still exchange data without encryption? Share your reasons with us at **lxf.letters@futurenet.com**. We (the NSA) are listening...

Also consider...

If you aren't averse to making use of CLI options then you can use *cryptsetup* to create encrypted volumes based on the dm-crypt kernel module. Then there's *tcplay*, which is a feature-rich BSD-licensed implementation of *TrueCrypt*.

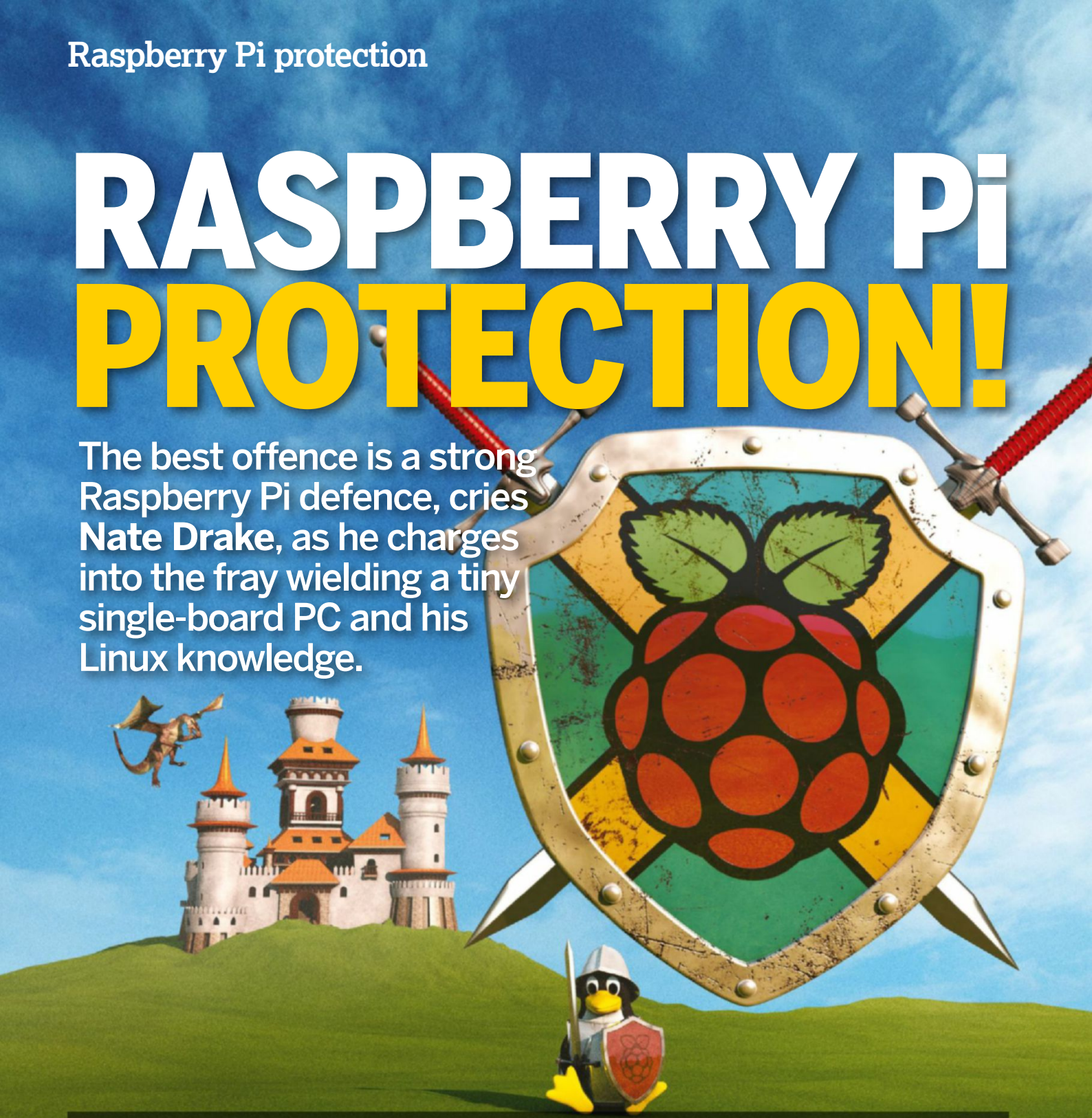
Another popular encryption utility that is able to carry out stacked filesystem encryption such as EncFS is *eCryptfs*, which operates at the kernel level. You can also use *gpg* directly from the CLI to take advantage of its symmetric encryption functionality.

Moving over to graphical alternatives, there's some encryption functionality built into Gnome's default file manager, which is called *Files*. The protection is rolled into the Compression dialog box. Then there's *CipherShed*, which is another alternative to *TrueCrypt* but the project shows no signs of active development.

Finally, there's *CryptKeeper*, which has been dropped from the repositories of Debian and Ubuntu because of the 'P' vulnerability that was uncovered earlier this year. LXF

RASPBERRY Pi PROTECTION!

The best offence is a strong Raspberry Pi defence, cries **Nate Drake**, as he charges into the fray wielding a tiny single-board PC and his Linux knowledge.



Your Raspberry Pi can be used as much more than a hobbyist's computer. As a separate machine, you can employ it to improve the security of your home or office network. Over the following pages, you'll learn how to transform your Pi into a wireless attack platform using the penetration OS Kali, capable of hacking networks.

The Pi can also function as a buffer between your computer and potential malware. For this reason, you'll also discover how to use it as an ad-blocking DNS server with *Pi-hole*, and a sanitiser for removing harmful files from USB sticks before you insert them into your PC.

More experienced users may also enjoy our guide on setting up your Pi as a network honeypot. This allows your Pi to masquerade as a full-blown web server complete with dummy files. All activity by hackers is recorded and no changes they make will affect other devices on your network, giving you crucial insight into who wants to access your data.

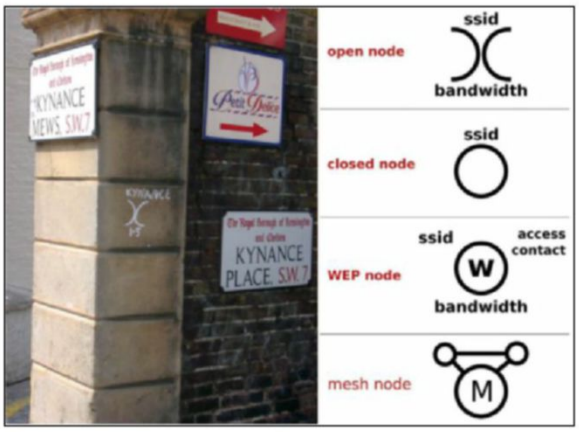
We start with the subtle art of Wardriving: driving around in a vehicle, while using a computer to search for vulnerable wireless networks to exploit. Wireless hacking software is freely available over the internet. Small computers like the Raspberry Pi are also very easy to power and conceal. As such, Wardrivers no longer require a team of men in a white van to break into wireless networks. Indeed, in recent years there have also been examples of Warcycling and Wardroning.

> **"Wardrivers no longer require a team of men in a white van to break into wireless networks"**

This guide focuses on how you, as an ethical hacker, can perform penetration tests on routers and IoT devices to make sure they're less vulnerable to this form of exploitation.

Before we begin, we'd like to offer the usual disclaimer that you should only perform penetration testing on networks with the permission of the owner, even (and especially) if you feel that their security is particularly lax.

As you're acting legally, there is no particular need to emulate Wardrivers fully by strapping your Pi onto a drone or vehicle. However, we encourage you to partner up with a fellow pen-tester and focus your Wardriving attempts on each other's networks instead of your own. This will make for a much more realistic test, as you can also see how easy it is for a stranger to gain physical access to the area where their network is based. It's also a good deal more fun!



Keep an eye in your area for War Chalking. Inspired by the Hobo signs of yesteryear, these alert fellow Wardrivers to various kinds of networks.

At the Wardriver wheel

To get started as a Wardriver, you'll need require a Raspberry Pi that supports wireless such as the Raspberry Pi 3 or Raspberry Pi Zero W. You'll also need a microSD card at least 16GB in size to install Kali.

All of the steps outlined below can be run from the command line, so technically you could boot your Pi and connect via SSH from a laptop or similar. However, it makes more sense as a Wardriver to connect your Pi to an external display such as the official Raspberry Pi seven-inch touchscreen display. This will save space, particularly if you install an onscreen keyboard such as matchbox-keyboard.

If you plan to use your Wardriving Pi in a vehicle, consider either connecting it to a portable battery pack or, better yet, to your car's lighter socket. Whichever power source you use, make sure it matches the Pi's requirements (5V, 2.5A).

You'll also need a wireless adaptor that can be connected via USB and is compatible with the *aircrack-ng* suite, in that it can enter monitoring mode and perform packet injection. When researching this article we used the Racksoy Professional Ralink 5370 (available from Amazon UK for around £6).

If you're pen-testing your own networks or a friend's then it's not very likely you'll need to map out exactly where they are as you'll already know! However, Wardrivers sometimes make use of GPS devices when running *kismet* to be able to locate target networks in their area.

As an ethical hacker, you might also wish to do this in order to seek out rogue Wi-Fi APs in your organisation, which can make your network more vulnerable. There are a number of GPS devices that are compatible. For this article we used the GlobalSat BU-353-S4 USB, which are available for around £30 on Amazon UK.

If you do want to plot Wi-Fi networks, open Terminal on Kali and run `apt-get install gpsd gpsd-clients`. This installs the basic GPS software. Connect the GPS device to a USB port on the Pi and run `dmesg | tail -n 5` to find out where it's mounted: for example, `/dev/ttyUSB0`.

Start the GPS Daemon at this location, such as `gpsd /dev/ttyUSB0`. Next, edit the *kismet* configuration file by running `nano etc/kismet/kismet.conf`. Uncomment the lines `gpstype=serial` and `gpsdevice=/dev/rfcomm0` by removing the # at the start. Replace `rfcomm0` with the actual location of the GPS device, for example `ttyUSB0`, then press Ctrl+X, Y, then return to save and exit.

Start the GPS device with `gpsd /dev/ttyUSB0`, then run `kismet -l`. You should now see the GPS data displayed in the *kismet* window.

This is automatically saved to an .netxml file in your home folder. You can use the program *giskismet* to transform this into a .kml file, which is compatible with map software like *Google Earth*. First install the program with `apt-get install giskismet`, then run it on the .netxml file: for example, `giskismet -x capture1-01.kismet.netxml`.

Next, use the command `giskismet -q "select * from wireless" -o <filename>.kml` to create the .kml file itself.

In the words of *Girls against Boys*, you can't fight what you can't see. This can lead some network managers and home users to think that using a hidden Wi-Fi network will protect them from Wardriving as a hacker would need to know both the name of the wireless network and the password.

Hidden networks are in fact extremely easy to detect. Using Terminal in Kali, simply run `airodump-ng <interface>`: `airodump-ng wlan1mon` for example, to list nearby networks. Any hidden networks will be listed, and only the ESSID (network name) is hidden.

Quick tip

For a full list of *aircrack-ng* compatible wireless cards visit **http://bit.ly/2gsD7HI**.

Quick tip

If, when trying to deauthenticate devices *aireplay-ng* says your device is using the wrong channel, run `iwconfig <interface> channel <channel>` to change it.

Yes you Cantenna!

Wardrivers sometimes employ long-range antennas to both access and exploit Wi-Fi networks from a great distance. The NextG TurboTenna (around £70) can detect networks from hundreds of feet away. It's popular with campers to make use of free Wi-Fi hotspots!

If you're handy with a soldering iron, consider building your own cantenna. This involves finding a Wi-Fi USB dongle with a detachable antenna and replacing it with one you've made yourself using a soup can or similar. You'll also need an N-male to RP-SMA-male cable, a female N-connector for the can and some 12-gauge copper for the wire element. The dimensions of the can (ideally 3-4 inches by 5) and length of the element are crucial. You can find a calculator here to get started (**www.changpuak.ch/electronics/cantenna.php**). You can also view instructions with photos at **http://bit.ly/LXF229can**.

A recycled can and a wire element acts as a powerful antenna for a USB Wi-Fi dongle.

» Run *airodump-ng* once again to focus on this specific network using its MAC address, such as `airodump-ng -c 1 --bssid CC:61:E5:CE:90:92 wlan1mon`. This will list any clients attached to the hidden network.

Finally, follow the steps in the guide to open a new tab in Terminal and try to deauthorise one or more of the devices, adding the `-c` option to target a specific client. For example:

```
aireplay-ng -0 5 -a C:61:E5:CE:90:92 -c 10:9A:DD:B3:48:0B wlan1mon
```

If you're successful, when you return to the original *airodump-ng* Terminal window, you'll see the name of the hidden Wi-Fi network has now appeared under 'ESSID'.

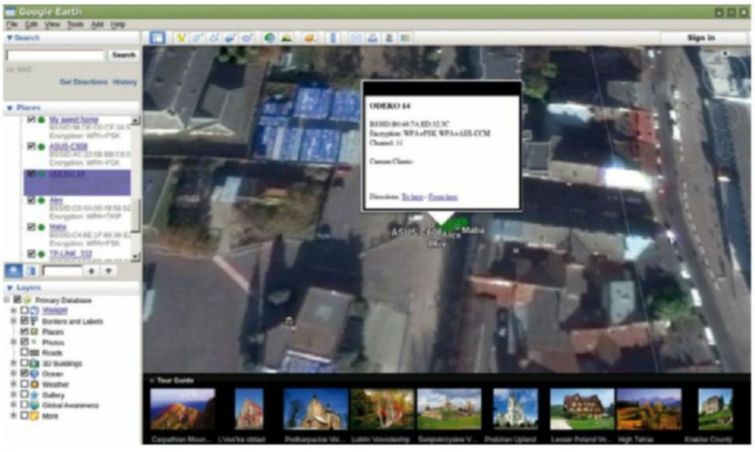
Helpful handshakes

When a client connects to a WPA-secured AP (Access Point), it engages in a four-way handshake. The AP initially sends a unencrypted nonce value to the client. The client then generates its own encryption key and nonce, and creates a transient key using its own nonce and the AP's. It then sends an unencrypted message to the AP containing this key. The AP can then extract the client's nonce and generate the encryption keys. It messages the client to verify it's the same device and asks if the client is ready to exchange encryption keys. The client responds and the connection between it and the AP is secured.

This is an oversimplification of how wireless networking operates, but it's important you understand this in general terms, as capturing the data packets used during handshakes is a crucial first step in breaking into a wireless network.

Quick tip

Check if you've captured a data handshake by looking at the top right of the window running *airodump-ng*. Press Ctrl+C to stop capturing any data.



Use software like Google Earth to display the .kml created by giskismet. This is an excellent way to find rogue access points.

As you'll see from the guide (*right*), it's easy to record data packets from a wireless network and capture handshakes. You can, however, increase your network security through using strong Wi-Fi passwords and changing them regularly.

Password cracking

If you follow the steps in the guide to capture data using *airodump-ng*, you'll find a data capture file with the extension .cap sitting in the **/root/kali** home folder **capture-01.cap**, say.

If the captured data contains handshakes between clients and the target AP, you can perform a dictionary attack on the password using *aircrack-ng*. This utility works by using a list of common passwords as well as words from the dictionary. You can find a number of popular password lists online, including that used by password cracking utility *John the Ripper*, which comes preinstalled in Kali.

To download the John the Ripper password list, open Terminal and run:

```
wget http://downloads.skullsecurity.org/passwords/john.txt.bz2
```

Extract the file by running `bzip2 -d john.txt.bz2` then begin trying to crack the Wi-Fi password with `aircrack-ng -w john.txt <capture-file-name>`, for example `aircrack-ng -w john.txt capture-01.cap`.

The Raspberry Pi 3 can check around 500 passwords a second which sounds impressive until you realise password lists can contain millions of words. As *aircrack-ng* can be run offline, we suggest you transfer the capture file to a desktop machine or use cloud computing to crack the password.

The speed at which you crack the password will also be determined by the quality of the password list. See **https://github.com/danielmiessler/SecLists/tree/master/Passwords** for a more comprehensive list of passwords. Be warned that some of these are well over 100MB.

If the target AP supports WPS, you may be able to break in using the Raspberry Pi alone without capturing or cracking any handshake data. See the Crack WPS with Reaver boxout (*below*) for help with this.

Wardriving works best when you're near the target network (the clue is in the name) and there are few other sources of wireless interference. If you want to make sure your network is safe at a range or find your current wireless card is impractical for pen-testing, consider using a larger antenna (*see the previous 'Yes you cantenna!' boxout*).

Even devices that are advertised as Linux compatible and/or you've used previously with Kali Linux on a desktop machine may not necessarily work with the version of the

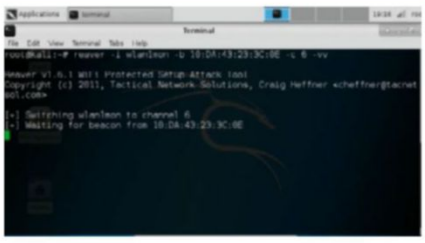
Crack WPS with Reaver

Many routers and IoT devices support WPS, which is supposedly a quick and easy way to connect to devices. In practice it's a security nightmare because connections are only secured by an eight-digit PIN. And the default PINS for certain devices can be found online!

If at all possible, try to persuade the network owner to disable WPS altogether. If this isn't possible (some routers don't support disabling WPS) you can at least see how easy they are to bruteforce using Reaver.

Open the terminal in Kali and run `apt-get install reaver` to get started. If you haven't done so already place your wireless card into monitoring mode with `airmon-ng start <interface>`. Next run `wash -i <interface>` to view all devices in range which support WPS.

Next, run the command `reaver -i <interface> -b <bssid> -c <channel> -vv`, for example, `reaver -i wlan1mon -b 00:19:70:70:15:2C -c 6 -vv` to begin cracking the PIN. Provided the device doesn't limit the number of attempts, it should take no more than 24 hours to access a device in this way by simply trying every combination. You can use Ctrl+C to stop the process, then resume from where you left off if you wish.



Reaver will attempt to bruteforce the PIN for clients using WPS. Typically, this takes around 10 hours.

Linux kernel you're using on the Pi. Make sure to research thoroughly before getting started.

If you boot all devices off a network but are still unable to capture handshakes, you may have more joy by targeting individual clients. See the previously mentioned section on hidden networks for details on how to do this.

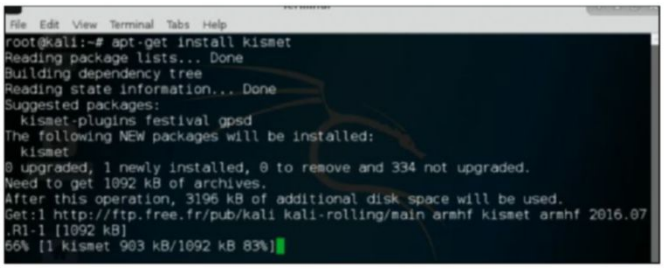
Both dictionary and brute force attacks on Wi-Fi passwords will take much longer on the Pi than a regular computer. We suggest you capture your files on the Pi, then transfer them elsewhere, for example to a desktop machine. If you've a few shekels to spare, consider cracking WPA2 passwords using cloud computing such as the Amazon Linux AMI, which can attempt dictionary and bruteforce attacks using GPUs, which considerably speeds up the process.

If you've been suitably terrified by what you've read here make sure to apply what you've learned and encourage your friends/clients to use WPA2-AES encryption where possible and disable WPS on all devices. If you discovered any lurking rogue APs, you may also want to ask them to draft up a policy for generating wireless networks in their workplace. »

Seek, locate and then deauthenticate

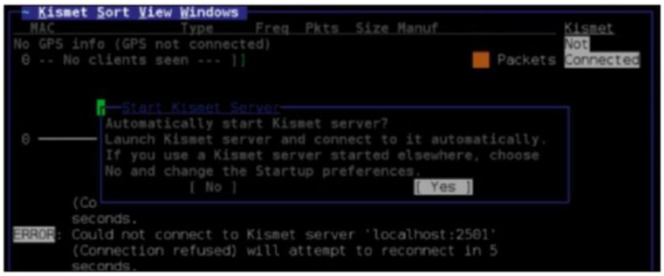
1 Install Kali

Download the ARM image for the Pi from **www.offensive-security.com/kali-linux-arm-images**. Navigate to your downloads folder with `cd Downloads`. Insert your microSD card, make sure it isn't mounted. Next use `dd` to write the image: `sudo dd bs=4M if=kali-2017.01-rpi2.img of=/dev/sdb status=progress`.



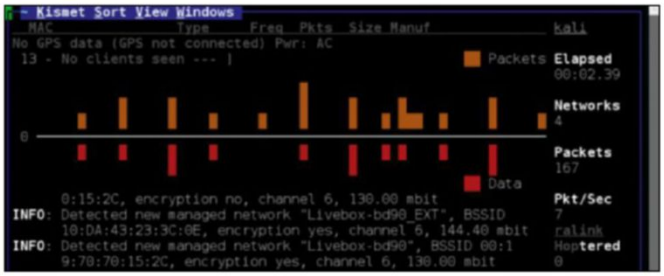
2 Set up Kali and install Kismet

Insert the microSD card and power the Pi. Once the login screen boots, use the username root and the password toor. Choose Use default config to display the desktop. Click the network icon at the top right to connect to your usual Wi-Fi network. Next, open Terminal. Run `apt-get update` and then `apt-get install kismet`.



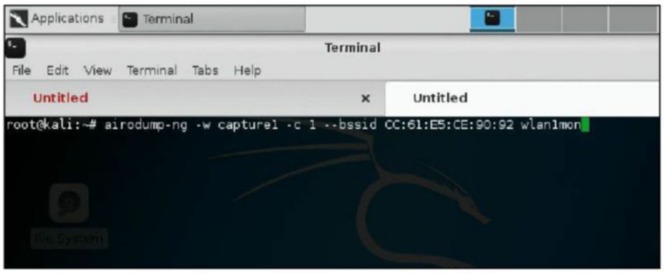
3 Configure Kismet

Use Tab and return to select OK on the root warnings, then select No when asked about setuid root. Choose OK to continue. Connect your Wardriving Wi-Fi card and antenna if applicable. Run *Kismet* in Terminal by typing `kismet`. Press return to ignore the warning about running as root. Press return again to start the *Kismet* server.



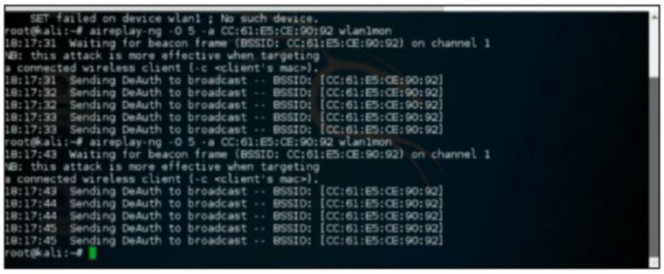
4 Run Kismet

Press return to Start and again on Yes. From here you can add the name of your interface, such as wlan1. If you're not sure of this, click File>Tab in Terminal and run `ip a`. Set a name if you wish then select Add. Press Return again to close the console and view the *Kismet* main window.



5 Capture Data

Note down the name, BSSID and channel of your target and run `airodump-ng –w <logfile> –c <channel> –bssid <bssid> -o pcap <interface>` to begin capturing data to a file, say `airodump-ng -w capture1 -c 1 --bssid CC:61:E5:CE:90:92 -o pcap wlan1mon`. The system will now capture and save data packets.



6 Deauthenticate devices

Open a new tab in Terminal and run `aireplay-ng -0 5 -a <BSSID> <Interface Name>`, for example `aireplay-ng --deauth 5 -a CC:61:E5:CE:90:92 wlan1mon` to try to boot off all devices five times and capture a handshake. Alternatively, run `besside-ng -b <mac address> <interface>` to keep going until you succeed.

Stop ads with Pi-hole

Set up your Pi as a powerful ad-blocking access point in minutes.

There aren't many projects on the Raspberry Pi that can be installed using just one or two lines of code. However, the creators of the *Pi-hole* project have created an installer that's so simple to download, you can set up your Raspberry Pi as an ad blocker in minutes.

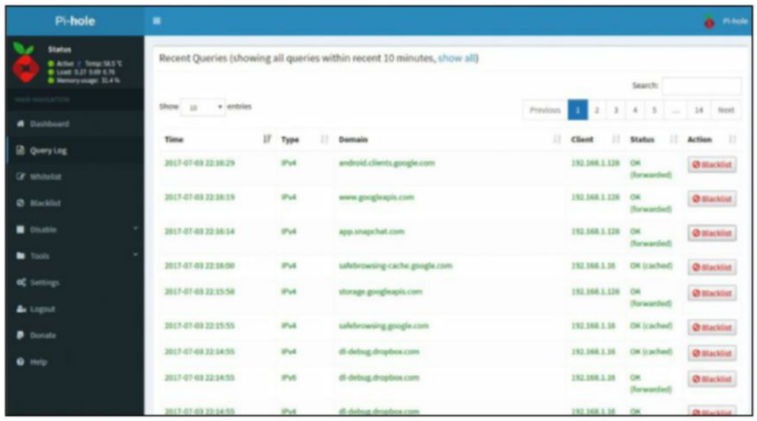
While you may be familiar with browser plugins such as AdBlock Plus, these are used to block code from web pages that have already loaded. The *Pi-hole* blocks advertising websites at a DNS level, so they're prevented from loading in the first place. This requires no client-side software and generally is a much smoother and easier way to keep your home or office network free of annoying adverts.

Quick tip

Visit **https://goo.gl/fcERhV** for tips on configuring DNS settings for common devices.

Opening your Pi-hole

In order to proceed, you'll need to install the latest version of Raspbian on your Pi. Make sure to run `sudo apt-get update` and `sudo apt-get upgrade` on your Pi before following the steps in the tutorial. The *Pi-hole* is compatible with all models of Pi, but if you wish to connect it to your router via Ethernet (which we recommend) you'll need to use either a Raspberry Pi 3 or a Pi Zero/Pi Zero W with a USB OTG Ethernet adaptor.



In Linux, choose Automatic (DHCP) Addresses Only, and then enter the Pi-hole's IP address in the box marked DNS Servers.

The term 'DNS Servers' is most commonly used to refer to publicly accessible computers, which contain hostnames such as **www.linuxformat.com** and their related public IP address. This enables websites to have recognisable names rather than a string of numbers. Every internet-enabled device you use queries a DNS server each time you try to access a website.

Pi-hole acts as a DNS server, connected to your local network. Any requests made from devices can be routed through the *Pi-hole* and checked against a constantly updated list called Gravity, which contains millions of domains that do little but offer advertising and spam. If a domain is listed, it'll be consigned to a black hole of internet advertisements. In less-poetic terms, *Pi-hole* will simply not load the domain.

For permitted domains, the *Pi-hole* accesses an upstream DNS server available on the public internet such as OpenDNS, which will then load your page seamlessly.

Follow the steps in the guide (*opposite*) to set up your *Pi-hole*. Once this is done you'll need to configure your router or devices to use the *Pi-hole* as their DNS server. See the boxout 'Down with DNS' (*below*) for more help with this.

Admin interface

If the prospect of automatically blocking millions of spammy websites isn't enough to entice you, *Pi-hole* also has a handy admin interface that's created automatically during setup.

You can access it either at **http://pi.hole/admin** or **http://yourPi-hole-ipaddress/admin**. The interface is protected by a password, which is generated for you during setup. You can change this from Terminal by running `pihole -a -p` .

Click Query Log in the left-hand side to view all the domains that have been accessed by devices on your network. Choose the Whitelist or Blacklist button next to each domain name to allow or block them respectively.

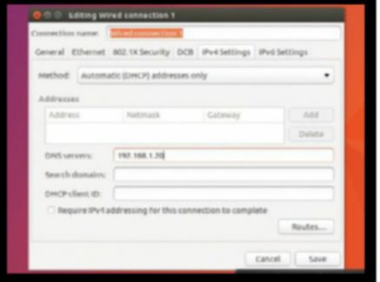
If you want to fine-tune your web filters further then click the Blacklist button on the left-hand side and enter a domain

Down with DNS

Once you've completed installing *Pi-hole* on the Raspberry Pi, you need to configure your various devices to use it as a DNS server. By far the most straightforward way to do this is to simply change the DNS settings on your router. This means all devices that connect to it will benefit from the *Pi-hole*'s adblocking features as well as logging all queries. If your router supports this feature, enter the *Pi-hole*'s IP address under the first DNS server, leaving the second blank.

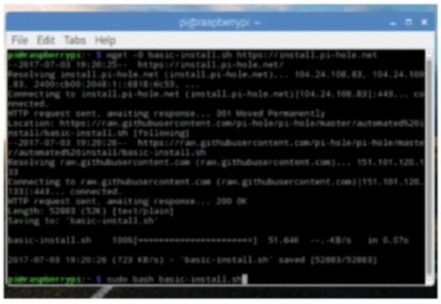
If your router doesn't make it possible for you to change your IP address then you may be able to change the DNS settings on individual devices. For machines running Microsoft Windows follow the steps at **https://support.microsoft.com/en-us/help/15089/windows-change-tcp-ip-settings** to change your DNS settings to the *Pi-hole*'s IP address. For Macs follow the instructions in the troubleshooting article at **https://support.apple.com/en-us/HT203244**.

Linux Users can usually modify their settings by opening Network Connections and then choosing Edit on their desired network interface. Click either IPv4 or IPv6 as you see fit and then choose Automatic (DHCP) addresses only from the drop-down menu. You'll now be able to enter the *Pi-hole*'s IP address in the DNS Servers tab. Click Save and exit the Network Connections dialog.



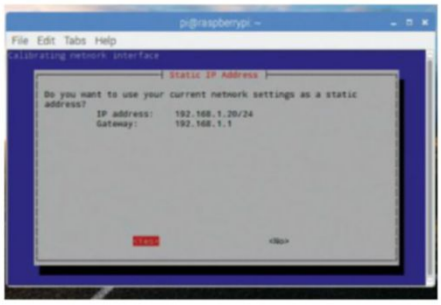
After setup, make sure your router assigns a permanent static IP to the Raspberry Pi using its MAC address.

Get Pi-hole up and running



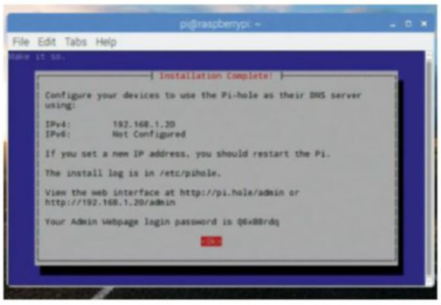
1 Run the Pi-hole installer

Open Terminal on your Pi or connect via SSH then run `wget -O basic-install.sh https://install.Pi-hole.net`. Next run `sudo bash basic-install.sh` to launch the installer. The program will automatically check for installed packages and ask you to confirm you want to install a network wide blocker. Press Return to confirm, then once again on the next screen.



2 Edit network settings

The latest version of *Pi-hole* can use your Raspberry Pi's current IP and Gateway address to assign itself a static IP. Use the space to select your network interface – for example, Ethernet – then choose your upstream DNS provider, such as OpenDNS. The installer will ask if you wish to keep your network settings for the Pi-hole's IP address. Select Yes.



3 Configure DNS settings

Once installation is complete, you'll need to configure your router or individual devices to use the Pi-hole's DNS address as your DNS server. (See boxout 'Down with DNS'). You may also wish to use your new password to visit the *Pi-hole* web admin interface at **http://pi.hole/admin** to review logs and blacklist certain sites. You'll need to restart the Pi before proceeding.

name to block it automatically. We recommend that you take some time to explore the Settings section to discover all of Pi-hole's features.

Is ad blocking theft?

Eyeo GmbH, the German company that makes Adblock Plus, has been through no fewer than six court cases by publishers, who claim that the blocking of their online ads is illegal, but so far the courts haven't agreed. In the case of Adblock Plus, websites are asked to pay in order to feature acceptable ads on a pre-approved whitelist.

Pi-hole, on the other hand, is a project that runs on a not-for-profit basis, relying on donations to stay afloat. This is one of many ways that websites can monetise their content besides displaying adverts. If you're using websites that are owned by multi-million dollar corporations, you also may not lose too much sleep in blocking their adverts before they reach your screen.

The strongest argument for ad-blocking software is that tracking cookies and advertisements can be used to gather information on your browsing habits and sometimes even install malicious 'junkware' on your devices. Most websites and apps have both a free and premium version, so if your conscience is pricking you, consider supporting your favourite services by subscribing or making a donation.

Getting Pi-hole up and running

During installation, the program will inform you that the *Pi-hole* needs a static IP address to function properly and offers to assign itself one using your network settings. Generally speaking, even if the device restarts any modern router won't try to assign it a new IP. However, to be on the safe side you may prefer to configure your router to assign the same IP to your Pi-hole's Mac address, so that none of your devices will run into problems.

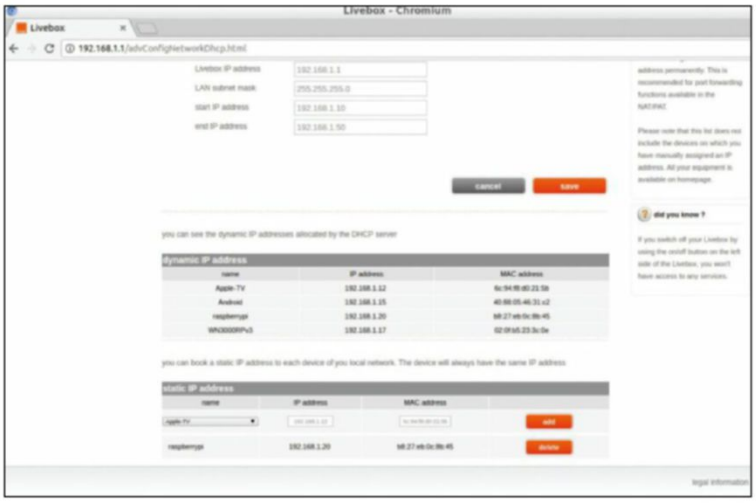
If your router doesn't support changing DNS settings, then you can configure each device manually to use the *Pi-hole*. See the Down with DNS box (*below left*) for more information. Bear in mind that some internet devices such as Google's Chromecast media streaming device don't support manually changing DNS settings. You can work around this by accessing Settings from the Pi-hole's web admin interface and use Pi-hole's DHCP server. Make sure to disable DHCP on your router if you do this.

If you're setting up the *Pi-hole* to protect your family from harmful websites, remember that anyone with access to a device can also change their own settings back to using a Public DNS. This will enable them to visit any site without being logged by the service.

Should you find the settings are too restrictive, then you can also whitelist certain domains by entering your admin password on blocked pages, or even disabling *Pi-hole* altogether for a certain amount of time – for example just for five minutes – from the web admin interface.

Quick tip

You can block all subdomains of a domain such as **888casino.com** by clicking Add (Wildcard) in Pi-hole's Blacklist section.



The Query Log dialog displays domains that are being accessed by devices. Click Blacklist to block a domain such as app.snapchat.com.

Wield a USB sanitiser

Use your Raspberry Pi to filter out harmful data from USB sticks.

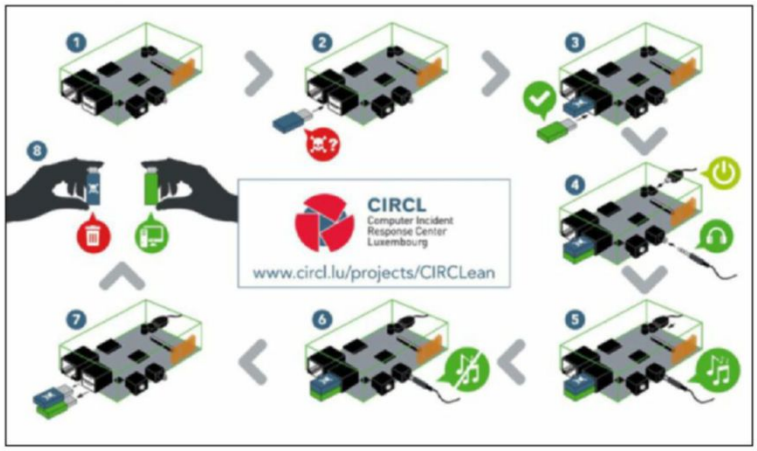
If you're reading this article, then you should congratulate yourself on taking one of the most important steps to protect yourself from viruses and malware. As a Linux user, your system can't be seriously impaired by harmful programs designed for other operating systems such as Microsoft Windows. Linux developers are generally security conscious and when vulnerabilities are discovered, updates are issued extremely quickly. This may help explain why there's never been a widespread infection of Linux systems.

Yet before you start to feel too smug, remember that although you may use Linux at home, your employer will most likely expect you to use a more mainstream OS for your work computer.

Linux also isn't immune from evil HID (human interface devices) such as the USB Rubber Ducky, which we covered in a few issues ago in **LXF226**. These devices can be made to resemble a USB stick, but when inserted into a computer act like a keyboard by running malicious code. The only requirement is for an attacker to persuade you or someone with access to your device to insert the evil HID into a vulnerable USB port. A 2011 study by Sophos also found that two-thirds (that's 66% to you lot – Ed) ttof a set of 50 USB keys bought at a major transit authority's Lost Property auction were infected with malware.

Quick tip

To be extra safe you can also check the gpg signature of the CIRCLean web page by visiting **http://circl.lu/verify/**.



CIRCL has supplied this handy infographic on how to use CIRClean. Remember to connect the UNSAFE USB stick first.

Enter the CIRCLean

In 2014, security expert Maya Bonkowski started working with investigative journalists and hackers on a project to sanitise USB sticks of malware, turning information into clean, readable data. The version of the project we'll focus on in this guide is named *CIRCLean*, and is maintained by the government-sponsored Computer Incident Response Center Luxembourg (CIRCL).

Maya originally envisioned the project to be primarily for activists and journalists who may need to exchange documents with contacts via USB stick.

CIRCLean is available as an image that can be written to your Raspberry Pi's SD card. Once this has been done, sanitising USB sticks is a breeze. The Pi is powered off and the 'UNSAFE' USB stick is connected to the top-left USB port. You must then insert a 'SAFE' USB stick of your own in the port below. The Pi is then powered on and *CIRCLean* begins the process of copying data from the 'UNSAFE' USB stick to the 'SAFE' one. This is done according to very specific criteria.

Plain text, audio and video files are simply copied across directly to your 'SAFE' USB Stick. XML files are converted to plain text and then copied across, too.

Image and archive files, for example .JPGs and .ZIPs, are copied after *CIRCLean* verifies they aren't "compression bombs". For this reason *CIRCLean* will only extract up to two levels of archives. This can result in the overall 'SAFE' data being larger than that on the 'UNSAFE' USB stick, so you may wish to use a 'SAFE' USB stick with a larger capacity.

Microsoft Office files are parsed with *oletools*. These are a handy set of Python applications that are used to find malware inside Office Documents and is marked as 'dangerous' if parsing fails.

Right on, write on

More canny readers will have noticed a potential flaw in the security offered by the USB sanitiser: if the target machine into which you plug the SAFE USB stick is itself compromised, then the files on it could be modified to include malware. You can reduce the risk of this happening by using a USB drive with a physical write protect switch.

If you flip the switch on these devices after files have been copied, the stick will be in read-only mode, so even if the target computer into which you insert it is compromised, this won't affect the data on the drive itself. There are certain software modifications you can make to put a drive into read-only mode, but these are much easier to reverse than a physical switch, such as that on the Kanguru ALK-FB30 (pictured), which retails for around £25 on Amazon UK and is Linux compatible.

Note that a read-only USB is only a protection against infection of the USB stick itself by the target system. If the USB contains malware, it may still be able to run in read-only mode. If there's sensitive data on your target machine, consider reading the USB using a live DVD or a virtual machine.

The Kanguru ALK-FB30 has a write-protect switch that can prevent files from being overwritten. No data can be added while in read-only mode.

Files deemed as potentially unsafe such as executables or PDFs that contain malicious code are marked as such by renaming them **DANGEROUS_<filename>_DANGEROUS**.

CIRClean can be run headlessly. To find out if the copy process is complete either connect the Pi to a speaker or headphones and wait for the music to stop, or simply wait for the green diode to stop flashing. Power off the Pi and insert your 'SAFE' USB stick into your own computer.

Cleaned out

The USB sanitiser is designed to protect you from a specific kind of attack, whereby malware is delivered during an exchange of data via USB stick.

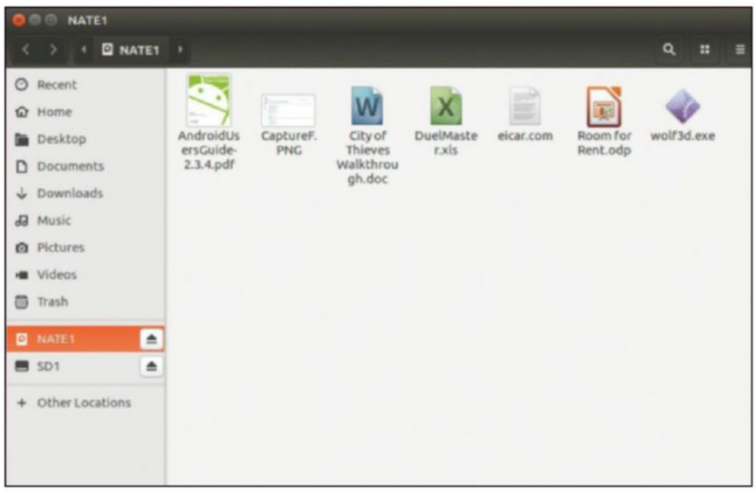
In order to reduce your attack surface as much as possible, make sure that you have a dedicated Pi for this project and don't connect it to the internet via an Ethernet cable at any time. If you need to update the software, do this by removing the microSD card and then following steps one and two in the walkthrough below.

If an adversary gains physical access to the sanitiser without your knowledge, they could modify the source code on the card. Either keep the Pi with you at all times or at the very least remove the microSD card and keep it on your person when not in use.

> **"Plain text, audio and video files are simply copied across to your 'SAFE' USB Stick"**

The security of the sanitiser also rests on the USB sticks being connected in the right positions, whereby the 'UNSAFE' USB stick is in the upper USB slot. (Think 'UUU' – Unsafe Upper USB). If you connect them in the wrong order, format the respective USB sticks and start again.

Currently *CIRCLean* only supports reading and writing to USB sticks formatted to FAT32 and NTFS. These are the most common formats, so this shouldn't pose any issues. However, for extra security you should seriously consider running Qubes OS on your own machine. The OS compartmentalises your digital life into various 'qubes' (virtual machines), including a dedicated one for reading USB sticks. This makes it extremely difficult for malware to infect your entire system.

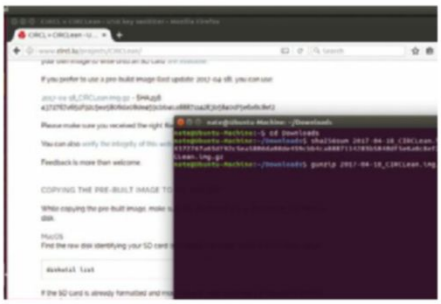


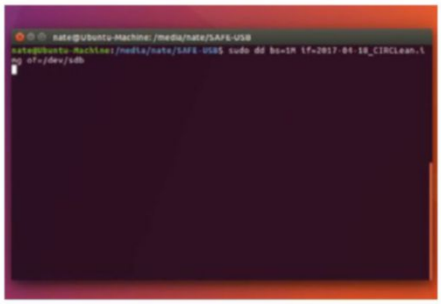
CIRCLean will automatically mark executable files (.exe, .com) as dangerous. Microsoft Office files (.doc, .xls) are scanned with oletools.

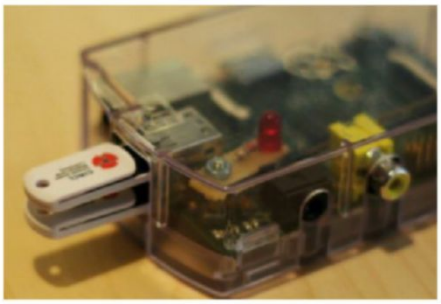
Quick tip

Ensure your Pi has enough power to transfer data between two USB sticks. Use an official 5V power supply.

Boost USB security with CIRCLean







1 Extract CIRCLean image

Browse to **http://circl.lu/projects/CIRCLean**. Click the link to download the most recent pre-built image (currently **2017-04-18_CIRCLean.img.gz**). Once the download completes use the `sha256sum` command, for example `sha256sum 2017-04-18_CIRCLean.img.gz`, to verify the integrity of the download. Next, run `gunzip` on the file, such as `sha256sum 2017-04-18_CIRCLean.img.gz` to extract it.

2 Write to an SD card

Insert your SD card into a card reader that's attached to your computer. Make sure that it's unmounted by using the *Disks* program if necessary. Now switch over to the Terminal and use `cd` to move to your downloads directory, for example `cd /home/nate/Downloads`, then use `cd` to write the image to your SD card, such as `dd bs=1M if=2017-04-18_CIRCLean.img of=/dev/sdb`.

3 Copy your data

Dismount the SD card safely to avoid any write errors. Disconnect the Pi's power supply and insert the microSD card. Next, insert the 'UNSAFE' USB key in the top-left USB port of the Pi (the one that's nearest the Ethernet adaptor on a Raspberry Pi 3). Plug your 'SAFE' USB key into the port below. Reconnect the Pi's power supply and the process will begin.

A taste of HoneyPi

Entice would-be hackers with a delicious honeypot of fake data ...

The honeypot is a traditional staple of Cold War era spy novels, whereby a socially awkward civil servant is seduced by a Russian femme fatale, then blackmailed into giving up precious state secrets.

In the Information Age, secret data is now no longer only at the mercy of balding government agents, but is stored on computer. Network administrators can reduce the risk of a breach through a combination of software updates, monitoring traffic, state-of-the-art routers and firewalls, but this may not put off a determined hacker.

What if there were a way, however, to convince a hacker that they had logged into your server when they actually were connected to a decoy machine? In this guide, we'll explore how to set up and install the honeypot software *Kippo* on your Raspberry Pi to do just that.

The basic premise is that once the software is up and running, you can configure port 22 on your router to forward automatically to Port 2222 on the Raspberry Pi. A hacker will access only the file system created by *Kippo* (designed to resemble a Debian Server). Any changes they make will be logged so you can view them later. Most importantly, none of the other devices on your network will be compromised.

Quick tip

Check that Kippo is listening on Port 2222 by running `sudo netstat -antp | grep 2222`. This will also display anyone who's connected via SSH.

To give you an overview of malicious behaviour, click the Geo tab in Kippo-Graph to bring up a list of probing attempts on a country-by-country basis.

Scores on the doors

Follow the steps in the guide (*below right*) to get started with *Kippo*. For security reasons, you should have a dedicated Raspberry Pi for this project, with a clean install of the latest version of Raspbian. You'll also need to be comfortable with forwarding ports on your router. The steps to do this vary from router to router but you can visit **www.portforward.com** to find instruction for the most common models.

Once *Kippo* has been running for a while, you can display the logs any time by running `cat /home/pi/kippo/log/kippo.log`. Bear in mind that this will display a huge amount of data as time goes on, however.

By way of an alternative, consider installing *kippo-graph* instead onto your Pi (see Install Kippo-Graph, *below*). Once the install is complete visit **http://ipaddressofyourpi/kippo-graph** to view any logged data. The Kippo-Graph tab will display the overall Honeypot activity such as the total number of login attempts and passwords used. Click Kippo Input to list which commands have been run. Selecting Kippo Play-Log will play a video in browser of all logins and commands run. Use the Kippo-Geo option to list incoming connections by country. From here you can trace the IP address of various connection attempts and even display the top 10 IP addresses on an interactive map.

Honey, I blew up the Pi

We can't emphasise strongly enough that this project is not for novices. If you aren't comfortable with managing routers, servers and firewalls, there's a real risk that in your attempts to set up a honeypot, you could make your network more vulnerable to attacks.

Install Kippo-Graph

If you're deadly serious about monitoring all connection attempts to your network, the program *kippo-graph* is able to aggregate all the data for you in a series of pie charts, bar graphs and maps. You can then study these to your heart's content.

To get started, open the Terminal on your Raspberry Pi or connect via SSH and then install the prerequisites for the software by running `sudo apt-get install libapache2-mod-php5 php5-cli php5-common php5-cgi php5-mysql php5-gd`. Next, restart the Apache service with `sudo /etc/init.d/apache2 restart`, then switch to the Apache directory by running `cd /var/www/html`.

Download the latest version of *kippo-graph* (note that the program is currently at version 1.5.1) by running `sudo wget http://bruteforcelab.com/wp-content/uploads/kippo-graph-1.5.1.tar.gz` and then extract it with `sudo tar zxvf kippo-graph-1.5.1.tar.gz –no-same-permissions`. Rename the *Kippo Graph* directory to **kippo-graph** – for example `sudo mv kippo-graph-1.5.1 kippo-graph` – then switch to it with `/cd kippo-graph`.

Run `sudo chmod -R 777 generated-graphs` to fix the file permissions, then create a configuration file with `sudo cp config.php.dist config.php`. Edit the file using `sudo nano config.php` and find the section marked `# MySQL server configuration`. Change the values as follows:

```
define('DB_HOST', '127.0.0.1');
define('DB_USER', 'kippo');
define('DB_PASS', 'password123');
define('DB_NAME', 'kippo');
define('DB_PORT', '3306');
```

Press Ctrl+X, then Y, then return to save and exit the software. Your graph should now be accessible at **http://ipaddressofyourpi/kippo-graph**, for example **http://192.168.1.17/kippo-graph**. If the graphs fail to load, then you'll need to modify the permissions on the entire *kippo-graph* directory with `sudo chmod -R 777 kippo-graph`.

Fortunately, you can reduce the risk of this happening by modifying the default SSH port of the Pi as outlined in the guide. The official *Kippo* page on Github (**https://github.com/desaster/kippo/wiki/Running-Kippo**) also recommends setting up a dedicated virtual environment for *Kippo* itself, which will sandbox it to some extent.

The *Kippo* software itself is designed to resemble a server resembling Debian 5 (Lenny). Because we're now up to Debian 9, this may alert some hackers to the fact that you're running a honeypot. One solution is to try and modify the configuration files to alter the server name and other default data, to throw off suspicious attackers.

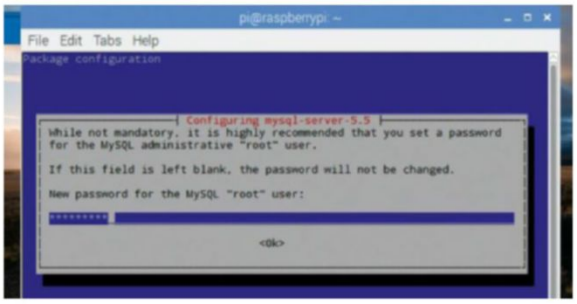
Remember that *Kippo* is only designed to protect against attacks via SSH, so hackers may be able to exploit services running along other open ports.

If you want a more comprehensive honeypot, consider using Michel Oosterhof's recent fork of *Kippo*, named Cowrie.

Anyone connecting via SSH will see a dummy file system. YAny files downloadd using wget will be stored in /home/pi/kippo/dl.

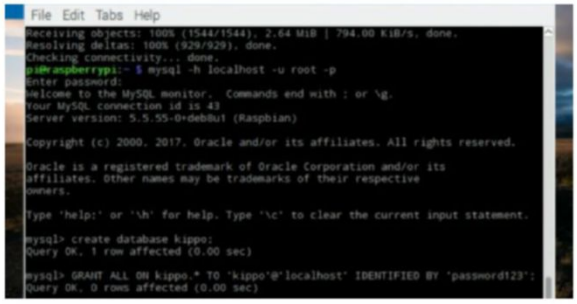
(find out more information about it at **https://github.com/micheloosterhof/cowrie**). Like *Kippo*, *Cowrie* still resembles a Debian 5 server but supports extra features, such as logging of ssh proxying attempts, forwarding SMTP connections to a separate SMTP Honeypot such as mailoney and storing log files in the universal JSON format. LXF

You're my honeypot



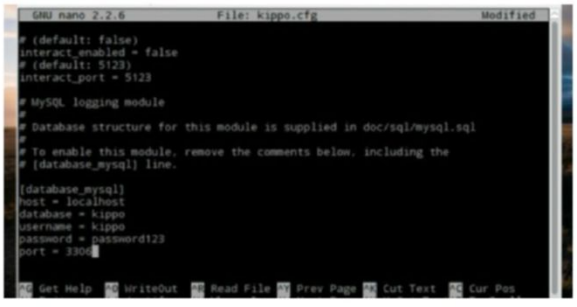
1 Install prerequisites

Open Terminal on your Raspberry Pi or connect via SSH and then run `sudo apt-get install subversion python-twisted python-mysqldb mysql-server apache2`. When MySQL opens, you'll need to configure a root password for the databases – enter this again to confirm your choice. Download the *Kippo* files themselves with `git clone https://github.com/desaster/kippo`.



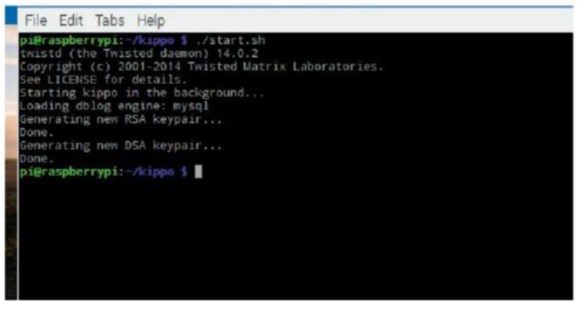
2 Configure Kippo database

Log in to MySQL with `mysql -h localhost -u root -p` and enter the root password. Enter `create database kippo;`. Assign user rights to "kippo" with the database password "password123" with `GRANT ALL ON kippo.* TO 'kippo'@'localhost' IDENTIFIED BY 'password123';`. Run `exit` then `cd kippo/doc/sql`. Load **mysql.sql** by running `mysql -u kippo -p`. Enter your database password.



3 Finalise Database configuration

In MySQL, enter `use kippo;`, then `source mysql.sql`. Run `show tables;` which will enable you to check the database fields such as 'ttylog' are present. Type `exit` and then move over to the kippo directory with `cd /home/pi/kippo`. Next, run `cp kippo.cfg.dist kippo.cfg`, then `sudo nano kippo.cfg`. Scroll down to the "[database_mysql]" section, comment out the lines and amend the relevant information as pictured above.



4 Configure Router, launch Kippo

Make sure that your router is configured to forward connection requests from port 22 to the Raspberry Pi's internal port (which is 2222). Next, change the Pi's own default SSH port to something higher – for example, 65534 – by running `sudo sed -i 's:Port 22:Port 65534:g' /etc/ssh/sshd_config`. Finally, run the script `./start.sh` from inside the kippo folder. You may now install *kippo-graph* if you wish.

Quick tip

The default root password for your honeypot is 123456. Change this by modifying **/home/pi/kippo/data/userdb.txt**. Change other values by modifying **kippo.cfg**.

Canterbury Christ Church University provided Oggcamp 2017's venue for free.

Oggcamp 2017

Les Pounder visits the historic city of Canterbury for the latest instalment in the Oggcamp story.

Interview

Oggcamp had a humble start, taking place the day after LUG Radio Live in Wolverhampton. It began life as a small event held on the top floor of a hotel, but interest grew and soon the OSS community formed around the event. The number of attendees grew, which meant hiring out larger venues with better facilities.

Oggcamp is something special: a community-focused unconference that has no affiliations to a technology or business. It was created by those who love Linux and the open source community, and chose to promote that love via their podcasts. Over the years the organisers have gradually stepped down in order to bring in fresh blood, and 2017 saw a new team assume the responsibility of running a two-day unconference.

Oggcamp still draws attendees from around the world, each with their own passion and project that makes up the diverse Linux and free software community. This year we saw a smaller, leaner event helmed by Oggcamp veteran Mark Johnson and Jon "The Nice Guy" Spriggs, himself a veteran of Linux/FLOSS events around the UK. Proceedings were held in Canterbury, in the historic city's Christ Church University's Powell Building.

Oggcamp 2017 may not have the grand market place and open hardware jam seen from previous events. But what it does have is a core team who work hard to meet the needs of the delegates. This isn't an event for the organisers. Instead, it's a service to the community that has grown around this popular event.

We spoke to the organisers Jon and Mark, Oggcamp veteran and podcaster Joe Ressignton, and community newcomer and PhD student Rachel Wong to understand what Oggcamp was, and why they had come along. We hope that there won't be such a long break until the next event: Oggcamp is an event that's unique and very much needed in our community.

Jon Spriggs and Mark Johnson

Linux Format: Hi Jon and Mark, thanks for taking the time to talk to us. As we sit here in the lovely Canterbury sunshine, Oggcamp is coming to an end. How was it?
Jon Spriggs (JS): It was really good – we survived it! I went into it thinking that there would be lots of little stressful moments. I don't know if it was either the awesomeness of the attendees or how amazing the crew has been, but generally there have been no "show stopper" issues. The T-shirts arrived late, which was a bit of a concern, but they made it through the doors in the end. If anything, that made the T-shirts more desirable!

LXF: Organising an event like Oggcamp isn't an easy task, is it?
JS: It's fair to say that this year's event took a little longer to spin up than in previous years.
Mark Johnson (MJ): Yeah I suppose so.
JS: So the building that we hoped to use in 2016 – well, that agreement fell through. It was nobody's fault. We were then on the back foot for 2016, which is why last year's Oggcamp didn't happen.

For 2017 we set ourselves a six-month window to get everything in place, and the first task for Oggcamp is always the venue. Once you have this, you can then pick your date. This gives our attendees time to arrange travel and accommodation.
MJ: It also gives us the time that we need to work with speakers and ensure that they haven't been booked already.
JS: We then have flurries of activity, organising the design of the logos for print and the website. Then we have a period of downtime while you wait for the assets to appear, then you get the logo and you have to think about getting the T-shirts printed, which then relies on having the funding to pay for it. So you plan "this bit" and then you wait for something to finish, then you stop. There were periods where you were less busy.
MJ: Yeah it's not a full-time job.
JS: I'm very fortunate in that people know what Oggcamp is. Once you have attendees who want to come and are ready to talk about their subjects, then everything else is just about making it right for them. You're not running a conference for yourself, but rather for those passionate attendees.

Organisers Jon Spriggs and Mark Johnson worked hard to bring Oggcamp back for 2017.

LXF: This year you have a rich mix of attendees, including those who are new to Oggcamp.
MJ: My favourite talk of the event was Rachel Wong, who has never given a talk at an unconference. She got up and did a lightning talk. In fact, I would say that this was the best talk of any Oggcamp. This shows that we have an event where people feel comfortable standing up and talking to the attendees. This is the best result that I could've asked for.
JS: I've been to previous Oggcamps as an attendee, and as such you're not looking at the event critically. You're there to enjoy yourself.

However, as an organiser you're constantly looking at the elements that make up your event. Are people happy? Is there something for everyone? In my talk this morning, there was an 11-, maybe 12-year-old girl who came along and was engaged with the subject of the talk. These are the next generation of people who want to work in IT.
MJ: Because we're not a vendor or corporate conference, we see a diversity of speakers who give more than just their scheduled talk. They make an effort to take part in the unconference element of the event.

LXF: Oggcamp is much more than just an unconference started by two podcasts...
JS: Yeah, the communities that surrounded the original podcasts (*Linux Outlaws* and *Ubuntu Podcast*) have a wide range of interests. I wouldn't say that the spirit has changed, but the message has certainly changed significantly over the years.

We're very keen to bring as many different threads and interests here. Ian Hutchinson from IF talked about using the products that they create for digital rights and so on. I can't imagine another conference doing what Oggcamp has achieved.

LXF: If there was one thing about Oggcamp that you could change, what would it be?
MJ: The only thing that I would change this year is that I would have liked to have seen more attendees. However, we had the right number of people for the size of the venue, so it worked out pretty well.
JS: Venues are unique, but they don't make or break a conference. While some people said they couldn't come due to the location, I had other who genuinely said to me that this venue is fantastic and wouldn't have come to it if it were anywhere else.
MJ: We have members of the community who live just down the road, and they have travelled across the UK for previous events. So it's great that for 2017 we've been able to bring the event to their doorstep.

MARK JOHNSON ON INTERACTION

"People liked to stand up and talk. This is the best result I could've asked for"

Rachel Wong

LXF: Hi Rachel, thanks for taking the time to speak to us. Please can you tell the readers a little bit about yourself?
Rachel Wong (RW): Hey there, my name's Rachel Wong and I'm a PhD student working in stem cell research, specifically the study of congenital blindness.

LXF: That's quite an impressive area of work. How do you manage to find the time to also be a maker?
RW: It's actually very challenging because a PhD takes up quite a lot of time. There are times where I feel torn, because as I'd like to do more with my PhD but at the same time I have a lot going on with my making and electronics. I do try to set myself some very strict boundaries, and I try to schedule my life so that I can balance the two.

LXF: So how long have you been a "maker"?
RW: I really got into electronics around March/April 2017 and that was due to the Raspberry Pi Zero W. At the Raspberry Pi Birthday Party I saw many great projects using it and I had the chance to talk to the makers about how they achieved it. It also helped that stalls were selling all of the components that you would need to make anything you want!

LXF: Being a maker isn't just about technology. You are also a crafter?
RW: Yes, I have an Etsy shop and before learning about the Raspberry Pi, electronics and Linux my shop was mainly craft related. But now there are projects involving LEDs, Neopixels and so on that merge craft, and wearable and technology together. In 2016 I took a gap year off to explore what I wanted to do and make and I got into quite a lot of things.

LXF: Being a newcomer to electronics and the Raspberry Pi, how was the learning curve and did you get any support?
RW: Learning the Raspberry Pi was actually pretty easy. I had some help from the great Raspberry Pi community, but I learnt about the Pi Zero W via the news, and as soon as I learnt they were selling out, I quickly bought one. I had to get one as everyone else was doing the same! Ever since then I've been learning new skills thanks to community members supporting me.

Rachel Wong enjoyed her first Oggcamp.

LXF: Do you have a background or interest in computer science?
RW: When I was in high school I taught myself some HTML and CSS, and this was mostly behind my mother's back. While she was away working, I would sit down and teach myself, although to be honest I didn't think that it would be much use, until recently!

LXF: So what's your primary programming language, would you say?
RW: Currently Python, because it's the one that I understand the most. It's also the most-used language for the many projects that have been created by the community, which gives me a rich resource to reference. I can read and understand the code written by others and I know that I can have an idea, search for it and then find an existing "skeleton" project that I can adapt and use to form the basis of my next project.

LXF: So your introduction to Python was via reading the code of others, tweaking it and reusing it?
RW: Yes, there are lots of great tutorials for Python, just as there were for HTML and CSS when I first started to learn those languages.

LXF: So now that you have your new "super powers" of electronics and coding with the Raspberry Pi, you've developed your own wearable projects. Can you tell us more?
RW: I've submitted a proposal to run an exhibition and the plan is that I'll show five complete outfits, with hats, jewellery, umbrella and bag – all of which will have elements of control based upon the Raspberry Pi.

LXF: Was wearable tech a natural avenue for you to explore?
RW: Because I've been selling jewellery and craft products on Etsy, my journey to creating wearables started by making crowns and from there I slowly started to introduce electronics. I then wanted to take the concept of wearables further and so I went online to search for new ideas and to see what other people have come up with. I then realised that it's still a niche group and that I could do more to create new wearable projects.

LXF: So this is your first Oggcamp? How have you found it?
RW: Initially, I felt very intimidated because the first talk was on security, and I didn't know what the speaker was talking about. But after that first talk I was more selective and found other talks that I could relate to and take part in. I also did my first talk, a lightning talk (five-minute presentation, and time for one question) which I didn't prepare for and just did on the spur of the moment!

RACHEL WONG ON SPEAKING UP

"I did my first lightning talk, on the spur of the moment – I didn't prepare for it!"

Joe Ressington

Joe Ressington favours a London venue.

LXF: Hi Joe, thanks for taking the time to talk to us. Please can you tell the readers more about yourself.
Joe Ressington (JR): Hi, I'm Joe Ressington and I'm the host of *Late Night Linux* podcast, and the co-host of *Linux Action News*.

LXF: So how long would you say you've been podcasting for?
JR: I've been podcasting for around four to five years and I started out on *The Mind Tech* podcast which was on the Mindset network, a network which isn't really going anymore.

The podcast was a mix of conspiracy theory-type stuff and technology. I presented the podcast with Gareth Davies, a Mac user and we bonded together over our dislike of Windows. We started the podcast just before the Edward Snowden event and all of the conspiracy theories we had been talking about came true! So we felt vindicated!

LXF: So your area of interest is in security?
JR: No, I would say that I'm more interested in Open Source Software really. Open standards and security work hand in hand. You can't really have proper security without open source or open standards.

LXF: So is Linux a secure operating system? Recently there have been some Linux security issues hitting the headlines: Dirty COW and the Bitcoin malware for Linux, and MulDrop.14 for Raspberry Pi.
JR: You're always going to have Zero day exploits because of code churn, but I like to think that Linux is more secure than Windows or Mac, insofar as someone can find the exploits in Linux and fix them. Whereas you're relying on a closed source company to fix exploits and close back doors. So government agencies may possess back doors into Linux, but hopefully they'll be found and fixed by the community.

LXF: So here we are at Oggcamp 17 in the lovely city of Canterbury, but this isn't your first Oggcamp. How have you seen Oggcamp evolve over the years?
JR: If I remember correctly my first Oggcamp was in 2011, in Farnham and I've only missed one event since then. I've seen the appeal of Oggcamp become more selective. It seems to be less popular than it once was. It's still a good event and the people who attend are really cool, but there are just fewer people here than there have been in previous years and this is a trend that I've noticed.

Maybe this is because there are now more maker events, and when Oggcamp first started it was the only big event that was happening, whereas now there are a lot more events taking place. I suppose the community has become "fractured" or spread out, but Oggcamp is still a great event and I had a great time this weekend.

LXF: Is there still a desire for Oggcamp?
JR: There's still a desire for Oggcamp – people still want to come. Apart from anything else this is a social event, giving the community a chance to catch up. Granted, it's good to talk online, but Oggcamp gives us the chance to have a drink and a chat face to face.

For me the most important aspect of Oggcamp is the "social track". Sure I can see lots of great talks – in fact I watched a great talk on Open Suse, and then had the chance to talk to the speaker about his project and other interests in the pub afterwards.

LXF: Do you think what makes Oggcamp different to other technology-focused or corporate conferences is that it addresses the community, rather than the technology?
JR: Yeah that's definitely it. Oggcamp is less corporate. Of course, there's some networking going on and people will be learning new skills for their jobs, but at the same time there are talks going on that cover personal projects. You would never see that at a corporate conference because it has no corporate value. Oggcamp is not about corporate culture, it's basically about enthusiasm.

LXF: If you could change one thing about any Oggcamp, either past or present, then what would it be?
JR: The location! So I can take the Tube there and back and not pay for a hotel! It's all "up in the air" but we may have found a suitable venue in London, which would be great! More people coming would also be great. That said, it's nice that it's a small event and that you have the time to properly catch up with people.

JOE RESSINGTON ON MAKER POSITIVITY

"Oggcamp is not about corporate culture, it's basically about enthusiasm"

This year, I think Canterbury put a few people off, especially those coming from the north because it's quite a journey. The venue, Canterbury Christ Church University has been great, and by offering the event to Oggcamp for free it's saved the organisers thousands of pounds and many weeks of work, which means they can concentrate on the event.

LXF: So what has been your favourite part of Oggcamp 2017?
JR: I think that for me it was the conversation that I had with Martin Wimpress (Ubuntu MATE Project Lead) and Richard Brown (Suse) in the pub last night. We talked about the new packaging format "Snap" and it became quite heated. I think that Snap is one of the biggest developments in Linux in recent years. LXF

20 Years of KDE

KDE has evolved into one of Linux's finest desktops. Jonni Bidwell reminisces, explores and konfigures

About 21 years ago Matthias Ettrich announced his plans for the Kool Desktop Environment: "a GUI for endusers." He recognised that there was much work to do, and initially sought between 20 and 30 developers to get the project underway.

Using an exciting new toolkit called Qt, he saw an opportunity to bring some uniformity to the hodge-podge of toolkits and homespun solutions used by applications hitherto. One year later KDE Beta 1 was released, and the rest, as they say, happened.

Active KDE contributors now number around 1,800 and the KDE codebase is made up of over six million lines of code. Today, criticism of the Linux desktop is commonplace (fragmentation, disregard for old paradigms, bloat, inconsistency, lack of integration) and it's easy to forget how far things have come. The initial release of KDE 4 in 2008 may have upset some people (including Linus Torvalds), with its instability, glowing-by-default Windows and the undue burden it placed on older systems. But it also heralded a new era of innovation, where Linux desktops no longer sought just to imitate Windows and OS X.

> "The Plasma desktop is one of the most advanced in the world, and one of the most popular"

The Plasma desktop today is one of the most advanced in the world and, thanks at least part to Gnome 3's radical departure from the conventional, is now one of the most popular in the Linux ecosystem. KDE's Visual Design Group ensures it's also one of the most stylish, with its polished Breeze theme, slick yet subtle effects and support for HiDPI displays.

You can read Matthias' historic announcement in full at www.kde.org/announcements/announcement.php. Back in 1996 he noted that most applications used their own widgets, and they all looked and behaved inconsistently. There were several widget toolkits around, but none of them were ideal or consistent. Some of that widgetry, for example GTK+ and Motif, is still alive today, and the idea of writing use-once code for home-brewed buttons and dialogs is mercifully a thing of the past. Matthias saw a glimpse of the future in an exciting new widget toolkit with a cute name, Qt.

Like its GTK+ counterpart today (which had its genesis in the *GIMP* application), Qt was originally designed for a single application, namely a database for ultrasound images. By the time Matthias was hatching his grand plan, the authors of said application had set up their own company, Trolltech (later Troll Tech, then reverting to Trolltech), where Qt development continued. Matthias saw Qt as "a revolution in programming X". Perhaps controversially, he also saw the that fact that it was developed commercially as an advantage.

Trolltech made Qt sources freely available under its own licence, but permissive though it was, this licence wasn't looked upon favourably by the Free Software Foundation. Specifically, it didn't approve of the licence forbidding the redistribution of modified versions – a key tenet of the open source principle.

Initial launch

Be that as it may, KDE (which was now just the K Desktop Environment) 1.0 was released in July 1998, incorporating Qt 1.3 under this licence. The release announcement expressed developers' hopes that KDE would "bring open, reliable, stable and monopoly free computing to the average computer." By this time, there was already considerable interest in KDE, and the idea that Linux's leading desktop environment (Gnome hadn't been released yet) might one day be proprietary led to concern and consternation. KDE 1.0's components included kwm and kfm, which would respectively inspire KDE 2's Kwin window manager and Konqueror file manager, both of which live on to this day (although after an extended period of identity crisis *Konqueror* is now a web browser).

The more zealous critics called for people to boycott KDE in favour of Motif (bizarre, since at that time it could require royalties to be paid) or its LGPL-licenced cousin LessTif. Qt 2.0 was made available under the Q Public licence in June 1999 and, after some wrangling, Qt 2.2 appeared in December 2000 under the GPL 2.

Those wishing to know more about what early KDE was like need not rely on the few screenshots one can find on the Internet Archive. This time last year, to celebrate the anniversary of the project announcement, the KDE Restoration Project revamped the KDE 1.1 codebase so that it could compile (using cmake rather than the antiquated `auto*` agglomeration of horrors) and run on a modern system. See **www.heliocastro.info/?p=291** for more info, or build it yourself from **phabricator.kde.org**: the repos you seek are called `KDE 1: Qt 1`, `KDE 1: Libraries` and `KDE 1: Base Module`.

OMG! KDE 1 running on a modern system. The KDE gear branding was present from day zero.

Three is the magic number

Desktop Linux welcomed many refugees when what's generally agreed was an abomination of an operating system, in the form of Windows Vista, was released. By this time KDE was well into its third outing (version 3.0 was released in April 2002) and enjoyed the dubious honour of being the most Windows-like of all the UNIX desktop environments. So readers of a certain age will probably remember it fondly.

A major feature of KDE 3 was its support for locking things down, a feature often requested by developers of kiosk-type systems. So KDE was experiencing adoption outside of the

Make way for Plasma Mobile

If you caught our Linux on Phones feature last month (and if you didn't you really should as it was written most eloquently), you may have been left feeling slightly despondent that there isn't really a way to have a 'proper' GNU/Linux installation on your phone while still retaining standard phone functionality.

This will change in the future and one of the key players here is *Plasma Mobile*, which you can already run on Nexus 5 and 5X devices. KDE Frameworks provides componentry much more advanced than what's available in Qt 5, and, thanks to forward-looking design decisions, can cater equally well to mobile and desktop form factors. These libraries sit on top of Halium, the exciting effort to standardise middleware and provide Android compatibility for GNU/Linux on mobile. On top of them sits the new and exciting Kirigami UI, which also defines its own philosophy about how programs should lay out their UI/UX. It's not just for mobile either, being suitable for use in convergent programs across devices.

Unfortunately, convergence seems to be a thing that's not quite fructified yet, but in the event that it does it's nice to know that there's a great framework for making them look pretty. You can see Kirigami in action on Android in the Subsurface diving app. It has as a contributor: one Linus Torvalds...

Soon your dragons will be able to run libre OSes on their phones. Soon...

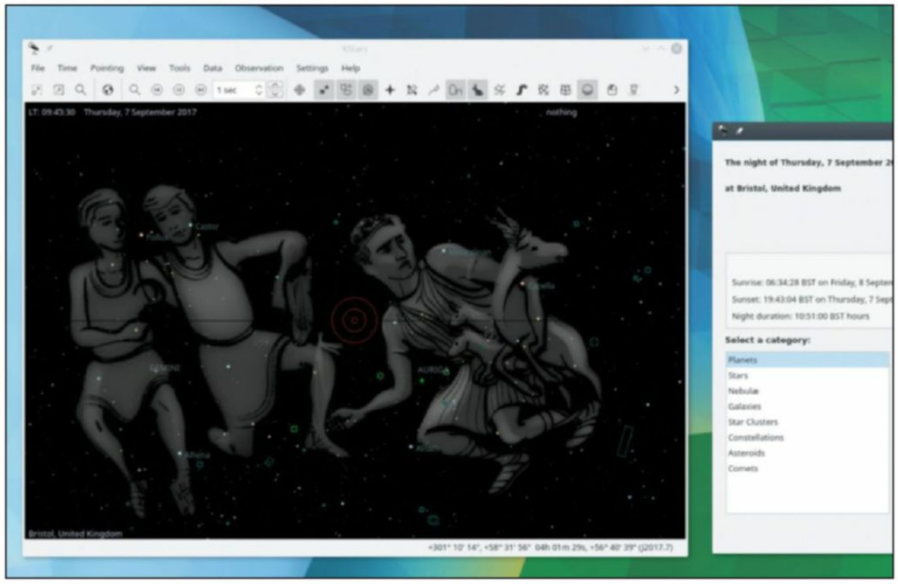
Image credit: KDE, CC BY-SA 4.0

traditional personal desktop sphere during the dot com era.

Changes in TrollTech's licencing policy enabled Qt 4's intrinsic operating system agnosticity to be fully harnessed, so that KDE 4 (released early in 2008) could be ported to other operating systems, the burgeoning mobile market being a major aspiration, as well OS X and Windows versions. The latter two never really saw serious adoption, but things look promising for mobile (*check out the box, below left*).

Controversially, version 4.0 was released in a deliberately unfinished state, as a sort of developer preview. The rationale was to garner interest and generate feedback. This more or less backfired, unless generating complaints and bug reports counts as success. Linus Torvalds was said to be bemused by the over-configurability of it all, claiming his explorations had made his shortcuts "look like a drunken fratboy had been messing with my desktop." There were also complaints about its resource usage and unnecessary frippery in the form of graphical effects. By the time 4.2 was released, in November 2008, most of the kinks had been ironed out.

A year later, KDE SC (Software Compilation) 4.4 was released, with the new name alluding to the wider scope of the KDE community. KDE 4.7 saw many of its components ported to QML, so that they could capitalise on GPU power and the new QtQuick rendering framework, so that those pleasing fade effects, shadows and glows all rendered with the upmost of slickness (unless your graphics drivers refused to play nice with Kwin). KDE 4 saw the aRts sound daemon replaced by the Phonon multimedia API. And Pulseaudio became the preferred means of messing up your audio. Another exciting addition was the Threadweaver library for harnessing the power of multicore systems.



Kstars is part of the science offerings in KDE Applications. Last month's full moon in Gemini caused some problems for our druid in residence.

Packaging KDE used to be an all or nothing task. Applications were bound to desktop libraries, so that if you wanted to install, say, *Kate* (the text editor), on a lightweight desktop, then you'd end up with most of KDE installed alongside, and your lightweight desktop would cease to deserve its adjective.

A modular approach

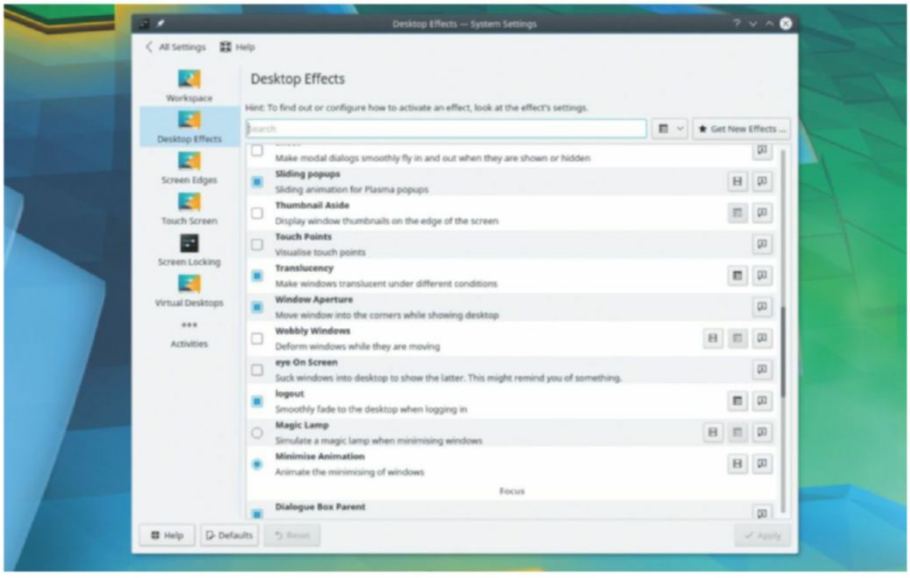
A change was signalled around 2006 with the KDEmod effort, which was aimed at boosting the modularity of the KDE desktop within Arch Linux. The idea of having to install a single monolithic package with many applications that went unused was an anathema to Arch's KISS (keep it simple, stupid) philosophy. This project eventually gave rise to Chakra Linux in 2010. KDEmod and Chakra were initiatives outside the KDE community, but with the benefit of hindsight it's easy to imagine that they influenced the restructuring that took place towards the end of KDE 4's reign.

Back then, in late 2013, the project was ostensibly divided into Platform, Plasma Workspaces and Applications factions, but development of these was sufficiently intertwined and interdependent as to make the distinction (at least from a project management point of view) moot. Their releases all had to be cadenced, and new features couldn't be introduced in one area unless the other two had accepted the changes necessary to accommodate them.

> **"Controversially, version 4.0 was released unfinished, as a sort of developer preview"**

For the next release, in which Platform (a monolithic collection of libraries and services on which Applications depended) would become Frameworks (featuring discrete libraries such as Baloo, Nepomuk and Phonon), a bold effort was undertaken to decouple these efforts. They could then have independent release cycles and things be more modular and manageable. This effort paid off, and the fifth KDE desktop was launched with a new name, Plasma 5, though calling it KDE Plasma 5 is still permitted.

The KDE tricronym today is used to refer to the community rather than the desktop or any particular component thereof. In some sense this was a much less 'in your face' release than KDE 4, although the



The default selection of desk effects is pleasing to the eyes, but less-subtle ones are available.

configurability is still there though. You can even have an Ubuntu-style global menu if you dig around in Application Style>Window decorations settings. Another welcome addition is the launcher being mapped to the Super (Windows) key out of the box.

KDE has been represented by the KDE eV (registered association) since 1997. German law required seven people to form an eV, and Matthias had to enlist housemates and developer's partners to make up the numbers. Today, the board has over 150 members and its remit is to provide assistance and distribute donations, but it has no influence on development of the software.

Getting Plasma

Desktop environments are tricky things to package, and fixed release distributions generally stick with whatever version of Plasma and friends the distro shipped with, providing only security updates and bug fixes. This makes sense, as there's a reasonable chance that pulling the rug out from your desktop and trying to slide a new one in to place without anyone noticing is tricky and there's potential for all kinds of breakage.

On Ubuntu, the easiest way to install Plasma is to install the corresponding *kubuntu-desktop* metapackage. Besides the desktop, this gives you some of the core KDE applications too. There are two other metapackages: `kde-full`, which gives you the entire *KDE Applications* suite, and `kde-plasma-desktop`, which will give you just the desktop and no applications.

If you're using stock Ubuntu 16.04 LTS, then installing these packages will give you Plasma 5.5.5, which was released in March 2016, just after we did our last Plasma feature. If you're on 17.04, then you can do slightly better: the repos provides 5.9 which dates back to the beginning of this year. Plasma 5.11 is scheduled for release right around the time you read this. The Kubuntu Backports PPA is the place to go to get fresh Plasma on Ubuntu, but remember these packages aren't tested to the standard of those in the main repo. Add it at your own peril with:

```
$ sudo apt-add-repository ppa:kubuntu-ppa/backports
```

Other distros provide similar mechanisms. OpenSUSE has repos for both Leap and Tumbleweed that provide daily builds of the required components, but these are called 'unstable' for a reason. Slackware users can get help with Alien BOB's repository. If you want a distro that provides newer KDE packages without risk, then Arch Linux will serve your needs. It provides modular packages, which enables more in-depth customisation of your KDE install. Start with the `plasma` group or `plasma-meta` package. Add the `kdebase` group. Then choose from the `kdeedu`, `kdegames`, `kdegraphics` groups and so on. Of course, Arch Linux isn't for everybody, and its easier-to-use descendents (Manjaro or Antergos, say) inherit these fresh packages. Other distros have up-to-date repos, too: Fedora tracks the latest Plasma updates, and they can be had with `dnf install @kde-desktop`. Then there are dedicated KDE distros such as Chakra and KaOS.

Neon lights

Probably the best showcase of the latest and greatest KDE technologies, though, is KDE neon (see **https://neon.kde.org**), the 'not-quite distro' based on Ubuntu 16.04. It can be used as a live disc without risk, or in a VM, but with all the graphical wizardry will look and perform best when installed on bare metal. Being an Ubuntu derivative, there's very little (*bwahaha–Ed*) chance installing it will do any unrequested damage. Neon is available as a user edition, or a choice of stable and unstable developer editions (in order of increased likelihood of things misbehaving). Out of the box, few applications are installed besides the KDE core ones, but all the latest KDE offerings are available by a PPA. You'll also find the growing in usefulness Discover app store.

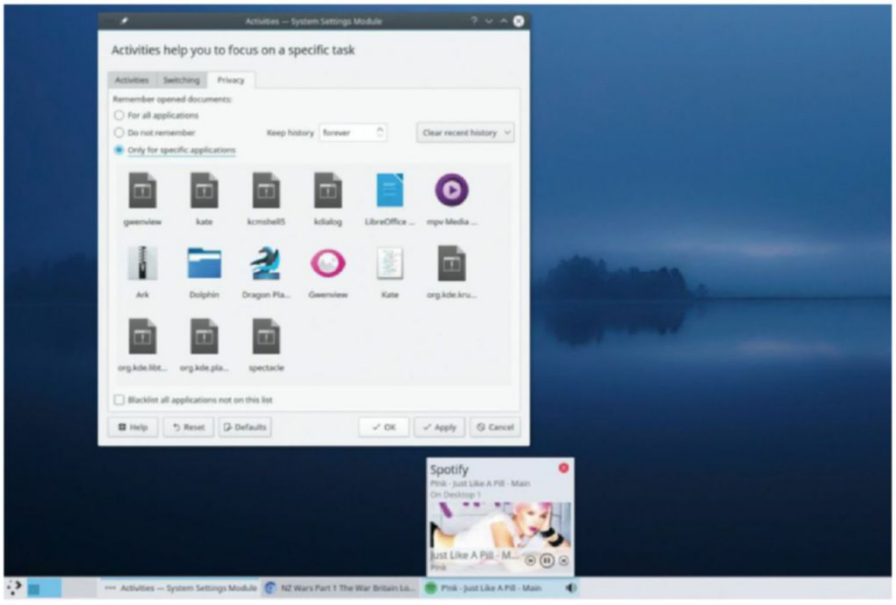
Note that nothing outside of the KDE ecosystem is packaged in Neon. That doesn't mean those things aren't available, it just means they're available in the versions packaged for Ubuntu 16.04. So you can end up with very new KDE applications alongside somewhat older ones from outside the fold. Of course, you can then add further PPAs to get newer versions, but this may introduce incompatibilities, or at any rate bring about undesired or unexpected quirkiness.

Plasma 5 makes heavy use of hardware acceleration for rendering, and offers a choice of rendering backends (accessible via the Compositor settings). Everything is rendered as an OpenGL (or OpenGL ES if the hardware supports it) scenegraph, so if suitable hardware is available all that fading and transparency doesn't tax the CPU at all. The converse of this is that if suitable hardware isn't available, then things will be slow. All the effects can be turned off via the desktop effects dialog, but we still wouldn't recommend running KDE with graphics hardware that predates the Obama (*those were the days–Ed*) era.

Resource-wise, it's not at all as bad as it used to have a reputation for; Gnome has definitely taken the memory hog crown. Our own experiments and anecdotal reports show Plasma 5 uses around 400MB to start a clean session. But such stats are pretty useless in an era when opening a couple of tabs in a web browser consumes a couple of gigabytes! LXF

KDE's mascot, Konqi the dragon, has been around since 1999. A new Krita-designed incarnation was prepared for Plasma 5.

Image credit: KDE CC BY-SA 4.0



The live window previews with player controls: great for skipping embarrassing music.

Future

techradar pro

IT INSIGHTS FOR BUSINESS

THE ULTIMATE DESTINATION FOR BUSINESS TECHNOLOGY ADVICE

- Up-to-the-minute tech business news
- In-depth hardware and software reviews
- Analysis of the key issues affecting your business

www.techradarpro.com

twitter.com/techradarpro facebook.com/techradar

Future

GIZMODO | UK

Not your average technology website

EXPLORE NEW WORLDS OF TECHNOLOGY GADGETS, SCIENCE, DESIGN AND MORE

- Fascinating reports from the bleeding edge of tech
- Innovations, culture and geek culture explored
- Join the UK's leading online tech community

www.gizmodo.co.uk

twitter.com/GizmodoUK facebook.com/GizmodoUK

LXFHotPicks

Alexander Tolstoy
smells the divine scent of lamb shashlik from his backyard barbecue and plans to show you some delicious open source trimmings.

» gImageReader » Notepadqq » QupZilla » Green-recorder » Qmmp » YouTube-DL » Torrent File Editor » 0 A.D. » Dolphin Island 2 » Fontforge » NanoTTS

OCR software

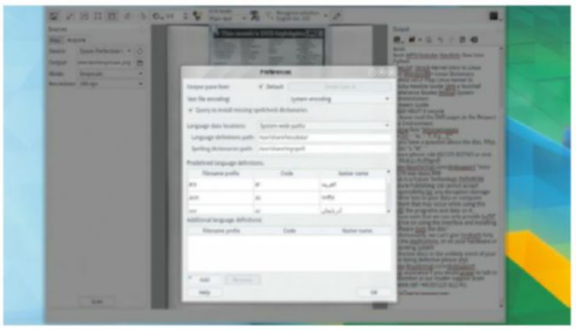
gImageReader

Version: 3.2.3 **Web:** https://github.com/manisandro

Accounting, formal office paperwork, library services and, of course, maintaining your own digital archive of historic documents and publications – this is just a short list of applications where optical character recognition (OCR) is welcome. The idea of extracting text from a scanned bitmap image became popular with the rise of home flatbed scanners in the 1990s (ancient times in computing terms), particularly thanks to the commercial *Abbyy Finereader* software.

In Linux, we have an analogue to *Finereader,* known as *Tesseract.* This is a community effort to bring professional-quality OCR to Linux, and we must admit, it works just fine. The hero of this review is a graphical front-end to *Tesseract,* which allows everyone to scan and extract text data from any paper document.

gImageReader is a sleek and easy-to-use application that enables you to escape having to deal with *Tesseract* via the command line. Don't get confused by that initial 'g' – it simply means 'graphical', and depending on your desktop of choice, you may want to use either the GTK3 or *Qt5* version of *gImageReader*, which are both supported officially.



Check predefined language definitions to make sure that *gImageReader* will work correctly.

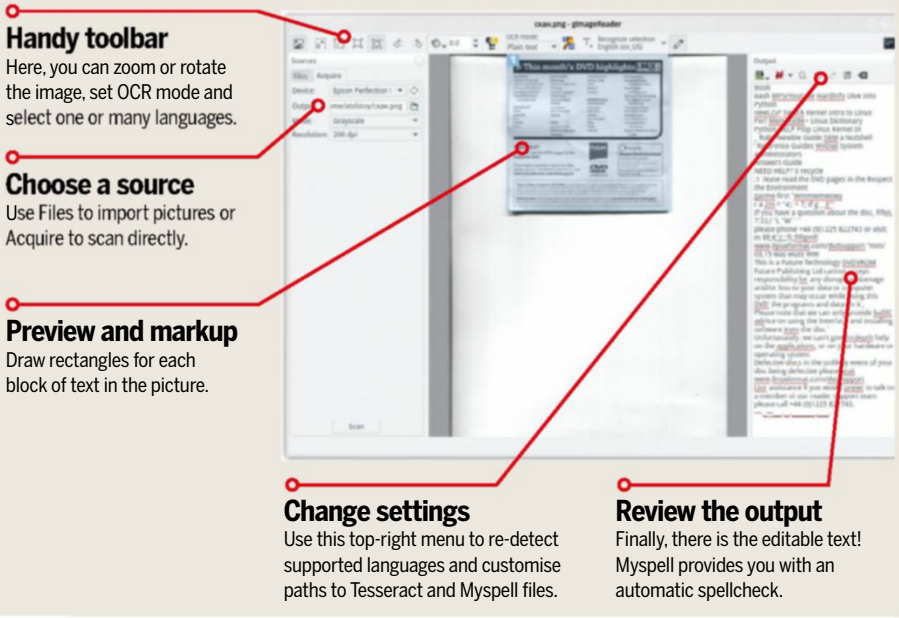
> "A community effort to bring professional-quality OCR to Linux"

The application doesn't have too many controls and configurables, thus is quite friendly to newcomers. You can import bitmap files or scan directly from *gImageReader*, if you have a physical scanning device. Remarkably, *gImageReader* distinguishes real scanners from the list of available V4L devices – so, unlike many other multimedia apps in Linux, this one ignores your webcam and shows only genuine scanners.

In order for the recognition engine to work correctly for your language, you must make sure you've installed the appropriate language packages for *Tesseract,* otherwise *gImageReader* produces iffy results. Luckily, *Tesseract* supports over 100 languages and writing systems, so you just need to check your package manager and install the required parts.

The results are editable text that you can copy and paste to any other application, such as *LibreOffice Writer, Scribus* and so on.

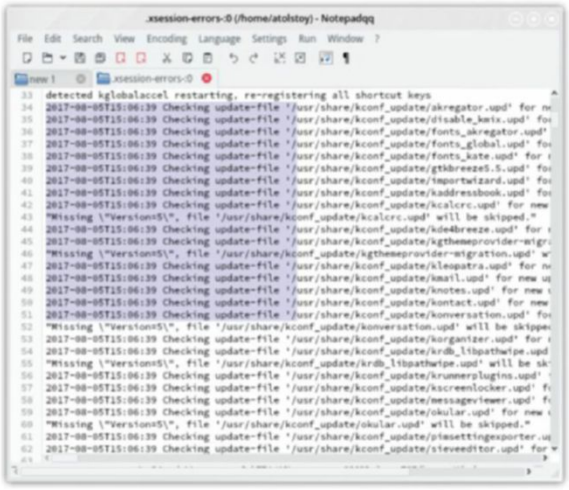
Exploring the gImageReader interface...



Text editor

Notepadqq

Version: 1.0.1 **Web:** https://github.com/notepadqq



The free selection tool for text is simply awesome!

People who write code are generally more demanding about the features of their text editor of choice, but that doesn't mean that other mortals don't care. Despite the fact that there's a plethora of text editors available, and we've reviewed the best of them, here we go with yet another. *Notepadqq's* distinctiveness lies in it mimicking the world-famous *Notepad++*. The latter is an open source, yet Windows-only, program, and is extremely popular among power users. *Notepadqq* isn't a direct port of *Notepad++*, but a third-party project that tries to implement the same features in a fully platform-independent text editor.

Naturally, there isn't a 100% match between the two, but *Notepadqq* comes very close to its counterpart. It supports code highlighting for more than 100 languages, search as you type, document re-encoding, multiple input, line sorting, line breaks and many other familiar features, all in a friendly tabbed interface.

Notepadqq is a very welcome tool for developers, thanks to its other cool features, such as column select mode (hold Ctrl+Alt and draw a selection), regular expression searches and real-time highlighting. As long as *Notepadqq* is a smaller project, it can't boast a massive number of extensions, but it still has the official connector to Node.js modules. You can write your own extension using the dedicated NPM module that provides the *Notepadqq* API, like this:

```
$ npm install notepadqq-api
```

When you have your own Node.js script or even NPM package, provide its path in Settings > Preferences > Extensions, and it should work right away.

> **"Even if the file is removed, Notepadqq still restores its contents"**

Notepadqq also follows the *Notepad++* style when it comes to managing unsaved files. When you quit the application, it doesn't ask you to save your changes, and it just quits. However, the next time you open *Notepadqq*, it gracefully restores everything. Even if the file is removed, *Notepadqq* still restores its contents – a very helpful feature!

Web browser

QupZilla

Version: 2.2 **Web:** www.qupzilla.com

***QupZilla* can restore your previous session after you accidentally close the browser.**

The *QupZilla* project is a source of inspiring news. After the previous review in **LXF212**, we decided to revisit this software, due to the introduction of the session manager – a breathtaking feature that brings this underdog *Qt5*-based browser closer to the big players. The session manager depends on the quite recent *Qt5.9.2* release (or newer), and in case your package manager shows older versions of both *QupZilla* and *Qt5*, you'll need to build it from source. If you've ever dealt with compiling applications that use the Cmake building system, you'll have no trouble with *QupZilla*. The only thing you need is time, because the browser relies on QtWebengine, a *Chromium*-based engine, which is rather large and takes an hour or two to build.

Other news came from this year's Akademy, where KDE devs traditionally meet and share their vision. As of now, the decision has been made to officially include *QupZilla* in the KDE family and also rename the browser. Regardless of the new name, the browser is now publicly recognised as a successor of the legacy *Konqueror* web browser that used to be one of KDE's defaults. Changing the name is a good idea, because it will stop false allusions to *Mozilla* products, and *QupZilla* has never used the *Mozilla's* Gecko engine in the past.

The new status of the *QupZilla* project is not going to introduce massive changes. For instance, the devs promise not to include KF5 dependencies, and to keep the browser fully desktop-agnostic. The changes in the new 2.2 version include many fixes and minor enhancements, such as a fixed application icon in the Pulseaudio volume control.

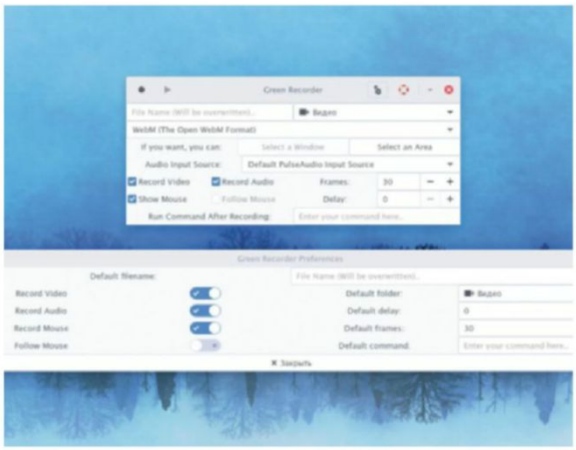
> **"The decision has been made to officially include QupZilla in the KDE family"**

Currently, *QupZilla* is perhaps the best way to use a *Chromium*-based browser without messing with Google services, so we think you should definitely give it a try.

Screen recorder

Green-recorder

Version: 3.0.4 **Web:** https://github.com/foss-project



Click the round Record button to make this window hide and begin capturing your desktop.

We're not sure whether Canonical is brave enough to boot the upcoming Ubuntu 17.10 to *Wayland* by default (looks like it is), but other major distros, such as Fedora, ship Gnome desktops that silently boot to a *Wayland* session instead of *X.org*. We really love *Wayland* as a cool next-gen technology, but it's still not mature enough in some regards. One of its shortcomings was its inability to record a screencast, which now seems to be fixed. Meet *Green-recorder*, a small, stylish GTK3 app that's ideal for recording videos of what's happening on your desktop, and the first recorder that works in a *Wayland* session out of the box.

Green-recorder is based on Python, GTK3 and *FFmpeg*, and thus supports many video formats, including MKV, AVI, MP4, WMV, GIF and more. When used with *Wayland*, *Green-recorder* can only write video using the WebM format and compress the data using the V8 encoder, instead of the more recent V9. That's the price you pay for *Wayland* right now, but we think it's not too high. *Green-recorder* captures video data using the built-in screencast feature in Gnome Shell and records audio using *FFmpeg*, then merges both tracks into a single WebM file. The application also produces very nice GIF outputs. We compared GIF recordings made using pure *FFmpeg* and *Green-recorder*, and admitted that the latter was far smoother – that's because *Green-recorder* makes GIFs out of the raw uncompressed video data.

Although the *Green-recorder* interface is very simple, it gives you all the necessary features right at your fingertips, and enables you to set up frame rate, destination directory, file format and audio input source, and even execute custom commands after the recording is finished.

> **"The first recorder that works in a Wayland session out of the box"**

The *Green-recorder* developer advertises pre-built packages for various Linux distributions, so in most cases, you only need to worry about copying and pasting the necessary lines from the **Readme.md** document.

Multimedia player

Qmmp

Version: 1.1.8 **Web:** http://qmmp.ylsoftware.com

A dozen equalizer presets are usefully already included within the jukebox.

This one is a nostalgic choice for anyone who remembers *Winamp* and its 'llamas' welcome sound. However, in every other regard, *Qmmp* is a modern multimedia player that continues to receive frequent updates, enhancements and fixes. We think it attracts a noticeable number of people who want to have a simple player with a playlist, and all the frequently used controls fit inside its small main window. *Qmmp* looks very similar to *Winamp*, and given the fact that it supports all of the uncountable numbers of WSZ skins ever created for *Winamp 2.x*, you can make it look identical to the prototype.

Qmmp was made with the idea that you may want to organise your music with a file manager and browse albums as folders, without keeping a separate library (which is commonly a SQLite database in most other players). Having said that, all you need to do is drag a selection of files or folders on to the player's window and watch them be added to the playlist. By default, the *Qmmp* window consists of three parts: the player itself, the equalizer and the playlist. All three stick to each other like magnets, but you can still detach the equalizer and playlist, rearrange them or keep them completely separate.

The player is a versatile multimedia machine, thanks to the bundled set of plugins. Most desktop environments associate *Qmmp* with audio files, whereas the player can open video files just fine. For that, *Qmmp* uses *FFMpeg* and *Mplayer* plugins, but it also supports *mpg123*, gme, Wasapi and more. For instance, using gme (which stands for 'game'), you can play games such as *Super Mario* for Game Boy right inside the *Qmmp* player. Looks as though we have an application for everything, which means you can get rid of other media players – if you like the approach and style of *Qmmp*. The player should already be available in your package manager, so no need to deal with sources (unless you need bleeding edge code).

> **"Organise your music with a file manager and browse albums as folders"**

Video downloader

YouTube-DL

Version: 2017.08.13 **Web:** https://github.com/rg3

The Google-owned YouTube website is indisputably the world's favourite video-hosting service and a popular source of entertainment for many people. Sometimes, though, you might want to take your favourite videos with you when you're out and about, or simply download and store them locally. This way, you don't have to depend on an internet connection, and you can also can get rid of those annoying advertisements that attack you online.

YouTube-DL is a dedicated command-line utility for extracting YouTube videos and storing them in your filesystem. The application uses the official YouTube public API for querying video details, getting the list of available quality options and obtaining download links. The simplest command syntax looks like this:

```
$ youtube-dl https://www.youtube.com/watch?v=XXXX
```

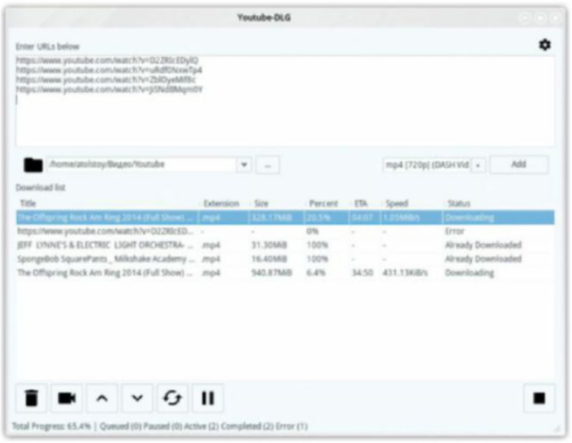
To get the necessary link, simply copy it from your web browser's address bar and provide to *YouTube-DL*. By default, you'll probably get your video in moderate or low quality, such as 360p. For better results, ask *YouTube-DL* to query what's available for your video:

```
$ youtube-dl -F https://www.youtube.com/watch?v=XXXX
```

You'll see the numbered list of quality and file format presets, usually populated with various combinations of 3GP, Webm and MP4, together with different video resolutions. Find the desired variant and pass its number to the command. For example:

```
$ youtube-dl -f 137 https://www.youtube.com/watch?v=XXXX
```

> "A dedicated command-line utility for extracting YouTube videos"



Paste in URLs and then hit Add to include them in the download queue.

If you want to escape the command line routine, take a look at the *YouTube-DL* GUI project at **https://github.com/MrS0m30n3/youtube-dl-gui**. This is a simple graphical front-end to *YouTube-DL*, written in Python and wxWidgets. A new version 0.4 was recently released, which has a cleaner design and lets you easily put many YouTube downloads in a queue.

Torrent editor

Torrent File Editor

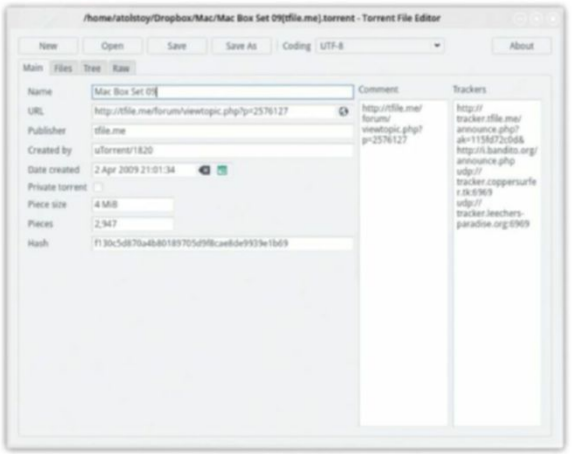
Version: 0.3.6 **Web:** https://github.com/drizt

Strangely, there is a false assumption that 'the torrent file format makes adding or removing files impossible without access to the original files' (found somewhere on the web), although there's a decent number of utilities and even online services that prove the opposite. Here, we're playing with a program with a self-descriptive name: *Torrent File Editor.*

This is a very helpful tool for anyone who doesn't just download large files using the BitTorrent technology, but also wants to modify the contents of their .torrent files. There can be various reasons for that, but usually people want to add or change the list of trackers (remove obsolete ones, add more speedy servers) and also add or remove certain files within the torrent share. If you've already downloaded the files from other peers and have the torrent contents stored locally on your hard drive, it's possible to manipulate the .torrent file in any way – add new content, for example. If you only have the .torrent file, not the data, you can only change a limited number of options, such as exclude (but not add) files from the list.

Torrent File Editor is an almost perfect tool for mastering your own torrents and a perfect addition to the existing features of popular BitTorrent clients, such as *Deluge, qBittorrent, Transmission* and others. The main interface divides all capabilities into four tabs: Main, Files, Tree and Raw. Here you can change almost any aspect of your existing .torrent file, such as the size and number of pieces, publisher's details and various custom options that you can add directly to the raw markup code of the .torrent file. A splendid utility for torrent lovers!

> "Change almost any aspect of your existing .torrent files"



Open, change what you want, then Save As – easy as ABC!

Get it from the project's GitHub page and compile using the standard sequence that applies to any CMake-wrapped source tree:

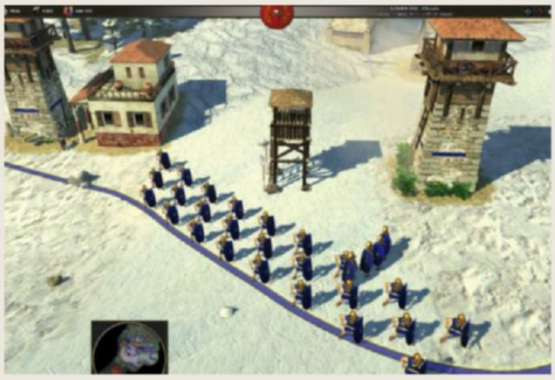
```
$ cmake . && make
```

HotGames Entertainment apps

Strategy game

0 A.D.

Version: Alpha 22 **Web:** http://play0ad.com



Our swordsmen prepare to defeat the enemy dictator.

It's been a while since we last admired the beautiful *0 A.D.* strategy game. Every Alpha ## release brings a massive set of updates, so the current Alpha 22 has hundreds of new features compared to the Alpha 17 from **LXF193**.

You instantly notice the improved graphics, including textures, model animation and weather elements, and crisper, more delightful music tracks. During play, you can enjoy other novelties, such as the new 'capture the relic' mode, designed for multiplayer. The relic is a wagon containing the sacred remains of a great leader; it boosts construction and gives various discounts, and therefore is priceless.

The best way to get started is with a single campaign, playing against an AI bot. By default, the game chooses a map and the civilisation you play for, though you can set everything manually. The goal is to develop a sustainable economy, with workers and warriors, and to find the right balance between civil and military development. You need more people to collect food, wood and stone, while simultaneously preparing for a fight, because as soon as your AI counterpart is ready, it attacks.

The graphics are stunning and invite you to zoom in to examine every pixel of clothing, waving flags, dust under moving horsemen and other beauties. It's worth playing as different nations to explore their distinctive features, as well as some of the new additions — for example, Ptolemians now have infantry champion pikemen, Romans have the new Temple of Vesta and Persians benefit from the Gate of Ishtar.

> "The graphics are stunning and invite you to zoom in"

At over 600MB, *0 A.D.* is packaged for many distros and takes little time to get installed.

Arcade game

Dolphin Island 2

Version: GIT **Web:** https://github.com/lavaduderDev

A colourful and picturesque retro-style game about a Japanese schoolgirl.

A while ago, we reviewed an unusual game, where you had to go fishing on a small boat and sail around a pond. Unlike *Mouse Boat* from **LXF209**, *Dolphin Island 2* features more familiar side-scrolling arcade gameplay, but both games were made using the same Godot engine. Godot-based games usually feature graphics with intentionally large pixels, reminiscent of old video games from the 1980s, and *Dolphin Island 2* is no exception.

Despite the name, it's not a sequel, but a standalone platformer with a fictional story behind it. It was created during an indie game contest a while back, in which developers had to write a story based on faked cover art by a third-party artist. That Japanese girl with a sword is Aisha, who dived into a game world to rescue her classmates, Sora and Momo. The titular Dolphin Island has been seized by the Devil King, who has sent various evil creatures, such as worms and red-eyed black cats, to stop Aisha in her tracks. She can't beat off certain threats, such as fireballs that cats throw at her, or falling icicles, but she is quick and can be gone before more sluggish enemies hit her. Usually, it takes two to four hits of her sword to defeat an enemy, but if they manage to wound Aisha, she needs extra time for her health to restore. The game has a nice intro level, which explains the controls, then leads you to further levels with different enemies and built-in dialogue, all accompanied by stylish background music.

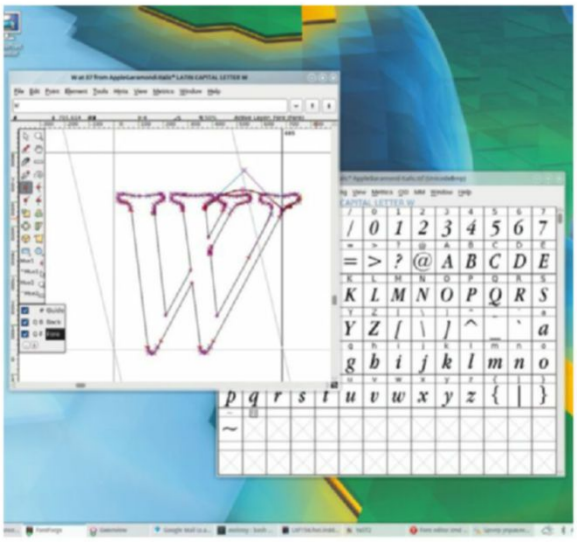
> "Aisha dived into the game world to rescue her classmates"

The game is obviously heavily inspired by Japanese culture, particularly anime, and it will be a pleasant time-killer for an hour or two, provided you're a fan of this type of game. To run *Dolphin Island 2*, you naturally need the Godot engine first. This provides a graphical interface for the list of Godot projects, so the only thing you need to do is open the game's project file and run it.

Font editor

Fontforge

Version: 20170730 **Web:** https://github.com/fontforge



Each glyph is the result of painstaking work by a designer.

A stalwart of the Linux graphics scene and the only capable open source font editor, *Fontforge* has a niche application and is targeted at graphic artists and font designers, although anyone else might find it useful as a convenient way of converting one font type to another. It can eat up a font in Adobe format and spit out TrueType, if that's what you need, and it understands a variety of formats not supported easily on Linux.

Fontforge is also a very good starting point for creating your own font. It offers rather sophisticated vector editing tools, so you can create your OpenType or TrueType font solely within *Fontforge,* but that isn't the only way to get the job done. For instance, you can design characters using *Inkscape* and then export them as a series of SVG files, which in turn can be pasted into *Fontforge.* But the main complexity of building a font is not in the shape of the letters you can design, but in the spacing between them. Typeface designers spend a tremendous amount of time fine-tuning the gaps between different pairs of glyphs (characters). They also need to bear in mind hinting – a set of custom rules that let a typeface scale up and down and remain crisp-looking.

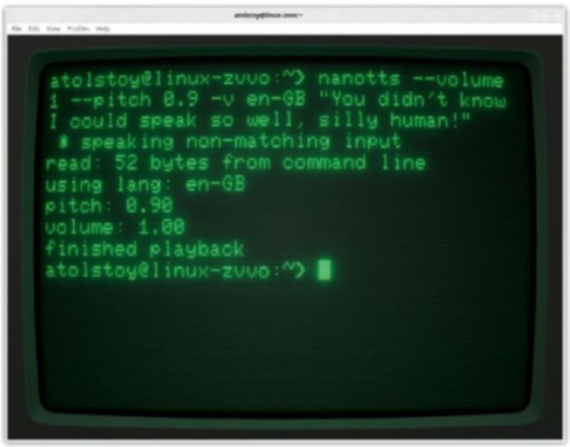
The best way to give *Fontforge* a try is to import an existing font and see how it looks like from inside. The main window shows you a table of available glyphs, each in an individual cell. Double-click a glyph to open it in a separate editing window. You'll see a toolbar along the left side of the window, with a plethora of vector editing tools. You can add, delete and move nodes, draw Bezier and Spiro curves, scale or flip parts of the glyph, conduct precise measurements and more. This can be a captivating experience, and if you feel you're ready to go more in-depth with *Fontforge,* don't miss its super-useful official guide, called 'Design with Fontforge', on the project's website.

> **"You can create your OpenType or TrueType font solely within Fontforge"**

Text to speech engine

NanoTTS

Version: GIT **Web:** https://github.com/gmn/nanotts



Enjoy the good-quality TTS engine with lots of options and its tiny footprint!

This pick was born out of an attempt to find a decent text-to-speech (TTS) synthesizer for Linux. You'd be perfectly correct to remind us about *Festival* and *eSpeak,* but the first of those was last updated three years ago, whereas the second provides poor speech quality. Perhaps there's a way to get decent computer-generated speech from software with a small footprint? To solve this puzzle, we had to combine a couple of projects into one solution. The first one is a Linux port of Android's SVOX TTS engine, also known as PicoTTS (**https://github.com/aserranoh/picotts**). We needed to compile this to have the command-line *pico2wave* utility, which could convert text into speech and write the result into a WAV file, like this:

```
$ pico2wave -w test.wav "Let me tell you something"
```

After that, we were ready to go with *NanoTTS,* which is described as the "improved SVOX PicoTTS speech synthesizer". This application provides a more versatile usage of the original PicoTTS engine. For instance, with *NanoTTS,* we can choose between different voices, and also change volume, speed and pitch for each voice. At the moment, *NanoTTS* provides British English and American English voices, and other voices for French, Italian, German and Spanish languages. It's hard to imagine how the whole bonanza fits into the ridiculously small **./nanotts** executable, which is about 1.3MB in size, but it works like a charm. The sound quality is not perfect (it's a little dull), but it's a cut above the funny robotic sound of *eSpeak.*

> **"Sound quality is a cut above the funny robotic sound of eSpeak"**

You can do various other tricks with *NanoTTS* after studying its command-line arguments. For example, you can vocalise a text snippet directly to MP3 using something like this:

```
$ echo "I'll tell one more thing" | ./nanotts -c | lame -r -s 16 --bitwidth 16 --signed --little-endian -m m -b 32 -h - out.mp3.
```

Have fun with this smart little app!

FOR THE GNU GENERATION

BUY YOUR ISSUE TODAY

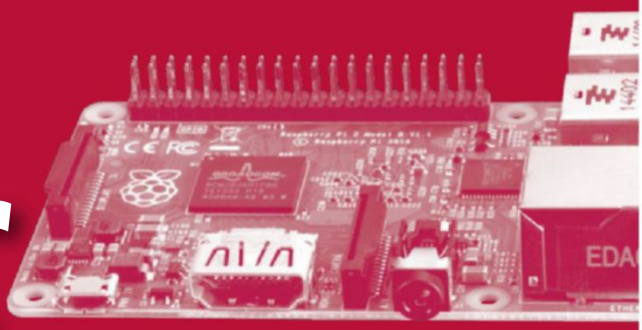
Available from all good newsagents & supermarkets

www.myfavouritemagazines.co.uk/sublud

*Prices and savings are compared to buying full priced print issues. You will receive 13 issues in a year. You can write to us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14 day cancellation period unless exceptional circumstances apply. Your statutory rights are not affected. Prices correct at point of print and subject to change. For full terms and conditions please visit: bit.ly/magtandc **Offer ends 31 October 2017.**

LINUX FORMAT

Pi user



Giving you your fill of delicious Raspberry Pi news, reviews and tutorials.

TANYA FISH is Pimoroni's chief laser operator and maker of flashy tech.

Welcome...

Making random things has always been in my nature, from various Heath Robinson-type machines in my youth to lab mash-ups at university. My parents owned a BBC Model B, and I spent my Saturdays typing in lines of code to make my own games.

Skip forward a few years where I dabbled in ticket fraud analysis, learned to fly, and qualified as a maths teacher. I later went on to teach physics, science and art, using zombie movies and race cars, and then ended up in my spiritual home, operating the lasers at Pimoroni. This opened me up to a whole new world of tinkering, and before I knew it I was attending maker events, and meeting a wonderful bunch of people.

The Raspberry Pi has enabled me to bring a whole new dimension to my silly inventions, and the sheer knowledge and friendship the community shares is inspiring.

I've been lucky to be given the opportunity to pass on my enthusiasm for physical computing, and I regularly get to spread the joy that is automating your art. Kids make tickle machines, they drive remote control cars they've programmed, they make disco machines. The Raspberry Pi enables that kind of inspiring interaction, even from the first blinking LED, that a glossy GUI never can. The feeling of "I made this" is such a kick.

On the way, I've picked up Python, got better at soldering, and been able to make all sorts of ridiculous objects, mostly made of cardboard. In fact, I'm off to give some workshops in a field while wearing my "big fish little fish" rave sleeve. See ya!

Code Club reaches one in five schools

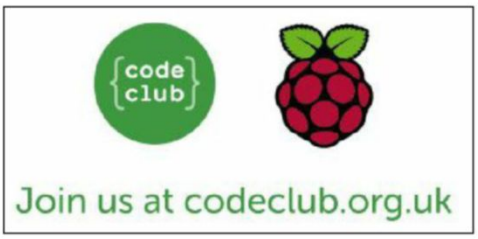
Filling young minds with code, those delightful monsters!

The Raspberry Pi Foundation has announced it's expanding Code Club to reach secondary school ages up to 13. This widens the original target age range of Code Club, which was from 9 to 11. To accommodate the broader range, new projects are available and Code Club will continue to create more resources to build on the support offered to this age group.

Before the announcement a pilot group of 50 UK secondary schools were enlisted back in May to discover how support for them could best be implemented and how Code Club could work for children aged 12 and 13.

The expanded age range helps Code Club respond to the huge demand and currently one in five UK state-sector secondary schools have registered with the programme and almost all 600 are running Code Clubs.

Code Club has gone from strength to strength and we wish it great success. More at **https://blog.codeclub.org.uk/2017/09/05/9-to-13**.



Body scanner

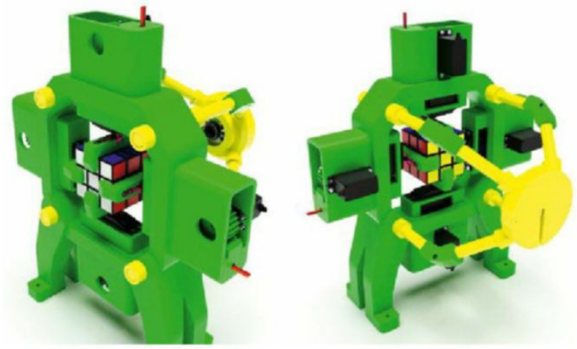
3D models made easy.

Twenty seven Pi Zeros and Pi cameras help make a perfect high-end, high-def 3D scanner. Fashion designer Poppy Mosbacher wanted a way to create 3D models of fashion models, so armed with a 3D printer and grant award from Brighton University she fashioned this custom scanning tent. Find out more at **http://bit.ly/LXF229pi3d**.

Rubik's solver

3D printed, Pi powered.

Behold the perfect storm of Pi, 3D printing and the Rubik's Cube. It's a 70-hour build that's all 3D printed, and amazingly no soldering is required – but you will need $200 worth of kit. Okay, so it's not the fastest solver we've seen, but it is the best Pi-power, 3D printed one around. Full details and software can be found at **www.otvinta.com/download12.html**.



Clever Card Kit

Les Pounder tries out English company Monk's RFID kit, with hopes that he can automate the dog opening the door. Now we just need a robot for walkies!

In brief...

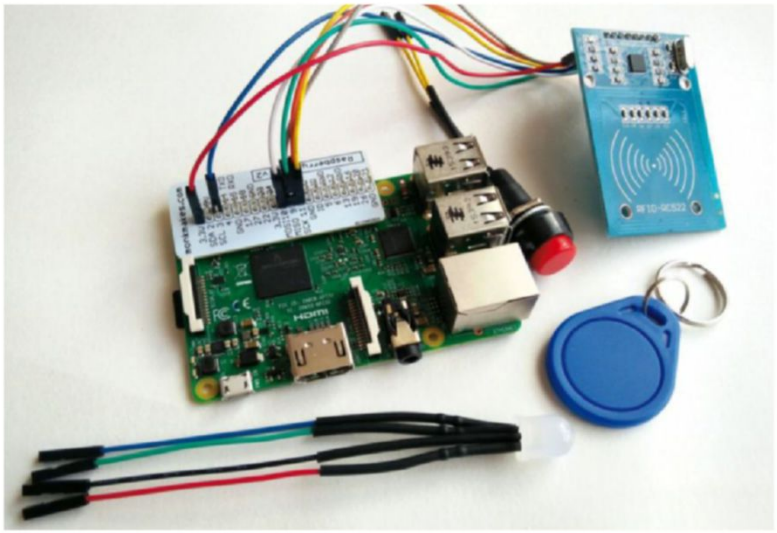
A self-contained kit for all models of Raspberry Pi that offers an introduction to RFID and how it can be used to trigger events. As well as the components to cover six projects there's a well-written guidebook that covers the accompanying Python 3 library, and explains how the kit can be used after the projects have been completed.

For new users, the sheer volume of projects and components is bewildering. Where do you start when you know nothing about electronics or the Raspberry Pi? Well, typically you purchase a kit from one of the many businesses that have popped up to support the Pi community.

Simon Monk, author and electronics tinkerer, has a range of Monk Makes kits. The latest is an RFID card kit for all models of Raspberry Pi. RFID (Radio Frequency Identification) is often used with key fobs to track employee movements around a building by touching readers located at doorways.

Inside the kit is the RFID reader that connects to the Raspberry Pi using the GPIO utilising the SPI protocol. There's also a series of cards and key fobs that can be read and written to, a momentary switch, and an RGB LED for use in projects. For those new to the Raspberry Pi, the GPIO numbering system can seem confusing, but presented with the kit is a "Raspberry Leaf" that fits over the GPIO and provides a quick reference for the GPIO Broadcom pin numbering.

By following the well-written guidebook we were able to quickly build our test project, which checked that the RFID reader was working correctly. This is the first in a series of six that form the majority of the guidebook. Each project came with a series of images and diagrams to explain how to connect the components, and was accompanied by code segments, which can also be downloaded from the website. All of the code for a project is fully explained and clearly presented in the guidebook.



Coming as a full kit, the Clever Card Kit offers six projects, fully supported and explained in a great guidebook. Hats off to the Monk team.

The projects vary from the simple (reading the contents of an RFID card) to the advanced (using RFID cards instead of cash in a game of Monopoly). But what links the projects is how practical they are. All six projects can be used as the basis of another project, and by including these projects in another we can remix and reinvent, which is a common project trait.

The next level

Once you've mastered the six projects, the guidebook provides a reference on using the SimpleMFRC522 library that enables the RFID reader to be used with the Pi. To test how easy it was, we used Project 2 to write a website URL to an RFID fob. We then created a simple application that read the URL from the fob and opened the default web browser to that URL. This was accomplished in just eight lines of Python. Speaking of Python, it's great to see that the projects and libraries are using Python 3.

This is a great kit that moves away from obvious topics for most starter kits. By containing all of the parts, the software and configuration steps as part of a well-paced and written guidebook, this kit will provide an exceptionally interesting routeway for those coming to the Raspberry Pi.

For those au fait with the Raspberry Pi, this is still a great kit. Using RFID readers with the Pi can be problematic, but following the guidance of this kit we were able to have a working project in under 30 minutes and the Python 3 library is easily usable. One minor nag about the library is that it needs to be in the same directory as the project code. This can be solved by copying the **SimpleMFRC522.py** file to **/usr/local/lib/python3.4/dist-packages/**, but it would be great if that were handled automatically by the installer.

But despite that, this is a great kit to use and it'll inspire a lot of makers! LXF

Features at a glance

Guidebook and parts
Having all of the code, guidance and parts in one kit enables you to quickly make great projects!

RFID-RC522
The RFID reader/writer enables the price to be kept low, while offering a great introduction.

LINUX FORMAT Verdict

Clever Card Kit

Developer: Monk Makes
Web: www.monkmakes.com/cck
Price: £15

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

A "go to" kit for those starting out with RFID. Easy to use, well supported and with a great Python 3 library, too.

Rating 9/10

Flask: Build physical setups

Les Pounder introduces a Python library to create and interact with physical, real-world electronics using a handy web interface.

Our expert

Les Pounder works with organisations such as Raspberry Pi Foundation and micro:bit Foundation to promote maker skills. His blog is over at **bigl.es**.

COMPATIBILITY ALL Pis

You need

- Any Pi
- A breadboard
- One LED
- 220 Ohm resistor (RED-RED-BROWN-GOLD)
- Five male-to-female Dupont connectors
- A DC motor
- An L298N motor controller
- 4x AA battery pack and batteries
- Code: **https://github.com/lesp/LXF229-Flask/archive/master.zip**

Flask is a powerful web server framework for Python and we're going to use it to control devices connected to a Raspberry Pi via the GPIO Zero library. By turning on an LED using a button on a web page, we introduce how that can be scaled up to do anything with Python and a Raspberry Pi. We also introduce the Bootstrap HTML framework, as used by Twitter, and use it to create a web interface for our controls.

First we'll connect an LED, inserted into a breadboard. The second part is to connect a motor to the Pi via a controller. We can't directly connect the motor because it would damage the GPIO pins. See the diagram for clarification.

To install Flask, open a terminal and type the following to update our repositories and then install Flask:

```
$ sudo apt update && sudo apt install python3-flask
```

We need to find and note the IP address of our Pi:

```
$ hostname -I
```

The project needs two directories; the first is **Flask-Demo**

```
$ mkdir Flask-Demo
```

The second is our templates directory, which contains the HTML of our projects. This is inside the **Flask-Demo** directory and we can create that using the following command:

```
$ mkdir Flask-Demo/templates
```

The software for our project is split into two sections. The Python code that will enable us to control the LED and motor and the HTML that is used to create the user interface.

Create the correct code

We'll start with the Python code. For this open *Thonny*, found in the Main Menu > Programming. Before we write any code, save your Python project (File > Save) in the **Flask-Demo** directory. Call the project **app.py**. Remember to save often.

We start the code by importing three libraries. Flask and its render_template class, GPIO Zero's LED and Motor classes and from Time we import sleep, which is used to control the pace of our project.

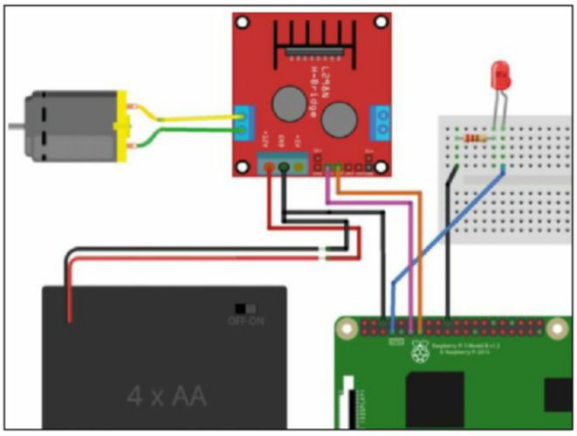
```
from flask import Flask, render_template
from gpiozero import LED, Motor
from time import sleep
```

Next, we create two objects to establish a connection between our code, and the LED and motor that's attached to the Raspberry Pi. The motor object has two parameters: the pins used to control the forwards and backwards movement.

```
led = LED(4)
motor = Motor(forward=17, backward=27)
```

Our next step is to create an app, which is how Flask refers to our project.

```
app = Flask(__name__)
```



Two separate circuits: from the Pi to an LED, and the connection from a DC motor via an L298N controller board.

Flask uses "routes". These are paths that are requested by the user when they either type the path into the address bar, or click a hyperlink that takes them to a certain path.

The first route is our root, identified by /. Here we use render_template to use the **index.html** file, which we shall later create. We create a function called index that will be called when the user visits the web page.

```
@app.route('/')
def index():
    return render_template('index.html')
```

We create another route that handles turning the LED on and off using GPIO Zero's "toggle" function. So when the user visits <IP ADDRESS>/toggle/ the LED will turn on/off as necessary. We also use the same render_template to return the user to the control page.

```
@app.route('/toggle/')
def on():
    led.toggle()
    return render_template('index.html')
```

Create two more routes: these will control moving the motor. The code for backwards is the same as forwards, just change 'forwards' or 'backwards' as applicable.

```
@app.route('/motor-forwards/')
def motorforwards():
    motor.forward()
    sleep(1)
    motor.stop()
    return render_template('index.html')
```

The background on Bootstrap

Bootstrap has been with us for a few years now. It was originally created by one of the developers at Twitter as an internal framework and style guide for internal tooling. After being heavily developed thanks to a hack week, the project went from its internal name Twitter Blueprint to Bootstrap. It's a responsive, mobile first framework that concentrates on providing a consistent experience no matter what device.

In our project we used Bootstrap to write the HTML that made up the interface for our controller. Rather than have either locally written or stored CSS to decorate our project, we used an absolute link to the centrally stored Bootstrap Content Delivery Network (CDN), which is updated by the team. This means that we get the latest version every time the page loads. However, it does follow that we need a constant internet connection. We also do this for two JavaScript frameworks: one from Bootstrap, the other a JQuery framework.

You can learn more about Bootstrap, and how it can be integrated into your next web project, by visiting the website at **http://getbootstrap.com**. There are a series of Getting Started guides to follow, and you can use the team's examples to form the bones of your project.

Finally, we instruct the Flask app to run. It will write a log to the shell and will accept connections from any IP address as we set the host to 0.0.0.0:

```
if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0')
```

Save your work and but don't close *Thonny*. We have completed the Python code for this project, so now let's move to the code for the HTML interface.

Dip into the HTML

The HTML template is how we provide a user interface for controlling the Flask web server. To edit the HTML we'll need to use *Geany*, also in the Programming menu. With the *Geany* window open, click File > Save and navigate to **/home/pi/Flask-Demo/templates**. Then save your file as **index.html**.

We'll be writing HTML using the Bootstrap framework. We start the HTML by instructing the web browser that we're writing HTML.

```
<!doctype html>
<html>
<head>
<title>Web Controller</title>
```

Next, we import a Bootstrap CSS stylesheet, used to decorate our page using pre-defined elements such as buttons and grids. We also import two JavaScript libraries to provide interactive elements on the page:

```
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
```

We tell the browser to use the full screen space available:

```
<meta name="viewport" content="width=device-width, user-scalable=no" />
</head>
```

On to the body! This is the on-screen portion of the HTML. We start by creating a row to store content. This is wrapped in a `div` : an element to divide up the screen into sections. Our `div` will create a row with one column where we'll place an image, in this case the circuit diagram for this project. This will be placed centrally, and it'll be a quarter of the screen size.

```
<div class="row">
  <div class="col-sm-12"><img class="center-block" src="https://raw.githubusercontent.com/lesp/LXF229-Flask/master/Images/Circuit-Diagram_bb.png" height=25% width=25%>
 </div>
</div>
```

Now let's create the controls for our project. We're going to create a container and in there we shall repeat the same `div` class "row", but this times we shall create two extra columns that are used to place our controls in the centre of the screen. We start by creating the first column:

```
<div class="container">
 <div class="row">
  <div class="col-sm-4" style="background-color:white;"></div>
```

Now we create another column, but in there we place elements such as a title `<h2>` , and some paragraph text `<p>`

```
  <div class="col-sm-4" style="background-color:white;">
  <h2>Web Controller v 0.3</h2>
  <p>Click on the buttons to trigger the action</p>
```

We add a new div which will be our group of buttons:

```
  <div class="btn-group">
```

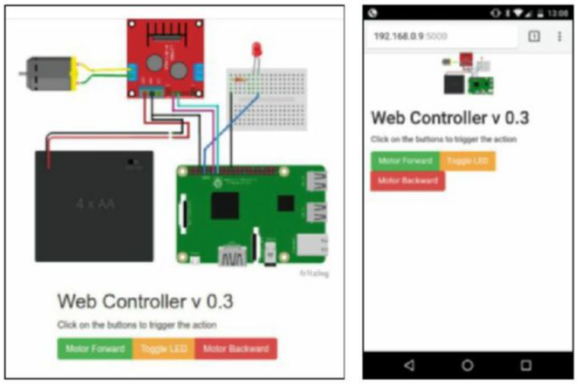
Our group of buttons is made up of three controls, Motor Forward, Toggle LED and Motor Backward. Each button launches a hyperlink – the routes that we set in our Python code. You can see the `href` for each refers to the routes that we created. Each button also has a class – success, warning and danger – and this colours the buttons accordingly.

```
   <a href="/motor-forwards/" data-toggle="tooltip" title="Make the motor spin!" class="btn btn-success">Motor Forward</a>
   <a href="/toggle/" data-toggle="tooltip" title="Toggle the LED on or off!"  class="btn btn-warning">Toggle LED</a>
   <a href="/motor-backwards/" data-toggle="tooltip" title="Send the motor into reverse!"  class="btn btn-danger">Motor Backward</a>
  </div>
```

We now create a final column, the same as the first, then close all of the `div` elements that we've created, before we close the body and the HTML document. Save your work and close the application. In *Thonny*, click the Play button to run the Python code. On another device, connected to the same network, visit the IP address of your Pi, followed by :5000, for example ours was.

```
192.168.0.6:5000
```

You will see the interface appear on the screen, so now you can press the buttons and make things come to life! LXF



The web interface is simple and clean. It responds to the user's devices making the best use of the screen, thanks to Bootstrap's basic yet capable framework.

Quick tip

GPIO Zero has an extensive number of classes and functions that enable working with electronics components. Motor controllers, analog device, sensors and LEDs are all easier to work with thanks to this beginner-friendly library. The team have great documentation, which can be found at **https://gpiozero.readthedocs.io/en/stable/**.

WolfenPi: FPS action in WWII

Nate Drake takes out waves of goose-stepping goons from the Third Reich in Wolfenstein 3D, the precursor to the ground-breaking game Doom.

Our expert

Nate Drake
Nate used to top up his income by offering a file recovery service using *scalpel*. His first customer was a burlesque dancer. It didn't end well...

Quick tip

Wolfenstein 3D also supports gamepads like PS3 controllers. See **https://goo.gl/VHLTyd** for more information.

Secret Agent, Captain William "BJ" Joseph Blazkowicz's commendable efforts to halt the Nazi war machine come to a crashing halt at the start of id software's *Wolfenstein 3D*. Having successfully thwarted an evil scientist's plans to create a race of mutant-Nazi hybrids, BJ has been captured and consigned to the dungeons of the Nazi stronghold Castle Wolfenstein. After quickly overpowering a hapless guard, you must now navigate BJ through the six episodes of the game, consisting of ten levels each, shooting evil henchmen and dogs while collecting looted treasure on the way.

Note that this guide focuses on the shareware version of Wolfenstein 3D, which only contains the first ten levels.

While using Nazis as bad guys is nothing new, when *Wolfenstein 3D* was first released in 1992, even its developers couldn't have known the staggering level of success it would enjoy. At its heart, the game is a first-person shooter using 3D graphics in the style of *Doom*, which id software would release a few years later. The game is notable for its use of ray casting: a technique whereby only surfaces bigger to the player were calculated, resulting in much smoother gameplay. This, of course was an issue much more relevant to users of early 90s PCs, but it's also important to your gaming experience on the Raspberry Pi.

Chocolate Wolfenstein

Aside from being both a showcase for the first-person shooter and shareware, *Wolfenstein 3D* was also notable in that its developers licensed the game engine to other companies. This resulted in the game being ported to various other platforms besides the original MS-DOS, such as Mac OS, the Acorn Archimedes and SNES. In 1995, the source code for the *Wolfenstein 3D* game engine was released, meaning fans were free to create their own versions.

In this guide we'll be focusing on running Fabien Sanglard's formidable Chocolate Wolfenstein engine. This is a slightly improved version of the original 'vanilla' engine, which is nevertheless designed to mimic the original game as closely as possible.

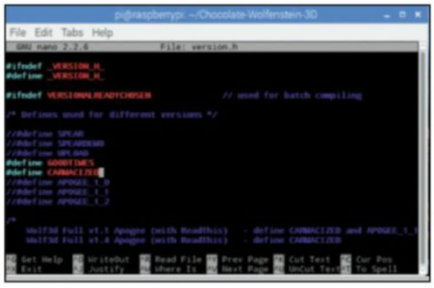
Will the 'real' Wolfenstein please stand up

If you install the Steam version of *Wolfenstein 3D* for Windows, navigate to **C:\Program Files (x86)\Steam\steamapps\common\Wolfenstein 3D\base** and copy all nine files with the extension .wl6 on to a USB stick, then insert it into the Pi. The files and extensions must all be in lower case to be compatible. Open Terminal and navigate to the USB stick, for example `cd /media/pi/USB1`. Convert the files to lower case with `for i in $(find . -type f -name "*[A-Z]*"); do mv "$i" "$(echo $i | tr A-Z a-z)"; done`. Finally move them to the correct folder with `mv *.wl6 /home/pi/Chocolate-Wolfenstein-3D`.

Next, run the command `sudo nano /home/pi/Chocolate-Wolfenstein-3D/version.h`. This file determines which version of *Wolfenstein* you want to compile. Find the section marked `/* Defines used for different versions */` and make sure that there's a `//` at the start of every line in that section except those containing the words GOODTIMES and CARMACIZED.

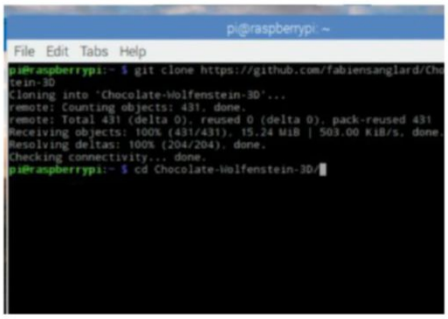
Press Ctrl+X, then Y, then return to save and exit. Follow the steps (*above right*) to compile the full version of *Wolfenstein 3D*.

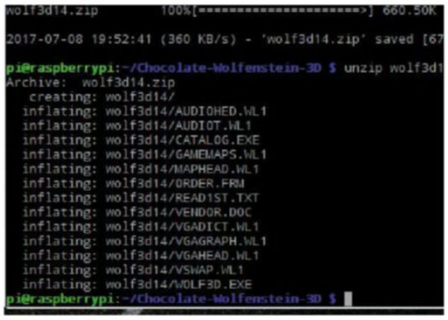
If you've purchased the prequel to *Wolfenstein 3D*, called *Spear of Destiny*, the data files are located in **C:\Program Files (x86)\Steam\steamapps\common\Spear of Destiny\base** and have the extension .SOD. Change these to lower case and move them into the **Chocolate-Wolfenstein-3D** folder as outlined above. Run `sudo nano /home/pi/Chocolate-Wolfenstein-3D/version.h` and make sure there's a `//` at the start of each line in the same section except those containing the words CARMACIZED, SPEAR and GOODTIMES.



Version.h identifies the iteration of Wolfenstein 3D. You must compile the game each time you want to play a new version.

Wolfenstein 3D on your Pi





1 Install preqrequisites

Open *Terminal* on your Pi and run `sudo apt-get install cmake libsdl1.2-dev libsdl-mixer1.2-dev libsdl-net1.2-dev libbz2-dev libjpeg-dev libgtk2.0-dev`. Next run `git clone https://github.com/fabiensanglard/Chocolate-Wolfenstein-3D` to download the Chocolate Wolfenstein engine. Move to the directory with `cd Chocolate-Wolfenstein-3D`. Compile the code by running `sudo make`. This will create an executable: *Chocolate-Wolfenstein-3D*.

2 Download data files

Next, run `wget http://maniacsvault.net/ecwolf/files/shareware/wolf3d14.zip`, then `unzip wolf3d14.zip` to download and then unzip the shareware version's files. Change over to the directory using `cd wolf3d14`. Next, run `for i in $(find . -type f -name "*[A-Z]*"); do mv "$i" "$(echo $i | tr A-Z a-z)"; done`, then relocate the data files to the game directory with `mv *.wl1 /home/pi/Chocolate-Wolfenstein-3D`.

3 Launch Wolfenstein 3D

Use `cd ..` to switch to the main game directory. The shareware version of *Wolfenstein 3D* can be launched any time by entering `./Chocolate-Wolfenstein-3D`. Add the parameter `--res 640 480` or similar to adjust the screen resolution. See **https://github.com/fabiensanglard/Chocolate-Wolfenstein-3D/blob/master/docs/README.Wolf4SDL.txt#L81** for a full list of command line parameters.

Although the engine itself is open source and can be compiled on your Pi, the game data files which contain details of maps, levels, enemies and so on are still under copyright. For this reason, this guide will focus on running the shareware version of *Wolfenstein 3D* on your Raspberry Pi, which contains only the first episode, Escape from Castle Wolfenstein. This will still give you ten levels of high octane, chaingun-smoking action, but if you want to play the full version then you can purchase the game and copy the data files over to your Pi if you wish. The Chocolate Wolfenstein engine also supports playing the prequel to *Wolfenstein 3D*, called *Spear of Destiny* (*see the boxout, Will the 'real' Wolfenstein please stand up, below left*).

Although technically you can compile and run Chocolate Wolfenstein on any model of Pi, we recommend using a Raspberry Pi 3 for best performance. The tutorial assumes you have a clean install of Raspbian on your SD card and that you have run `sudo apt-get update` and `sudo apt-get upgrade` before proceeding.

Digging for victory

Because the Chocolate Wolfenstein engine only requires you to run the `make` command and place the data files in the same folder as the executable, there's very little that can go awry with this project. Data files for different iterations of *Wolfenstein* games have different extensions, such as the shareware version of *Wolfenstein 3D* (.wl1), the full version (.wl6) and *Spear of Destiny* (.sod). If you see an error message when trying to launch the game that it can't find the right data files, the most likely reason is because all data filenames and extensions must be in lower case – this process is initially done in the walkthrough above.

The Chocolate Wolfenstein 3D engine also has to be compiled differently, depending on the version of the game that you're using. You can change this in the file **version.h** in the game directory.

If you don't know how to get the data files for the full version of the game, consider either buying the Steam Windows version of *Wolfenstein 3D* (**http://store.steampowered.com/app/2270/Wolfenstein_3D**), which is available for around £4 for Windows machines. See the boxout (*left*) for help with finding the data files for installation. Steam also sells a Windows version of *Spear of Destiny* for around £3. (**http://store.steampowered.com/app/9000/Spear_of_Destiny**). Non-Windows users may be able to find a second-hand version of the game CD online.

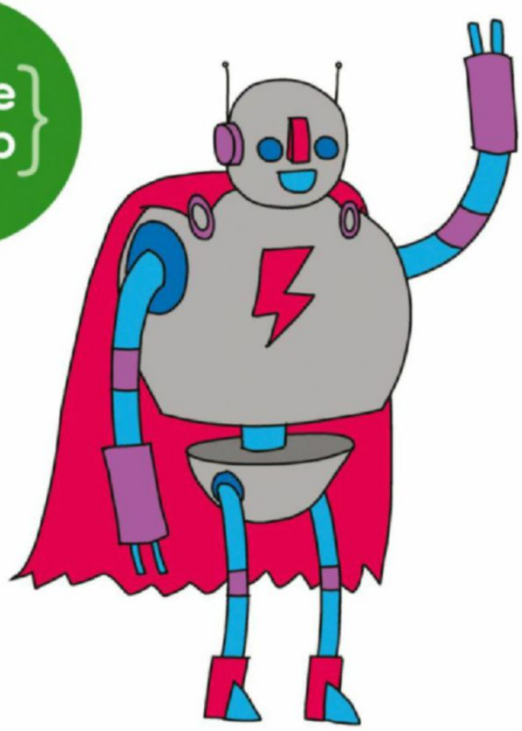
If you enjoy the premise of gunning down Axis troops but aren't happy with the clunky graphics, you'll be pleased to know that the game spawned the free and open source Linux game *Wolfenstein: Enemy Territory* (**www.splashdamage.com/content/download-wolfenstein-enemy-territory**). The game gives you the choice of fighting on the side of either the Axis or Allied powers, playing as part of a multiplayer team with your friends. LXF

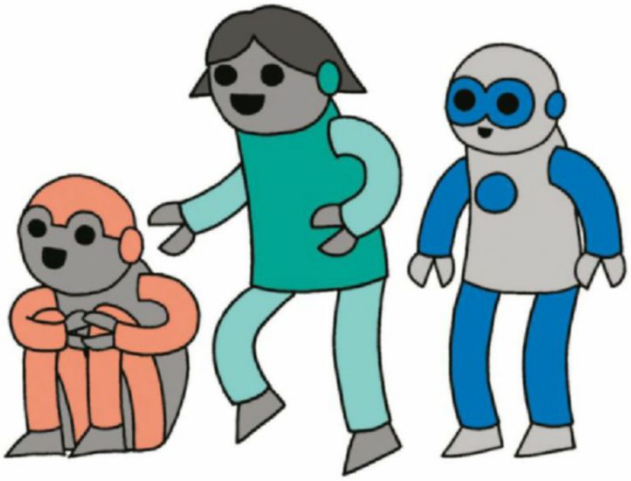
Quick tip

The *Wolfenstein 3D Shareware* data files can also be found on the DVD (**wolf-demo-data.zip**). Extract, then place them inside the same folder as the **Chocolate-Wolfenstein-3D** executable.

Wolfenstein 3D was banned from sale in Germany due to its use of Nazi symbols. The SNES version removed these as well as the gory effects.

code club





Can you help inspire the next generation of coders?

Code Club is a nationwide network of volunteer-led after school clubs for children aged 9-11.

We're always looking for people with coding skills to volunteer to run a club at their local primary school, library or community centre for an hour a week.

You can team up with colleagues, a teacher will be there to support you and we provide all the materials you'll need to help get children excited about digital making.

There are loads of ways to get involved!

So to find out more, join us at **www.codeclub.org.uk**

LINUX FORMAT Back issues » Missed one?

Issue 228
September 2017
Product code:
LXFDB0228

In the magazine
We fill a laptop with the best educational FOSS and explore hardware projects to make you top of the class! Why did the Ubuntu Phone fail? We find out... Plus expert sync tools, Audacity, Scalpel recovery and more!

LXFDVD highlights
Fedora 26 (both 64- and 32-bits of it), and Mageia 6.0 XFCE.

Issue 227
Summer 2017
Product code:
LXFDB0227

In the magazine
Everything is virtual nothing is real, at least it is where we live. We test business servers, jump to the ZFS, pit BSD Vs Linux, chat to Canonical about Edge Computing, use Blockchains for fun and encrypt with LUKS.

LXFDVD highlights
Debian 9.0.1 Stretch and Voyager Live 9.0.

Issue 226
August 2017
Product code:
LXFDB0226

In the magazine
Give Windows the elbow and enter the world of Linux – our in-depth guide shows you how. We review animation programs, assess VPN services and charge around the deadly levels of Chocolate DOOM!

LXFDVD highlights
Linux Starter Kit: Zorin OS 12.1, Manjaro 17, Elementary OS 0.4.1

Issue 225
July 2017
Product code:
LXFDB0225

In the magazine
Hone your security skills with pro advice on how to shore up your network's weak points. We also test a slew of animation tools, talk to Katrina Owen, the creator of exercism.io, use the Pi Zero kit to build a robot, and plenty more!

LXFDVD highlights
Ubuntu Desktop 17.04 remix, plus Solus 2017 and Android x86

Issue 224
June 2017
Product code:
LXFDB0224

In the magazine
Hang on to your hats – the latest version of Ubuntu is out and we've got the lowdown on what to expect from Zesty Zapus. Plus, we show how easy it is to build a router and firewall, and create a smart calendar out of Pi.

LXFDVD highlights
Ubuntu 17.04 all 64- and 32-bits of it, Linux Lite 3.4, and more!

Issue 223
May 2017
Product code:
LXFDB0223

In the magazine
The greatest Pi ever? Of course! We reveal the new Pi Zero W and how you can build better devices, AMD Ryzen gets reviewed, Roundup of CAD packages, create Android apps and we talk Open democracy (sob).

LXFDVD highlights
Ubuntu Studio 16.04, openSUSE Tumbleweed and XenialDog 2017.

To order, visit myfavouritemagazines.co.uk
Select Computer from the all Magazines list and then select Linux Format.

Or call the back issues hotline on 0344 848 2852 or +44 344 848 2852 for overseas orders.

Quote the issue code shown above and have your credit or debit card details ready

Future

GET OUR DIGITAL EDITION!

SUBSCRIBE TODAY AND GET 2 FREE ISSUES*

Available on your device now

Available on the App Store
Available at amazon
GET THE MAGAZINE ON Google play
nook
zinio™

*Free trial not available on Zinio.

Dirb: Creating bookmarks

Shashank Sharma shows you how Bash ninjas like himself create bookmarks to make navigating directories quick and easy, using dirb.

Our expert

Shashank Sharma is a trial lawyer in Delhi and avid Arch Linux user. He's always on the hunt for geeky memorabilia.

Features like history expansion, tab-complete and reverse lookup make working with Bash easy and fun. Yet no amount of tab-completion can help you when you frequently have to navigate between deep-nested directories. While it wouldn't be a bother if you had to access these directories only once or twice, if you have to concurrently work on multiple directory trees, then having several instances of the terminal application open isn't a practical solution, either.

Dirb enables you to create bookmarks to nested directories that you routinely access, and then navigate to them with only a few key strokes, instead of typing in the complete path every time. The tool utilises Bash shell functions to bring the bookmarks functionality, commonly associated with browsers, to the command line. Although the original program has since been lost, the tool is still available thanks to its permissive license, albeit as a fork on Git.

First, clone the **dirb.sh** file from the project's Git page on to your disk (click the green Clone or Download button on the project's Git page at **https://github.com/icyfork/dirb**). Now extract the contents of the downloaded zip file, which contains the **dirb.sh** file. Next, enable *dirb* to work with Bash by informing the **Bashrc** file of the exact path to the downloaded **dirb.sh** script. You can do this by adding `source /path/to/dirb.sh` to your **~/.bashrc** file using your usual text editor.

Dirb ships with a handful of commands to help you minimise the keystrokes when navigating directories:

s..... save a directory bookmark
g..... go to a saved directory bookmark
r..... remove a saved directory bookmark
d..... display bookmarked directory path
sl..... print a list of directory bookmarks

The `g` command works the same as `cd`. In fact, the `g` command has been written to replace the in-built `cd`. This is because users shouldn't have to use both `cd` and `g` to explore directories. While you can't use the `cd` command to navigate to a saved bookmark, you can use `g` in place of `cd`. For example, the command `g /etc` works just like `cd /etc`.

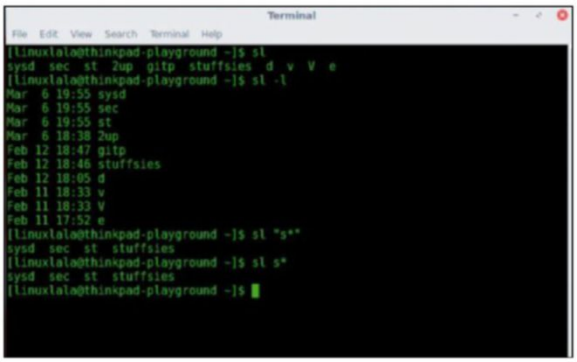
Quick tip

If the default function names conflict with any aliases already on your system, you can change the offending functions by editing the **dirb.sh** file to something more suitable.

Creating bookmarks

You can use any key to create a bookmark, except the special function keys such as F1-F12, Ctrl, Alt, Shift, and so on. As a rule, if pressing the key prints something on the screen, it can be used as a bookmark.

If you want to create a bookmark for the **/run/media/linuxlala/Stuffsies/Vbox-Machines/** directory, switch to the directory on the terminal and then run the `s V` command. You can now use the `g V` command to switch to this directory irrespective of your current path.



You'll learn the usefulness of the sl command once you get started with bookmarking your most-used directories.

However, there are only so many bookmarks you can create using a combination of letters and numbers. Thankfully, *dirb*, much like the command line, is case sensitive and so the letters v and V can be used to bookmark different directories. However, taking this approach may lead to confusion in the long term.

You can also combine letters and numbers to create even more bookmarks. For instance, if the letter d is used to create a bookmark for the **~/Downloads** directory, you can use `dc` to bookmark the **~/Documents** folder.

Because `g` has been designed to completely replace the `cd` command, a tricky situation may arise when you attempt to navigate to a directory that has the same name as a saved bookmark. For instance, below there's a bookmark d for **~/Downloads** and also a directory named d in **~/**.

```
1. $ mkdir d
2. $ cd Downloads/
3. $ s d
4. $ cd ..
5. $ g d
6. $ pwd
7. /home/linuxlala/Downloads
8. $ cd ..
9. $ g ./d
10. $ pwd
11. /home/linuxlala/d
```

The lines above have been numbered to help you follow along the example. In line 5, the command `g d` is used to

Under the hood

Bookmarks that have been created with *dirb* are unique to each user, because such bookmarks are stored in the **~/.DirB** directory. This directory contains a corresponding file for all the user-created bookmarks and each of these files contains a single command.

The file for the 2up bookmark that we created contains the following line:

```
$ cd .DirB
$ ls
2up  d  e  gitp  sec  st  stuffsies  sysd  v  V
$ cat 2up
$CD  "../../"
$ cat V
$CD "/run/media/linuxlala/Stuffsies/VBox-Machines"
```

As you can see, the `cat` command displays the content of the **2up** file, which is the `$CD "../../"` command. The shell variable $CD is set by the `g` command to `cd` and the quoted variable is expanded by Bash whenever the bookmark is invoked. Essentially, the command `g 2up` is translated into `cd ../../`. You can also refer to the **DirB.sh** script file for more usage examples of each of the *dirb* commands.

The **DirB.sh** script file is broken into sections for the various commands such as `g`, `s`, `d` and so on. Each code block also features helpful and pointed comments before every if and else loop to describe the objective of the code.

navigate to the **~/Downloads** directory. If, however, you had used the `cd d` command then you'd have been taken to the **~/d** directory. Because there's a saved bookmark with the same name as a directory, you must use the `g ./d` as shown in line 9 in the previous example. You can similarly use the full stop (.) with the `g` command when you want to switch to a sub-directory that has the same name as a previously saved bookmark.

You also don't need to switch to a directory before creating a bookmark for it. The `s` command accepts the bookmark name and path as arguments with the `s <bookmark name> <path>` syntax as demonstrated in the following example:

```
$ s gitp /run/media/linuxlala/Stuffsies/git-projects/
$ g gitp
$ pwd
/run/media/linuxlala/Stuffsies/git-projects
```

With *dirb*, you can also create a bookmark for a relative path instead of specifying the path to an exact directory. One of the most common directory changes one must do when working with the terminal is the `cd ../../` command. The following example demonstrates the creation and working of a bookmark for this operation:

```
$ cd /home/linuxlala/Documents/project/files
$ g 2up
$ pwd
 /home/linuxlala/Documents
```

Irrespective of your current path, when you now run the `g 2up` command, you'll move two directories up, relative to your current path. Because the 2up bookmark isn't anchored to any specific directory, you can run the `g 2up` command from within any directory with the same result – moving two directories up. Depending on your usage, you can similarly create bookmarks for frequent relative path movements that you make during your Bash excursions.

Removing bookmarks

Use the `sl` command for a list of all the saved bookmarks:

```
$ sl
2up  gitp  stuffsies  d  v  V  e
```

You can also run the `sl -l` command, which displays the time and date when the bookmarks were last used:

```
$ sl -l
Mar  6 18:38 2up
Feb 12 18:47 gitp
Feb 12 18:46 stuffsies
Feb 12 18:05 d
Feb 11 18:33 v
Feb 11 18:33 V
Feb 11 17:52 e
```

Naturally, your bookmark list will grow with time and so you can use regular expressions with the `sl` command to narrow down your search for bookmarks:

```
$ sl -l "s*"
Mar  6 19:55 sysd
Mar  6 19:55 sec
Mar  6 19:55 st
Mar  6 18:46 stuffsies
```

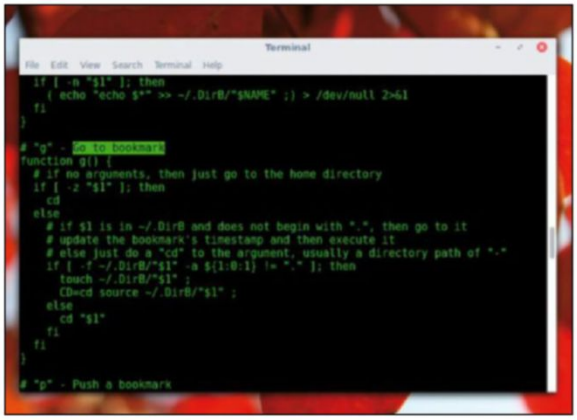
The above example lists all the bookmarks beginning with s. You must remember to enclose the regular expression in quotes. The timestamps are updated only when the *dirb* commands `s` and `g` are used to access a bookmarked directory. So if you were to access the directories with the `cd` command, the timestamp wouldn't be updated.

Chances are that you might not remember the path of a saved bookmark if you haven't accessed in a long time. When this happens, you can run the `d <bookmark name>` command to view the path of the saved bookmark:

```
$ d V
/run/media/linuxlala/Stuffsies/VBox-Machines
```

To maintain good housekeeping, you should regularly prune the bookmarks and remove ones that are no longer needed. The `r` command, designed to remove bookmarks that aren't required, accepts a bookmark name as the argument. The command `r 2up` will remove the 2up bookmark we created. You can now recycle the freed-up bookmark for some other directory or path.

Despite being simplistic, *dirb* is a highly useful tool that can be an asset for home users and sysadmins alike. What's more, it doesn't require any time to setup or master and makes working with Bash even more convenient. LXF



If nothing else the DirB script can help you master the art of variables and the if/else loops in Bash scripting.

Snaps: Package for freshness

Mats Tage Axelsson shows you how the methods for distributing applications for Linux are changing – and why we should care.

Our expert

Mats Tage Axelsson
After years of fiddling with Linux, Mats has moved on to educate others on how to screw up and recover all parts of your systems.

Linux users often argue with users of other operating systems about how brilliant Linux is. Yet one thing that's always been hard to defend is how tricky it is to download your favourite application directly from the developer's website and run it. Usually, it's a deliberate choice to have a central repository with well-tested and secure software. However, for desktop applications, users prefer to find the developer's website and download from there.

Until lately, you could only choose to use a distribution or compile your own. Most regular users won't compile their software packages. For the developers, maintaining a package for all the distributions available is a nightmare. They need to adapt to the libraries in each distribution, including which version they need to support. If they're using a new feature in one of their libraries, they may not be able to convince distribution maintainers to add it. This situation especially applies to applications with a small user base.

Ubuntu has introduced snaps as one of many solutions to obtain application packages that aren't distro specific and are also available as downloads. We'll cover snaps and appimage in this tutorial, and mention Flatpack and a few others.

WhatsAppImage?

An Appimage contains the executable you want, but that's not the first executable that runs when you start an Appimage. The first binary to run is AppRun, which sets up the system for the application.

The appimagekit creates AppRun during the process of making the AppImage. The resulting AppRun executable is very similar in different appimages. After all, it only sets the system up with variables and libraries. You're also free to create the AppRun yourself, but you'd probably only do this after you've used the appimagekit a few times.

A good idea for your first few projects is to write your recipe when you create an appimage. The recipes are YAML files, one for each AppImage. The file contains all the information to create the AppImage, including a script to download and configure the package.

A Snap has a similar structure, but with a stricter specification. The package contains a local root **/**, **/meta** and **/bin/**. The local root contains the source and wrapper files, **/bin** contains binaries and **/meta** contains configuration files.

The difference between snaps and AppImages is that snap needs a daemon to operate, while an Appimage doesn't. When you download a snap, you have to install it before you can use it. That is one task that the snap daemon performs.

With an AppImage, you must have AppRun in the root and **/usr** directory of the image created. It's not necessary to integrate your application into your system. If you do want it to show up in your favourite desktop environment, then you can choose to use the optional appimage daemon. In practice, you must follow the standards laid down at **www.freedesktop.org** for your applications to run correctly.

The big difference between the two packaging formats is that AppImage uses an ISO 9660 file, and a Snap uses a squashfs filesystem. Furthermore, Snaps and AppImage come from different lines of thought. The most important priority of the AppImage project is to make it easy to distribute and use software directly from the developer. In other words, the user and developer shouldn't depend on the distribution maintainers to move over to newer versions.

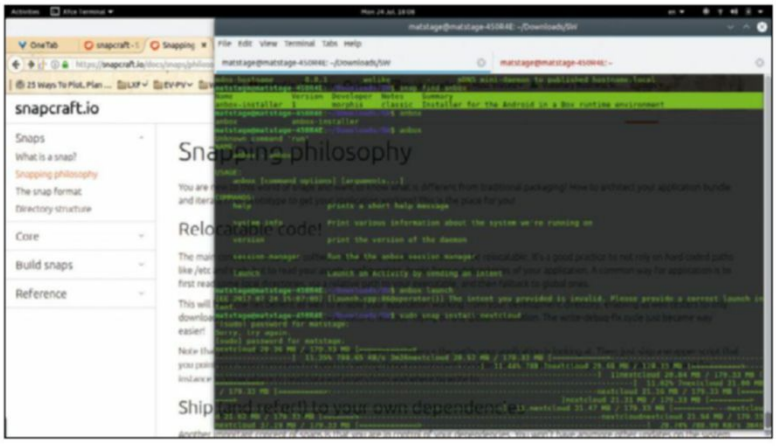
There's a greater emphasis on correct confinement in the snap design. The system also requires a central repository, called stores. And so when you want to use Snaps, you first need to install the Snaps system for your distro. The snap system is a traditional package for many distributions, so in Ubuntu you use `apt`:

```
$ sudo apt install snap
```

In Fedora you use `dnf`.

```
$ sudo dnf install snap
```

Fedora started the Flatpack project, which is competing to solve the same problem. So you need to go through a lengthier procedure before you can install snap. In other distributions, you need to check if they've added the snap packages. It's already native in Arch, Debian, Fedora and the main Ubuntu distributions. The most common problem is missing support for the confinement, and you need to run your snaps in development mode.



To learn more about making snaps, turn to www.snapcraft.io. The site has all the formal specifications and many tutorials to help you in your quest.

Consider online build services

If you want to build a package, either your own or a favourite application, then you can use a build service. An online build services is a good way to save yourself from weighing your system down with compiling jobs. Compiling code is, in itself, not particularly heavy on the system, but there are other advantages of using a build service. For instance, you can set it up so that the system compiles the code as soon as it's updated. You then have the option to have the application updated on each nightly build.

The latest addition to this group is Open Build Service, from SUSE. This service is built to manage the OpenSUSE distribution in different forms, but it can also use repositories, such as git, to build AppImages.

The technique is simple if you have your appimage.yml file ready. You only need to perform three steps before you start the compiling. If you set it up correctly, then it also builds all new versions as soon as they exist.

You do have the option to use Travis CI, but then you'll need to combine your git repository with docker manually and set up build scripts in your docker instance. While Travis CI is the older and more mature, the Open Build service gives you the direct route to AppImages.

Once snap is installed the process is very similar to using a package manager. Below is a typical flow for getting a snap.

We start by finding the snap package.

```
$ snap find nextcloud
Name Version Developer Notes Summary
Name Version Developer Notes Summary
nextcloud-port8080 1.01 arcticslyfox - Nextcloud Server
nextcloud-nextant 11.0.0snap3 rmescandon - Nextcloud Server + search support
nextcloud 11.0.3snap7 nextcloud - Nextcloud Server
cashbox-nextcloud 11.0.2snap2 cashbox - Nextcloud Server for www.cashBOX.plus
spreedme 0.29.5snap1 nextcloud - Spreed.ME audio/video calls and conferences feature for the Nextcloud Snap
qownnotes 17.07.6 pbek - Plain-text file notepad with markdown support and ownCloud integration
solr 0.1 rmescandon - Starts up solr as forking daemon
mdns-hostname 0.0.1 welike - mDNS mini-daemon to published hostname.local
```

As you can see, the command shows all versions available along with related snaps, too.

Once you've verified that it's available, install it.

```
$ sudo snap install nextcloud
```

Now, run the command.

```
$ nextcloud
```

Nextcloud is now running and you can confirm this using systemctl:

```
$ systemctl status snap.nextcloud.*
```

To obtain more information use the info command:

```
$ snap info nextcloud
```

A Snap will be active once it's installed. For regular applications, this isn't a major issue. However, if we install *nextcloud*, as in the above example, then the service will continue running. When you no longer need the server to run you have two options: disable it or remove it. A service like this can, of course, be stopped with the *systemctl* commands and leave it at that. Here, we wanted to see if it's better to use a snap instead – and indeed it is! With a snap, you can list all your snaps and choose whether you want them active, disabled or thrown away. If you use the disable function, then the package will still be on the disc with all settings intact until you start it again or remove it.

The best thing about this approach is that snap will check all functions and stop the service without any additionals commands. From past experience, we expected to have to stop the service first and then disable it. If you've decided that this snap isn't your thing, you can erase it. All settings go with it so make sure you know that you've not made changes that you want to bring over to either a new install or perhaps another snap:

```
$ snap remove nextcloud
```

The above actions are the most common when you use snaps. If you've followed this tutorial thusfar, you have now done everything needed to verify the *nextcloud* package. Testers will find this useful because all files and configurations are in the snap tree. When you remove a snap, no files will be left behind.

The next section is our recommended approach for using AppImage. Hang on to your hats!

Download the file

Set execute permissions for the file. We have chosen to allow all users. It works even if you add only yourself.

```
$ chmod a+x Downloads/TheApp.AppImage
```

Next, run the application.

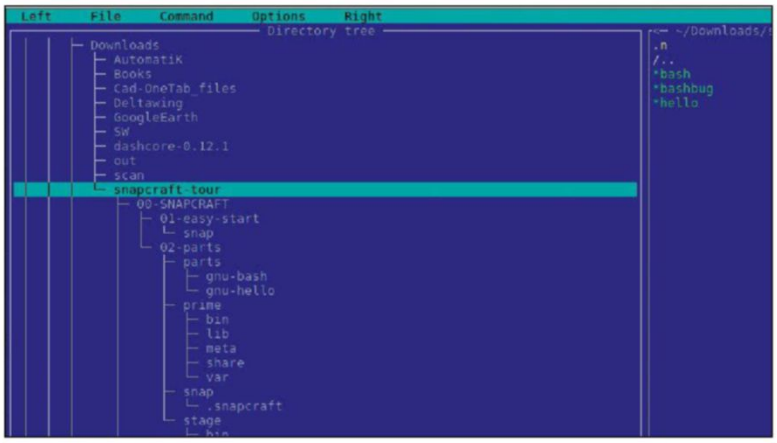
```
$ Downloads/TheApp.AppImage
```

That's it! As you can see, the system is created to make using software simple. The big challenge is for the programmers to make the package work in most distributions without creating a bloated download. To make things even simpler, use the appimage daemon. The daemon scans predefined directories for AppImages, sets the executable bit and then adds them to your desktop.

Snap requires a store. The default store is the Ubuntu one, but the idea is that anyone can run a store. If you're interested, check out **https://github.com/noise/snapstore**. This is a snap store example on Github that will enable you to »

Quick tip

If you have an application that's crashing, try the appimage to see if your libraries are to blame. For instance, *Calibre* crashed every time we tried to open the file dialog. It turned out that the library for gdk+ needed an update to handle Wayland correctly.



The snapcraft tour command creates directories that enable you to learn the basics. During the tour, you create regular snaps as well as web-based software.

gain some experience in running a store. A quick search on the internet doesn't reveal any stores – yet.

AppImage, on the other hand, is designed to enable anyone to create and distribute software without any central repository at all. With that said, there are already over 100 applications on bintray for you to download.

Living in Flatpak land

When using Flatpak, the installer checks dependencies and downloads runtimes. Runtimes are the libraries and other binaries that are necessary to run the packages. The Flatpak structure is similar to that of AppImage, except that it has a metadata file for configuration information, **/files** directory for source code and application information. The binaries go in **/files/bin** and any data that the application needs to share with the environment goes in the **/export** directory.

Flatpak is more like a package manager compared to AppImage, which should make the Flatpak packages smaller. With so few applications though, it's hard to judge right now.

Quick tip

If you're just starting out with snap and want to create snaps, install *snapcraft*. Then, follow the tour to get a feel for how the system works, including creating and removing them.

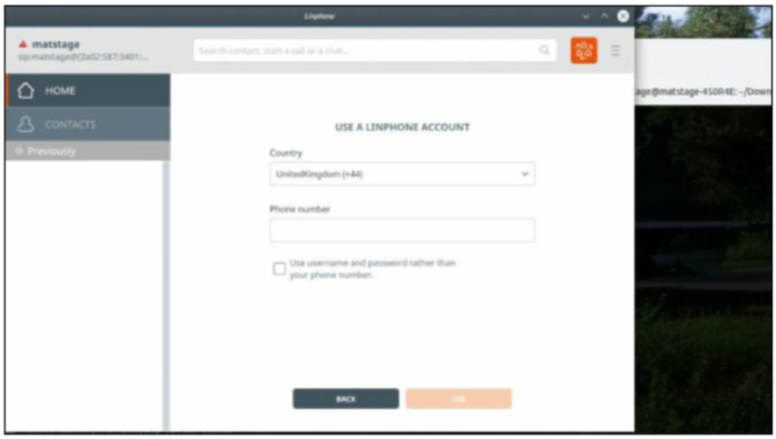
Creation is a snap

We'll start looking how to create packages, either as a developer or just because you want to try the application out. First up is snaps. The build tool for snaps is called *snapcraft*, so install that with your regular package management wizardry.

```
$ sudo apt install snapcraft
```

The *snapcraft* tool will compile the packages for you, so the **build-essential** package is required.

```
$ sudo apt install build-essential
```



Linphone is installed as a flatpak. When the flatpak installs, it also installs dependencies to the flatpak system without upgrading the rest of the system.

For the *snapcraft* tool to do anything useful, you need a configuration file called **snapcraft.yml**. The contents of this file describes the application you want to compile.

Running the *snapcraft* tool is simple: there are two different procedures. The simplest approach is to just run the command without switches in the directory where you put your **snapcraft.yml** file.

```
$ snapcraft
```

In this case, *snapcraft* will read the **snapcraft.yml** file and compile the defined package. After this, you install the snap.

```
$ snap install [yourApp.snap] –dangerous
```

When the packages become bigger, the procedure has a few more optional steps. This is because the system can add libraries that are uncommon. If your application needs a new library, you can add it as a part of your configuration file. When your package is this big, you have the option to compile one part at a time. In fact, *snapcraft* will detect if you already have the latest of any part and only recompile the updated part when you want to upgrade. You have the option to build a part at a time, use the build command.

```
$ snapcraft build [part]
```

Without the part parameter, *snapcraft* builds all the parts that need compiling. You can use this if you want to change only one part.

```
$ snapcraft clean [part]
```

The **snapcraft.yml** file controls it all.

The top of the **snapcraft.yml** file contains the descriptive information such as the name, the version and the description. The other sections control the behaviour of *snapcraft*, which in turn configures your snap.

The next section is confinement, because in a production snap this value needs to be strict. For security reasons, the strict value stops the snap from communicating with the rest of the system. While you're developing a snap, you need to set the value to devmode or use the switch `–dangerous` to make it run outside of confinement.

For the snap to work properly in any environment you then need to define what it's permitted to interface with. For example, a web browser needs to access the network as a client. To do this, you need to add the `network` parameter in the file. This parameter is called a `plug` when the snap needs a resource and a `slot` when the snap provides one.

The other sections are apps and parts. In the apps section, the name parameter names the executable and points to the command. Usually, they'll both be the same, but you may need to start it with bash, python or the like.

Moving on to the parts section, you need to define what you need to build your code with and where your code is. In

Zeroinstall, the original snap model

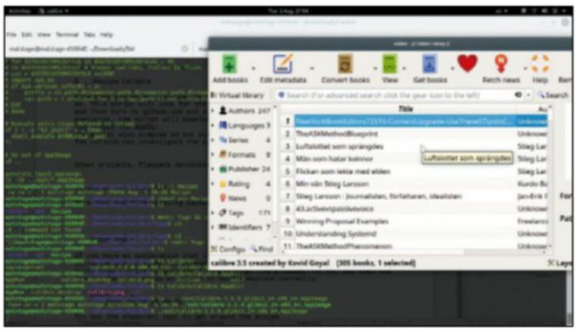
We briefly mentioned Zeroinstall elsewhere in this article, but it's an interesting project and is worth a little more of your time.

To install a program with zeroinstall, you use a public web address where the publisher has put the software. When you run the GUI, you'll find that there's no 'install' option. This is because packages run from a single directory. Similar to snap, zeroinstall will place all files you need in one folder. The vital difference is that the cache is under your user directory, usually with your permissions in place.

Since you're the only one with permission to run these files, another user on the system can't use the application and will be forced to download its own version.

In your home computer, this won't be an issue usually. However, Zeroinstall supports sharing of these files between different users and even virtual machines. To make sharing secure, zeroinstall checks the software against the sha signatures included by the designer, or the package maintainer if the two differ.

The contents of the xml file for distributing applications is simple, because the file only consists of 10 components. This makes maintaining a zeroinstall file a relatively straightforward task. However, bear in mind that you'd have to have a web address set up specifically for the program.



When you run the recipe script from http://bit.ly/2vYe2pG, all the required files are downloaded to the AppDir that will become part of your AppImage.

its simplest form, the parts directive contains the name of the app, the plugin and the source.

A simple example here:

```
parts:
  hello:
    plugin: autotools
    source: ./src/hello
```

The plugin will vary, depending on your source code. A reference list available at **https://snapcraft.io/docs/reference/plugins**.

Creating an AppImage

To make an AppImage, it's likely that you'll use the build services available online. One example is Travis CI.

Usually, the developer will create either the recipe file or the directory structure. If an existing package doesn't exist as an AppImage, you can create the directory structure yourself.

The directory structure stems from the ROX desktop project. The structure contains at least the files and directories shown below.

```
MyApp.AppDir/
MyApp.AppDir/AppRun
MyApp.AppDir/myapp.desktop
MyApp.AppDir/myapp.png
MyApp.AppDir/usr/bin/myapp
MyApp.AppDir/usr/lib/libfoo.so.0
```

The key file is **AppRun**. As mentioned earlier in this article, it'll configure your package. **AppRun** is a wrapper script that sets all environment variables at runtime and launches the application.

The myapp.desktop file follows the freedesktop standard. The appimage daemon will find and read this file to integrate the application into your desktop environment. Look inside the file to learn the details.

To show how this works, we'll make an AppImage using the *Calibre* recipe.

The **Calibre.yml** file already exists on **www.github.com**, and there's also a script that reads the recipe file.

Download the script with wget.

```
$ wget https://raw.githubusercontent.com/probonopd/AppImages/master/recipes/meta/Recipe
```

Set the executable bits.

```
$ chmod a+x Recipe
```

Make a directory for your AppImage and *cd* into it.

```
$ mkdir Calibre && cd Calibre
```

Finally, run the recipe script.

```
$ ../Recipe Calibre
```

You may notice that we've not included the yml extension, if you do, the script stops early. This script will look for the recipe file, first in your current directory and then turn to **www.github.com** and execute the recipe as a script. During the execution, the script will download the source code, binary files, and dependencies and create the AppDir.

The script also creates an out directory where you'll find your AppImage. Curious readers will no doubt investigate the AppDir under *Calibre*.

To make an AppImage for a developer requires just a little bit more work first. If you're interested, read the install document and then adjust the recipe file to the build system for the particular application. Then run the same script for that application, and increase the number of applications delivered by AppImage. Because the particular project will be pioneering, you'll run into a few more issues to handle, such as hard-coded directories in the source code.

The zeroth law

Until now, we've not mentioned zeroinstall. This is an earlier project that solved the same problem, but does not seem to have experienced the same take-up as snaps. We find this all rather puzzling because making your application ready for zeroinstall is very simple. You create a web page in the standard XML format that the makers prescribe in their instructions and then you test. The project is also very active, and to top it all off there are three versions that supporting the three big operating systems. You can read more about zeroinstall in the box (*below left*).

Turning back to snaps, you have secure confinement built-in if you register your application and set your signature. Appimage doesn't have this natively so for the security conscious use *firejail*. The package is simple to use just add the appimage switch.

```
$ firejail –appimage krita-3.1.2-x86_64.appimage
```

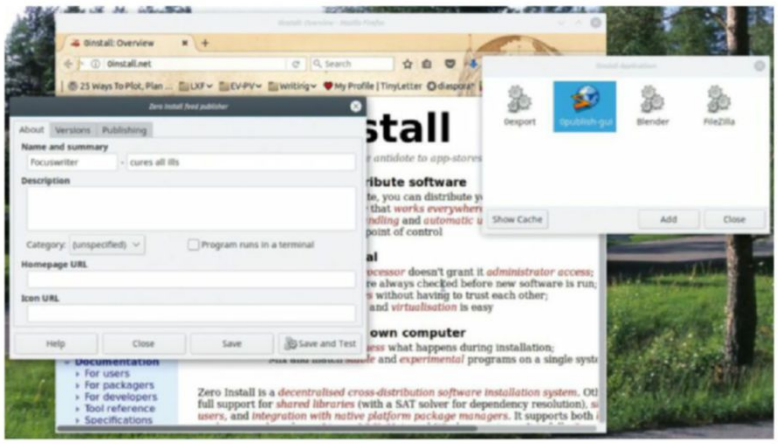
Inside the *firejail*, the application can still read your user data, so any files that you've put in your home directory will be accessible.

Some readers will be wondering why we need snap and AppImages at all. For many people, the distribution system works fine, and there's no problem to solve. Yet while the distributions are useful for keeping a stable system, it also requires more people between developer and end-user.

The problems start when you want to try an application, and the developer doesn't support your distribution. What will you do? Changing distribution is a big job, and other applications may not work in your new distribution. These solutions makes it easier to test applications and to get hold of uncommon software. LXF

Quick tip

An AppImage file is available for many applications. These files are excellent references if you want to make another application as an AppImage. To mount the AppImage, use the mount command. It may be necessary to use the offset parameter. Run `*.appimage –offset` to find the correct value.



Zeroinstall has both a GUI helper and a command line tool that enable you to create your publisher feed file. Publishing to the web is a built-in feature.

Swap: Migrate to a swap file

Mats Tage Axelsson reveals methods for migrating your system, enabling you to take full advantage of your available disk space.

Our expert

Mats Tage Axelsson
Mats has spent decades making his computers run Linux. The first was a laptop from IBM back when they were still making them.

Back in the early days of Linux, we measured RAM in megabytes. Under those conditions, you could only run a few applications before you ran out. A swap partition was necessary to keep the system running.

However, new computers won't run out of memory, so what's the point of having swap space at all? The answer is that it's a good idea to have some swap space for the few occasions you do run out of memory.

Ubuntu uses a swap file by default instead of a partition by default. The recommendation for a swap file is either two per cent of the available disk space or 2GB, whichever is smaller. Consider that a partition would be double the size of your RAM, in my case 4GB times two equals 8GB. That means I can save 6GB of disk space.

It's easier to create a new swap file than create a new partition or change the size of it. When you do a fresh install of Ubuntu 17.04, the installer will create a swap file, unless you specifically ask for a swap partition. The upgrade procedure, on the other hand, uses your current swap partition.

Where does that leave users who upgrade? They don't want to be left behind, but also have no wish to reinstall completely. Fortunately, the procedure isn't complicated, although a degree of caution is advised – so make sure you have a recovery disc to hand. Don't worry if you're not using Ubuntu. The procedure is simple enough and can be applied to other distros.

Swap creation

We can use either `fallocate` or `dd` to create a swap file. Root will be the file's destination. The fastest method is to use `fallocate`, but it uses the filesystem and so isn't ideal with all filesystems. However, it's fine to use with the ext4 file system.

```
$sudo fallocate -l 2G /Swapfile
```

Use your partition editor to remove the swap partition. GParted, disks and fdisk work equally well.

Using `dd` is slower but will reliably create a file space:

```
$sudo dd if=/dev/zero of=/swapfile bs=1024
count=2147483648
```

Secure the swap file. all programs use this file, so sensitive data may be inside it. Give permissions only to root. Then ensure the file has the correct permissions and ownership:

```
$sudo chown root:root /swapfile
$sudo chmod 0600 /swapfile
```

Initiate the file as swap:

```
$mkswap /swapfile
```

Check that your current swap has priority -1, which means no priority assigned.

```
$ swapon –show
```

To make sure the system uses your new swap first, use the priority parameter when you initiate the swap file. The swap space with the lowest priority will be quicker to turn off.

Your current swap partition will have it turned off (set to -1) by default.

```
$ sudo swapon -p 10 /swapfile
```

Have your hybrid swap and eat it too…

Systemd-swap is a tool that manages how you use zswap or zram. zswap compresses data that applications don't use and places it into a cache. The kernel then moves the data to the hard disk, and in most cases this speeds up the swap operation.

The *systemd-swap* utility can be used to create zram instead, which creates a virtual disk in the computer's memory, to minimise the number of swap operations. Both methods require memory space and so when zswap uses less memory, zram will make use of that memory to stop the swap process from occurring. Essentially, you need to choose one of the two approaches.

To install the tool, go to **https://github.com/Nefelim4ag/systemd-swap** and find the git address and download:

```
$git clone https://github.com/Nefelim4ag/
systemd-swap.git
```

Even though *systemd-swap* is outside of the standard distribution packages, the software is well written. You have the option to install with the `make` command or create a package file for Arch or Debian. The Debian package works on Ubuntu, too.

As soon as you've installed *systemd-swap*, you can configure it to use your choice of zram or zswap. The package installs the configuration files. In **/etc/systemd/swap.conf**, set zram_enabled or zswap_enabled depending on your needs. Use `systemd` to activate it:

```
$ systemctl start systemd-swap.service
```

Faster suspend

Tuxonice is an alternative method of suspending and hibernating your system. It requires a patch to the mainstream kernel, but there are packages available that will do this for you.

For Ubuntu, you have a PPA at **https://launchpad.net/~tuxonice**. Note that installing the package is simple, but making it suspend to a file is a little trickier, we've found.

One reason to use *Tuxonice* instead of the standard package is that the former doesn't clear the cache when suspending. The effect is that returning from hibernation with *Tuxonice* will usually make the system feel responsive faster than *uswsusp*, which dumps the cache when hibernating.

There are also many other features available, such as power management and improved ACPI support, so why not give it a try.

Check your swap usage: you won't see the file fill up immediately. The kernel keeps a lot of memory in the cache and on the hard disk. Finally, you may end up with following:

```
$ swapon –show
NAME       TYPE       SIZE   USED   PRIO
/dev/sda10 partition  12.1G   0B     -1
/Swapfile  file       2G     11.8M  15
```

When you can see that the system is using your swap file, you can turn off swap (you can use `swapoff -a`) for your partition. Ideally, your old swap partition should be empty, but this isn't necessary:

```
$ sudo swapoff /dev/sda10
```

The command `swapoff` may fail due to a kernel bug, without any obvious system effects, so check if the partition has disappeared from the listing:

```
$ swapon –show
NAME       TYPE       SIZE   USED   PRIO
/Swapfile  file       2G     11.8M  15
```

The whole procedure can be done on a running system and will only change things for the current session. Next, we will configure the system to do this at boot.

Making it stick

To use the swap file, we need to make the system start it at boot. The easiest way to do this is to add a line in the **fstab** file, such as the following:

```
/swapfile   none    swap  sw      0 0
```

You can prioritise the file in case you have several files on different media. For example, you may want to use an old slower drive as a second swap:

```
/swapfile_1   none    swap  defaults,pri=100 0 0
/swapfile_2   none    swap  defaults,pri=10  0 0
```

The system uses swapfile_1 to a higher degree than swapfile_2, so this would only be useful if you had swapfile_2 on another disk. You'll also need to remove your swap partition from the same file. But **fstab** is the older way of doing things and in most distributions, *systemd-fstab-generator* will convert this to a swap unit file during boot.

Now, you might think that the system won't use the swap partition, but see what *systemd-gpt-auto-generator* does. It finds all swap partitions and makes a unit file for it early in the boot sequence. If you wanted to boot your system once or twice before reclaiming the swap partition space, other measures are necessary. To have a swap partition without using it, you need to mask it:

```
$ systemctl mask dev-sdXX
```

Now try out the system for a while to make sure that there's enough swap space. When we decide to reclaim our disk space, remove the partition. To remove it, use *GParted*, *fdisk* or similar. *Systemd* may end up showing a failed service.

```
$ systemctl --type swap -a
  UNIT          LOAD   ACTIVE SUB    DESCRIPTION
  dev-sda10.swap loaded failed failed /dev/sda10
  swapfile.swap  loaded active active /swapfile
```

To solve this in *systemd*, you need to reset it.

```
$ systemctl reset-failed dev-sdXX
```

What about hibernating?

One way to hibernate is *uswsusp*. When you configure *uswsusp*, it's controlled by `uswsusp.conf` with the parameters resume_device and resume_offset. You'll find the offset parameter with swap-offset on your file.

```
$swap-offset /swapfile
```

The same needs to go into the **initramfs** resume file, in a slightly different format.

```
$ cat /etc/uswsusp.conf
# /etc/uswsusp.conf(5) -- Configuration file for s2disk/s2both
resume device = UUID=bb6a1ba2-1196-405b-825d-
5b7caf5347cc
resume offset = 32937984
$ cat /etc/initramfs-tools/conf.d/resume
resume=UUID=bb6a1ba2-1196-405b-825d-5b7caf5347cc
resume_offset=32937984
```

The resume parameter can be in /dev/sdXX format too, but UUID is a more robust solution. To make the values active, you need to update **grub** and **initramfs**.

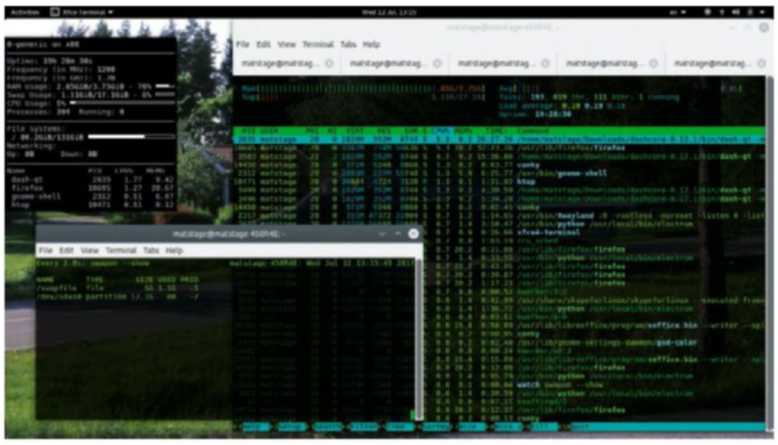
```
$ update-initramfs -u && update-grub2
```

If you skip one of the two, your system will lock up trying to find the resume file. But usually you can recover using the recovery option from **grub**. The problem is now the uswsusp encryption scheme isn't up to date. Support for encryption seems a long way off, judging from activity by the developer.

What's worse is that encryption is automatically on if your home is encrypted. To avoid this issue, you can use *Tuxonice*, which is available as a separate kernel. There's a PPA available if you have Ubuntu. LXF

Quick tip

One problem you may run into is that you have the wrong kernel running. Kernel 4.10.22 has a bug that makes the system unresponsive due to swapping functionality. So make sure that you have at least 4.10.26 in place. If you turn off the swap with the wrong kernel, your system may fail.



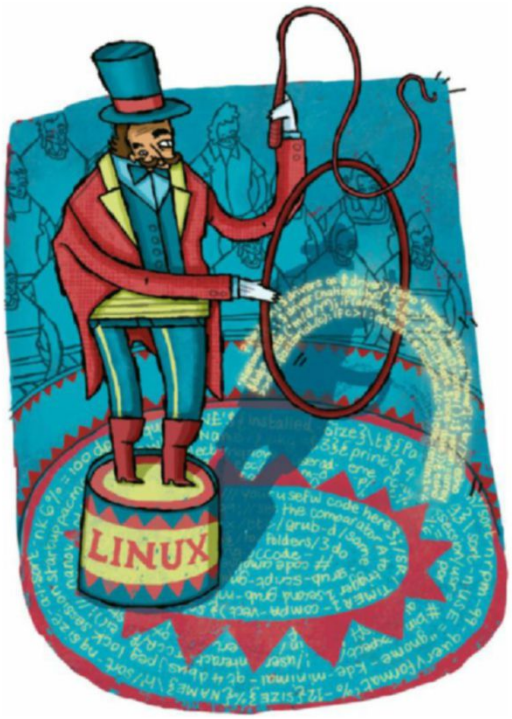
Use conky or htop to see your swap space usage. Note that the swapon –show command only displays data in text format.

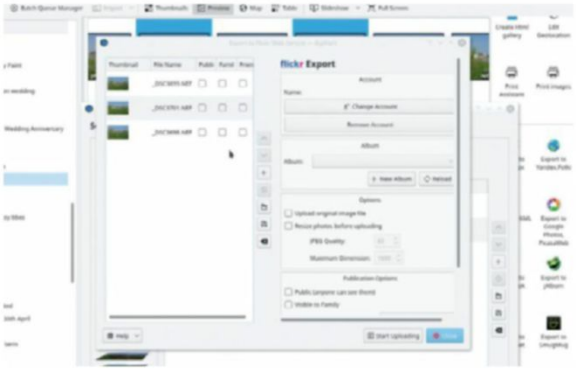
Digikam: The ultimate toolkit

There's nothing quite like Digikam for all-round photo mastery. **Adam Oxford** gets you started with this comprehensive tool.

Our expert

Adam Oxford
Adam has been using Linux for semi-professional photography for about 10 years. The guys in these shots are from rhino protection team, **www.pittrack.co.za**.





Extend the features of digiKam and enable automatic uploads to Flickr, Facebook, Google Drive and more by grabbing the common Kipi plugins from your repository.

Gnome's default photography tools are a disaster. The ageing *Shotwell* is okay, but inclined to hang if you ask it to deal with more than a couple of year's archives, and the new *Photos* has a mind of its own when it comes to building a navigable archive. Gnome users who are serious about their photography can draw upon a hotchpotch of decent tools, but nothing comes close to the joy that's expressed by their KDE-preferring peers who use *digiKam*.

digiKam is unique not just in the Linux world, but pretty much in all of computing for its comprehensive approach to photo management. Some software is good at RAW conversion, some is good for building a navigable library of thumbnails. Only *digiKam* has it all, from lightbox to facial recognition and even a "fuzzy search" tool for looking up similar photos to the one you're looking at now.

It's phenomenal, and it's also – at first glance – a touch intimidating. Here's how to get started.

Download the update

At the time of writing, the latest version of *digiKam* is 5.6.0. It's a recent update, though, so your repository is probably stuck at 5.5.0. There are a couple of features in the newer code that you'll want, specifically the ability to build HTML galleries and upload them directly from digiKam and to edit together photo slideshows. In Kubuntu, you can install the update by going to Konsole and entering:

```
sudo add-apt-repository ppa:philip5/extra
sudo apt update
sudo apt install digikam
```

The first time you run *digiKam*, it'll take you through a set-up wizard. For the most part, you won't want to change any of the default options, but you will want to tell *digiKam* where you keep your images. This could be in **/home/user/Pictures**, or it could be in a separate drive with an archive built up over the years.

digiKam will also need to build a few database files, which can be saved in the default location or somewhere else. If you do want to move it, make sure it's not on a removable or networked drive.

On its first run, *digiKam* will begin thumbnailing and sorting all the images in the location you set.

Finding your way around

For the most part, *digiKam* is straightforward to navigate. On the left-hand side of the screen there's a vertical icon menu that selects how you want to search for images. These are based on the metadata of the images in your library. On the right, there's another vertical menu that enables you to manipulate the currently selected of image or images.

When you select an option from the left or right, it'll open a pane on that side of the main *digiKam* window that can be hidden by clicking again. The top set of icons controls what you see in the main pane, or open up separate windows for the Light Table and Image Editor.

Quick tip

To reset *digiKam* to its original state, go to **/home/user/.config** and rename the file **digicamrc** to **digicamrc.bak**.

HDR and panoramic images

You can stitch multiple photos together to make a panorama or high dynamic range (HDR) image without leaving *digiKam*. You will, however, need to install the Hugin plugin first. KDE will direct you to the Hugin website (**www.hugin.sourceforge.net**) to do this, but chances are that you can grab it from your distro repository using `sudo apt install hugin` or equivalent.

To create an HDR image, you'll need access to a bracketed photograph. This just means that three identical images have been taken at different exposures: the one in the middle is the right exposure and the others are under- and over-exposed brackets. Most modern cameras have a setting that will enable you to do this automatically.

In *digiKam*, you can combine these three images into one HDR image using the Create Stacked Images option from the Tools menu. Select your brackets images, click the option and then just click through the options without changing anything: *digiKam* will do the rest.

The Create Panorama option will do the same thing and align up multiple images shot in a row.

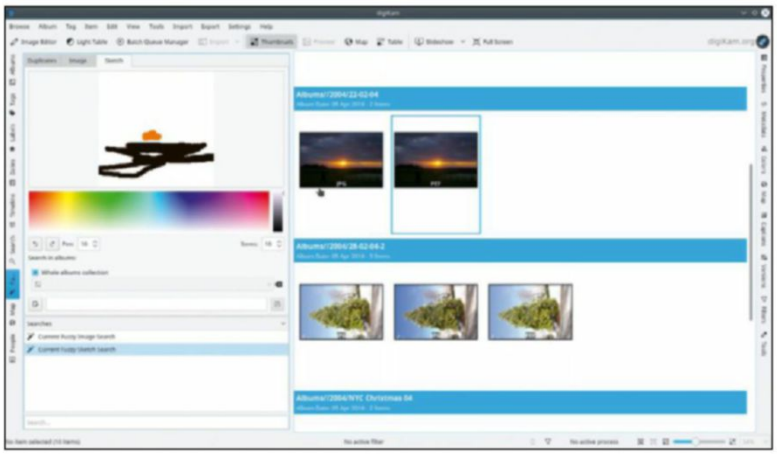
The interface has a few odd quirks you'll need to be wary of. For example, there are three buttons called Map, one on the right, one on the top, and one on the left. The left is to search for images with geolocation tags in their metadata; the one on the right is to edit the metadata of the currently selected image. The top one should be similar to the one on the left, but operates in the main pane, but in our testing is pretty buggy. There's also some repetition: the Tools button on the right is almost, but not quite, identical to the Tools menu in the top bar.

As well as maps, there are other fun features you can play around with at your leisure. There's an experimental feature for automatically recognising faces in your archive, and a "Fuzzy" search with which you can draw a very rough outline of what a photo looks like in your memory, and test *digiKam's* ability to find it from your doodle. More usefully, Fuzzy search can be used to find images similar to, or duplicates of, the one you're currently looking at.

In order to use the Fuzzy search function, *digiKam* will need to fingerprint all the images in your library, which can take a long time if you have a lot of shots.

Realistically, however, of all the navigation functions on the left the ones you're most likely to use are Albums, Labels, Tags and Dates. Labels, Tags and Dates are based on metadata captured from the images themselves, and Timeline is effectively the same as Dates, but uses a graph instead of a folder tree to sort images by date captured.

The Albums view is a blend of a file browser and a virtual album library. There are two ways to get images into *digiKam*. If you have a large library of images already on your hard drive sorted into meaningful folders (such as Year/Date/Location), the easiest way to import them is to add your Pictures folder as an Album. This can be done either in the set-up wizard or by going to Import>Add folder and then selecting Pictures>New album>Pictures.



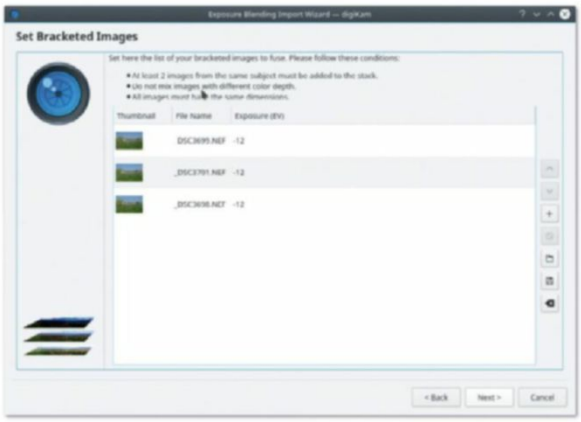
Fuzzy searching by sketch is surprisingly fast, but the results can be variable.

Now, whenever you add a new folder to your Pictures folder on your hard drive, by copying it over from your camera for example, *digiKam* will update its library.

If you want to use *digiKam* to import images directly from a camera or SD card, just connect them to your PC via USB (or a built-in card reader) and use the Import menu to pull the photos over. *DigiKam* doesn't use virtual albums, so if you tell it to import pictures to an album it'll make copies in that physical location. That's fine if you're importing from a camera, but could see you fill a hard drive with duplicates if you already have the pics on your PC already.

Quick tip

Use the Settings icon in Map mode to switch between a 3D globe or a flat Google Map.



ExpoBlending in action: how to make pseudo HDR images with digiKam using a stack of bracketed images.

Working with your photos

Once you've got to grips with importing and sorting images, the main image pane should start to make more sense. Call up a file location, date, search or tag in the navigator and all the images linked to that will appear as a thumbnail grid in the centre pane.

Using the top icons, you can also view this list in a row-by-row format by clicking "Table" or on a map, but realistically the two views you'll use most are Thumbnails and Preview. Preview shows the currently selected image with the rest of the folder as a film strip at the top, while Thumbnail is a grid of images. You can change between the two views quickly by clicking the selected image.

By default, there's a lot of information in the Thumbnail view. Each thumbnail includes the image name, rating, tags, format, caption, description and label below it, as well as some basic editing tools for rotating the view above. Unless

you're an aspiring archivist, it's unlikely that you'll use all of these all of the time.

Typically, you might reference an image by filename, label and tags, for example. Without adding the rest of the information to all the images, it's going to leave a lot of blank space reserved for that information.

You can tighten up this view by going to Settings > Configure digiKam. Under the Views tag, you can customise the information that appears underneath each image. You'll most likely want to restrict this to the filename, rating and label. Now select Tooltips. In the Icon Items tab you can make all of the other information about the picture that you might want appear when you mouse over a thumbnail.

Sorting your collection

DigiKam will enable you to edit any part of the metadata stored as part of an image, right down to the type of camera used to take the shot, and the lens settings. Select an image and go to Item > Edit Metadata. This is more useful if you want to scrub metadata from a shot for privacy reasons than if you want to fool your friends into thinking you own a £6,000 camera by changing the EXIF information. From the same Item menu, you can also (more usefully) add geolocation data if it's not been written in by your camera.

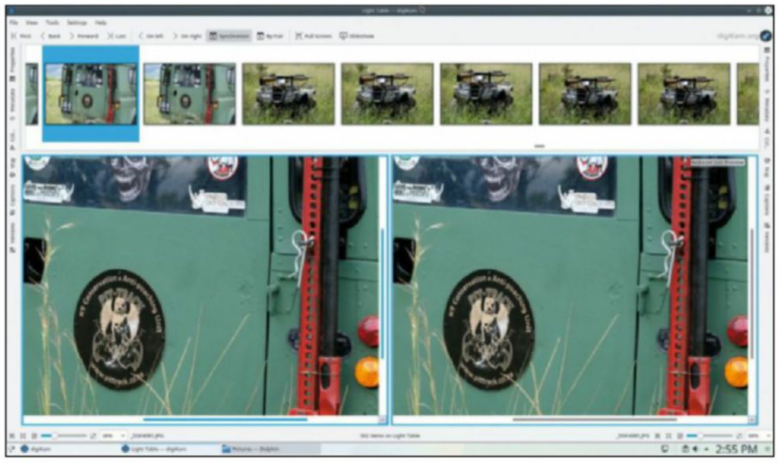
You can also view the metadata for a particular image using the Properties and Metadata menus on the right, but these won't enable you to make any changes to an image.

The three main types of metadata that you will want to edit, however, are Tags, Ratings and Labels. Each has a key role to play when building up a workflow for processing pics.

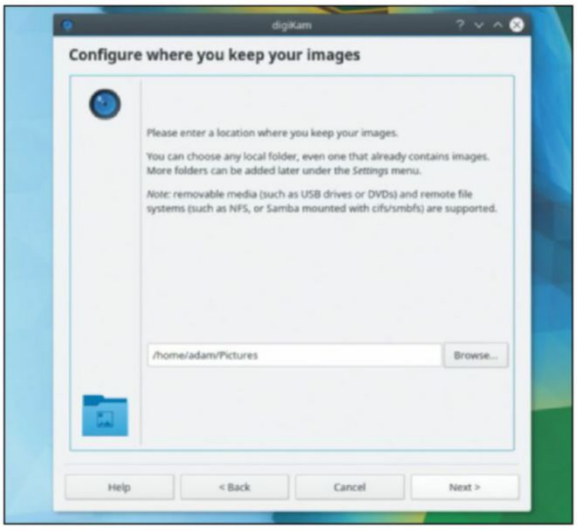
Tagging photos is time consuming, but useful when you're searching a large archive at a later date. You can apply tags either by right-clicking a thumbnail or selection of thumbnails, or by using the Filter or Captions menus on the side. Each of these has a list of current tags which you can drag and drop onto images or groups of images. Both of the Tools menus contain a link to open the more powerful Tags manager, which is great for creating and editing tags in batches.

Quick tip

You can customise RAW processing by going to Settings>Configure digiKam>Image Editor and select Always open the RAW Import Tool.



For comparing similar shots use LightTable to scrutinise their exposures.



Make your life easier by setting a sensible working folder. DigiKam does support network shares.

Labels perform a similar purpose in that they make it possible to add flags to your shots to filter later. If you're trying to whittle down a thousand images from an afternoon shoot to just a dozen, they're invaluable.

The simplest way to add a label is to right-click an image or selection of images and in the context menu go to Assign Labels. If you're working with a lot of images, though, mastering the shortcuts is essential.

Alt+3, for example, will label an image as Accepted, while Alt+1 will label it rejected. You can then filter the thumbnail grid using the Filters menu to only show Accepted pics. Labels can be Picks, Colours or Ratings – it's best to choose one of the three to work with and stick to that for consistency.

The final tool for selecting images is the Light Table, which enables you to do side-by-side comparisons of similar shots. Open the Light Table, then drag-and-drop the images you want to compare into the two panes that appear. Each pane has its own information menu running vertically down the side. This is particularly useful because the exposure histograms for each image is in the Colors tab, so you can see if one is blown out while the other is perfectly exposed.

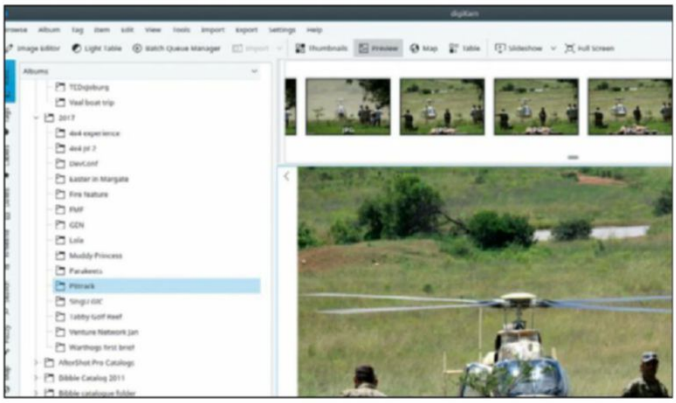
Throw in a passable RAW converter and image editor, and you can see why *digiKam* is a compelling reason for photographers to opt for a KDE desktop environment. There's nothing quite as comprehensive as this photo management suite on any platform. LXF

Baking batches of photos

One of the standout features of *digiKam* is the Batch Editor. As its name suggests, it enables you to perform the same edits to multiple pictures as a group. While many other batch process tools exist for renaming or resizing photos, *digiKam's* editor makes it possible to perform any edits that you can apply to a single photo to a group of images.

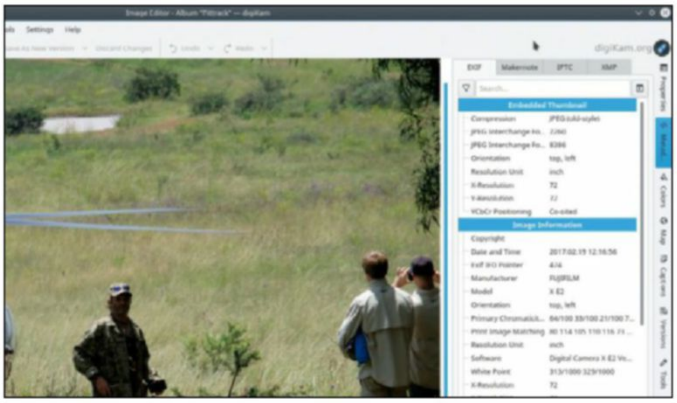
Either select a group of pics in the browser and open up the Batch Editor window, or drag-and-drop files into the left hand pane of the editor. Now you can build up a process by dragging tools from the Control Panel in the bottom right to the Assigned Tools space in the middle. In the top right pane, add tool settings such as the new size, metadata, effects applied or name. It's especially useful if you want to apply copyright (or copyleft) details to every pic.

Get to know digiKam's image editing tool



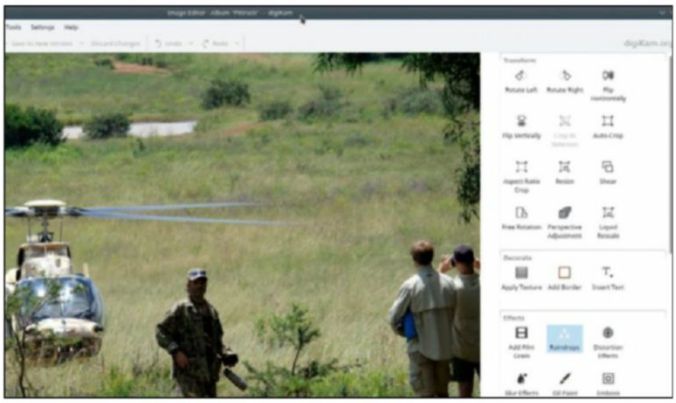
1 Getting started

GIMP or *Krita* will be the go-to tools for image editing for most Linux photographers, but for basic retouching *digiKam* has a comprehensive suite of tools built-in. Select the image you want to change, then click Image Editor at the top of the screen.



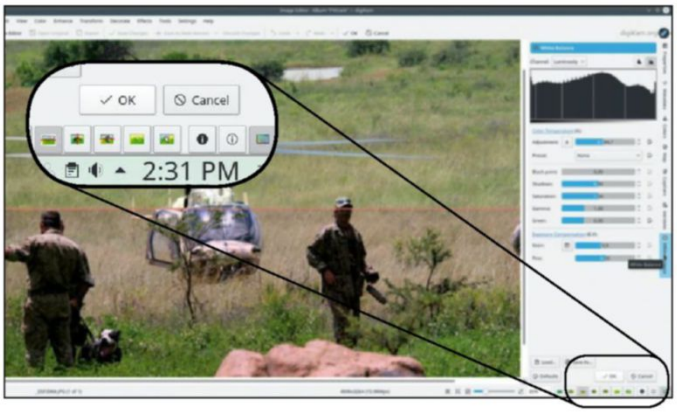
2 Edit metadata

The Image Editor has a vertical window on the right that gives you the ability to rewrite metadata and add captions to the photo in question. It also has a handy Versions tab, which makes it possible to see before and after takes of an edit.



3 Find the tools

All of the image-editing features are available from the menus in the window header, grouped under Color, Enhance, Transform and so on. For a more traditional feel, however, click Tools on the right to display a list of all the editing features as icons.

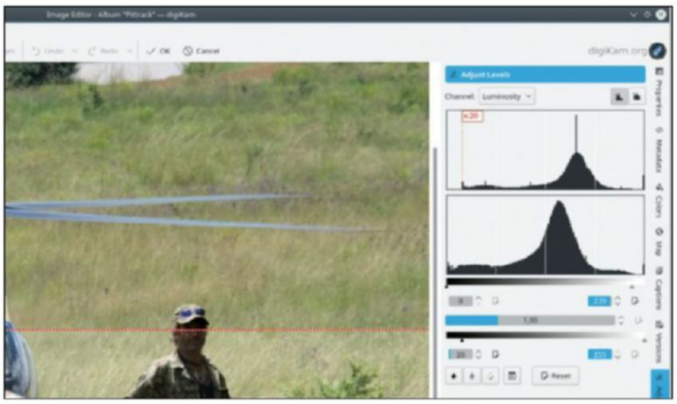


4 Check it out

The tiny green icons along the base of the toolkit icon window enable you to see changes that you're about to make to an image on a split screen. This is handy if you're making global changes to the White Balance (which is the best place to start editing).

5 Exposure check

Next to the green icons, there are two information icons. One has a dark background, the other has a light background. These will highlight areas that are under- or over-exposed in a shot. For some quick fixes, try the Autocorrect button under Colors.



6 Exposed

The most important tool that everyone should master is the Adjust Levels tool. This shows a histogram of the brightness level across the photo. A well-exposed photo will be smoothly distributed across the range, so use the sliders to adjust your shot.

SDN: Network with ease

The Dark Lord of Network Operations, **Tim Armstrong**, sheds some light on software-defined networking and open network switches. Fear his power!

Our expert

Tim Armstrong is the network architect at Nerdalize. He designs and implements datacentre and ISP networks. He's a real control freak when it all comes to Bits.

The rise of open network switches and Software Defined Networking (SDN) has begun a paradigm shift in datacentres, and with a global shortage of network engineers now is the time to get on the bandwagon.

Deep in the heart of both the Linux kernel team and the engineering department at Mellanox, engineers have been working hard to bring the biggest change to Linux networking since Intel opened the source code for the e1000 driver. And just like that driver changed the face of Linux networking 10 years ago, SwitchDev does it again. It embodies the beginning to true SDN to the Linux community, enabling complete control of a switch's hardware with the tools you use every day for managing any Linux machine's network stack.

SwitchDev provides an abstraction layer over the switch's hardware, making it possible to configure your switches as if they're nothing more than a Linux server with an enormous number of NICs. All the heavy lifting is handled for you by the kernel: bridges converted into FIBs, interfaces into ports, VLANs are... well VLANs, and all the forwarding and routing is off-loaded onto the hardware automatically.

Network emulation with GNS3

To open this tutorial up to everyone who doesn't happen to have an £8,000 Mellanox Switch just sitting around doing nothing, we're going to use GNS3 and some virtualised appliances. Because Mellanox's drivers have been upstreamed to the Linux kernel tree our Open Network Switch appliance and all the kernel we'll be using is up to date and compatible with the majority of features in the Mellanox Spectrum ASIC.

Specifically we configured the switch appliance to be as close as possible to a Mellanox SN2100 as is currently possible in GNS3. As for the distro that we're going to use on the appliances, we're going to go with Devaun because it's lightweight, stable and secure.

Instructions for installing GNS3 on pretty much any OS are available on **www.gns3.com**, but for simplicity we've included the Ubuntu instructions here:

```
$ sudo add-apt-repository ppa:gns3/ppa
$ sudo apt-get update
$ sudo apt-get install gns3-gui gns3-server
```

Because the images we're using are KVM/QEMU, you'll also need to install *qemu* and *libvirtd* again. We've included the instructions for Ubuntu below, but you can find instructions for your preferred distro quite easily online.

```
$ sudo apt-get install qemu-kvm qemu-system-x84 qemu
$ sudo apt-get install libvirt-bin
```

Now GNS3 is installed we need to import our appliances: a GNS3 compatible facsimile of the Mellanox SN2100, a simple Devuan server, and a Devuan desktop. These will be downloaded automatically. When you open GNS3 for the first time you'll be prompted with a wizard that will guide you through importing appliances, and because this is incredibly straightforward we won't duplicate the instructions here.

Plugging and playing

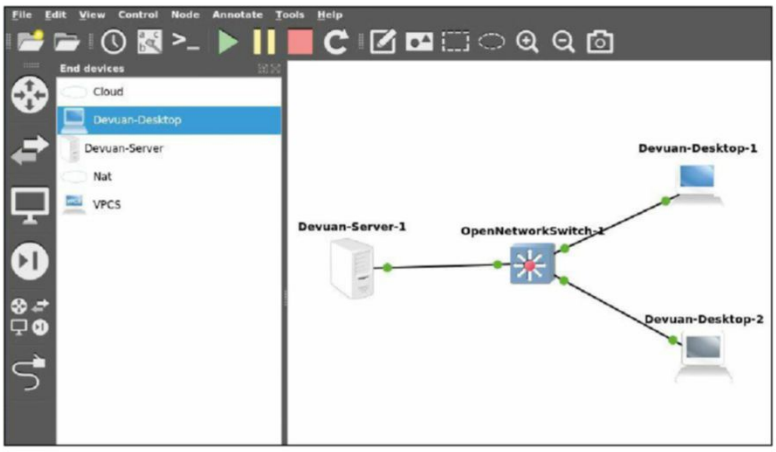
For this tutorial we want to keep things simple so as to keep the focus on Linux, or rather more specifically SwitchDev, without going off into an overwhelming dissertation on the merits and issues of different topologies. To this end we'll use only one switch, two clients and one server.

To do this simply select the switch drawer on the left and drag an OpenNetworkSwitch into the middle of the work area. Next, drag a couple of Devuan Desktops and a Devuan Server from the End devices drawer into the work area. Arrange your devices around the switch however you feel comfortable.

To create a connection from Server to the Switch select the Add a Link tool from the bar on the left, then click thse server, select eno1, and then click the switch selecting SW1P1. Now do the same for Desktops 1 and 2 selecting SW1P11 and SW1P12, respectively. Then click the "Add a Link" tool again to disable it.

Once all the devices are connected correctly we can start them up. Press the play button on the top toolbar. The VMs will take a short time to launch; you can monitor the progress of the boot by opening the OpenNetworkSwitch's console. Open the console by right-clicking the device icon and selecting Console. Once booted you can login with the user root and the password nauved. Once logged in you can see how the interfaces are presented with the `ip` command:

```
$ ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc
```



Once you've wired up the first few devices your network should look something like this.

The OSI zombie

While it's becoming increasingly less relevant, with the addition of overlay protocols, the OSI model is still the most prevalent method for describing the network stack The OSI model classifies the network into seven layers, the most relevant of which are:

OSI Layer	Role	Examples
Layer 1	Physical	Cables (CAT5e, CAT6, CAT7), Fiber (G.652, G.657), Radio (802.11ac, GSM 3G, LTE 4G)
Layer 2	Data Link	Ethernet, ATM, PPP
Layer 3	Network	IPv4, IPv6
Layer 4	Transport	TCP, UDP
...		
Layer 7	Application	HTTP, DNS, TLS, SMTP

Frequently, you'll see network engineers shorten the layer names such as L2 or L3, just to save themselves a bit of time.

Sometimes however, we need to use protocols that don't fit neatly into this model. This results in the use of terms like Layer 2.5 or Layer 3.5 to describe protocols such as MPLS and GRE tunnels, respectively. Furthermore, there's a common consensus that layers 5 and 6 don't really exist anymore. Coupling this with the rise of SDN and Overlay technologies is leading many to pronounce the OSI model to be dead. Yet as it stands no viable replacement has taken hold, and so OSI lurches on, like some kind of heavily mutilated Frankenstein-esque zombie.

```
noqueue state UNKNOWN mode DEFAULT group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: mgmt0: <BROADCAST,MULTICAST> mtu 1500 qdisc
noop state DOWN mode DEFAULT group default qlen 1000
    link/ether 00:75:f0:86:64:00 brd ff:ff:ff:ff:ff:ff
3: sw1p1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu
1500 qdisc pfifo_fast master br0 state UP mode DEFAULT
group default qlen 1000
    link/ether 00:75:f0:86:64:01 brd ff:ff:ff:ff:ff:ff
...
18: sw1p16: <BROADCAST,MULTICAST> mtu 1500 qdisc
noop state DOWN mode DEFAULT group default qlen 1000
    link/ether 00:75:f0:86:64:10 brd ff:ff:ff:ff:ff:ff
```

Because Mellanox's Spectrum series ASIC supports everything from 100Gbps down to 1Gbps it's normally helpful to set the port speed before activating any interfaces or including them in a bridge. In this case we're using a virtualised environment so this isn't required. However, if for example we were going to use a 1Gbps connection for each desktop and a 10Gbps connection for the server, then we could configure that by setting the following:

```
$ ethtool -s sw1p1 speed 100000 autoneg off
$ ethtool -s sw1p11 speed 1000 autoneg off
$ ethtool -s sw1p12 speed 1000 autoneg off
```

Bridging the gap

Now that the ports are all set to the right speeds we can create the bridge. This is largely the same as creating any Linux bridge on Debian-based systems (such as Devuan). This is achieved by editing **/etc/network/interfaces**.

In order to associate a switch port with a bridge you need to decide on what type of association it has with the bridge. At the time of writing Linux's VLAN-aware bridge system isn't particularly user friendly when it comes to attaching access ports, therefore it's recommended that you use classic bridges. If a given interface needs to be a trunk port you also need to define each VLAN at the port level and attach it to each of the VLANs (bridges). In this example however, we're keeping it simple, so we won't be using any trunk ports.

Let's set up one bridge for each of the ports we're using, so that we have the option to attach more devices to each subnet at a later date without having to bring existing connections down. Append the following to the **/etc/network/interfaces** file:

```
auto br0
iface br0 inet static
  bridge-ports sw1p1
  bridge-stp off

auto br1
iface br1 inet static
  bridge-ports sw1p11
  bridge-stp off

auto br2
iface br2 inet static
  bridge-ports sw1p12
  bridge-stp off
```

Each bridge acts as a virtual segment in the switch and enslaves the ports we're using to the segment. This automatically alters the port-mapping and FIB.

Bringing each bridge up now would enable Layer 2 switching on that segment, which would be an acceptable solution if we wanted a Layer 2 solution, but Layer 2 networks aren't very scalable, and no-one buys a £8,000 switch to do something a £800 switch can do. Because we want to work with modern architectures, we'll need to enable Layer 3 switching and Inter-VLAN/cross-segment routing. On a SwitchDev platform it's as simple as adding the relevant IP definitions on the bridges. To do this just modify **/etc/network/interfaces**:

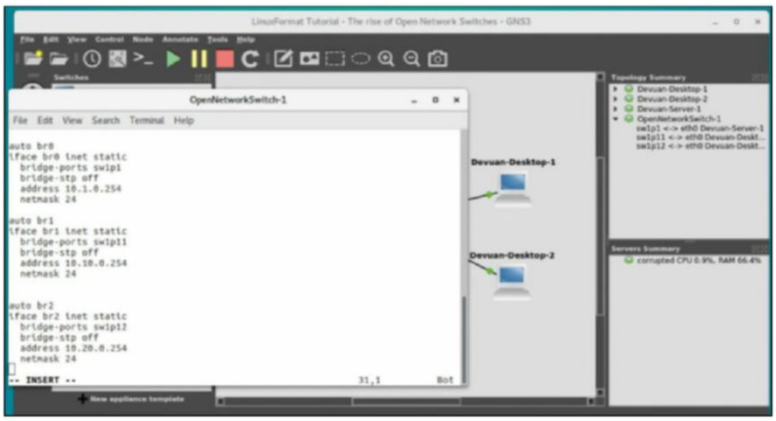
```
auto br0
iface br0 inet static
  bridge-ports sw1p1
  bridge-stp off
  address 10.1.0.254
  netmask 24

auto br1
iface br1 inet static
  bridge-ports sw1p11
  bridge-stp off
  address 10.10.0.254
  netmask 24


auto br2
iface br2 inet static
  bridge-ports sw1p12
  bridge-stp off
  address 10.20.0.254
  netmask 24
```

Now we can start bringing interfaces up.

```
$ ifup br0
$ ifup br1
$ ifup br2
```

Ugh... clients

Now the switch is configured it's time to configure the server and the clients. This is fairly standard boilerplate but for verbosity it bears repeating. On the server you'll want to edit **/etc/network/interfaces**, setting the eth0 IP in the range of »



Double-clicking a device that's booted up opens the appropriate terminal.

10.1.0.1 to 10.1.0.253 and the gateway to 10.1.0.254. In CIDR notation this would be 10.1.0.0/24 (excluding the network address [0], broadcast [255]). This would result in a config block similar to the following:

```
auto eth0
iface eth0 inet static
  address 10.1.0.1
  netmask 24
  gateway 10.1.0.254
```

Next, we need to do the same for the two desktops, modifying the IP addresses appropriately. For example, the config block for Desktop 1 would be as follows:

```
auto eth0
iface eth0 inet static
  address 10.10.0.1
  netmask 24
  gateway 10.10.0.254
```

Once we have both the server and the two desktops configured we can bring the networks up as normal:

```
$ ifup eth0
```

Finally, we can test the connection from each end-device to the switch. From the Server to the switch:

```
$ ping  10.1.0.254
PING 10.1.0.254 (10.1.0.254) 56(84) bytes of data.
64 bytes from 10.1.0.254: icmp_seq=1 ttl=64 time=0.359 ms
64 bytes from 10.1.0.254: icmp_seq=2 ttl=64 time=0.493 ms
```

From Desktop 1 to the switch:

```
$ ping 10.10.0.254
PING 10.10.0.254 (10.10.0.254) 56(84) bytes of data.
64 bytes from 10.10.0.254: icmp_seq=1 ttl=64 time=0.513 ms
64 bytes from 10.10.0.254: icmp_seq=2 ttl=64 time=0.471 ms
```

From Desktop 2 to the switch:

```
$ ping 10.20.0.254
PING 10.20.0.254 (10.20.0.254) 56(84) bytes of data.
64 bytes from 10.20.0.254: icmp_seq=1 ttl=64 time=0.314 ms
64 bytes from 10.20.0.254: icmp_seq=2 ttl=64 time=0.499 ms
```

Finally from the Server to the each Desktop:

```
$ ping 10.10.0.1
PING 10.10.0.1 (10.10.0.1) 56(84) bytes of data.
64 bytes from 10.10.0.1: icmp_seq=1 ttl=64 time=0.712 ms
64 bytes from 10.10.0.1: icmp_seq=2 ttl=64 time=0.520 ms
$ ping 10.20.0.1
PING 10.20.0.1 (10.10.0.1) 56(84) bytes of data.
64 bytes from 10.20.0.1: icmp_seq=1 ttl=64 time=0.849 ms
64 bytes from 10.20.0.1: icmp_seq=2 ttl=64 time=0.591 ms
```

You should now see all the connections are functioning correctly and our switch is handling the routing for us nicely.

So what's really happening here? Well, in our simulation, nothing much: we effectively just created a Linux-based router, but had this been a SwitchDev compatible switch it would be a very different story. We just did something that a few years back would have required two NDAs and an SLA just to get the SDK. We configured a switch, with Linux, using the tools we've known for years, without a single binary blob in sight. Not even a shim, just pure open source Linux. Time to stand back and admire your handiwork.

Entering the real world

For those who have configured Linux bridges before, all this will have seemed very familiar, but that's the point. SwitchDev is supposed to make things easy, by reducing complicated switches CLIs, APIs and SDKs into simplistic Linux commands that just work. You install your distro of choice and then get on with it.

What's more, you can fully automate the process using your favourite tools such as *Chef*, *Ansible* or *Puppet*. If it can configure a Linux network stack it can configure a SwitchDev compatible switch. Thanks to the engineers at Mellanox and the Linux Kernel Developers, SDN is no longer complicated, expensive or reliant on specialised black box controller units.

Unlike other SDN technologies such as OpenFlow, SAI and OpenNSL, SwitchDev is keeping the brains in the box, resulting in network latencies in the range of tenths of a millisecond – all while simplifying the implementation of massively redundant network topologies.

Furthermore, thanks to SwitchDev the convergence of Switches and Routers becomes possible, resulting in a simplified network edge where one device supports both functions. Adding BGP support to your switch is as simple as installing *Quagga* and hey presto, you have all the most popular routing protocols available.

With any luck the pioneering work done by the Linux Kernel developers and the engineers at Mellanox will be the beginning of the end for the layers of bureaucracy and licensing required to gain access to Switch APIs.

Beyond the basics

Now that all the basics are up and running, let's tackle a more realistic network, splitting the responsibilities of the server(s) and the desktops into a closer approximation of a mid-size company network (not quite as complex as an Enterprise network, but it's got most of the building blocks in place).

To start off, because we're using KVM/QEMU as our hypervisor we need to shut down our appliances before we can rewire our network in GNS3. To do this hit the big stop

Virtual routing and forwarding

Virtual routing and forwarding (VRF for short), isn't entirely as it sounds. VRFs resolve the issue of both wanting the benefits of L3 switching, without needing to worry about potential IP conflicts over the network.

This is even more useful than it first appears, because it removes the limit of only having approximately 18 million private IPv4 addresses, which is essential when designing large cloud computing networks. Furthermore, it makes possible the additional option of carrying foreign L3 traffic from clients without needing to encapsulate it in an overlay protocol.

VRFs are a staple of carrier networks, ISPs, clouds and enterprise networks. They simplify network designs, and reduce FIB size and network latency. Effectively, VRFs are simply a way to divide a switch's RIB table between separate namespaces, providing a L3 equivalent to VLANs.

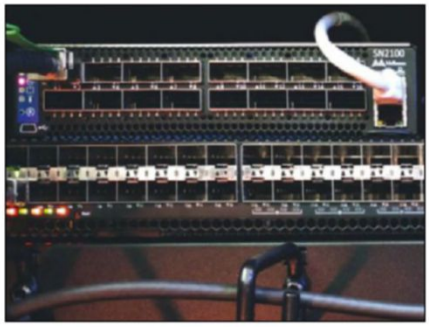
Unlike VLANs however, VRFs, being a L3 feature, are device specific. This means that for a VRF to span multiple devices across the network both a routing protocol (such as iBGP) and a transport protocol (MPLS, say) are required for traffic to pass from the source to the destination.

Comparing Layer 2 with Layer 3

When you first enter the networking world, Layer 3 switching can seem too complicated for its own good. Introducing segmentation across the network and increasing overall complexity, with additional protocols required to handle the distribution of routing information just so two or more switches can exchange traffic... this all makes Layer 2 networking child's play by comparison!

Yet this additional complexity brings more flexibility. It enables loops and multiple paths through the network to be fully utilised, thereby increasing throughput and redundancy while decreasing latency. There's also the possibility of using protocols such as OSPF to further optimise traffic flow through the network.

A common example of Layer 3 network architecture's superiority to Layer 2 network architecture is the implementation of a Clos network. This is a type of Full-Mesh network consisting of two layers of switches, a high throughput core (called Spine switches) and Top of Rack distribution switches (called Leaf switches). The structure of a Clos architecture means both link saturation and redundancy are possible. The nearest L2 facsimile of this would be a tree, where in order to enjoy any form of redundancy a spanning tree protocol would have to be deployed forming a very complex and strict hierarchy tree over your network, where any single interconnect can become a severe bottleneck affecting the whole network.



Gratuitous shot of a Mellanox SN2100 in a test rack. There's is a 40G QSFP cable in port 1 connecting it to a EdgeCore AS5712.

button on the top toolbar in GNS3. Next, drag another switch from the appropriate drawer on the left onto the workspace.

Now click the path between the first switch and the server and press the delete key. Then pull up the Add a Link tool, connect the server to the new switch on `swlp10`, and connect `swlp1` from the first switch to `swlp1` the second switch. Then start the Appliances again with the big play button.

Reconfiguring ports

After the switches have booted up it's time to reconfigure the interfaces. First we need to adjust the IP range of the VLAN associated with `swlp1` on the original switch so that we don't have to reconfigure the server also. Let's have the switches talk to each other over the 172.16.0.0-255 range. To do this, edit the `br0` definition in **/etc/network/interfaces**:

```
auto br0
iface br0 inet static
  bridge-ports swlp1
  bridge-stp off
  address 172.16.0.254
  netmask 24
```

Then simply restart the interface.

```
$ ifdown br0; ifup br0
```

Now let's take a look at the configuration for the new server switch. We connected `swlp1` from the desktop switch to the `swlp1` port on the server switch. So we need to add a config for that port that's in the same range, for example:

```
auto br0
iface br0 inet static
  bridge-ports swlp1
  bridge-stp off
  address 172.16.0.253
  netmask 24
```

Next we need to configure the server switch port. We'll create a VLAN-aware bridge and enslave the port so that the server is connected to the bridge, and then enable L3 on that VLAN.

```
auto br1
iface br1 inet static
  bridge-ports swlp10
  bridge-stp on
  address 10.1.0.254
  netmask 24
```

Once we've saved that config we can bring those interfaces up and test the connection to the server:

```
$ ifup br0
$ ifup br1
$ ping 10.1.0.1
PING 10.1.0.1 (10.1.0.1) 56(84) bytes of data.
64 bytes from 10.1.0.1: icmp_seq=1 ttl=64 time=0.312 ms
64 bytes from 10.1.0.1: icmp_seq=2 ttl=64 time=0.314 ms
```

You might have noticed we haven't put any routes in the switches. As a result we can't ping from the server to the desktops or vice versa. Because we don't want to end up managing a growing mess of static routes, we're going to use OSPF to safely distribute and synchronise our internal routing tables. As we weren't using VRFs in this tutorial, setting up OSPF via *quagga* is quite simple. Start by entering the *quagga* terminal:

```
$ vtysh
```

The brings us into a read-only mode, where we can use the show command to inspect the various routing tables and links. To enter a write mode so that we can add our OSPF config, run the following:

```
$ configure terminal
```

Now we're in the configure mode enabling OSPF and distributing our locally attached routes is as simple as:

```
$ router ospf
$ ospf router-id 172.16.0.1
$ network 0.0.0.0/0 area 0.0.0.0
$ redistribute connected
```

Then we need to exit the router config, configure mode, and vty so run the `exit` command three times.

```
$ exit
Devuan-NOS(config)#
$ exit
Devuan-NOS#
$ exit
```

Now repeat this on the client switch, replacing the router-id with the ip address of the `br0` bridge, and we should be able to ping from the server to the desktops again.

This may seem a little overkill for a network with only one server and two switches, but we have laid the groundwork for building a fully meshed Clos network. This set-up, with only a few minor changes, is scalable to the extent of even the most demanding enterprise environments without significant increases in latency or decreases in throughput.

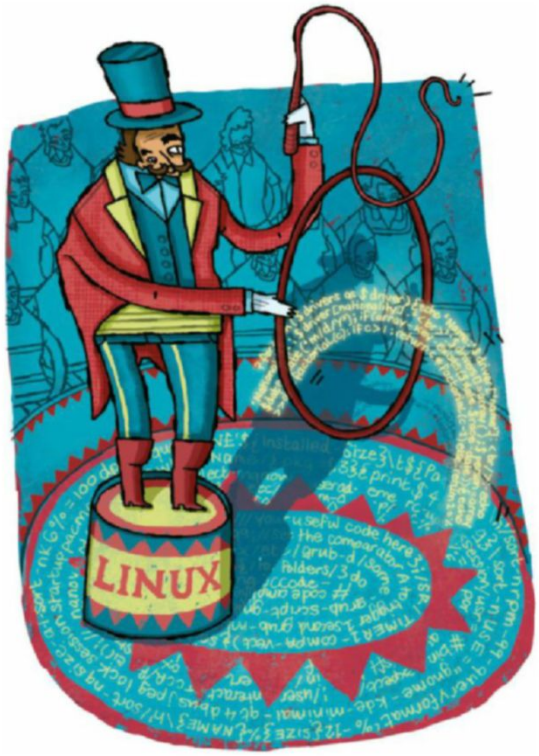
After around 30 seconds the routing table should be synchronised and you'll be able to test the connections. This is best done with a nice cuppa and a few choice biscuits. LXF

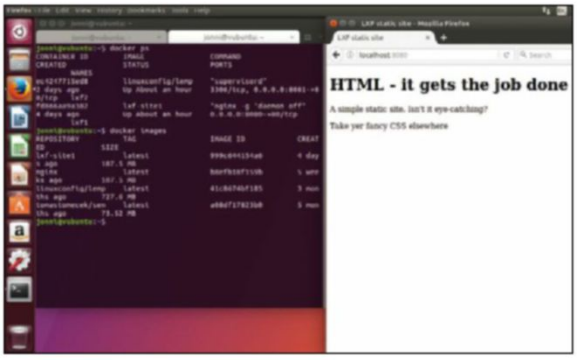
Docker: Isolated web servers

Jonni Bidwell finds Docker containers running tasty Nginx containers much more preferable to the cordial discussion at tupperware parties…

Our expert

Jonni Bidwell loves to dabble with Docker. The number of times he's wiped out his whole system has been vastly decreased since embracing the new technology





You could spice up this basic webpage by going full Geocities and adding animated gifs, making judicious use of the <blink> and <marquee> tags.

We've covered setting up webservers like Apache (see **LXF197**) and Nginx (pronounced engine-x, see **LXF222**) before. It's really not that hard and it's edifying to do on your own infrastructure, sidestepping the usual template-driven webhosts. But once your friends see what you've done they'll want you to host their website, and their friends websites, and soon you'll be the web hosting king. Either that or you'll have no friends left on account of none of them knew what they wanted and you told them to go bother some other chump.

Or maybe neither, but if you do find yourself hosting multiple websites or applications on the same server then it's a reasonable idea to isolate them from one another in some way. There's nothing particularly advanced about having different websites on the same box: once you have DNS records correctly set up then any old webserver, with some gentle massaging of configuration files, will happily serve the right pages (based on the URL) from a single host.

This is a little more complicated with HTTPS, because the server needs to present a certificate before it knows which site is being requested. We won't get into this, but there's a solution in the form of the server name indication (SNI) extension to the TLS protocol. HTTPS or not, the problem is that having lots of sites in one place is potentially risky – , especially if they belong to different people.

Ideally, none of the sites that you host will be compromised, but if one is and is done so severely enough, then potentially everything on the server becomes vulnerable. There are measures that can be taken to mitigate against this, ranging from the draconian (such as refusing to host anything but static html) to the more permissive (using PHP FPM pools to segregate script access, or *Firejail* to lock things down a bit more).

Yet we're still riding the "containerise all the things wave", and hosting multiple websites is a great way to introduce containers. For good measure, we'll also put our containers behind an Nginx reverse proxy. We could, of course, put that in a container too, but let's not go crazy (*no, let's not – Ed*). We're going to use Ubuntu Server 17.04 for this tutorial, but the basics will be the same whatever your distro. We'll need Nginx and *Docker*. The former is easy:

```
$ sudo apt install nginx
```

The version of *Docker* in the Ubuntu repos (the package is called `docker-io` – the one called `docker` is a stylish application launcher) is pretty old. It more or less works for this tutorial, but it's probably a better idea to follow the instructions in the box and get the latest and greatest. You'll find instructions for other distros at **https://docs.docker.com/engine/installation/#server**.

We'll start with the plain Nginx image from Dockerhub, which just needs to be pointed to a static html directory. Start the docker daemon and grab this with

```
$ sudo systemctl start docker
$ sudo docker pull nginx
```

Quick tip

There are official Docker images for Wordpress, Nextcloud and so on that you can use, but it's fun to build out from less complete images to further your understanding.

Installing Docker on Ubuntu

Because *Docker* development is pretty rapid, distribution repos struggle to keep up and indeed, most distros have given up trying. But don't worry, there's no need to build it from source every time there's a new release.

Docker maintain repositories for Ubuntu LTSes (14.04, 16.04) as well as the latest 17.04. Note that Docker isn't available for 32-bit machines. You probably will have these packages already, but it doesn't hurt to be sure:

```
$ sudo apt install apt-transport-https
ca-certificates curl software-properties-common
```

Next we add the *Docker* GPG key:

```
$ curl -fsSL https://download.docker.com/linux/
ubuntu/gpg | sudo apt-key add -
```

Now you need to verify the key's fingerprint. It should end in `0EBF CD88`, if it doesn't then either something sinister is afoot (*no need to be so dramatic – Ed*) or a new key has been issued since we wrote this article. Check the official docs either way.

```
$ sudo apt-key fingerprint 0EBFCD88
```

Now we can add the *Docker* repo. Replace zesty with the codename of your Ubuntu version (for example, `trusty` or `xenial`). You can put all of this command on one line, but we've use slashes to try and make it look neater (*try harder next time – Ed*):

```
$ sudo add-apt-repository \
"deb [arch=amd64] https://download.docker.
com/linux/ubuntu \
zesty stable"
```

Finally we can update the package indices and install *Docker Community Edition*:

```
$ sudo apt update
$ sudo apt install docker-ce
```

To host Wordpress (shudder) sites, we'd need to add PHP and MySQL type things to this image, but we won't do that. That wheel has already been invented so we'll use a different pre-fabbed LEMP container later in the tutorial to host some PHP. For now, we'll create a directory for our static site:

```
$ mkdir -p ~/site1/html
```

Nobody writes HTML by hand anymore, so this next bit may serve as a history lesson for some. Create an index file with the following:

```
$ nano ~/site1/html/index.html
```

and populate it with some good ol' fashioned markup:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title> LXF static site </title>
  </head>
  <body>
    <h1> HTML – it gets the job done  </h1>
    A simple static site. Isn't it eye-catching?
  </body>
</html>
```

The HTML5 specifying header character set are optional, but if you're going to write HTML, it may as well be by the book. Save this with Ctrl-X, Y. It doesn't really matter that this file (and the rest of the **site1/** directory) is 'outside' the container that's going to use it: from a security point of view the important thing is that it's 'outside' the container which isn't going to use it. We are, however, going to have to tell *Docker* to point our container at the **site1/** directory. Nginx usually serves pages from **/usr/share/nginx/html/** so we need to map this directory inside the container to our **site1/html/** directory on the host. We could do this with a bind mount from the command line (using the -v switch) every time we bring up the container, but it's tidier to make a new image based off the old one.

Create a simple Dockerfile with `nano ~/site1/Dockerfile` and add the following to it:

```
FROM nginx
COPY html/ /usr/share/nginx/html
```

Now we can create and tag our new image with:

```
$ cd ~/site1
$ docker build -t lxf-site1 .
```

Next, we can fire up a container built from this image and check that everything works:

```
$ sudo docker run --name lxf1 -d -p 8080:80 lxf-site1
```

It's good to name your containers (and probably to do so differently from the underlying image) because it makes them easier to keep track of (for example, via the `docker ps` command). The -d flag tells *Docker* to detach and we've used -p to forward port 80 on the container to port 8080 on the host. Thus, all going to plan, pointing your web browser at **http://localhost:8080** should display the index page we constructed earlier. Later on, we'll set up a reverse proxy on the host so that appropriate traffic can be marshalled to this site based on the hostname requested. For now, we'll set up another website in another container.

Contain yo'selves

As you might imagine, a ready to go LAMP (or LEMP in our case, E for nginx – remember our pronunciation lesson earlier) stack is pretty handy for all manner of projects. As a result there are a number of pre-rolled images. We'll use the one from **www.linuxconfig.org**, which will give us a Debian 9 base, nginx, MariaDB and PHP 7. Delicious. Grab it with:

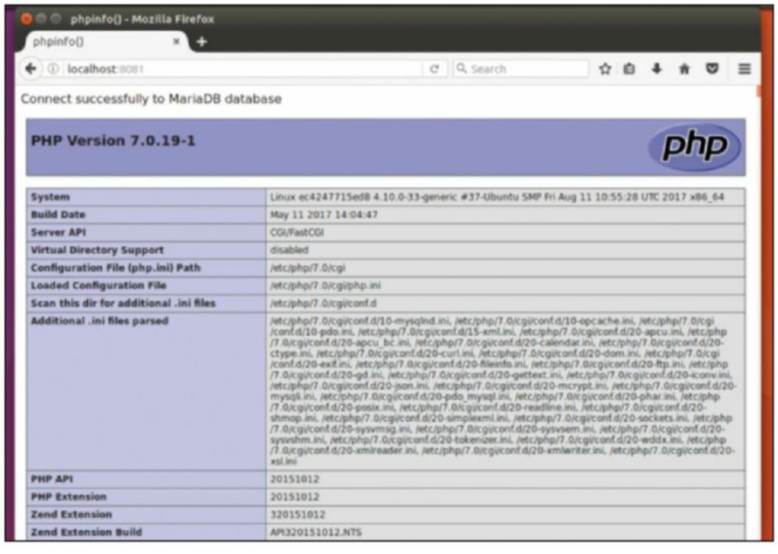
```
$ sudo docker pull linuxconfig/lemp
```

This one's a couple of hundred megabytes so will take a while. As before, we'll create a directory for our other website and another to store our database.

```
$ mkdir -p ~/site2/html
```

Quick tip

For more information about Docker volumes, check out **www.nschoe.com/articles/2017-01-28-Docker-Taming-the-Beast-Part-4.html** and indeed the rest of the Docker: Taming the beast series.



It's nice to know that our database and PHP are working. Having the database in a volume means it can be used with another container.

```
$ mkdir ~/site2/mysql
$ cd ~/site2/html
$ nano index.php
```

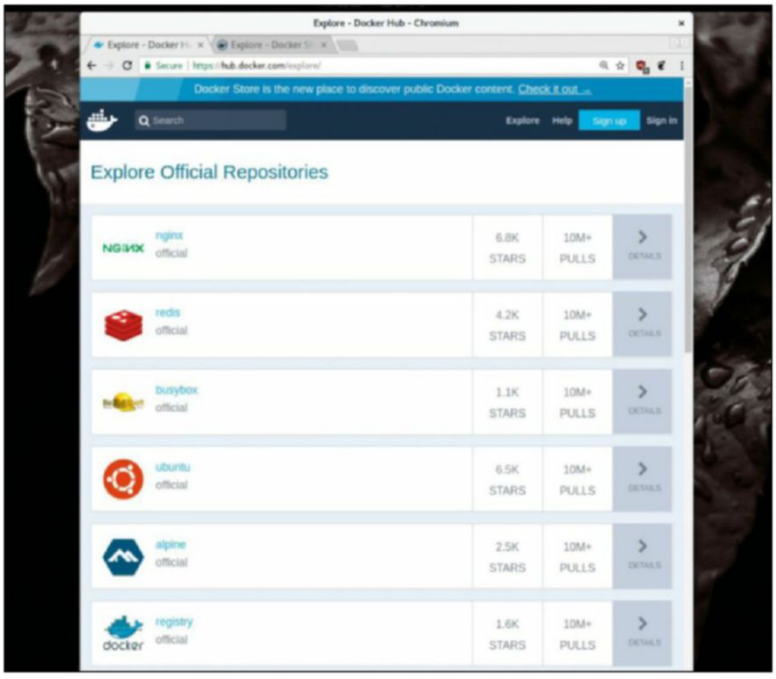
We'll write some lovely PHP to check that this container is working. The container actually includes a file similar to this one, but we'll make our own (because everyone loves writing PHP):

```
<?php
$dbh = mysql_connect('localhost', 'admin', 'pass');
if (!$dbh) {
    die('Could not connect: ' . mysql_error());
}
echo 'Connected successfully to MariaDB database';
mysql_close($dbh);
?>

<?php
phpinfo();
?>
```

Save and exit as before. The admin account with password `pass` is preconfigured with the image. This time we'll use volumes because they're the least fussy way of having a persistent database. Data in our container will survive the container being stopped and restarted, but ideally we'd like to be able to spin up the image (or another like it) somewhere else and easily move the data to it. We could use a designated directory on the host for our database (for example, **~/site2/mysql/**), but we would run into all kinds of permissions issues (translation: we tried this and weird stuff happened that we couldn't fix). We'll forward port 80 on the container to port 8081 on the host this time.

```
$ cd ~/site2
$ sudo docker run --name lxf2 -d -p 8081:80 -v $PWD/html:/var/www/html -v mysql:/var/lib/mysql linuxconfig/lemp
```

Again, you'll get a big long hex string that identifies your container, but it's easier to refer to it by the name we've bestow'd upon it, `lxf2`. After a few seconds this container should be accessible via your web browser at **http://127.0.0.1:8081**. Hopefully, you'll be greeted with a successful database connection message and some lovely PHP debug info.



You'll soon discover that the Docker hub hosts more images than you can shake a pointy stick at. They're all just a docker pull away.

Here we've bind mounted the **~/site2/html** directory inside the container, which means we can modify it from the host and changes should be reflected instantly. Depending on your purposes you may wish to build a new image and copy this directory into it, as we did earlier. We've also created a volume called `mysql` and we can check the details of this newly created volume with:

```
$ sudo docker volume inspect mysql
```

and this tells us that the volumes are stored in **/var/lib/docker/volumes**.

You can put anything you like in the **html/** directory – your favourite webapp (Wordpress, phpBB, Nextcloud, for example), or write your own PHP site with its own database, the possibilities are endless. Containers are strange things to get your head around at first: a lot seems unintuitive or inaccessible. But they're your friends really, and if it helps you feel more at home, then navigate them as you would a normal system. Just do:

```
$ docker exec -it lxf2 /bin/bash
```

The `exec` keyword runs a command inside the container, the `-i` option stipulates interactive mode (so that the process doesn't exit immediately) and `t` allocates a pseudo-TTY. Together the two options make the container behave like a traditional terminal. Containers by design tend to be quite minimal, so you won't find a `systemd` running as PID1 in there. Nor will you find lots of the command line tools you may be used to. Of those that are there, many will not function correctly, for example, `top`, because various environment settings haven't been defined. But you can still poke around, and hopefully convince yourself that there's not too much voodoo going on. Exit the container by typing, most unimaginatively, `exit`.

Perverse Roxy

So far we have created two containers each hosting fairly primitive websites which we can access on our host machine's ports 8080 and 8081. We also installed nginx on the host, so (knowingly or otherwise) we're also serving a 404 Not Found page on the host. Visit **http://localhost** if you don't believe us.

When Nginx first came into being its modus operandi was not as a webserver, but as what's known as a reverse proxy. The idea of a proxy is reasonably well known: it's something that acts as an intermediary between two things. In networking terminology, it's a server that relays requests from one machine to another. This could be because one machine is behind a firewall and needs the proxy to talk to the outside world, or it could be, as in the case of VPNs, to obfuscate the IP address of the original machine.

At any rate, a reverse proxy is more or less the inverse of both of these, acting as an intermediary for a group of associated servers rather than associated clients. This is good news because it's just what we need in order to make both of our docker-ised web servers accessible without having to resort to port numbers. Besides routing traffic appropriately, a reverse proxy may be used as a load balancer, distributing traffic to servers hosting the same

Like Docker? Like Minecraft? Then you'll love Dockercraft. Just don't use it on production systems...

content, or as a cache for static content, so those servers don't get unduly hammered.

Rather than serving a 404 page, we'd like our nginx instance on the host to direct traffic to the appropriate nginx container. The marshalling will be based on the hostname requested by clients connecting to our host. It's not necessary (but this is how it would work in the real world) to have DNS records pointing to your host's IP address, be they paid for registrations or a free service such as Duck DNS. Instead, we can use a well-known web developers trick of putting a couple of entries in our **/etc/hosts** file. This is checked before Linux attempts to look up DNS records externally so, as far as our machine is concerned, we can give it any hostname we want. We'll use the dummy .local domain to keep things simple. Add these lines to the hosts file with `sudo nano /etc/hosts` – you can use a vanity name such as **myawesomeserver.com** if you really want:

```
127.0.0.1 lxfdocker1.local
127.0.0.1 lxfdocker2.local
```

Save this and now you should find that both those domains now resolve to our host, so that visiting either of them displays the 404 page again. Progress, but not quite what we want. We need to tell Nginx on the host about our two domains. For our static site do `sudo nano /etc/nginx/conf.d/lxf1.conf` and fill in the following:

```
server {
        listen 80;
        server_name lxfdocker1.local;
        location / {
                proxy_pass
http://127.0.0.1:8080;
        }
}
```

Save this file and repeat for our other container, changing the `server_name` to lxfdocker2.local and the `proxy_pass` to **http://127.0.0.1:8081**. Now restart *nginx* with

```
$ sudo systemctl restart nginx
```

You should find that **http://lxfdocker1.local** refers us to our static site, and **http://lxfdocker2.local** visits our LEMP container. Fantastich.

For fully featured proxying, you should look at some `proxy_set_header` options in the .conf file, in particular `X-Real-IP` and `X-Forwarded-For`. These help pass the client's details through our reverse proxy, which may be required for more advanced web apps. You may also wish to use IP tables or another firewall to disable direct access to our websites via our host's external IP to ports 8080 and 8081. There's no real harm in leaving them accessible at this stage. As long as they are accessible via the loopback address things will still work.

Finally do check out some of the amazing things other people are doing with *Docker* at the Awesome Docker page: **https://github.com/veggiemonk/awesome-docker**. LXF

Monitoring Docker containers

If you extend this tutorial and develop fully featured websites in your containers, then you'll want to keep an eye on how much resources each container is using. You could do this by logging into each container and parsing output from `ps` or other traditional commands, but that seems painful. You could also modify the image to include Munin or Awstats, which would give access to all kinds of information via a web page. But it's nice to keep our containers simple. Running a Munin node, say, would require setting up an RRD database, and generating a lot of data, which inside the container could cause more problems than it solves.

You can get some rudimentary statistics with a simple:

```
$ docker stats
```

This will show you CPU, memory, network I/O and block device I/O for each container, which is enough to get a rough handle on what's agwarn and see if anything needs remedying.

There are also countless projects for more in-depth monitoring. One such is the excellent `sen` from Red Hat's Tomas Tomecek. It's available as (you've guessed it) a docker image that you can run from your host as follows:

```
$ sudo docker run -v /var/run/docker.sock:/run/
docker.sock -ti -e TERM tomastomecek/sen
```

The initial screen shows the status of any containers currently running (including the `sen` container itself, whoa) and you can select one of them using the cursor keys and Enter to find out more specifics and view some nice ASCII graphs. It's an awesome tool, and you should read more about it at **https://github.com/TomasTomecek/sen**.

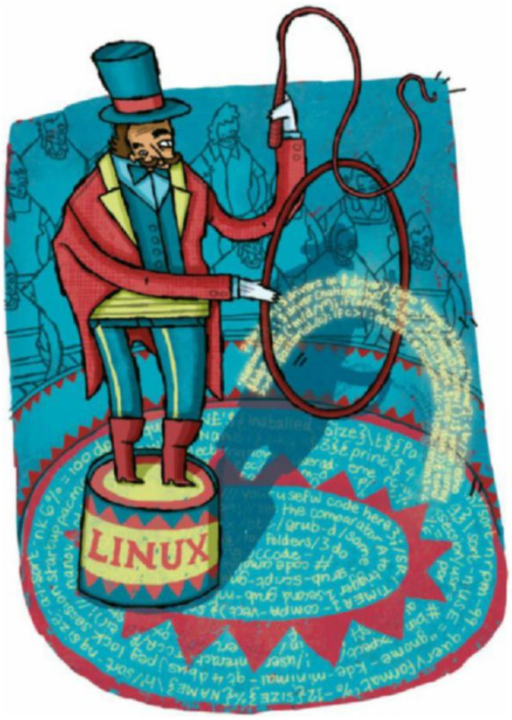
Finally, we couldn't do this without mentioning the wonderful (though possibly impractical) Dockercraft, which enables you to control containers from the comfort of *Minecraft*. Find out more at **https://github.com/docker/dockercraft**.

Kotlin: Creating commands

Mihalis Tsoukalos shows you how to start using Kotlin by explaining basic concepts so you can get stuck in with this interesting language...

Our expert

Mihalis Tsoukalos is a UNIX administrator, a programmer, a DBA and a mathematician who enjoys writing articles and learning new things in life.



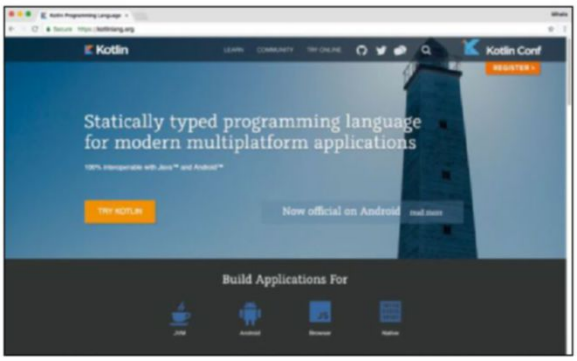


Figure 1 shows the initial screen of the official Kotlin web site, which you should regularly visit to get information on the latest updates about the programming language.

This is an introductory tutorial to the Kotlin programming language. After reading this article you'll be able to install Kotlin, and know how to execute Kotlin code and understand the structure of Kotlin programs.

However, because this is the first tutorial in the Kotlin series, it won't go into too much depth. Forthcoming tutorials will explore many important Kotlin topics including Kotlin data types as well as object-oriented programming, systems programming and web development, so make sure that you understand the concepts presented in this month's tutorial. The best way to take advantage of this advice revealed here is to try to develop your own Kotlin programs using the presented code as a point of reference. After all, you can only learn programming by writing code!

The Android connection

The big news is that Kotlin can be officially used for developing Android applications, which automatically makes Kotlin a major player in the programming languages area. What's important to understand is that Kotlin uses the Java Virtual Machine (JVM), which means that Java is still there for you and that your existing Java knowledge is still valid, even if that happens behind the scenes! Additionally, Kotlin makes use of the standard Java libraries.

If you like Java, you'll most likely enjoy developing software in Kotlin, which is much easier to learn. However, if you're not a big fun of Java, you might still like Kotlin, so give it a try – you won't regret it! First of all, Kotlin code is clean and simple, which means that it's also easy to read, in fact much easier to read than Java code. Additionally, Kotlin is a safe programming language that can save you from null pointer exceptions. It also offers many advanced features including Static typing, Extensions, Lambdas and Delegated properties. Finally, Kotlin requires less coding compared to Java.

Figure 1 (*above*) shows the initial screen of the official Kotlin site, which can be found at **https://kotlinlang.org**.

Installing Kotlin

Strictly speaking, Kotlin is a statically typed programming language for the Java Virtual Machine. However, none of its characteristics will make sense unless you install it on your Linux machine. The installation process on an Ubuntu Linux machine includes the following steps:

```
$ wget -O sdk.install.sh "https://get.sdkman.io"
$ vi sdk.install.sh
$ bash sdk.install.sh
```

The second step is optional and you'll most likely not need to make any changes to **sdk.install.sh** so don't worry about it – you can just look at its contents. The third step is where the actual installation of Kotlin starts taking place.

After performing the third step, you're advised to execute the next command, which is for setting up the Kotlin environment for the current user:

Quick tip

You can find more about Kotlin at **https://kotlinlang.org**, which is the official Kotlin site. If you don't want to install Kotlin right now, you can still try it at **https://try.kotlinlang.org**. *Programming Kotlin* and *Kotlin in Action* are two very good books that will get you up to speed with the language.

Kotlin data types

Kotlin supports basic data types such as integers, floats, strings and doubles using the Int, Float, String and Double keywords, respectively. Moreover, Kotlin supports arrays with the help of the arrayOf() function. So, in order to create an array of integers named myInts, you should employ the following Kotlin statement:

```
>>> val array = arrayOf(1, -1, 2, -2, 3, -3)
```

Additionally, Kotlin enables you to create arrays using a function that's used for generating each element of the array:

```
>>> val arrayFunction = Array(5, { i -> i * 2} )
>>> println(arrayFunction[3])
6
```

Kotlin offers its own array classes for the primitive types. As a result, we have ByteArray, CharArray, ShortArray, LongArray, IntArray, BooleanArray, DoubleArray and FloatArray. Worth mentioning is the Unit type that's used in the section of this tutorial, which talks about functions. The Unit type is equivalent to the void type that can be found in C and Java programming languages.

The Kotlin tutorial of the next issue of *Linux Format* will clarify many things about Kotlin data types, so be sure to catch it!

```
$ source "/home/mtsouk/.sdkman/bin/sdkman-init.sh"
```

So far you've just installed the *sdk* tool that will help you install Kotlin proper:

```
$ sdk install kotlin
```

Because Kotlin needs a Java installation, you might need to execute `sudo apt-get install openjdk-8-jre-headless` on your Ubuntu machine. After that, you'll be ready to start writing Kotlin programs. But first, you can find the version of Kotlin you're using by executing the next command:

```
$ kotlinc -version
info: kotlinc-jvm 1.1.3-2 (JRE 1.8.0_131-8u131-b11-2ubuntu1.16.04.2-b11)
```

As you can see, the Java version you're using is also mentioned in the previous output because Kotlin is based on Java. This also means that you might see some Java error messages while writing software in Kotlin.

Now that you've Kotlin successfully installed, it's time to start looking at some Kotlin code!

Hello Kotlin!

This section will gently present you some Kotlin code and the various methods of executing it – the *Hello World* program will be used as an example. The Kotlin version of the *Hello World* program is as follows:

```
package hw

fun main(args: Array<String>) {
  println("Hello World!")
}
```

Kotlin code is usually included in packages because packages enable us to split Kotlin code into separate namespaces – in this case the name of the package is **hw** and contains a simple function called `main()`, which is where the execution of an autonomous *Kotlin* program begins. Although using packages is a good practice, you're not obliged to get involved with them.

As you can also see, semicolons are optional in Kotlin, which is also true for most modern programming languages. Additionally, you can see that you can print output on the screen using the `println()` function. There's also the `print()` function that does the same job as `println()` without automatically inserting a new line character at the end of its output. Finally, by looking at the way the `main()` function is defined you can tell that function's definitions in Kotlin start with the fun keyword.

Please note that the same program can be written using objects and classes, which will be presented and explained in a forthcoming tutorial.

If you save the previous code in a file named **hw.kt**, then you can compile it into a JAR file as follows:

```
$ kotlinc hw.kt -include-runtime -d hw.jar
$ file hw.jar
hw.jar: Java archive data (JAR)
$ ls -l hw.jar
-rw-rw-r-- 1 mtsouk mtsouk 865307 Jul 31 11:00 hw.jar
```

The `-include-runtime` command line option instructs the compiler to create an autonomous and runnable JAR file by telling the compiler to include the Kotlin runtime into the generated file. This is the main reason why the **hw.jar** file is so big. The `-d` flag that's also used enables you to specify the name of the JAR file which is going to be generated.

When the previous command is executed successfully, it generates no output on your screen. After that, you can execute *hw.jar* using Java in order to see the desired output:

```
$ java -jar hw.jar
Hello World!
```

Because Kotlin supports scripts, you can also make *Hello World* program a Kotlin script, named *hw.kts*:

```
println("Hello, world!")
```

So, when you're creating a Kotlin script, there's no need to have a main() function. Next, you can execute *hw.kts* as follows:

```
$ kotlinc -script hw.kts
Hello, world!
```

The only difference here is initiated by the `-script` option.

So far, you've learnt how to execute Kotlin code in two different ways. Next, you'll learn is that Kotlin offers an interactive shell that you can enter if you execute *kotlinc-jvm*. »

Quick tip

Kotlin code can be provided as a regular source file or as a script – the differences between the two types of Kotlin files are minor. Files with Kotlin code have the .kt extension whereas Kotlin scripts have the .kts extension.

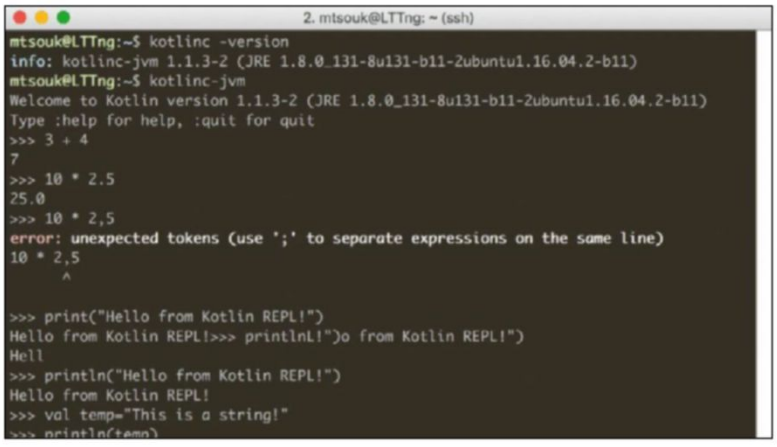


Figure 2 shows a small interaction with the Kotlin REPL, which is an interactive Kotlin shell where you can try things without fear of breaking anything!

Figure 2 (*on the previous page*) shows an interaction with the Kotlin interactive shell. The interactive shell, which is also called REPL, is the perfect place to try new things, make mistakes and learn – so don't hesitate to use it.

Variability

As you might expect, Kotlin enables you to define new variables, change the value of old ones as well as combine existing variables to create new ones.

First of all you should be aware that Kotlin supports two kinds of variables: mutable and immutable. Mutable variables are declared using the `var` keyword and immutable variables are defined using the `val` keyword. An immutable variable must be initialised when they're created, because their value can't change afterwards.

The following interaction will take place in the REPL:

```
>>> var str1 = "This is "
>>> var str2 = "a String!"
>>> val s1s2 = str1 + str2
>>> println(s1s2)
This is a String!
```

The aforementioned code defines three string variables named str1, str2 and s1s2 and demonstrates that you can link strings together using the + operator.

Figure 3 (*below*) displays further examples of variable definitions that sit inside the Kotlin REPL. As you can see, if you attempt to change the value of an immutable variable, you'll receive an error message, which is the case with the s2 variable.

Need input!

Being able to obtain user input is very important; so we're going to show a way of getting information from users in Kotlin. The name of the program will be *userInput.kt* – its most important code is the implementation of the `main()` function:

```
fun main(args: Array<String>) {
    println("Enter Two numbers:")
    var (a, b) = readLine()!!.split(' ')
    println("Min of $a and $b is: ${findMin(a.toInt(), b.
toInt())}")
}
```

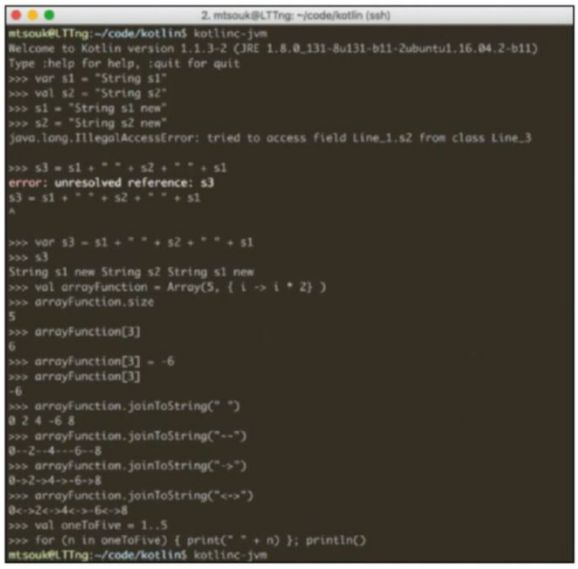


Figure 3 uses the Kotlin REPL to illustrate Kotlin mutable and immutable variables as well as Kotlin arrays.

Quick tip

Although namespaces require a little more work, they permit you to use classes, objects, variables, constants and functions implemented in other packages without conflicts with your own class, object, variable, constant or function names.

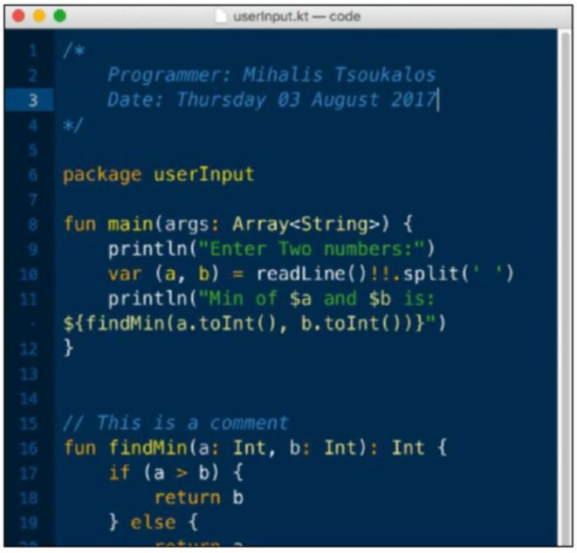


Figure 4 displays the Kotlin code of userInput.kt, which is a program that reads two integers and calculates the smaller integer of the two.

Here you can see a technique for reading two values from the user that will be saved in the a and b variables, using a single Kotlin statement.

If you execute *userInput* you'll have the next type of interaction with it:

```
$ kotlinc userInput.kt -include-runtime -d userInput.jar
$ java -jar userInput.jar
Enter Two numbers:
1 2
Min of 1 and 2 is: 1
```

The Kotlin code of the entire *userInput.kt* program can be seen in Figure 4 (*above*). If you look carefully enough at this screenshot, you'll understand that Kotlin supports two kinds of comments: line and block. Line comments begin with `//` whereas block comments begin with `/*` and end with `*/`.

There are alternative ways to obtain data from the user – the method discussed here is handy when you want to read just two values from the user.

Taking arguments

Let's take a small Kotlin program that finds the sum of its command line arguments, provided that they have valid numeric values. As you show in *hw.kt* and *userInput.kt*, the main() function takes one argument named args that's an array of string variables. These are the command line arguments of your program and are automatically assigned by the compiler, which is also the case in the majority of programming languages.

The name of the next program is *args.kt* and contains the following sample of Kotlin code:

```
fun main(args: Array<String>) {
    if (args.size == 0) {
        println("Please give me at least one argument!")
        return
    }
    println("1st argument: ${args[0]}!")

    for (k in args)
```

Functional and object-oriented programming in Kotlin

Kotlin can achieve many more things with function because Kotlin is also a functional programming language. Functional programming is a programming paradigm where functions can also be used as variables, arguments and return values of other functions. Put simply, functional programming considers functions as first-class citizens.

Kotlin is also an object-oriented programming language. Essentially, everything in Kotlin is an object! Additionally, Kotlin has complete support for encapsulation, polymorphism and inheritance.

The following Kotlin code shows the definition of a simple class using the class keyword:

```
>>> class myClass {
... val s1: Int = 100
... val s2: String = "My String"
... }
```

The presented class, which is called myClass, contains two properties, called s1 and s2. However, as you'll learn in a forthcoming tutorial myClass is far from complete! Please note that methods in classes, myClass has no methods at the moment, and can be private, which means that they can only be called by the other methods of the class, or public, which means that they can also be called from the outside world.

A forthcoming tutorial will talk about the object-oriented capabilities of Kotlin in more detail, where you'll learn how to use existing classes and create more complex objects.

```
    println("$k")
}
```

The first thing this program does is make sure that it's given at least one command line argument. Failing to test this might make your program crash.

You can also see here how you can access array elements in Kotlin: the first element of the args array can be accessed as `args[0]`, the second as `args[1]`, and so on.

However, *args.kt* reveals an additional method of accessing the command line arguments of your program – the second technique uses a `for loop` and won't crash if there aren't enough command line arguments. Nevertheless, if you want to process a particular command line argument, it makes more sense to access it directly.

Compiling and executing *args.kt* will create the following kind of output:

```
$ kotlinc args.kt -include-runtime -d args.jar
$ java -jar args.jar
Please give me at least one argument!
$ java -jar args.jar 1 2
1st argument: 1!
1
2
```

It's now time to see the Kotlin code of the program that finds the sum of its command line arguments – this is presented in Figure 5 (*above right*). If you understand the code of *args.kt*, then you'll have no problem grasping the ideas behind the code of *sumArgs.kt*.

Please note that the `toInt()` function converts a string that represents an integer to an actual integer value – in this case `toInt()` is used to convert the command line arguments of *sumArgs.kt* to their integer values. Although Kotlin has exception handling capabilities that are similar to Java, the relevant code was omitted here for reasons of simplicity and code clarity.

Compiling and executing *sumArgs.kt* will generate the following type of output:

```
$ kotlinc sumArgs.kt -include-runtime -d sumArgs.jar
$ java -jar sumArgs.jar 1 2 3 4 5
Sum: 15
```

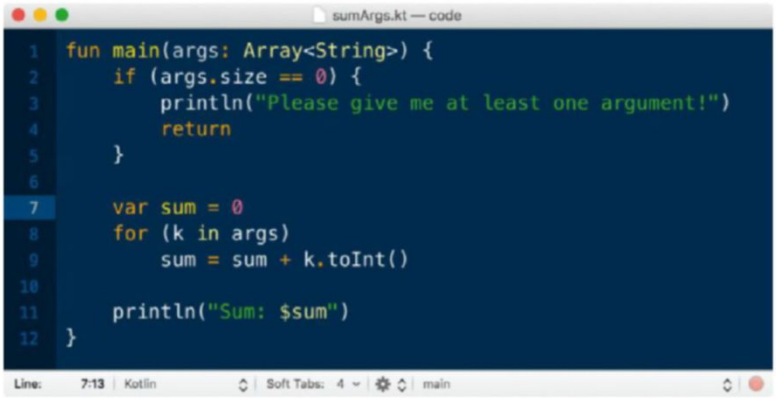


Figure 5 demonstrates the Kotlin code of sumArgs.kt, which is a program that finds the sum of its integer command line arguments.

Functions in

In this section you're going to learn some basic things about Kotlin functions. The function that's going to be implemented here will calculate natural numbers that belong to the well-known Fibonacci sequence. The name of the Kotlin program will be *fibo.kt* and the implementation of the fibo() function is the following:

```
fun fibo(n: Int): Unit {
    var f1 = 0
    var f2 = 1

    for (i in 1..n) {
        print("$f1 -- ")
        val newTerm = f1 + f2
        f1 = f2
        f2 = newTerm
    }
}
```

As you already know, function definitions in Kotlin begin with the keyword 'fun'. Additionally, the type of the return value of a Kotlin function is defined using a colon after the function. If a function has nothing to return, then you can put Unit as its return type. However, you're also free to omit the return type for such functions – that's why the `main()` function presented in *hw.kt* has no return type. Therefore, the next two function declarations are completely equivalent:

```
fun fibo(n: Int): Unit
fun fibo(n: Int)
```

If you compile and execute *fibo.kt* you'll produce the following kind of output:

```
$ kotlinc fibo.kt -include-runtime -d fibo.jar
$ java -jar fibo.jar 10
Fibonacci Sequence: 0 -- 1 -- 1 -- 2 -- 3 -- 5 -- 8 -- 13 -- 21 -- 34
--
```

Kotlin functions can carry out many more things than the ones presented here – wait until the Kotlin tutorial of the next *Linux Format* issue to learn more!

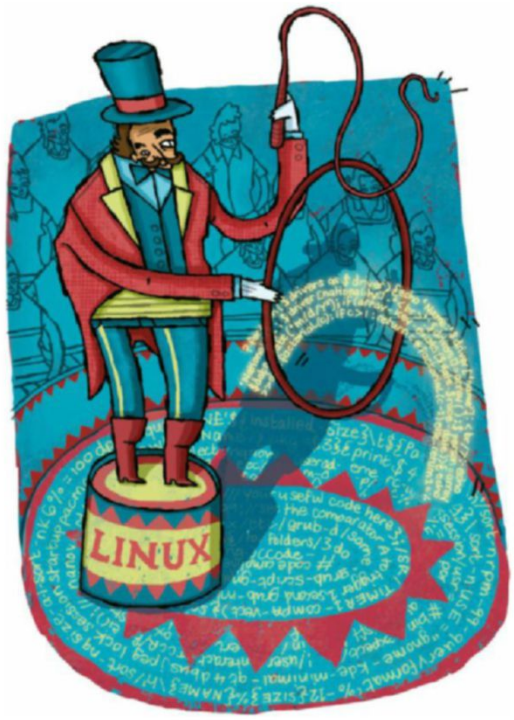
Sooner rather than later, you'll have to try Kotlin, you'll start developing systems software and new Android applications and maybe start rewriting old ones in Kotlin so why not start learning Kotlin today? And if you run into the odd difficulty, don't worry. *Linux Format* is here to help! LXF

Lex: Build a Slack chatbot

Chatbots are everywhere. Amazon Web Services' Lex makes it easy to build and deploy bots to Slack, Facebook and more, as **Dan Frost** demonstrates

Our expert

Dan Frost
Dan works at Cambridge Assessment, experimenting with technology in education. He's a writer and explorer of ideas and technologies. He's often found on Twitter (**@danfrost**) dismissing new fads in computing. Until he champions them.



Bots have a varied presence online, from Facebook messenger to websites, the Quartz news app and on Slack. Done well, they're helpful, can save you time and glue together existing data and infrastructure.

However, building a bot can be fiddly. Thankfully, Amazon Web Services (AWS) released Lex, which is the natural language processing engine and bot system behind Alexa. It enables you to build and deploy a bot in very few lines of code. In fact, once you have Lex wired up to your chat platform iterating and adding features is extremely easy.

To get you started building interactive chatbots we're going to put together a really simple recipe-lookup bot called chefbot. We're going to do this using AWS Lex's platform, with Slack as the chat platform using the serverless framework.

Quick tip

The working code for the tutorial is available on github (**https://github.com/danfrost/lxf-slack**) for you to clone, spin up and fork

Chefbot has the answers

All the code is on github so you can iterate on the code. Chefbot is both simple and generic enough to be a good-starting point for any bot that asks humans questions and looks up data from a MySQL (or any other database).

For this recipe you'll need an AWS account, a Slack account, a MySQL server running somewhere and the Serverless framework (see **LXF228**) installed. We assume that you have admin access to the AWS and Slack accounts.

Before we dive into the code, let's understand what we're about to build. In user interface terms a bot is a computer that sits on a messenger platform (Slack, Facebook messenger) and interacts with humans in a conversational style using text, images and other media. All the interaction is chronological and linear, unlike apps and web pages where the interaction is directed by the user.

In system architecture terms, a bot is piece of code that either responds to a message command or pushes a message to a human user in the hope of a follow-up command. In terms of our architecture here, a python function will be called each time the user sends a message in Slack. This means that we need to create an API endpoint, configure Slack to know about that endpoint and then code the endpoint to respond usefully to messages from users.

But if we're doing anything more complex than responding to literal, perfectly matching strings then we'll have to build a whole stack of natural language processing using machine learning. This is non-trivial, so it's nice to offload the work onto Lex, which does this for us.

Building the Lex bot

Navigate to **https://console.aws.amazon.com/lex/home?region=us-east-1#bots** and create a bot, choosing Custom Bot. Enter the bot name as chefbot, select None for voice, 5 for timeout, select No for COPPA, then click Create.

Now we create an intent which is something the user wants to get out of the bot. This might be a holiday, the answer to a question, the weather tomorrow or anything else that you can answer. For our example, it's going to be a recipe.

The intent might require a few more details which the bot gets out of the user by asking questions, such as "What is the main ingredient?" Each of these extra details fill what Lex calls "slots". Once the intent is clear and the slots are filled, Lex will pass both to our function which should then be able to provide a final response to the user.

First, create a few intents which reflect how our users might express their need for a recipe:

```
Find my a recipe for fish
I want to cook with fish
Fish recipe
...
```

Add each of these by typing the sentence into the utterance input and clicking the + icon.

Next we need to describe the two slots of information that we need. We'll create a dummy slot type called Ingredient

with an example value Fish. Now add one required slot called main_ingredient of type Ingredient and make it required. Then create another slot called cooking_time of type AMAZON.NUMBER and also make it required.

Now, returning to our sample utterances we need to label which of the words in the sentences relate to our slots, since all we're really interested in is getting the slot values. Label fish as an ingredient and the numbers as cooking_time.

(It's possible to use NLP to turn human duration phrases into numbers, but that's outside the scope of this tutorial. Have a look at AMAZON.DURATION and play around!)

We aren't going to bother with a confirmation prompt since what we do with the intent isn't exactly life-changing: we aren't ordering a pizza, taking down a server or moving money between bank accounts. In those more serious situations you would have a confirmation like "Okay – I'm going to push the big red button. Are you sure about that?" before doing it.

For now, leave fulfilment as Return Parameters to Client, which does what it sounds like. We get the slots back so we can see if our little conversation worked.

To test your bot, first click Build to build it and then open the Test Bot dialog in the bottom right of the console. Type one of the sample utterances or a slight variation such as "Got a recipe for fish?" You should see that the slight variations in language are dealt with by the Lex NLP, and after answering a couple of questions you get the slot values back.

Our next task is to plumb in Slack so the same result can be seen there. After that, we'll do something more exciting with the data.

Plumb in Slack

Getting Lex tied into Slack requires copying a few keys from Lex to Slack and vice versa. This is donkey work, but necessary so let's get on with it...

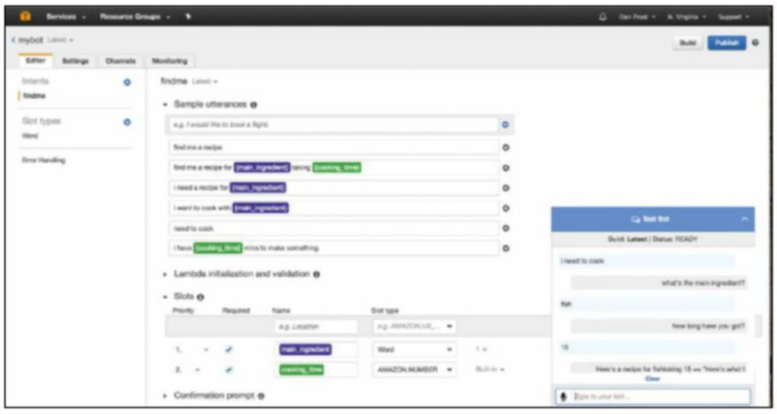
In the Settings tab create a new alias by giving it a name – for example Beta – and selecting a version. Select the most recent version you've built. Click the Channels tab and then click Slack.

You'll need some information from slack first, so open a new tab and go to **https://api.slack.com** and login. Click Add a Bot and then on the next screen click Add a Bot User. On the next screen set the bot name to chefbot and set Always Show my Bot as Online to On and click Add Bot User.

Click Interactive Messages in the left menu and then click Enable Interactive Messages. For now, just put any valid URL in the URL field – we'll come back to this later. Now click Basic Information in the left menu and copy the values for Client ID, Client Secret and Verification token from the Slack interface into the corresponding fields in Lex and, in Lex click Activate.

We now have two values to copy back to Slack: Postback URL, which is used for event subscriptions and interactive messages; and oAuth URL, which is used for the oAuth handshake to authenticate with Slack. Copy these into a text file as we'll need them in a few places.

Go back to the Slack tab (we're nearly done, we promise) and click oAuth Permissions. Click Add Redirect URL and paste in the oAuth URL. Click Save. Now add the scope permissions: `Chat:write:bot, team:read` and save changes. Click Interactive Messages and copy the postback URL from Lex into the Request URL and click Save changes.



The AWS Lex console, where you design your chatbot's conversations. Play around with new phrasings and forms of chat to keep the experience engaging.

Finally (yes, really...), click Event Subscriptions and enable them with the toggle. Paste in the postback URL to the request URL field. In Subscribe to Team Events add message.channels, and in Subscribe to Bot Events type message.im and select the option that comes up. Save the changes.

So that's the config done. Now we need to deploy it to Slack. In Manage Distributions click Add to Slack and then Authorize on the following screen. You're then redirected to the Slack web UI where you can test the bot. Select the bot from the left list of channels and start chatting using the utterances that you configured earlier.

(Before we go any further, if you get stuck or if the process has changed, consult the AWS documentation on integrating Slack, **http://docs.aws.amazon.com/lex/latest/dg/slack-bot-association.html**.)

Okay, this took some boring configuration but the upshot is that you can easily message the bot and obtain a response. We now have Slack sending messages to Lex, Lex working out what we need from the user and then dumping the slots of data back to the user. The only uncool part of this is that the Lambda function isn't doing anything very interesting, so let's solve that next.

Bring our Frankenbot to life

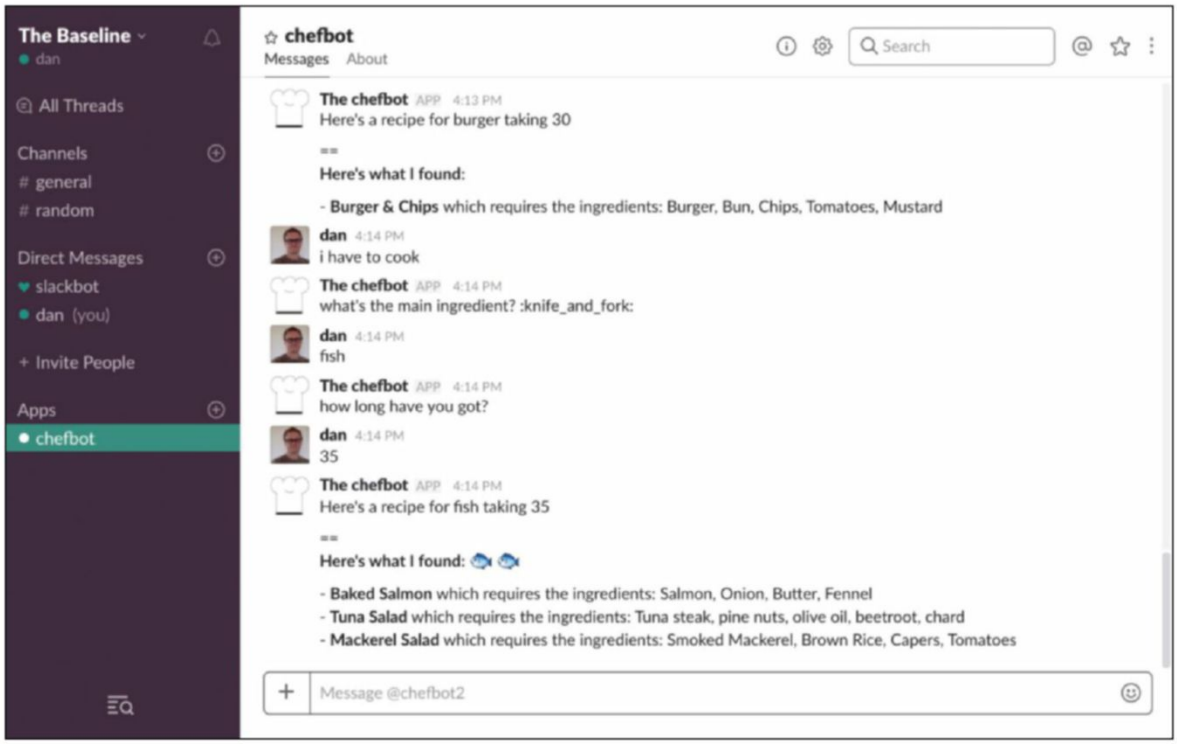
Instead of just dumping the values back to the user, Lex can hand off to Lambda, AWS's Serverless environment. I'm going to use the Serverless framework which does lots of the complex orchestration required to use AWS, so get yourself an AWS account and install the Serverless framework and let's bootstrap the project. We're going to use the Python 3

Who is your bot?

We've been lucky enough to work with some early thinkers working on bots. Like every new digital thing, thinking about bots requires thinking about how it'll be used.

Not all bots are the same. Plan carefully what information you really need from the user and how your bot will come across. Will it be serious or playful? Will it guess if the user isn't sure or be strict about filling the slots? Does the bot need to hand off to another platform if things get too complex?

Thinking about bots as having a personality helps a lot, since the user will be interacting with it as they would interact with a human through chat. This means you limit what you promise to the user and make it clear upfront what the bot is capable of.



Chefbot in action. Simple additions like emoji make a dry bot chat seem fun or more human to deal with.

environment as that's our preferred language these days. Let's kick things off...

```
npm install -g serverless
serverless create --template aws-python --path MyChatBot
```

Now create an IAM profile for yourself and set up your credentials as follows:

```
serverless config credentials -p aws -k XXX -s XXXXX
--profile tutorial-profile
```

Now modify the **serverless.yml** file to contain the following. You can remove all the boilerplate config if you wish.

```
provider:
  name: aws
  runtime: python3.6
  profile: tutorial-profile

...
functions:
  handle_lookup:
    handler: handler.handle_lookup
    events:
      - http:
          path: lookup
          method: any
```

Now add the handler function to **handler.py**:

```
def handle_lookup(event, context):
    logger.info(str(event))

    return {
        'sessionAttributes': event['sessionAttributes'],
        'dialogAction': {
            'type': 'Close',
            'fulfillmentState': 'Fulfilled',
            'message': {
                'contentType': 'PlainText',
                'content': 'Look at my bot!'
            }
        }
    }
```

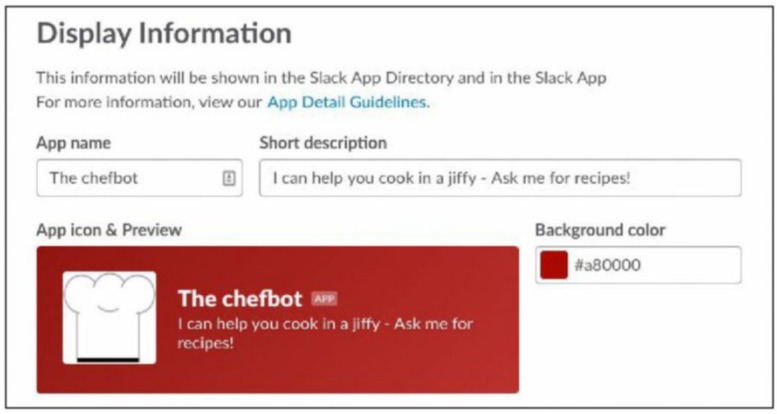
And deploy:

```
serverless deploy -v
```

(At this stage it's worth tailing the log in your terminal: `serverless logs -t f handle_lookup`.)

In the Lex UI, change fulfilment to "AWS lambda function" and select your function from the dropdown. Save the intent. Give it a whirl in slack and you should see the few slot-filling steps performed by Lex and then the final "Whoa!" response from our Python method.

Let's finish off by making this do something interesting.



It's important for people to know what your bot does. Creating a simple icon, adding a colour and a compelling short description will get users chatting.

Pulling in some real data

We've created a simple MySQL dataset that you can put into any MySQL database instance. For the purposes of this, I created an AWS RDS instance, but your MySQL server can be anywhere so long as Lambda can see it. There aren't enough words in the article to get MySQL setup and do the chat stuff

so just do what works for you (we love a challenge!–Ed). All you need is the hostname, user, password and database name and to have the DB open to the internet... (Warning: the method of opening the database to the internet is not fit for production systems. This is just for demonstration only.)

Install the MySQL connector and then the code to connect, select the records and return them. First install the package:

```
virtualenv .myenv
source .myenv/bin/activate
pip install mysql-connector-python-rf
```

Now add the following to the top of the handler file:

```
sys.path.append('.myenv/lib/python3.6/site-packages')
import mysql.connector
```

The first line is because we need to bundle up all dependencies inside the Lambda and then include the site packages directory in our path. The second is a normal Python import statement.

Now we can get down to the task of writing the code to connect and return results. For this example, we're using a plain MySQL connector, but you can employ whichever fancy database connector takes your fancy.

```
def handle_lookup(event, context):
    logger.info(str(event))

    hostname = '...'
    username = '...'
    password = '...'
    database = '...'

    main_ingredient = event['currentIntent']['slots']['main_
ingredient']
    cooking_time = event['currentIntent']['slots']['cooking_
time']

    cnx = mysql.connector.connect(user=username,
password=password, host=hostname, database=database)
    cursor = cnx.cursor(buffered=True)
    query = 'select * from recipes where main_ingredient = %s
and cooking_time <=%s'#.format(main_ingredient, cooking_
time)
    cursor.execute(query, (main_ingredient, cooking_time))

    reply = "\n*Here's what I found:*\n"
    for r in cursor:
        logger.info(r)
        recipe = "\n- *{}* which requires the ingredients: {}".
format(r[1], r[3])
        reply = reply + recipe

    return {
        'sessionAttributes': event['sessionAttributes'],
        'dialogAction': {
            'type': 'Close',
            'fulfillmentState': 'Fulfilled',
            'message': {
                'contentType': 'PlainText',
                'content': 'Here\'s a recipe for ' + main_ingredient +
' taking ' + cooking_time + "\n\n== " + reply
            }
        }
    }
```

Now let's take it for a spin. We've loaded up the recipe database with a few entirely nonsense recipes, but the results should give you an idea of what you can do with bots. Start asking chefbot for recipes and you'll be asked for a main ingredient and cooking time, and get recipes in response. The only piece that's coded is the final query to the database and the response to the user. This is simple, but there's a lot of potential.

The limits of Lex

We've tried Lex on a few projects. For those that follow the question-answer or order-response format it works extremely well, and not having to worry about the natural language processing can leap-frog you into a production bot in hours. However, this model doesn't always work, for example when streaming the firehose of all messages from a channel, or if the bot is going to start the conversation then Lex doesn't quite do everything we need.

We've found that combining Lex with custom integration gives us the flexibility of the custom with the NLP power of Lex. There are a lot of tools for building bots, but they often suffer from limiting what you can do by, oddly enough, enabling you to deploy to a range of platforms.

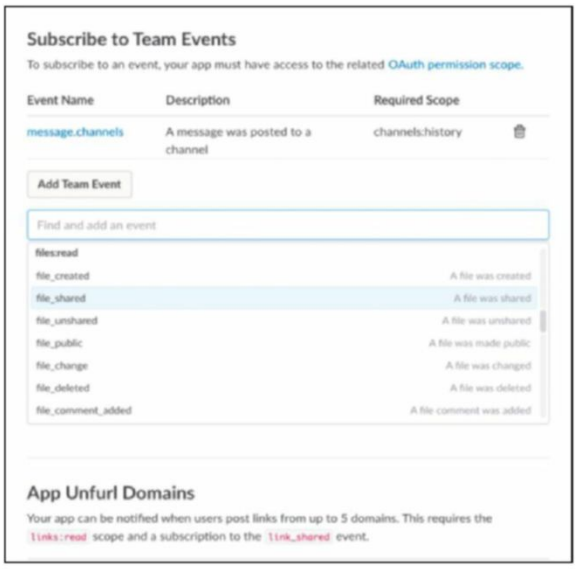
Another point to consider is where you persist data about the user – do you remember all the recipes they've asked about? If so, where? And does every intent need all the data? This is session management and, like on websites, you quickly want to build something to manage that for you.

Where to go from here

In the article we've configured a natural language processor, hooked up a Lambda function to react to the utterances of users and hooked in a MySQL database of recipes via a Serverless Lambda function. All this is far more configuration than programming, so it's important to keep in mind what can be achieved if you take this further.

The deployment we ran had the interaction take place in private human-bot channels, but bots can sit in on public channels as well. This can be useful if you're pulling in data for your team to reference such as "**@issuetrackerbot** how many open issues are there?" or "**@uptime** much much downtime on server X last week".

The interaction we built was also entirely text based with a text input and text output. This is nice for proof of concept, but you can also add card responses which make the process more visual and, on a platform like Facebook messenger, much more engaging. Exploring places such as Google docs, Dropbox, traffic monitoring, billing, social and other data sources can broaden the scope for what just a couple of questions to a bot can do for you and your users. LXF



Experiment with events – we've only scratched the surface of the events you can use here, but it's possible to have your Lambda function react to other data. Play around in the Slack config and see what else you can create.

On the disc

Distros, apps, games, books, miscellany and more...

The best of the internet, crammed into a phantom-zone like 4GB DVD.

Distros

SparkyLinux, one of this month's distros, offers a customised desktop. It enables you to see what else is possible, but there's no need to change distros to obtain a better desktop environment. Most distros tend to come with a single desktop. Yet that's only a starting point, not the final destination.

With Linux, just about everything can be changed. Most aspects of your desktop can be tweaked: wallpaper, icons, fonts and much more. Most of this can be done from the standard settings program, while other changes may need you to install a tweak tool first. Or you can download and install themes for a complete desktop makeover. If that's not enough, there are plenty of other desktop environments to choose: ranging from the full-featured GNOME and KDE to the lightweight Fluxbox and JWM.

The key thing to realise with any distro is that what you first get isn't what you have to use, and customising the desktop is easier than installing a new distro. I'm a long-time KDE user, but I dislike the default setup of KDE. Yet that doesn't matter because I haven't used it for years. Defaults are there to be changed.

Neil

A distro for the security conscious

32-bit

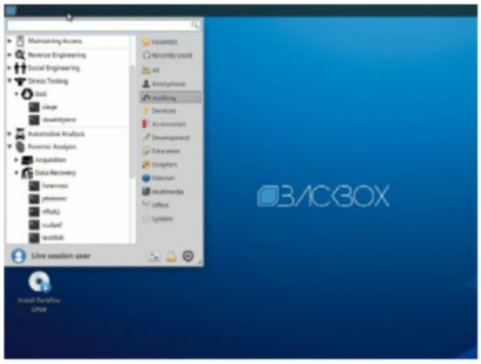
BackBox 5

If you want to follow our features on security this month, BackBox is an ideal distro for you. It's packed with a variety of tools for penetration testing, forensic analysis and all those other things beloved of wearers of monochrome hats. However, it's also a good distro in its own right, because it's based on Ubuntu. That gives it a solid and up-to-date base onto which to build the security tools. The desktop is XFCE, which is lightweight without being too minimal, and enables you to get on with using the computer without getting in the way or making things difficult.

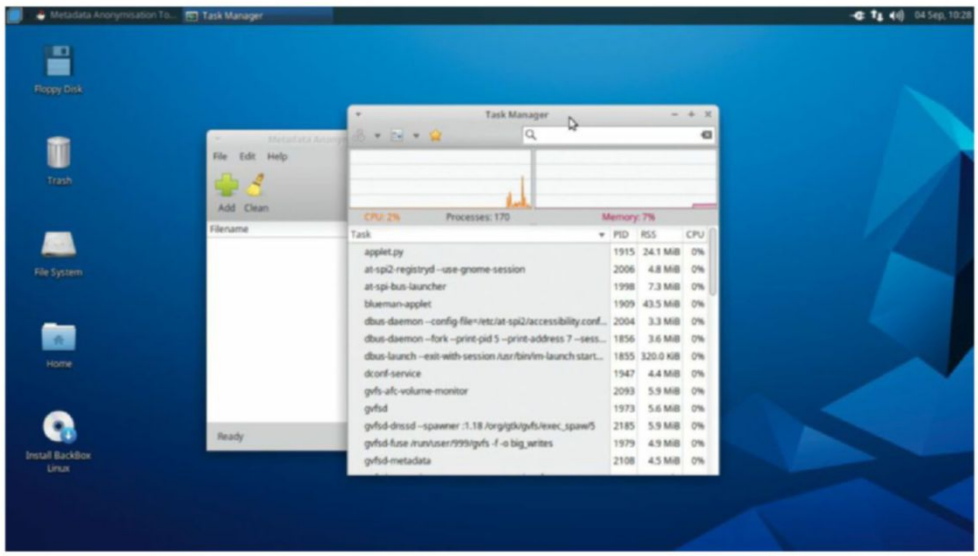
BackBox can be run directly from the DVD, or you can copy the ISO image from the DVD to a USB stick with *dd* for an even more portable version. Running directly from an ISO image means you can be sure that nothing gets changed or corrupted in the OS, which is always important but much more so on a security testing system. It is also possible to use an area of a USB stick as persistent storage, enabling you to preserve settings and data between boots. To do this, use the *Startup Disk Creator* tool from inside BackBox. Alternatively, you can boot from the CD and keep data on a separate USB stick. Either create a filesystem labelled **casper-rw** on the stick or create a file called **casper-rw** in the root of the stick. This will then be used for persistence data storage.

It's also possible to install BackBox to your hard disk in the usual way. The installer provides an option to encrypt the whole installation and we did hit a slight problem here, at least when installed in a virtual machine. If you've set up an encrypted disk then you may see a splash screen asking for a password that doesn't enable you to type anything. If this happens it's safe to reboot and try again as, because this point BackBox hasn't mounted any filesystems and so there's no risk of corruption. When this happened the next time it asked for the password at a normal text prompt, and so all was well. We only saw this on a VM, so it may just be caused by the drivers used for the virtual hardware.

Login details: username **backbox**, and password is left blank.



BackBox comes with plenty of security, auditing and forensic software already installed.



Important NOTICE!

Defective discs

For basic help on running the disc or in the unlikely event of your *Linux Format* cover disc being in any way defective, please visit our support site at: **www.linuxformat.com/dvdsupport** Unfortunately, we're unable to offer advice on using the applications, your hardware or the operating system itself.

New to Linux? Start here

» What is Linux? How do I install it?
» Is there an equivalent of *MS Office*?
» What's this command line all about?
» How do I install software?

Open **Index.html** on the disc to find out

Are you reading this on a tablet?
Download your DVD from **www.linuxformat.com**

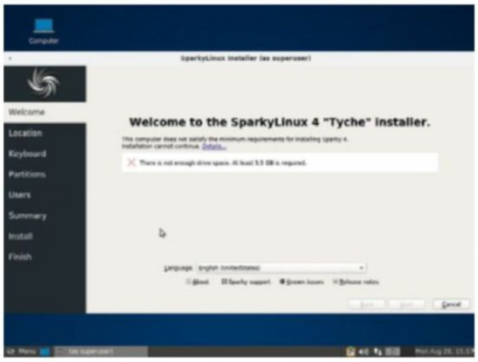
Ideal distro for new open source user

64-bit

SparkyLinux 5

SparkyLinux is another Debian based distro. It uses the testing branch of the Debian repositories, so it features more up-to-date software than the standard Debian experience. It also has a selection of customised desktops to choose from. We've picked the MATE version: this gives a pleasant experience for new users, enhanced by the fact that all the multimedia codecs you should need are already installed. Furthermore, the range of software and desktops available means this is a distro that can grow with you.

Login details: username and password **live**.



Intuitive recovery distro

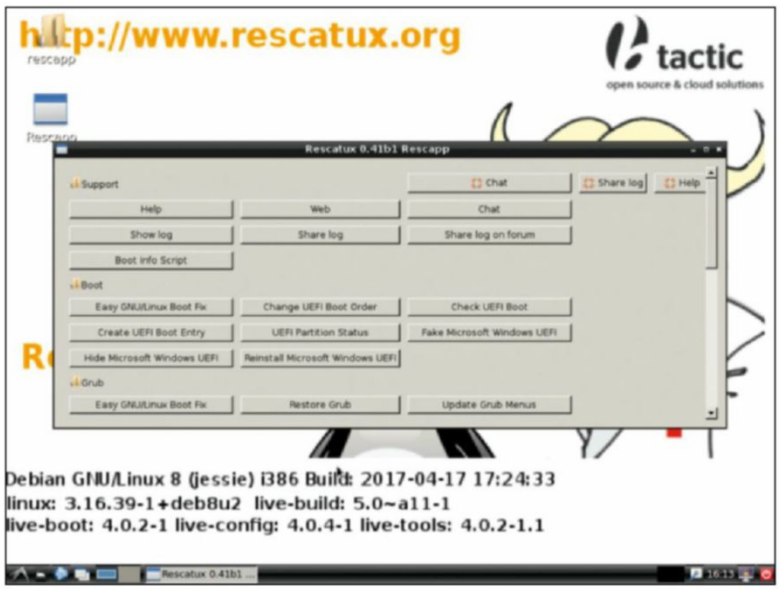
32- and 64-bit

Rescatux 0.41b1

There are a good number of rescue distros available as a live distro, ready to boot into action if something goes wrong with your system. Rescatux is a relative newcomer, but one that's gaining a lot of friends, mainly because it's so friendly. The majority of rescue systems provide all the tools you need to fix most problems, but assume you have the knowledge to use them, or at least know which to use so you can read the man page.

Rescatux is different in that it opens a window with buttons that deal with the most commonly occurring problems. Bear in mind that Rescatux runs commands as root and so it's as easy to make things worse as make them better. Rescatux helps you avoid this by presenting helpful information before giving you the option to proceed with an action, so be sure to read it.

Login details: username **user**, password **live**.



Have Rescatux on standby if your system runs into difficulties.

And more!

System tools

Essentials

Checkinstall Install tarballs with your package manager.

Coreutils The basic utilities that should exist on every operating system.

HardInfo A system benchmarking tool.

Kernel Source code for the latest stable kernel release, should you need it.

Memtest86+ Check for faulty memory.

Plop A simple manager for booting OSes, from CD, DVD and USB.

RawWrite Create boot floppy disks under MS-DOS in Windows.

Smart Boot Manager An OS-agnostic manager with an easy-to-use interface.

WvDial Connect with a dial-up modem.

Reading matter

Bookshelf

Advanced Bash-Scripting Guide Go further with shell scripting.

Bash Guide for Beginners Get to grips with *Bash* scripting.

Bourne Shell Scripting Guide Get started with shell scripting.

The Cathedral and the Bazaar Eric S Raymond's classic text explaining the advantages of open development.

The Debian Administrator's Handbook An essential guide for sysadmins.

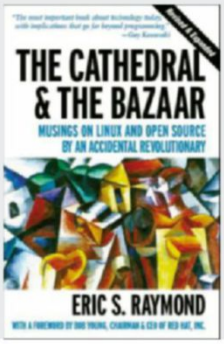
Introduction to Linux A handy guide full of pointers for new Linux users.

Linux Dictionary The A-Z of everything to do with Linux.

Linux Kernel in a Nutshell An introduction to the kernel written by master hacker Greg Kroah-Hartman.

The Linux System Administrator's Guide Take control of your system.

Tools Summary A complete overview of GNU tools.



The secrets of Home Streaming

From Live TV to lossless audio, we show you how to build the ultimate Linux-based streaming solution.

Android apps on Linux

ChromeOS now runs Android apps and so can you! We bring Android apps to your Linux desktop.

Engineering Linux

A professional engineer tries to switch to Linux full-time. You won't believe what happens next!

Master your email server

Time on your hands? Then master your very own email server with web access and more. It's "easy"!

Create 360 VR videos

Edit your own 360 degree VR-style videos, in Linux no less. We outline the tools and the software you need.

Contents of future issues subject to change – we might be too busy boogying in the **LXF** disco dungeon.

LINUX FORMAT
Get into Linux today!

Future Publishing Limited,
Quay House, The Ambury, Bath, BA1 1UA
Email linuxformat@futurenet.com

EDITORIAL
Editor Neil Mohr
neil.mohr@futurenet.com
Technical editor Jonni Bidwell
Art editor Efrain Hernandez-Mendoza
Operations editor Cliff 'No' Hope
Editorial director Paul Newman
Senior art editor Jo Gulliver
Editorial contributors Tim Armstrong, Mats Tage Axelsson, Neil Bothwick, Stuart Burns, Nate Drake, Dan Frost, Matthew Hanson, Adam Oxford, Les Pounder, Mayank Sharma, Shashank Sharma, Alexander Tolstoy, Mihalis Tsoukalos
Cover illustration magictorch.com
Cartoons Shane Collinge

ADVERTISING
Media packs are available on request
Commercial director Clare Dove
clare.dove@futurenet.com
Senior advertising manager Lara Jaggon
lara.jaggon@futurenet.com
Advertising manager Michael Pyatt
michael.pyatt@futurenet.com
Director of agency sales Matt Downs
matt.downs@futurenet.com
Ad director – Technology John Burke
john.burke@futurenet.com
Head of strategic partnerships Clare Jonik
clare.jonik@futurenet.com

INTERNATIONAL LICENSING
Linux Format is available for licensing. Contact the International department to discuss partnership opportunities:
International licensing director Matt Ellis
matt.ellis@futurenet.com **Tel** + 44 (0)1225 442244

SUBSCRIPTIONS & BACK ISSUES
Web www.myfavouritemagazines.co.uk
Email *linuxformat@myfavouritemagazines.co.uk*
UK 0344 848 2852
International +44 (0) 344 848 2852

CIRCULATION
Circulation director Darren Pearce
Tel 01202 586 200

PRODUCTION AND DISTRIBUTION
Head of production UK & US Mark Constance
Production project manager Clare Scott
Advertising production manager Joanne Crosby
Digital editions controller Jason Hudson
Production controller Nola Cokely

MANAGEMENT
Finance & operations director Angie Lyons-Redman
Creative director Aaron Asadi
Art & design director Ross Andrews
Printed by Wyndeham Peterborough, Storey's Bar Road, Peterborough, Cambridgeshire, PE1 5YS
Distributed by Marketforce, 5 Churchill Place, Canary Wharf, London, E14 5HU www.marketforce.co.uk
Tel: 0203 787 9060

The Electronic Frontier Foundation is the leading nonprofit organization defending civil liberties in the digital world. Founded in 1990, EFF champions user privacy, free expression, and innovation through impact litigation, policy analysis, grassroots activism, and technology development. We work to ensure that rights and freedoms are enhanced and protected as our use of technology grows.

EFF.ORG

ELECTRONIC FRONTIER FOUNDATION

Protecting Rights and Promoting Freedom on the Electronic Frontier


LOSE YOURSELF IN A WORLD OF
Vinyl
FIND YOURSELF IN
OXFAM'S ONLINE SHOP
oxfam.org.uk/shop
COLUMBIA
TIME ON
MY HANDS
OXFAM