FASTER Pis
Build better projects by tweaking Raspbian

ARCH DISTROS
We reveal the best Arch Linux to use

LINUX FORMAT

The #1 open source mag

FREE 4.7GB DOWNLOAD
FOR PCs OLD & NEW

ALL-NEW DISTROS
- BODHI LINUX 5.0
- VOYAGER 18.10
- SPARKY 4.9

THE ULTIMATE LINUX TOOLKIT

We pick only the best apps for:
- Photo & video
- File utilities
- Internet
- Office
- Security

TOP 100 TOOLS!

64 pages of tutorials & features

Build a custom USB boot drive with Grub

Get the best audio streaming with Ubuntu

Jorge Salamero
on bringing security to the crazy container world

STAR GAZING
Explore space with open source

FASTER WEB SITES
Build fast and secure websites with Middleman

FUTURE

Managed VPS

All of the power.
None of the hassle.

You take care of your business. We'll take care of your server.

From £39.99 per month

- Fully monitored by our experts
- All updates installed by us
- Automatic site and server backups
- Plesk Onyx control panel
- WordPress Toolkit
- Let's Encrypt included

Call us today about moving your sites for free 0330 660 0255

Heart Internet

Visit: www.heartinternet.uk
Email: sales@heartinternet.uk

LINUX FORMAT

Corporate Patron Free Software Foundation

WHO WE ARE

This issue we're discovering the best open source software out there, so what's your favourite "unknown" open source tool?

Jonni Bidwell
I recently dug up my old Sansa Clip MP3 player (the greatest budget MP3 player of the mid-2000s) which is, or was, not a Linux tool. But then I installed *Rockbox* on it and it became such. Now it can play not only MP3s (and OGGs and FLACs) but also *Doom*, in glorious monochrome.

John Knight
My brother swears by *PDFsam*, which splits and merges PDF files. I don't know about "unknown", but *Qtractor* is very underrated as an audio editor. Definitely not underground, but if you've been pulling your hair out with *GIMP*, *Krita* might be the alternative you've been looking for.

Nick Peers
I recently stumbled across the perfect open-source password manager tool: *BitWarden*. It's cross-platform, boasts strong encryption and has a host of user-friendly features to make password and form filling easy. And its *Firefox* extension kicks *LastPass's* half-baked version into orbit.

Les Pounder
My favourite hidden gem is *photorec*. Coming home from an event, my SD card that was full of pictures corrupted and I lost everything! That was until I used this free file recovery tool. It might just save your data, one day!

Mayank Sharma
Mine's an alternate OS called KolibriOS. It's written in assembly, weighs in less than a 100MB and boots in the blink of an eye. It's got loads of tools including an image editor, a word processor, a web browser and dozens of games, and works wonderfully well inside VirtualBox.

Win a secure Nitrokey Storage 2

Send us your thoughts to the *Linux Format* dungeon server at **linuxformat@futurenet.com** and secure your chance to win a 32GB Nitrokey! The complete open hardware-encrypted storage solution!

Learn more at **www.nitrokey.com**.

An open source world

HotPicks is one of, if not the most popular section of *Linux Format* and while the reader survey tells me that, I don't actually understand why! My gut feeling is that people love the choice, variety and freedom *HotPicks* delivers every issue. I guess the truth is the sheer variety of open source means it can be hard to discover the best tools for the job and *HotPicks* offers a way to discover the best each issue... so say hello to our *HotPicks* Special!

It's a guide to this vast open source world and isn't that what this magazine is here for? So we're running a best open source software list for 2019. We've not done anything like this for over two years, so it's more than time we help people discover new software that's just waiting for an `apt install` to download.

The availability of open source is a curse and blessing. It makes some see it as free of value while the sheer abundance makes it hard for others to cut through the noise and get to the tools they need.

In the past, accessing the latest software involved compiling source code, but these days systems like Snaps and Flatpaks make it straightforward for anyone to grab the latest builds of their favourite programs. This is making software more accessible and better for everyone, which can only be a good thing.

Looking more widely, this issue offers perfect examples of easily overlooked open source, from the astronomy feature on page 52 and Jonni's Arch Linux *Roundup*. Every issue of *Linux Format* we cover some fascinating software I've never heard of, but that I'm always glad we have. Write in and let us know about your FOSS finds, we're always fascinated to hear what you're using out there. Enjoy!

Neil

Neil Mohr Editor
neil.mohr@futurenet.com

FREE GIFT

Contents

REVIEWS

THE ULTIMATE LINUX TOOLKIT!

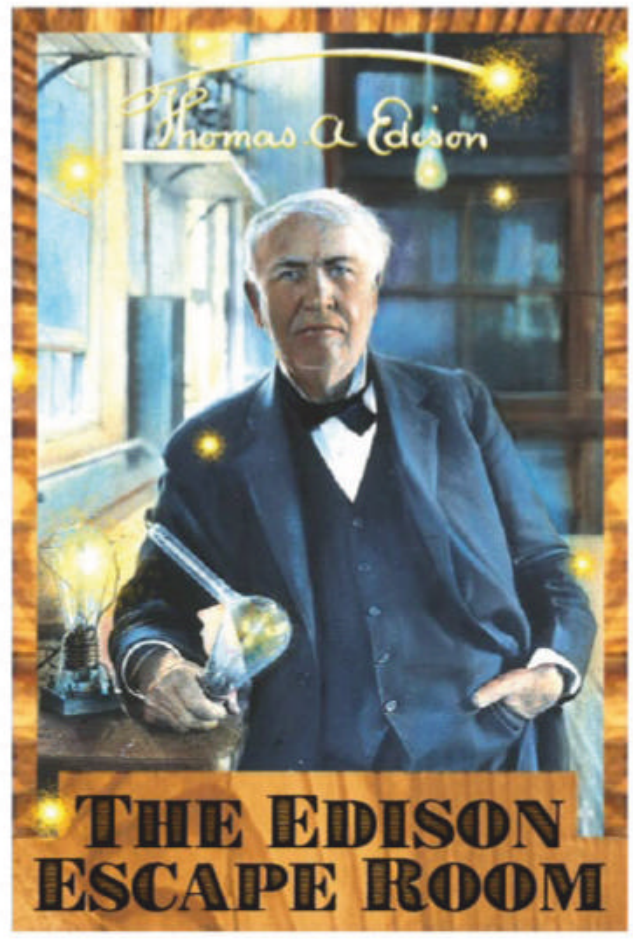
Whatever the computing task, there's a tool that will help you get it done. Find out more on **page 36**

ROUNDUP

INTERVIEW

CONTENTS

RASPBERRY Pi USER



CODING ACADEMY

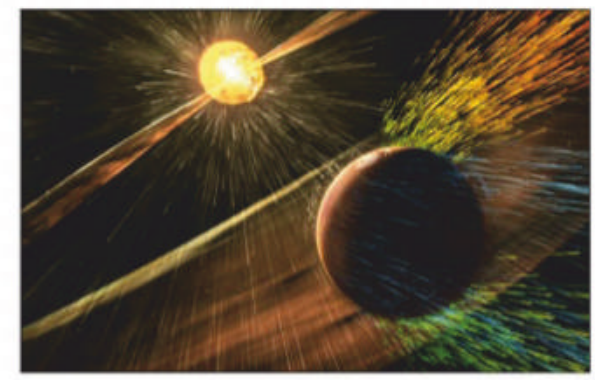
REGULARS AT A GLANCE

ON YOUR FREE DVD
- Bodhi Linux 5.0
- Voyager 18.10
- Sparky 4.9

Page 96

TUTORIALS

IN-DEPTH



Newsdesk

THIS ISSUE: Edge browser dropped » Nvidia gets PhysX-ical » Libre SoC project » Steam Link discontinued » Necuno Mobile

BROWSER TECH

Microsoft confirms that Edge is dead

After days of speculation, Microsoft has confirmed that Edge will drop EdgeHTML and become a Chromium-based browser.

Although Microsoft bundles its *Edge* web browser with Windows 10, the software continues to go unloved. A recent Netmarketshare report indicated that *Edge's* user base has been shrinking (**http://bit.ly/LXFNetMarketShare**).

Microsoft has now confirmed that *Edge* will be rebuilt on the Chromium rendering engine. While Microsoft isn't ditching *Edge*, it's embarrassing for the company to admit that its *Edge* browser, and the EdgeHTML browser engine have failed.

In a blog post (**http://bit.ly/LXFEdgeBlog**), Joe Belfiore, corporate vice president of Windows, stated that "we intend to adopt the Chromium open source project in the development of Microsoft *Edge* on the desktop to create better web compatibility for our customers and less fragmentation of the web for all web developers".

Edge has its own browser engine, EdgeHTML, which had been praised for its performance in some areas, but also brought frustration to web developers resulting from compatibility issues. Belfiore noted that the move means that "people using Microsoft *Edge* (and potentially other browsers) will experience improved compatibility with all web sites."

Microsoft also indicated that this new version of *Edge* will also be made available on various platforms and not just Windows 10, such as ARM-based Windows devices, and even macOS. It's been 15 years since a Microsoft browser was last available for Apple PCs. It also means we could see *Edge* come to Linux as well.

A Reddit post (**http://bit.ly/LXFEdgeReddit**) by Kyle Alden, project manager for *Edge*, also confirmed that, "It's our intention to support existing *Chrome* extensions." This will offer a huge boost to *Edge's* rather paltry collection of extensions, and it brings *Edge* and *Chrome* closer together when it comes to feature parity. However, there's a danger that *Edge* will morph into a clone of *Chrome*, reducing the number of alternative browsers for people who don't want to use Google's web browser.

Chrome, and Chromium-based browsers, are now so widely used in the web browser market that the loss of an alternative only strengthens *Chrome's* dominance.

Edge moving to Chromium could turn the web browser into just another Chrome clone.

> "BY ADOPTING CHROMIUM, MICROSOFT HANDS OVER CONTROL OF EVEN MORE OF ONLINE LIFE TO GOOGLE"

In a blog post on the Mozilla website (**http://bit.ly/LXFGoodByeEdge**), creator of rival web browser *Firefox*, Chris Beard states that, "by adopting Chromium, Microsoft hands over control of even more of online life to Google."

Yet while Microsoft appears to have given up trying to compete with Google in the web browser stakes, Mozilla isn't budging. "We compete with Google not because it's a good business opportunity. We compete with Google because the health of the internet and online life depend on competition and choice."

GRAPHICS

Nvidia makes PhysX open source

The company opens up its physics simulation engine.

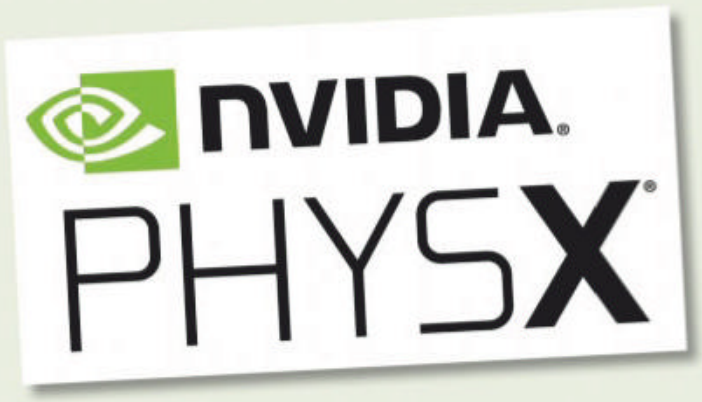
Nvidia has announced that its PhysX toolkit, which is used to create realistic computer physics in games and demos, will become open source. This means that under the BSD-3 licence, anyone can use the technology. In a blog post announcing the news (**http://bit.ly/LXFPhysX**), Nvidia stated that, "PhysX will now be the only free, open-source physics solution that takes advantage of GPU acceleration and can handle large virtual environments."

While PhysX is often associated with PC games, its physics simulation has applications in a wide variety of fields, including AI, robotics, computer vision, high-performance computing and self-driving vehicles. By making PhysX open source, these tools will be available to more people, and in our view this is a very welcome move on Nvidia's part.

As Rev Lebaredian, Nvidia's senior director of engineering for content & technology, explains, "We're doing this because physics simulation – long key to immersive games and entertainment – turns out to be more important than we ever thought." As he explains, "for self-driving cars, PhysX allows vehicles to drive for millions of miles in simulators that duplicate real-world conditions."

The source code is available right now on Github (**http://bit.ly/LXFPhysXGitHub**) and people are free to distribute, modify and sell the software, as long as the BSD license and copyright are included.

In the past, Nvidia hasn't been the most open source-friendly company, but it has made moves to address this, such as launching the NVDLA (NVIDIA Deep Learning Accelerator) open source hardware project in 2017.



In a welcome move, Nvidia has open-sourced its PhysX tools for everyone to play with.

OPEN HARDWARE

Libre RISC-V M-Class SoC launched

Can this SoC for mobile and embedded systems really deliver?

Over on CrowdSupply (**http://bit.ly/LXFCrowdSupply**) the Libre RISC-V M-Class project is looking for funding. Described as a "100 per cent libre RISC-V + 3D GPU chip for mobile devices", it promises to be a low-power 64-bit quad-core SoC running at 800MHz and suitable for tablets, netbooks and embedded systems.

Its full source code and files are available, including operating systems, processor and 3D GPU and VPU. This system-on-chip features a RISC-V CPU, along with the Kazan GPU (**http://bit.ly/LXFKazan**) which was previously known as Vulkan-CPU back when it was a Google Summer of Code 2017 project. It's a libre-licenced software-rendered Vulkan driver written in Rust, and uses optimised 3D instructions. According to the Libre RISC-V M-Class SoC project, it's targeting graphics performance of 25fps at 1,280x720 resolution, an 5-6 GFLOPs.

The project looks set to be rather complicated and pricey, too. Luke Leighton, one of the people behind the project, explains that, "I have no illusions about the cost of development of this project: it's going to be somewhere north of USD $6 million, with contingency of up to USD $10 million. This is just how it is."

Leighton's responses to several Phoronix articles at **http://bit.ly/LXFPhoronixResponse** reveal more about the work that's going into the project. The project will likely begin crowd-funding rounds soon, with a release date unlikely to be before 2020.

OPINION

PIPEWIRE EXPANDS

George Kiagiadakis
is a senior software engineer at Collabora

"Recently, a group of Linux multimedia and desktop experts, including myself, gathered in Edinburgh to take part in a hackfest with a common goal: improving the state of multimedia handling under Linux by exploring a new paradigm: PipeWire.

PipeWire is a new daemon that aims to improve the way applications share multimedia content and access devices under Linux. Originally designed to provide access to webcams, it now also supports audio. By enabling existing JACK applications to run unmodified alongside normal desktop programs, PipeWire is bringing Linux to parity with macOS and meeting one of its key design goals: supporting JACK-like low latency while having PulseAudio-level automation and ease of use.

PipeWire also introduces an enhanced security model for interacting with audio and video devices. It allows both sandboxed and normal applications to access devices easily and safely, utilising a sandbox portal and assigning fine-grained per-application permissions based on the functional role of the application.

PipeWire is the way forward for the desktop and beyond. If you're interested, then now is the time to get involved!"

OPINION

SPREADING THE WORD

Jon Masters is a kernel hacker who has been working on Linux for more than 22 years and works on energy-efficient ARM servers.

Years ago, I started an experiment that became the Linux Kernel Podcast (**kernelpodcast.org**). For over a year I produced a free daily show that had over a thousand regular listeners on all of the major platforms. Then, life intervened, and the daily show became weekly, and finally I couldn't keep up with it by myself any more.

The problem with the podcast wasn't so much the content (although it takes a vast effort to keep up with the Linux Kernel Mailing List or LKML), but in the creation of narrative, recording, editing and other production involved in maintaining even a modest show on a regular basis. With this in mind, I have recently decided to solve the problem of tracking and summarising community mailing lists through the creation of better (open) tooling and will be working on that in 2019.

I hope to bring back the podcast, but I also plan to benefit the production of this column. This will enable me to dive more deeply into topics that will be of interest to *Linux Format* readers. At the same time, I remain very interested in hearing from you about those noteworthy topics. Drop the editor a line and let us know what you'd like to learn more about in 2019. Happy New Year!

Kernel Watch

Jon Masters summarises the latest happenings in the Linux kernel community, for your reading pleasure.

Linus Torvalds announced Linux 4.20 Release Candidate (RC) 6, saying that he was happy with the general trend toward smaller RCs typically expected at the tail end of a kernel development cycle. His main concern was that with holidays upon us he didn't want to see the Linux 5.0 (formerly 4.21) merge window opening on Christmas Day. Beyond that, the performance troubles with some recent security patches (STIBP, covered in **LXF245**) had calmed down and a fix had been incorporated for a filesystem issue (*see the box, right*).

In a typical kernel development cycle, there are up to eight release candidates following an initial couple of weeks of (potentially) disruptive changes as new features are merged into the kernel, known as the "merge window". During the latter part of a cycle, stabilisation is the name of the game, as is preparation for the next merge, which is where we find ourselves at now. If everything goes according to plan, we'll be covering the 5.0 merge next time. There was nothing special about the use of 5.0 in place of 4.21 except that Linus had "run out of fingers and toes" to count to 21.

> THE IDEA OF AN X32 ABI SOUNDS REASONABLE, BUT THE PROBLEM COMES IN THE IMPLEMENTATION.

Removal of x32 support

A proposal has been floated (by Andy Lutomirski) to remove support for x32 from the upstream kernel. x32 is a 32-bit userspace ABI targeting the 64-bit x86-64 instruction set. While almost everyone who's running a 64-bit x86 machine today is using 64-bit native binaries that can take advantage of the huge amount of memory available with 64-bit pointers, there are some applications that can theoretically benefit from having smaller 32-bit pointers.

The idea of such ABIs, then, is to bring the best of both worlds: 64-bit instructions and all of the goodies from recent architecture evolution, while retaining 32-bit pointers to save on memory utilisation (and possibly benefit from a jump in performance).

The idea of an x32 ABI sounds reasonable (as does, perhaps the Arm one, known as "ILP32"), but the problem comes in the implementation. As a separate ABI, x32 isn't compatible with nearly all of the existing applications out there, meaning that distros have to build special x32 releases. Needless to say, the idea of two releases for a given architecture is not generally well received, and the adoption of x32 by distros has been near non-existent.

This was certainly cited by Andy in his post, "It's not entirely clear that it has users. As far as I know, it's supported on Gentoo and Debian, and the Debian Popcon [popularity contrast user supplied metrics] for x32 has been falling off dramatically. I don't think that any enterprise distro has ever supported x32". But this wasn't even the main reason for his ask.

More of a problem than the lack of users are the fairly fundamental problems with the current x32 implementation, which does some very strange things with its system call handling, and has build problems without very specific toolchains being installed. As a result, Andy proposed that "we made CONFIG_X86_X32 depend on BROKEN for a release or two and then remove all the code if no one complains. If anyone wants to re-add it, IMO they're welcome to do so, but they need to do it in a way that is maintainable".

Mika Weterberg (and others) subsequently posted patches changing IOMMU behavior on recent platforms designed to support Windows 10. An IOMMU (IO Memory Management Unit) is designed to provide IO isolation (as well as virtualisation support) such that an IO device cannot just access any system memory, but instead only that memory to which it is explicitly granted access.

This is especially important for certain classes of IO devices, such as those attached via the Thunderbolt port on modern laptops. Without any limits, Thunderbolt devices would be able to read and write whatever memory they want, snooping your data when attached.

Thunderbolt and lightning...

This well-known potential problem has previously been mitigated through an ability to change the mode of operation of the

Thunderbolt port, but it will now be treated by Linux similarly to Windows 10 with complete IOMMU isolation.

Dave Hansen posted a patch entitled "x86: remove Intel MPX", which follows through on an older proposal to remove support for Intel "Memory Protection Extensions" from Linux. MPX was originally designed to enable software to be built with more explicit information about its use of memory, such as the valid ranges of pointers to ensure they couldn't be used in a buffer overflow.

Adoption was spotty at best, while a recent Spectre-like paper has also demonstrated some problems with the existing implementation. The latter wasn't the reason for the technology to be removed, but it didn't help it stick around.

» A SEVERE FILESYSTEM BUG

In early December, Jens Axboe posted a patch entitled "blk-mq: fix corruption with direct issue". This addressed a bug in Linux 4.19 that had been seen with (for example) Ext4 filesystems running on certain SCSI devices. Those bitten by this bug would typically not know much about it until they started seeing error messages of the form "EXT4-fs error (device sda1): ext4_iget:4831: inode #7142: comm dpkg-query: bad extra_isize 24937 (inode size 256)". This message was the (Ext4) filesystem code complaining that it was reading "garbage" from the disk block layer underneath, which it was, because silent data corruption had already occurred by the time the message was displayed.

By then, it was too late to save the filesystem from corruption, and a fairly significant number of user bug reports started to trickle into the distributions who had consumed 4.19 into their updates. Unfortunately, the bug was rather stubborn and difficult to reproduce, especially for the kernel developers who might otherwise have seen and fixed it during development prior to anyone else being impacted.

Silent data corruption is what keeps kernel programmers up at night. Generally it's to be avoided at all costs. Indeed, most of the Enterprise distros are configured to automatically crash if they suspect any kind of situation might exist that could lead to silent data corruption.

The problem turned out to be an indirect consequence of a benign looking optimisation ("blk-mq: issue directly if hw queue isn't busy in case of 'none'") that had previously been made to the block multiqueue (blk-mq) layer.

The blk-mq layer (aka the "block layer") is responsible on contemporary Linux machines for handling the backend of IO operations, passing them from filesystem drivers to underlying drivers that talk to specific hardware. When IO operations are performed, they are typically tracked in software managed queues until the hardware is ready to process them, at which point the request is issued to a hardware managed queue. The "optimisation" that was previously merged attempted to short cut the software queue "when hw queue isn't busy".

Unfortunately, this didn't end well in the case that the hardware was unable to complete the operation. This could happen, for example, due to a system being too low on available memory at a critical moment. Normally, such operations are simply retried later, but in this case some of the state information about the underlying IO might have been lost.

Things become really bad if another part of the blk-mq layer decides to merge the failed request with another (another common optimisation) because the loss of state information now resulted in only part of the write taking place on retry.

At the end of the day, the solution turned out to be quite simple (do not merge requests when one of them tried the shortcut trick), but the fallout was more severe, with many questioning how this could have happened in 2018. How did this not get flagged by some kind of quality assurance process? A number of recommendations have been made to improve the process of vetting such changes ahead of time.

OPINION

STRIKE A BALANCE

Jonni Bidwell might seem familiar to readers of *Linux Format* magazine...

"I write to you from long in the past. No, I haven't created a time warp by messing with my system's locale variables, I speak of the time warp created by print media. That which enables me to write you from 2018, unaware of whatever political brownstorms marred the early part of the 2019.

Political angst aside, working around these delays is sometimes tricky. But that's not all which is tricky in the weird, anachronistic world of magazine publishing. Let me elucidate.

Every month people write in criticising the 'gobbledegook' that peppers our pages. By the same token people, will also write in accusing us of pandering to newbies with our more accessible features.

I like to think we strike a good balance, and as long as we keep getting letters from both sides (*like any good deal!–Ed*) that'll stay the case. But greybeards ought to remember they were once newbies too, and newbies ought to be aware that Linux is hard. We want more people to get involved with it and we want them not to be daunted. We salute anyone who wants to get into Linux, but remind them that part of that is being able to do some independent research, too."

OPINION

STRIKING A BALANCE

Keith Edmunds
is Tiger Computing Ltd's MD, which provides support for businesses using Linux

"Security is a compromise between keeping the bad guys out and providing easy access for the good guys. At least with open source software we can examine the code for security vulnerabilities, intended or otherwise.

The corollary is that open source also opens the door for some to insert malicious code. One such example came to light late November when the popular NPM Node.js library was found to be shipping with embedded malware. That's been fixed now, but the truth is it will happen again.

For all the plus points of open source software, this is a potential negative.

Far worse, though, is what's happened in Australia. The recently-passed Assistance and Access Act gives the Australian government the right to mandate "back door" access to software.

Couldn't happen here, could it? Wrong: it already has. The Investigatory Powers Act gives the UK government the exact same rights.

All of this makes open source more compelling for businesses. We might not be able to stop governments trampling over our privacy (although we should try), but at least we can see when they have."

FUNDING

$1 million for Free Software Foundation

Handshake, an organisation developing a peer-to-peer domain naming system, is behind the handout.

More support for the Free Software Foundation.

The Free Software Foundation (**www.fsf.org**) has received a number of charitable donations worth $1 million from Handshake (**https://handshake.org**), an organisation that has created a decentralised, permissionless peer-to-peer naming protocol compatible with DNS.

According to the FSF (see **http://bit.ly/LXFFSFFund**), these contributions will "support the FSF's organisational capacity, including its advocacy, education, and licensing initiatives, as well as projects fiscally sponsored by the FSF." This follows a similar gift of $1 million from the Pineapple Fund, and helps the FSF to continue to protect people's rights to use, study, copy, modify and redistribute computer programs, particularly GNU/Linux software. Handshake's donations will also be invested in Replicant (**https://replicant.us/**), an Android distro for mobile devices, the *GNU Project*, *GNU Guix* and *GuixSD*, *GNU Octave* and *GNU Toolchain*.

John W Eaton, original author and primary maintainer of *GNU Octave*, a high-level language for numerical computations, said, "We are grateful for such a generous donation. It is by far the single largest monetary contribution we have ever received, and we thank Handshake for including *Octave* in this select group

HARDWARE

RIP Steam Link

Valve kills off its streaming box, but you can build one...

Valve has discontinued its Stream Link device, which enabled you to stream games running on a PC to a TV in another room. Valve stated that it "intends to continue supporting the existing Steam Link hardware as well as distribution of the software versions of Steam Link, available for many leading smartphones, tablets and televisions."

The good news is that Valve has released the Steam Link app for the Raspberry Pi 3 and Raspberry Pi 3 B+ models running the Raspbian Stretch operating system, albeit only in beta at this point. Instructions on how to install and launch the app on your Pi in detailed in Valve's announcement: **http://bit.ly/LXFSteamLinkApp**.

The Steam Link lives on thanks to Raspberry Pi.

SMARTPHONES

Necuno Mobile

Necuno Solutions and WebOS Ports now working together

After you looking for an open source smartphone? Then it looks like you'll soon have even more choice, with Necuno Solutions and WebOS Ports collaborating on a "truly open source and secure mobile hardware platform" called Necuno Mobile.

Using LuneOS, which is an open source operating system based on Palm and HP's long-dead WebOS, this collaboration aims to offer Necuno Mobile as a platform for open source smartphone handsets.

In an announcement posted at **http://bit.ly/LXFNecuno**, Tanja Drca, Chief Communications Officer at Necuno Solutions, says that, "We believe users should be able to choose an operating system for their mobile device and LuneOS can be one of the choices. LuneOS is a good effort of bringing open-source alternative to the mobile operating system world."

While "a good effort" isn't the most enthusiastic endorsements we've read, we're looking forward to seeing handsets built on the Necuno Mobile platform. You can find out about the specifications of the Necuno's mobile device at **https://necunos.com/mobile**.

Distro watch

What's down the side of the free software sofa?

FREEBSD 12.0

The first entry in FreeBSD's 12 series is now available to download, and brings a range of updates including penSSL 1.1.1, OpenSSH 7.8 and Clang 6.0.1. KDE has been updated to 5.12.5, there's been some driver updates and various Lua loader improvements as well. According to the release announcement, TRIM support has been improved as well for the UFS filesystem. Find out more and download at **http://bit.ly/LXFFreeBSD12**.

The Unix-like OS FreeBSD has a new version available.

4MLINUX 27.0

A new version of this desktop Linux project has been released with a number of new features and improvements, mainly relating to media playback. There's support for AV1 video codec, a quick way to disable PulseAudio (which the release notes state is important to play classic video games like *Doom*), and better support for people using KVM. More classic games are also supported, and for a full list of changes visit **http://bit.ly/LXF4MLinux27**.

A new version of 4MLinux has been released with better support for video games from yesteryear.

DRAGONFLY BSD 5.4

Based on FreeBSD-4.x, DragonFly BSD has been updated with better support for asymmetric NUMA (Non-Uniform Memory Access) configurations, which is particularly important if you have one of AMD's Threadripper 2990WX processors. There's also been tweaks to improve performance and a number of program updates. GCC 8.0 is now the default complier. Find out more at **http://bit.ly/LXFDragonFlyBSD**.

DragonFly BSD 5.4 comes with a new compiler: GCC 8.0.

TAILS 3.11

This Debian-based live distro has just released a new version that concentrates mainly on tweaks and bug fixes. There's an update to the kernel to 4.18.20, while *Tor Browser* has been updated to 8.0.4. Because Tails is a distro that's aimed primarily at browsing the internet anonymously, this is a welcome update. *Thunderbird* has also been updated to 60.3.0.

For more information, read the release announcement at **http://bit.ly/LXFTails311**.

Tails is a great live distro that's ideally suited for anonymous browsing.

OPINION

UBER'S IN IT DEEP

Ibrahim Haddad
is The Linux Foundation's director of research

"Horovod, originated by Uber, is an open source distributed training framework for TensorFlow, Keras and PyTorch, which improves speed, scale and resource allocation in machine-learning training activities. It's also the newest project to come under the LF Deep Learning Foundation.

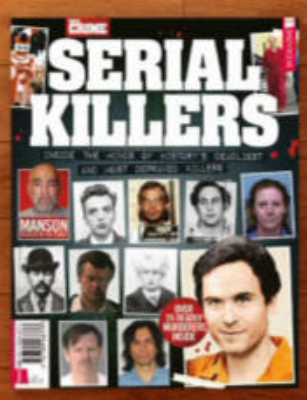
Horovod makes it easy to take single-GPU TensorFlow programs and successfully train them on many GPUs faster. It uses advanced algorithms and leverages features of high-performance networks to provide data scientists, researchers and AI developers with tooling to scale deep learning models with ease and high performance. In benchmarking Horovod against standard distributed TensorFlow, Horovod has come in roughly twice as fast.

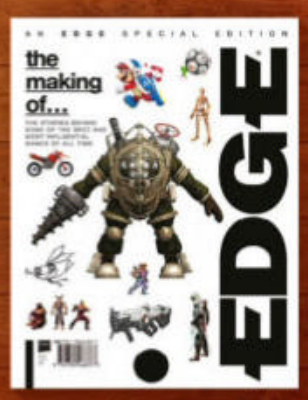
Uber has used Horovod to support self-driving vehicles, fraud detection and trip forecasting. It's also used by Alibaba, Amazon and Nvidia.

Horovod joins existing LF Deep Learning projects: Acumos AI, a platform and open source AI framework; Angel, a high-performance distributed machine learning platform based on Parameter Server; and EDL, an Elastic Deep Learning framework to help cloud service providers build cluster cloud services using deep learning frameworks."

Discover another of our great bookazines

From science and history to technology and crafts, there are dozens of Future bookazines to suit all tastes





Get great savings when you buy direct from us

1000s of great titles, many not available anywhere else

World-wide delivery and super-safe ordering

FUTURE

www.myfavouritemagazines.co.uk

Magazines, back issues & bookazines.

LINUX USER GROUPS

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

FIND AND JOIN A LUG

Build Brighton Thursday evenings is open night. **www.buildbrighton.com**

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

Huddersfield Raspberry Jam Meet every month at Huddersfield Library, typically on the fourth Saturday of each month. **https://huddersfieldraspberryjam.co.uk**

Horsham Raspberry Jam Park side, Chart Way, Horsham. **www.facebook.com/hackhorsham**

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

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

rLab Reading Hackspace Unit C1, Weldale ST, Reading, Open Sessions Wednesday from 7pm. **http://rlab.org.uk**

Glasgow Makers and Hardware Hackers Mitchell Library, Glasgow. **https://m.facebook.com/groups/115303729096198**

Teesside Hackspace Tuesday Evenings at Teesside Hackspace. **www.teessidehackspace.org.uk**

Goodbye 2018, hello 2019!

How the Linux community has grown in a year.

As the year draws to a close we can look back at what's been an exceptional year for our community. Linux has seen more impressive distributions from the big names and some smaller, community-developed projects. There have also been more open source software and hardware projects, increasing the diversity of platforms and projects made possible by the community.

The Python community have seen further updates to the v3 of the popular programming language. And there have been more boards supporting MicroPython a version of Python 3 for micro controllers, typically from Adafruit. This offers learners more scope when choosing the right platform for their project, meaning we have more choice – and that's what makes our community so diverse.

Community is an often-used word, but what we have in the Linux community is really a community of communities. These "sub-communities" are our diversity. Want to hack hardware? There's a Linux community. Want to create art? There's a Linux community. What brings these communities under one banner is Linux. We all have our chosen distro, we all support one another at our local LUG, makerspace or social meeting. Linux has enabled us to do so much using an operating system that's free and libre!

Celebrate our community in 2019. Share your passion with others and enable them to see that they can benefit from Linux. From your uncle sending an email on Ubuntu, to your daughter learning to code with a Raspberry Pi. There's a Linux distribution for everyone and the community to support them as they learn.

Leeds Hackspace has a vibrant community of makers, many of which use Linux to complete day-to-day tasks and control devices.

COMMUNITY EVENTS NEWS

PYCON 2019

PyCon is awesome! We took part in 2018 and spent five days in workshops, talks, lightning pitches and speaking to vendors. For 2019 the event returns to Cleveland, Ohio, US from 1 to 10 May. Around 4,000 visitors from across the world will be attending the leading Python event for every Pythonista. Tickets and more information can be found on the PyCon website: **https://us.pycon.org/2019**.

FOSDEM 2019

Once again the beautiful city of Brussels plays host to FOSDEM on 2 and 3 February 2019. This event is where open source projects big and small rub shoulders and share ideas, including a retro-computing workshop. At FOSDEM you can become accredited, take part in workshops and talks, and take part in a series of fringe events. This event is awesome and attracts over 3,000 people each year. If you're into open source, then you need to go to FOSDEM. More details at **https://fosdem.org/2019**.

MIXIT

A truly mixed conference. Starting in 2011, the MIXIT conference takes place in Lyon, France and it is open to everyone, no matter their programming language or discipline. Taking place between May 23 and 24, this event has a main conference for professional developers working alongside a conference track for children and educators. See **https://mixitconf.org/en** for more information.

Answers

Got a burning question about open source or the kernel? Whatever your level, email it to **lxf.answers@futurenet.com**

Neil Bothwick solves FOSS problems in his sleep.

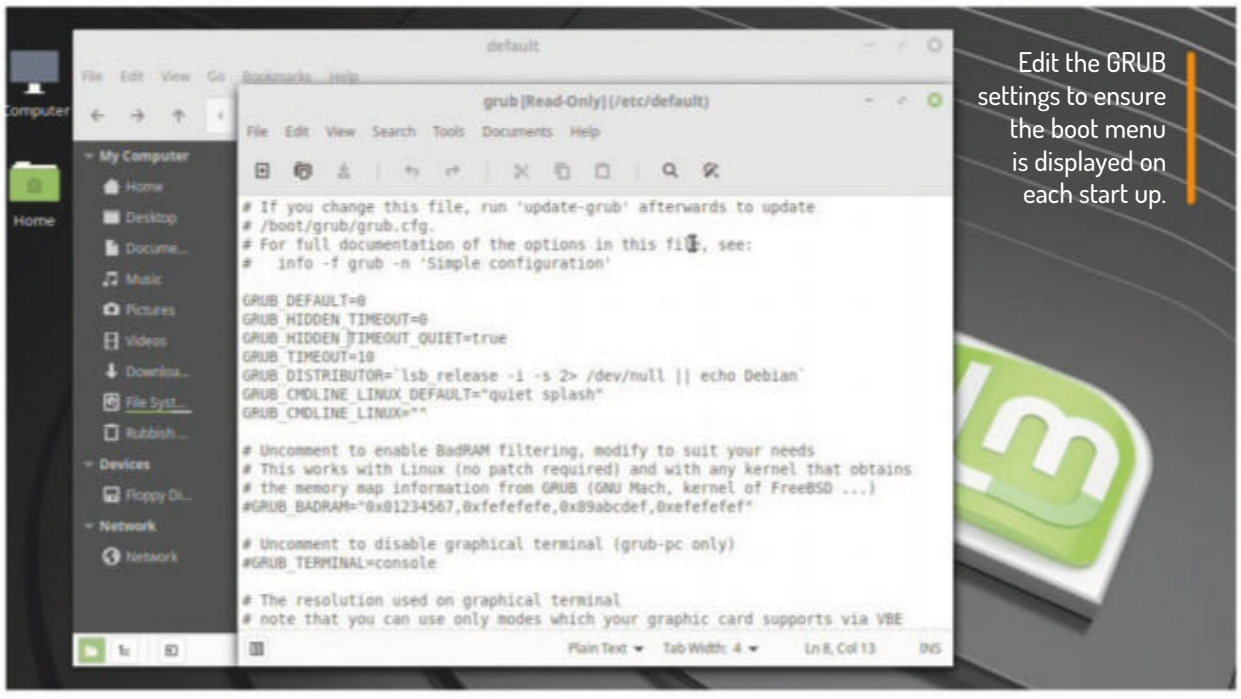
Q Where is Windows?

I've used Linux Mint for 12 months and find it okay. But then I bought your latest issue and it featured dual-booting with the latest Mint. I'm now on my fourth attempt but I can't get Mint to dual boot when I install it. It detects my Windows installation, and asks if I want to delete it. I click no then it just loads Mint and I never see my XP again.

Alan Crofts, via email

A First, identify if your Windows installation is still there but not showing up or whether it's actually gone. Load up the partition editor in Mint and look to see whether you have a Windows partition. Then you can mount this partition to see whether it contains Windows. You may be able to do this from the GUI or you can open a terminal and run the following:

```
$ sudo mkdir -p /mnt/windows
$ sudo mount /dev/sda1 /mnt/windows
```



Edit the GRUB settings to ensure the boot menu is displayed on each start up.

replacing `/dev/sda1` with the Windows partition you identified in the partition manager (it's often `sda1`). Now look at the contents of **/mnt/windows**. If it contains the usual Windows file structure, you're fine and you just need to be able to boot it. However, if you can't find your Windows installation then it's been erased by the options you chose when installing Mint.

If Windows truly has shuffled off this mortal coil, you'll need to reinstall from your restore disc and then install Mint alongside it. You should back up your Mint home directory before doing this to preserve all your files and settings.

If Windows is still there, either it is not in the GRUB boot menu or the menu is defaulting to hidden mode. GRUB is the bootloader, a small program that is loaded from your disk that then takes care of loading your operating system. If GRUB is hiding its menu, you only need to hold down the Esc or Shift key (try each in turn) when booting to make it appear, then you can select Windows or Mint to load. To save having to do this every time, you can change GRUB's default behaviour, by editing the file **/etc/default/grub**.

First remove, or comment out by placing a # at the start of the line, any existing references to GRUB_HIDDEN_TIMEOUT, GRUB_TIMEOUT_STYLE and GRUB_TIMEOUT, then add these two lines

```
GRUB_TIMEOUT=10
GRUB_TIMEOUT_STYLE=menu
```

Now regenerate the boot menu by running the command in a terminal:

```
$ sudo update-grub
```

You should now see a menu when you reboot that will appear for 10 seconds before proceeding with the first option if you select no other.

If the menu does not appear when pressing Shift or Esc, you can still generate a new one using the above steps, which should detect your Windows and Linux installations and create a suitable set of menu choices.

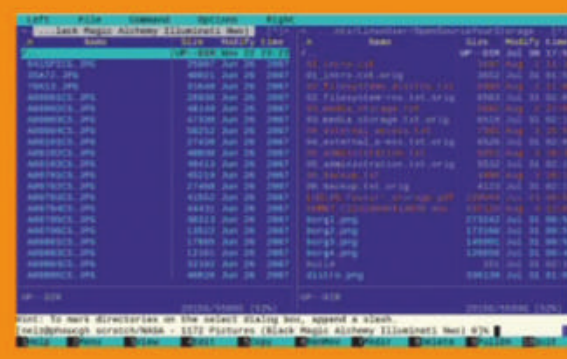
Waiting for network

Q **I recently upgraded my Debian 8 installation (LXDE) to Debian 9**

» A USEFUL TOOL

Gone are the days when every Linux user needed command line skills, although many of us still find it a more effective way of getting some things done. However, the need may still arise, either directly or via SSH. Once that happens you may find the need to work with files and miss your graphical file manager.

Enter *Midnight Commander*. This program, invoked by running `mc`, is an old style, twin-pane file manager. It's not a graphical program, instead it uses ncurses to produce a pseudo-graphical appearance in a text terminal. To control it, use a list of numbered commands at the bottom of the display, which correspond to your keyboard's function keys. *Midnight Commander* can do more than move files around. It can descend into archives, edit text files and even connect to remote hosts. Apart from the function keys, other important shortcuts are Tab to switch between the panels and Insert to select files.



Midnight Commander is a graphical (-ish) file manager in a text terminal.

and there's one remaining issue that's beating me. It seems that the wireless network reports that it's up about 10 seconds before it finishes the authentication process and gets an IP address (wireless is handled by *Wicd*). This caused problems in, for example, *Conky* where I look up my email account to check for new mails. I have a script that starts *Conky* as soon as the network is up. This caused *Conky* to try to start but fail and terminate as it couldn't log in to the email account. I also have a NAS on the wireless network and an entry in fstab to set it up via CIFS. The options that work in Debian 8 are:

```
noauto,x-systemd.automount,uid=1000,gid=1000,rw,guest,file_mode=0777,dir_mode=0777,iocharset=utf8 0 0
```

However, it doesn't work at all in Debian 9. After the network is fully up, I can mount the NAS manually with `sudo mount -t ...` and if I leave out the `noauto` option in fstab I can mount it using `sudo mount -a`. Once mounted everything works fine, but I would rather have it mount automatically as it should.

The problem seems to be with the `x-systemd.automount` option. It should make it possible to mount the NAS as soon as I access the mount directory but when I try, there's nothing there. I came across something that said I should use systemctl to enable `network-fs.target` but it didn't seem to make any difference. Most posts on this type of issue seem to use a trial-and-error approach to problem solving, such as trying various options in fstab: `_netdev`, `nofail` and so on. I have tried many permutations but none that works.

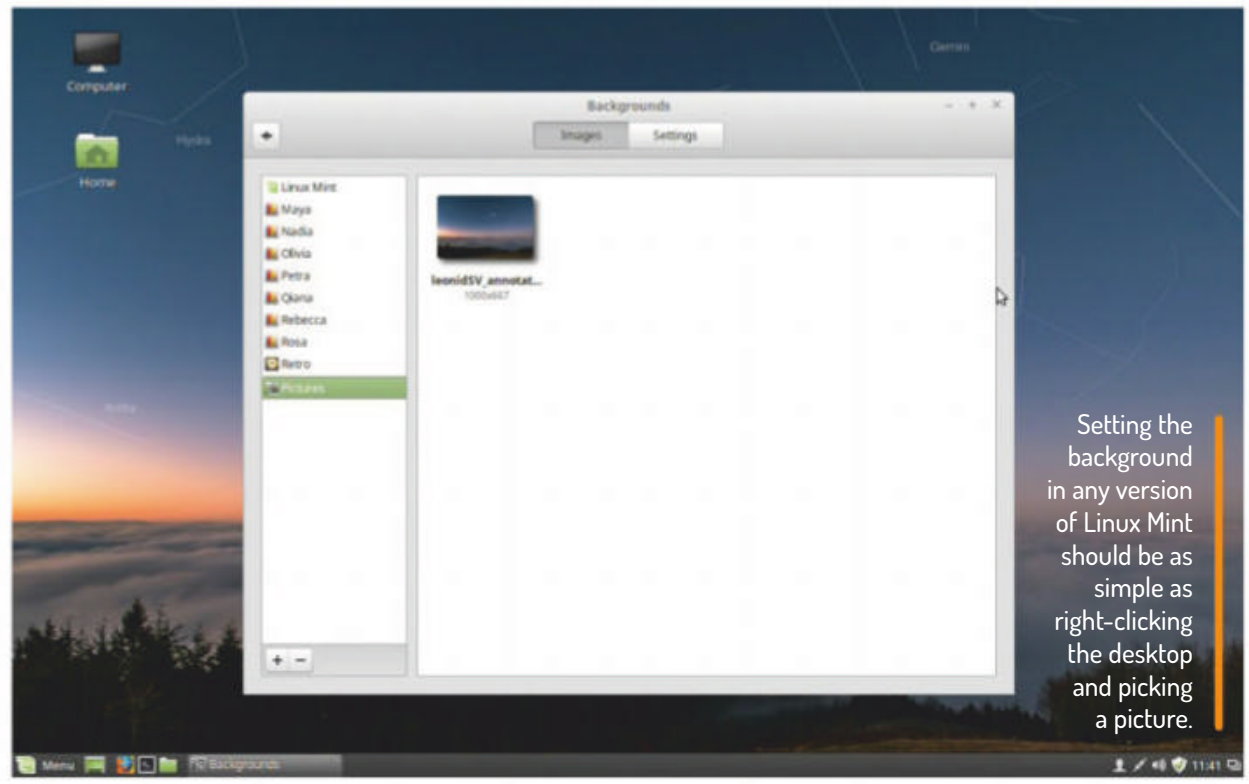
I'm not sure how to check what `x-systemd.automount` is doing, what it should do behind the scenes or how to actually get it to work.

Hugh McCullough, via email

A This problem is probably caused by the way in which `systemd` determines whether the network is up, and what a vague concept that is. Is the network up when one interface is active, when you can connect to other hosts on the LAN or when you connect to the Internet? The default behaviour is the first option, as soon as network hardware becomes available, which is not what you want. For those using `systemd-networkd` or *NetworkManager* to manage their networks, there are specific targets you can enable to have network-dependent services, such as NFS mounting, wait for the network to become active – `systemd-networkd-wait-online.service` and `NetworkManager-wait-online.service` respectively. The `x-systemd.automount` mount option simply tells `systemd` to automatically mount the share, effectively overriding your `noauto` setting. The is not the root of your problem.

If you want to continue using *Wicd*, it's been unmaintained since about 2012, you can remove the automount options and mount your network shares from a script called from *Wicd* after it establishes a connection. I haven't used *Wicd* for several years, but that's how I used to do things. It has the advantage that you can have it mount the shares only when connected to your home network.

To learn more about the **fstab** options used by `systemd`, read the systemd.mount man page. For more details on how `systemd` handles the network's status, see **https://www.freedesktop.org/wiki/Software/systemd/NetworkTarget/**.



Ugly Rosa

Q **I was running Mint 17.1 on my machine and in preparation to upgrade to Mint 19 I upgraded to Rosa (17.3). Now I have an ugly background on my desktop that I can't get rid of. I am running Cinnamon. On my other Mint versions (17.2 or 18.3 etc) there's no problem. How do I fix this? Is this an artefact of the fact that I upgraded rather than did a clean install?**

Greg Morse, via email

A We can't reproduce that problem here, but it's unlikely to be down to your upgrading rather than reinstalling. Debian-based distros are intended to be upgraded in this way. The exact cause is not easy to determine because you have given no description of the steps you took and the failure that occurred. For example, is the option to change the desktop background no longer there? Is it there but no other backgrounds are available, or does it give an error when trying to apply a new background? Or are you using a different method to set the background? You can right-click an image in the file manager and use the Set as wallpaper option. Similarly, you can right-click an image in *Firefox* and set it as the desktop background.

If all of these methods fail, and no useful error messages are given, consult the system and X server logs. To read the system log, open a terminal and run

```
$ journalctl -f
```

Then try to set the background and look for any error message output to the terminal. Similarly, look for errors from the X display server by running this command before trying to change the background:

```
$ tail -f ~/.xsession-errors
```

Bear in mind that other software could be logging errors, but if you try more than once and the same error appears each time, you have a good pointer to the cause of your problem. One possible cause may be that something changed in your user setup when upgrading. You don't specify exactly how you performed the upgrade. If you're using the same username but have somehow ended up with a different numerical user ID, it can happen. You'll be unable to change settings, although this situation is likely to produce other errors too. A brute force approach to fixing this across your home directory is to run this terminal command:

```
$ sudo chown -R username: ~username
```

Replace both instances of `username` with your actual user name. This will set every file in your home directory to be owned by you.

Q Ubuntu 18.10 suspend

I'm trying out Ubuntu 18.10, having used 18.04 for some time. It does start up a little faster than 18.04. However, I have been using the suspend function recently because it suits my user pattern well. But there's a problem with 18.10. It has a routine for setting an automatic suspend after a predetermined time. I set mine at 15 minutes. Much easier than using the command line `pm-suspend` I thought!

Well, it does suspend after 15 minutes, but it doesn't restart on pressing the Space bar as previously. First, the Ubuntu 18.10 screen appears, with the time and date on it. Then a few chevron arrows appear moving upwards, then the "no signal" message appears, then a blank screen. On repressing the space bar the login screen appears, requiring me to login again.

I then switched off the auto suspend and tried the command line `pm-suspend` – `no such command` was the response! There's no point in a suspend function that requires me to login again, I may as well switch off and save electricity! Is this a bug to be fixed or should I go back to 18.04?

Bryan Mitchell, via email

A To cover your last point first, Ubuntu 18.10 uses `systemd`, so the correct command to suspend from the command line is the following:

```
$ systemctl suspend
```

If you really want `pm-suspend`, install the *pm-utils* package, but it's not needed with `systemd`.

The screen you see shouldn't be a login screen but the unlock screen. They look almost the same, apart from the text in the action button. When the computer is suspended, the desktop is also locked, which is why it asks for your password, but when it opens you should see the same desktop with the same applications open. You can turn off this behaviour if you wish, but it's a security feature.

Suspend is commonly used on laptops and this feature means that if your laptop is stolen while suspended, its contents won't be so easily accessible to the thief. A similar argument applies to desktops if others have physical access, such as in an office. If you're using a desktop where no one untrustworthy has access, you can turn off the lock function in the privacy section of the main settings window.

The sequence of seeing the screen with the time and chevrons on the first press of the Space bar and the unlock screen on the second press is perfectly normal, the no signal message is usual. This message is from the display hardware and indicates that the computer is asking it to display an unsupported video resolution. It's apparently harmless as you are still able to get to the unlock screen, but it may be worth checking your display preferences, too. LXF

» JAVASCRIPT PDFS

Q

Having just completed my tax form again, which required the use of *Adobe Acrobat* reader using JavaScript, and needing me to dig out my Windows machine to do so. Are there really no open source PDF readers out there, or in the pipe line, that will run with JavaScript?

For a small, one-man business like myself, the situation is manageable, but for larger organisations who need to use these active JavaScript PDF documents, the situation must be, or would be intolerable. This would surely not help the cause of a larger uptake of Linux desktops in the business world.

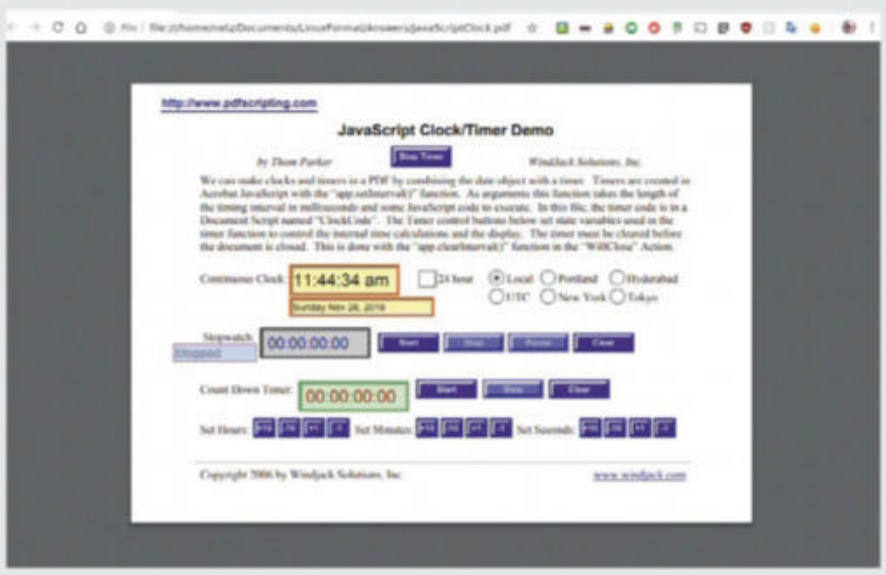
While I appreciate the security issues surrounding the use of JavaScript, these types of documents are out there, so one must simply live with them.

Keith Wilton, via email

A I'm surprised that a government department requesting personal and financial data would use anything as insecure as JavaScript embedded in a PDF, but this is possibly one of those situations where your choices are restricted. You're right in that most of the standalone PDF readers don't support JavaScript, but that does leave you with two alternatives. The first is to install Adobe's own reader. It stopped supporting Linux a few years ago, but the latest version Adobe released, 9.5.5, is still available from its web site. This is a 32-bit application, so you'll need a multilib system to run it. On Ubuntu and its derivatives, that means you need to install *libxml2:i386* and *gdebi-core*, then download the **.deb** file from **ftp://ftp.adobe.com/pub/adobe/reader/unix** and install it by double-clicking the downloaded file or running the following terminal command:

```
$ sudo gdebi AdbeRdr9.5.5-1_i386linux_enu.deb
```

The alternative is to use the *Chrome* browser, or *Chromium* with the *Chrome* extensions, as this has up-to-date PDF support. We don't have access to the exact file you need to work with, but Chrome worked with a number of JavaScript enabled PDFs we found on the interwebs.



JavaScript in PDFs is generally considered bad practice, but if you have to work with them in Linux, Chrome is one option.

GET HELP NOW!

We'd love to try and answer any questions you send to **lxf.answers@futurenet.com**, no matter what the level. We've all been stuck before, so don't be shy. However, we're only human (although many suspect Neil is a robot), so it's important that you include as much information as you can. If something works on one distro but not another, then tell us. If you get an error message, please tell us the exact message and precisely what you did to invoke it.

If you have, or suspect, a hardware problem, let us know about the hardware. Consider installing *hardinfo* or *lshw*. These programs list the hardware on your machine, so send us their output. If you're unwilling, or unable, to install these, run the following commands in a root terminal and send us the **system.txt** file too.

```
uname -a > system.txt
lspci >> system.txt
lspci -vv >> system.txt
```

ON SALE NOW!

Available at **WHSmith, myfavouritemagazines.co.uk**
or simply search for 'T3' in your device's App Store

SUBSCRIBE TODAY AND SAVE!
www.myfavouritemagazines.co.uk/T3

GET IT ON Google play zinio Available on the App Store

Mailserver

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

Octomonster

Because of your 9/10 review I went and bought an OCTOCAM kit. I will never trust you again (!) The hardware assembles easily enough, but the software has been a week-long horror show with no end in sight. Because the Pi Zero W has only two USB ports and one of those must be used for Power, it can handle a keyboard but no mouse.

The *MotionEYE* software, which should be used with this kit, crashes regularly after one or two minutes. Its install process is almost impossible without major searches and a bit of extra hardware, such as a LAN-to-USB ethernet dongle. You need to do a much better job in future of actually getting the kit up and running before you give it a 9/10.

Greg Morse, via email

CREDIT: Pimoroni

Les replies

Well, yes that's a caveat of the Pi Zero W range, but it doesn't affect the installation and configuration of this kit. Installation of *MotionEYE OS* is via downloading the pre-made image **https://github.com/ccrisan/motioneyeos/releases** and then following Pimoroni's excellent guide **https://learn.pimoroni.com/tutorial/sandyj/motioneye-os-on-your-octocam**, which is just what I did!

What about...?

Reading through LXF240 it mentions on page 84 that *Qt314wall* can upscale using the Waifu filter. Could this be used change VHS videos into high definition? Could you also take a look at doing a big feature on setting up the *M.A.M.E.* emulator and where to get legal ROMS for it. And please do another article on *LMMS* and review the latest *GIMP*. Thanks!

Ian Learmonth, via email

LETTER_OF_THE_MONTH

The converted

Decades ago, I was a complete beginner when it came to Windows, I remember calling the Microsoft helpline and not realising initially where the Start button (*we're still looking for the Any key–Ed*) was! And so with Linux I've been intrigued yet still very much a novice. Thankfully, *Linux Format* has showed me the way.

I was working in a school's science department and the English department had put 100 Eee PC notepads into storage, no doubt because their operating system didn't have *Word* and also they were cumbersome with Wi-Fi. You had to configure it each time!

We acquired these notebooks and with the help of *Linux Format* and its cover disk, I was able to find a suitable light version of Linux – Lubuntu – with no Wi-Fi problems. I spent a few days loading up all the machines and setting them up so all the science classes could get convenient internet access for their lessons. Happy teachers and kudos for me, the lab tech!

So thanks *Linux Format* – that was a practical result in the real world!

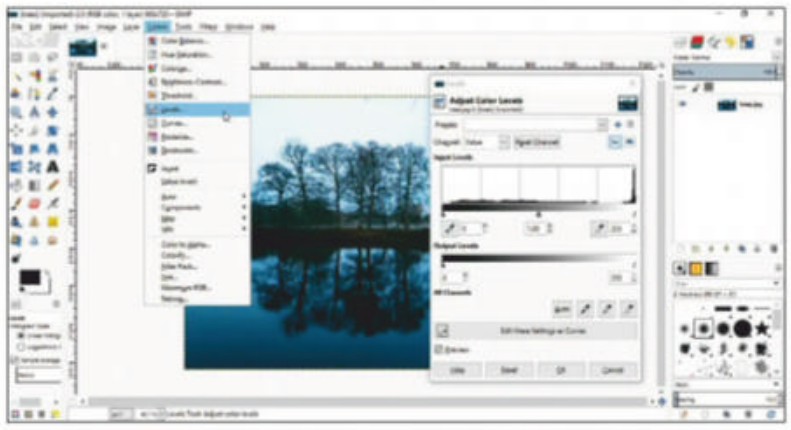
Dominic Coltman, via email

Neil

Wow, that's amazing work. Have a secure USB key for your trouble.

Helpdex

shane_collinge@yahoo.com



Brace yourself – GIMP reviews are coming!

Neil replies

So the short answer is: I guess, perhaps. *qt314wall* (*Hotpicks*, **LXF240**) is designed for still images, so you could video capture an old SD video to individual frames, run these through *qt314* and then compose a video. But we're sure you could use an *AVISynth* – designed for video – script to do the same more easily.

We're planning a retro special in the future, so expect *M.A.M.E.* to be in there, and legal ROMS is always an interesting section of that.

Perhaps Jonni fancies taking on LMMS? I always find those technical areas odd, because anyone that has to use them will already know anything that we can fit in the space of an article.

Review *GIMP*? Funny you mention that – it'll be coming up in a *Roundup* soon. Do you fancy any tutorials using it to achieve anything specific?

Noddybook

I recently acquired a netbook computer (a Medion S2218 from Aldi for £99) that was preloaded with Windows 10. While the hardware specification was modest, I thought it should be perfectly capable of supporting a lightweight Linux distro.

I have tried on many occasions to use a Live USB to boot up various distros (including 32- and 64-bit Linux Mint 19) The USB drive boots perfectly on a standard BIOS desktop system, but the netbook (with UEFI) appears singularly reluctant to oblige. Clearly I'm not doing something right, but I'm wary of changing anything which might ultimately brick the device.

I don't so much dislike Windows 10 as an idea, but things which really hack me off include: a requirement to register with Microsoft; forced "all or nothing" updates that have now filled the 32GB SSD; and continual nagware inviting me to pay for subscription-based add-ons.

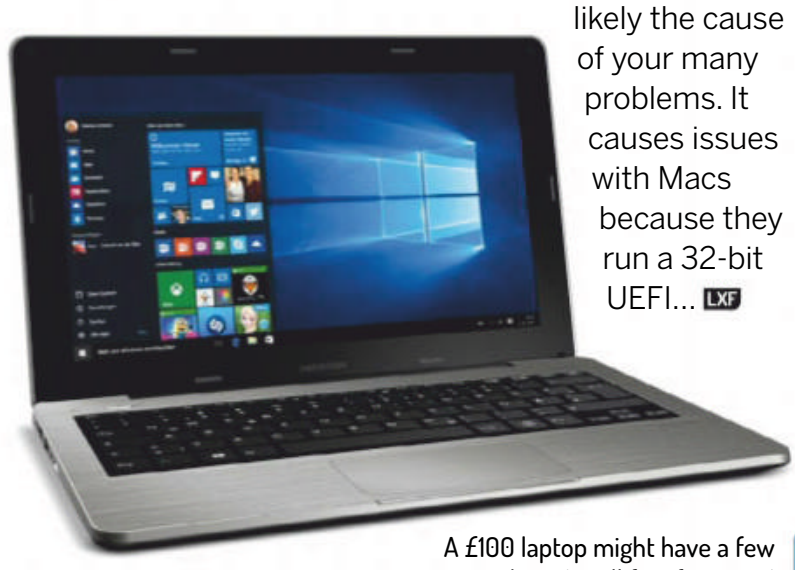
So far, I think Windows 10 is the strongest argument to date for adopting Linux!

I appreciate there may be limited interest in answering such a question, but I surely can't be the first person to bring this up. Is there any prospect | on doing an in-depth tutorial showing readers how to break open UEFI?

Jan Dobie

Neil replies

I'll forward your letter onto Neil Bothwick because he might be able to give you a full answer... I can say the issue here is something we've come across, in terms of this is a 64-bit processor but looks like it's using a 32-bit UEFI (something that never was suppose to be). The 32-bit UEFI can't see any 64-bit OSes, which is most likely the cause of your many problems. It causes issues with Macs because they run a 32-bit UEFI... LXF



CREDIT: Medion

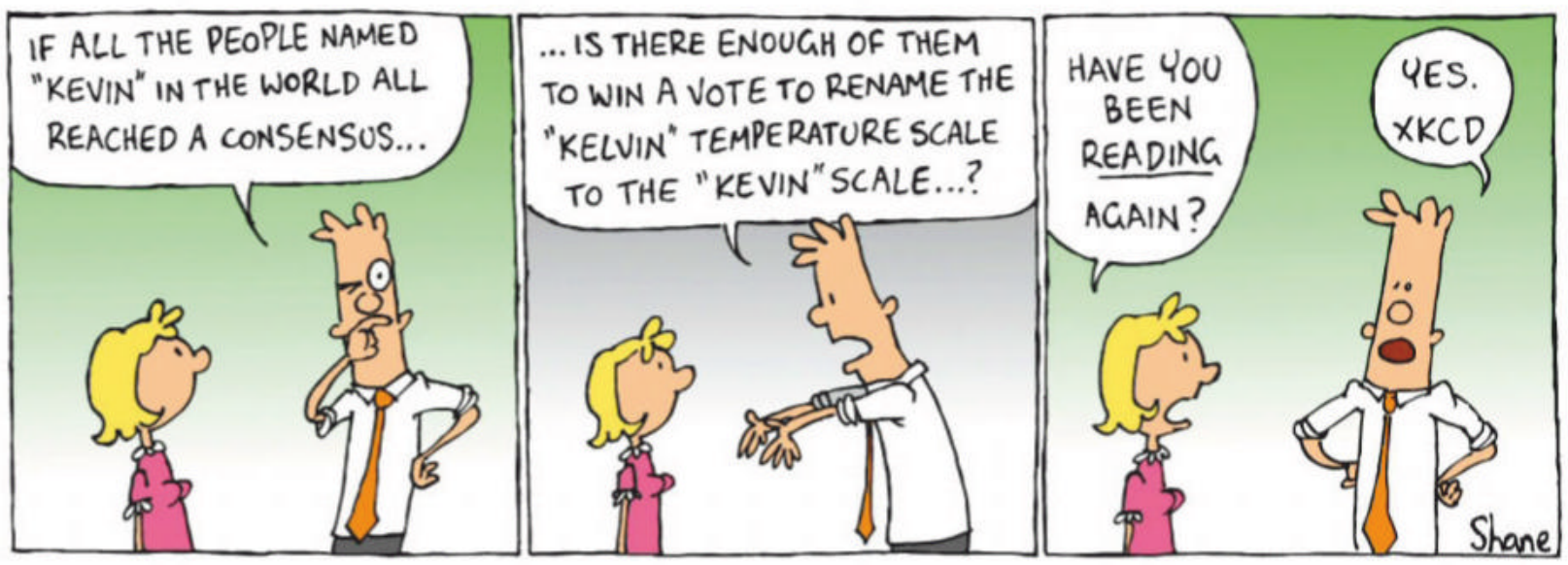
A £100 laptop might have a few corners cut, but it's still fine for Linux!

WIN A SECURE NITROKEY STORAGE 2

Send your thoughts to the *Linux Format* dungeon server at **linuxformat@futurenet.com** be picked as Letter Of The Month and win* an amazing 32GB Nitrokey Storage 2!

The Nitrokey is worth a cool €149, has 32GB of hardware-encrypted storage, supports hidden volumes, secures your online accounts with one-time passwords, can encrypt emails, files and hard drives and is an open source, open hardware solution. Protected with AES-256 and RSA keys up to 4,096! Learn more at **www.nitrokey.com**.

The Nitrokey is your one-stop security solution.



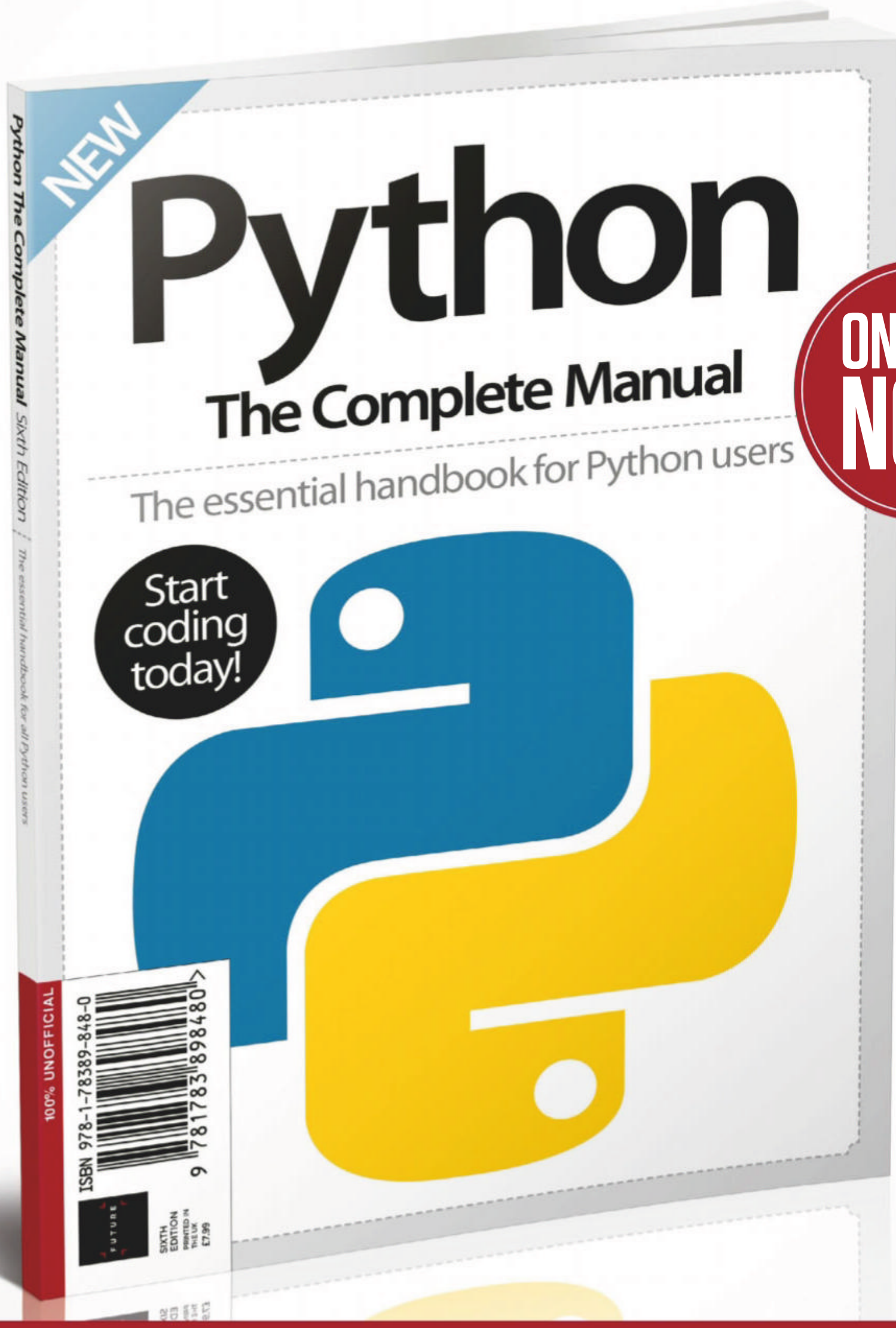
WRITE TO US

Do you have a burning Linux-related issue that you want to discuss? Write to us at *Linux Format*, Future Publishing, Quay House, The Ambury, Bath, BA1 1UA or email **lxf.letters@futurenet.com**.

* For full terms and conditions see: **www.futureplc.com/terms-conditions**

THE ESSENTIAL HANDBOOK FOR PYTHON USERS

Learn how to program in Python, discover amazing projects to improve your understanding, and find ways to enhance your experience of computing



ON SALE NOW

FUTURE

Ordering is easy. Go online at:

www.myfavouritemagazines.co.uk

Or get it from selected supermarkets & newsagents

AMD Athlon 200GE

Everyone said **Alan Dexter** was a cheapskate and here's the CPU he's been waiting for.

SPECS

- **Socket:** AM4
- **Type:** 64-bit
- **Clock:** 3.2GHz
- **Cores:** 2 (4 threads)
- **Process:** 14nm
- **Cache:** L3 4MB, L2 1MB, L1 192KB
- **Mem:** 64GB DDR4 Max, dual channel
- **GPU:** 3 core Radeon Vega 3, core 1GHz
- **PCIe:** 8 lanes
- **TDP:** 35W
- **Ext.:** SSE4.2, AES, AVX2, SHA, AES, AMD-V

Conventional wisdom says that AMD is the budget darling, while Intel has you covered at the high end. Conventional wisdom is out of step with reality. Because, right now, if you want serious performance, AMD's Threadripper should be at the top of your list, while anyone on a super-tight budget is better served by Intel's Pentium chips and the likes of the Core i3-8100. AMD doesn't want to leave any market segment to Intel, though, so it has revived the Athlon brand for this release, its most affordable Zen chip to date.

Before we delve into the details, it's worth mentioning the price. In theory, this is a £50 CPU, but in truth, you'll do well to actually find it for that, because it's one of those chips that tends to sell out. Note that this paltry sum also includes a CPU cooler, and in our testing, it is indeed every bit as "whisper quiet" as AMD claims, for once. So either way you get a lot of hardware for your money.

Apart from a rather unimpressive-looking, yet still capable, CPU cooler, what does your small chunk of cash get you? This is a dual-core, quad-thread processor, with a 3.2GHz base clock and Radeon Vega 3 integrated graphics. It uses the same AM4 socket as the bigger Ryzen chips, has a low (for the desktop) TDP of just 35W, and supports dual-channel DDR4 memory.

Obviously, the integrated graphics aren't going to have you playing the latest games at maximum settings, but with sufficient tweaking, you should get some classics and efficient modern titles running at passable frame rates.

A quick look at the benchmark table below reveals that it performs pretty much where we'd expect it in most CPU tests. Where it does falter is when it's paired with a decent graphics card, in the form of the Nvidia GeForce GTX 1080, and this is because the inclusion of the integrated graphics butchers the PCIe connectivity down to just eight channels, which is simply not enough to keep any discrete card fed with enough data. If you're looking to get into gaming on the cheap utilising a separate graphics card, this is essentially a false economy, and you'd be better off going for a Ryzen 3 or Core i3 option.

On the plus side, this is a low-power chip, and even under load, it doesn't draw much from the wall. It also offers great value for money, and boasts better bang for your buck than most chips. If you're looking for a small system that is going to make the most of the integrated graphics, for a media centre or a discreet server, then there's a lot to like here.

Gamers, serious users, and anyone looking for an affordable base system for bigger and better things are going to be ultimately disappointed, unless you factor in a CPU upgrade down the line as part of it – it does use the standard AM4 socket, after all. LXF

» ON THE TEST BENCH

Benchmark	AMD Athlon 200GE	Intel Core i3-8100
Cinebench R15 Single thread	121	153
Cinebench R15 Multi-thread	348	598
x264 (fps)	7.81	14.76
Memory Latency (ns)	84.1	63
Memory Bandwidth (MB/s)	37,859	42,076
Power Draw Idle/Load (W)	44/67	43/112

VERDICT

DEVELOPER: AMD
WEB: www.amd.com/athlon
PRICE: £50

FEATURES	**8/10**	**EASE OF USE**	**9/10**
PERFORMANCE	**6/10**	**VALUE**	**10/10**

Great value for money and low(ish) power draw, but don't expect much beyond 720p gaming or media centre work..

» Rating 8/10

Sailfish X

Now that more devices work with Jolla's Sailfish OS, **Alexander Tolstoy** compares Sailfish X with Sailfish 3 and sees if he can tell them apart...

IN BRIEF

The long-awaited version of Sailfish OS for another set of Sony Xperia smartphones. If you enjoyed using Nokia N9 and Meego back in the day, you're definitely the target audience of this offering. Hopefully this version of Sailfish X will soon get the official support layer that can riun Android apps alongside native ones.

Shortly after Jolla rolled out Sailfish 3, the third major release of this alternative mobile OS (see **LXF236** news), there was more good news. The Finns rolled out the updated version of Sailfish X for Sony Xperia XA2 devices. Let's find out what's new.

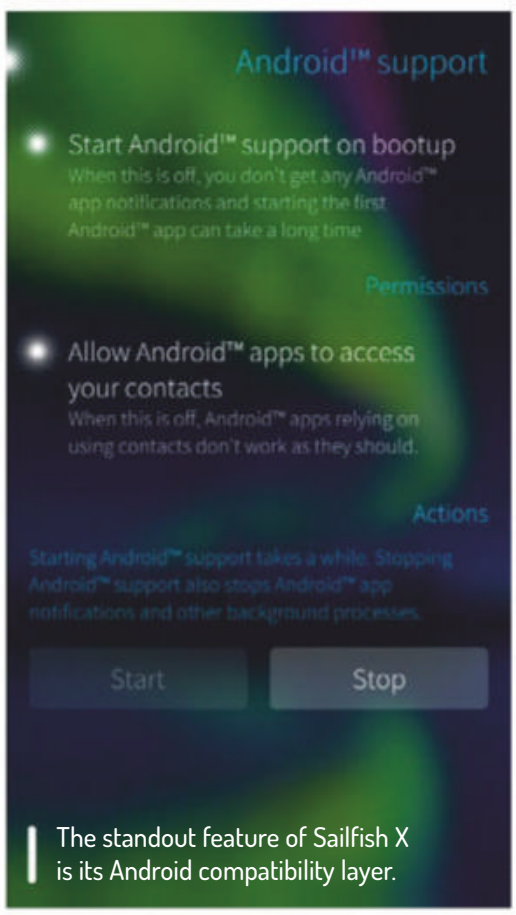
First, some clarification is required to tell X and 3 apart. Sailfish 3 is the current version of the OS for Jolla's own two phones (Jolla and Jolla C), together with some other phones with limited availability as well as the basement of community builds and adaptations for a few more devices. Sailfish X is just a re-branded version of Sailfish OS for Sony Xperia X mobile devices. The initiative was made possible thanks to Sony Mobile's attitude towards the open source community. The vendor launched the Open Devices programme, which featured an open bootloader coupled with an officially supported ROM flash tool.

You can run Sailfish X on a Sony Xperia handset, or Jolla's own mobile hardware. CREDIT: Jolla

Sailfish X is a commercial offering with support from Jolla. It can be purchased as a standalone product and installed on supported Sony Xperia devices, with Jolla providing instructions about flashing your device from Linux, macOS or Windows. We tried the latest Sailfish X version on the newer Sony Xperia XA2 and older Xperia X.

Paranoid Android

There's always been a question mark over the capabilities of Sailfish running on different devices. Does it run Android apps? This is a killer feature that attracts a lot of praise and puts Sailfish firmly in customers' sights. A custom Linux-based OS that includes an Android compatibility layer and enables users run their favourite Android apps is an undisputed hit, but take away such functionality and live with native apps only, and the hit OS becomes a miss. At least, this is true for an average tech-savvy user who can download APK files manually, but will avoid running a mobile OS that lacks banking apps, games, taxi ordering and so on as their main system.



The standout feature of Sailfish X is its Android compatibility layer.

What makes Sailfish X special is the inclusion of that third-party Android layer. Technically, it's Alien Dalvik from Myriad Group AG, a German vendor. Alien Dalvik is a proprietary tech and thus it can only be shipped within a commercial Jolla offering. Jolla's own two handsets and older Sailfish X for Sony F512X had Alien Dalvik, whereas custom community-made Sailfish ROMs and some regional offerings (Jala, Intex, Inoi and so on) lacked this aspect.

Taking Sailfish X for a spin on a modern Sony smartphone raises the same question. Oddly, the latest Sailfish X offering, rolled out in November 2018 for Sony Xperia XA2 (H4XX3) devices, only mentions that Android app support is on its way – it doesn't have it out of the box. The Jolla team calls it the 'free trial' of Sailfish X for those devices and has promised to deliver Android support by the end of 2018. Behind the scenes there's a strong commitment of Jolla and Myriad Group to update Alien Dalvik from version 5.1 (Lollipop) to 8.0 (Oreo) and offer the licensed version to Jolla customers. The full-featured Sailfish X normally costs 49 euros and is available in EU countries, and Norway and Switzerland.

So for now, the older Xperia X models runs fully fledged Sailfish X and the latest Xperia XA2s run so-called 'trial' versions of Sailfish X, for the time being at least.

The good and (not so) bad

Sailfish OS is mainly made up of open source bricks. The system runs kernel 3.10 with plenty of backported goodness and security fixes. It boots using Systemd and uses the Wayland graphic stack. The UI is implemented using Qt5 technology and written in QML and C++. The fluid semi-transparent interface of Sailfish, which is easy to navigate using swipes, has been built using both the OSS QtQuick module from Qt, and the proprietary Silica module from Jolla.

A swipe-controlled interface is of huge benefit to the OS, but there are other features that made us even more excited. The first one is desktop-class multitasking. When compared with Android, where running apps are suspended and killed shortly after you switch the focus away, Sailfish behaves like a desktop system. Background apps remain active and can be restored quickly, and it doesn't affect battery life significantly.

The second point is package management. Sailfish doesn't need rooting or 'jailbreaking'. Instead, it grants full

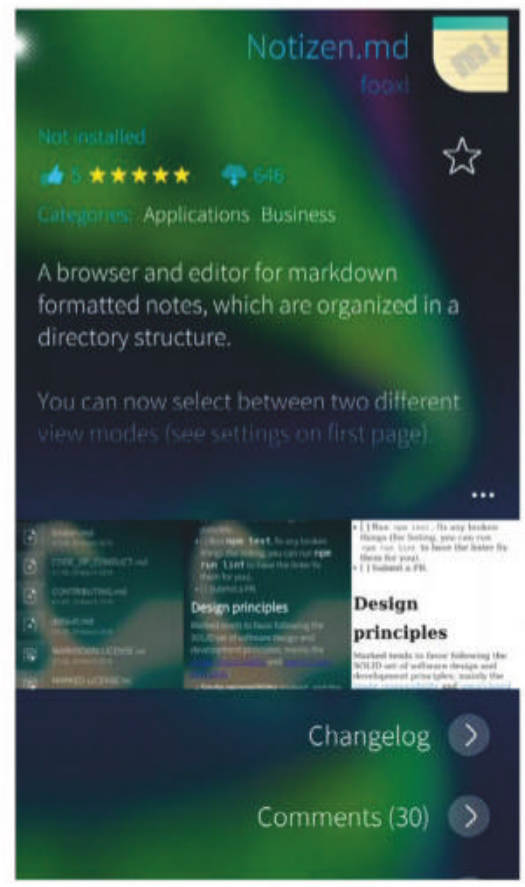
access using existing system settings and only wants a user to enable developer tools. Once you've done that, you can fire up a terminal and enjoy this mobile OS as if you've been in a desktop Linux console. Sailfish also gives you root access via the `devel-su` command and a password that's defined in the Developer Tools menu.

To manage software one can use either the default *pkgcon* package manager, or install *zypper*. The latter behaves the same as *Zypper* in openSUSE and enables you to manipulates repositories, RPM packages and package groups. Be careful, though: removing one of Sailfish core packages can eventually wipe away half of the system, rendering it inoperative. Software management is controlled in two ways. One is *Pkgcon* and *Zypper* for which you can optionally install the *Storeman* app as a GUI and obtain various community-made apps with ease. The other one is the official *Jolla Store* client, which lists select native apps together with some Android apps that are guaranteed to run in Sailfish. The experience of Jolla Store is perfect: both native and Android apps are installed seamlessly.

We spent a decent period of time using Sailfish X as our main mobile OS with a generally positive experience. We assessed the availability of essential apps, services and features. In that regard, most of things that Sailfish couldn't do by itself, such as instant payments or running high-quality games, were fulfilled by an appropriate Android app. Using community-made instructions it was possible to install the official *Google Play* app and enjoy the same comfort as with the stock Android OS, so you may not need to bother with APK downloads.

However, living with Sailfish X was still a compromise. The quality of Alien Dalvik is comparable with that of *Wine*: both run hundreds of non-native apps, but there are still things that crash or don't work at all. Jolla's offering is great, but it's not mature enough to rival stock Android-based ROMs, mainly because it still requires a lot of effort to keep things running. The lack of comprehensive documentation also makes solving common problems difficult.

From another perspective, Sailfish is suitable for every advanced user who seeks liberation from Google's non-removable apps, tracking and so on. Hopefully, Jolla will one day lower the barrier for this new audience and burst into the market with more devices, which will attract a lot of privacy-worried users. In contrast to that, Sailfish X without Alien Dalvik is purely a test environment for a small group of Sailfish and Mer developers. It's of limited use.



Storeman is a great way to obtain extra open source software easily.

Catching a Sailfish

Alien Dalvik isn't the only thing that ties Sailfish to Android. Another is Libhybris, a mediator that connects GNU Libc components with Android middleware. Libhybris makes it possible for Sailfish to run on handhelds for which Jolla doesn't have native hardware drivers. There's a list of supported phones, where each model has its own Libhybris-based adaptation and uses drivers from Android. Working with Libhybris was a compulsory step for Jolla, because a small tech company doesn't have enough resources to develop hardware drivers. Libhybris solves that and enables developers to adapt their OS to existing hardware, for which there are open source Android drivers. The Sailfish community has since produced many adaptations for various HTC, Motorola, OnePlus phones and others. They're more or less working, but the best Sailfish experience for newcomers remains buying officially licenced devices.

Small vendors often announce limited offerings in different parts of the world, and it's often possible to order such phones at a reasonable price. Such phones will support only native Sailfish apps, but this may sometimes be enough for daily life. There are dozens of high-quality titles at **https://openrepos.net**, including maps, office and email apps, games, sport trackers and utilities. The ecosystem is constantly growing! LXF

Experience mobile multi-tasking, where the OS doesn't kill background apps without your say-so.

VERDICT

DEVELOPER: Jolla Oy/Jolla Ltd.
WEB: https://jolla.com/sailfishx
PRICE: €49.90

FEATURES	8/10	EASE OF USE	6/10
PERFORMANCE	9/10	VALUE	7/10

Sailfish X is still not mature enough for widespread adoption, but it's moving in the right direction.

» Rating 7/10

GhostBSD 18.10

The prospect of experiencing BSD from within a familiar environment is too hard to resist for **Mayank Sharma**. Altogether now: who you gonna call?

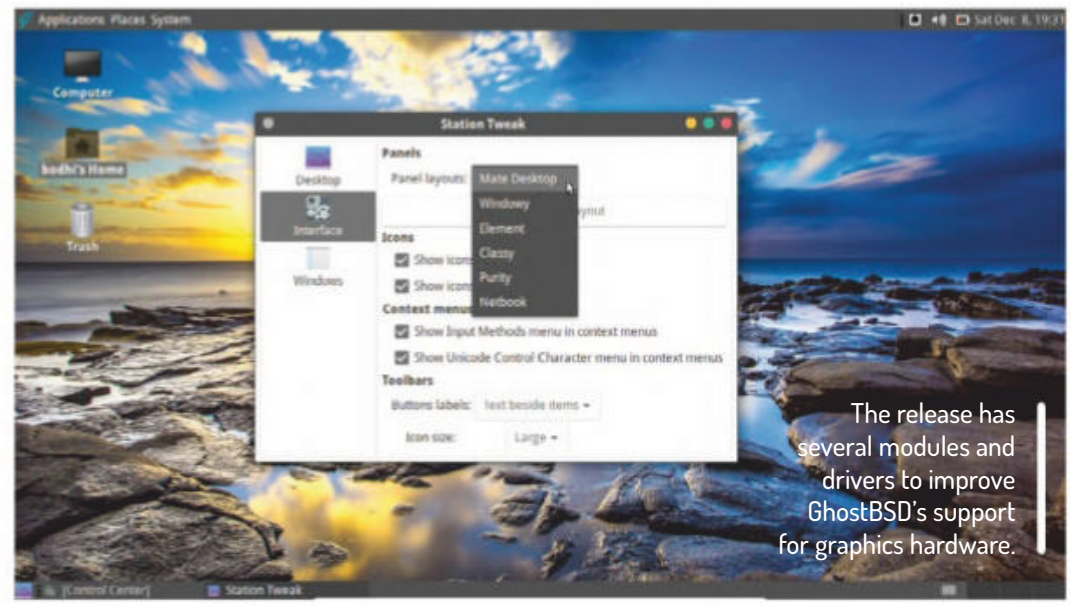
IN BRIEF

A BSD that closely resembles a typical desktop Linux distribution in both form and function. Unlike most of the other popular BSDs, GhostBSD boots straight into a live graphical desktop environment. TrueOS is another desktop-oriented BSD that's worth looking at.

Linux isn't the only open source operating system out there. While they haven't caught the fancy of the mainstream press, BSDs are popular with experienced users and administrators for their robustness, reliability and security.

Although you can transform a BSD installation into a desktop, it'll require some work. This is where GhostBSD shines. It offers a default out-of-the box experience that's virtually identical to Linux. On the face of it you wouldn't know you're running a BSD since the GhostBSD Live environment boots into the Mate desktop whose menus house all the familiar tools you'll find on just about every desktop Linux distro such as *Firefox*, *Thunderbird*, *Shotwell*, *LibreOffice* and such.

The latest release, GhostBSD 18.10 isn't any different from the previous releases in terms of the desktop. Behind the scenes however, the project has shifted allegiances. Instead of being based on pure FreeBSD, GhostBSD is now built from a FreeBSD-based OS called TrueOS. As a result, FreeBSD ports and packages are now incompatible with GhostBSD whose packages are built from the TrueOS ports collection. Furthermore, GhostBSD now inherits TrueOS goodies like the OpenRC init system and the LibreSSL stack. Another major change is the overhaul of the Live system, which has been rewritten to fix many issues. The live system is now a uzip compressed file that's uncompressed inside a chroot environment during boot. The developers have also removed the UnionFS file system because of several problems, which has restricted the dexterity of the live system.



The release has several modules and drivers to improve GhostBSD's support for graphics hardware.

Haunted introduction

You get the first sign that you're running a non-Linux OS while anchoring GhostBSD on to the hard disk. The OS relies on its own multilingual graphical installer called GBI that uses the pc-sysinstall back-end also developed by TrueOS. The installer offers an automated partitioning scheme if you wish GhostBSD to take over the entire disk and eases the process of getting the OS onto the hard drive. For familiarity on the CLI, you can choose to use the *Bash* shell instead of the default *fish* shell.

Instead of the GRUB bootloader, the OS uses the FreeBSD hybrid loader. The other big difference is the default filesystem. Instead of the familiar Ext4, GhostBSD offers a choice between two BSD filesystems UFS and ZFS. GhostBSD defaults to the developer's favourite ZFS. Once installed, the OS looks spiffy and is responsive on both older and newer hardware. For package management GhostBSD relies on the OctoPkg front-end to the *pkgng* package management system. You shouldn't have any issues navigating the package manager as it's fairly intuitive. You can also use FreeBSD's well-documented CLI package manager *pkg*, which can be a good starting point on the journey to get familiar with BSD.

It's also easy to update both the system as well as installed programs using *Update Station,* GhostBSD's custom update utility. It's the project's utilities that set it apart from other desktop-oriented BSDs. Besides *Update Station*, there's *NetworkMgr* that now supports multiple wired and WiFi card connection and *StationTweak*, which is a fork of the *Mate Tweak* app to customise the desktop. You can use *StationTweak* to toggle Compositing effects and control various elements of the UI.

One of the most interesting ones is the choice of the various panel layouts. In an interview with FreeBSD Bytes, GhostBSD's lead developer has said that the next goal of the team is to fix or replace more components from the Mate desktop that don't perform as per their expectations. In 2019, the project will also be pushing updates to the base system packages and work to improve the usability and functionality of the desktop. LXF

VERDICT

DEVELOPER: Eric Turgeon
WEB: www.ghostbsd.org
LICENCE: BSD License

FEATURES	**7/10**	**EASE OF USE**	**8/10**
PERFORMANCE	**8/10**	**DOCUMENTATION**	**7/10**

An impressive project that makes BSD accessible to first-time users. If dual-booting it with Linux is an issue, it'll keep chugging along happily inside a virtual machine.

» **Rating 7/10**

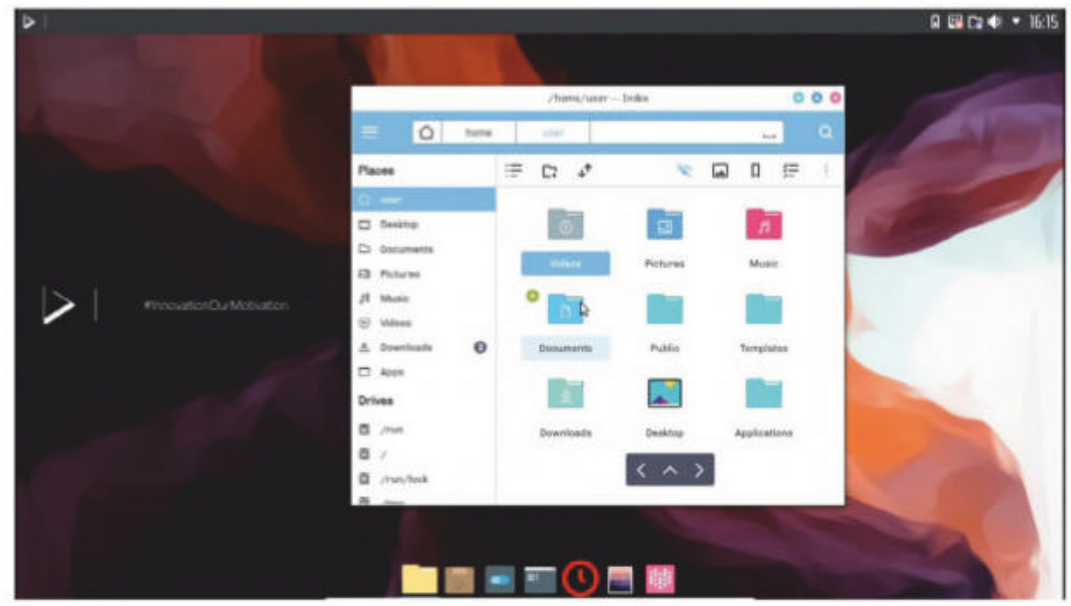
Nitrux 1.1.1

Exploiting his soft spot for good looking desktops, this release hits all the right notes to impress **Mayank Sharma**.

IN BRIEF

Ubuntu-based distribution that takes inspiration and components from the KDE project. The aim of the project is to build a modern desktop, which is why it doesn't support legacy hardware (no 32-bit image) and legacy boot mechanisms (no BIOS and no MBR partitions). It also features a long list of custom apps built atop a lightweight toolkit.

The last time we looked at Nitrux (Pg 21, **LXF235**), the distribution had all the makings of a pleasing desktop thanks to its slew of custom apps, but it wasn't suitable as an everyday desktop. We're happy to report that the latest iteration hasn't only improved its troupe of custom tools but has overcome the shortcomings of the earlier release.



When a new release pops up, znx can help you save a lot of bandwidth by doing a differential zsync update on the running ISO.

Our biggest criticism was its minimal set of default apps. This time around Nitrux 1.1.1 packs in all the apps you'd expect from an everyday desktop distribution. While there are a few mainstream ones like *LibreOffice*, *VLC* and *Chromium*, a majority of the apps in the distribution are developed in-house by the project using its Maui framework, which helps build apps that work across multiple form factors. Sure, they don't have all the advanced features of their mature counterparts, but the Maui-built apps have all the essentials ones. For instance, the *Index* file manager can preview files and has a built-in terminal, the *Pix* image viewer has basic image-editing functions, and the *Buho* note-taking app can string multiple notes into books.

Nitrux is built on the Ubuntu LTS series of releases and makes liberal use of the KDE Plasma 5 desktop and apps. Its *Nomad* desktop environment is based on Plasma and offers the same usability as the mature desktop, specially tuned for inexperienced Linux users. The desktop's *Nomad Simplemenu* application launcher is a spinoff of its KDE namesake, but just as straightforward to use. It divides apps between two screens, with one listing favourite apps and the other hosting an arranged list of all installed apps. The menu has a search bar to hunt for apps and you can add frequently-used apps as favourites for quick access. The other custom components of the *Nomad* desktop include the *Nomad System Tray*, the *Nomad Audio Controller*, *Nomad Networks* and the *Latte* dock at the bottom of the desktop. You can pin additional apps to the dock and change its location, alignment, appearance and behaviour. Besides its customs apps, the distribution also includes a host of KDE apps and utilities. There is however one disadvantage of bundling the different types of apps. While the Maui-built apps have menus smartly rolled into the apps, the KDE ones use a global menu. The mix of the two menu styles makes the desktop look inconsistent.

Another major Nitrux undertaking is the *NX Software Center*. The app has undergone a major overhaul in the backend. The earlier versions of the appstore were built around Snap images but the limitation of the Snap architecture to not support non-Canonical repositories didn't go down well with the Nitrux developers. Instead of snaps, the *NX Software Center* now installs AppImages. The frontend however can use some tune up. It's still very barebones with most apps sporting a one-line description and many missing an icon.

The other noteworthy enhancement in the distribution is the move towards an installer-less deployment. Inspired by AppImages, Nitrux has developed a tool dubbed *znx* to deploy distributions without repartitioning drives. You can use *znx* to prep a removable device and deploy the Nitrux ISO on to it along with a partition to house your data for persistence. The app is available as an AppImage so you can use it from your existing Linux distribution to transfer the Nitrix ISO on to a removable drive. On a related note it's also important to note that the Nitrix ISO doesn't use the MBR partition table nor does it use SYSLINUX. Instead it uses the GPT partition table and GRUB2 to boot only on UEFI-enabled computers. This is why you'll have to use the *znx* tool to transfer the image on to a USB disk instead of the usual mechanisms to flash the ISO. LXF

VERDICT

DEVELOPER: Nitrux Project
WEB: http://nxos.org
LICENCE: GPL and others

FEATURES	**8/10**	**EASE OF USE**	**8/10**
PERFORMANCE	**9/10**	**DOCUMENTATION**	**7/10**

Nitrux is working to leverage the latest developments in hardware and software into a modern desktop distribution. It use of AppImages and znx is praiseworthy.

» Rating 8/10

Warhammer 2

Living in a fantasy land and dispensing of minions offhand, **Jody MacGregor** think he's found a new game even The Management would enjoy…

SPECS

Minimum
OS: Ubuntu 18.04 64-bit
CPU: 3.4GHz Intel Core i3-4130
RAM: 6GB
HDD: 52GB
GPU: Nvidia GTX 680, AMD R9 285 (GCN third gen), 2GB VRAM
Note: Nvidia GPU v396.54 drivers, AMD GPU Mesa 18.1.5, Intel GPUs are not supported

Recommended
RAM: 8GB
GPU: Nvidia 970, AMD RX 480, 4GB VRAM

The name of the desert remains a mystery. It was somewhere near the Black Pyramid of Nagash, split down the middle by a crevasse. A narrow band of dirt connected the two sides. The undead, lacking missile troops, scurried across the sand towards this bridge. My High Elves moved to block them with a line of spears, but we'd forgotten that the general's last upgrade was a warhorse. Instead of moving up with the line he raced ahead unnoticed and was now alone, facing down sprinting ghouls, shambling zombies, and the Vampire Countess leading them.

Giant Skaven rat troops doing what they do best… scaring the bejesus out of everything else on the battlefield.

Rather than retreat we ordered the Silver Helm knights to rush in (we'd been keeping them in reserve to flank-charge anyone who broke through the spear line and pushed toward our archers), and instead of a strategic defence the Battle of Nameless Crevasse turned into a massed and messy cavalry charge. In war, they say no plan survives contact with the enemy. In *Total War: Warhammer 2* no plan survives our ability to get distracted by how nice the unit animations are.

Where the first game gave us a fantasy analogue of continental Europe, *Total War: Warhammer 2* includes mythed-up analogues of the Americas, Africa and Britain as well as a horseshoe-shaped continent in the mid-Atlantic called Ulthuan, home of the High Elves. It breaks up the ocean nicely so you're not staring at a ship for 50 turns to get anywhere, although there are also plenty of sunken treasure galleons and lost islands and dead behemoths to encounter along the way as well. Every time you set sail a little story plays out, which is a nice way of livening things up without having to add naval combat.

As well as being the centre of the ocean, Ulthuan is the centre of the campaign. In *Warhammer 2* you can win by old-fashioned domination or by meeting the objectives required to take control of the great vortex, a cyclone of magical energy hovering over an island in the middle of Ulthuan that the four playable factions all want to control using a sequence of rituals. It provides more structure to the game, and since everybody's after the same thing, it feels more like a race.

See that magical vortex of energy? You wanna get your grubby hands on that bad boy.

In the historical *Total War* games the factions are usually pretty similar, which helps make them balanced. *Total War: Warhammer* emphasised the differences between its fantasy armies. Vampire Counts had no missile troops; Dwarfs had no spellcasters; and Greenskins had no trousers. Those contrasts made for differences in strategy that gave it more life, and replaying the campaign became a much more enjoyable prospect than it was when choosing slightly different Japanese clans. We racked up over 200 hours in the first game due to that, although the DLC helped extend that longevity.

Fantastic four

At first glance the four factions of *Warhammer 2* seem less distinct from each other. There are two flavours of Elves – High and Dark – and two kinds of animal-people: the dinosaur-riding Lizardmen and the mad science rats called Skaven. All are good at magic, all have a mix of melee and ranged specialists. In play, they're more different than expected. Like the armies added to the first game as DLC, extra care has been made to make them mechanically distinct in both battles and campaign.

The Skaven, for instance, spread corruption like the Vampire Counts did, but this corruption affects their own troops as well as enemies. Instead of squatting in a castle brooding while the surrounding lands turn dark and then finally slouching across the border, the Skaven are constantly moving, ruining each province – literally, their settlements appear as ruins to casual inspection – and moving on. They struggle with hunger as well; their food supplies run low if they stop raiding. Playing as the Skaven means you're discouraged from settling down.

Compare that to the courtly High Elves, who instead only have to worry about a secondary currency called

Siege towers about to suffer more than scratched paintwork.

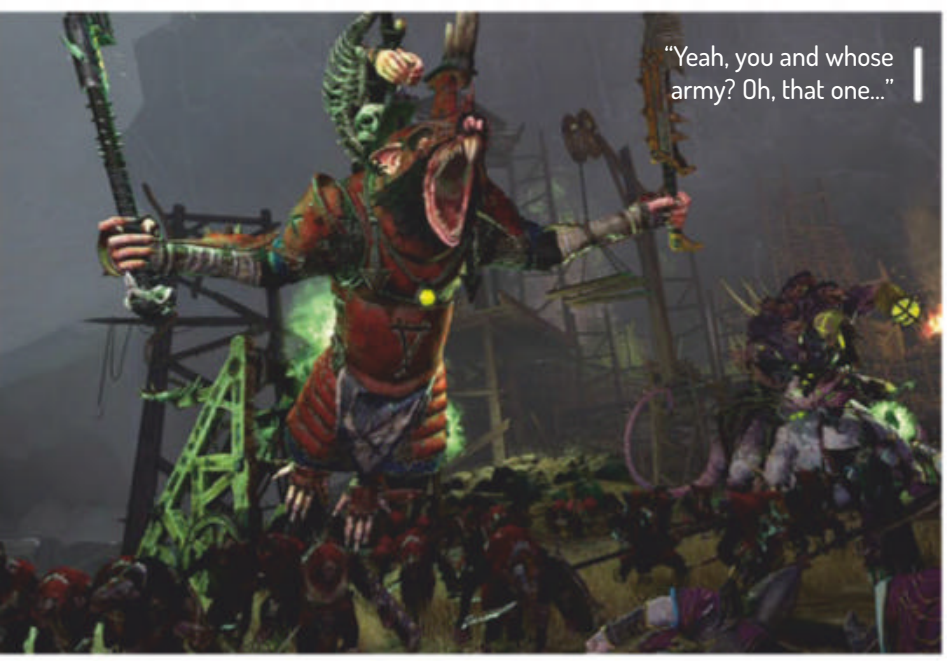
influence they can spend to engage in intrigue, altering how much other countries like each other. The High Elves can, with patience, win over distrustful and distant nations and turn them into allies, while breaking up alliances between their opponents. Their spies can also see any port they have a trade agreement with, lighting up far parts of the map. You can manipulate foreign affairs without even leaving your defensible island home.

Distinctions on the battlefield matter too. The Skaven's menace below trait means they can summon units from beneath the ground to block charges or harass archers. Some Lizardmen go into an uncontrollable rampage when hurt, while the Dark Elves can call in bombardments from their floating black arks if fighting on the coast. Magical units like the Skaven's screaming bell as well as giant monsters like the Lizardmen's carnosaur and Elven dragons all play their part.

You dirty rat

The ratmen are the most distinctive of the setting's creatures and its most beloved. So yes, if you play as the Skaven you get to throw glass balls full of poison at people, roll around in a giant electric wheel like a hamster, blast flamethrowers with abandon, and hear everyone talk in high-pitched squeaky voices. You get Rat Ogres who knuckle the ground like gorillas and bellow, and the unit animations really are distractingly nice.

There are so many tiny improvements in *Warhammer 2* that it's hard to list them all. The notifications when you press end turn without doing something the computer expects you to do aren't as obnoxious; heroes are less essential and the AI relies on them less; the map seamlessly zooms up into the tactical level; rogue armies made of mixes of different factions roam around to add variety to battles. Instead of constructing buildings that unlock another building's ability to produce a specific elite



"Yeah, you and whose army? Oh, that one..."

You'll have time to admire the scenery in between battles.

troop, you're often constructing buildings that unlock parts of the tech tree. These are all individually small tweaks but definite improvements nonetheless.

That does rather highlight the one thing that hasn't been improved, which is the bizarre side of diplomacy. People will still scream "Traitor!" if they have any reason to dislike you no matter how slight, and neighbours will greet neighbours as if meeting travellers from distant lands. Once you get used to the maths behind it the diplomacy starts to make a kind of sense, but the disconnect between what's being said and what's meant is still distracting. So is the endless repetition of the same demands being made every turn no matter how many times you refuse to sue for peace or declare war on a trade partner's enemy.

One new flaw is that the growing scale has resulted in a map with slightly more obvious gaps in it. The Southlands are a patchwork mish-mash of factions new and old, and presumably the pseudo-Egyptian Tomb Kings will be added to it as DLC. It would be great if the army of Araby (which last appeared in a tabletop game called Warmaster in 2009) showed up as well, since their home is full of zombies and white guys. Over in the Lustrian jungle, home of the Lizardmen, the Amazons are another omission.

Though the first game's aggressive DLC schedule upset some players, it was clearly a success. And the free offerings alongside each one seemed generous enough – in fact, the Bretonnians were better than any of the paid DLC factions. Among the additions coming for free this time is the ability to combine both games into something called the "Mortal Empires Campaign", making this ridiculous map even bigger.

Total War: Warhammer was a great game at launch that became even better as it grew. *Warhammer 2* feels like it's starting from the point it left off from. LXF

VERDICT

DEVELOPER: Feral Interactive
WEB: www.totalwar.com
PRICE: £40

GAMEPLAY	**9/10**	**LONGEVITY**	**9/10**
GRAPHICS	**8/10**	**VALUE**	**8/10**

A maximalist sequel that improves on pretty much every aspect of the first game. Highly recommended.

» Rating 9/10

Overseas subs on p67

» PLUS: Exclusive access[1] to the *Linux Format* subs area!

1,000s of DRM-free PDF back issues and articles! Get instant access back to issue 66 (May 2005) with tutorials, interviews, features and reviews. At **linuxformat.com**

1) Only available to MyFavouriteMagazines.co.uk subscribers.

» CHOOSE YOUR PACKAGE!

PRINT EDITION

» Every issue comes with a 4GB DVD packed full of the hottest distros. Plus your free **STM Trilogy backpack**

Only £76

Annual print by Direct Debit

DIGITAL EDITION

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

Only £22

Every 6 months by Direct Debit

PRINT + DIGITAL

» Get both the print & digital editions for one low price. Plus your free **STM Trilogy backpack**

Only £90

Annual bundle by Direct Debit

*Terms and conditions: This offer is only available for new UK subscribers taking out a Print or Print & Digital subscription. Gift is subject to availability. Please allow up to 60 days for the delivery of your gift. In the event of stocks being exhausted we reserve the right to replace with items of similar value. Prices and savings quoted 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. UK calls will cost the same as other standard fixed line numbers (starting 01 or 02) or are included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit: **www.bit.ly/magterms**. Offer ends 31 March 2019.

Roundup

Antergos » ArchLabs » KaOS »
Manjaro » Netrunner Rolling

Jonni Bidwell
required judicious application of Jura whisky to fend off the germs that tried to interfere with this *Roundup*.

Arch-based distros

Ever more distros are being built upon Arch Linux. Once **Jonni Bidwell** would have called this sacrilege, but now he finds it a glorious development.

HOW WE TESTED...

We installed each distro on a Dell XPS13, which enabled us to test HiDPI and touchscreen support. We checked that the distros happily installed alongside other operating systems without breaking them. Further testing was done in virtual machines (limited to 3GB of RAM), which gave us some idea of how the distros would run on older hardware.

We looked at what came with an out-of-the-box install, and whether this struck the right balance between usefulness and bloat. Since we've focused on distros providing a desktop, we looked at how stylish and effective those desktops were. Bear in mind that it's possible to install and configure alternative desktops to your liking.

We compared repositories to Arch Linux's, renowned for offering the very latest software and making minimal changes to upstream releases. In particular, we looked at the desktop packages used by each distribution. We also checked how each distro integrated with the Arch User Repository (AUR).

Few would disagree that Arch is a fine distribution: cutting-edge, customisable and possessed of an enthusiastic community and excellent documentation. But it's not for everyone.

Many people are put off by the lack of an installer: having to manually partition disks and `chroot` into a bootstrapped install to get things going doesn't really score any user-friendliness points. Those that surmount that obstacle are met with a new challenge: namely, building the minimal install into something useable. If you're installing it for desktop use, you'll find getting that desktop to look and play nice is decidedly non-trivial. In short, it really makes you appreciate the efforts that other distros go to, to have all this set up out of the box.

But the goodness of Arch can be married with the convenience of other distros. Imagine never having to add a PPA to obtain new releases. Imagine being able to rebuild custom versions of stock packages. Imagine, and then imagine no more. We're focusing on desktop varieties here, so we'll be paying particular attention to modern desktop innovations, but people have used Arch Linux as the base for all kinds of things (security, Raspberry Pi, 32-bit hardware, the list goes on), such is its flexibility.

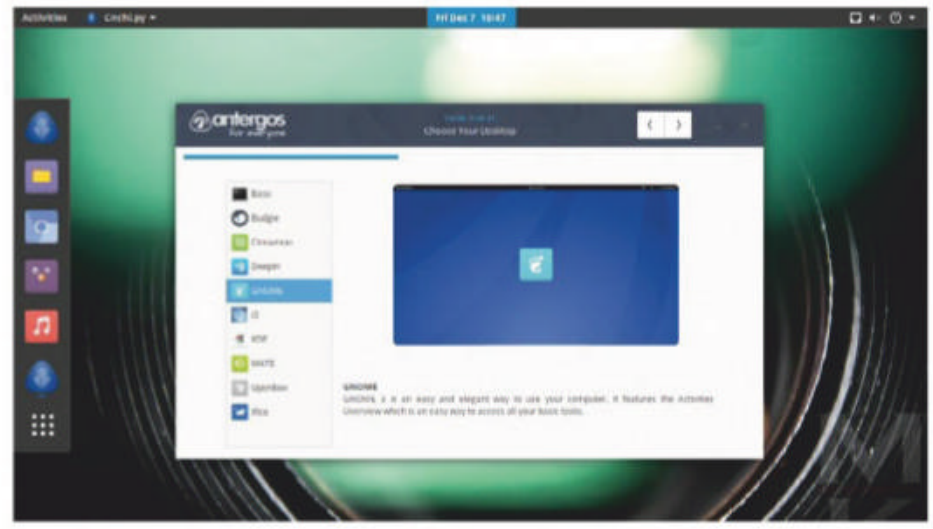
Install and first run

Does the installer welcome you?

Antergos impresses right from the outset with the stylish *Cnchi* installer. You can choose to set up popular applications and desktops. These all need to be downloaded, which puts the kybosh to offline installs, but it does do installees the courtesy of offering to find the fastest mirrors. If you're worried about the install going south and having to start over then you can opt to create a recovery partition on the target device. This is a nice touch, but mercifully not one we had reason to use. You can even choose between GRUB2 and systemd-boot, encrypted installs and even using ZFS.

Manjaro's live medium starts with a slightly spartan menu for selecting keymaps, free vs non-free drivers or adding kernel parameters. Once the *Calamares* (*see* **LXF233**) installer fires up we're met with the usual options (LVM and encryption) and you can choose between wiping your drive, installing alongside other OSes, or manual partitioning. Once you boot into Manjaro, a welcome screen displays release information and provides links to the Wiki. There's a handy tool for installing popular apps, too.

Netrunner, being a Manjaro descendent, has a similar installation process. Newcomers might be a little upset to find the Readme icon on the desktop takes them to a 404 page, with a mixed content warning, but this is a cosmetic thing. KaOS also uses the *Calamares* installer. XFS is a slightly left-field choice for a default filesystem, but why not? Once installed the *Croeso* (Welsh for "welcome") tool greets you and provides shortcuts to documentation and common system settings. This is handy if you don't know where Plasma hides these things, and even if you do it's nice to have them all in one place. *Croeso* has a few more options than Manjaro's welcome tool.



Antergos is something of a desktop shape-shifter. Much like its Arch parent, it can be anything that you want it to be.

ABIF, Archlabs' Ncurses installer isn't newbie-friendly, but offers the same options as Antergos and carries out a full system update before rebooting. Since the live medium hasn't been updated since July 2018, this is a fairly lengthy process. Once you reboot, a menu enables you to install popular applications, fonts and so forth. If you don't like logging in from the console, the LightDM display manager is readily available from here, too.

VERDICT

ANTERGOS	**9/10**	**MANJARO**	**8/10**
ARCHLABS	**6/10**	**NETRUNNER ROLLING**	**8/10**
KAOS	**7/10**		

Antergos's Cnchi sets a new standard for Linux installers. ArchLabs's textual experience can't compete, but console warriors won't mind.

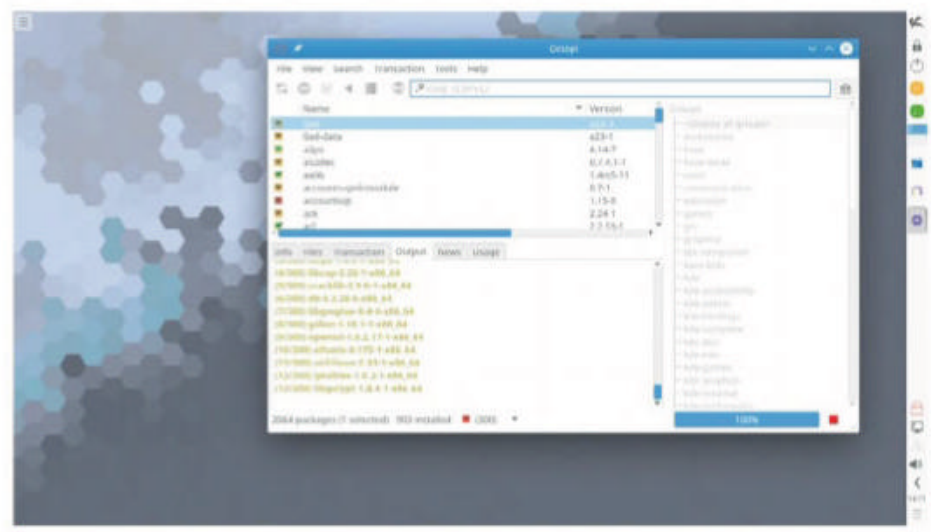
Repositories

How do these repos stay up to date?

Be mindful that we're using the stock Arch repositories as a baseline here, and short of packaging things yourself or using snaps this is pretty much the fastest way to get the latest upstream releases.

Manjaro takes a leaf out of Debian's book, offering Stable, Testing and Unstable branches. The Unstable packages track around three days behind the Arch repo, which should be fresh enough for anyone. Packages make it into stable "when they're ready", but this is usually pretty swift unless some exigency arises. At the time of writing there's a stability issue with new versions of Systemd, but otherwise we found no major differences. Netrunner Rolling is based on Manjaro's Stable channel (though a handful of custom packages come from its own repo), so it can enjoy the 12,000 odd packages in Manjaro's repos.

KaOS tends to lag behind a little further, and the only graphical apps you'll find in its repos are Qt ones, such is its KDE Plasma focus. If you need anything else, you'll have to build it yourself. You can do so from the KaOS Community Packages repo, which given KaOS's limited repos, vastly widens its scope of interest. KaOS uses Plasma 5.14, compared to Netrunner's Plasma 5.12, and both currently use version 18.08 of the KDE applications suite, hopefully the just-released 18.12 apps bundle blesses these KDE distros soon. Finally, Antergos maintains a handful of its own packages, but otherwise relies on the Arch Linux repos, and ArchLabs uses exclusively the Arch repos.



KaOS has only a couple of thousand packages in its repositories. This is a fraction of what you get with Arch, but maybe all you need.

VERDICT

ANTERGOS	**8/10**	**MANJARO**	**9/10**
ARCHLABS	**8/10**	**NETRUNNER ROLLING**	**7/10**
KAOS	**8/10**		

Manjaro's channels cater to all users, and their generous repositories are almost as well-stocked as Arch's.

Style

Which distro offers desktop panache?

Comparing different desktops is a tricky business, especially when we have species from all over the spectrum. We're not going to mark Archlabs down because it uses a lightweight window manager, and we're not giving extra points to Antergos because it uses a readily available (though undeniably cool) icon theme.

Instead, we'll focus on how each desktop is styled relative to the components installed. We're interested in icons and colour schemes working together, considered layouts, and freedom from clutter. Since we're au fait with setting up various desktop environments from scratch in Arch, we'll give credit for sorting out common quirks or deviations from the defaults (as long as they're useful deviations).

Whatever desktop you end up using, since Arch always ships the latest releases, you'll probably end up with something that looks and acts better than it does in other distros. And if you don't, well you might at least have found a brand new bug, or you might have caught a glimpse in to what users of other distros will be kvetching about in a few months.

Antergos Gnome 7/10

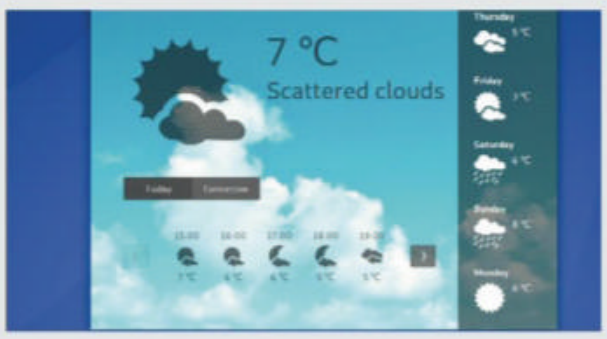
Antergos is our only Gnome offering, but even before we'd got there we were impressed with the stylee console font setting during boot. Dash-to-dock provides a launcher on the left (à la Unity and Budgie), emblazoned with icons styled in the bold, Numix Square theme which is used throughout the desktop. The bright orange directory icons in Files might not be to everyone's tastes, but we liked them. Tweaks and the User Themes extension are installed for tweaking and theming things to your taste.

If you choose to use Gnome with Antergos then you'll probably want to replace the clunky-looking LightDM greeter with GDM, and there's a wiki page dedicated to doing just that. If there were an award for 'largest title bars' then Antergos would win, but Gnome apps that use new-fangled GTK3 headerbars look quite comfortable with them.

ArchLabs 9/10

We like the minimal, dark stylings applied to OpenBox and the Polybar panel. Thanks to Compton, some shadows and fades mean the desktop, albeit understated, still looks modern. Archlabs' colourful icon theme contrasts nicely with all the darkness. You can switch Polybar for Tint2, or add a dock (such as Plank or Docky), all from the comfort of the Preferences menu.

If you feel Conky is polluting your desktop background, then it can be taken away too. It does provide some helpful hints though, most importantly that the Drun launcher is available by pressing the Super key. The visual cues on the desktop pager (Home, Web, Files and Pictures) are great for making the most of four virtual desktops. A few Xfce components are used and a few more – or indeed the whole of Xfce – can be added from the welcome menu.



Package management

How do our candidates tame Pacman and integrate with the AUR?

All the distros enable you to use excellent Arch's Pacman tool, but this is probably not ideal for people who want to do things graphically. The *Pamac* GUI is included in Antergos and the Xfce and Gnome flavours of Manjaro. This is a little bit slicker than the *Synaptic*-like *Octopi* used by Netrunner and KaOS (and Manjaro KDE). KaOS includes a GUI cache cleaner and package log, but command-line tools can do these tasks just as well. Netrunner also offers the appstore-style KDE *Discover*. ArchLabs enables *Pamac* to be installed from its welcome menu, but doing things CLI-stylee seems more fitting here and, speaking to that, it includes the text-based *Pacli* front-end. *Pacli* can help with downgrading packages, something not supported in Arch Linux but something that is on occasion necessary. Manjaro's kernel tool will advise you of new kernels, it's in Netrunner too, but we found we had to invoke it manually to upgrade to a new major version. This could've been impatience on our part.

Pacman can be further augmented by various helpers for building and installing custom packages from the famously useful Arch User Repository (AUR). *Pamac* and *Octopi* both show new package notifications and can optionally integrate with the AUR. However, AUR packages may not get on with anything from outside the Arch repos, and so some caution ought to be exercised on those distros that use custom repos, particularly Manjaro. Currently, ArchLabs (where you're least likely to run into AUR conflicts) uses the *Aurman* helper, but since development of this is fizzling out *Yay* is probably a better bet. Alternatively, *Pacli* can work as a front-end to the once premier, sadly now-defunct *Yaourt AUR* helper, too.

VERDICT

ANTERGOS	**8/10**	**MANJARO**	**9/10**
ARCHLABS	**6/10**	**NETRUNNER ROLLING**	**7/10**
KAOS	**8/10**		

Manjaro enables you to care for your packages any way you want. ArchLabs loses out for a lack of GUI offerings, but again, this may not matter to you.

KaOS 7/10

KDE has long been noted for its configurability. This can be a curse as well as a blessing. KDE 4 in particular dispersed options all over the shop, and also shipped with a near-infamous window glow enabled by default. KDE Plasma 5 has a perfectly cromulent default setup, but offers plenty of options to manipulate this.

Being a KDE Plasma affair, we were expecting KaOS to show us something a little different from usual KDE layouts, and that it certainly did. A menubar on the right-hand side of the screen, and Dolphin demonstrating similar right bias with its navigation panel. You can probably get used to this, but it's a jarring first experience. Call us old-fashioned (*'old' is more accurate – Ed*), but we were much happier with the panel at the bottom, and the file manager navigation panel on the left. We heartily approve of the Powerline terminal stylifier, though.

Manjaro 8/10

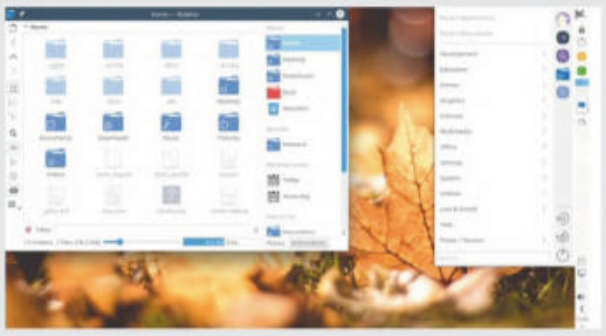
The principle Manjaro edition is the Xfce one. This isn't going to resonate with people wanting desktop bling, since Xfce hasn't changed in nearly three years. But it's functional, far from ugly and gentle on system resources. It also features display scaling, so 4K users won't be left squinting at tiny fonts, but setting this up is more involved than KDE/Gnome.

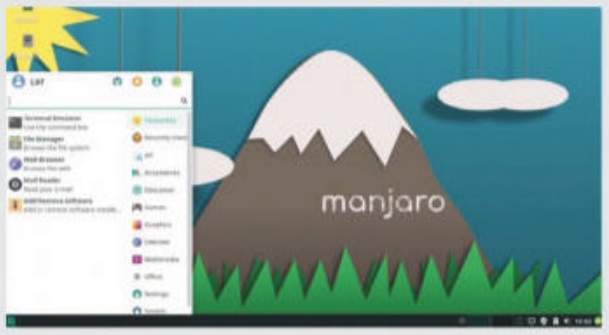
Remember that official Manjaro flavours exist for those desktops, and community flavours exist for others. Traditionalists will appreciate the inclusion of a shade control (for 'rolling' windows into their titlebars), and the ease and configurability of appearance/window manager settings. The default (Papirus-Maia) icon theme is bold without being overbearing, and there are several other icon and GTK themes to choose from. Kvantum theme manager can jazz up Qt apps, but that'll be more useful in the KDE edition.

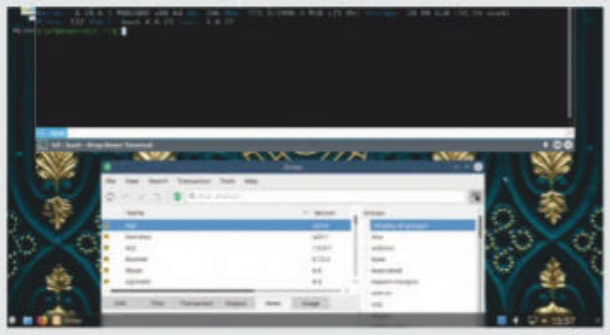
Netrunner Rolling 7/10

Netrunner's Plasma setup features the full screen application dashboard, which like Gnome's can also search documents. The large fonts look a little odd, though. We much prefer Plasma's hybrid launcher, and you might prefer the classic cascading menu used by KaOS. The black panel looks stylish and we can't get enough of drop-down terminals so we applaud the inclusion of Yakuake.

The My Computer and Network icons on the desktop seem a bit Windows-ey, but are easily ignored/expunged. The Breeze AlphaBlack theme, powered by modern versions of Plasma (5.14) and Qt (5.11) give some neat transparency effects. Minor annoyance about the dead link on the desktop, but plenty of nice wallpapers are included. The Breeze theme has a GTK counterpart, so such applications will conform to the desktop finery. No desktop widgets are on by default. Good.







Documentation and support

Where can you turn when it all goes sideways?

The Archwiki is a truly great learning resource. It's well-written (and consistently formatted), covers pretty much everything you can imagine, and is kept up to date. A lot of what you'll find there is applicable to other distros, all the more so if Arch is their progenitor. The Arch forums are likewise a veritable fount of knowledge, but beware – fools are not suffered lightly there.

In terms of bespoke support Manjaro shines ahead here. It has its own mailing lists, bugtracker, IRC channel, a comprehensive wiki and friendly (relatively speaking) forums. There are also subforums for discussion in about 30 other languages, because Manjaro users are a diverse bunch. Netrunner benefits from this by osmosis, but they too have their own (rather more modest) forum.

The Antergos wiki appears pretty threadbare, although it does have a thorough guide to setting up dual-booting on Windows 10 which ought to appease newcomers. It also has a thriving, multilingual forum and an IRC channel. The ArchLabs forum is likewise buzzing. However, beyond some handy hints about the included programs and customising the desktop there's not a lot on the wiki. The smaller distros really benefit from community involvement and so if you solve some niggling problem (whether or not someone's asked about it) do consider adding your knowledge. For the greater good and all that.

The KaOS team use several bugtrackers and has a quiet forum. Its website has a handy guide to asking and reporting things sensibly and the world would truly be a better place if everyone read these kinds of things.

VERDICT

ANTERGOS	**8/10**	**MANJARO**	**9/10**
ARCHLABS	**7/10**	**NETRUNNER ROLLING**	**6/10**
KAOS	**6/10**		

Manjaro's huge userbase means a strong community, many of whom won't default to "RTFM" if you pose a genuine question.

Notable additions

Sometimes it's the little things that make all the difference...

Netrunner includes the retro Burgerspace game. It's just a shame that we didn't have time to play it much. Honest.

Setting up printers in Arch is a bit of a faff, but Antergos and Manjaro include Redhat's *system-config-printer* utility. All our distros bar ArchLabs include the *HP Device Manager*, which is great if you have such devices to manage.

Netrunner includes Grub customizer, a nice touch, and like Manjaro also includes out-of-the-box support for hibernation. *KDE Discover* is a good way to, um, discover new applications and Plasma addons. It's much more solid than it used to be and we found it much less annoying than *Gnome Software*. Netrunner's Firefox install comes with uBlock Origin and several popular blocklists installed, making the web a marginally less obnoxious place.

KaOS includes a Wayland session which is a nice/brave touch for a KDE distro. Gnome on Antergos will try and start a Wayland session. Being lazy typists, we were sad to see the `bash-completion` package missing from Manjaro and Netrunner, but heartily approved of KaOS's Ash configuration and the tab completion there. Manjaro and KaOS both feature simple frontends for the UFW firewall. We think it's high time that intermediate and higher users start paying attention to which packets get in or out, so applaud both these distros. Antergos and ArchLabs both retain Arch's customisability, primarily by providing direct access to the multitudes of packages in the Arch repos. We think more people should give lightweight desktops a chance, and liked the Openbox/Polybar/Compton setup that came with ArchLabs. Setting this up from scratch is much harder than recreating KDE/Gnome/Xfce setups.

VERDICT

ANTERGOS	**7/10**	**MANJARO**	**8/10**
ARCHLABS	**8/10**	**NETRUNNER ROLLING**	**8/10**
KAOS	**9/10**		

There are hidden gems in all our offerings, but the KaOS devs have really thought about what their users will appreciate.

Included tools

What's the out-of-the-box experience?

We're not sure we know anyone who uses Handbrake, Vokoplayer and gmusicbrowser. If you do, Netrunner has you covered.

Antergos installed a light selection of the Gnome application suite, but more than enough to get started. Having Gnome's new *Documents* tool is a nice touch, likewise *Weather*, *Maps* and *Books*. There's no *LibreOffice*, so paper-pushers will need to do that themselves, but *Chromium* is installed which will keep 50 per cent of everyone happy. Critics of *Gnome Software* will be appreciative of its absence.

Manjaro (and by extension Netrunner Rolling) includes a hardware detection tool (*mhwd*) that runs on install and can be conjured up post-install from the settings manager or terminal. There's a handy tool for adding/removing kernels too, which is great if you run into hardware regressions. It's doubly useful in Netrunner Rolling if you need to roll with a newer (or even real-time) kernel. Less useful in both these distros is the Flash player.

Gamers will appreciate the shortcut to the Steam installer included in the Manjaro menu. And some will appreciate the links to the *MS Office Online* web tools, but others might just see these as needless menu overcrowding. NetRunner comes installed with a number of multimedia applications. There are some fun games too and we approve of the *GIMP-Inkscape-Krita* graphical trio. *Skype* and Steam and *Virtualbox* are available, too.

KaOS features the *Elisa* music player, the *Karbon* vector graphics tool and the *Seafile* syncing utility. What distinguishes it from the rest is the use of the *Calligra* office suite. This looks stylish in a Plasma setting and is in many ways less clunky than *LibreOffice*, but will be hard for some people to get used to. The QtWebEngine-powered *Falkon* (formerly *QupZilla*) is a great web browser if you don't need anything fancy, but some will be itching to get something more mainstream installed.

ArchLabs is proud of its minimalism but you can install anything you need from its welcome menu. Music and movies are taken care of with *mpv* and *Audacious*. In a nod to developers it includes the *Geany* code editor, as well as Arch's base-devel package group. We approve of the inclusion of *Nmap*, and *Firefox* will please a different 50 per cent of users.

VERDICT

ANTERGOS	**7/10**	**MANJARO**	**9/10**
ARCHLABS	**6/10**	**NETRUNNER ROLLING**	**6/10**
KAOS	**7/10**		

If you think less is more, pay no heed to ArchLabs score. Manjaro have really sorted the wheat from the chaff though.

Arch-based distros
The Verdict

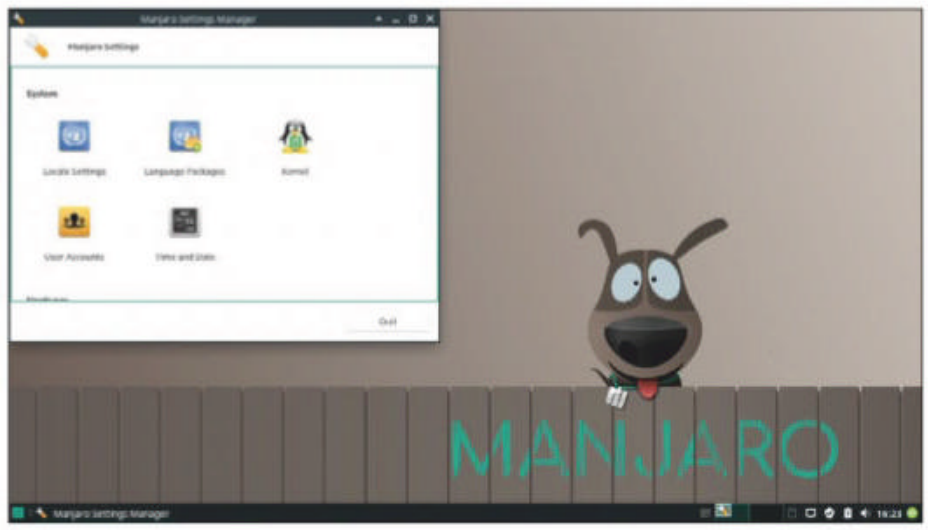
For some users, having a purely GTK or Qt system will be a boon, even if it drastically restricts the number of applications that are available to them. For those people Antergos, and in particular KaOS, may have some appeal. Others will be happy that cross-toolkit theming is in much better standing than it once was and not mind installing both toolkits. Storage is cheap, after all.

In many ways we felt that KaOS should be offering newer KDE packages, being as it does hit the Arch repos soon after release. Of course, KDE neon is considered the premier distro for showing off the latest and greatest on planet Plasma, so maybe there's no point competing with that. The Blue Systems sponsored Netrunner Rolling is a fine effort, but we felt it was trying to cover all bases.

Fantastic efforts of each distro notwithstanding, users who are unfamiliar with Arch and its ways of working will sooner or later find themselves confused. Manjaro may be the most user-friendly of this month's picks, but Ubuntu or Mint it is not, and treating it thusly will lead to disappointment. (*Life is full of disappointments, but that's a story for another Roundup.*) You'll probably have to get your hands dirty at some point. Conversely, if you're familiar with Arch then you may very well prefer to stay there (*up on that high horse you mean? – Ed*), or perhaps take inspiration from these distros and apply it to your Arch install. We were certainly enamoured by ArchLabs' desktop zen, and what can be done with a less than 200MB memory footprint.

Manjaro does a great job with its flagship Xfce edition. It shows that this desktop is capable of looking good as well as being considerate on resources. It's become hugely popular over the past year, not so much because of people's gripes with Ubuntu, but because it's a darn fine distro that gives people exactly what they want. When Xfce 4.14 is released and journalists become tired of making jokes about that, this will surely become ever the more greater.

The Gnome and KDE flavours are fine too. Great if you prefer the latest desktop fripperies, but we admire Manjaro's boldness in focusing on Xfce.



1st Manjaro 9/10

Web: https://manjaro.org Licence: Various
Version: 18.10
Manjaro has tamed Arch into something both humans and robots can enjoy.

2nd ArchLabs 9/10

Web: https://archlabslinux.com Licence: Various
Version: 2018.07
You don't need an ancient machine to enjoy a lightweight distro.

3rd Antergos 8/10

Web: https://antergos.com Licence: Various
Version: 18.11
Fantastic installer and a great all-round distro.

4th Netrunner Rolling 7/10

Web: https://netrunner.com Licence: Various
Version: 2018.08
KDE Plasma and Arch are a winning combination, but still a few rough edges.

5th KaOS 7/10

Web: https://kaosx.us Licence: Various
Version: 2018.10
One for Plasma/Qt purists, but this makes it limiting for everyone else.

» ALSO_CONSIDER

Several Arch-based distros have come and gone over the years, but some have stood the test of time and new ones spring up regularly. It's common for distros to add non-free software to improve the user experience, but Parabola and more recently Hyperbola go to the other extreme. These two, as well as being named after curves, are both FSF-approved and use the Linux-libre kernel. Hyperbola takes inspiration from Debian, using an older, tried 'n' tested kernel. Arch may have stopped supporting 32-bit, but the Arch Linux 32 distro is alive and well.

Arch-XFerience and Anarchy Linux, both Xfce affairs, are worth a look. Chakra Linux, a KDE offering, impressed us last year, but hasn't been updated since then. Those wanting something even more lightweight than ArchLabs should check out ArchBang, which was inspired by the now-defunct CrunchBang. In a roundabout way this inspired ArchLabs too.

THE ULTIMATE OPEN-SOURCE TOOLKIT!

As **Mayank Sharma** knows, it's vital to have the right tools for the job. That's why he's burnt his fingers to get you these red-hot pieces of open source software.

CONTENTS

Ultimate open-source toolkit

The mainstream Linux distributions spend a lot of time selecting their default bouquet of programs. After all, they need the tools to cater to the largest number of users. Sure, these programs excel at what they do, but there are thousands of other brilliant software lurking in code-sharing silos like GitHub and Sourceforge, and even inside the official repositories of your favourite distribution, waiting to be discovered.

Realistically though, do you have time to try all of these tools? Do you even want to, considering that your needs are served well enough by the distro's default options? Add to that the fact that most of us have our favourite open source apps and a conviction that they work for us better than any available alternative. That said, we'll encourage you to take a step outside your comfort zone to marvel at the diversity of the Linux-verse. And don't let the number of low-quality, unmaintained programs deter you from exploration.

EXPAND YOUR FRONTIERS

"Don't let the number of low-quality, unmaintained programs deter you from exploration"

At *LXF Towers* we spend a lot of time rummaging around the source code mirrors and other places to discover new gems to cover in the magazine. This issue we've decided to collate all the best ones we've encountered in our travels. We've stayed away from including the popular mainstream tools, but you might still be familiar with some of them because they're just that good.

For the most part though, the next few pages will introduce you to graphical apps and CLI utilities that you've never come across before. No matter the type of user you are or how you use Linux, the following pages will give you plenty of open source goodies to spruce up your Linux box.

Image tools

Check out these nifty options for managing or editing your image collection.

PENCIL 2D

You can use *Pencil 2D* to create both static and animated drawings. It's a comprehensive program that will help digital designers create professional-grade cartoons and animations in multiple layers. The intuitive tool is well complemented by the video tutorials on its website. **www.pencil2d.org**

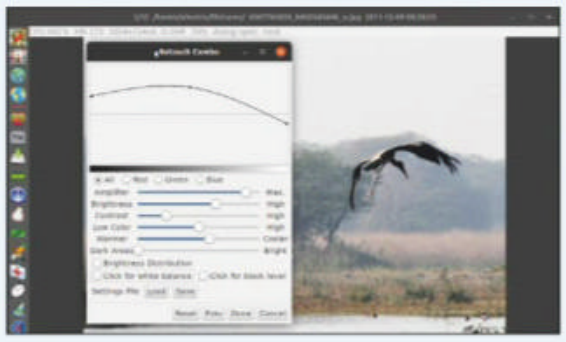
DARKTABLE

If you work with RAW images, *darktable* will help you create a professional digital darkroom to process them. The tool's interface gives you access to a full gamut of image manipulation and editing tools on par with commercial heavyweights from the likes of Adobe and Apple. **www.darktable.org**

FOTOXX

A comprehensive image-manipulation program with a rich set of retouch and edit functions that go beyond changing brightness, contrast and colour. Like most dedicated image editors, *Fotoxx* enables you to select an object or area within an image using various tools such as freehand outline, follow edges and select matching tones. Another unique feature of the tool is that it makes it possible to edit the images without using layers. You can also use *Fotoxx* to create HDR and panoramic images, reduce noise and remove dust spots, create collages, mashups, and slideshows with animations. There are also several artistic effects to help convert a photo into a line drawing, sketch, painting, embossing, cartoon, dot image, or mosaic.

On the initial launch, *Fotoxx* fires up its quick start guide in the web browser, along with a dialog to index your image library. This process can take some time depending on the number of images you have in your library. All its features can be accessed from the relevant options that are clearly labelled in the left-side panel. **https://kornelix.net/fotoxx/fotoxx.html**



You can use Fotoxx on resource-strapped machines and the program can be navigated entirely with the keyboard.

RAPID PHOTO DOWNLOADER

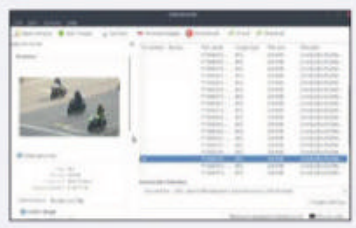
This program might seem redundant considering that most photo-management tools can import photos themselves. *Rapid Photo Downloader*, however, is designed for transferring photos and videos. It offers a lot more functionality that make it a perfect tool for downloading, processing and organising photos and videos.

The tool gives you control over how it processes and sorts the downloaded photos. The program's default rules automatically transfer the downloaded photos inside date-based subfolders.

You can also define custom rules as per your requirements. For example, you can ask the program to sort photos by their type, which comes in handy if you shoot both RAW and JPEGs. In fact, you can create complex subfolder hierarchy by defining naming rules based on Exif values, such as focal length, ISO, and so on. You can also specify an external USB storage device as the back-up destination, and the program will automatically back up the photos while downloading them from the camera. **https://damonlynch.net/rapid**

CONVERSEEN

Converseen is a straightforward frontend to various command-line conversion utilities that can convert images to and from over 100 formats, rotate and flip them, change their dimensions, and rename them in a fraction of the time it would take to perform these tasks manually. **http://converseen.fasterland.net**



Manipulate any number of images within the Action panel.

FROGR

Use this tool to upload content to Flickr and also access its basic upload features such as the ability to describe images, set specific licenses and categorise them into sets and group pools. **https://mariospr.org/category/frogr**

NOMACS

This image viewer can display images in all the popular image formats. In addition to the usual features it has some surprising ones such as the ability to synchronise viewing between multiple network instances. **https://nomacs.org**

ENTANGLE

Use this tool to tether and control your camera from the computer. *Entangle* will also help you set up the shot by tweaking the aperture, shutter speed, ISO and other settings and then download the image. **https://entangle-photo.org**

FLAMESHOT

You can use *Flameshot* to capture the whole or a specific portion of the screen. Additionally, it also provides you with a whole set of drawing tools to add annotations to your screenshot. **https://github.com/lupoDharkael/flameshot**

LYCHEE

Lychee has a simple user interface, and is a platform for storing and sharing photos, You can run it from a SBC like the Raspberry Pi and use it to upload, manage and share photos. **https://lychee.electerious.com**

Audio and Video

Edit your videos within an inch of their lives, and organise your music.

UMS

This streaming server will transform your Linux desktop into the ultimate media streaming station to broadcast media to other computers, smartphones, tablets, gaming consoles and even TVs. *UMS* streams media via the Universal Plug and Play (UPnP) protocols to any DLNA-compliant device. You'll have to roll-out *UMS*, although the procedure is fairly simple.

The first time you launch it, *UMS* takes you through a basic three-step configuration wizard. Although you can start streaming without further configuration, UMS does include an admin panel that offers customisable options and tips to guide new users. It also includes a minimal web interface for streaming your content.

www.universalmediaserver.com

MPV

A resource-conscious video player that also looks good on modern machines. Based on *mplayer2*, *MPV* continues the tradition of *CLI* by introducing optimised and cleaned-up code with new configuration options and features. It offers a minimal user interface that stays out of the way, enabling you to watch your videos in peace.

https://mpv.io

SHOTCUT

A fairly advanced video editor, *Shotcut* is one of the few tools that supports editing 4K videos. The program supports many audio and video formats along with a variety of transitions and effects. It has a long list of impressive features and a collection of video tutorials to help orient first-time users.

https://shotcut.org

CURLEW

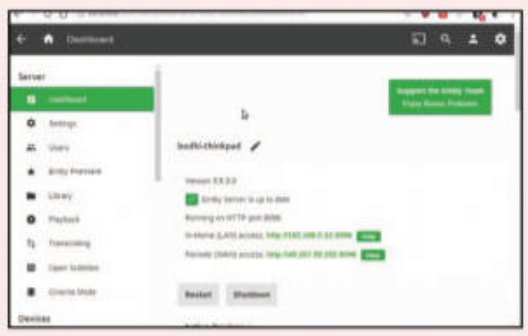
Powered by the *ffmpeg* library, *Curlew* is an easy-to-use media converter. It has the ability to crop and pad videos and convert only specified portions of files. Select from one of its preset settings for various devices and formats and you'll also be able to preview the files before the conversion.

https://curlew.sourceforge.io

EMBY

This is an all-inclusive media server that comes with plenty of features. *Emby* is a one-stop shop for accessing and viewing all your media on any device: from other computers and mobile platforms to Android TV, Chromecast, Roku, Xbox and more. Besides local folders, *Emby* enables you to add network shares, which is a great time-saver if you have media on different computers in your network. The *Emby* server is available as a pre-packaged binary and its website has installation instructions for all popular distros including Arch, CentOS, Fedora and Ubuntu. You can right-click items to reveal useful options such as downloading them, adding them to playlists and collections, converting them or editing their metadata. You can change the stream quality during playback and display full-screen. *Emby* also enables you to set up parental controls, and you can block items that have a higher rating than the maximum you've defined.

https://emby.media



Emby can also connect to TV tuner cards and DVR devices to stream and record live TV.

AIRSONIC (3S)

Based on the now closed-source project, the *Airsonic* audio streamer makes your music omnipresent. It does on-the-fly conversion and can stream files in almost any format to multiple players simultaneously.

https://airsonic.github.io

TUPITUBE

A set of tools to encourage kids to create 2D animations, *TupiTube* has an easy-to-use interface with just the right number of features and a host of textual and video documentation to get started.

www.maefloresta.com

PICARD

A nifty little tool, *Picard* will get your music collection back into shape. The program can sort your music library and fill in missing tags, rename oddly named files and identify incomplete albums.

https://picard.musicbrainz.org

TRAVERSO DAW

A powerful audio recording and editing platform, *Traverso* is a digital audio workstation tool that can do everything from creating simple recordings to editing multi-track audio.

https://traverso-daw.org

MIXXX

If you need to play DJ at an event, run through your music library with *mixxx* and use its scratchable waveform to loop beets, alter the tempo of tracks and change their pitch to create your own mix.

www.mixxx.org

Disk tools

Keep your hard drive in tip-top shape, recover lost data or delete it safely.

FOG

The constant barrage of repetitive tasks such as running checks, troubleshooting errors, swapping out dead hard disks, doing fresh installations over and over again can sap the energy out of any system admin, irrespective of the size of their network. Using *FOG* you can image and clone machines from the comforts of the admin HQ.

FOG is a complex piece of software and offers plenty of options, with various fields to describe the host images. It can also arrange them into groups for easier management. There are also several options to schedule the imaging process.

FOG can also be used to debug imaged computers, remotely wipe hosts and more. The server also handles regular admin tasks such as installing software and can even manage printers on the network. The *FOG* server is scalable and can manage large networks spread over multiple locations in the same building or on the other side of the planet. One of the most useful features of the *FOG* server, especially for admins of larger networks, is the multicast ability. Using this feature you can deploy multiple machines in one go. To supplement it on such large networks, you can have multiple *FOG* installations configured as storage servers that help take the load of the main *FOG* server when imaging computers.

https://fogproject.org

DUPLICATI

An easy-to-use back-up application, *Duplicati* enables you to fine-tune the list of locations you want to preserve by either defining filters or toggling one of the predefined options to exclude certain types of files. You can manage the tool via a browser-based interface. The program breaks down critical tasks into wizards and also exposes just the right number of features for the job in hand, while advanced options are just a pull-down menu away.

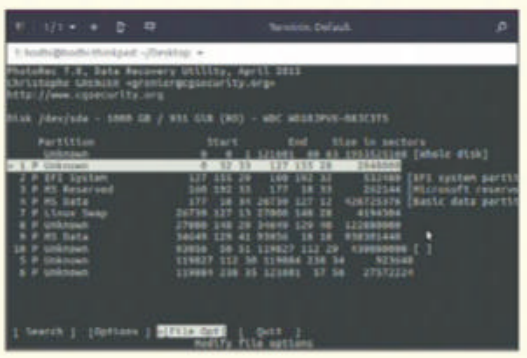
Files are encrypted with AES-256 and you can also use GPG before sending the backups to their destination. *Duplicati* also compresses all data before it's encrypted. It supports Zip and 7Z compression, and can skip compression of already compressed files such as MP3 and JPG.

www.duplicati.com

PHOTOREC

Photorec is a nifty little command-line based tool that can restore accidentally deleted files. When you delete a file, the file system just marks it as deleted, and makes the space the file occupies available to other files. *Photorec* works by recovering such files that are missing regular metadata such as a filename. Despite its name, the CLI utility can sniff files in various formats.

www.cgsecurity.org/wiki/PhotoRec



Photorec is part of almost every recovery distro and it ships along with TestDisk.

GPARTED

One of the best graphical utilities to manage partitions, you can use *Gparted* to create, delete, resize, move and copy partitions while preserving the data they house. The program works with all kinds of hard disks, SSDs, and RAID devices and supports all the filesystem in vogue.

https://gparted.org

SECURE-DELETE

The secure-delete package contains various utilities to securely delete files and wipe all traces of data in the free space on the disk. There's *srm* that make it impossible to remove any deleted files, *sfill* to wipe all data from the free space on the disk, and *sswap* to wipe data from the swap partition.

http://srm.sourceforge.net

CDEMU

The *CDemu* tool enables you to mount disc images in various formats including .ISO, .bin and .nrg. The *CLI* tool also has several graphical front-ends for Gnome and KDE desktops.

https://cdemu.sourceforge.io

DUC

If you're looking to free up some disk space, use *Duc* in the first instance to gain a better idea of the files occupying the space. The tool does a better job in inspecting your hard disk than the file manager.

https://duc.zevv.nl

DDRESCUE

Try image a failing drive with *ddrescue* before attempting recovery. The utility doesn't write zeros to bad sectors, and tries to fill in the gaps without wiping out the data already rescued.

www.gnu.org/software/ddrescue

PYDF

A replacement for *df*, *pydf* is a Python script that highlights the different types of filesystems. It has an large set of configurable parameters and various options to control its output.

https://pypi.org/project/pydf

SMARTCTL

Bundled as part of the *smartmontools* package, the *smartctl* utility controls and monitors the SMART system built into hard disks. Use the tool periodically to run self-assessment health checks on the disks.

www.smartmontools.org

File management

Control your data, where it can be accessed, and archive it, too.

EICIEL (45)

Everything in Linux is a file and the access control list (ACL) helps determine the access rights for each file. You can modify the permissions from the command line or use the graphical *Eiciel* utility that's in the official repositories of most distros.
https://rofi.roger-ferrer.org/eiciel

FSLINT

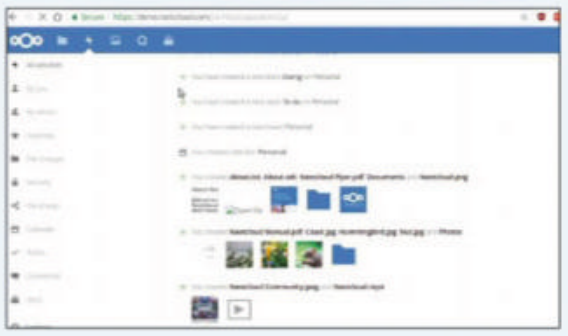
You can use the graphical *FSLint* tool for a comprehensive cleanup of your file system. It can remove duplicate files, temporary files as well as files that are otherwise difficult to locate and remove, such as files with invalid names, empty directories and bad IDs.
www.pixelbeat.org/fslint

NEXTCLOUD

Online storage services are a convenient option for accessing and sharing your data anywhere on the planet. With *Nextcloud* you can roll out a hosted storage server that gets you the convenience of an omnipresent storage service without shelling out wads of cash and your data to a third party.

Installing and rolling out *Nextcloud* is a fairly straightforward and well documented process. It has a feature-rich and intuitive web interface that shouldn't prove to be difficult to navigate even for first-time users, which is a feat in itself considering the sheer number of features it wraps underneath. The project scales well and can be used on a home network as well as an enterprise one. You can use its administration section to hook it up with other related network services such as a directory server to import users.

Sharing files with other users on the network or publicly via URLs is simple enough. The server ensures that changes made to shared files are synced to all users. It can also access files stored on a variety of cloud services such as Amazon, Google and Dropbox.
https://nextcloud.com



Nextcloud has an extensive plugins framework that enable you to use it for many deployments.

OPEN MEDIA VAULT

If you need more protection for your data than a simple back-up solution, then you need to convert an unused computer (or even a Raspberry Pi) into a dedicated network attached storage (NAS) device with the Debian-based *Open Media Vault* (*OMV*) server.

OMV is straightforward to roll out and simple to manage, thanks to its well-designed browser-based user interface, which makes it suitable for even non-technical users. It supports all the popular deployment mechanisms, including several levels of software RAID, and you can access the data it holds using all the popular network protocols such as SSH, SMB/CIFS, FTP and Rsync. The server also has an extensive permissions systems to control access to the shared volumes and folders. The server is modular and can be extended with a variety of official and third-party plugins. For instance you can turn your NAS into a torrent client to download data directly into the NAS storage or use it to stream stored music across the network.
www.openmediavault.org

SYNCTHING

The open source alternative to the popular but proprietary *BitTorrent Sync*, *Syncthing* can sync your files between computers over your LAN or across the web. The service uses a global discovery server to connect clients anywhere on the Internet.
https://syncthing.net

Use Syncthing's browser-based interface to add shared folders and all of your various devices.

CATFISH

A graphical alternative to the find and locate CLI utilities, *Catfish* is a versatile tool that has various options to help you prune the search results and find the file you were looking for.
www.twotoasts.de/index.php/catfish

PEAZIP

The graphical compression utility can read most archiving formats and boasts of some useful data security features such as the ability to create archives with encryption and two-factor authentication.
www.peazip.org

EXIFTOOL

A CLI utility for working with metadata in images, *exiftool* helps you manipulate the files based on their metadata. Users can also use it to strip all metadata before sharing images online.
www.sno.phy.queensu.ca/~phil/exiftool/

TKDIFF

A graphical front-end to the *diff* utility, you can use *TkDiff* to compare two files. The tool can connect with a range of source code management systems and has several other features.
https://tkdiff.sourceforge.io

RANGER

Ranger has a curses-based interface that works inside the console. The tool supports *Vi* keybindings, makes use of the rifle file launcher and is able to preview images and videos.
https://github.com/ranger/ranger

Communication

Chat securely or set up your own social network. In your face, Facebook!

JITSI

An all-in-one instant messaging app, *Jitsi* has an impressive set of audio and video-conferencing features. As a VoIP client, it enables you to make calls using the Session Initiation Protocol (SIP). It has all the features you'd expected from a softphone in that it can mute, put on hold, transfer and record calls. It can also make registrar-less SIP calls to other *Jitsi* users on the local network.

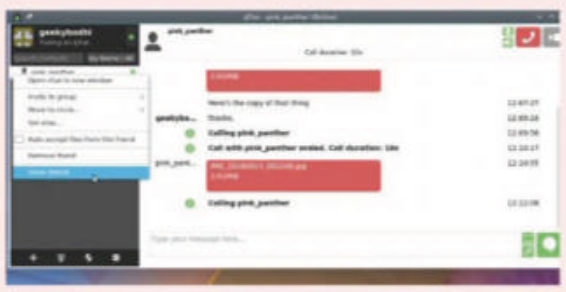
You can use *Jitsi* to make video calls to one user or to several on both SIP and XMPP networks, while using *Jitsi Videobridge*, you can host multi-user video calls. It's also a capable IM client with several unique features including the ability to stream and share the desktop of users at both ends of the call at the same time.

https://jitsi.org

TOX

If regular IM clients aren't secure enough for you, use the *Tox* IM protocol which encrypts all chats with the NaCl encryption library and instead of a central server, routes them over direct P2P connections between users. The IM uses *Tox* IDs, which are public keys of peers instead of a user account and also allow for greater anonymity.

https://tox.chat



There are several apps that can communicate using the Tox protocol and one of the most popular ones is qTox, which works on several platforms.

BEEBEEP

An IM client with a difference, *BeeBEEP* is designed to connect you directly with peers on your network. It'll automatically detect other *BeeBEEP* users on the network and provides all the features of a modern chat client and apart from encrypted text messages, can also share files.

http://beebeep.sourceforge.net

OPENFIRE

There are several XMPP-based IM servers available but *Openfire* is one of the easiest to manage. It implements many of the commonly used functions of the XMPP protocol. While you can use any XMPP-compatible IM client to chat through *Openfire*, it works best with its own *Spark* client. **www.igniterealtime.org/projects/openfire**

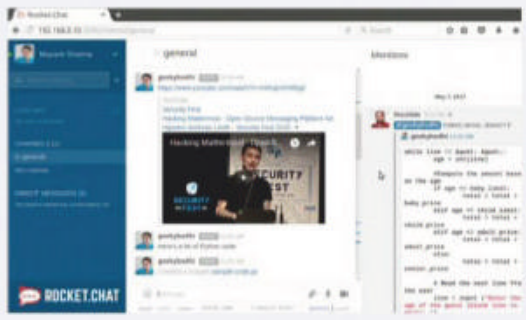
ROCKET.CHAT

Billing itself as the 'ultimate chat platform', *Rocket.Chat* is filled with useful features. In addition to essentials ones like threaded conversations and the ability to edit and delete messages, the collaborative communications server offers several interesting ones such as live chat, with which you can use *Rocket.Chat* to have conversations with visitors to your website.

You can roll out *Rocket.Chat* on a snap-supported distro with a single command, and it also support deployments over Docker and a wide range of PaaS services. It can also import your data from another service like *Slack* or *Hipchat* and with the SlackBridge plug-in can also mirrors messages received in a *Slack* channel or private group into *Rocket.Chat* in real-time.

Rocket.Chat also enables you to have a video conference with your team. You can also record and send voice messages to a public or a private conversation.

https://rocket.chat/community



Rocket.Chat enables you to preview links shared during a conversation including those from popular services like Twitter and YouTube.

SIGNAL

The *Signal* messaging app helps secure communications over the mobile network. The app is endorsed by top security experts and uses end-to-end encryption to secure all your text, audio and video calls.

https://signal.org

RIOT.IM

The Matrix protocol enables users registered with one service provider to seamlessly communicate with users of another. *Riot.im* is one of the best clients based on the federated Matrix protocol.

https://about.riot.im

RETROSHARE

A P2P communications and file sharing platform, *RetroShare* creates encrypted connections between friends. You can add friends by sharing your keys privately or you can exchange them via a chat server.

http://retroshare.net

SOGO

One of the best alternatives to MS Exchange, *SOGo* has all the necessary groupware functions with native desktop and mobile clients. Try the online demo before deploying it on your server.

https://sogo.nu

HUMHUB

With *HumHub* you can deploy your own social network on the internet network. It has all the standard social networking features you'd expect and the project's website hosts an online demo as well.

www.humhub.org

Productivity

Put some order into your life with these finance, web filtering and admin tools.

OSMO

A personal information manager (PIM) helps you keep track and organise all the information coming in from various sources. *Osmo* is a lightweight PIM that helps manage appointments, tasks, contacts and notes. The application has a straightforward and integrated interface with four tabs on the top for Calendar, Tasks, Contact and Notes. The program lacks any menus, but the action-linked buttons at the top change as you switch between the four components. The Calendar pane provides a simple functional calendar along with some related information such as the week number and the number of days to the end of the year. You can double-click a date to add notes or right-click a date to add a task.

The options for defining a task in *Osmo* are housed within a basic and an advanced tab. Use the basic tab to define the due date and time and assign the task a priority, while the advanced tab enables you to mark the task as recurrent and define relevant properties. You can store the address of your contact using the Contacts pane. It sports additional features including a search function, a birthday browser, as well as the ability to point out the address of a contact on Google Maps. The Notes pane makes it possible to jot down text using rich text editing functions and you can optionally encrypt all your notes using a password. **http://clayo.org/osmo**

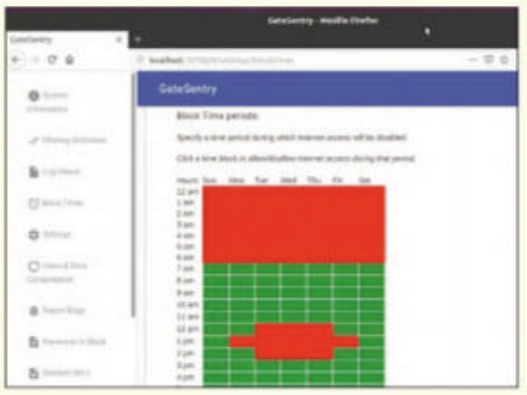
FOCUSWRITER

Scribble without distraction with the *FocusWriter* text editor, which runs full-screen and uses an auto-hide menu for a clean workspace. The editor supports the popular text formats and has basic support for ODT files as well.

The primary goal of *FocusWriter* is to enable you to focus on the writing, so while the tool might lack some of the dexterity of the mainstream text editors, it has several unique features that are more tuned towards this objective. You can set alarms to trigger after a certain period has elapsed, or at a particular time and also set yourself targets in *FocusWriter*. An interesting feature is Focused Text, which fades out everything except the section you're typing – a whole paragraph, a block of three lines, or just the current line. **https://gottcode.org/focuswriter**

GATESENTRY

A simple web filter, *GateSentry* works on Linux as well as the Raspberry Pi. It's easy to deploy and configure. *GateSentry* authenticates users and filters traffic by websites, content type as well as by time. The simple proxy server can display data consumption statistics and can also enables you to remotely disable a user's access. **http://gatesentryfilter.abdullahirfan.com**



GateSentry can also filter traffic over HTTPS once you install its certificate on your devices.

KIMAI V2

Kimai tracks work times and prints out a summary of your activities. It can be used for a single user or multiple ones and can manage multiple customers, projects. The program has a browser-based dashboard that you can fiddle around with using the demo installation on the project's homepage. **https://v2.kimai.org**

SIMPLESCREEN RECORDER

Contrary to its name, *SSR* is flush with features and gives its users a good amount of control over the screencast. The tool can record the entire screen and also enables you to select and record windows and regions on the desktop. **www.maartenbaert.be/simplescreenrecorder**

TURTL

A note-taking tool with a focus on privacy, *Turtl* has an impressive list of features. It can create different types of notes and makes it possible for you to attach images and other files to the notes that you've created. **https://turtlapp.com**

DICTION

A CLI utility, *Diction* helps flags words that are commonly misspelt. It can create better copies by flagging words that are commonly associated with beginner's mistakes and to even suggest better wording. **www.gnu.org/software/diction**

TASK COACH

A simple to-do manager, *Task Coach* simplifies the process of defining and following up on tasks. You can also use the tool to assign a priority to tasks along with a completion date and an optional reminder. **www.taskcoach.org**

EQONOMIZE

Eqonomize is a personal expense manager with a clean interface and all the features you need to help organise your finances. The program can track single or multiple accounts and fix budgets. **https://eqonomize.github.io**

STRETCHLY

A simple app that pops in to remind you to take a break. It has a couple of preset break intervals and the break windows share ideas to help you relax. You can control the tool from its tray icon. **https://hovancik.net/stretchly**

System Administration

Find out exactly what's going on under the hood of your Linux box.

GLANCES

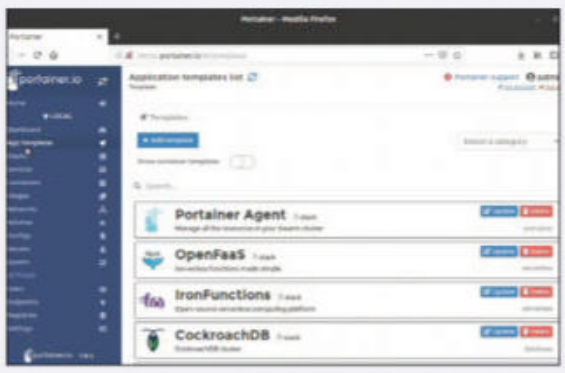
Glances is a Python script that smartly displays a lot of information about your current session inside a standard terminal window. You can use the tool to monitor remote machines and use the built-in web interface to monitor machines using a web browser.
https://nicolargo.github.io/glances

FIREJAIL

Firejail enhances security by isolating programs and processes inside limited sandboxed environments. By locking away the web browser or any tool that can be compromised, *firejail* prevents a compromised program from accessing critical areas of the filesystem.
https://firejail.wordpress.com

PORTAINER

Using Docker via the terminal isn't all that cumbersome and the tool is well documented. To make your life easier however, you can use *Portrainer*, which is an open source, web-based graphical front-end that supports all the features exposed by the Docker API. You can use it to manage containers, images, networks, and volumes and it can manage a standalone Docker environment, or a Docker Swarm.



You can use Portainer to pull images from Docker Hub or a private registry.

From its dashboard you can create containers and view and manage all available ones. You can obtain real-time stats such as CPU and memory usage and various other details about the running containers and even access a container's console, all from within the browser window. *Portainer* is available as a Docker container so you can install it with a single command. To create containers you can use one of the dozens of predefined templates. The tool can be used by multiple users and has useful user-management functions. You can for example use it to define the levels of access any other users have to *Portainer*, and the aspects of Docker they can manage from within *Portainer*.
https://portainer.io

BOOTCHART

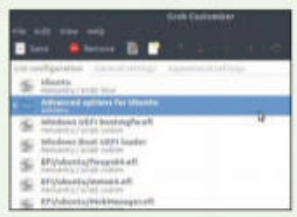
One of the major causes of longer boot times is that your system starts unnecessary tools and services during startup. But before you axe them, it's best to get a picture of what's happening while your distro boots up.

Bootchart is a simple utility that enables you to profile your Linux boot process and help measure the loading times of different services. It's now merged with Systemd. Fire up a terminal and enter `systemd-analyze time` to know the breakup of the boot duration. Similarly, `systemd-analyze blame` will list all running units ordered by the time they took to start.

To generate an image of the boot process, type `systemd-analyze plot > boot.svg`. From this image you can find all the active processes and can remove any you don't need. For example, if you print occasionally, you can disable CUPS from starting at the boot time. Furthermore, the image also helps you spot processes that take control of all resources and force the other processes to wait, slowing boot up.
www.bootchart.org

GRUB CUSTOMIZER

There are several reasons you might want to change the default boot loader behaviour and the *Grub Customizer* tool will help you do just that. The graphical tool can modify all aspects of the boot process and can even tweak how the Grub screen looks. **https://launchpad.net/grub-customizer**



Grub Customizer also helps restore order when you've accidentally wiped clean the MBR.

REMMINA

A very usable remote desktop client, *Remmina* supports a wide range of protocols. It performs well and offers useful features such as the flexibility to change the quality settings of the connection on the fly.
https://remmina.org

INXI

A CLI tool that lists info about the hardware in your computer. With *inxi* you can collate details about the kit, including vendor details, device driver configuration and more.
https://github.com/smxi/inxi

POWERTOP

Intel developed the *PowerTOP* utility to track down apps that eat up your laptop's battery. The CLI utility offers suggestions to tweak power-guzzling tools to squeeze more juice from your laptop's battery.
https://01.org/powertop

STACER

Stacer is a system monitor that helps you track different aspects of your installation. It can also show alert messages when values for a parameter exceed a certain threshold.
https://oguzhaninan.github.io/Stacer-Web

LOGWATCH

Logwatch parses, analyses and filters logs and then generates daily reports on your system's log activity. The utility also enables you to control the level of the report's verbosity.
https://sourceforge.net/projects/logwatch

Internet tools

Make your online time more efficient with this collection of utilities.

UGET

By default, the lightweight *uGet* download manager relies on *curl*, but if you install the aria2 package, it can take on some more features, such as the ability to download torrents.

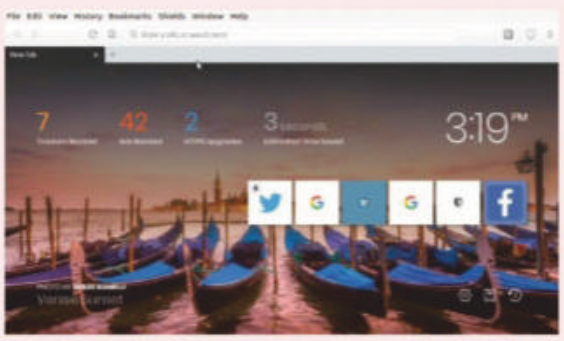
UGet is an all-round downloader that has all the features you'd expect from a download manager. It features a download queue, can pause and resume downloads and also accelerates downloads by grabbing files from multiple parallel streams. Furthermore, batch downloads are one of its specialities. You can use the tool to prioritise the download queue and even regulate the speed of the downloads individually. *UGet* can also shut down and hibernate your computer once it's finished downloading all the files.

https://ugetdm.com

BRAVE

While popular browsers can be equipped to thwart information leaks, *Brave* ships with privacy-strengthening features. The Brave project promises two improvements over the competition: privacy and speed. One of them is achieved by blocking ads and trackers, which has the pleasant side effect of improving the browsing speed.

https://brave.com



Since blocking ads may rob sites of revenue, Brave has some unique compensating tricks.

WALLABAG

A read-it-later tool, *wallabag* began when Google Reader was shuttered in 2013. You can install it on your own server or use it via its own service that costs about £8/year. *Wallabag* has impressive features including the ability to save articles, filter them by reading time, and more.

www.wallabag.org

TOR BUNDLE

Tor enables users to browse the web anonymously. Its goal is to prevent people from tracking you online. The *Tor Browser Bundle* includes everything you need to connect to the Tor network of relays including a customised version of *Firefox* known as the *Tor Browser*.

www.torproject.org/projects/torbrowser.html.en

RAMBOX

Most desktop distros ship with a default messaging tool that enables you to sign into multiple online messaging services. These tools can sign into only a handful of the popular services.

In contrast, *Rambox* can plug into over 100 online messaging and email services. Every added service resides within its own tab, from where you browse the message history, write messages to your contacts, or use the other means of communication supported by the service. The tool will also display notifications. *Rambox* can also sync configurations if you use it on multiple computers. The program respects your privacy. Instead of storing your personal data, *Rambox* uses the **partition:persist** attribute of the <webview> tag to create a persistent connection with the signed-in service. You can also set a master password and ask the tool to lock itself after a predefined period of inactivity.

http://rambox.pro

The tool is available as a Snap and AppImage binary that you can use without installing.

MAGIC WORMHOLE

A Python script that creates single-use encrypted channels to ferry files between computers across the Internet identified via pronounceable code.

https://github.com/warner/magic-wormhole

UBLOCK ORIGIN

The *uBlock Origin* browser extension is an ad-blocker that also blocks tracking servers, malware domains, and more. It's available in the app stores of all mainstream web browsers. **https://github.com/gorhill/uBlock/**

QUITERSS

QuiteRSS has all the features that you'd expect from a news reader. It can pull in RSS and Atom feeds, apply labels to each item and offers plenty of customisation options to boot.

https://quiterss.org

VOCAL

A feature-rich podcast grabber, *Vocal* can subscribe to and stream podcasts on-the-fly or download them for offline listening. It integrates with popular desktops and has smart library management functions.

https://vocalproject.net

UFTPD

uftpd ships with defaults that work for most users. It's designed for home users and developers who need a simple FTP server, but aren't too particular about being secure.

http://troglobit.com/projects/uftpd

Security tools

Ensure ne'er-do-wells don't get their hands on your system and files.

ZULUCRYPT

While you can control access to the data on your computer using user accounts and file permissions, they aren't enough to prevent a determined intruder from gaining access to your private files. The only reliable mechanism to keep your personal data to yourself is to encrypt it. Sure, working with encrypted data is an involved process, but it'll go a long way in reinforcing your security and insulating your data.

zuluCrypt is a graphical encryption tool that has an intuitive, easy-to-follow interface. Using the program you can create an encrypted disk within a file, a partition and even USB disks. It can also encrypt individual files with GPG. *zuluCrypt* can 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. With block device encryption, the user creates the file system on the block device, and the encryption layer transparently encrypts the data before writing it to the actual lower block device. While encrypted, the storage areas just appears like a large blob of random data and doesn't even reveal its directory structure. All these functions and more and accessed via the excellent user interface. **http://mhogomchungu.github.io/zuluCrypt**

GUFW

Gufw is the graphical front end for *UFW*, the uncomplicated firewall, which is one of the easiest front-end for *iptables*. The tool has a simple interface and includes predefined profiles to regulate incoming and outgoing traffic. You can alter the incoming and outgoing policies of the profiles as per your requirements. You can also use *Gufw* to define specific rules for allowing traffic for individual apps and services. **http://gufw.org**

Switch to the Report tab to get the live network traffic report.

BITWARDEN

The *BitWarden* password manager does end-to-end encryption and uses AES 256-bit as well as PBKDF2 to secure your data. It goes through this trouble because unlike its peers it stores your encrypted passwords in a remote Microsoft Azure cloud. The app can also import passwords from over 24 sources. **https://bitwarden.com**

OSQUERY

You can keep an eye on your network using the *osquery* CLI tool that enables you to query your devices just like a database. You can use the tool to check logged in users, keep an eye on the firewall, identify outdated kernels and keep an eye on the loaded kernel modules and the running processes. **https://osquery.io**

STEGOSUITE

Steganography is the art of concealing data inside another seemingly harmless message or image. The most common mechanism for implementing it is by replacing unused data in regular computer files with bits of information that aren't visible when viewing the original piece of data. Steganography is mostly used to complement encryption.

Stegosuite is a graphical tool that can hide text messages as well as any type of file inside an image. Fire up the tool, select an image, point to the files you want to hide inside it and enter a secret key to burn it in the image. To get the secret files from the image load it once again and use the Extract button along with the secret key to grab the hidden files and message from the image. **https://stegosuite.org**

MTR

MTR is a simple CLI network diagnostic tool that combines the functionality of the *traceroute* and *ping* utilities. So along with the path of the packet it displays a wealth of other relevant information. **www.bitwizard.nl/mtr**

AIDE

You can use *AIDE* to help spot intrusions by checking the integrity of the files by comparing their properties such as permissions against a baseline database created on the initial run. **http://aide.sourceforge.net**

NIKTO2

A CLI web server scanner that hunts for potential problems like server misconfigurations and outdated versions and version-specific issues. It can also perform automated tests against security vulnerabilities. **https://cirt.net/Nikto2**

JOHN THE RIPPER

A password cracker, *John the Ripper* is used for exposing weak Unix passwords. It's a CLI tool but also has a graphical interface called Johnny that exposes its various command line options. **www.openwall.com/john**

MALTRAIL

MalTrail is a malicious-traffic detection system that uses publicly accessible blacklists, custom user-defined lists and more to detect and log any malicious traffic. **https://github.com/stamparm/maltrail**

Developer tools

Make your coding life easier with this collection of capable utilities.

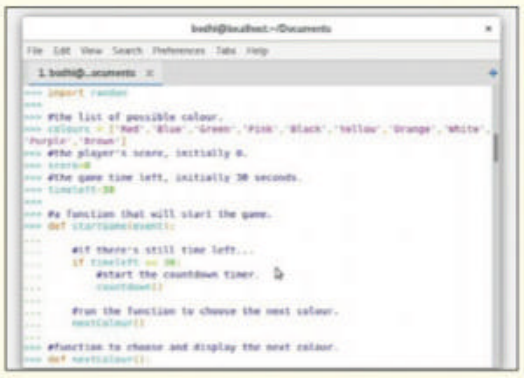
ATOM

Atom describes itself as a hackable text editor. Developed by GitHub, the tool has a built-in package manager that enables users to search and install plugins from within it. About 80 plugins ship with the utility by default. One of its more interesting features is the find and replace function that can also modify text across multiple files as you type.

Atom can also be used as an IDE and one of its highlights is the smart autocomplete feature. There's also a bracket matcher feature that highlights the line-number of the closing bracket corresponding to the one under your cursor. You can also define custom key bindings and add more functionality with packages for items such as minimaps and syntax-specific snippet libraries. **https://atom.io**.

BPYTHON

A Python shell that provides modern IDE-like features inside a terminal window. You can start using it with the default settings and then customise it by editing its config file. The shell does code completion and will also display a list of expected parameters. It highlights syntax as you type and the Rewind feature checks the entire code, which is kept in memory.
https://bpython-interpreter.org



Bpython enables you to save a session to a file, or even send it to pastebin.

ETHERPAD

Collaboration is key to any project. The *Etherpad* text editor enables users to collaborate on text in real-time. It runs inside the web browser and enables participants to interact via a text-based chat. It's full of features and you can take it for a spin on its website.
http://etherpad.org

VIRT-MANAGER

The *Virtual Machine Manager* is the open source graphical frontend for creating KVM-based VMs. It uses the *qemu-kvm* hypervisor, which is a version of the *qemu* machine emulator, and includes a *VNC* and SPICE client that displays a graphical console to the running VM. **https://virt-manager.org**

TURNKEY APPLIANCES

Installing network accessible software or web applications on a server can be quite a task because they require a lot of infrastructure software, which might take you hours to put together. From database servers to basic libraries, you'll have to spend quite a while to assemble a fully functional web server before you can deploy an application on the network.

Furthermore, if you plan to serve users from outside the network, you need to thoroughly examine your web server for any security leaks. This is a major undertaking in itself.

This is where *Turnkey Linux* shines. Using its virtual appliances you can deploy a new server app in no time. Put simply, a *TurnKey* virtual appliance is a self-contained system that packs in a fully functional instance of a web app with just enough components of an operating system to power that tool. The system works straight out of the box and can be deployed on either bare metal or on top of virtual hardware.

Turnkey Linux appliances are available in several formats depending on the hardware you want to deploy them on, from bare metal to OpenStack clouds. Importantly though, once they're up and running, irrespective of the platform, they all give you the same interface to administer and manage the web app.
www.turnkeylinux.org

OCELOT GUI

The database client can connect and interact with a MySQL or MariaDB server. You can use to create queries and it can highlight syntax and can fetch and display results from the **database.**
https://github.com/ocelot-inc/ocelotgui

DOXYGEN

Use this CLI tool to generate HTML documentation from the comments in your code. The tool supports many programming languages and can cross-reference the documentation with the code for referrals.
www.doxygen.nl

INFER

If you work with Java, C, C++ or Objective C, use *Infer* to weed out bugs The tool includes various analysis and its most interesting feature is that it can work across several procedures across various files.
https://fbinfer.com

MARK TEXT

The markdown editor supports the GitHub markdown spec and the CommonMark spec. It has a live preview, several editing modes and inline Math support and can export documents.
https://marktext.github.io/website

MANTA

When you've completed a project, use *Manta* to generate invoices and receipts. The invoicing app has a clean interface and several templates that you can customise as per your requirements.
https://github.com/hql287/Manta LXF

INTERVIEW Jorge Salamero

Jonni Bidwell escapes **LXF** *Towers*, his container as it were, to discuss container security with Sysdig's **Jorge Salamero**.

CONTAINER INSIGHTS

Jorge Salamero **INTERVIEW**

Special thanks to PR extraordinaire Stephanie Crisp who not only took some photos, but also achieved the not insignificant task of getting Jonni to be in the right places at the right times.

Jorge Salamero Sanz is technical marketing manager for Sysdig where he works on monitoring and security for containers. We caught up with him at the O'Reilly Velocity conference in London to chat about how greybeard-style monitoring doesn't help in the magickal world of DevOps, microservices and CI/CD – where containers flit in and out of existence like so many quantum particles. Besides computers, he's and avid hiker and skier, and most recently has gotten into motorsports – maintaining his own vehicle and driving it round the track at terrifying velocity. Ah well, that's safer than curl | sudo bash we suppose.

Linux Format: How did you get into computer science and open source?

Jorge Salamero: I guess it was a natural move, my parents bought a laptop and before the day was done it was already broken, so I had to figure out how to fix it. I guess a lot of people start off this way. I've been involved with open source for many years, probably since university. I worked with Debian, I created an open source startup in Spain, then I worked for a monitoring company here in London. And I've been at Sysdig for the last two years, so it's been nice to continue that monitoring experience. Sysdig have a number of open source projects that form the foundation of many of their commercial products. Part of my responsibility is to educate people about DevOps, monitoring, and especially anything to do with containers.

LXF: I've met Berto from Igalia at a couple of conferences now. Igalia's story is pretty inspiring – friends fresh out of college setting up shop – but what I was going to say was that Debian seems to be very big in Spain, that's how Berto started out with Linux and he's still a Debian maintainer besides doing his day job. Is there a more prominent open source culture in Spain or some other thing that leads people Debian-wards?

JS: Historically, there has been a strong community of Spanish Debian developers, and South American, too, in fact. I could probably name about ten [reels off names that confound our monolinguistic ears] – some of them are still active, some have moved on.

JORGE SALAMERO...

"One of the reasons people are so excited about containers is that they allow people to move development into production in a very easy way"

LXF: Your bio says that, besides all the fancy monitoring stuff that I'll get to later, you also monitor Raspberry Pi clusters and things. Is that just for fun and games?

JS: Yeah that's just for fun. I'm a big fan of IoT and metrics so I spent some time writing my own firmwares for chips like Arduino, doing stuff with sensors and doing various projects so the family can monitor things at home, too.

LXF: Take us, if you will, on a journey through the monitoring school jungle. I've done a bit of stuff with *Munin* and *Monit*, but beyond that it's all unknown.

JS: In the past I've worked with traditional monitoring tools like *Nagios*, but in the last two years my main focus has been *Prometheus* and various open source tools that connect to that. My main focus, through my work, has been how to leverage system calls for monitoring rather than traditional approaches.

LXF: I feel like I'm finally starting to understand the basics of containers, I've moved a lot of the (admittedly very basic stuff) our server does into Docker and I feel like its a much better arrangement. But there's a lot of stuff that's hard to get your head around – you can't just log into your Apache container, edit a config file and restart the service, because there's no shell, no editor and no configuration manager. And don't even get me started on volumes. So I'm guessing monitoring these things presents all kinds of new challenges.

JS: One of the reasons people are so excited about containers is that they allow people to move development into production in a very easy way. You package up your application and it runs in a nice isolated fashion. The traditional approach, which was "Okay, I'll have an agent check my software and return some metrics",

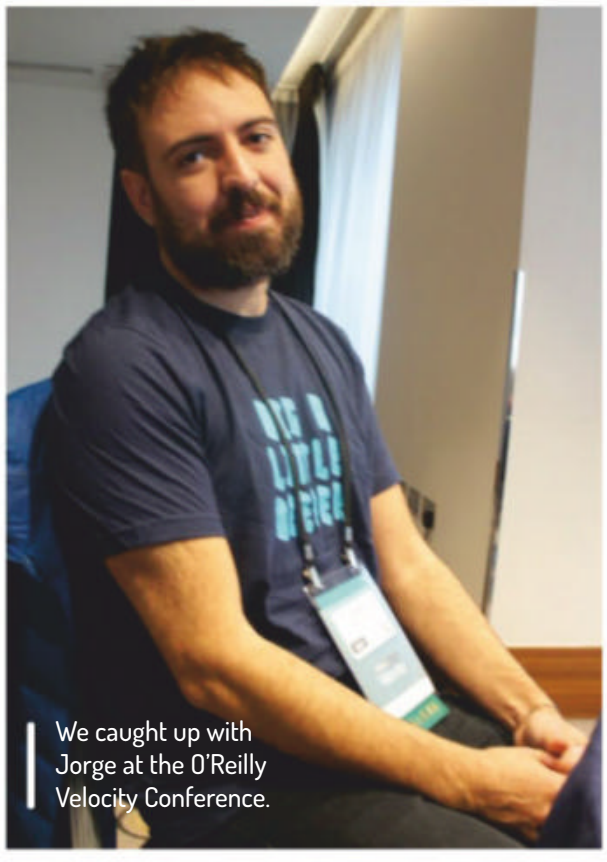
doesn't work – if you put an agent inside the container, you break Docker principles. Google created a project called cAdvisor which uses the Docker API to retrieve some metrics, but only a very limited set.

Our approach has been that containers share the kernel with the host, so what if we could get these metrics from kernel instrumentation – system calls? We looked at traditional tracing tools, starting with *Strace* then moving on to other things like *Perf* and *BCC*. Those were really interesting for doing very low-level tracing, but they didn't provide a very easy learning curve. Basically, they require a very complex syntax or hand-coding filters, which is totally out of scope for someone who just wants some metrics.

The founder of the company, Loris Degioanni, created *Sysdig* as a tool to do system call tracing in a much simpler way, taking familiar concepts from network tracing tools like *Wireshark*. In fact, people say *Sysdig* is like a *Wireshark* for your whole system, as opposed to just network traffic.

LXF: Latterly, we're hearing a lot about eBPF, but it's yet another thing I don't understand. Can you help?

JS: The technology actually goes all the way back to 1992 when they implemented Berkeley Packet Filtering (BPF) in BSD. The idea was when people have to filter network traffic, rather than bringing all that traffic into userspace, why not have a framework in the kernel space? Then you wouldn't have to worry about expensive context switches. That's how BPF was born. It took some years to move it into Linux, and it really didn't change for quite some time. That's why tools like *DTrace* in Solaris or *Perf* in Linux, were created.



We caught up with Jorge at the O'Reilly Velocity Conference.

Around 2013 Alexei Starovoitov decided to extend BPF into something that he imaginatively titled Extended BPF– eBPF. Basically, it's like a virtual machine with an instruction set where you compile your code – it can be C, Python, anything really – into bytecode that's executed inside the kernel by eBPF. eBPF had way more features, it was a more complex stateful machine so you could do more advanced filtering. But, probably, the main change was extending beyond just networking, so now you could filter on any system call – not just network packets. That really opened up functionality, so eBPF started to be used outside of just networking – for monitoring, for system call tracing and for security as well. My talk covers these three areas, as well as the open source tools for working with eBPF.

LXF: What about forensics, how do you investigate what happens when a container suddenly falls over without warning, or starts acting suspiciously?

JS: That's actually very, very interesting. One of the challenges with containers is that they're essentially black boxes. They're orchestrated by tools like Kubernetes (*see feature* **LXF245**), Mesos, or any number of other tools. Since they're typically used for applications built on microservices, so you end up with a very dynamic infrastructure – services scale up on demand to a large number of instances, then when traffic dies down they scale back down, possibly to one or even zero instances. So containers are moving across different nodes and they are spawned or destroyed all the time.

Traditionally when you have a security incident, you would SSH into the machine and start checking out logs. But there is a security incident on an application running inside a container you cannot do that – the container is gone. So how do we do forensics? Again, we try not to reinvent the wheel, so we considered what we did in the past when we had to inspect a network protocol. You'd use Wireshark to take a packet capture into a PCAP file, then you'd do a very detailed, in-depth analysis of that. So we do the same with containers – we record all the system calls into a file, then even if the container is long gone you can still open that file and reproduce every single step. If you're lucky, you might see how an attacker broke in, what they did, what access they had to privileged data.

We became so passionate about this that we created an open source UI, it's called *Sysdig Inspect*, to navigate through these system calls in a very easy fashion. So a security analyst doesn't need to understand low-level system calls to make sense of those capture files. They can be navigated in terms of files, network connections or container events – things that are much more accessible.

LXF: One of Sysdig's projects I hear lots, but understand little about, is Falco. Can you tell me a little bit about that?

JS: Falco is pretty buzzing just now, much like eBPF. Falco was born as a weekend experiment inside Sysdig. We created a technology to enable observability in containers so then we thought about all the things that we could do with this information. When you do tracing basically you look at the system calls, and you generate metrics – numbers – so you can draw graphs and charts with that data. So we thought "What if we work the other way around?" We can define a profile and check that the container behaves the same way as was defined in that profile. There are tools like Seccomp, which actually uses eBPF, but creating those security profiles requires coding.

Sysdig has a very easy to use filtering language, and that same language can be used by Falco. Basically using YAML, any human can create Falco profiles that vaguely resemble natural language and use these to monitor a container for any kind of misbehaviour or anything at all out of the ordinary. Falco is then much like a surveillance camera for containers. By itself it follows Unix KISS design principles, it just monitors. But we have created integrations so we can send events to ElasticSearch, or we can use other cloud-native tools like Kubeless or NATS to collect those events, those notifications, from Falco and react. And to react in an

JORGE SALAMERO...

"Basically using YAML, any human can create Falco profiles that vaguely resemble natural language and use these to monitor a container for any kind of misbehaviour"

Johni and Jorge talk containers and more.

automated way, such as with a security playbook. We created an integration that can basically respond within the Kubernetes layer to any security incident. So if we detect a threat, we can implement a Kubernetes network policy to isolate that container, or we might decide to just take a capture and kill it. Essentially, we have created a complete, open source stack for container runtime security.

LXF: Tell us more about the relationship between Sysdig's commercial products and their open source offerings.

JS: On the security field, Falco is just for runtime security and it's open source. Our commercial product uses the very same Falco, but on top of that there's a management UI and some things like compliance, vulnerability management with image scanning. So the commercial one is a more an end-to-end solution for dealing with container security, with Falco being one piece of that solution

On the other hand, Falco could be one part of a DIY security stack. Actually, we partnered with Anchore who have an open source scanning tool, and we created some integrations and blog posts about how to do your own open source version of a complete container security stack (**https://anchore.com/blog/anchore-and-falco**). I guess we could've called it *Open-chore* or something like that.

LXF: I prefer software titles that reference Greek mythology or bird of prey. Anyhoo, I always end up asking this, and it's probably because microservices and Kubernetes and kata containers and all these things that I don't understand have, seemingly overnight, taken over. But do you think for all the problems these things solve they're going to give rise to new, unforeseen threats?

JS: Yes. I mean, we're going through the biggest revolution in infrastructure since virtualisation. *VMware* radically changed how people ran software in their datacentres, that was a little over ten years ago. Containers take this one step further, by making deployment faster and generally facilitating DevOps workflow. Containers have been adopted by sectors that usually are much slower to adopt new technologies, like big enterprises. We could say that this is because their value is immediately obvious, whereas with other technologies that value wasn't always so clear. For example, everyone knew configuration management was important and everyone had their own way of doing it, but it wasn't clear which solution to go for, and what would be the benefits and cost savings of one approach over another. But with containers you can say "I can deploy ten times faster and my hosting invoice is a third of what it used to be". CEOs like to hear that sort of thing.

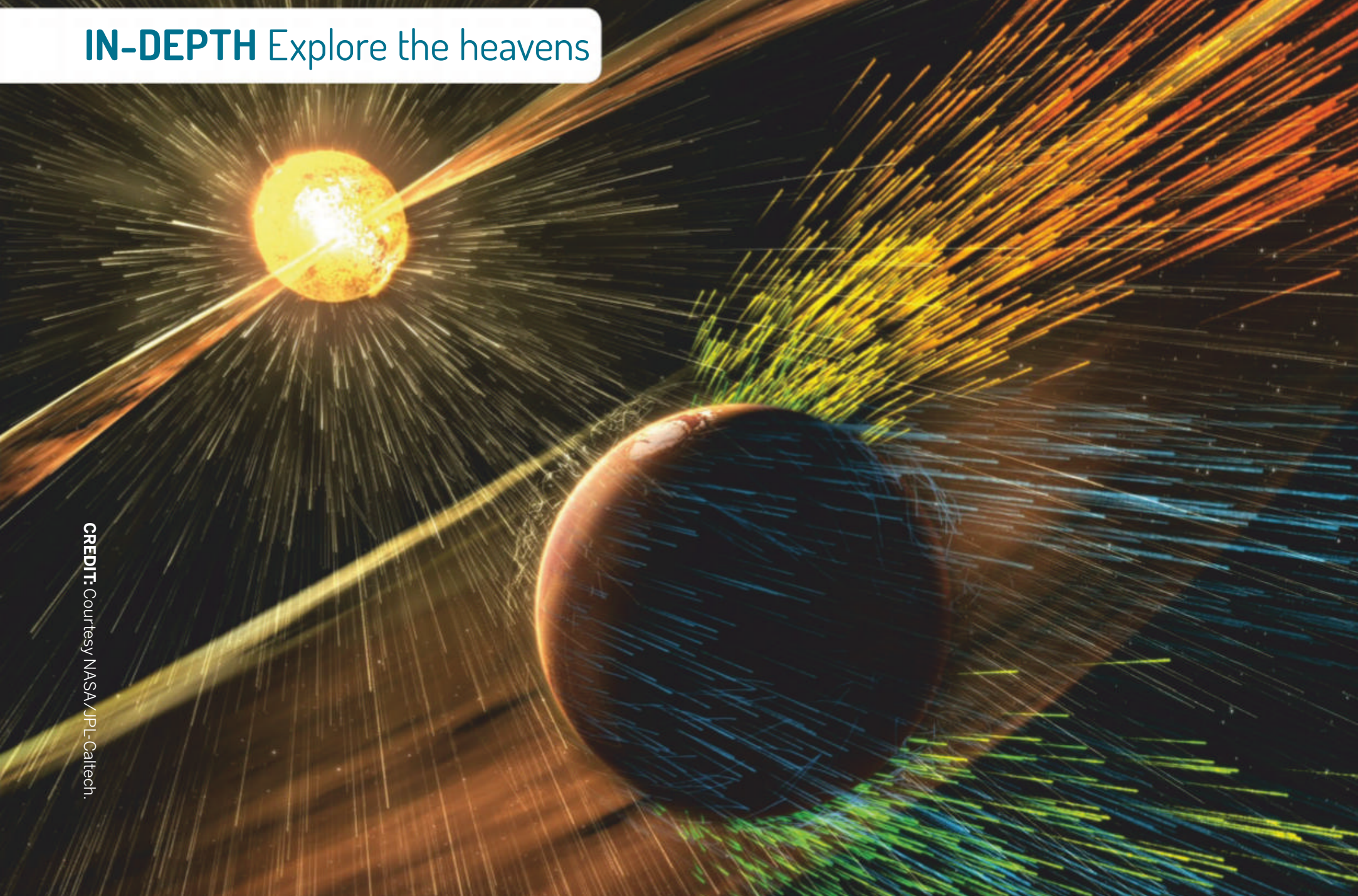
LXF: Continuing the theme of things I keep hearing about, but of which I possess not the slightest grain of understanding: Continuous Integration and Continuous Delivery. How do you hard-boil security into these very dynamic pipelines?

JS: A few months back, at Continuous Lifecycle here in London actually, I gave a talk about just this (see **www.youtube.com/watch?v=yoznpXrsFAU**). We hear about CI/CD a lot, so I thought I'd extend this and came up with CI/CD/CS, the last bit stands for Continuous Security.

Once you have technologies like Falco that provide container observability, those tools can't really be handled manually – they have to become a part of the container, or even the software, lifecycle. So we came up with the idea of defining those runtime security policies as code. Then you can handle them in a Git repository and your CI server, at the same time it builds the container, can apply those security policies into your production environment. Basically every application is shipped with a container and then a set of runtime security rules. Runtime security is important – development happens very fast and then that software runs for a very long time. Many people started implementing security by scanning container images, but that only works for known vulnerabilities. It does nothing for configuration errors or yet-to-be-discovered vulnerabilities. There are also vulnerabilities that are specific to certain pieces of software, and these might never be part of any public registry. You also need to prepend problems there in production, so there's a lot more to it. It was important to create those policies to work at runtime but at the same time to do so in a consistent way. This then becomes a part of the shipping cycle, the whole development process from the coder pushing code into a repository, then on to production and testing. So this is an end-to-end security approach.

LXF: I started this job right when Docker came to the fore. Back then people more or less assumed you got some security for free, just because containers exist in isolation. That's only true for theoretical examples, and when you need containers to talk to each other the easiest way is often just to expose everything. It seems like a lot of people made this mistake and by doing so made some spectacularly insecure things, and I worry about how perceptions of technology will lead to further such examples.

JS: Yes, you have changed your entire infrastructure, so all your previous security/monitoring/backup process need to be updated to accommodate that. Just like moving from BSD to Linux, or, I dunno, Linux to Windows, you have to change how you do things if you don't want nasty surprises. LXF



CREDIT: Courtesy NASA/JPL-Caltech.

EXPLORE THE HEAVENS

It's time to discover and explore the night sky with a little open-source software, your own two eyes and the help of **Mike Bedford**

Space is big. Really big. You just won't believe how vastly hugely mindbogglingly big it is. I mean you may think it's a long way down the road to the chemist's, but that's just peanuts to space." Okay, so these words from Douglas Adams' spoof sci-fi novel *The Hitchhikers' Guide to the Galaxy* might have been intended as a bit of silliness, but the sentiment is no less true.

Based on the latest data from the Hubble Space Telescope, astronomers now estimate that there are 700 sextillion (that's 700 thousand billion billion) stars in the visible universe. Trying to get your head around such a vast number of objects – or even just those that can be seen from Earth with the naked eye, still a huge number – is, well, mindboggling. However, we're here to help.

What's more, since this issue of *Linux Format* will reach you during the depths of winter (*apologies Southern hemisphere chums–Ed*), there couldn't be a better time of year to make the most of those long dark nights by discovering the wonders of the heavens.

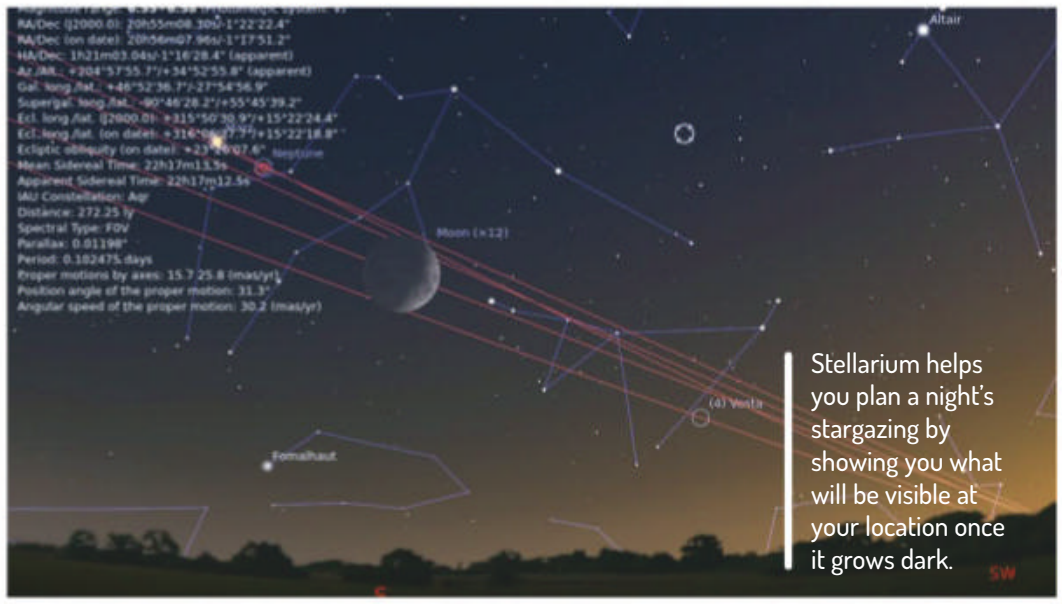
Our aim here is to help those who are fascinated by the sky at night, but know little about those countless pin pricks of light on display, to take their first steps in practical astronomy. This is a three- or four-stage process. First, we'll see how open source software can help you plan your star-gazing by showing you what will be visible at any time and where to look. Then, armed with this information, we'll provide you with some practical guidance how to observe some of these astronomical objects with nothing more than your naked eyes and a pair of binoculars.

Some objects will never look like more than tiny points of light so, next, we'll use software to shed more light on these alien worlds, even showing you how they'd look if you were to travel there. And finally, for those of you for whom this experience proves an eye-opener, we'll point you in the right direction should you consider the next step of investing in an astronomical telescope.

The software we've chosen to help you plan a night of practical astronomy is called *Stellarium*. It's highly regarded, runs on most major operating systems, and is an open source project.

Probably the best description of it is a software-based planetarium. In other words, it generates an image of the sky as it would appear from any point on the Earth's surface, at any time – past, present or future. Commonly, you'd use it to show the night sky as it will appear at your planned observation location at the current time. However, there'll be times when you'll want to see the sky as it'll look a short time into the future, as an aid to planning your observations.

As with the Sun and Moon, other astronomical objects rise and set at particular times. For example, those planets that are close to the Sun, such as Venus and Mercury, are only visible for a short time after sunset and for a short period before sunrise. In addition, some planets aren't visible at all at certain times, sometimes for several weeks at a time. Looking into the future with *Stellarium* will, therefore, help you decide when to venture outside.



Stellarium helps you plan a night's stargazing by showing you what will be visible at your location once it grows dark.

Get the most from Stellarium

We'll leave you to get to grips with using *Stellarium* yourself, but we will give a bit of guidance on the user interface and indicate some of its useful features. *Stellarium* is unusual in not having a conventional menu bar. Instead, it's driven mainly via two toolbars, which are normally hidden. They are displayed by moving the pointer to the bottom or left of the screen close to the bottom-left corner. Be sure to select a town or city close to your location and the time you're interested in, using the appropriate options in the left-hand toolbar.

Other than that, though, we suggest you try out the various options yourself to get a feel for what's on offer. You'll find, for example, that you can select which types of objects to display (stars, planets, moons or asteroids) and you can also filter the display labels according to the brightness of the objects. With this latter option, you can find a suitable compromise between not being able to identify objects you're interested in, and seeing so many labels that the screen is utterly confusing. Note also that, if you click an object, information on that object will appear in the top-left of the screen.

Another option is to display the lines that define constellations, to show their names, and even to add an image. So, for example, having displayed the lines and the name of the constellation Orion, you can superimpose the artwork of the hunter. Of course, constellations have no astronomical reality. So, for example, the seven stars that constitute the Plough or the Big Dipper, which form part of the constellation Ursu Major, appear to be relatively close together, but this is an illusion as they vary from 58 to 124 light years away (that's 500 to 1,200 trillion km). Often, the stars in a constellation appear to be of similar brightness because the more distant starts are larger than the closer ones.

Despite the fact that constellations are largely mythological in nature, they can be useful, nevertheless. If you're able to recognise some of the main constellations, this can be a big help in spotting astronomical objects that *Stellarium* shows as appearing close to a particular constellation.

As a final comment on *Stellarium*, it'll often display stars that you can't see in the sky with the naked eye, even on the darkest of nights. This might seem pointless, but don't forget that more objects will be visible if you use binoculars and more still will appear if you use a telescope. In addition, *Stellarium* can also display objects that you won't always be able to see on-screen because they're so faint. For this reason, the authors suggest using *Stellarium* in a darkened room if you want to see everything it has to offer. It also offer extensive options to mimic real-world viewing conditions including atmospheric affects.

Practical observation

The easiest way to start your voyage of discovery is observe the night sky outside your home. However, unless you live away from centres of population, the sky will suffer from light pollution which means you'll struggle to see some objects that would be clearly visible in a more remote area. The upshot of this, therefore, is that you might need to pack your laptop in the car and go in search of darker skies.

And it's not just the sky that can hinder your observations; your own eyes can work against you. When you first go out into the dark, your eyes won't be

QUICK TIP

If you're confused by light years, it's a measure of distance, not time. It's the distance light travels in a year which is 9,500,000,000,000km. Even so, a light year is a short distance in astronomical terms – the nearest star other than the sun is 4.3 light years away,.

» BUYING A TELESCOPE

Our first advice here is not to rush into making a purchase. Spend lots of time observing with your naked eyes and binoculars. Only if you find you're still fascinated by the vastness of space after several months, and if you're still braving the cold several nights each week, this might this be the time to consider buying an astronomical telescope. This isn't the place to give detailed guidance on how to choose a telescope, let alone reviewing a range of models. But if all you take away from this is that you really ought to spend a lot of time reading up on the subject before investing, perhaps, several hundred pounds, we'll have accomplished our mission.

In a nutshell, however, you can pay anything from less than £50 to many thousands and your first job is to decide whether to go in at the bottom end, knowing that you'll soon want to upgrade if you decide to pursue astronomy more seriously, or whether to invest in a bit of future-proofing. The only other things we should point out at this early stage is that, having used *Stellarium*, you should consider a telescope with a motorised Goto mount. This way, your software can drive your telescope to home in on your chosen planet, star or nebula.

sensitive to dim objects. For this reason, unless you're looking for something bright like the Moon, Mars, Saturn or Jupiter, you'll need to wait a few minutes for your eyes to become dark adapted. Remember that this adaptation will be reversed if you look at anything bright, and that includes your laptop screen. So, if you want to refer to *Stellarium* during your observations, be sure to select its night vision display mode which renders the sky in dark red on black. The same is true if you use an Android app, as discussed later, although this feature won't necessarily be available on all apps.

Stellarium will show compass bearings so you know which direction to look to see a given object. Unless you know where each of the compass points lie, you'll need to use a compass or a compass app on your phone, to figure out where to look. Alternatively, there are Android apps that provide similar functionality to *Stellarium*, but which use your phone's various sensors so that the display reflects the actual direction the phone is pointing. We used *SkEye*, which looks just fine, but there are loads of others, many of them free, so you can try several out to see which you like best.

So, now you know what you ought to be able to see, and where to look, we need to cover the practicalities of observing astronomical objects. To start, we suggest you use nothing more than your own eyes. You won't need any instructions, or the help of *Stellarium*, to see the Moon but do spend a few minutes to identify those bright planets that might be visible, and here we include Venus, Mars, Jupiter and Saturn. Next, try looking for some of the constellations that *Stellarium* will show you, while bearing in mind that they have no real astronomical significance.

The next step up is to use a pair of binoculars. These will help you to see lots of objects that *Stellatrium* identifies but are totally invisible to the naked eye. However, some of the most interesting objects to observe when you're starting out are ones you've already identified with the naked eye, simply because you'll often be able to see so much more.

We'd recommend starting with the Moon, where binoculars will help you to differentiate the dark-coloured plains from the light-coloured highlands and reveal some of its many craters. You could turn your attention to the Red Planet, Mars, but although your binoculars will show it as a larger and redder dot, you won't see any additional detail.

However, for something that's almost guaranteed to amaze you, if you've not seen it before, train your binoculars on Jupiter. With the naked eye it's just a dot but, through binoculars, that dot will have up to four smaller dots close by. These dots, which will shift in position with respect to Jupiter over time, and will sometimes be behind the planet and hence invisible, are called Callisto, Europa, Ganymede and Io. These are the four largest of Jupiter's many moons, and are known as the Galilean moons after their discoverer.

QUICK TIP

Fancy looking for evidence of aliens? The SETI@ Home project (setiathome.berkeley.edu) uses spare processing time on volunteers' PCs to analyse radio telescope data in the search for extra terrestrial intelligence.

Choose to display as much or as little information as you want, thanks to Stellarium's vast number of options.

Onwards to Venus!

Now to a couple of more challenging observations and here, if your binoculars have a suitable mount, it would help to attach them to a tripod to help keep the image steady. First, take a look at Venus. If your binoculars are good enough, and depending on the date, instead of the dot you saw with your naked eyes, it might appear as a crescent or a half circle. This is because, just like the Moon, the planet Venus displays phases. And second, turn your attention to Saturn. If you notice that it isn't perfectly circular or, if it appears to have ears, you're actually seeing its rings, even though you'd need a telescope to see them more clearly. You might also be able to see Saturn's brightest moon, Titan.

As we turn our attention beyond the Solar System, binoculars are again able to reveal detail that is otherwise invisible. For example, not all objects that appear to the naked eye as a single dot of light are what they seem. Some objects that look like stars are actually double stars and will appear as such through binoculars. In some cases the two stars only appear to be close together from our viewpoint on the Earth while others are actually a pair or stars orbiting around each other.

A good one to spot is Mizar in the handle of the Plough, aka the Big Dipper, although some people can see two stars unaided. However, one of the most spectacular double stars is Albireo in the constellation of Cyguns. You'll have to wait until summer, and you need a good pair of binoculars to see it, but it's particularly spectacular because the two stars are

» ONLINE RESOURCES

The Internet offers a vast amount of astronomical related resources but if you want to be inspired by the variety and beauty of all those distant words, visiting the NASA Image Galleries at **www.nasa.gov/multimedia/imagegallery** is a must.

Although NASA's images are static, the amount of detail you can see is staggering. Many of the images were obtained from the various space probes that have been launched towards every corner of the Solar System for decades. So, for example, returning our attention to Jupiter's moon Io, you can even see the plume of one of its volcanoes firing sulphur 300km into space. And in addition to seeing how planets and moons looked from fly-bys or from orbit, in the case of Mars you can look at images from the planet's surface as captured by the various Martian rovers. You can even view historical images of the Moon's surface which were captured by astronauts in the Apollo missions that took man to the moon from 1969 to 1972.

It's not just the Solar System that you can see in minute detail, either. Thanks to the Hubble Space Telescope, you can start to appreciate the wonders of the stars beyond our immediate neighbourhood and even galaxies.

different colours – blue and amber.

Moving on from using binoculars, the next obvious step is to use a telescope and we provide some general information in the box on page 45.

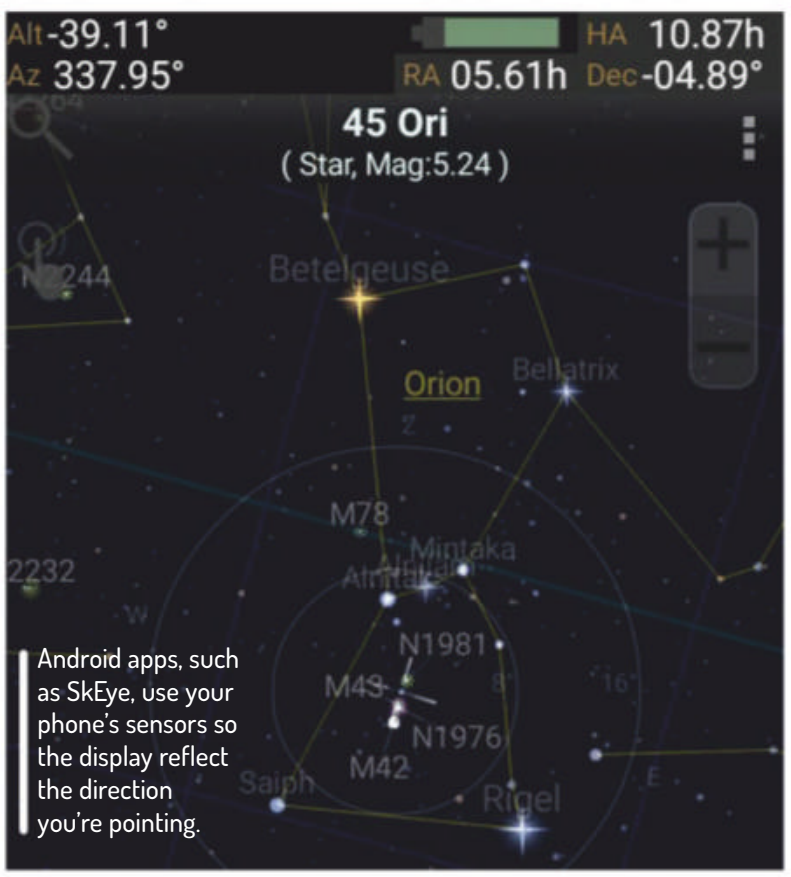
Zooming in with Celestia

Having seen some of the remarkable objects in our Solar System and beyond with your own eyes, perhaps aided and abetted by a pair of binoculars, you'll surely be interested to learn more about some of these planets, moons and stars. After all, even with good binoculars, many of them still only appear as specks of light, and that begs the question of what they'd really look like if you were able to visit them.

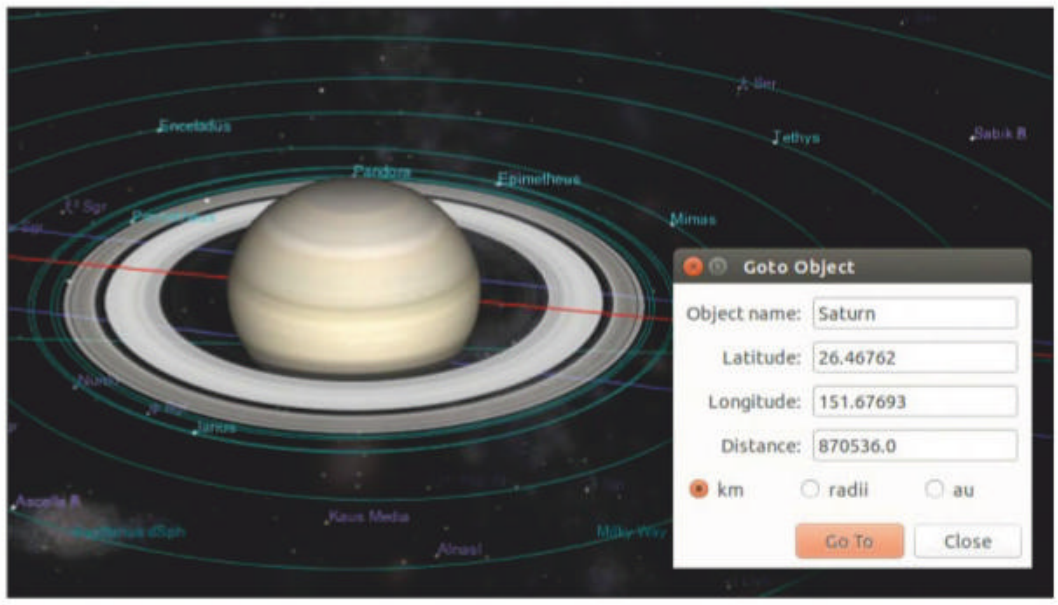
To help answer that question we're turning our attention to another open source package called *Celestia*. Before we get started, though, we should point out that, despite it having an enthusiastic following, it's not updated as frequently as we might hope with the result that it doesn't appear in some of the more recent repositories. For example, it's available in the Ubuntu 14.04 repository but not 16.04, which means you'd have to employ other means of installation. Note also, that in addition to the package itself, you'll need to separately install two sets of data files that are called **celestia-common** and **celestia-common-nonfree**.

When you first start *Celestia* you'll see an image of the Earth as it would appear from orbit and, already, you're seeing something markedly different from the view provided by *Stellerium,* which shows you how the sky looks from the Earth's surface. Indeed, *Celestia* enbales you to define a viewpoint anywhere in the Solar System or, for that matter, far beyond. For views of the planets in our Solar System and their moons, the view that you'll see uses actual photography of those bodies, as captured by NASA's various planetary probes. And for more distant objects that spacecraft have never visited, the image you'll see is based on the scientific view of what they will probably look like.

Let's imagine that you observed the planet Jupiter with your naked eyes and then discovered its four Galilean moons using binoculars. In that case, you could



Android apps, such as SkEye, use your phone's sensors so the display reflect the direction you're pointing.



Like Stellarium, Celestia provides options to you can decide what to display, even including orbits, like those of Saturn's moons.

well be interested to see what those white dots would really look like from a spacecraft in close proximity.

Let's start with Io. Search for it in the Navigation menu and then, in the same menu, select Goto Object. Io will be displayed from some arbitrary distance, but if it's too small, select a smaller distance in the Goto Object window. When you start to see the detail on the surface of Io, you'll notice several small dark circles which, in fact, are volcanoes. It transpires that Io is the only known object in the Solar System, other than the Earth, with active volcanoes.

So far, you've seen Io but, depending when you view it, it might not be obvious that it's orbiting Jupiter. Probably the best way to address this is to fast forward, by repeatedly using the 10x Faster option in the Time menu and, eventually, you'll see Io in relation to Jupiter.

Going beyond Io, you really ought to take a look at Jupiter's other large moons. If you expected that all Jovian moons would look similar, you'll be in for a surprise. Also, as with *Stellarium*, you have total control over what's displayed on-screen. You can turn the outlines of constellations on and off, choose what, if any, types of objects to label, display orbits, information on the selected object, and so much more.

In our use of *Celestia* so far, we've positioned ourselves at fixed points in space to get a good view of the objects we're interested in. This is just a start, though, and if you want something a bit more interactive, you can even view the heavens while taking the controls of a simulated spacecraft. In addition to flying between planets and stars, you can go into orbit around them, and you can even land on the surface of planets and moons. We'll leave you to figure all this out yourself for which you'll probably need to consult the comprehensive user guide at **www.celestiamotherlode.net/catalog/documentation.html**.

Celestia will keep you entertained for countless hours, just by trying things out yourself. However, your enjoyment of *Celestia*, and the whole universe for that matter, can be enhanced by taking advantage of the contributions of others. A useful feature of *Celestia* is the ability to create and playback scripts, something that lots of users have employed to produce animated tours of parts of the Solar System and beyond. The best place to find add-ons for *Celestia* is **www.celestiamotherlode.net**. We're confident that you'll find plenty here to enhance your appreciation of the sky at night and help you to better understand the remarkable worlds that are just waiting to be discovered. LXF

QUICK TIP

Did you know that a telescope is a time machine? Because the light we see from distant stars and galaxies will have been travelling for many years to reach us – often millions of years in the case of other galaxies – we're seeing those objects as they were some time in the past, sometimes the distant past.

Alan O'Donohoe is a specialist leader in education at exa.foundation, part of Exa Networks Ltd.

» KICKING IT ALL OFF

Back in 2011 I felt a compelling need to change the curriculum in the school I taught at. I was the subject leader for ICT, and I wanted to lead the move away from a dependency on proprietary software that our students simply couldn't afford to use at home unless I encouraged them to break copyright law! My interests in Linux and the FSF were developing, and I wondered if this new device called the Raspberry Pi might be the answer I was searching for.

After attending BarCamps, Python meetups and Linux/FSF themed un-conferences in Manchester I was inspired to start something similar in Preston where I taught, but with a focus on the Raspberry Pi computer, digital making and supporting computing education. Although many forums, channels and online video tutorials existed, I thought they fell short of the deep impact that a human interaction can have.

My idea was to establish a regular monthly meet-up where anyone with an interest in free software, Linux, Raspberry Pi, electronics, robotics and education could meet with others in a supportive space. When I described the idea to my wife, she suggested the name Raspberry Jam. I booked a meeting room in my school to host our first event, and within days all 30 tickets had sold.

Martin Bateman from UCLan in Preston offered a teaching room we could fit more people in, Room 101! A few months later we held our first Preston Raspberry Jam and we haven't stopped.

No escaping the Raspberry Pi

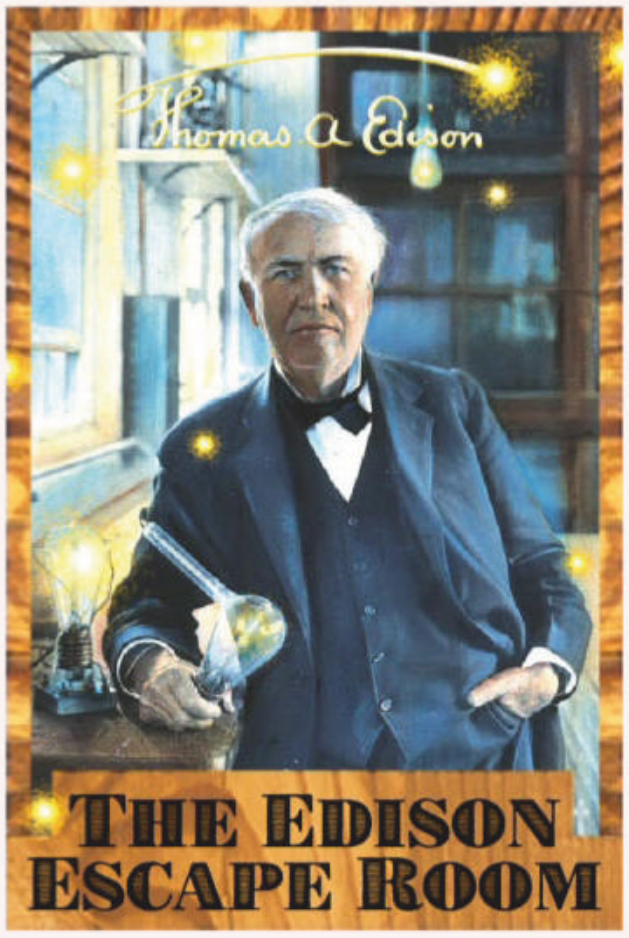
Discover the escape room that's Pi powered!

The awarding-winning Palace Games, which is behind a string of award winning escape rooms across the US – has utilised the Raspberry Pi to power the ticks and puzzles it uses in its escape rooms.

What's an escape room, we imagine you just asked? Escape rooms are real-world puzzles contained in the form of a room, which you need to escape. Rooms are usually themed and can be scary, historic and just silly, but all will require you to solve a number of puzzles within a set time to escape. They're group events and are often used for team-building exercises.

Behind the scenes of the Edison Escape Room is a network of Raspberry Pis running the Banyan Framework, which is designed to orchestrate event-driven, physical computing devices in an asynchronous manner. Within the game this translates into monitoring sensors, opening doors, controlling lights and playing media.

Discover the fun the Raspberry Pi US team had there by visiting **www.raspberrypi.org/blog/raspberry-pi-escape-room**.

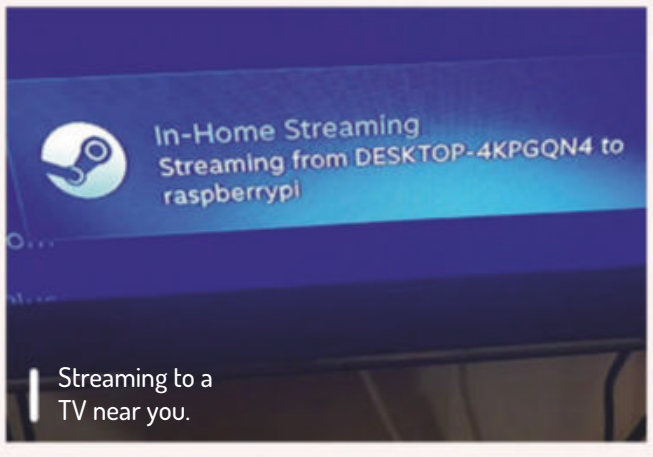


A room from which one can escape... any bright ideas on what to call such a location?

Steaming Pi!

Game streaming.

As we reported in our main news section, Steam Link is dead. Yet as the Pi foundation has discovered the open source release of the software runs almost flawlessly on the Pi. So if you have a Pi going unused it might be time to dust it off! Learn more at **www.raspberrypi.org/blog/steam-link-raspberry-pi**.



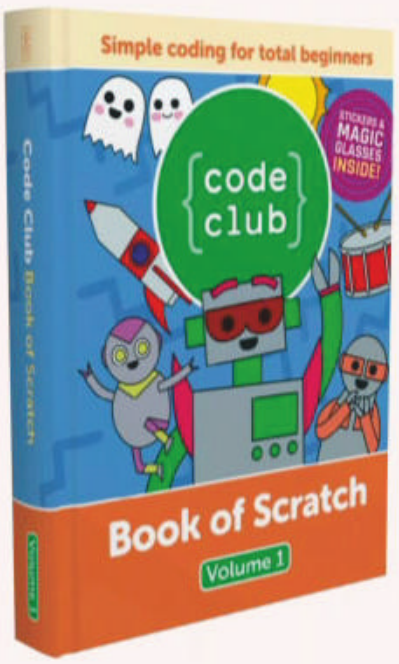
Streaming to a TV near you.

CREDIT: Raspberry Pi Foundation

Code Club Book

It's your own bookie book!

What happens when you write a book from scratch? Why, a book on Scratch, of course! Code Club has a fresh, easy-to-follow guide to everyone's favourite visual programming language. It's packed with amazing projects and stickers, and is sure to be a hit! More at **https://blog.codeclub.org.uk/2018/12/06/welcome-to-the-first-ever-code-club-book**.



What's this strange object?

Picade X HAT

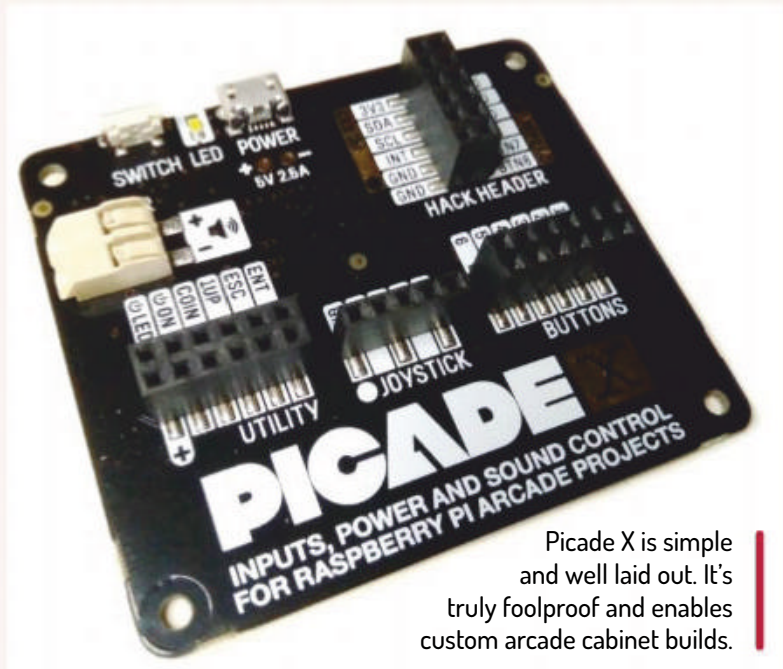
Like a fun-loving Terminator, **Les Pounder** has travelled back in time to the 1980s to play arcade games using his Raspberry Pi.

IN BRIEF

An all-in-one solution for arcade hobbyists using the Raspberry Pi as their development board. It's made up of an audio interface and an interface for your own arcade control inputs. Picade X is a simple way to start your adventures in arcade emulation. The board also offers a power switch to safely power up and down your cabinet.

Coming from a seaside town, the arcade was the pinnacle of entertainment in the 1980s. We cut our teeth playing *Afterburner*, *Space Harrier*, *Bombjack* and spent far too much money to prove our worth. Over the years arcades have seen a downturn with more people electing to play home consoles. But retro is now cool and the Raspberry Pi is now part of the "desktop arcade" experience. You can either buy a complete kit or make your own, and this is where Picade X HAT comes in.

Picade X HAT is an all-in-one control, power and audio solution for 40-pin GPIO models of Pi. Fitting it on top of the GPIO is simple, and the accompanying software install is easy to complete. Picade X HAT is compatible with Retropie which is the de facto standard for Raspberry Pi arcade, computer and console emulation. Aspiring hackers can also use Picade X HAT in their projects, especially games made using PyGame or other game dev libraries.



Picade X is simple and well laid out. It's truly foolproof and enables custom arcade cabinet builds.

Get connected

There are many connections around the board, starting with the audio out connection that offers a mono push fit connector with 3W of power. That's enough for the thumping bass of many classic scrolling beat 'em ups! We also see a micro USB port used for power. The Picade X HAT receives the power before the Pi and then feeds the power to the Pi via the GPIO.

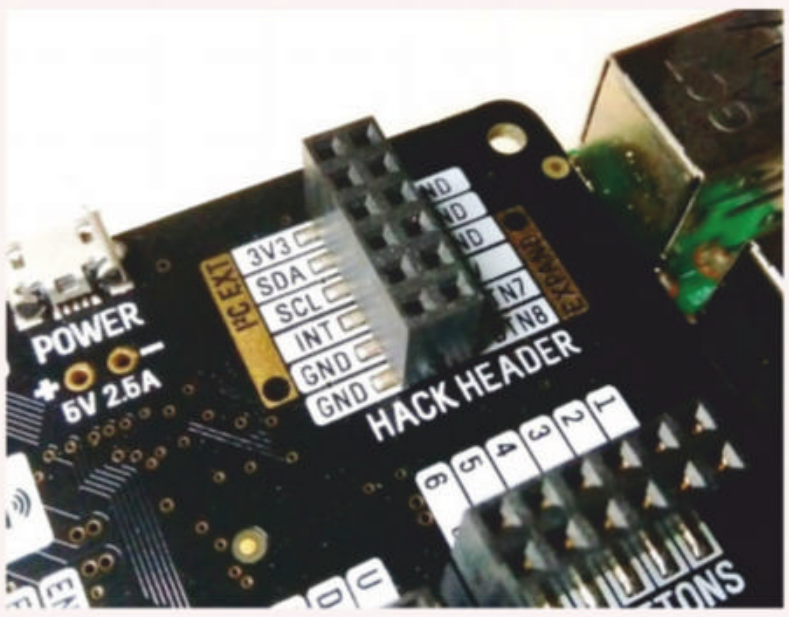
Control inputs are for a four-way joystick, and six player buttons – plenty for *Street Fighter 2* fans! There's also a selection of utility inputs for the simulated coin mechanism, player selection, menu options and power control. The software that this switch triggers can be used to power up the Pi and to soft shutdown the Pi, which preserves the filesystem. There's also a "Hack Header" which offers connections to some of the underlying GPIO pins, and to the I2C interface, so makers can use these pins to drive extra functionality.

Picade X HAT is all about choice. It enables you to source your own controls and inputs for your custom arcade experience. Sure, we can source an arcade to USB interface, use the onboard audio out and write our own software to control it all. But with Picade X HAT we're buying a product that has all the essentials in one board. We just need to bring a screen, controls, speaker and Pi.

Building a cabinet around this board is super simple and anyone can do it. Just place this board on your Pi, install the software and then plug in the buttons or controls where instructed and you too can be playing *Streets of Rage* like a pro. As for cabinets, a quick look on eBay will provide a plethora of kits ready to build.

Picade X HAT is for those of us that want to build their own arcade experience around the Raspberry Pi but have little electronics knowledge. It's easy to use and sensibly priced when we consider how much a USB arcade interface board can be. Typically i-PAC interface boards retail for $40 on their own! The Picade X HAT is best used with Pimoroni's own arcade cabinet, Picade, but you can build your own cabinet for not much money.

Picade X HAT is excellent value for money, and worthy of your time. Relive your youth with the board, and save those coins for penny shuffles and claw games! LXF



The Hack Header feature on the Picade X HAT gives you extra connection options.

VERDICT

DEVELOPER: Pimoroni
WEB: https://shop.pimoroni.com/products/picade-x-hat
PRICE: £15

FEATURES	9/10	EASE OF USE	9/10
PERFORMANCE	10/10	VALUE	9/10

If you have ever wanted to build your own arcade cabinet, then the Picade X HAT is the perfect go-to solution.

» Rating 9/10

DROPBOX

Sharing data from remote sensors

Les Pounder reveals how to capture temperature data from remote devices, and how teams can work on a single data source. All for one and one for all!

OUR EXPERT

Les Pounder is a freelance maker who works with organisations such as the Raspberry Pi Foundation to promote maker skills. He blogs at bigl.es

The Raspberry Pi is ideally suited to working in remote locations and in this tutorial we shall use a remote Raspberry Pi to gather temperature data using a DHT11 temperature sensor and then share that data with our data centre team via Dropbox. Then the data centre team can use their awesome Linux supercomputers to crunch the data and save the world!

All of the code, further screenshots and a detailed high-resolution circuit diagram can be downloaded from **https://github.com/lesp/LXF246-Sharing-data-from-remote-sensors/archive/master.zip**.

Sensor sweep

Connecting the DHT11 to the Raspberry Pi is simple: we used the three-pin version of the DHT11. Connections, looking at the blue plastic cage, are from left to right:

- **Pin 1 to 3V3 GPIO**
- **Pin 2 to GPIO17**
- **Pin 3 to GND**

Use the female-to-female jumper wires to connect the DHT11 to the Raspberry Pi. Now power up your Pi and from the desktop open a Terminal (*LXTerminal*) and type the following command to install the software for the DHT11:

```
$ sudo pip3 install Adafruit-DHT
```

Our next install will be the **dropbox** library for Python. Again in the terminal we type the following:

```
$ sudo pip3 install dropbox
```

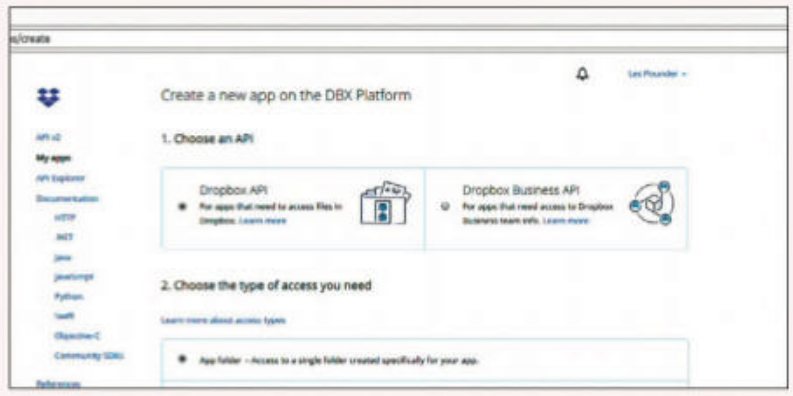
With the software installed, we now turn our attention to setting up our first Dropbox app.

YOU NEED

- **Any model of Raspberry Pi**
- **Internet connection**
- **The latest version of Raspbian**
- **DHT11 temperature sensor**
- **3 female-to-female jumper wires**

Dropbox setup

Dropbox is a cloud storage application that's cross-platform and it has a rather good Python API. To use the API we need to have a Dropbox account. So head over to **www.dropbox.com** and sign up. Once we have an account we next need to login to Dropbox for Developers (**www.dropbox.com/developers**) and from there click Create Your App. This will open a new page that asks us to choose the API. In this case we need the Dropbox API. The required access is "App Folder" and for the final step we need to name the app. We called ours Experiment Data. Click Create App and we shall go to the next screen that gives us a breakdown of the app.



Creating an app gives us lots of choices. The first is to use the Dropbox API, then to save the file to its own app folder. Finally, name the app.

Right now all we need to do is look for Generated access token and click Generate. This will create your access token which our project will use to access Dropbox via Python. Make a note of the token, but don't share the token because it's private to your account.

With Dropbox setup complete, we can now focus on the Python code. Open the Python 3 editor, found in the main Raspbian menu under Programming. Now click File>New to create a new blank document. Immediately save the blank document (File>Save) as **experiment.py** before moving on. Remember to save often.

Our first line of Python code will tell Raspbian where to find the Python interpreter. We'll need this to later run our code autonomously via the terminal:

```
#!/usr/bin/python3
```

The next line of Python code sees us importing libraries to provide extra functionality. In this case we import the Adafruit DHT and Dropbox libraries that we installed earlier. We also import the datetime library used to provide calendar and time data:

```
import Adafruit_DHT, datetime, dropbox
```

To use the Adafruit library we create an object that references the library. By typing "sensor" we can access the entire library without the need to type lots of text:

```
sensor=Adafruit_DHT.DHT11
```

The next object tells the code where to find the DHT11 sensor. The output for that sensor is connected to the GPIO 17 of the Raspberry Pi:

```
gpio=17
```

Our next step is to create an object called `temperature`, which will be a tuple containing two items. First the humidity is saved as item 0 in the tuple.

The next item, at position one is the current temperature in degrees Celsius. The object requires two arguments and these are the type of sensor DHT11 and to which pin it is connected 17:

```
temperature = Adafruit_DHT.read_retry(sensor, gpio)
```

To use Dropbox with Python we need to use the token, which you hopefully recorded earlier. The variable `token` is used to store this long string of characters:

```
token = "<YOUR TOKEN HERE>"
```

Let's make a connection to the Dropbox API by creating an object called `dbx`.

```
dbx = dropbox.Dropbox(token)
```

For debug purposes, we'll print the temperature data to the Python shell. This isn't required, but useful to check that the sensor is working. We use simple positional formatting in the print function. You can see `{}` in the text, and in here Python will drop the temperature date stored in the temperature tuple, at position 1:

```
print("The temperature is {}C".format(temperature[1]))
```

To obtain the current date and time we'll create a variable called `timestamp` that will get the current time, and then use a little string formatting to convert it into something human readable:

```
timestamp = datetime.datetime.now()
timestamp = str(timestamp.strftime("%Y-%m-%d %H:%M"))
```

With the temperature and time data saved, we now need to write this to a file. For this we use Python's built-in file class to create a file called **data.csv** (if it doesn't exist, Python will create it) and set the file to "append" – this adds the data to the end of the file.

```
f = open("data.csv", "a")
```

We then write the timestamp (date time) to the file, and a trailing `,` as our file is a comma separated value file, used in spreadsheets.

```
f.write(timestamp+",")
```

Then we write the temperature data:

```
f.write(str(temperature[1]))
```

Our next line forces Python to insert a new line, effectively creating a new row in our CSV data ready for the spreadsheet. Then we close the file to ensure that everything is saved correctly.

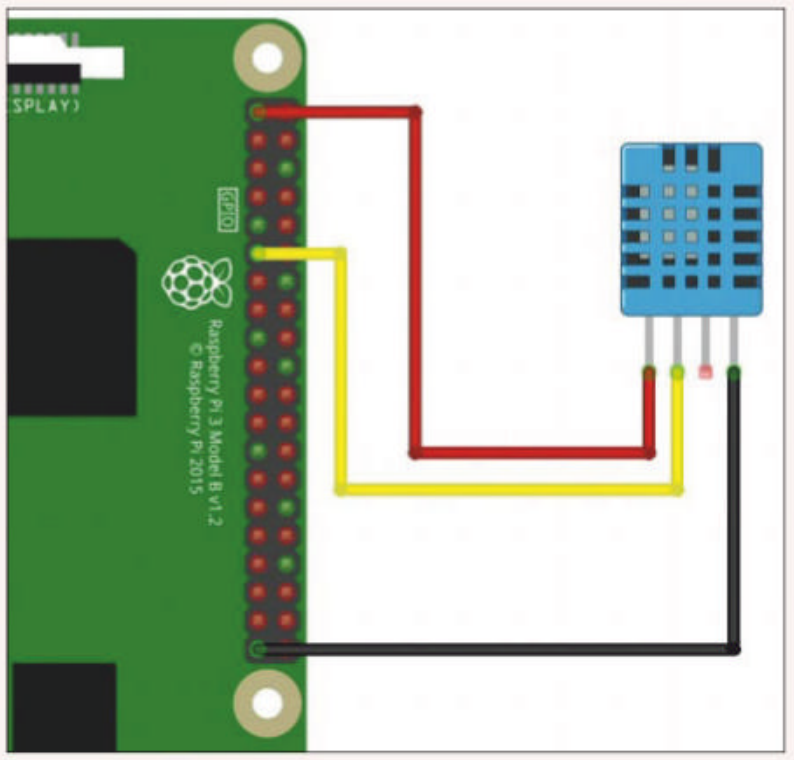
```
f.write("\n")
f.close()
```

The final two lines of Python! Here we open the **data.csv** file in read-only mode and then instruct Python to read it line by line into an object called data. Then we use the Dropbox library to read the file, then save it as **/data.csv** (this is the location in the Dropbox app folder), we mute the output, and then tell Dropbox to overwrite any files matching the name of our file. This will update the file each time the code is run:

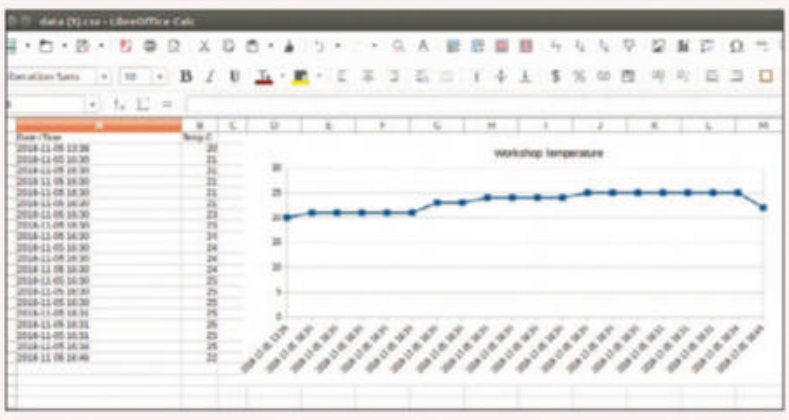
```
with open("data.csv", "rb") as data:
    dbx.files_upload(data.read(), '/data.csv', mute = True, mode=dropbox.files.WriteMode.overwrite)
```

That's the code! Click File>Save and when ready start the code for a quick test by pressing Run>Run Module from the menu. We can see the Experiment Data folder on the Dropbox website, and via the Dropbox app if used.

So now we can record data, but how can we have it without pressing a button? Well in the boxout (*below*) we show you how to make your code into a application that will start on boot. LXF



Our circuit is simple, thanks to the DHT11 requiring only three connections. In this diagram we have the four-pin version of the sensor. But use the three-pin version instead.



Downloading the data was easy and thanks to LibreOffice Calc we were able to produce a chart for management to study!

QUICK TIP

The easiest way to get the sensor data from Dropbox is to install the app. This will create a folder structure for the application we have created, and our data.csv file will be in the root of the Apps directory.

» CREATING A BACKGROUND APP

At the start of the project we instructed the code to look for the Python interpreter, and now we need to take two steps to make our app run.

First, we need to make the code run from the terminal rather than open the Python editor. In a terminal we shall make the file executable by changing its permissions:

```
$ chmod +x experiment.py
```

Now give it a quick test by typing this into the terminal:

```
$ ./experiment.py
```

If the code works, great! If not, double-check before moving on. The last step is to tell our Raspberry Pi to run the code on every fifth minute in the hour (5, 10, 15, 20 and so on), for that we need to use *cron*. In a terminal type the following:

```
$ sudo crontab -e
```

It will ask which editor to use, choose *nano* to follow this tutorial. Go to the bottom of the file and type.

```
*/5 * * * * /home/pi/experiment.py
```

Then press Ctrl+O, then Enter, then Ctrl+X to exit. Reboot your Pi and every five minutes the code will run and update the **data.csv** file.

Now all we need to do is load the CSV file into our favourite spreadsheet application and start crunching data!

RASPIAN

Optimise your Pi for better performance

Christian Cawley presents his tips and tricks for squeezing as much juice out of your Raspberry Pi as possible.

OUR EXPERT

Christian Cawley has spent the past 15 years telling people how to do the same computing tasks but still has no idea how cars work.

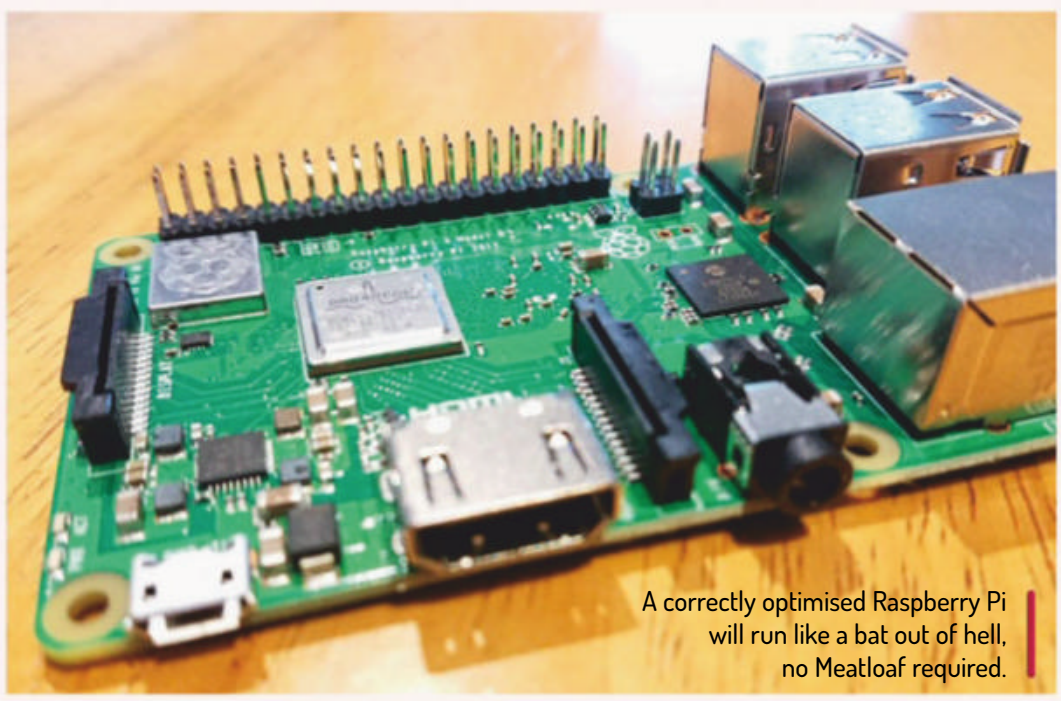
Has your Raspberry Pi slowed down? It's a common problem – it doesn't matter which version of the Pi you're using, performance always suffers if the operating system isn't correctly configured. Using a lightweight operating system such as Raspbian Lite, DietPi, or piCore can help, but there's much more you can do. Performance boosts can be gained from overclocking, changing how RAM is used, and ditching bloatware, among other things.

Power rangers

The wrong power supply can wreak havoc with your Raspberry Pi's performance. With insufficient power to deal with processing requirements, bottlenecks can occur, slowing the computer right down. Low power can also result in read/write issues, which in turn can damage the microSD card (more on that later). Then there's the usual end result of a frozen Raspberry Pi: pulling the power lead and rebooting.

To avoid this, you need to give the Raspberry Pi a reliable power supply. This means using a good-quality PSU (if not the official device) rather than a spare USB port on your PC, TV or power bar.

The official Raspberry Pi power adapter has 100-240V input, and 5.1V 2.5A output. If you're not using the official PSU, you should ensure that your supply matches this, and that the micro USB cable is of good quality, in working order, and fits the Raspberry Pi correctly.



A correctly optimised Raspberry Pi will run like a bat out of hell, no Meatloaf required.

Super-size Pi

Using Raspbian Lite or some other lightweight distro is a good option for a stripped-back experience, but if you're looking for a more feature-packed operating system, the standard Raspbian installation is your best option.

A full gigabyte of storage space can be freed up by removing unused tools such as *LibreOffice*, the *Wolfram Engine*, *Minecraft* and *Sonic Pi*. To benefit from this fully, however, you need to have the full capacity of the SD card. This means using the Expand Filesystem option in the *raspi-config* tool, or the Raspberry Pi Configuration menu on the desktop. You'll find this on the System tab.

Removing what might be unfairly termed "bloatware" on Raspbian is easy. Open a Terminal prompt and type

```
sudo apt purge libreoffice*
sudo apt clean
sudo apt autoremove
```

This will remove *LibreOffice* from Raspbian. Replace the `libreoffice*` command with `wolfram-engine`, `minecraft-pi` and `sonic-pi` to remove those tools.

With these four options removed, you'll have the advantage of increased storage space on your Pi without having the limitations of a lite distro.

Kill JavaScript dead

While it's wise to limit the number of tabs you have open in your preferred browser, further performance boosts can be gained by disabling JavaScript. How you do this will vary. In *Chromium*, use Menu>Settings>Advanced>Privacy and security>Content settings>JavaScript. Here, switch Allowed to Blocked.

Disabling JavaScript can improve overall performance, as well as speed up some any pages that rely on it. While this might mean pages become less functional, it will also help to reduce the load on your Raspberry Pi from adverts and trackers. Note that exceptions can be added to your browser settings to allow JavaScript to run on specific sites.

Underclock, overclock, Pi-clocking's free

Massive performance boosts can be enjoyed by your Raspberry Pi by overclocking the processor. Raspbian

offers this as a built-in feature that can be found in the *raspi-config* tool (other distros may require additional software). Access the overclocking tool by selecting Overclock on the main *raspi-config* menu. Depending on your Raspberry Pi model, you'll have several options here. The default option is None, with the clock running at the basic 700MHz, but you could ramp things up to 1000MHz and above with Turbo mode.

Using the overclocking menu means you can avoid having to worry too much about stress testing as the values on offer have already been tested. However, Turbo mode might cause problems, so it's a good idea to use a cooling solution (see Quick Tip, *right*).

With a Raspberry Pi 3 and above, you can also edit the **config.txt** file in *nano* to specify values for the four overclock settings: arm_freq, core_freq, sdram_freq, and over_voltage. The usual overclocking rules of taking a baseline reading and increasing incrementally apply.

The benefits of overclocking your Raspberry Pi include increased performance for video decoding in *Kodi* or playing more games from the fifth and sixth generation consoles that can currently be emulated on the Pi. Emulation of Windows apps with the *ExaGear* software is also improved.

Ditch the cheapo SD card

Finding the right microSD card for your Raspberry Pi can be tricky, but with the right capacity and speed it can make a huge difference to how your Pi runs.

The secret here is to stick to named brands such as SanDisk, Samsung and Kingston, while keeping an eye on the card's speed and rating. A higher rating indicates better speed and error connection, which will improve your Pi's performance.

Once you've added a quality SD card, it's time to start looking after it. By know you should know how to correctly shutdown a Raspberry Pi. Doing so will prolong the lifespan of the SD card, saving you further outlay (however inexpensive a replacement might be) and the fuss of setting up your Pi again from scratch.

From the desktop environment, be sure to shutdown from the main menu. If you're using a *Kodi* media centre distribution or a retro gaming suite, take the time to find the shutdown option and use this.

If it's quicker, or you're connected to your Pi over SSH, it's easier to shut down from the shell:

```
sudo shutdown -h now
```

You can also schedule a time to shut down the Pi:

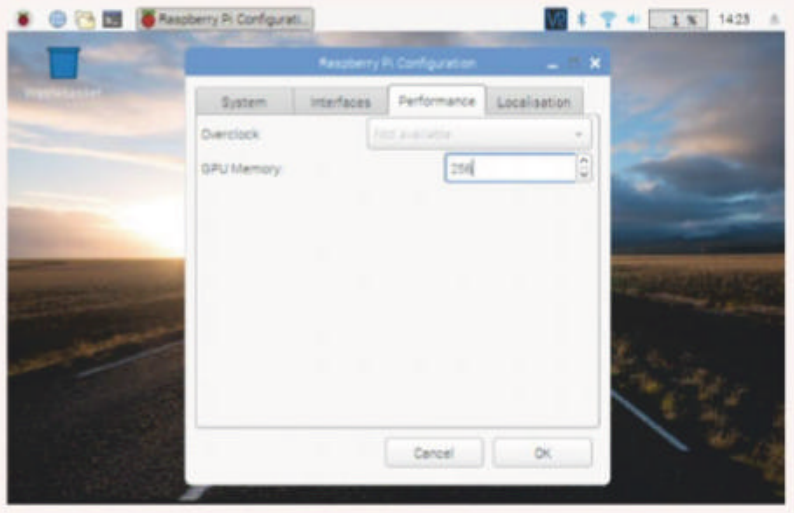
```
sudo shutdown -h 02:46
```

An orderly shutdown will ensure your SD card's integrity is maintained, as well as avoid data loss.

Zippy ZRAM

RAM configuration options for the Raspberry Pi mean that you can easily designate a portion of RAM to the GPU if necessary or reserve it for non-graphics tasks.

Using the Advanced Options sub-menu in *raspi-config*, find the Memory Split option and use this to specify how much memory the GPU should have. For example, if you're not using a desktop environment, then you can set this to 16MB, allowing the Raspberry Pi's processor to benefit from extra memory. For graphic-intensive tasks, however, your GPU will need more of the RAM. Note that even with a Raspberry Pi 3 B+ (which has 1GB of RAM), you'll be unable to assign more than 256MB to the GPU.



Tweak the Raspberry Pi's memory split to assign more RAM to the CPU or the GPU, whichever takes your pick.

Perhaps the best performance tweak for the Raspberry Pi can be made to the RAM, utilising unused space for swap data.

As explained by YouTuber NovaSpirit Tech, ZRAM can be used as a swap disk. While the data won't persist following a reboot, this can speed your Raspberry Pi up considerably.

Begin by grabbing the script from GitHub:

```
sudo wget -O /usr/bin/zram.sh https://raw.githubusercontent.com/novaspirit/rpi_zram/master/zram.sh
```

Next, make the file executable and edit **rc.local** to run the script when you reboot:

```
sudo chmod +x /usr/bin/zram.sh
sudo nano /etc/rc.local
```

Look for the exit 0 line, and above it add:

```
/usr/bin/zram.sh &
```

Hit Ctrl+X to save and exit the file. ZRAM swapping is far faster than relying on even the fastest microSD card, with an instant performance boost. LXF

QUICK TIP

Prolong performance boosts from overclocking by installing heatsinks on the SoC and network controller chips. For the Raspberry Pi 3B+, an onboard fan cooling solution is available for under £10.

» DON'T PUSH YOUR Pi

As versatile as the Raspberry Pi is, it's a good idea to be certain that it's the right device for the project you have in mind.

You might just be using the Raspberry Pi for retro gaming or *Kodi*. It might be acting as a desktop replacement, or as the main component of an IoT project. If your Raspberry Pi is up to the task, then it should work fine. The problem is when you choose a task that the Pi just can't do, even when the hardware appears to be a good fit.

Conversely, the Raspberry Pi might be far too advanced for the task you have in mind, such as flashing an LED over the web, or something equally low-spec.

Perhaps there isn't quite enough RAM; or maybe the issue is with the ARM architecture. Whatever the problem, all the configuration tweaks and tricks in the world won't help if the hardware can't handle the project.

Even the Raspberry Pi 3 B+ isn't powerful enough for some situations. So, what's the alternative? Well, you could look at some of the Pi's nearest rivals, such as the ASUS Tinker Board, Pine64, or even Odroid-XU4. Rather than trying to do everything with the Raspberry Pi, choose the right device for the job.

BASH

File access with ACLs

Providing access to, and ensuring security of, files and directories requires more than chmod – **Shashank Sharma** has the low-down.

OUR EXPERT

Shashank Sharma is a trial lawyer in Delhi and avid Arch user.

You can use the permissions system on Linux distributions to define access rights for the owner/group/others for each file or directory. But this traditional system of permissions-driven access suffers from a critical flaw: namely, you can't define different permissions for different users for the same file. This is where Access Control Lists (ACLs) come in to save the day.

Using the `setfacl` command, you can define the ACL permissions for a file or directory. The `getfacl` command can similarly be used to retrieve the ACL information for any given file or directory. But before we dive into understanding how ACLs work, and putting them to use, a quick primer on the permissions system is in order. Refer to Primer On Permissions *(below)*.

PRIMER ON PERMISSIONS

In the traditional system, for each file you must define read, write and execute permissions, and you must do so for the owner of the file, as well as the group which the user belongs to, and 'others'.

The `chmod` command is used to set permissions, and accepts numerical or symbolic values as argument. Any file can be set to be read, written to, or for scripts, executed. The acceptable values are:

Type of access	Value
Read	4
Write	2
Execute	1

Setting the permission to 664, with `chmod 664 filename` for instance, means that the owner and the group can both read and write to the file, while others can only read it.

Permission	Meaning
777	Owner, group and others can read, write and execute.
755	Owner read, write, execute; group/others read, execute.
640	Owner can read, write; group can read; others no access.
644	Owner can read, write; group and others can only read.

We can't set different access permissions for different users. For instance, there's no means of allowing user 'linuxlala' to read, write, and execute files, while at the same time limiting user 'bodhi' to only read files, especially when the file in question is owned by another user 'knowitall'. That's why ACLs are so useful.

An ACL comprises a set of rules which describe how a specific user or group can access a particular file or directory. There are two types of rules: Default and Access rules. The former applies to directories only, whereas the Access rules can be for individual files or directories. If a given directory doesn't have individual access rules for all the files it contains, such files are governed by the defined Default rules.

Enabling ACL

Unlike other command-line utilities, which require quick installation, the process is slightly more involved in the case of ACL. This is because support for ACL is provided in the kernel itself, so unless your kernel supports it, you can't use ACL on your Linux distribution. While this isn't a problem for any of the popular Linux distributions, which all ship with ACL support, you might have to make some changes if you're running a custom kernel.

To check if your kernel supports ACL, run the `grep -i acl /boot/config*` command:

```
$ grep -i acl /boot/config*
/boot/config-4.15.0-36-generic:CONFIG_EXT4_FS_POSIX_ACL=y
/boot/config-4.15.0-36-generic:CONFIG_REISERFS_FS_POSIX_ACL=y
/boot/config-4.15.0-36-generic:CONFIG_JFS_POSIX_ACL=y
/boot/config-4.15.0-36-generic:CONFIG_BTRFS_FS_POSIX_ACL=y
```

Due to space constraints, we've had to truncate the complete output of the command. If you see `ACL=n` in your output corresponding to any filesystem, or don't find a filesystem in the list, you can't use ACL on those filesystems on your distribution.

The next step is to check if ACL is already enabled for your mounted filesystem:

```
$ sudo tune2fs -l /dev/sda5 | grep acl
[sudo] password for linuxlala:
Default mount options:    user_xattr acl
```

As you can see from this block, ACL is already enabled for our **/** partition. To enable ACL for another filesystem, such as a separate **/home** partition, or any other mount point, you must manually edit the entry for the filesystem in the **/etc/fstab** file.

Open the **/etc/fstab** file in your favourite (*so that's Vi?–Ed*) text editor (you need root privileges to edit this file, so use *sudo*) and change the relevant entry:

```
/dev/sda7  /home  ext4   defaults,acl 1 1
```

You can similarly make changes for other filesystems on

your disk. After making changes, the next step is to remount each modified filesystem for the changes to come into effect, If the filesystem is already mounted and in use, such as **/home**, you must run the mount command with the `remount` option: `sudo mount -o -v remount /home`.

With all the basics out of the way, and if your distribution doesn't ship with it out of the box, install the `acl` package with `sudo dnf install acl` or `sudo apt install acl` on *RPM*- and *DEB*-based distributions respectively.

Using ACLs

You can run the `getfacl <file|directory>` command to retrieve the ACL information for any file. The first three lines of the output reveal the name of the file, its owner and the name of the group the file is associated with, followed by the permissions for the user, group and others. To set ACL, you must use the `setfacl -m ugo:name:permissions: filename` command. Here, the `-m` command option is used to modify the rules. The `ugo` can be either user, group or other; `name` is the name of the user or group for which you wish to define the permissions, and `permissions` are the specific permissions you wish to set, which can be numerical or symbolic. The final argument is the file or list of files for which you wish to set the ACL.

You must always specify the name of the user or group when modifying their ACL; however no name is to be given when modifying the ACL for others. In the following example, we'll first set the ACL for user 'advlala', and then for user 'knowitall' for the same file:

```
$ setfact -m u:advlala:6 ~/Music/'The Shadow'/1938-0213\ -\ House\ Of\ Horror.ogg
$ setfacl -m u:knowitall:0 ~/Music/'The Shadow'/1938-0213\ -\ House\ Of\ Horror.ogg
$ getfact ~/Music/'The Shadow'/1938-0213\ -\ House\ Of\ Horror.ogg
# file: 1938-0213 - House Of Horror.ogg
# owner: linuxlala
# group: linuxlala
user::rw-
user:advlala:r--
user:knowitall:---
group::rw-
mask::rw-
other::r--
```

As you can see, user 'advlala' has only read permission, whereas user 'knowitall' can't access the file at all. Apart from using numerical values to set the permissions, you can also use symbolic values. For instance, to permit user 'knowitall' to read a file, but not write or execute, you can run the `setfacl -m u:knowitall:r-- filename` command. The command `setfacl -m u:advlala:rw- filename` similarly permits user 'advlala' to read and write, but not to execute the specified file.

Advanced options

The process of running the `setfacl` command repeatedly for the same file to set the access rules for different users or groups can become quite tedious. Thankfully, *setfacl* lets you provide a comma-separated list of ACLs. For instance, we ran two separate commands above to define ACL for users 'advlala' and 'knowitall'.

The same exercise can also be done with a single command: `setfacl -m u:advlala:6,u:knowitall:0,g:busybee:rwx filename filename2 directoryname`. If you wish to deploy the same ACL rules to multiple files or directories, you can even do that by providing a list of all the files or directories to the *setfacl* command.

If you already have a predetermined ACL in mind for your various users and groups, and wish to use it for multiple files or directories, you can add all that information to a text file in your favourite editor:

```
group:groupname:permissions
user:username:permissions
default:user:username:permission
```

Let's save save this file as **preset.acl**, for example. You can now run `setfacl --modify-file preset.acl filename`, and all the access rules as defined in the file will automatically be added to the specified file.

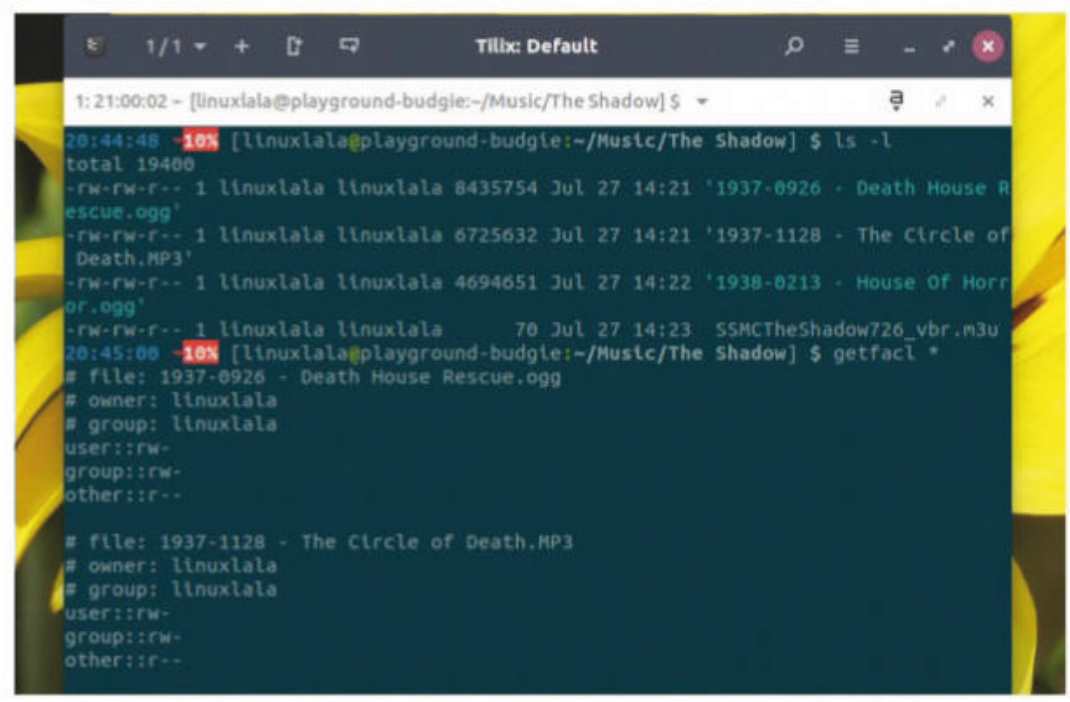
There might come a time when you wish to overwrite the existing ACL for a file with different ones, or remove the ACL altogether. You can use the `setfacl -x` command to remove the ACL for the specified file or directory. But if the ACL were configured using a preset file, you must run the `setfacl -X` command.

Bear in mind that removing the associated ACL for a file or directory doesn't affect the permissions for the owner of the file, or the group the file belongs to. When using the `-X` or `-x` command options, you only have to specify the type of ACL, whether user or group, and the respective name. For instance, the command `setfacl -x u:advlala filename` will remove the ACL for user 'advlala' from the specified file.

When copying or moving files, unless you use the `-p` command option, the associates ACL rules aren't preserved. Run the `getfacl directoryname/* > backup.acl` command to backup the ACL for all files. You can then restore the ACL rules with the `setfacl --restore backup.acl` command. LXF

QUICK TIP

If the output of the ls -l command shows a + sign at the end of permissions, this means the file has an ACL attached to it.



If a file doesn't have any corresponding ACL rules, the output of getfacl is similar to that of ls -l command, only presented differently.

RHYTHMBOX

Listen to music and other audio in Ubuntu

Nick Peers shows you how to use Rhythmbox to play and stream music and spoken audio from a variety of different sources.

OUR EXPERT

Nick Peers is beginning to lose track of where all the copies of his music files are. Thank goodness.

Rhythmbox is Ubuntu's default music player, and it's all you need to both rip and play music stored on your PC. It goes further, too, enabling you to stream music from several sources. In this tutorial we'll show you how to get started by adding your own music, mastering the library navigation and playback controls, plus how to plug in extra functionality where required.

Feel the rhythm

Launch *Rhythmbox* from the Show Applications button by searching for *rhythmbox*. The program will immediately update its library based on the contents of your **Music** folder. Anything copied here will appear in the library ready for playback. *Rhythmbox* supports several music formats out of the box in Ubuntu – basically, if it's supported by *GStreamer*, it'll play in *Rhythmbox*. If you didn't opt to install the restricted extras when setting up Ubuntu, but suddenly want to play MP3, AAC and other music files, simply open your web browser and enter the following URL:

apt://ubuntu-restricted-extras

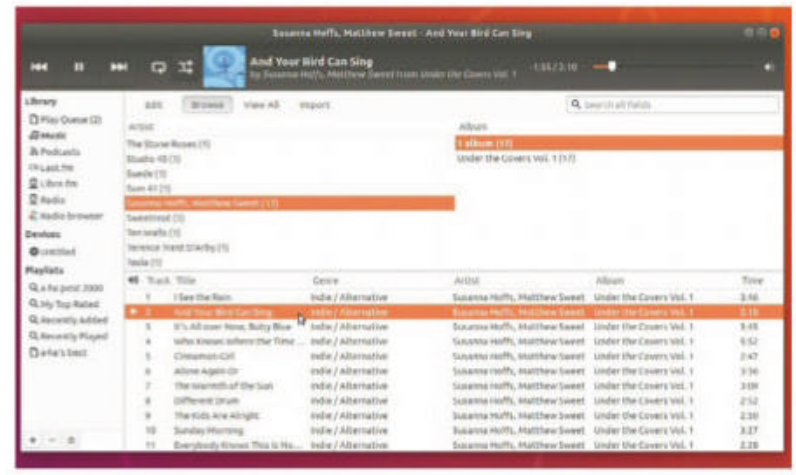
When prompted, leave AptURL selected and click Open Link followed by Install.

Rhythm is a dancer

When the program first launches, you'll see it quickly scours through your Music folder adding all the music files it finds to its library. You're then given a list of all the available tracks, plus two ways to filter them: by artist (left-hand pane) or album (right-hand pane). Also note the column headers above the track listing – if the order seems weird when browsing an album or artist, chances are the list is being sorted by title or possibly artist (on compilations) – you can resolve this by clicking the Track column header.

If you have a gargantuan library, note the Search All Fields box, from where you can quickly filter what's shown by keyword – say an artist or album name, or a simple word such as 'love' or 'rock'.

When you've filtered the view to show the tracks you want to listen to in the order you'd like to hear them, double-click the first track and use the playback controls at the top to play/pause and skip tracks. You'll also notice Repeat and Shuffle buttons next to the album art and track information, plus a timeline slider to move around the track and a Volume button to control the sound level.



Rhythmbox has all the key functionality you need to manage, rip and play your music collection, plus stream from other sources too.

Slave to the rhythm

Another way to listen to music is to create your own custom playlists – the digital equivalent of mixtapes if you're feeling nostalgic. To do this, click the + button in the bottom left-hand corner of the *Rhythmbox* window and choose New Playlist. You'll see New Playlist appear under Playlists – change its name to something suitable and then switch back to Music view.

From here, locate the tracks you wish to include, then simply drag and drop them on top of the playlist title.

» EXPAND RHYTHMBOX

Thanks to its support for plugins, you can add extra features to *Rhythmbox* easily. To do so, click *Rhythmbox* on the desktop's top menu and select Plugins from the drop-down menu. Plugins include *Grilo Media Browser*, which adds support for a wide range of local and internet streaming services, including the ubiquitous DLNA standard.

Fed up of constantly adjusting volume between albums? Enable *ReplayGain* to keep the level consistent. *Song Lyrics* will enable you to display lyrics when listening to music by pressing Ctrl+L, while *SoundCloud* lets you browse and play sounds from the website.

These plugins scratch the surface of what's available – for more head to **wiki.gnome.org/Apps/Rhythmbox/Plugins/ThirdParty**.

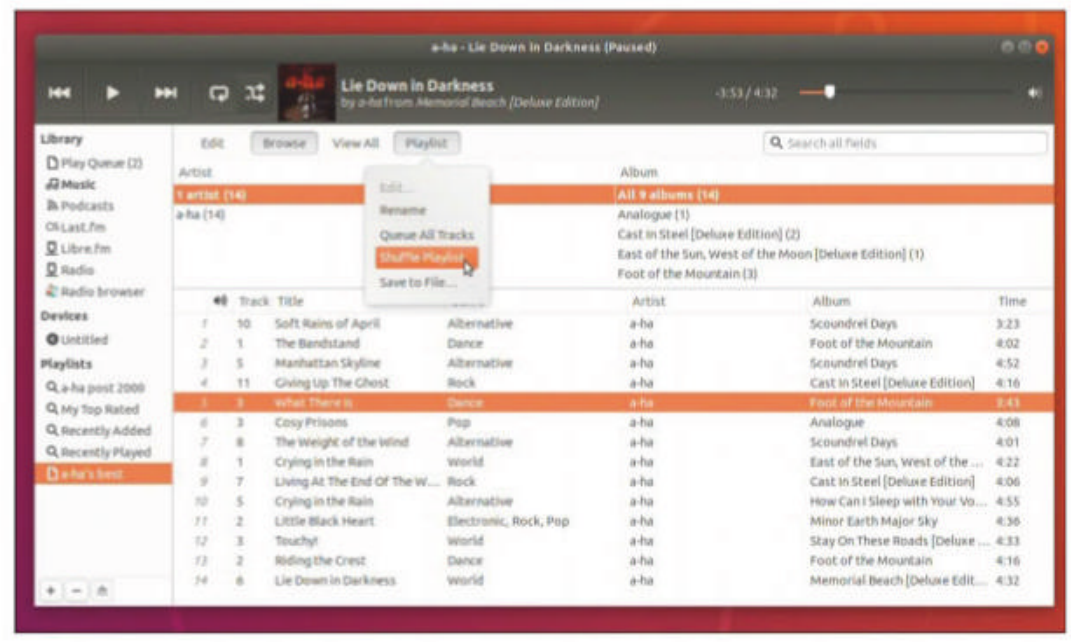
Review the playlist by selecting it, then use the Playlist button underneath the playback controls to bring up a pop-up menu from where you can queue the tracks for playing, shuffle the playlist order and even save the playlist as a file for back-up purposes.

If you'd like to burn a playlist to a CD, first open your web browser to install a required plugin:

apt://rhythmbox-plugin-cdrecorder

Once installed, open the *Rhythmbox* menu from the top bar and choose Plugins, then tick Audio CD Recorder. Now browse to the playlist in question, click the Playlist button and choose Create Audio CD.

These user-generated playlists are called 'static playlists' – to create dynamic playlists that automatically include tracks of a certain genre, artist or other criteria, check out the step-by-step guide below.



Spice up your music collection by generating playlists – choose the right songs and you too can drive your loved ones to distraction by playing the same tunes over and over again.

Rhthym is gonna get you

Not everyone wants to store their music in the Music folder. You may have your music on an external drive, for example, so how do you point *Rhythmbox* to this new folder? Simple, from your main Music library view, click Import. Click the drop-down menu next to Music and choose Other. Browse to the folder you wish to include in the library and click Open.

A list of tracks will appear. Leave Copy Files That Are Outside The Music Library unticked unless you wish to physically transfer the files to your main Music folder. Either click Import XX Listed Tracks to add all of them to your library, or simply select those you want to include before clicking the button. Once complete, click Close.

Alternatively, you might prefer to point *Rhythmbox* to the drive or folder containing your music – do this by opening the *Rhythmbox* menu again and choosing Preferences. Once there, click the Music tab and click Browse next to Library Location to switch.

Rhythmbox can also rip music from CD to your Music folder (and hence into your library). Pop the disc in the drive and wait. After a short pause you should see Audio CD or the album title appear under Devices with a CD icon. Select this and if it's been correctly matched to *Rhythmbox*'s online database you can either play the CD directly or click Extract to rip its contents to your library. If any details are incorrect, you can manually change them by clicking on the erroneous value and typing replacement text.

Other sources include portable music players (these should be detected automatically, and include iPods, MTP players and Mass Storage players – if they only show up in Ubuntu and not *Rhythmbox*, open Terminal, then use the *cd* command to browse to the top level of your music player before issuing the following:

```
touch .is_audio_player
```

The player should then appear under Devices in *Rhythmbox*, ready for browsing and playing.

Fascinatin' Rhythm

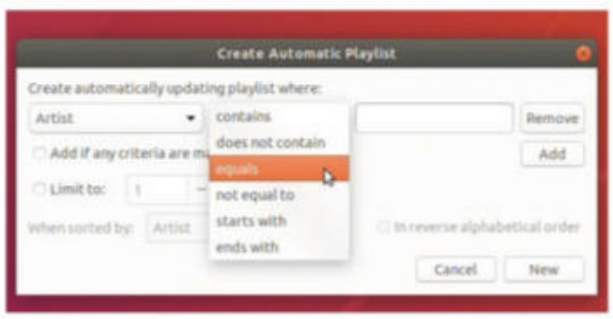
These days, streaming is all the rage, and *Rhythmbox* can handle that, too. DAAP shares are standard, and *Rhythmbox* should be able to detect any advertising themselves on your local network, including shared *iTunes* libraries. For other streaming protocols, check out the box on the opposite page on extending *Rhythmbox*'s functionality using plugins.

Rhythmbox also supports internet radio – click Radio under Library as well as Last.fm and Libre.fm services. You can also add podcasts – from the Podcasts screen click Add to search supported stores (including iTunes) for content or manually enter a podcast's RSS feed if you know it. From here you can browse and listen to individual episodes or click Subscribe to have content automatically download and new content added when it becomes available. LXF

QUICK TIP

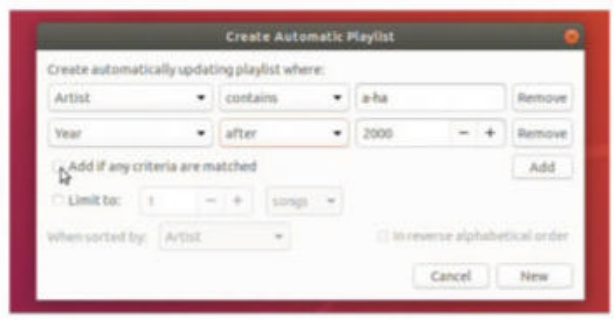
Don't like the *Rhythmbox* interface? Open the *Rhythmbox* menu from the top bar, select Plugins, then Alternative Toolbar. Or use Rhythmbox > View to choose which elements to show.

CREATE SMART PLAYLISTS WITH RHYTHMBOX



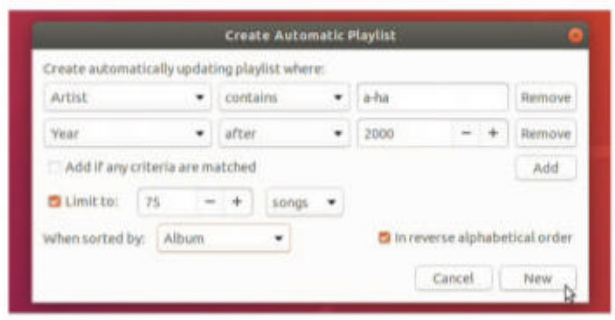
1 Set basic criteria

Click the + button and choose New Automatic Playlist. Choose which field (from Title or Artist to Composer or song duration). Choose your matching criteria. Available options depend on the criteria that you choose.



2 Choose multiple criteria

Enter your first value. To match (or not match) using multiple criteria, click Add as many times as you like and repeat. By default, all criteria must be reached, but you can tick Add If Any Criteria Are Matched to turn it into an OR search.



3 Final options

You might end up with a playlist that contain hundreds of songs. Tick the Limit To button to restrict the number by songs, minutes or file size. Next, set a sort order. Finally, click New to review – choose Playlist > Edit to make further changes.

BACK ISSUES » MISSED ONE?

ISSUE 245
January 2019

Product code:
LXFDB0245

In the magazine
Roll up your sleeves and get making, with our handy feature on bread boards, soldering irons, LEDs and more. Then design a magazine in Scribus, write a Sudoku checker in Go, learn what you can do with Kubernetes, and we review five web servers.

DVD highlights
Experience the might of Manjaro 18, and try out 32- and 64-bit Fedora.

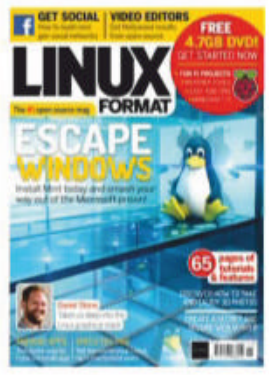
ISSUE 244
December 2018

Product code:
LXFDB0244

In the magazine
Discover how to run Linux in a virtual environment with our 10-page guide. We review five office suites, reveal how to make your web forms bot-free, and run through how to set up a video player on the Raspberry Pi. And we go cave diving with open source technology!

DVD highlights
Try Elementary 5.0, the distro that's ideal for beginners. Plus Ubuntu 18.10.

ISSUE 243
November 2018



Product code:
LXFDB0243

In the magazine
Leave Windows for Mint-flavoured pastures new – we have an in-depth guide. Five video-editing packages go under the *Roundup* microscope, we explore 3D photography, build an Android app and get the most from weather sensors. Plus: run a secret server.

DVD highlights
Cehck out Linux Mint 19 Cinnamon, Peppermint OS 9 and Slax Linux 9.5.0.

ISSUE 242
October 2018

Product code:
LXFDB0242

In the magazine
Pro advice on choosing your next distro, how to build one yourself and 20 of the best available. We test five email clients, reveal the story behind open-source gaming, how to use *Audacity* to create show-stopping art, and ways to jazz up your shell.

DVD highlights
A triple whammy: Manjaro Gnome, Mint 19 MATE and Bodhi Linux 5!

ISSUE 241
September 2018

Product code:
LXFDB0241

In the magazine
Get ahead of hackers by using their techniques and tools. Then dive into the world of retro gaming, get Bluetooth working with Linux, explore machine-learning frameworks, and find out how algorithms have helped to solve heinous crimes!

DVD highlights
Complete hacking toolkit, including the Light version of Kali Linux 2018.

ISSUE 240
Summer 2018

Product code:
LXFDB0240

In the magazine
Discover how you can get more from the latest version of Mint. There's also insights on machine learning bots, desktop security, video-ripping tips using *Handbrake*, while our *Roundup* covers rolling distros. Plus, we report back from this year's Pycon.

DVD highlights
Check out Mint 19 Cinnamon and MATE, together with Q4OS 2.5.

To order, visit myfavouritemagazines.co.uk

Select Tech from the tabs of magazine categories, 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

UK subs turn to p28

NOT FROM THE UK?

SUBSCRIBE!

Don't wait for the latest issue to reach your local store – subscribe today and let *Linux Format* come straight to you!

3 GREAT WAYS TO SUBSCRIBE

Print, digital, and print + digital bundles!

» USA
From $15
every 3 months

» REST OF THE WORLD
From $15
every 3 months

» EUROPE
From €15
every 3 months

IT'S EASY TO SUBSCRIBE!

Click: **www.myfavouritemagazines.co.uk/sublin/**

Call: **+44 344 848 2852**

Lines open 8AM–7PM GMT weekdays, 10AM–2PM GMT Saturdays*

Savings compared to buying 13 full-priced issues. You'll receive 13 issues in a year. You can write to us or call us to cancel your subscription within 14 days of purchase. Your subscription is for the minimum term specified and will expire at the end of the current term. Payment is non-refundable after the 14-day cancellation period unless exceptional circumstances apply. Your statutory rights are not affected. Prices correct at time of print and subject to change. *UK calls will cost the same as other standard fixed line numbers (starting 01 or 02) and are included as part of any inclusive or free minutes allowances (if offered by your phone tariff) For full terms and conditions please visit **http://bit.ly/magtandc**. Offer ends 31 March 2019.

SILVERJUKE

Create your own music jukebox

Jamie Munro shows you how to set up your own touch-screen jukebox using Ubuntu 18.04 and Silverjuke. Anyone got any requests?

OUR EXPERT

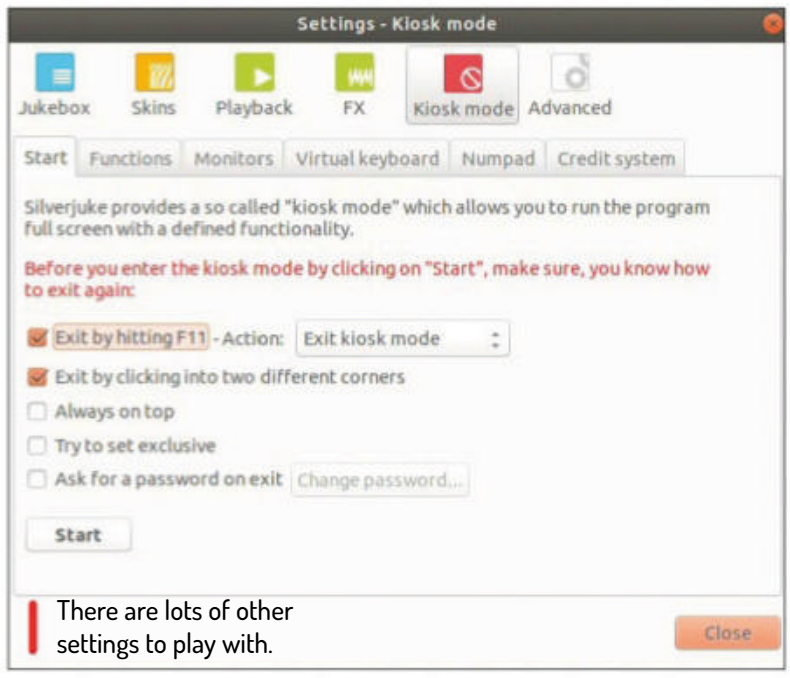
Jamie Munro was in charge of music at his own wedding. After a momentary lapse of reason and a short DuckDuckGo search later he was well on his way to a silver-tongued, all-singing-and-dancing, open-source solution.

This tutorial will show you how to install and configure *Silverjuke* for use on a touchscreen monitor. By the end of the tutorial you'll have a standalone jukebox that will boot automatically into *Silverjuke's* kiosk mode that doesn't require a keyboard or mouse to be connected.

We settled on Ubuntu 18.04 which supports the Intel NUC and Acer T232HL monitor we were using. *Silverjuke* is available in many different distros, and the source code is available on github. We used a network cable to connect, but if your device has Wi-Fi you can configure this when you install Ubuntu. You'll need a bootable Ubuntu USB key which you can download from **http://releases.ubuntu.com/18.04**. We opted for a minimal install with no additional drivers, and used the entire disk. Enter user and system names making a note of the password as you will need this later, and select the Login Automatically option as you will want to bootup, login and start *Silverjuke* with no intervention.

Once Ubuntu is installed open a terminal and run the following commands to update Ubuntu and install *Silverjuke*. When prompted, use the password you made a note of earlier.

```
$ sudo apt update && sudo apt install -y
$ sudo apt install silverjuke
```



There are lots of other settings to play with.

Silverjuke opens in windowed mode. We'll change this later but for now we'll add our music to the library and change some of *Silverjuke's* default settings to get a more robust jukebox experience. The Settings menu can be found under Edit>Settings and has six main areas, each with a number of sub-tabs. We'll run through the main options here. This is where you can

Batteries not included, please drink responsibly.

add sources to your music library and manage the Music library settings.

Because our NUC has limited hard drive capacity we're using an external drive to store our library. Depending on your setup you may have your files on a local drive, an external USB drive or on your network. *Silverjuke* enables you to add music from any of these options from the Add source button. You can set options for how *Silverjuke* scans and identifies your music files (for example, including zip files), but generally the defaults should work fine. Once you've added a source click Update music library to start scanning your source. This may take some time so we recommend making a cup of tea, as is tradition.

Select your settings

On the Skins tab there are six different skins to choose from (trying to find more online leads to a deleted forum) and while the Old-style Jukebox skin certainly looks the part we found some of the navigation buttons were too small to be useful. Silveriness Touched is simple with reasonably sized controls.

In the Fonts and covers tab we set the font size to 18 points, the column width to 400 and the cover size to 90 per cent. These settings made good use of our touch screen real estate and made selecting tracks by touch much more accurate. By default, there are no restrictions on adding tracks to the queue but it's sensible to prevent the same track being queued twice and prevent track repetitions within half an hour.

The Automatic control tab makes it possible to specify how long *Silverjuke* should wait before starting to play tracks if there is nothing in the queue. You can also customise how *Silverjuke* selects which tracks to play automatically. *Silverjuke* has a tool under Edit>Music selection, which can be used to create selections, for example 'Music from the 1970s' based on a date range. These saved selections can then be used as the basis for *Silverjuke* to automatically start playing tracks. This will come in very handy if your party has a 70s disco theme.

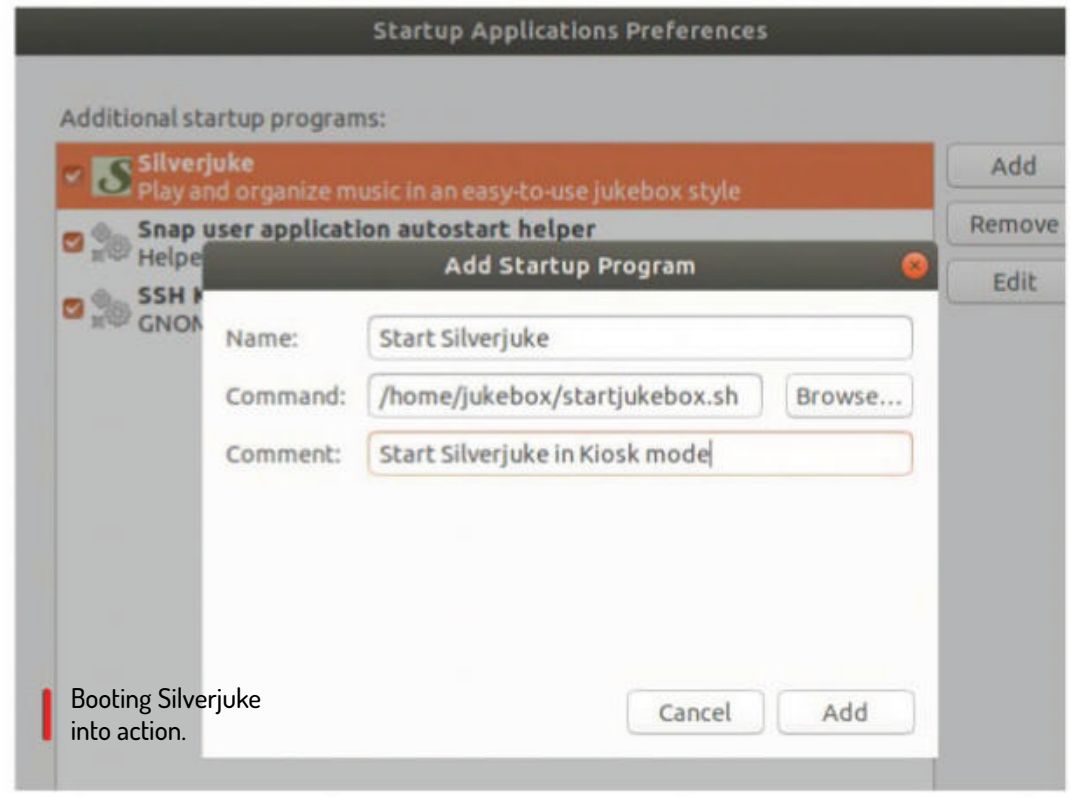
Kiosk mode forces *Silverjuke* into full screen mode. Unfortunately, there's no GUI setting to make *Silverjuke* start in Kiosk mode so we'll do this from a shell script. The Functions tab enables you to customise which functions you want to allow people to use when running in Kiosk mode. For example, you may not want guests to be able to adjust the volume or to edit the queue. You could also limit the available music selection to a pre-saved list. Again, useful for your disco party.

The other main setting for our touch screen jukebox is enabling a virtual keyboard, which makes it possible for people to search. When you enable this you'll also need to select the US International layout, because *Silverjuke* will default to Belgian French.

Starting up

The final step is to configure Ubuntu to start *Silverjuke* in Kiosk mode at every boot. This will allow for the keyboard and mouse to be removed and for the power button on the PC to be used to turn the jukebox on and off as required.

It's possible to force *Silverjuke* into kiosk mode by starting it from the terminal with the `--kiosk` switch.



Booting Silverjuke into action.

This means we can create a simple shell script using any text editor with the following lines.

```
$ #!/bin/bash
$ silverjuke --kiosk
```

You can run this on boot by pressing the 'super' key and searching for Startup applications or by running `gnome-session-properties` from the terminal. Then it's simply a matter of clicking Add, browsing to the `startjukebox.sh` script and adding a suitable name and description for your command.

To prevent our jukebox from going to sleep we'll disable some of Ubuntu's default power-saving options. These are easy to find in Ubuntu's settings under Power, where you can set the screen to never go blank, and to never suspend. Make sure that pressing the power off button is set to Power Off. Under Privacy you can turn off screen lock and under Notifications disable popup notifications.

You're all set. It's a good idea at this point to reboot the system and make sure everything comes up as expected before disconnecting the mouse and keyboard and heading to the bar. LXF

QUICK TIP

If you're using an external USB hard drive for your music library you may want to edit your computer's BIOS to make sure your internal hard drive has a higher boot priority than the external drive.

» COVER ME

Cover art is an essential part of any jukebox experience and your jukebox will look more authentic and be easier to navigate if your albums all have the correct covers. *Silverjuke* will recognise cover art if it's in the same folders as the music files and although it does have the capability to search and update covers from the internet it's not great for updating covers automatically or in bulk.

Searching for an album in *Silverjuke* will start a Google (or similar) search and you need to download any resulting images to the right folder. Not something you would want to do several hundred times. So if *Silverjuke* can't find a cover it will generate an Avatarish (think social media, not James Cameron) cover using the album title and artist name. Although *Silverjuke* will use different colours, if you have too many generated covers, your albums will all look the same and none will stand out. In this case it's a good idea to clean up your library and download new cover art using a tool like *MusicBrainz Picard* first and then add your music to *Silverjuke*.

SCRIBUS

Design better posters and book covers

Part Two! Catch up, go get the last issue on page 66!

John Knight plunges into the depths of Scribus once again – this time his mission is to try to make a decent magazine cover.

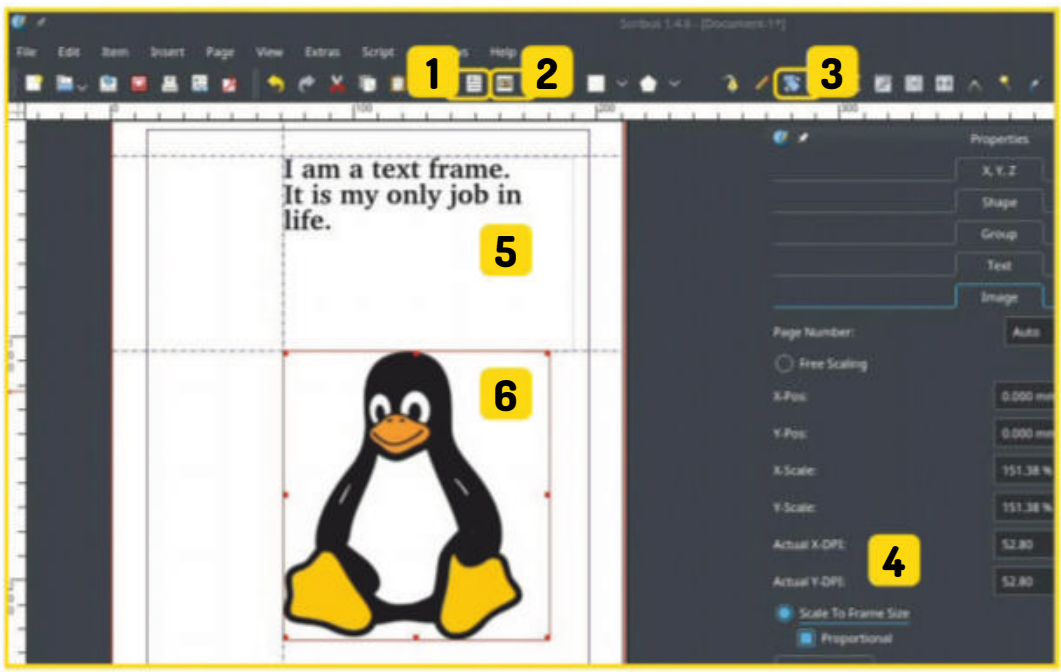
OUR EXPERT

John Knight writes ebooks on how to play the drums, when he's not playing with a Commodore 64 emulator.

For anyone who isn't familiar with it, *Scribus* is a free publishing application that gives ordinary users the power to create highly polished layouts and intricate designs. In this issue we'll be exploring its interface with a view to creating cool graphical layouts such as posters and book covers.

Note that as this is a standalone feature, we'll be covering a lot of the same ground as in **LXF245**, except this time with a greater emphasis on graphics. While you don't need to read last issue's article, each one has parts the other doesn't, and the two combined should complement each other nicely. Covering all of *Scribus* could fill an entire magazine, so rather than exploring it as a whole, we'll take you through enough of the interface to get you started.

Getting around Scribus



1 Insert Text Frame
Create freehand text boxes on the page in any dimensions you like, which you can then fill with text.

2 Insert Image Frame
Much the same procedure as with text frames but with images instead. Fancy layering options are available.

3 Rotate Item
Enables you to freely rotate any text or image frame.

4 Image Tab
The Image tab under Properties is where you should instantly head when an image is loaded, especially if you need to fix its proportions.

5 Text Frame
Required for inserting any text, which can be styled at will with the Story Editor.

6 Image Frame
A filled image frame, these can be resized at will and be re-formed into elaborate shapes.

Scribus should open straight to the New Document window where you define your layout; otherwise, select File>New. To keep things simple we'll stick with a Single Page layout and leave most of the defaults alone. Everything's pretty conventional except the choice of Default Unit, which is 'Points (pt)' – we suggest you switch to millimetres instead.

Admittedly *Scribus* can appear daunting at first because the interface may not be like anything you've used before, but ultimately it revolves around two chief concepts: frames and layers. Let's look at frames first.

Being framed

Put simply, frames are boxes drawn on screen that contain images or text, and can be easily moved or resized. In regards to text, they can be used for something smaller like magazine or newspaper pages, but they can also be used for creating big bold headings, such as on posters or covers.

To start creating, look in the middle of the main toolbar where you'll find the Insert Text Frame button. Click and place your first frame wherever you like on the page, with whatever dimensions you wish – you can even overlap the page margins if you want.

Note that *Scribus* switches back to Select Item mode after you've inserted a frame, so you'll need to click the button each time you want to make a new text frame. To write something, right-click in the frame and choose Edit Text. This opens the Story Editor window, which resembles a very basic word processor.

Although there are a few oddities here you should be able to guess most of the controls, though text formatting isn't really updated inside the Story Editor. It helps to move the window off to the side so you can see the text below, and to keep selecting Update Text Frame. When you've finished editing, click the big green tick button. If you want to move the text frame around, just click and drag inside the frame, or drag a corner to resize it.

Creating an image frame is essentially the same process as with text frames. Click the Insert Image Frame button in the main toolbar, then click and drag the dimensions of the frame to suit. As with text frames, you need to select Insert Image Frame each time you want to make a new one.

Right-click inside the frame and choose Get Image which opens the file browser, then select your file. The image will now be in your frame, but unless the frame's aspect ratio was the same as that of the image, the dimensions will be wonky – with some white space left in the frame, or with only part of the image showing.

To alter an image's dimensions, right-click the frame and choose Properties. In the Properties window, click the Image tab – the option Proportional is probably enabled. If so, one possible fix for a badly filled frame is to untick the Proportional box and tick Scale To Frame Size, which will then stretch the image to fit the frame.

If you leave Proportional unticked, the image will be stretched to fill the dimensions of the frame. This may look horrid, so you can fine-tune the image by clicking the Free Scaling option, where you can move and stretch the image to your heart's content.

However, if Proportional is ticked, the image will maintain its original aspect ratio and there will likely be some leftover bits of frame, meaning you will need to move the borders back manually to suit the image. If you're resizing an image frame but want to keep it in proportion while doing so, hold Ctrl+Alt while you click and drag one of the borders.

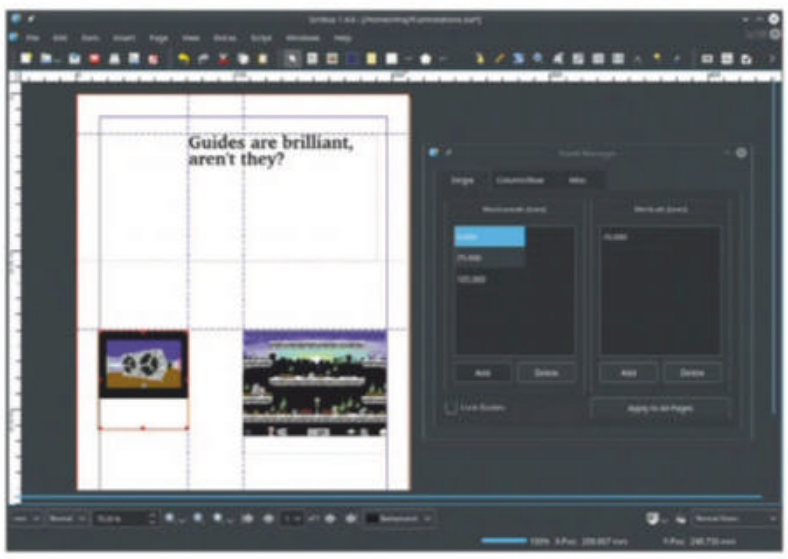
If you want to get gaudy with your creations *Scribus* has your new best friend: a rotate button! To use it, first select a frame then click the Rotate Item button in the main toolbar. Then just click and drag on the frame and it will rotate with the mouse.

Note that for things like clipart and small logos, you probably want transparent backgrounds. Transparency allows other things to sit behind the main object of an image, and won't leave clashing borders or colours. They also help when you want to have text flow around a frame *(see Transparency With Gimp box on page 72 for more)*.

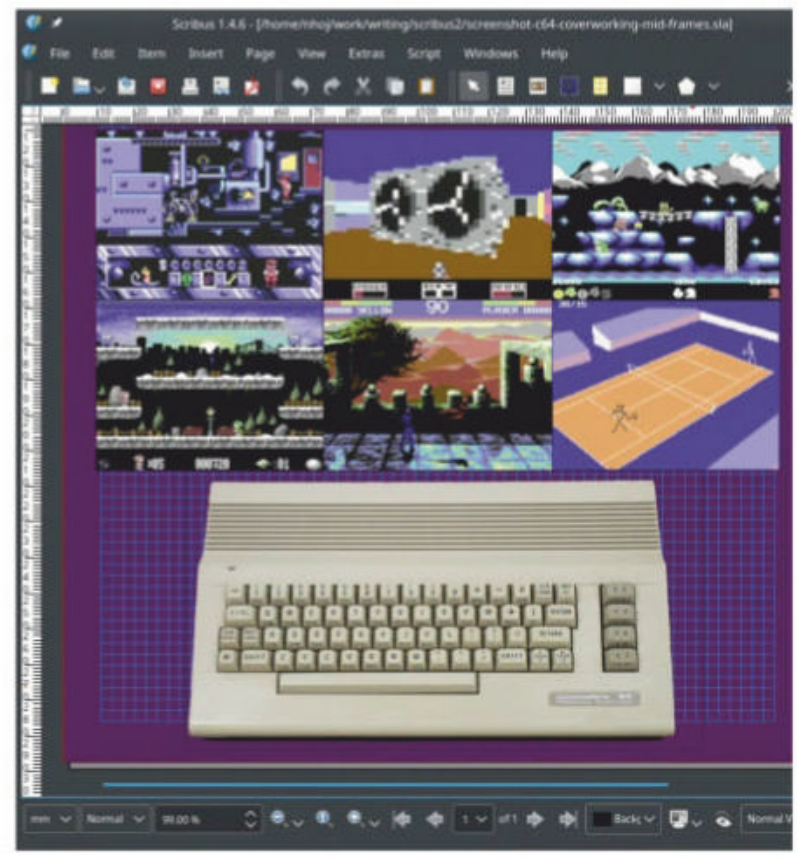
Total layers

Once you start getting into layers *Scribus* really gets interesting. Layers enable you to work on different parts of a scene independently, allowing for complex imagery. For instance, you can have a fixed background image with many separate pictures placed on top – which by the way is a primary technique in animation.

In fact, if you think of an animator with celluloid frames stacking individual drawings over backgrounds, it's the same concept. Let's say you're working with an image and a bold text heading. You want one frame to



Using Guides, you can draw, well, guidelines across your page that can both visually guide you and have frames 'snap' to them.



Having a grid background that other objects bleed over creates a feeling of jumping off the page.

overlap the other, but right now they're sitting the wrong way around. You can fix this by either right-clicking whichever frame is on top and choosing Level>Lower, or selecting Level>Raise from the frame that is underneath. (Note that *Scribus* can have a lot of layers and you may need to do this multiple times).

If you want to move one layer back behind another, you can always choose Level>Lower again, but it's probably easier to cycle through the layers. If you hold Ctrl while you click on a layer it will bring up whatever is underneath it, and if you keep Ctrl+clicking, it will cycle through however many layers you may have.

If you want to work with a background image, counterintuitively it may be easier to place this last. If you position all of your primary visual elements first in their respective frames, it will be one less layer to interfere with your layer controls.

Once you have all the main frames sitting nicely, you can simply draw an image frame over the entire page. Right-click somewhere on its frame and choose Level >Lower To Bottom and all of your initial frames will reappear, now over a background. Conversely, you can bring any image you want to the foreground by right-clicking and choosing Level>Raise To Top.

Using Guides

If your scribblings look amateurish at best, a great way to improve is to find a design you want to imitate and measure everything. Designers follow visual guidelines such as 'golden ratios' and 'rule of thirds', and these deliberate proportions are the key to attractive form, making things feel right in ways you can't explain.

Doubtless you can conjure images of designers and architects with ruled lines streaked across their creations. *Scribus* is made to do the same thing. If you physically measure the features of a design you want to imitate, guides can help you replicate their placements. And don't worry, they won't appear in the final print!

Open the Guide Manager by going to the main menu and choosing Page>Manage Guides. The Guide Manager has three tabs: Single, Column/Row, and Misc. Single is for placing individual lines across the page according to your own measurements. Column/Row

QUICK TIP

If you look in the Shape field at the top of the Shape tab and click the white box, you can choose from a number of preset shapes for your image frames, including our old friend Tux!

QUICK TIP

If you have the Guide Manager window off to the side, any changes you make in the Guide Manager will immediately take place on the page in the main window below, saving you a lot of time in the long run.

QUICK TIP

Scribus **enables you to place page margins and images wherever you like, but average home printers won't go all the way to the edge. Full-page printing may be fine for professional machines, but check your printer's manual before you commit.**

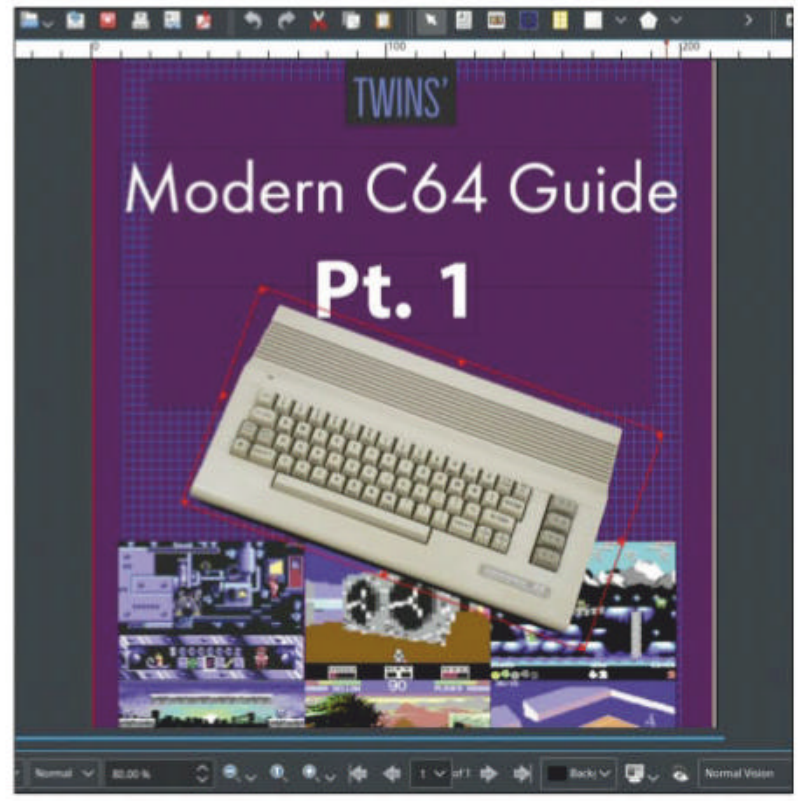
enables you to place a series of lines over the page in evenly spaced intervals, and Misc has buttons to delete the lot and start over.

To create your first guide, open the Single tab and you will see two fields: Horizontals and Verticals. To place horizontal guides across the page, click Add and enter the number of millimetres you want the guide to be from the top of the page. Press Enter and a black dotted line will be drawn across the page. The Verticals field does the same thing, but with vertical lines, starting from the left of the page.

If you would rather generate a series of evenly spaced lines, open the Column/Row tab. Once again it is split into Horizontals and Verticals, this time with a choice of how many lines you want to draw across the screen. The field 'Refer to' has the options of Page and Margins. Page will have the lines start from the actual edge of the 'paper', whereas Margins will have the measurements begin from the page's margins instead.

Guides are more than just a useful visual reference. If you enable Snap To Guides (Main Menu>Page>Snap To Guides), whenever a frame is in close proximity it will lock itself to the guide. This can save a lot of time otherwise spent placing frames carefully by hand, and if you're going to make a lot of designs with the same proportions, it's an excellent way to make a template.

As with layers, this will work best with images that have transparency enabled. This usually involves a primary object against a clear background. Without transparency you will probably be stuck with ugly bits of empty frame ruining your design, when what you really want is a solid object that text can flow cleanly around.



With the rotate button and some gaudy layering, you can make some truly disgusting eyesores such as this one!

Go with the flow

To recap on some basics, first create a text frame that will be big enough to wrap around your image. Now create an image frame and make sure its level is above the text frame (Level>Raise To Top), not forgetting to make sure it's properly scaled and proportioned. Initially the text will simply run underneath your image frame, so in order to make it flow around the image, right-click the image frame, choose Properties, and open the Shape tab.

Looking at the field Text Flow Around Frame, we'll focus on four options: Disabled, Use Frame Shape, Use Bounding Box, and Use Contour Line. Disabled turns off the feature entirely and text will just overlap the image,

Using indivdual guides, generated guides and custom margins, we were able to lock images in place with an even grid.

» TRANSPARENCY WITH GIMP

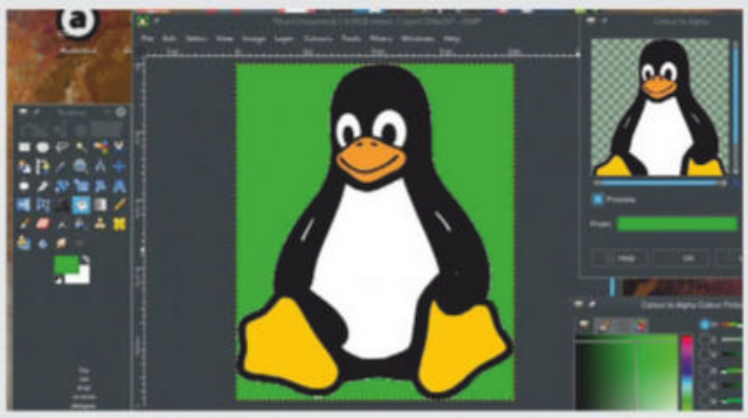
Scribus outsources its imaging editing to *GIMP*, so we may as well use it for transparency. First, make sure your image has a solid background colour that can be easily differentiated by the computer from the rest of the image. White is good, but green is a common choice for its use in digital effects.

Open your image in *GIMP*, and from the main menu choose Colours>Colour to Alpha, then click OK. A new window will appear with an image preview showing whatever *GIMP* has autodetected. If you don't have the option to choose Colour to Alpha, change the image to RGB mode using Image>Mode>RGB and repeat.

It will attempt to detect everything automatically and separate the object from the background. If it can't, you'll need to use the colour picking tool to give it a hand. Or if that's still not working, using either the Intelligent Scissors or perhaps the Free Select Tool. Use the Path tool for a selection you can edit, it's tricky but worthwhile to learn.

This will likely have your chief foreground object selected, but it's the background you want to clear. Use Select>Invert and this will change the selection to the background instead. Press Delete and you should now have a grey chequered background, indicating your chief object is above a layer of transparency.

Save your image as either a PNG or GIF, as JPEG files do not have the ability to have background (alpha channels) transparency.



A solid-coloured background in an image can easily be removed automatically.

ignoring its shape altogether. The other three options will all make the text flow around the image frame, but until you start making tweaks to the image's shape, all three will have exactly the same effect.

Use Bounding Box bases the text flow around the original rectangle of the image frame, meaning the text will always conform to a perfectly rectangular field around the image. This can leave big empty gaps, but may be just the look some people are going for.

However, Use Frame Shape makes your text adhere to whatever shape you have given the frame. If you look in the 'Shape:' field at the top of the tab, you can choose from a number of shape presets, or if you click the Edit button you can make the shape just about anything you like using some fairly intricate controls, which is great for awkwardly shaped images.

For most people Use Frame Shape will be fine. For more advanced designers, Use Contour Line provides another layer of functionality by adding a second shape for the text to flow around, independent of whatever form the frame shape has been hacked into. This allows you to create elaborate frame shapes while making another shape for text to flow around.

Scribe this!

Scribus may be initially intimidating, but that's only because most people using it won't have any reference points. If you ever dabbled with early desktop publishing software, or have any experience with commercial layout tools such as *InDesign*, things won't seem nearly so complicated. There are some oddities and irritations here and there, but once you've got the hang of the basics it's really easy to get into a groove.

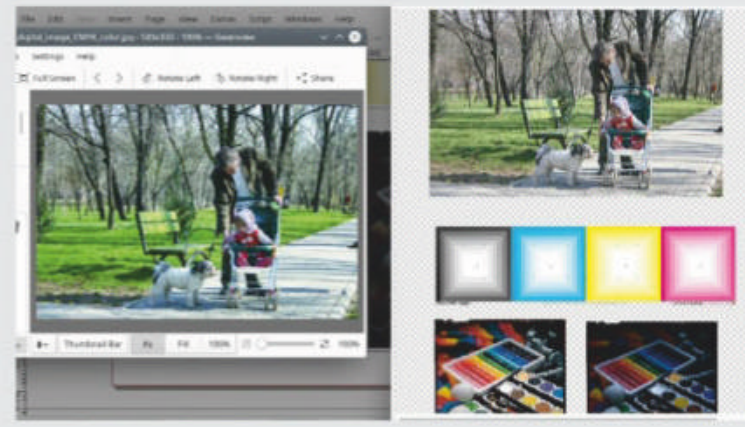
Bits of *Scribus* exist in other programs, sure, but something that may take hours in an application like *GIMP* – to which *Scribus* delegates its image editing anyway – may only take a few minutes in *Scribus*, and there's no reason you can't use both programs and combine their results.

DEALING WITH CMYK

Although your computer screen uses a light-based colour system called RGB (Red Green Blue), printers use a system called CMYK (Cyan Magenta Yellow and 'blacK'), which builds a composite image of several colour layers painted one over the other. If your design is just meant for screens or home printing you can ignore CMYK, but if your image is for mass printing, print houses may require you use it.

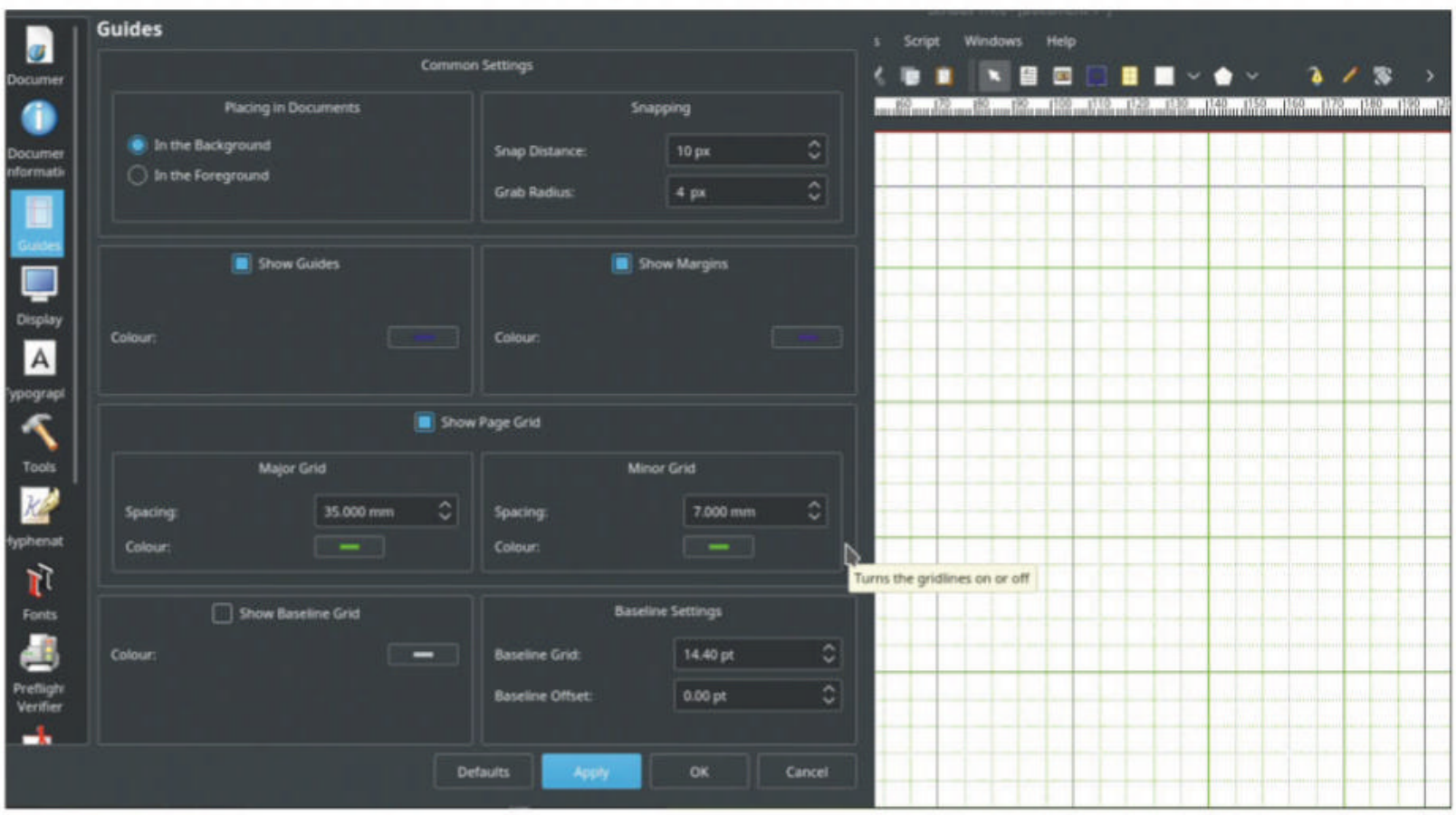
Conversion can be a tricky process. CMYK output usually looks duller than RGB, meaning the colours will need to be enriched to compensate – CMYK images often look grossly saturated on a normal PC! Graphic artists have complex setups to colour-match what's on screen with what's on paper. For hobbyists it's much easier to just use an online converter such as **www.rgb2cmyk.org**, where you can upload your image and choose from a number of standard colour profile presets.

In Print Preview, you can enable Display CMYK to enable the field Separation Name, where you can enable or disable individual CMYK layers for complex printing tweaks. *Scribus* has more advanced printing emulation options, such as Under Colour Removal and Convert Spot Colours. Because of RGB and CMYK, Print Preview and PDF exports will invariably look different, so always do test prints.



On the left is a CMYK image in an RGB viewer, with the CMYK Print Preview on the right. Note the other examples with CMYK preparation versus non-prepared.

Ultimately, if you're willing to learn just a few basics then *Scribus* is one of those classic tools that will remain permanently in your Linux toolbox. LXF



Scribus cleverly works out its scaling so that 1mm on screen equals 1mm in real life. Check out the Grid system to really exploit it.

Part Two!
Don't miss next issue, subscribe on page 28!

EMACS

Extending your editor to email and video

Carrying on from last month's Emacs instalment, **Mats Tage Axelsson**, turns his attention to how the editor handles email, video editing and more.

OUR EXPERT

Mats Tage Axelsson has been using Linux since it was available on floppy discs.

Once you get used to *Emacs*, you won't want to leave. Well, at least that's what uber-geeks think. The thought isn't totally nonsensical, though, because most people use a range of applications. Whatever you're writing, be it code or documents, *Emacs* can be used as a way to reduce distractions while you tackle the task at hand. From experience, if you start switching to work in other applications, your attention may start to wander.

Why not consider using *Emacs* for your email, at least when you're in the middle of something? Another advantage with this approach is that you can send what you're already writing directly from inside *Emacs*. Cut and paste will be less hassle. As you'll see, you're also able to format your emails.

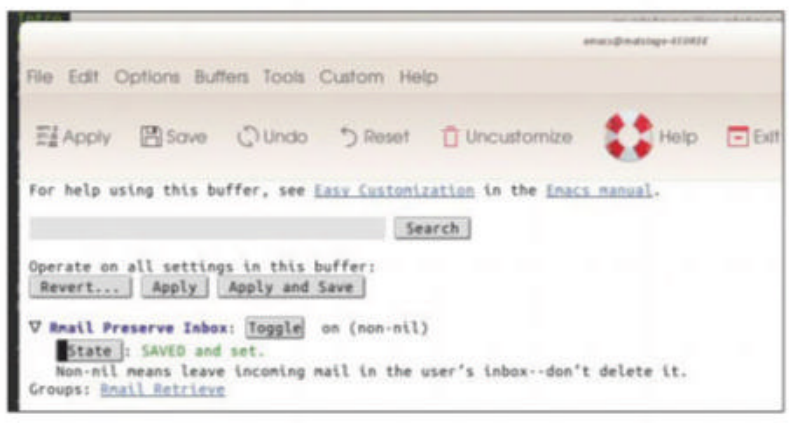
There are a few more points to consider. One is whether to use *Emacs* for connecting to your mail server. When you're fetching emails, *Emacs* will stop working. For this reason, it's a good idea to employ external tools to receive any emails while you use *Emacs* to read and write. You can also use *Emacs* for video-editing, although whether it's a fast and efficient option is another matter altogether.

Rmail is the default email reader for *Emacs*. It supports POP3 and can handle newsgroups, too. As a matter of fact, *Rmail* has most of the features available like many other packages, but you'll need to collect them yourself. The configuration is simple, and usually it just works. This is Richard Stallman's favourite, or at least he's stated that he'll consider it important even though he doesn't know how many users have it.

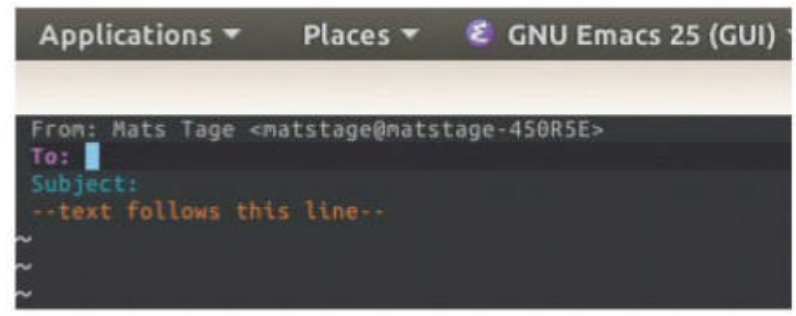
Gnus combines email and news reader functions. It was initially used as a newsreader, but the program has been developed to support RSS and email. With the

QUICK TIP

Export your document to a blog using the org-export-to blog feature. You can write to wordpress, using the org2blog feature in Emacs, and with this feature you can log into wordpress and your own blog.



This is the most important parameter, set this value to non-nil to ensure you do not lose your emails.



When you start composing an email you reach the compose mode and its simple interface. Don't be fooled – many features await you!

correct settings, you can read, edit and send your emails from within *Emacs*. A word of warning: you must get used to thinking of your emails as news. This aspect will not change, so get used to it or use one of the other packages.

The email alternatives.

There are several alternatives available for reading and sending emails in *Emacs*. Your choice will depend on your current system, your requirements and how important it is for you to have either speed or low storage space. Some clients store more emails so they're likely to be faster, but this isn't always desirable. When reading emails in *Emacs*, you'll just want to stay with your editor without switching over to a separate email client, but there are other reasons to use email directly in *Emacs*. One is to be able to copy and paste, or even run scripts that create the email.

Before you start using email with *Emacs*, you need to decide how to fetch and send your emails. First, decide on the protocol: either POP3 or IMAP4. You must also make sure you know what the system does with your emails on the server.

Before you do anything, set **rmail-preserve-inbox** to **t** or **vm-pop-expunge-after-retrieving** to **nil**. This is especially for using POP3. Oddly enough, the standard mode is to delete everything on your email server.

There are many options for fetching your emails. The easiest method is to use separate client software, most commonly either *offlineimap* or *mbsync*. For *Emacs*, the standard email reader is *rmail*. The most basic use of *rmail* is to read your mail spool directories. You can also instruct *rmail to* connect directly over the network, but this will usually result in *Emacs* being unavailable during

the update procedure. This will impede your writing, assuming you only use *Emacs* for your work.

Many packages remove files from the server when they fetch them. Make sure you stop that from happening. Another great package that helps is the *offlineimap* package. You can use it to read your emails and store them locally, or sync to your local file system. You can set up *offlineimap* in the corresponding **.offlinemaprc** file, but that's a little cumbersome and error-prone. A better alternative is to use *mutt-wizard* to set your accounts up. Combining this with *mu4e* is also useful. To install *mu4e*, use `apt` as usual:

```
$ sudo apt install mu4e
```

This is just the beginning. *Mu4e* is part of *mu*, but if you look through the packages in Melpa you won't find *mu* or *mu4e*. What you will find are extensions to the suite that meets special requirements, such as if you come from *vi* and use evil mode. Using any of these tools may also fail even after this point because the load file isn't available. Here's how to set the load-path, using one of your configuration files:

```
(add-to-list 'load-path "/usr/share/Emacs/site-lisp/mu4e/")
```

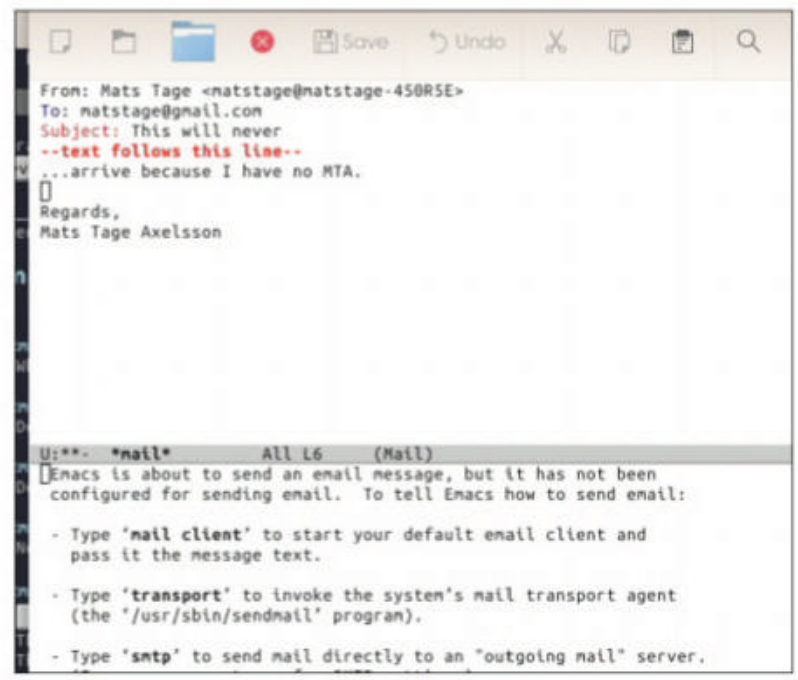
The path will usually be the same on all systems, but if you have a problem use the `find` command to figure out where it is. You should be able to read the mail directory that's specific for your system. The usual place is in your home directory as a hidden file: **~/.mail/** or **~/.Mail/**. The location is configured by your mail fetching agent. Many users have reported that they find *offlineimap* to be slow, so they're using *mbsync* instead. This package is included in *isync* when you install it using your package manager.

You can also configure your system for several accounts. To do this you'll have to add some more code to your configuration. The first thing is a list of your accounts; the code below is an example of how to do it:

```
(defvar my-mu4e-account-alist
'(("yahoo"
 (mu4e-sent-folder "/yahoo/Sent")
 (mu4e-drafts-folder "/yahoo/Drafts")
 (user-mail-address "matstage@yahoo.com")
 (smtpmail-default-smtp-server "smtp.yahoo.com")
 (smtpmail-local-domain "mail.yahoo.com")
 (smtpmail-smtp-user "matstage")
 (smtpmail-smtp-server "smtp.yahoo.com")
 (smtpmail-stream-type starttls)
 (smtpmail-smtp-service 25))
("abc"
 (mu4e-sent-folder "/abc/Sent")
 (mu4e-drafts-folder "/abc/Drafts")
 (user-mail-address "taggen@abc.se")
 (smtpmail-default-smtp-server "smtp.abc.se")
 (smtpmail-local-domain "abc.se")
 (smtpmail-smtp-user "m9779")
 (smtpmail-smtp-server "smtp.abc.se")
 (smtpmail-stream-type starttls)
 (smtpmail-smtp-service 587))))
```

Once you have those things set up, it needs to be called. In this case, we create a dialog that asks which account you want to send from:

```
(defun my-mu4e-set-account ()
"Set the account for composing a message."
(let* ((account
    (if mu4e-compose-parent-message
```



If you start writing an email without setting up your account, *Emacs* will ask you for the parameters while displaying the manual.

```
    (let ((maildir (mu4e-message-field mu4e-compose-parent-message :maildir)))
      (string-match "/\\(.*?\\)/" maildir)
      (match-string 1 maildir))
    (completing-read (format "Compose with account: (%s) "
                             (mapconcat #'(lambda (var) (car var))
                                        my-mu4e-account-alist "/"))
                     (mapcar #'(lambda (var) (car var))
                             my-mu4e-account-alist)
                     nil t nil nil (caar my-mu4e-account-alist))))
  (account-vars (cdr (assoc account my-mu4e-account-alist))))
 (if account-vars
  (mapc #'(lambda (var)
            (set (car var) (cadr var)))
        account-vars)
  (error "No email account found"))))
```

For this code to run you add a hook for when you want to send an email. This results in *Emacs* asking you which account to use before you send:

```
(add-hook 'mu4e-compose-pre-hook 'my-mu4e-set-account)
```

QUICK TIP

You can also read Twitter with twittering mode, the code is available on MELPA and as a GitHub repository. You can read and send tweets, there is even an icon mode so you can see those.

» PRIMER: SPACEMACS LAYERS

Because *Emacs* has so many options, enthusiasts have created so-called distributions that set everything up the way they like it. *SpacEmacs* is such a distribution but with added flexibility. One of the features of *SpacEmacs* is configuration layers. Each layer is used for a specific specialisation, usually a programming language. An example is HTML and JavaScript: they're the first entries in the default configuration file **.spacEmacs**, where the layers are defined. That definition is done in the variable **dotspacEmacs-configuration-layers**. If you want to tweak your setup, you have two options. One is to add variables in this file and the other is to create your own layer.

Experienced users of *Emacs* might wonder why *SpacEmacs* users don't install packages using MELPA? The reason is that *SpacEmacs* will consider your packages unnecessary at the next restart and remove them. Your own variables have their own space in the file **dotspacEmacs/user-config**. When you have many changes to make and also the packages you need, the better choice is to create a layer. Just add it to the *spacEmacs* config file and put your code in the private directory with the matching directory name.

QUICK TIP

There are many parts that are set and ready in SpacEmacs, but if you feel lost without having set it yourself, make sure you figure out Configuration Layers if you're going to use it.

When *mu4e* is set up, you start in the main mode and this is where you start your journey. The options are mostly different search options conveniently presented on this page. You choose the four main ones with the mnemonic [bu], where u means unread, [bt] means today's, [bw] means the week's and [bp] means pictures. You can also configure your own search. Apart from searching for specific files, it can also update your local mail. This is what you configured earlier when you picked either *offlineimap* or *mbsync*.

Once you've carried out a search, you end up in the headers view where all the matching emails show up. Here you can manage the email using d to mark for deletion and r to archive.

You can also view the mails by pressing Enter. The emails are rendered fairly well – while HTML support is there, you don't always get the correct view.

If it's important to see graphics, then you need to set your values for allowing inline images in your view:

```
;; enable inline images
(setq mu4e-view-show-images t)
;; use imagemagick, if available
(when (fboundp 'imagemagick-register-types)
  (imagemagick-register-types))
```

You'll also want to set a large number of other parameters. For reading more advanced emails, you can obtain better results using *pandoc*. To define this you need the following code in your **init.** file:

```
(setq mu4e-html2text-command "iconv -c -t utf-8 |
pandoc -f html -t plain")
```

This is the main menu when using mu4e. From here you can start a search, jump to other mailboxes and update from the server.

With this in your settings, you can now view HTML messages with the pictures.

A powerful way to combine tools is to integrate your emails with org-mode. This can be useful in many ways, one is to be able to add tables in your emails. The other is to read an existing email and directly make it a TODO item in your main **TODO.org** file. To do this, set up the connections in your config:

```
(require 'org-mu4e)
(define-key mu4e-headers-mode-map (kbd "C-c c")
'org-mu4e-store-and-capture)
(define-key mu4e-view-mode-map    (kbd "C-c c") 'org-
mu4e-store-and-capture)
;;store org-mode links to messages
;;store link to message if in header view, not to header
query
(setq org-mu4e-link-query-in-headers-mode nil)
```

With this in your settings, the keys are available in headers-mode and view-mode so you can plan immediately. Of course, email isn't your to-do list, and you can only do five things to an email: delete, read and delete, read and archive, reply and archive and finally add to the to-do list and archive. With this system in your mind and a great search function, you can also skip all file folders and learn how to use *Mu4e's* search functions more effectively.

To make the to-do list a real agenda, you'll need to set a date and time, including a priority for the entry. The best way to do this is to create a template for the item. Here's an example of how to do that:

```
(setq org-capture-templates
   '(("t" "todo" entry (file+headline "~/todo.org"
"Tasks")
     "* TODO [#A] %?\nSCHEDULED: %(org-insert-
time-stamp (org-read-date nil t \"+0d\"))\n%a\n")))
```

The template takes the headline of your email and creates the to-do item with those values and sets the current date. When you enter the agenda in org-mode you can then tweak the values the same way as when you're editing it directly.

Twitter in Emacs

You can also tweet from *Emacs*. At first glance, it seems complicated but the designer has made a great job in paving the way for you. The package is **twittering-mode** and is available on MELPA. The install is the usual `package-install` or use the package list and type I next to it and x to execute.

The next issue that's fantastically simple is authorisation. The first time you run Twitter, it takes you directly to the Twitter front page where you log in as usual and then get a PIN code. You return to *Emacs* and type in the PIN and you're immediately in your timeline.

In your timeline view, you use some of the common keys to move around. Pressing J goes to the next tweet, K to the previous, while P takes you to the next tweet of the same author.

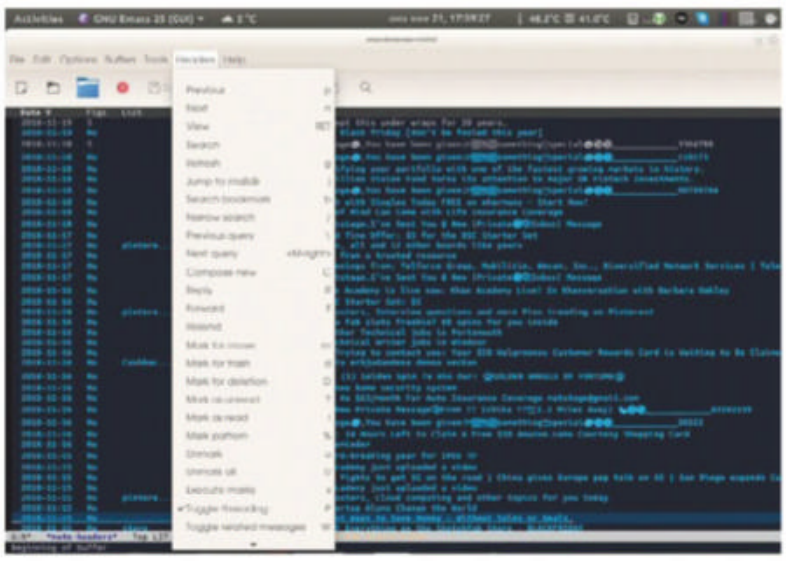
Tweeting, replying and retweeting is also supported. Press U for a spontaneous tweet, and to interact with someone else's tweet go to that tweet and hit Enter. A new buffer shows up where you type your tweet. C-c C-c sends and you can also cancel after you've started. You'll probably write a few tweets by mistakes while getting the hang of it so delete your own tweets with C-c D. To pick a timeline, press V and type in your hashtag.

» KEEPING EMACS SIMPLE

Emacs has so many settings and functions that it can get a little overwhelming. Sometimes all you want is a limited range of functions that are suitable for the task in hand. To work with an *Emacs* client that does very little, you need to launch it with only what you need. But then you have an *Emacs* session you'll need to start all over again.

A better solution is to have flexible setups and be able to quickly switch between them. CHEmacs is a script that you can download from GitHub and install in your home directory. It replaces your **.Emacs** file and is run before you start *Emacs*. Because it's run before *Emacs* itself, you can't download it from MELPA or any other repository. The setup is fairly simple: after you've downloaded the repository, you need to create a file with the names and paths in elisp format. With this setup, you can also have different *Emacs* setups running in parallel.

The best use of this package is to try out other people's configurations and different distributions. *SpacEmacs* is a great example, because getting used to handling your own preferences takes some time to get used to. Another distribution you can use is *Doom-Emacs*, together with an *evil* plus *Emacs* version.



While in headers mode you have all the functions available in a drop-down menu if you're using X*Emacs*. This can be turned off, though.

Video editing in a text editor sounds bizarre, but it's definitely possible. The *GNEVE* package enables you to edit your video in EDL mode. It's supported by the MLT framework, which *Shotcut* also uses. The editing process takes place with your video in the background with your notes for the timestamps on top. You can change this setting in the tool.

To set up the tools, put the **gneve.el** file in your load path and add the code to your configuration file. You need to have a Miracle Server running on port 5250. You access this server from **www.mltframework.org**, which redirects to Github. It also requires *Mplayer* for the playback. As soon as you have the server, start *gneve* with `M-x gneve-mode`. Find a file by pressing V.

While playing, you can play normally, skip forward by frame, by second or by five seconds. To prepare to edit you mark the start (E) and end (R) of a section through the video, after that you play the result ([) and check your work. The package is fine for removing bad takes and making the flow correct. But as the designers rightfully say, it needs support for transitions and so if you're serious about videos you'll need to use something else or add such functionality yourself.

Blogging with Emacs

Emacs has several modes to publish to blogging platforms, and org2blog is a prominent one. Others have support for Jekyll and there are even several ones using `xml-rpc`. To blog with *Emacs*, you still need to use some patchy software, but it's far from complicated. The mode file is named **org2blog**, and it's suitable for blogs using Wordpress. This mode exists on MELPA, but look out for the instructions on GitHub so you don't trip over the `xml-rpc` bug. The install is otherwise the usual routine: you need to place the **org2blog** file in your load path and load it in your **init.el** file. In addition, find the **htmlize.el** file from GitHub and set that in your **init.el** file. The basic code is as follows:

```
(require 'auth-source) ;; or nothing if already in the load-path

(let (credentials)
  ;; only required if your auth file isn't already in the list of auth-sources
  (add-to-list 'auth-sources "~/.netrc")
  (setq credentials (auth-source-user-and-password "myblog"))
(setq org2blog/wp-blog-alist
     '(("wordpress"
       :url "https://username.wordpress.com/xmlrpc.php"
       :username ,(car credentials)
       :password ,(cadr credentials)
       :default-title "Hello World"
       :default-categories ("org2blog" "*Emacs*" "vimgolf")
       :tags-as-categories nil)
      ("Blog-o-matic"
       :url "http://username.server.com/xmlrpc.php"
       :username "admin")))
```

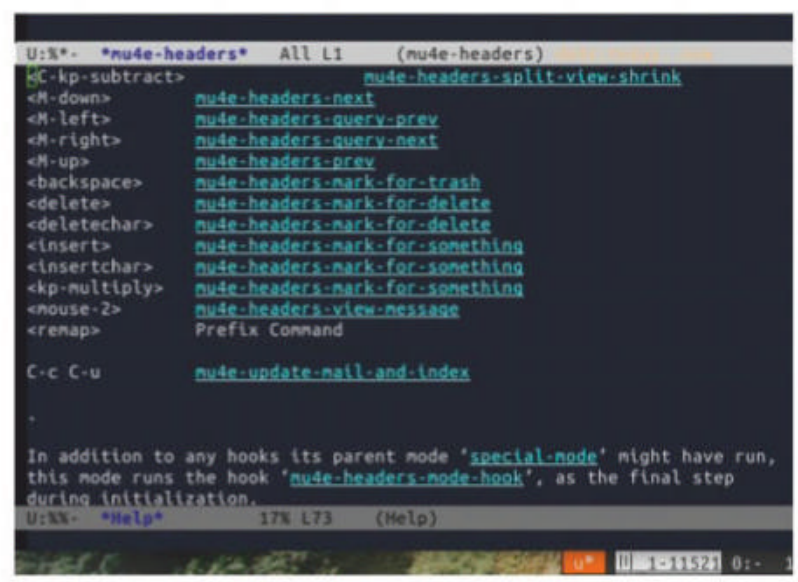
The two entries in this list are for **wordpress.com** and a private server on your own domain. To secure your password, use **.netrc** to store your passwords out of your *Emacs* configuration files. After that's set up you can start blogging. If you have special tastes, set your parameters with `M-x customize-group org2blog/wp`.

Get started by logging with `M-x org2blog/wp-login`. The function will read the list you defined earlier and enable you to pick one of the blogs to use.

Time for your first post. Using `M-x org2blog/wp-new-entry`, you make a list of tags and categories by using comma separation. You can also set post templates so that the date is set immediately and the post gets a standard title. Posting is a simple keystroke away: C-c M-p and after that, press p for publish, d for draft, D for page draft or P for post as a page.

This article has covered a lot of possibilities available with *Emacs* and this is just scratching the surface. With elisp available for adapting to web APIs, the uses of *Emacs* as a tool for your programming and other text-based activities are endless. *Emacs* won't always be the best tool for the job, but if you're already familiar with it, creating a mode to handle your requirements is usually straightforward. If you train yourself in *Emacs* you'll have a powerful tool for to writing documents and blogs at your fingertips. You can even use it for communicating with the world in a more focused way than a full web-page ever could – no more distracting pop-ups!

Even if you're using other tools for development, it's worth knowing how to use *Emacs* since it's almost always available to you. If nothing else, use it as an organiser for your work and hobbies. This makes *Emacs* as relevant as ever. LXF



There's help available for beginners. Use C-h to display local help in the lower part of the screen.

GRUB

Build a GRUBby USB flash drive

Filthy sticks? Never! A bootable USB stick is a must-have for the techie toolbox, clean-living **John Lane** makes one with Grub and a bunch of ISOs...

OUR EXPERT

John Lane is a Linux solutions specialist who never leaves USB sticks lying around.

QUICK TIP

Mebibyte partition alignment is recommended to optimise filesystem performance.

QUICK TIP

The latest Ext4 includes features that break backward compatibility. Consider disabling 64bit and metadata_csum.

The USB stick has transformed our concept of disposable media. Gone are the piles of floppy disks, CDs and DVDs. Wherever there's a need for ad-hoc storage we now reach for the essential tool that USB storage has become. We use them to "netwalk" files when we find ourselves without a network connection. They're invaluable as boot, recovery and install media and are a portable home for personal files.

In this tutorial we'll create a versatile USB stick you can use as an emergency boot device, to boot live or installation ISO images, and as a data store – with and without encryption. It'll work with either BIOS or UEFI while also making data accessible to macOS, Windows or even a smart TV or camera. We'll show how to boot popular live distros including Tails Linux, for which we'll configure the encrypted space as persistent storage.

Begin by collecting the tools for the job: the GRUB bootloader (version 2.x, not the older legacy versions), the command-line *gdisk* and *sgdisk* partitioning tools, *cryptsetup* for encryption and the file system utilities for VFAT and Ext4. You should verify you have the necessary commands available (most distributions include them):

```
hash grub-install {s,}gdisk mkfs.{vfat,ext4} {e2,dosfs}
label cryptsetup
```

No output means you're good to go; errors indicate any tools you need to install. Some distributions may use different names (Fedora uses *grub2-install*). The needed packages are *grub* or *grub2-tools*, *cryptsetup*, *gdisk* or *gptfdisk*, *dosfstools* and *e2fsprogs*. We'll assume you have a *Bash* shell terminal and can issue privileged commands either using sudo or as root.

You'll need a spare USB drive – the bigger the better, at least 16GiB to be useful! Plug it in and take a moment to identify its device path – you can use *dmesg* and `fdisk -l` to help with this. You should substitute the **/dev/sdX** in the examples that follow with your actual device path. Note: this may change each time it's plugged in!

We'll assume you know your device, that you'll substitute it in the commands that follow, and that you've taken care to make sure it's right – otherwise it is very easy to overwrite the wrong device.

Hybrid partition magic

Maximising portability requires understanding formats from the lowest level, so we begin with partitioning and

```
Main partition table begins at sector 2 and ends at sector 33
First usable sector is 34, last usable sector is 61056030
Partitions will be aligned on 2-sector boundaries
Total free space is 0 sectors (0 bytes)

Number  Start (sector)    End (sector)  Size       Code  Name
   1              34            2047   1007.0 KiB  EF02  BIOS Boot Partition
   2            2048          104447   50.0 MiB    EF00  EFI System Partition
   3          104448        12582911   6.0 GiB     8300  boot
   4        12582912        36818943   11.6 GiB    0700  DATA
   5        36818944        61056030   11.6 GiB    8301  TailsData

Expert command (? for help): o

Disk size is 61056064 sectors (29.1 GiB)
MBR disk identifier: 0x00000000
MBR partitions:

Number  Boot  Start Sector   End Sector   Status      Code
   1             12582912     36818943   primary     0x0C
   2                    1     12582911   primary     0xEE
   4             36818944     61056063   primary     0xEE

Expert command (? for help):
```

Smart formatting and partitioning will ensure that your USB stick will work just about anywhere – and on any device or OS.

the need to support two schemes: the contemporary GUID Partition Table (GPT) and the legacy Master Boot Record (MBR or MS-DOS) partitioning. We use a hybrid scheme which enables both GPT and MBR to co-exist.

We need five partitions to cater for the use cases we want to support: Partitions for BIOS and GPT booting, one to hold boot files and ISO images, plus plain-text and encrypted data partitions. The plain-text data partition will also be exposed to systems that only understand MBR, such as Windows prior to version 8.

You can use shell (e.g. *Bash*) variables to help with sizing those partitions, starting with the stick's device node and its capacity as reported by the kernel:

```
dev=/dev/sdX
devs="/sys/block/${dev##*/}"
DEV_SIZE=$(<$devs/size)
```

Size it up...

Size is expressed as a number of blocks (or sectors) of, usually, 512 octets (8-bit bytes). You can confirm your device's block size and preferred partition alignment:

```
(( BLOCK_SIZE = $(<$devs/queue/logical_block_size),
  ALIGN_OFFSET = $(<$devs/alignment_offset),
  IO_SIZE = $(<$devs/queue/optimal_io_size),
  ALIGN_SIZE = (IO_SIZE + ALIGN_OFFSET) / BLOCK_
SIZE ))
```

Use a 2048 block (1 MiB) alignment size if your device does not express a preference. The other sizes we'll refer to are the powers of 2 'MebiByte' and 'GibiByte' expressed as numbers of blocks:

```
(( KiB = 1 << 10,
  MiB = KiB ** 2,
  GiB = KiB ** 3 ))
```

In other words, a Mebibyte is 2048 blocks and a Gibibyte is 2,097,152 blocks.

We must reserve the first 34 blocks for the GPT. This also contains the Master Boot Record that is used when booting from a BIOS and the partition table used by operating systems that lack GPT support.

```
GPT_SIZE=34
```

Immediately following this are special boot partitions – the BIOS Boot Partition occupies the remaining space up to the first Mebibyte-aligned block, which is where the EFI System Partition begins. We reserve 50MiB for that, so maximising the space available for other things.

```
(( BIOS_START = GPT_SIZE,
  EFI_START = MiB / BLOCK_SIZE,
  EFI_SIZE = 50 * MiB / BLOCK_SIZE ))
```

GRUB uses the BIOS Boot Partition when booted by a BIOS and the EFI System Partition is used when booting by UEFI.

The next partition is used to hold your boot files. These are needed by GRUB (around 15MiB) plus any bootable ISO images that you want. You should size it appropriately for your needs while considering the size of the USB device and to leave enough space for general data storage. The examples that follow reserve 6GiB for the GPT and these first three partitions, and reasonably assumes that the USB device has at least a 16GiB capacity (otherwise it wouldn't be much use for data).

```
(( BOOT_SIZE = 6 * GiB / BLOCK_SIZE,
  BOOT_START = EFI_START + EFI_SIZE ))
```

The GPT scheme includes a backup of the GPT at the end of the device. This means that the final 34 blocks are also reserved. The space that remains is available for data. You can create one or more partitions to suit your own requirements. Our examples split the space in two to accommodate plain-text and encrypted storage:

```
(( DATA_SIZE = DEV_SIZE - BOOT_SIZE,
  DATA_CRYPT_SIZE = DATA_SIZE / 2,
  DATA_PLAIN_SIZE = DATA_SIZE - DATA_CRYPT_SIZE,
   DATA_PLAIN_START = BOOT_SIZE,
  DATA_CRYPT_START = DATA_PLAIN_START +
DATA_SIZE / 2 ))
```

Dividing the space this way may result in a split that is out of alignment. Aligned block numbers are divisible by the device's alignment size – dividing and multiplying by that amount will bring the split point into alignment due to rounding caused by the shell's integer division:

```
(( DATA_CRYPT_START = DATA_CRYPT_START /
ALIGN_SIZE * ALIGN_SIZE ))
```

You also need to know where each partition ends:

```
(( BIOS_END = EFI_START - 1,
  EFI_END = BOOT_START - 1,
  BOOT_END = DATA_PLAIN_START - 1,
  DATA_PLAIN_END = DATA_CRYPT_START - 1,
  DATA_CRYPT_END = DEV_SIZE - GPT_SIZE ))
```

Lay it out...

That's everything we need. Let's start with a clean slate:

```
sgdisk --zap-all $dev
```

Now create the partitions by number, giving the blocks where they begin and end. Set types (**-t**) to reflect to their content and label them (**-c**) accordingly:

```
sgdisk -o -n 2:$EFI_START:$EFI_END -t 2:EF00 -c 2:'EFI
System Partition' \
      -n 3:$BOOT_START:$BOOT_END -t 3:8300 -c
3:boot \
      -n 4:$DATA_PLAIN_START:$DATA_PLAIN_END -t
4:0700 -c 4:DATA \
      -n 5:$DATA_CRYPT_START:$DATA_CRYPT_END -t
5:8301 -c 5:TailsData \
      -n 1:$BIOS_START:$BIOS_END -t 1:EF02 -c 1:'BIOS
Boot Partition' \
      /$dev
```

The BIOS boot partition is laid down last because it isn't aligned and doing it this way prevents it from being moved from where it should be. Mandatory type codes identify the partitions used for booting: the EFI system partition must be type **EF00** and the BIOS boot partition must be type **EF02**. The other partitions' types reflect the filesystems they'll contain. The encrypted data partition is typed and labelled in such a way that allows its use as a Tails persistent volume.

Next add the hybrid MBR. Due to a limitation with *sgdisk* you will need to use *gdisk*, for this step. Launch it with **gdisk /dev/$dev** and issue the following:

- **r** to select the recovery and transformation options sub-menu;
- **h** to make hybrid MBR;
- **4** to select the data partition (using its number);
- **N** not to place the EFI GPT first;
- **0c** to set the partition type (Windows FAT32);
- **N** not to set the bootable flag;
- **Y** for an additional protective partition;
- **EE** to set the additional partition's type;
- **w** to write the table to disk, and
- **Y** to confirm the write action.

QUICK TIP

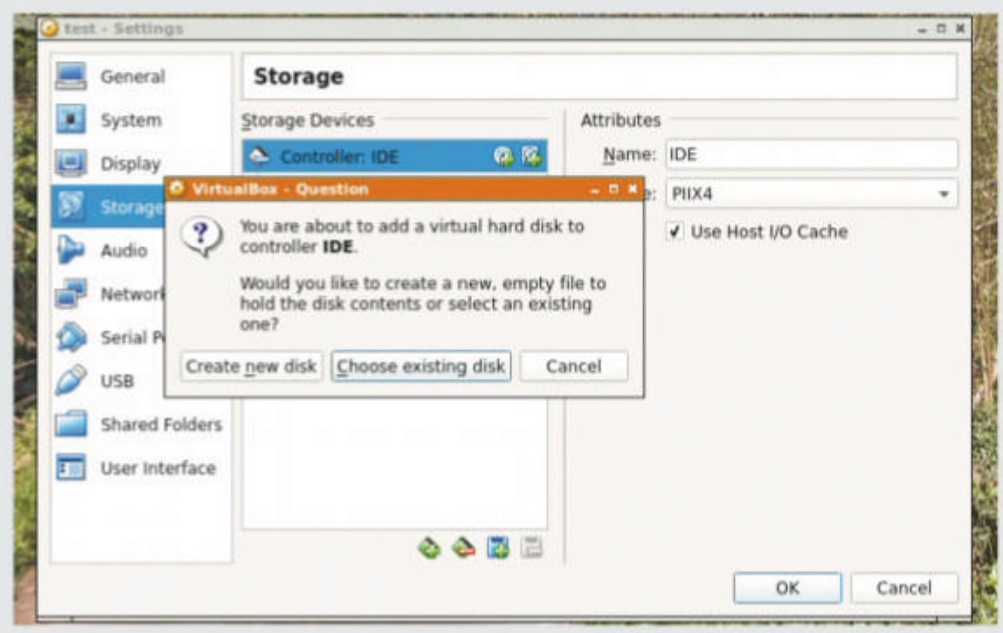
Labelled partitions are easily identified using /dev/disk/by-partlabel and `sgdisk -L` **lists the available partition type codes.**

VIRTUAL SANDBOX

You can use *VirtualBox* to test your USB stick. Ensure you have write access to the device (**chown $USER /dev/sdX** if necessary) and then create a raw virtual hard drive mapping like this:

```
VBoxManage internalcommands createrawvmdk -rawdisk /dev/sdX
-filename usbstick.vmdk
```

Create a virtual machine and opt to use an existing hard disk when prompted, selecting the virtual disk created previously. Boot the virtual machine to test your USB stick. *VirtualBox* machines usually boot using a BIOS but you can also test your USB stick's UEFI capability. Bring up the machine's settings and select the Enable EFI option on the System tab.



QUICK TIP

Using *sgdisk* makes scripting easy. You can use a GPT-aware interactive tool such as *gdisk* or *gparted* if you prefer. You can loop mount an ISO and look at its boot configuration files to discover the parameters required to boot it.

This configuration presents the plain-text data partition to systems lacking GPT awareness. It's placed first since such systems are otherwise easily confused.

With the partitions in place, you can create file systems. GRUB uses the BIOS boot partition directly – it does not need a file system – but the EFI system partition must be FAT32. This is also used for the data partition so that Windows can read it. The boot files' partition can use a Linux-native file system like Ext4:

```
mkfs.vfat ${dev}2
mkfs.ext4 ${dev}3
mkfs.vfat ${dev}4
```

Labelling the file systems make identification easy (via **/dev/disk/by-label**):

```
e2label ${dev}3 boot
dosfslabel ${dev}4 DATA
```

The data partition's label is upper-case because Windows prefers it that way.

Finally, encrypt the last partition and create its file system – Ext4 because it's only intended for Linux use:

```
dm="dm-$RANDOM"
enc=$(mktemp -d)
cryptsetup luksFormat ${dev}5
cryptsetup open ${dev}5 $dm
mkfs.ext4 /dev/mapper/$dm
cryptsetup close $dm
```

Here, **dm** is a temporary volume name created using the device name and a random number. The device is formatted for LUKS encryption, unlocked to write the file system and locked again afterwards. Enter a suitable password when requested.

Double GRUBble...

The boot loader needs to be installed onto the stick twice: once for BIOS and again for UEFI. Mount the boot and EFI filesystems and then run *grub-install* for each:

```
efi=$(mktemp -d)
boot=$(mktemp -d)
mount ${dev}2 $efi
mount ${dev}3 $boot
grub-install --target=i386-pc --recheck --boot-directory="$boot" $dev
grub-install --target=x86_64-efi --recheck --removable --efi-directory="$efi" --boot-directory="$boot"
umount $efi
```

QUICK TIP

You can specify the imgdevpath using a partition label, UUID or any valid device path. You can prepare a used stick by wiping it with random data: cat /dev/urandom > /dev/sdX.

Don't unmount the boot file system just yet because the boot menu must be configured. This is done using the *grub-mkconfig* command-line tool to generate GRUB's configuration file, **grub.cfg**. But we're going to configure it manually by writing **grub.cfg** ourselves, producing a leaner and simpler configuration.

GRUB uses the same configuration whether it was BIOS or EFI booted. It expects those files to be in a **/grub** subdirectory, so create that first:

```
mkdir $boot/grub
```

The EFI boot menu needs a font file; copy it from the host system:

```
cp /usr/share/grub/unicode.pf2 $boot/grub/
```

Then begin the menu configuration by loading the font: create **$boot/grub/grub.cfg**:

```
insmod all_video
font_path=/grub/unicode.pf2
loadfont unicode
```

On the menu...

The remaining configuration work is to define menu entries for the systems that you want to boot with the USB stick. This requires adding **menuentry** blocks to the **grub.cfg** file that instruct GRUB to load a kernel and its associated initial RAM filesystem image. We'll define menu entries to boot systems on the hard drive as well as the bootable images stored on the USB stick.

A useful function of a USB boot stick is recovery booting – an alternate way to boot your system in case your boot loader or its configuration gets corrupted in any way. The easiest way to do this is to reuse your system's existing boot menu (assuming, of course, that it uses GRUB to boot). Your system will have its own **grub.cfg** that you may have used *grub-mkconfig* to generate. You can copy it to your USB stick – renamed so it doesn't overwrite the USB stick's own config:

```
cp /boot/grub/grub.cfg $boot/grub/grub-recovery.cfg
```

then create a **menuentry** in **grub.cfg** to load it:

```
menuentry 'recovery config' {
 drivemap -s hd0 hd1
 configfile /grub/grub-recovery.cfg
}
```

We use **drivemap** to make **hd0** the system's drive because its boot configuration will have been written assuming that's the case (the USB stick boots as **hd0** and assigns the system's drives from **hd1**).

Selecting that menu option will display the system's original boot menu and enable you to boot your existing system in the event that its own boot loader lets you down. Note: if you update your systems' configuration, to copy the updated version to the USB stick.

The other purpose of the boot stick is to boot from a collection of image files stored on the USB stick, for example in an **/iso** directory:

```
mkdir $boot/iso
```

You should store any images you'd like to boot here. We will create a **menuentry** for each of them so one can be selected at boot time. Such menu entries follow a similar structure that is, in essence:

```
menuentry 'some iso' {
 loopback loop /iso/some.iso
 linux (loop)/...
 initrd (loop)/....
}
```

This loop mounts the image file **/iso/some.iso** and boots using the kernel and initial RAM filesystem contained within. Any kernel can be booted this way but booting the kernel is only half of the story. The system's startup code mounts the root file system which usually means looking at real devices to locate it. But an image file isn't a real device, so help is therefore required and the startup code must be willing to receive it!

In other words you need to pass the location of the image using a kernel command line parameter that the startup code, not the kernel itself, accepts. So the ability

» MAKING ISO IMAGES

Most ISO images you'll download directly, either via your web browser or perhaps using a command-line tool such as *Wget* or *Curl*. But it's also easy to make an ISO image from a DVD – just pop it into an optical drive and copy it (really – you don't need **dd**):

```
cp /dev/cdrom $boot/iso/some.iso
```

to boot an image file depends on how that startup code is implemented and whether it accepts such arguments.

It likely parses the kernel's command line by reading it from **/proc/cmdline** and parsing for arguments it understands, and different implementations look for different things. There is no standard for this and that makes booting images more difficult than it could be.

SystemRescueCD

For our first menu entry, save a SystemRescueCD (**www.system-rescue-cd.org**) image as **iso/systemrescuecd.iso**. Write the menu entry like this:

```
menuentry 'System Rescue CD' {
  isofile="/iso/systemrescuecd.iso"
  loopback loop $isofile
  linux (loop)/isolinux/rescue32 isoloop=$isofile
  initrd (loop)/isolinux/initram.igz
}
```

The image file's path is stored in a variable because GRUB needs to loop mount it and also pass it on the kernel command line. The paths to the kernel and initial RAM filesystem are relative to **(loop)**, which is how GRUB represents our loop-mounted image file. Those paths are specific to the System Rescue CD, as is the **isoloop** parameter used to pass the image path.

As an example of a real live Linux distro we can boot Ubuntu. With the image file in place, an appropriate menu entry would be:

```
menuentry 'Ubuntu 18.04 Desktop' {
  set isofile="/iso/ubuntu-18.04-desktop-amd64.iso"
  loopback loop $isofile
  linux  (loop)/casper/vmlinuz file=/cdrom/preseed/
ubuntu.seed boot=casper iso-scan/filename=$isofile
quiet splash  -
  initrd (loop)/casper/initrd.lz
}
```

As you can see, the arguments passed to **linux** and **initrd** are quite different to the previous example. Any Debian based distro (Ubuntu, Mint, Elementary OS, etc) should follow this pattern. Red Hat based distros (Fedora, CentOS, etc.) follow a slightly different pattern:

```
linux  (loop)/isolinux/vmlinuz
root=live:CDLABEL=Fedora-WS-Live-28-1-1 iso-scan/
filename=$isofile rd.live image
initrd (loop)/isolinux/initrd.img
```

These distributions are able to scan file systems on real devices to locate a given path, some others may need to be told a specific device. So it's worth specifying the image files' device as a global variable that all menu entries can use:

```
imgdevpath='/dev/disk/by-partlabel/boot'
export imgdevpath
```

The definition should go before any menu entries that need it. Note how it's specified using a Linux path, not GRUB, because it is the operating system – not the bootloader – we're helping out here. This relies on the **boot** partition label that was set during the partitioning phase. We need to *export* the variable to make it visible inside the menu entry functions.

The Arch Linux image (**www.archlinux.org/download**) is an example of one that needs to be given the image device path:

```
linux (loop)/arch/boot/x86_64/vmlinuz
archisolabel=ARCH_201807 \
    img_dev=$imgdevpath img_loop=$isofile
earlymodules=loop
initrd (loop)/arch/boot/x86_64/archiso.img
```

You will need to research the boot command line required by the images you want to boot.

You can download a script that implements the steps we've followed, along with example GRUB configuration from **bit.ly/grubbyusb**. And a final, but nonetheless important, tip: if you rely on a USB stick to boot a system or store data, make sure you have more than one – USB sticks can die without warning! LXF

PERSISTENT TAILS

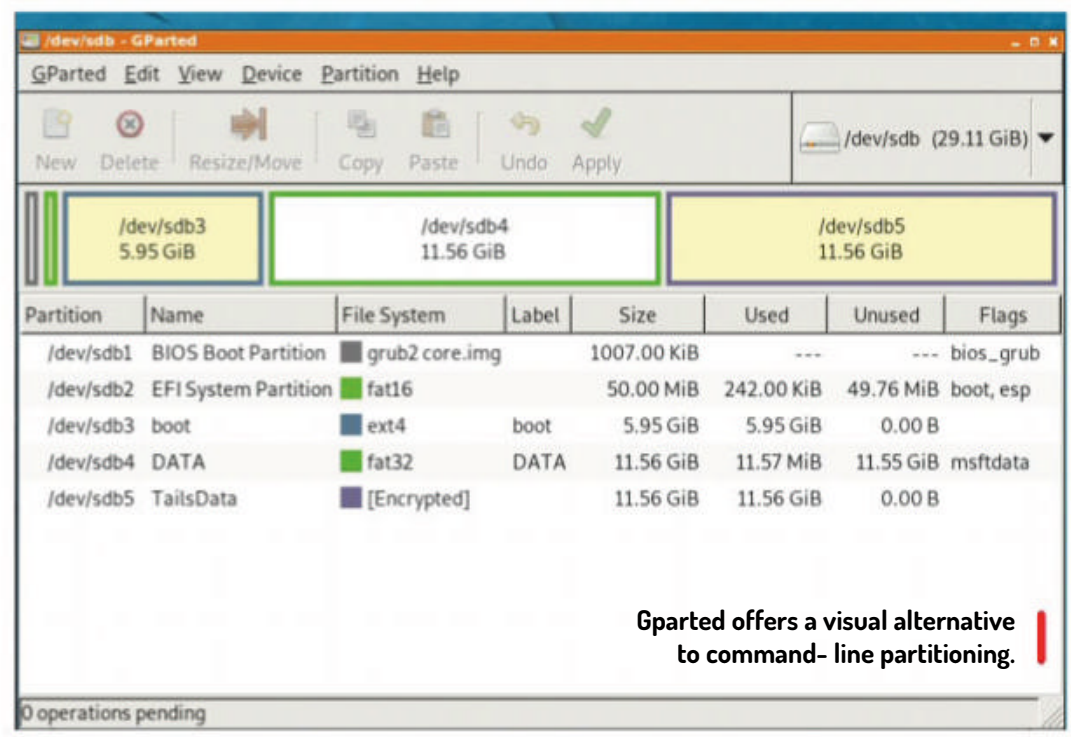
A final example is Tails Linux – we previously set an encrypted data partition that we can use as a persistent volume. If you aren't familiar with Tails then you should read more about this privacy distro at **tails.boum.org**. A menu entry for Tails Linux looks like this:

```
menuentry 'Tails Linux' {
  set isofile="/iso/tails-amd64-3.6.2.iso"
  loopback loop $isofile
  linux (loop)/live/vmlinuz fromiso=$imgdevpath/$isofile boot=live
config splash noautologin
  initrd (loop)/live/initrd.img
}
```

A few additional steps are required if you want to use the encrypted data partition for Tails persistence. Unlock it like we did when formatting it and do the following:

```
mount /dev/mapper/$dm $enc
echo '/home/amnesia/Persistent source=Persistent' > $enc/
persistence.conf
chown 115:122 $enc/persistence.conf
chmod 600 $enc/persistence.conf
setfacl -m user:115:rwx,group::rwx,mask::rwx $enc
umount $enc
```

These steps configure the encrypted volume so that Tails recognises it as a persistent volume, otherwise it will be ignored. Finish by unmounting and locking. Booting your Tails image should now recognise the persistence partition.



Gparted offers a visual alternative to command- line partitioning.

ESSENTIAL ART RESOURCES
Exclusive videos and custom brushes are available with your digital editions!

Try out a digital edition for **FREE** today!

Just search for 'ImagineFX' on these selected platforms…

Available on the App Store · nook by Barnes & Noble · zinio™ · Available on amazon apps kindle fire · Google play

ADMINISTERIA

Iwd: The wireless kid

Valentine Sinitsyn considers a balance between keeping the status quo by adding another abstraction layer and rewriting everything from scratch.

Wi-Fi isn't the same as Ethernet with no patch cords. Given the media is accessible to anyone, care needs to be taken to stop eavesdropping. The first take on that, Wired Equivalent Privacy (WEP) was simple to configure. You only needed a static key, basically a sequence of hexadecimal digits which could be as short as ten of them. You supplied these to *iwconfig* and the kernel happily handled the rest. However, it was discovered that WEP is neither wired-equivalent or private. Stronger encryption schemes required more sophisticated authenticators, often called supplicants as they supply credentials. Those were typically implemented as userspace daemons.

OUR EXPERT

Dr. Sinitsyn is a cloud infrastructure developer at Yandex by day, and an open source contributor by night, with interest in everything from AH to X509.

» LINUX ISN'T AN ONION

All problems in computer science can be solved by another layer of indirection. Except it introduces yet another level of indirection. The aphorism coined to David Wheeler isn't news but, surprisingly, it stays up to date. In Linux, it is further amplified because the system is open, and all of the abstraction layers it implements can be clearly seen.

This isn't to say abstraction layers are not doing any good. Many primitives we take for granted – sockets, processes or files, say – are actually abstractions on top of memory buffers, CPU contexts and device blocks. And, yes, these are abstractions as well. Without this high-level view, you'd be forced to connect logic gates, which makes the road to your database backup script way too long.

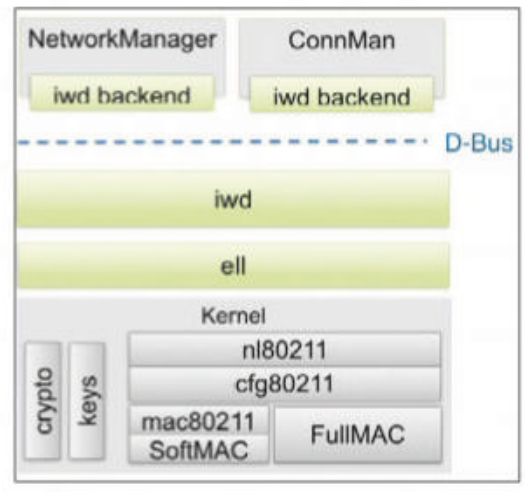
The tricky part is keeping the number of these abstraction layers right. If it's too low, you don't get far from logic gates and spend time programming nuts and bolts instead of solving your problems. If it's too high, abstractions become a burden. Top layers don't add too much value but consume resources. You still pay the abstraction cost even if you don't need it. Even worse, these abstractions often become 'leaky', forcing you to understand what they tried to hide.

Linux is special in that no single entity controls it. So, there are competing implementations for many tasks and some abstractions exist just to close the gap between these implementations. Whether this works well or not depends on how different these implementations are. One of such abstractions is in the focus of this Administeria. Does it do any good? Speak your brains: **lxf.letters@futurenet.com**.

Until recently, there was essentially one such daemon in Linux: *wpa_supplicant*. It is a time-proven technology you are likely to find everywhere: from your home router through to an Android smartphone and your Linux laptop. It's a real Swiss army knife which supports everything including WPA3. However, *wpa_supplicant* is complex, lacks management APIs, and these things aren't likely to improve.

So Intel decided that it's time to start over. The new contender, *iwd*, claims to address these issues and adds some more, say, multi-client support. Another goal is to optimise resource utilisation. To this end, *iwd* purposely doesn't support anything but Linux, communicates via D-Bus and re-uses the kernel's crypto subsystem (yes, no OpenSSL). The fact that *iwd* resides in **git.kernel.org** signifies the above. The number of external dependencies is kept at the minimum: essentially, the only one is ELL (Embedded Linux Library), which comes with *iwd* itself. The downside is that you need a relatively new kernel to run *iwd: 4.20* and above unless you are willing to do some patching.

Another goal was to ask users only for data that is impossible to learn any other way, such as passphrases. The daemon keeps track of users' input, so they shouldn't have to repeat themselves. The only pain is WPA Enterprise. The authors consider it too complex to configure interactively in all cases – which is true – so a 'network administrator' would need to provide a configuration file to store in **/var/lib/iwd** instead. It is unclear whether this idea would work out or not. Otherwise, *iwd* integrates with *systemd-networkd*, *NetworkManager* and *ConnMan* so switching from *wpa_supplicant* should be transparent. However, if you don't use any of these, you can still configure a connection with minimum effort via the *iwctl* command-line tool. Much like you would have done in ye olden days of *iwconfig* and WEP, I'd say.



Iwd reuses as much of Linux as possible. It's better to optimise D-Bus than to run iwd without it, its authors believe.

Netplan: One to rule them all

It has been over a year since Ubuntu quietly deprecated venerable ifupdown. Was this really a change for the better?

Network configuration in Linux was never straightforward. It wasn't overly complicated either, but every distro seems to do it in a different way. Red Hat has **/etc/network** and stores configuration as **key-value value**, Slackware relies on two shell scripts under **/etc/rc.d**. Debian and (until recently) Ubuntu employs *ifupdown*, which reads **/etc/network/interfaces** and its contents.

Does this world need yet another network configuration system? The quick answer would be 'no', but it depends on the motivation. Neither of the existing implementations show great support for Wi-Fi and mobile broadband, which are typically managed through *NetworkManager*. The all-encompassing *systemd* has *networkd*, and sometimes these two run side-by-side. Wouldn't it be nice to have a single configuration file, one could almost say one file to rule them all?

Meet Netplan

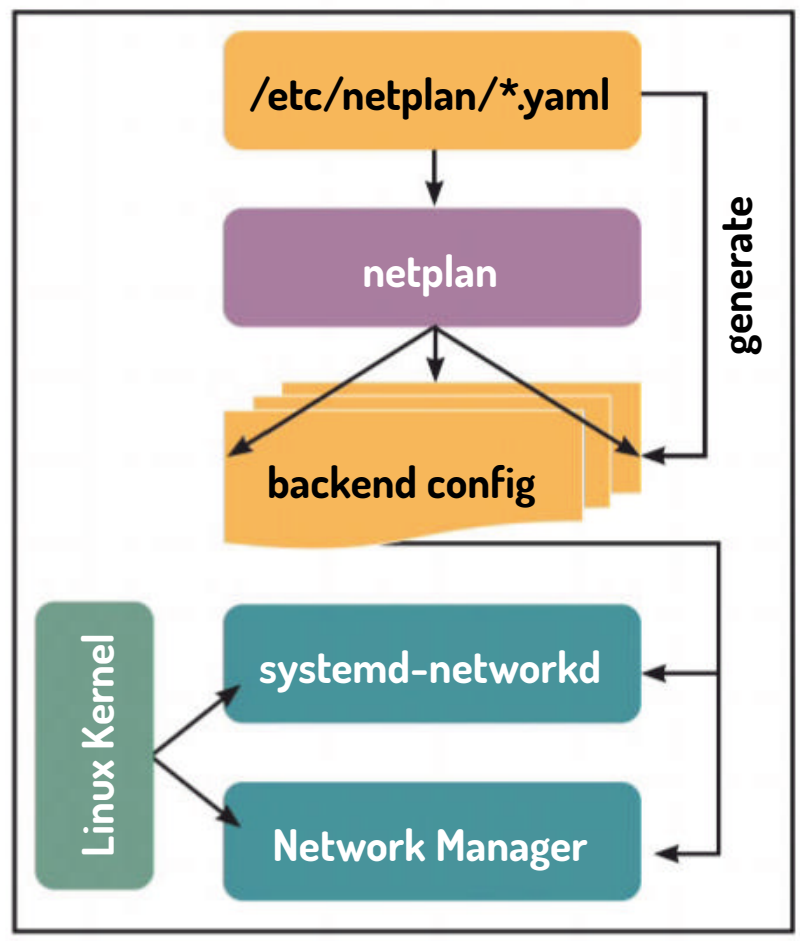
This is exactly the reasoning behind *Netplan* (**http://netplan.io**), a Canonical-backed, open source network management tool. It is formally distribution-agnostic but not widely adopted outside Ubuntu, which made it an official network configurator in 17.10. This fact alone makes *Netplan* worth your attention despite the scepticism some folks express over the internet. Networking is essential to systems operation, and how are you going to manage your Ubuntu fleet if you don't understand the component that manages it?

Netplan isn't a direct replacement for the *ifupdown* it deprecates, however. For instance, it doesn't currently handle dispatcher scripts, which some packages (*you*, *avahi, openvpn*) still bring to **/etc/network/if-up.d** and the like. The official workaround would be to use *networkd-dispatcher* (**https://bit.ly/2OYRvmd**). This service listens to D-Bus events which *systemd* generates and runs hooks in **/etc/network-dispatcher** as appropriate. *NetworkManager* can do the same out of the box. You can reinstall *ifupdown* and leave some of your network interfaces under its control, since *Netplan* never touches anything not listed under **/etc/netplan**.

Powerful generators

Netplan is purposely small and has just the most basic dependencies. This is because it runs very early in the boot process as a *systemd* generator. Generators are small binaries that get executed before *systemd* units so they have a chance to symlink or otherwise change unit definitions. *Netplan* generator is a compact binary that is written in C. On top of it, a Python-based command-line tool is implemented. This leaves you an option to regenerate, migrate and inspect the configuration after the system is booted.

Netplan tries to bring together *NetworkManager* and *networkd*, two network management daemons that ship on Ubuntu and many other distros. Besides, there are few key architecture goals not covered yet. *Netplan* considers generated configs ephemeral, and you shouldn't have to deal with them directly. It also supports multiple config files so it should be easy for a



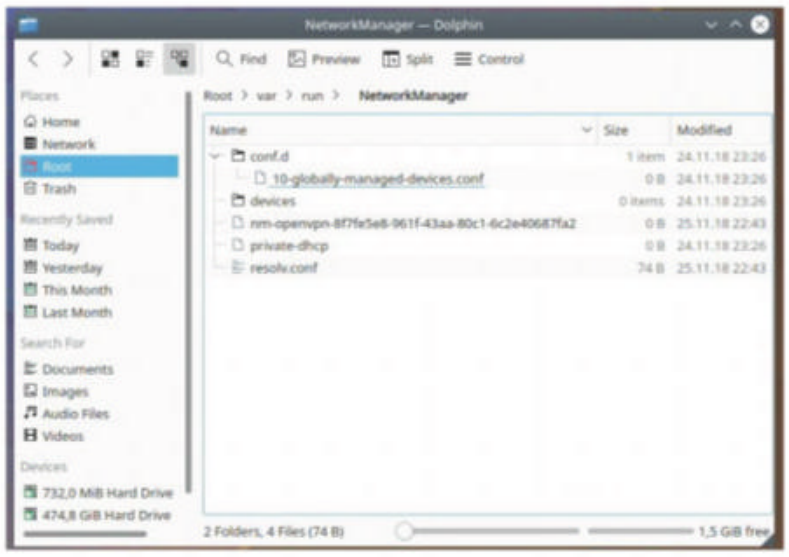
Netplan is tricky YAML preprocessor that takes network definitions and renders configs for NetworkManager, networkd or both.

A LOOK AT SYSTEMD-NETWORKD

Systemd is often criticised for trying to be everything, which is a clear departure from the Unix way. Like it or not, *systemd* can manage network connections, and a service to do this is *systemd-networkd* (or *networkd*, for short).

Like *Netplan* (and other daemons), *networkd* can read files from three locations, of which *Netplan* stores its generated configs in **/run/systemd/network**. But instead of a single config spanning multiple **conf.d** snippets, *networkd* provides three config types. The first one is **.network**, which contains [Match] sections to select physical devices based on their name, driver or MAC address. Then it applies network-level settings, such as whether to enable DHCP or IP forwarding or which DNS server to use. The second is **.netdev**, which serves roughly the same purpose but creates virtual devices. Finally, there are LINK files. Those are alternatives to custom *udev* rules. Here you tune link-level knobs, such as MTU, hardware offloading or naming policy. Depending on your network definition, *Netplan* may generate all three types. It also creates custom *udev* rules to rename network interfaces, and if you have **wifis:** in your YAML, it starts a sidecar *systemd* service called *netplan-wpa*. It wraps *wpa_supplicant*, and the reason it exists is that while *networkd* can configure wireless adapters, it doesn't (currently) handle Wi-Fi authentication.

For more details, see systemd-network(5), systemd-netdev(5), systemd-link(5) and friends man pages.



Netplan treats generated configs as ephemeral, so they typically end up in /var/run. Manual configs in /etc take precedence.

third-party application to drop in a snippet for the virtual bridge it creates, for instance. Last but not least, all *Netplan* configuration are YAML.

The simplest *Netplan* config (and the one on our laptop) is as follows:

```
network:
  version: 2
  renderer: NetworkManager
```

It just puts *NetworkManager* in charge of all devices in the system. Note: the latter is implicit. Although the config doesn't list any specific device, they are de facto under *Netplan* control.

This tiny snippet already highlights a few key points. Configuration is a single map keyed under `network:`. It must have `version: 2` for backwards compatibility, and it may have a `renderer:` key that says which backend to generate the configuration for. The default is `networkd` and it can be overridden globally (as it is done here), per device type (e.g. for all Ethernet adapters) or per device. *NetworkManager* and *networkd* differ in capabilities, and not all features (especially matches) are supported across both backends. One could argue this undermines the whole idea behind *Netplan*, but such is life.

Netplan configuration can span multiple files. In this case, each file would have its own `network:` map, and lexicographically later files would take precedence. So, it is quite common to call these snippets **01-foo.conf**, **02-bar.conf** and so on. Configs may come through one of the three locations: **/{lib,etc,run}/netplan**. The idea is that files in **/lib** are kept static. You never edit them but rather copy a file to **/etc** and do changes there. **/run** is for volatile configuration, such as the one *Netplan* generates. It is essential that the file retains its name, otherwise, the contents will be merged, not overridden.

Mixing and matching

Devices under *Netplan* control are grouped by type, e.g. `ethernets:` or `wifis:`. These come as keys in the top-level `network:` map and, naturally, are maps themselves. A few other types are recognised, namely bridges, bonds and VLANs, while other types (e.g. Veths) can be added in the future. The main difference between these two device groups is that the latter ones are virtual, so you can create as many bridges (or bonds, or VLANs), as you need. On the other hand, devices in the former group are physical: they either exist in your system or not. Of course, it would be nice to pull a new $100 network card out of thin air, but that's beyond *Netplan*'s capabilities. So `ethernets:` and `wifis:` can only be matched, but not created.

There are a few attributes you can use for matching, although depending on whether you choose *networkd* or *NetworkManager* as a backend, they vary slightly. The specifics are covered shortly, but now, it is important to understand that a match generally consists of multiple devices. Some attributes, say MAC addresses or kernel names, are unique. If you expect a match to select one device, use the `set-name:` key to assign the device a meaningful name. In case there are two or more devices that match, only the first one gets renamed and others generate an error in *dmesg*.

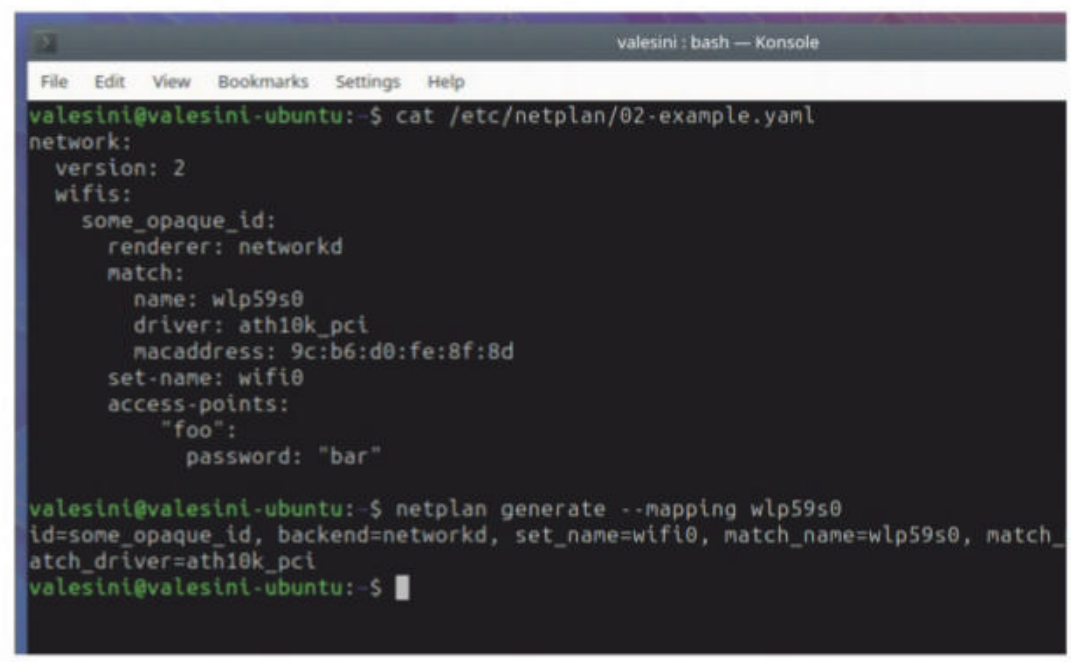
Keys in a per-device-type definition map (such as `ethernets:`) are called Device Configuration IDs. If there is no `match:` defined for a particular key, it is considered to be a kernel-assigned interface name. Otherwise, the key is just an opaque identifier. This saves you some boilerplate. The two snippets below do the same thing:

```
ethernets:
  enp2s0:
    # settings
  some_opaque_id:
    match:
      name: enp2s0
    # settings
```

As of time of the writing, three match attributes were supported: **name**, **driver** and **macaddress**. All but **macaddress** may include **globs (enp1*)**. This is where the abstraction becomes leaky, however. *NetworkManager* can't match anything by the driver, while *networkd* can't match devices under **wifis** by name. **Globs** in names are available in *networkd* only. This doesn't look very clean or consistent, and *Netplan* would complain if you get any of these rules wrong. The reason *Netplan* doesn't abstract out these differences is that the match happens in the specific backend. If *Netplan* was to match devices itself, it would be limited to those present at the generation stage, and features such as hotplug won't work.

Once a device is matched, *Netplan* applies the settings you specify. It would be quite unfeasible to dig

With the --mapping switch, Netplan calls its own version of the matching algorithms. Sadly enough, it also lacks support for globs.



into details here: there are many knobs, as *Netplan* can configure bridges, bonds, Wake-on-LAN, MTUs and many others. Moreover, it has a decent reference manual available online at **netplan.io/reference**. Instead of repeating it here, let us have a quick look at the commands that *Netplan* makes available for you.

Simple commands

Netplan wraps all of its features within a single tool: *netplan*. You call it with an action (or command) which tells *Netplan* what to do. Internally, The *Netplan* CLI is implemented in Python, and it calls into the C-based generator where appropriate.

Proper tools distinguish between generating a configuration and putting it into effect. This is a safe approach: one can inspect the produced configuration before it takes effect, and restart the process if it's invalid or otherwise inappropriate. *Netplan* is the same: it has `netplan generate` and `netplan apply`, and these are the two commands you typically run to change *Netplan* configuration on running systems.

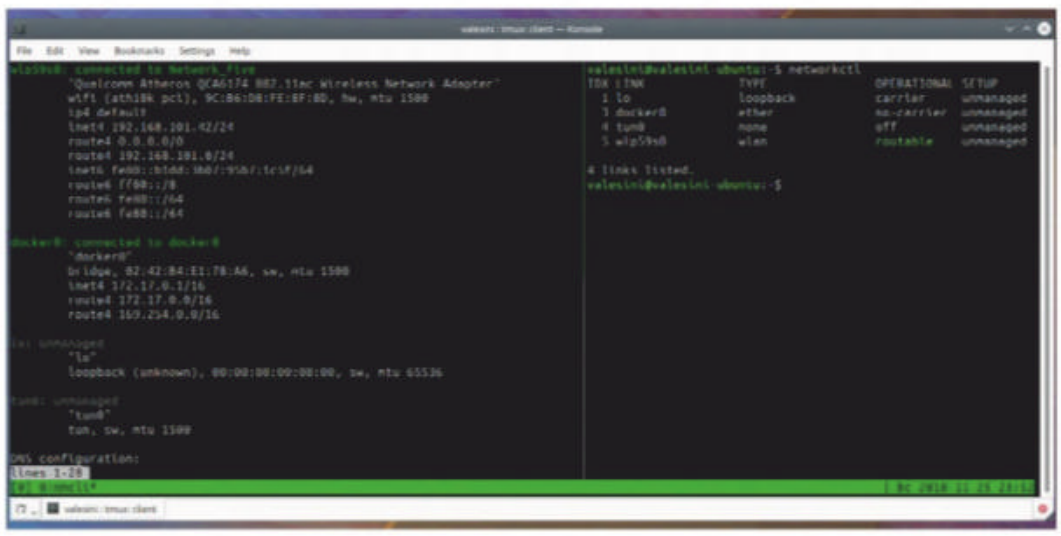
The first command, `netplan generate` is really a thin wrapper on top of the *systemd* generator. It changes files under **/run** so you run it with **sudo** unless you redefine a filesystem root with `--root-dir`. This is rarely used, though. Another switch, `--mapping`, tells `netplan generate` not to generate anything at all. Instead, it finds an interface you specify as an argument in YAML network definitions and then prints the device configuration ID, backend, MAC address, driver and the name this device would get assigned with **set-name** (if any). This helps with troubleshooting, although **globs** are not recognised.

A second command, `netdev apply`, 'commits' the generated configuration. This boils down to restarting the appropriate backend and fixing the device names. There is no need for the latter on a fresh system, so `netplan apply` doesn't run during boot.

A third related command, `netplan try`, tries to apply the configuration, but does an automatic rollback if things go wrong. Internally, it backs up the configuration, wraps `generate` and `apply` and asks you if it's OK to keep the new settings. If you fail to confirm it within two minutes, the command restores the backup.

Do note the restoration may fail, for instance, if bridges or bonds with custom parameters were used. I don't have those, yet `netplan try` once failed with an obscure `KeyError` for me. There is no silver bullet, and changing network configuration over SSH is not a good idea.

NetworkManager and networkd provide command-line tools. Here NetworkManager is in charge of the connection, so networkd reports it 'unmanaged'.



Hidden 'gems'

Besides these two (OK, three) *Netplan* commands there are few lesser-known ones. Most networks provide dynamic host configuration via DHCP these days. Lease file locations are naturally different per backend, so Netplan provides the `netplan ip leases <interface>` command to get the current lease for the interface. Internally, it calls `netplan generate --mapping` to learn the backend used to configure the interface. However, none of the interfaces are explicitly defined on my laptop (*NetworkManager*'s improved a lot since 2006!), so `netplan ip leases` does nothing here.

The last one is `netplan migrate`. This is how you are supposed to convert your old *ifupdown*-based setup into a shiny new *Netplan*-managed one. I say 'supposed' because this command is marked as 'testing' and it'll be hidden unless you invoke *Netplan* as `ENABLE_TEST_COMMANDS=1 netplan migrate -h`. There is a `--dry-run` switch so you can try everything safely. I did it against a non-trivial multifile *ifupdown* config we use at Yandex. Cloud test labs and had bad luck. The main issue is our interfaces are configured as either manual or auto, while *Netplan* recognises loopback, static and DHCP only. Given the nature of the manual network configuration method – which employs an arbitrary script to do the configuration – automatic migration for these configs seems convoluted, if they're possible at all.

That was a deep dive and now it's time to call it a day. Upon this examination, *Netplan* appears to be a custom tool to solve Ubuntu problems across Canonical offerings (Server, Cloud, MAAS). Perhaps this is the reason *Netplan* didn't receive much attention outside Ubuntu. There are some rough edges as well, although the development is still ongoing and they could be smoothed out by the time you read this *Administeria*. *Netplan* may feel superfluous, and to some extent, it really is. However, Ubuntu is popular, and understanding *Netplan* renders your systems administrator skill set more valuable. LXF

» NETWORKMANAGER IN 5 MINUTES

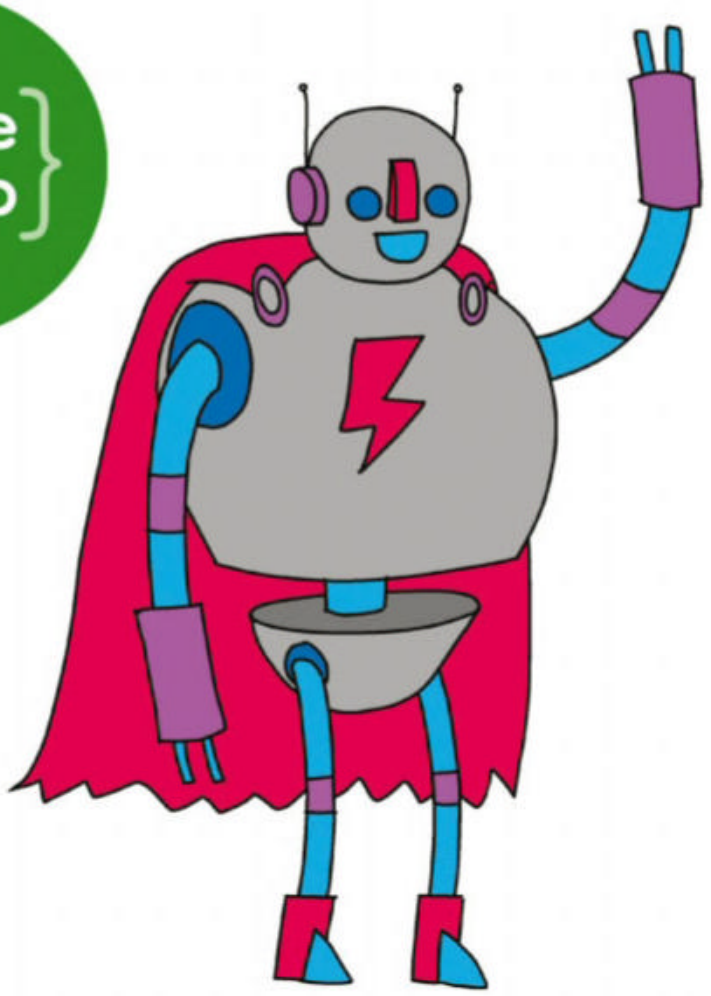
Many of us recognise *NetworkManager* as signal strength indicator in the system's dock. This is an applet backed by the *NetworkManager* daemon. The applet and the daemon communicate via D-Bus, and *NetworkManager* has its own config file and management CLI.

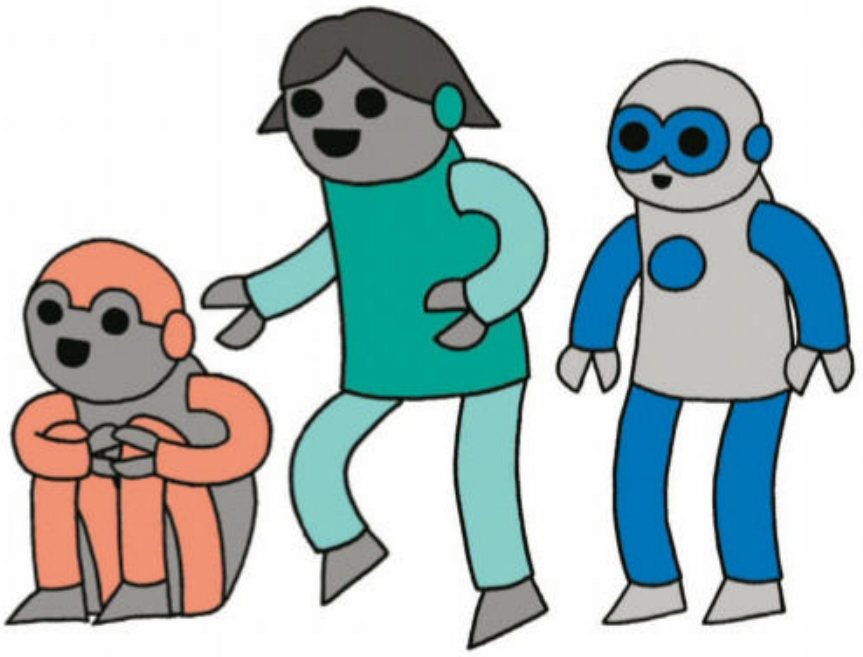
NetworkManager considers three locations: **/usr/lib**, **/run**, and **/etc**. The priority is different, though: a config in **/etc** shadows the one in **/run**. Users don't edit the main config file, **NetworkManager.conf**, but add their own snippets under **conf.d**. *Netplan* does exactly this as generated snippets go in **/run/NetworkManager/conf.d**.

NetworkManager uses a keyfile format A typical config lists plugins to be loaded and a few settings. There is one plugin that's always present, called *keyfile,* which manages system-wide connection storage (**/etc/NetworkManager/system-connections**) for *NetworkManager*.

Most of the time, you manage *NetworkManager* through the desktop applet. You rely on the GUI to go online. If the GUI is broken, and you download a fix, you are in trouble. In this case, the *nmcli* tool helps. `nmcli connection` lists available (system-wide) connections, and brings them up. Another often used (sub)command is `nmcli import`, which imports an external VPN config into *NetworkManager*.

For further details, consult the NetworkManager.conf(5) and nmcli(1) man pages.





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**

MIDDLEMAN

Develop secure and static websites

Mihalis Tsoukalos explains how to create inherently more secure and faster websites with the Middleman static site generator.

OUR EXPERT

Mihalis Tsoukalos is the author of *Go Systems Programming* and *Mastering Go*. You can reach him at **www.mtsoukalos.eu** and @mactsouk.

Most people don't like having a *Middleman* in the way, but for a change let's embrace this one and create websites with static web pages. The main advantage you get from having a static website is security. On top of that, static sites tend to be faster than dynamic sites because they only have to serve static content without waiting for a database connection to return data or a PHP script to execute–this in itself adds both additional security, due to less reliance on additional components, but also speeds page access times too. Over the next four pages you'll learn how to install and customise *Middleman* in order to create a regular website as well as a blog.

Tuning into static

Middleman is a static website generator. Put simply, it's a command line tool written in Ruby that enables you to create websites with static HTML pages using Ruby and Ruby Gems; the latter are like packages or modules found in other programming languages such as Python and Perl. Other similar tools that can help you create static websites include *Hugo*, *Sphinx* (which is mainly for creating documentation), *Flask*, *Roots* and *Jekyll*.

For our example we'll install *Middleman* on an Ubuntu Linux machine. The convenient thing is that *Middleman* is distributed and therefore can be installed

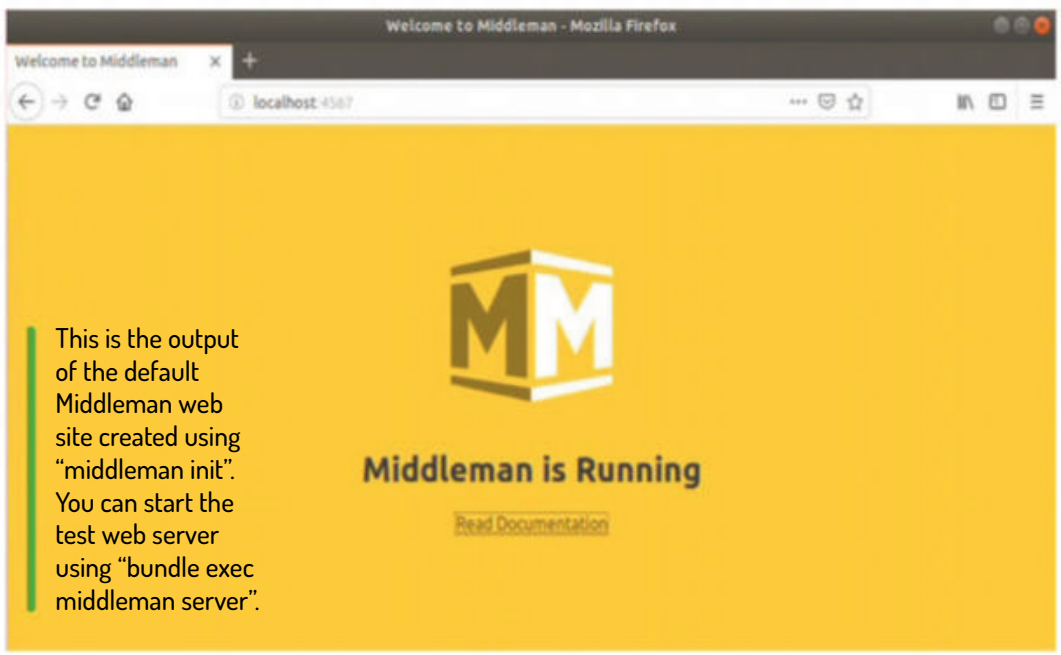
Here's how the index page of the lxf Middleman website looks after we've made some small changes to it.

using the package manager of Ruby, which is called *gem*. This also assumes that you should have Ruby already installed on your Linux machine, before being able to install *Middleman*. For the purposes of this tutorial the following commands are going to be executed on the Ubuntu machine:

```
$ sudo apt-get install ruby-dev
$ sudo gem install middleman
$ sudo apt-get install nodejs
$ sudo gem install middleman-blog
```

The second command will install lots of Ruby packages that are needed by *Middleman*. After that you can find out the *Middleman* version you have on your machine by executing `Middleman version`. The output you get will look similar to `Middleman 4.2.1`. To see the available commands of the *Middleman* command line utility, you can execute `Middleman help` to get a concise output or `Middleman --help` for a more detailed output. As *Middleman* is written in Ruby, you can see or modify its Ruby code. You can start that adventure by executing the command `less `which Middleman``.

Bear in mind that instead of installing *Middleman* locally, you can create a *Docker* image and use that for creating, building and deploying your *Middleman* projects. Although this method might look more difficult



This is the output of the default Middleman web site created using "middleman init". You can start the test web server using "bundle exec middleman server".

```
Using middleman-core 4.2.1
Using middleman 4.2.1
Using middleman-blog 4.0.2
Using redcarpet 3.4.0
Bundle complete! 6 Gemfile dependencies, 44 gems now installed.
Use `bundle info [gemname]` to see where a bundled gem is installed.
mtsouk@ubu:~/code/middleman$ cd myBlog
mtsouk@ubu:~/code/middleman/myBlog$ vi source/aFile.html.md
mtsouk@ubu:~/code/middleman/myBlog$ vi config.rb
mtsouk@ubu:~/code/middleman/myBlog$ middleman article LXF-issue-200
WARN: Unresolved specs during Gem::Specification.reset:
      rack (< 3, >= 1.4.5)
      minitest (~> 5.1)
      concurrent-ruby (>= 1.0.2, ~> 1.0)
WARN: Clearing out unresolved specs.
Please report a bug if this causes problems.
      create  source/2018-11-24-lxf-issue-200.html.markdown
mtsouk@ubu:~/code/middleman/myBlog$ bundle exec middleman article LXF-i
      create  source/2018-11-24-lxf-issue-210.html.markdown
mtsouk@ubu:~/code/middleman/myBlog$ middleman build
WARN: Unresolved specs during Gem::Specification.reset:
      rack (< 3, >= 1.4.5)
      minitest (~> 5.1)
      concurrent-ruby (>= 1.0.2, ~> 1.0)
WARN: Clearing out unresolved specs.
Please report a bug if this causes problems.
== Blog Sources: {year}-{month}-{day}-{title}.html (:prefix + :sources)
      create  build/aFile.html
      create  build/2018/11/24/lxf-issue-200.html
      create  build/index.html
      create  build/2012/01/01/example-article.html
      create  build/2018/11/24/lxf-issue-210.html
      create  build/feed.xml
      create  build/tags/example.html
      create  build/2018.html
      create  build/2018/11.html
      create  build/2018/11/24.html
      create  build/2012.html
      create  build/2012/01.html
      create  build/2012/01/01.html
Project built successfully.
```

This is the output of the Middleman commands that were used for creating the blog website and adding two blog posts to it.

at first, it can ultimately save you lots of time if you are planning to use multiple UNIX machines for creating *Middleman* websites.

You might begin by using the following Dockerfile:

```
FROM ruby:2.4-alpine
RUN apk --update add alpine-sdk nodejs
RUN gem install middleman
WORKDIR /app
```

This tutorial does not rely on any particular method of installation for *Middleman*, and it would be a good exercise for those of you who are familiar with *Docker* to try using a *Docker* image when following this tutorial. In this case, bear in mind that you might need to change some of the presented commands in order to match the working environment of your *Docker* container image.

Your first static site

To get going let's learn how to create your first static website in *Middleman*. The command that you need to execute first in order to create a project for a website and begin working with it is `Middleman init` followed by the name of the project. Therefore in order to create a project that will be named **lxf**, you need to execute `Middleman init lxf`. After downloading lots of files and dependencies, this command will create a new directory that will be named after the name of the project. At this point, it would be really helpful to discuss the directory structure and the files included in a *Middleman* project, taking the **lxf** project as an example.

Some important files exist inside the **lxf** directory. The single most important file is called **config.rb** which is a Ruby file that holds the configuration of the *Middleman* project and is located at the root directory of each *Middleman* project. The easiest way to change a setting is by using the `set` command or the newer `config[PARAMETER] = VALUE` syntax. You will also find a plain-text file called **Gemfile** that holds the Gem dependencies of your *Middleman* project – you should put any extra plugins and libraries that you will be using in your project in there.

The actual files for the website are located inside a directory called **source**. Inside there is a file called **index.html.erb**, which is the home page of the website (**index.html**). ERB files enable you to create dynamic templates because they allow you to have embedded Ruby code in them. Apart from that ERB file, the **source** directory contains four other directories named **images**, **javascripts**, **layouts** and **stylesheets**. You are free to create your own directories inside **source** in order to organise your project the way you want.

The **images** directory is for storing the images that you want to use in your web site, unsurprisingly; you will have to reference them in your files in order for them to be included in the final output. The **javascripts** directory is for storing any JavaScript files you use for your website. The **layouts** directory is for storing template files that have to do with the appearance of the pages of a website. In our case, it contains a single file named **layout.erb** – if you want to create a navigation menu, a header or a footer for your website, this is the right place to put them. Lastly, the **stylesheets** directory is for storing your CSS files – right now this directory contains a single file named **site.css.scss**.

You will need to execute the `bundle exec Middleman server` or `Middleman server` command to test that website on your local Linux machine – it is considered good practice to test the look and the contents of your websites locally before deploying them on the internet. What both commands will do is to compile everything that exists on the *Middleman* site and serve it to you on

QUICK TIP

You can get more info about Middleman at https://middlemanapp.com. The documentation, can be found at https://middlemanapp.com/basics/ install and the Community site at https://middlemanapp.com/community/. However, nothing can replace the experience you are going to get from actually using Middleman.

» MIDDLEMAN VERSUS HUGO

The first difference between *Middleman* and *Hugo* is that the former in written in Ruby and requires a Ruby installation in order to be executed, whereas the latter is written in Go and is a statically linked binary, which means that it has no other dependencies. The point here is that, if you are familiar with Ruby, choosing *Middleman* is the rational choice.

The second difference is that *Hugo* has a larger development community than *Middleman*, which means that you might be able to choose from a bigger variety of themes. Additionally, the documentation site of *Hugo* is friendlier than the documentation site of *Middleman*, although this is a matter of personal taste.

Middleman can handle larger blog sites than *Hugo*, which means that if you want to create the blog site for a big company, *Middleman* might be a better choice. Last, *Hugo* is faster than *Middleman* mainly because it executes a binary executable instead of Ruby. However, if speed is not your main concern, this will not be a problem.

The main thing to remember here is that both *Middleman* and *Hugo* can easily create modern static websites. Therefore, the best way to choose between *Middleman* and *Hugo* is to try both and choose the one you like the most. If you have the time you can also check other alternatives, starting with *Sphinx* and *Jekyll*.

QUICK TIP

If you think Middleman is too complex for your tastes, take a took at Hugo (https://gohugo.io), which is also a framework for building static websites. Hugo is simpler than Middleman whereas Middleman can support more complex sites than Hugo. You can also have a look at the Jekyll static website generator (https://jekyllrb.com).

a local web server. Their output should look similar to the following:

```
$ bundle exec middleman server
== The Middleman is loading
== View your site at "http://ubu.local:4567", "http://10.0.2.15:4567"
== Inspect your site configuration at "http://ubu.local:4567/__middleman", "http://10.0.2.15:4567/__middleman"
```

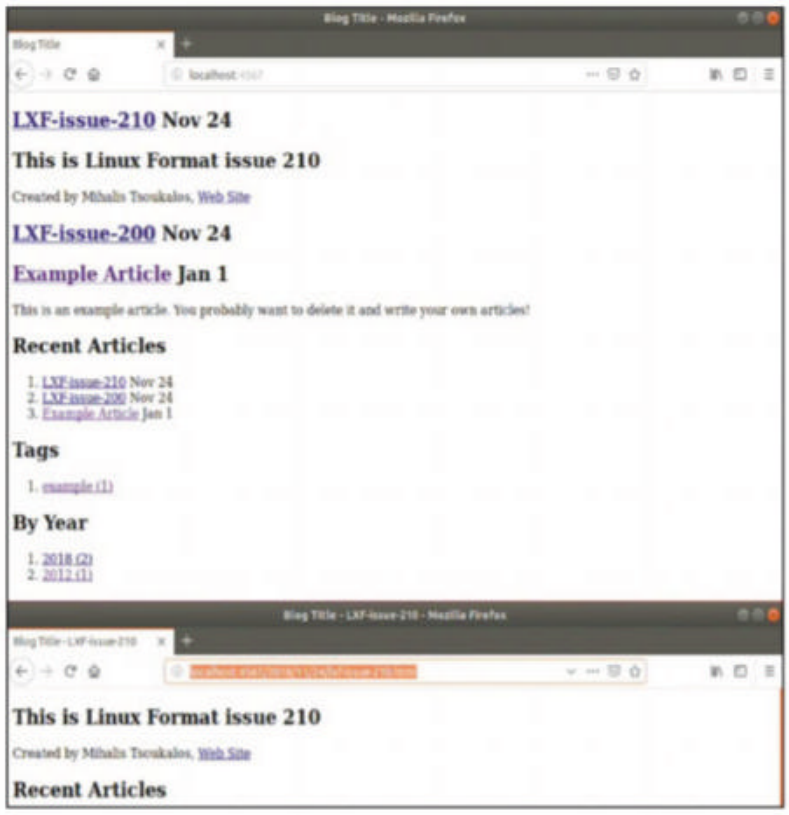
The good thing is that the aforementioned output illustrates the URL that you will need to visit in order to view and inspect the configuration of your website. The screen on page 88 *(bottom left)* shows the output that you will get when you connect to **http://10.0.2.15:4567** using your favourite web browser – this URL shows that the port number used by the *Middleman* web server is 4567. This also verifies that everything is working perfectly with your *Middleman* installation.

So, after learning about the directory structure of a *Middleman* project and how to start the test web server, let's try to add a new page to that site and modify its index page by executing the following commands:

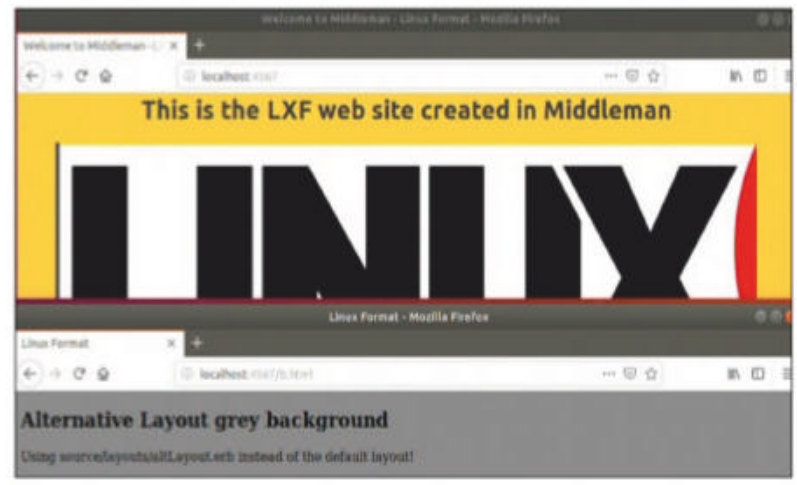
```
$ vi source/firstPage.html.erb
$ vi source/index.html.erb
$ bundle exec middleman build
```

For the purposes of this tutorial, we will replace the default *Middleman* logo with the *Linux Format* logo by putting the **LXF.png** file inside the **./source/images/** directory and referencing it from the index web page of the *Middleman* web site; this requires some small changes to the **index.html.erb** file. Basically, you delete the code that displays the *Middleman* logo and replace it with HTML code in order to reference the desired image file. By the way, the **erb** extension stands for Embedded Ruby – as we said earlier, this also means that you can embed Ruby code inside **erb** files! *Middleman* also supports plain HTML files.

The **firstPage.html.erb** page we created can be accessed as **http://10.0.2.15:4567/firstPage.html** from the test website. The `bundle exec Middleman build` command that was executed last is very important as it generates the final static HTML output based on the current version of your site. Among other things, the `Middleman build` command creates a new directory named **build** that contains all the static files of the website. As the **build** directory is recreated each time you execute `Middleman build`, you should be very careful and not put any new files in it or make changes to existing files inside the **build** directory.



The appearance of the index page of the Middleman blog website, as well as the format of a blog post. Below, you can see the look and the URL format of a blog post.



This shows the use of layouts in Middleman web sites. Defining multiple layouts and choosing the desired one is an easy process.

The screenshot (*see page 88*) shows the home page of the "lxf" website after the logo modifications we have applied.

Creating a blog

The easiest way to create a new blog using *Middleman* is to create a new project using the `--template=blog` command line parameter. So, for the purposes of this section, the blog website will be created using the `Middleman init myBlog --template=blog` command.

The `--template=blog` parameter tells *Middleman* that you want to use an official extension named *Middleman-blog* with support for blogging, articles and tagging. This also means that you will need to execute `gem install Middleman-blog` as root in order to install that Gem. The `--template=blog` parameter also creates the appropriate entry for *Middleman*-blog in the Gemfile of your *Middleman* project, which is very convenient.

This time the contents of the **source** directory are different from before as there are no additional directories in it – just six files named **2012-01-01-example-article.html.markdown**, **feed.xml.builder**, **layout.erb**, **calendar.html.erb**, **index.html.erb** and **tag.html.erb**.

The **2012-01-01-example-article.html.markdown** file is an example blog post. The filenames of the blog posts follow a strict rule: they should begin with the year (2012 in this example), followed by the month (01), followed by the date (01) and, last, followed by the name of the article (example-article). So, the date prefix (2012-01-01) of each blog post is compulsory. The **layout.erb** file is a template that defines the layout of the blog posts.

We might want to use Markdown for writing blog entries as it's easy to learn and easy to use. For example, adding a single HTML page and two articles to the blog using Markdown requires the execution of the following commands:

```
$ vi source/aFile.html.md
$ vi config.rb
$ middleman article LXF-issue-200
$ middleman article LXF-issue-210
$ middleman build
```

The changes to the **config.rb file** are made for putting the `set :relative_links, true` in order to use relative links – you might not need that if you are using images from an external site. The **aFile.html.md** file is not a regular blog post and will be not created as such – it just illustrates how to reference image files in Markdown. New articles in *Middleman* are created with the `Middleman article` command followed by the name of the article. So, the two `Middleman article` commands will generate two new files named **2018-11-24-lxf-issue-200.html.markdown** and **2018-11-24-lxf-issue-210.html.markdown** inside the **source** directory, which you should edit on your own.

The screenshot on page 89 shows the output of the previous commands. The screenshot on page 90 *(bottom left)* shows the appearance of the blog site after the changes we made to it, as well as the contents and the look of one of the blog posts. As before you will need to execute the `Middleman build` and `bundle exec Middleman server` commands first. The **index.html** file of the blog post is automatically updated when you add new posts.

As an exercise you can try adding two more blog posts to it on your own. You can find more information about *Middleman-blog* at **https://github.com/Middleman/Middleman-blog**.

Layouts

It's important to know how to make changes to the layout of a *Middleman* website by making modifications to the **source/layouts/layout.erb** file of the "lxf" site created previously. As **source/layouts/layout.erb** is a plain-text file, which is also the case with all *Middleman* files, you can edit it using your favourite text editor. Additionally, we are going to create a new layout that will be saved as **source/layouts/altLayout.erb**. One way to use that alternative layout on a *Middleman* page is to create a new **.html.erb** page with the following header:

```
---
layout: altLayout
---
```

The screenshot (*see the top of page 90*) shows the results of the changes we made to the default layout of the "lxf" website and how the alternative layout we created looks. Among other things, layouts in *Middleman* enable you to define the header and the footer of a page. Please feel free to look into the files, make changes to them and see how the changes you made affect the look and the contents of the site.

Deploying your web site

For the purposes of this tutorial all files will be deployed manually on a remote Linux server using SFTP. Once your static HTML files are on the remote computer, you should set up your web server in order to be able to see them from the internet. The image (*see right*) shows the output of the *scp* command as well as the output of the *wget* command used for testing the *Middleman* website. Remember that all you have to do is to copy the files into the build directory. You are now ready to create your own websites using *Middleman*! LXF

ABOUT MARKDOWN

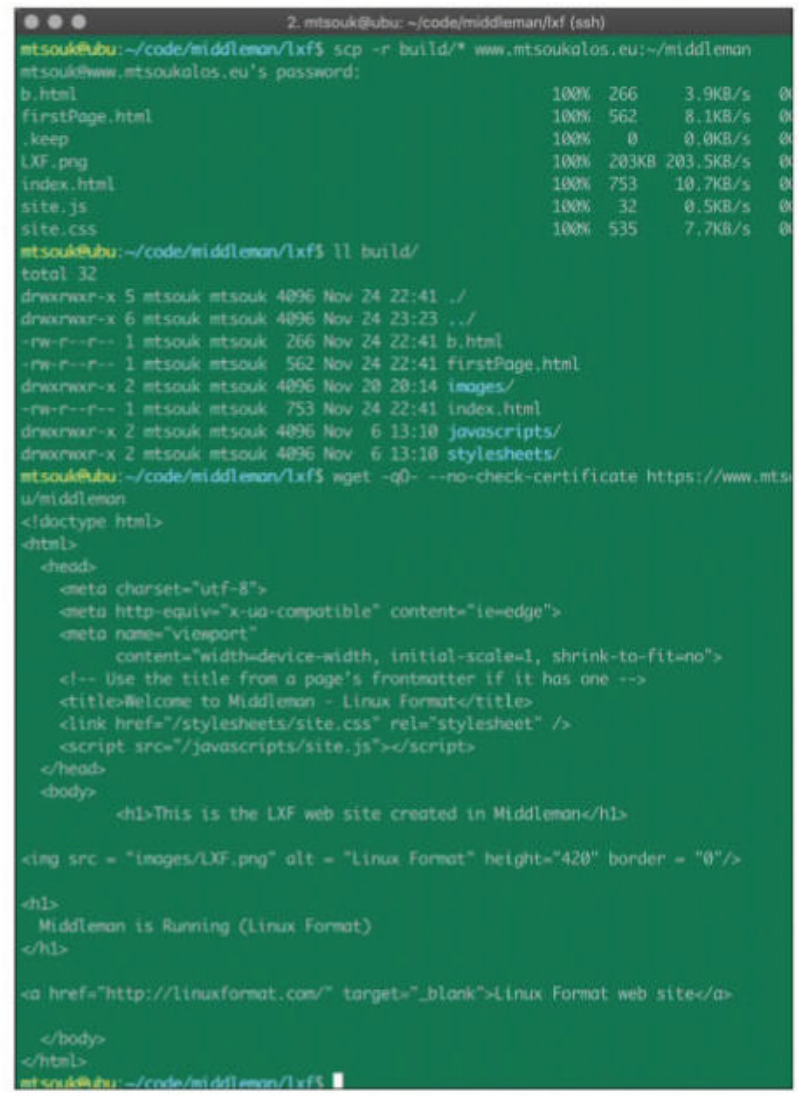
The Markdown language is a way of writing formatted text that can be easily converted into HTML or other formats. Markdown saves you from having to write HTML code, which is pretty ugly, and therefore can help you write posts and articles easily and quickly.

Beginning a new line with a single hash character will create an H1 HTML header. If you use two hash characters (##) at the beginning of a line, you will get an H2 HTML header, and so on.

In order to present a word or a phrase in italics you should write it as `_Italics_`. For bold, you should use `**Bold**`. You can include small code exempts using backquotes like `` `middleman build` ``, multiple lines of code using ``` at the beginning and ``` at the end, and shell commands by putting at least four space characters at the beginning of a line. Markdown also allows you to crate tables, but this is a little trickier. The following Markdown code will be formatted as a beautiful table with three columns and two rows as well as a header column:

```
| Left     | Right     | Centre |
|:---------|----------:|:------:|
|   1      |     3     |    5   |
|   2      |     4     |    6   |
```

The current version of Middleman uses the *redcarpet* Ruby Gem for parsing and converting Markdown to HTML. You can find more information about *redcarpet* at **https://github.com/vmg/redcarpet**. Last, bear in mind that when writing Markdown you can still include HTML code, and that you do not have to use every capability of Markdown just because it's available – work with whichever code feels most natural!



This output shows how you can manually deploy a Middleman website using SFTP and then test that your site is working as expected, as covered at the end of the tutorial.

QUICK TIP

In order to create your content you will need to learn a language; Markdown is a very good candidate as it is easy to learn and easy to use. Additionally, Middleman itself is responsible for converting a Markdown document to HTML.

JULIA

Discover the new language for science

Lee Phillips evangelises FOSS tools for science to anyone who will listen – and this time he thinks he's found a game-changer.

OUR EXPERT

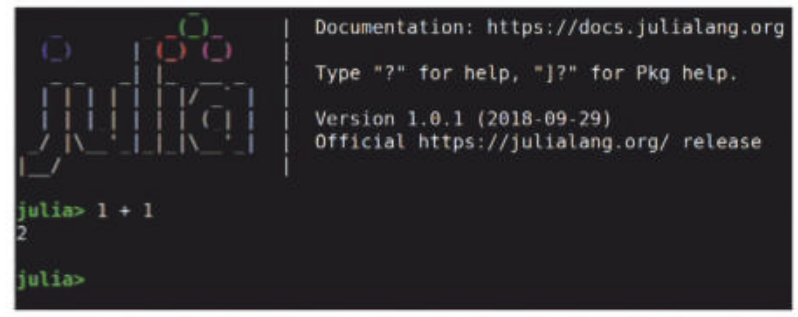
Dr. Lee Phillips is a physicist who worked for the US government for 22 years. His hobbies are writing popular science and computing articles, plus science education and outreach.

Early in 2012 a project emerged from the depths of MIT's computer science laboratories. It was a new computer language with the disarming name of Julia. Over the next six years, as the language, now developed in the open on GitHub, matured and accreted a large and widely varied constellation of libraries, it gained a devoted following of scientists, engineers and others who were involved with high-performance numerical computing.

This year the language reached version 1.0, and with that milestone comes a promise of long-term stability. Its new release saw a renewed surge of interest by research groups and the press, even including an interview on the US National Public Radio network.

But do we need yet another programming language? The hundreds of research groups and computational science teams that have adopted Julia seem to think so. The old standards, the statically typed Fortran and C, certainly do the job in regard to efficiency of runtime code. But they turn development into an edit-compile-run loop that can become tedious, and require (possibly implicit) type declarations for all variables. In addition, they provide few architectural devices for organising code, aside from subroutines and modules. These drawbacks have goaded many investigators in the fields of data science, bioinformatics, physics and other compute-intensive areas to turn to interactive, dynamically typed languages, especially Python. Here development is far more rapid, and its exploratory nature makes it more enjoyable.

However, these languages come with a severe performance penalty. To address this, scientists resort to a variety of strategies: rewriting the expensive parts of the code in C or Fortran; using a nonstandard language implementation, possibly with static typing, but offering only a subset of the full language; importing a library of numerical routines written in Fortran or C (*Numpy* for Python); or combinations of these. Although, for many problems, these strategies yield sufficient speed-ups, their use mires the programmer in what the MIT team calls the 'two-language problem': the need to maintain code in (at least) two languages, and attend to the interface between them. In large projects you may wind up with two teams with expertise in the different languages, and the concomitant management problems.



Here's the glorious Julia REPL, as it appears in the Linux terminal when you type 'julia'. The mastery of basic arithmetic reassures us that our Julia installation is working.

Julia intends to be a solution to the two-language problem. Its creators designed it to be as easy and fun to program in as Python, and as easy to read, but to perform comparably to Fortran or C. Judging by published benchmarks, and the reported experience of its growing cadre of users, it has largely succeeded in these goals. Julia's syntax and organisational principles also make the life of the scientific programmer easier, as we'll see below.

This tutorial will not make you a seasoned Julia programmer, but, after following along, you will have a decent grasp of the language, and be in a position to decide if you want to pursue it further and perhaps adopt it for some of your future projects.

A Julia of your own

If you use Linux, you can find Julia in your package management system. However, you probably don't want to install it from there, as it will quite likely be severely out of date. If you want version 1.0 (the current is 1.0.2), you must head over to the official Julia site and download sources or (more conveniently) a binary package. They'll set you up for Linux, Mac or Windows. Take a look at Essential Resources *(see page 93)* to find out where to go.

There's not much more to say about installation, because all the dependencies are included with the binary packages. After you unpack, navigate to the

QUICK TIP

Julia is full of shortcuts for expressive code. Instead of using the `collect` function, you can say `[a:b:c;]` to turn an iterator into an array.

provided **/bin** directory and run `julia`, or copy it to your path. If all is well, after you type `julia` at the terminal, you'll be presented with the interactive prompt. You are now in the Julia REPL and can begin typing Julia code.

Julia's basic data types

Before we get into the actual syntax, it will help to introduce Julia's fundamental data types.

First of all, we have numbers – many kinds of numbers! Of course, there are integers and floating point numbers. But we can also compute using integers with arbitrary precision, and, for you physicists and engineers out there, complex numbers. Using the latter is straightforward, once you know that `im` is Julia for the square root of -1:

```
julia> (1 + 2im) * (1 - 3im)
7 - 1im
```

One interesting twist that Julia provides is rational numbers, using the syntax `1//2`. These are stored as exact rationals, internally in their reduced form, so you can do, for example:

```
julia> 2//4 + 4//8
1//1
```

So much for numbers. Julia has strings, like all languages, but has the advantage of having grown up in the Unicode age. Strings in Julia are Unicode through and through, but it goes deeper than that: even the names of variables can use Unicode. This allows you to write code that looks more like maths. For example:

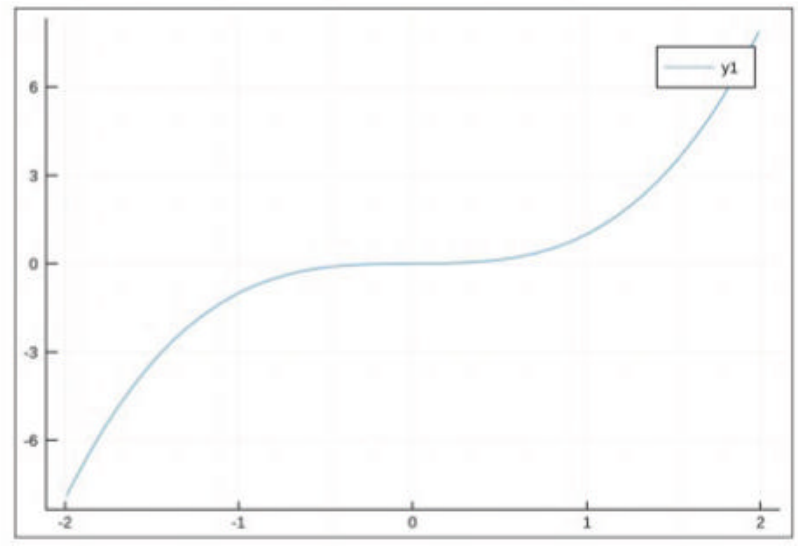
```
julia> a = 1; ß = 2;
julia> √(a + ß)
1.7320508075688772
```

If you don't have a convenient way to enter a Unicode symbol at the prompt, you can enter the LaTeX code for it and press Tab. Notice the semicolon ending the line in our REPL input above: this suppresses the output.

Now we come to the real workhorse in most Julia programs: arrays. Most scientific computing involves the manipulation of arrays, and Julia is replete with expressive syntax that makes working with these data structures convenient. The brief REPL session below shows how you can initialise arrays using square brackets, glue them together horizontally using a space, stick them together vertically using a semicolon, and take the transpose with a quote:

```
julia> a = [5 6];
julia> b = [7 8];
julia> [a b]
1×4 Array{Int64,2}:
 5  6  7  8
julia> [a; b]
2×2 Array{Int64,2}:
 5  6
 7  8
julia> [a' b']
2×2 Array{Int64,2}:
 5  7
 6  8
```

Some functionality that may not be used in every program is stored in separate libraries, to speed Julia's startup time. To this basic array syntax you can add all the usual linear algebra machinery for matrices (2D arrays) by loading the routines from the standard library that ships with Julia. This will get you various ways of factoring, calculating eigenvalues, eigenvectors, matrix inverses, and performing all the familiar manipulations on matrices.



A simple plot of $y = x^3$, courtesy of the Plots package.

Julia's array syntax can lead to very expressive code, freeing the numerical programmer from having to write loops when the algorithm can be expressed as operations over array elements. One powerful construct along these lines is the dot (.) operator. This extends a function or binary operator so that it operates elementwise over an entire array. For example, referring to the arrays in the REPL session above, `a .+ b` would get you the array `[12 14]`, which you can verify is the result of adding the two arrays together element by element. You can probably guess what `a .* b` will give you, but omitting the dot yields something very different: `a* b'` calculates the matrix product of the two arrays, which in this case is 83.

The dot operator is even more powerful than this. You can use it with any mathematical function to create one that operates over array elements. For example, `cos.(a)` returns the array `[0.283662 0.96017]`. You can do the same thing with your own, user-defined functions. All this may remind you of the map function provided by many modern languages, including Julia. In fact, the preceding cosine example gives the same result as `map(cos, a)`. But using the dot gets you something extra, besides more compact code.

When you chain together functions with dots, as in `sin.(cos.(a))`, Julia 'fuses' together the operations to create a combined operation that it applies elementwise, rather than applying the first one and passing the array result to the second one. This can

» ESSENTIAL RESOURCES

- Julia headquarters **http://julialang.org**
- Installation **https://julialang.org/downloads**
- Julia benchmarks **https://julialang.org/benchmarks**
- Libraries **https://juliaobserver.com**
- Documentation **https://docs.julialang.org/en/stable**
- About the language design: *Julia: A Fresh Approach to Numerical Computing* by Bezanson J, et al., SIAM Review 2015. doi:10.1137/14100067.
- Julia 1.0 announcement **https://julialang.org/blog/2018/08/one-point-zero**
- Syntax cheat-sheet **http://bogumilkaminski.pl/files/julia_express.pdf**
- *Plots* metapackage **http://docs.juliaplots.org/latest**
- Learning links **https://julialang.org/learning**
- Official package registry **https://pkg.julialang.org**

be a significant optimisation of time and space usage for complex operations on large arrays.

While unconventional in some ways, Julia's control structures are pretty standard. A `while` loop (note the local scope):

```
i = 1;
while i < 7
  print(i^2, "; ")
  global i = i + 2
end
```

A `for` loop, illustrating iterator syntax:

```
for n in 1:5
  print(2^n, "--")
end
```

You can iterate over collections, of course:

```
for e in ["first", "second", "third"]
  println(e)
end
```

These shows how to construct a `while` loop, a `for` loop using an iterator, and a loop over a collection. These are fairly self-explanatory, but, as with everything else in this tutorial, it is more illuminating to try the code out yourself at the REPL rather than merely reading it. Note that the indentation is not part of the syntax, but is added for readability. Since the whitespace is not significant, we need a way to indicate the end of a block, which in Julia is the `end` keyword. One unusual aspect of Julia's blocks is that they create a local scope, which explains why we need the `global` keyword in the `while` loop; without it, we get an error.

You define a named function using a block as well:

```
function f(x, y)
  return sin(x) + cos(2*y)
end
```

A single liner like this can be more compactly defined with the line `f(x,y) = sin(x) + cos(2*y)`, but the block form is what you'll usually need.

The package system

On one hand, it may be regrettable that every language these days has its own package system – and sometimes more than one. On the other hand, this seems to be an unavoidable aspect of modern software development. Julia's sophisticated package system is powerful and easy to use, being an intimate part of the language environment, rather than an add-on. Most importantly, it enables the software developer to ship projects with their dependencies and avoid the aggravating 'dependency hell' that has become all too familiar to most coders.

Here we'll focus on how to use the package system to add functionality to our Julia programs. For, fun as it is to program in this expressive language, there's no reason to reinvent the wheel if you need to program GPUs, solve differential equations, serve data through HTTP, parse JSON or analyse astronomical images – all of which are available, as are hundreds more, as third-party libraries that you can directly and automatically import into your Julia projects. You can browse the approximately 2000 packages maintained at the Julia Observer website *(see Essential Resources on page 93)*, but remember that this repository contains code at all stages of development, from polished and ready-for-prime-time to experimental works in progress.

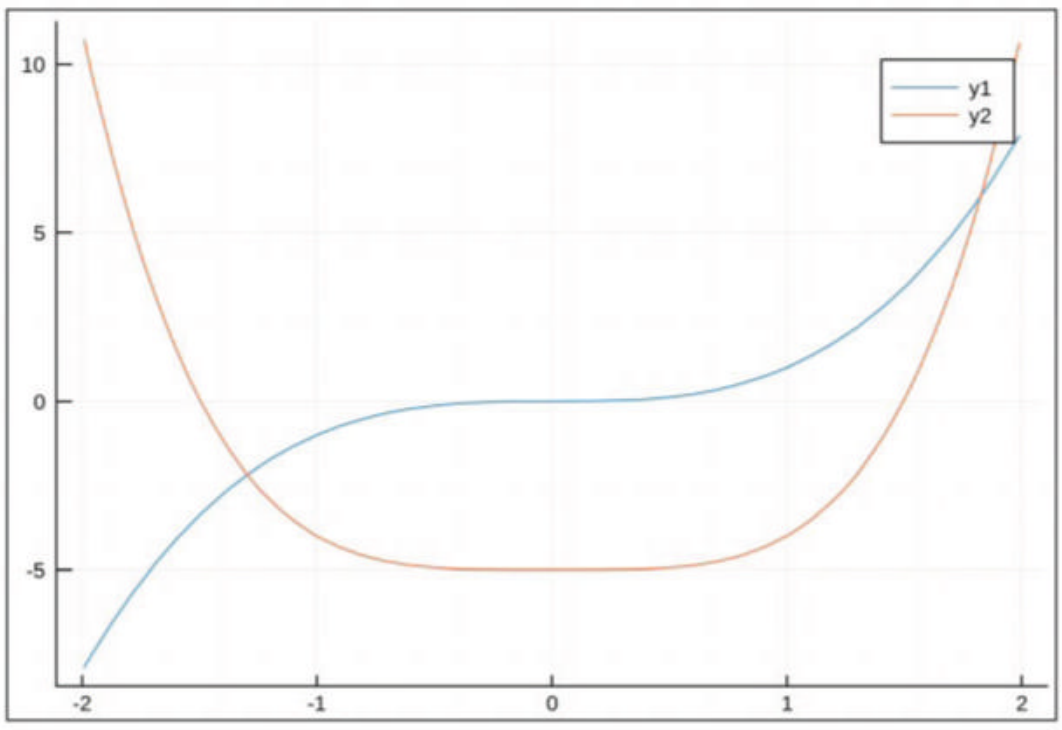
Before you can use the package system, you must import it, for it, too, is an external library. But it's part of the standard library, which means it comes with the system, and is already on your machine if you have a normal Julia installation.

To import it using the REPL or in your programs, the incantation is `using Pkg`. (And to use the linear algebra routines mentioned above, just say `using LinearAlgebra`.) After you do this, the package system's routines are imported into your local namespace, so you can invoke them directly. If you'd rather keep the imported names in their own namespace, use the command `import modulename` instead, after which you must invoke the imported routines as `modulename.function`.

After importing the package system, you can use its commands to fetch packages. By far the most important command is `Pk.add("packagename")`. If you don't already have the package on your system, this command will download and install it for you. The system keeps track of what versions you have installed, so if you have a version older than the one available in the official repository, you'll get an upgrade, or if you're up to date nothing will happen. In any case, you'll be kept informed, if you're at the REPL, with messages on the console. After doing this, you still need to import the package with the `using` or `import` command; the first time you do this after installing a new package, you will usually have to wait until the module compiles, and you will get compilation messages on the screen.

Here is where things may go wrong if the package is not compatible with your currently installed version of Julia. Unfortunately this is a frequent problem at the moment, since recent language versions introduced syntax changes that broke many packages. There seems to be a lot of repair work underway, however, and the promised stability of the 1.0 series means that you can use any currently working package with confidence.

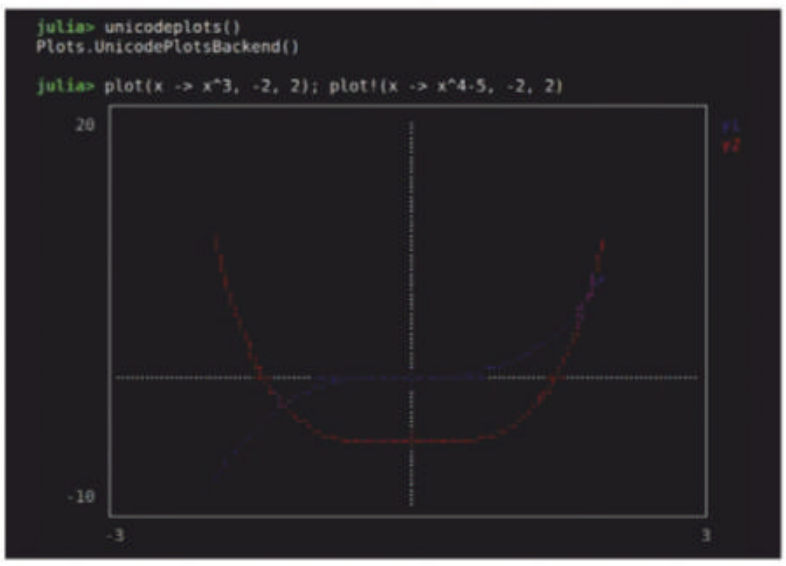
Building up a plot step by step.



Looking good!

The results of a research project in scientific computing usually involve plots and graphs. Fortunately, there are some excellent libraries in the official repository that make drawing graphs directly from Julia code easy, intuitive and which provide excellent results.

We'll now turn to some detailed examples showing how to use one of the most popular of these: the Plots



Colour plots right on the console – great for a quick look at data or functions, for example.

graphing package. Now that you know how to install packages, you might have guessed that the first step is to issue the command `Pkg.add("Plots")`.

The plot thickens

Plots incorporates a clever idea: instead of writing a graphing system from scratch for use from Julia, it implements a uniform interface to a number of plotting back-ends. This means that, without changing the plotting commands in your code, you can render your plots using an assortment of plotting libraries that create the actual output behind the scenes (including back-ends that have not yet been written).

Current back-ends include a very capable driver that uses Unicode symbols to make rough plots on the console, called *UnicodePlots*; *GR*, a well-established plotting system for 2D and 3D publication-quality graphs; *Matplotlib*, the highly popular Python plotting system; *Plotly*, which can make interactive plots for the web; *pgf*, an interface to the sophisticated PGF/TikZ drawing system for LaTeX; and more.

Most of these back-ends need to be separately installed and imported before use. If you try one and Julia complains that it never heard of it, you know what to do. To select a particular back-end, issue a command formed from its name, transformed to all lower case, as a function call with no arguments.

For example, to select the *UnicodePlots* back-end, enter the command `unicodeplots()`. To have Julia tell you the currently selected back-end, just say `backend()`. On our system, the default back end is *GR*. Entering the command `plot(x -> x^3,-2,2)`, we get the plot shown at the *top of page 93*. This example also illustrates the syntax for defining an anonymous function.

Each subsequent plot command will replace the previous one. If you want to add a new curve to an existing plot instead, use the form with an exclamation point. The command `plot!(x -> x^4-5,-2,2)` will now yield the plot *shown on page 94*. As we mentioned, the nice thing about the Plots library is that you can use the same plotting commands with different back-ends. After switching to the *UnicodePlots* backend with the command `unicodeplots()`, issuing the previous two plot commands will get you a plot on the console, shown in the screenshot *(top)*.

MULTIPLE DISPATCH

The central organising principle of a Julia program is based on the language's sophisticated type system, which includes user-defined types. You can define different versions of your functions that operate on different types of arguments, and the compiler will dispatch the version most specific to all its arguments. You organise your program around types and functions rather than classes, as when using an object-orientated language.

Every plotting back-end has its limitations. *UnicodePlots* for the terminal can't do 3D plots, for example, but the *GR* back-end can. In order to show how to do this, we first need to learn a little more about iterators, which we'll use to make the x and y coordinates.

```
x = collect(0:0.01:2ϖ)
y = x
```

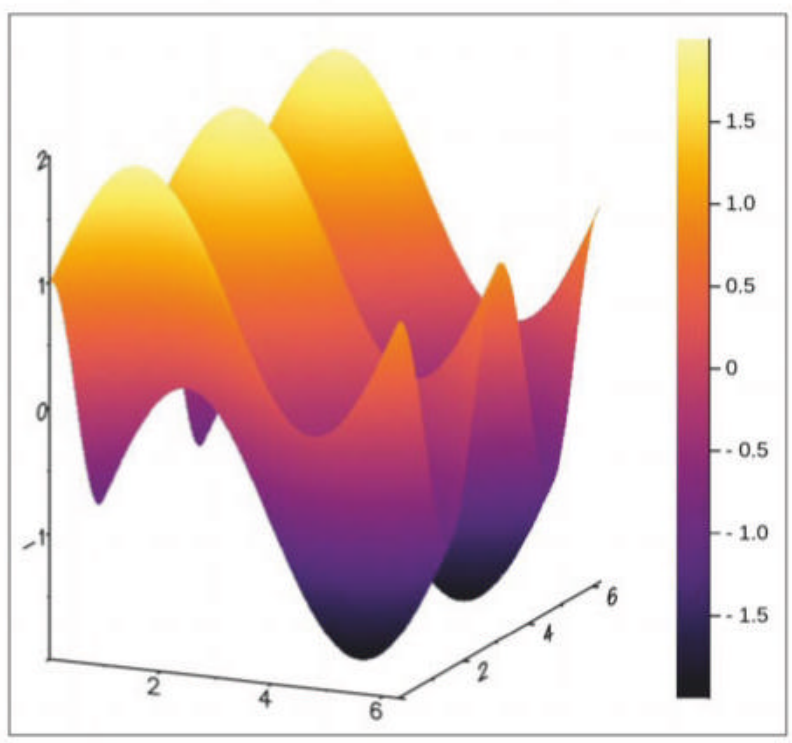
The bit above, inside the parentheses, is an iterator (an object that can be iterated over, for example in a `for` loop) that goes from 0 to 2*pi in steps of size 0.01 in this example. The `collect` function turns this iterator into an array containing all the elements that we would get if we stepped through the iterator; so x now contains 629 elements in this case.

Now that we have our x and y coordinate arrays, we can make a surface plot of, for example, our previously defined function `f(x,y)` just by saying `surface(x,y,f)`. The results can be seen below.

Have fun

We've only scratched the surface of Julia here, but this taste may be enough to encourage you to download this free software and try it out on one of your own projects. Julia is a sophisticated language that has some surprising features – for example, a macro system as powerful as Lisp's.

It may turn out to be the holy grail that number-crunching scientists have been pining for for decades: a single, versatile language that is as fast as Fortran while being as painless to develop in as Python. LXF



A surface plot using the GR backend and the Plots library.

QUICK TIP

Want to learn from Julia code written by experts? Study the source code of a package of interest, or of Julia itself, which is mostly written in Julia.

On the disc

Distros, apps, games, books, miscellany and more...

START HERE

USING THE LXFDVD

Using Linux for the first time can be very confusing. It'll be unlike anything that you've likely used before, especially if you're used to Microsoft Windows or Apple Mac OS.

Generally our DVDs are designed to be run directly, which is to say that when you first power on your PC (or Mac) it should "boot" from the DVD – so before Windows or Mac OS even starts to load – with Linux running directly from the DVD. This trick is known as a Live Disc. It enables you to try out the various versions of Linux without having to install or change anything on your PC. Just remove the DVD, restart your PC and it'll be exactly as you left it.

While many systems will boot off a DVD when it finds one, many will not. See below for the standard process for enabling booting from a DVD on various desktops and laptop PCs.

The alternative option is to locate the ISO file on the DVD and write this to your own USB thumb drive and attempting to run that. We recommend using *Etcher* from **https://etcher.io** that's available for Windows, Mac OS and Linux. Good luck!

BOOT THE DISC

Many PCs should boot automatically if they're turned on with a disc in the drive. If not, many offer an early Boot Menu accessed by tapping a key while powering up from cold: F9 (HP), F12 (Dell, Lenovo), F8 (Amibios) or F11 (Award BIOS). Alternatively, use the BIOS/UEFI to adjust the boot order to start with the optical drive. Again, this is accessed by tapping a key during power up, usually Del but sometimes F1 or F2. Some new UEFI PCs require access via Windows: holding Shift select its Restart option. If you're still having problems using the DVD visit: **www.linuxformat.com/dvdsupport**

Mac owners: Hold the C key while powering on your system to boot from the disc.

AN EASY DEAL DISTRO

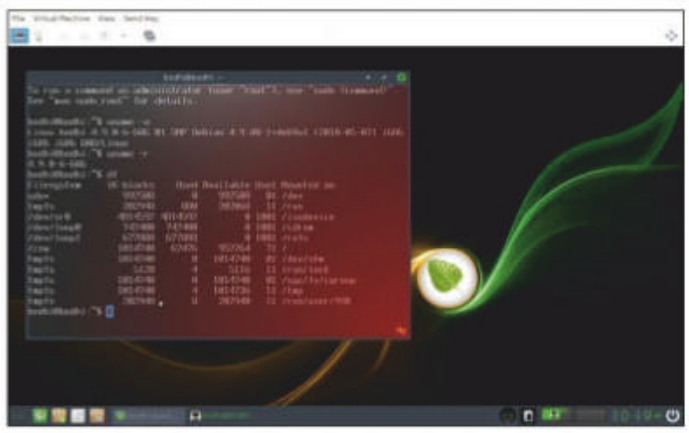
Bodhi 5.0

32-bit

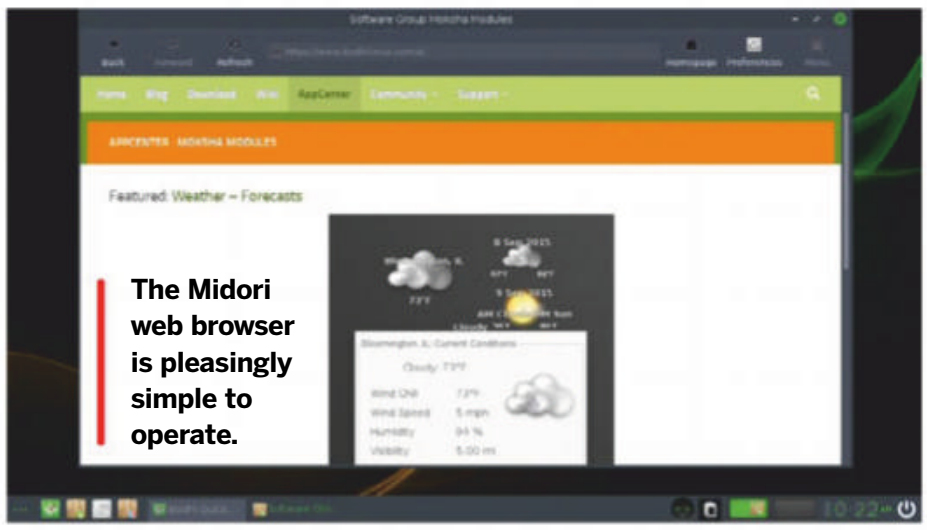
This isn't the first time we've featured Bodhi 5 on the disc, but it is the first time it's appeared in 32-bit form. The future of 32-bit Ubuntu isn't clear. All the official flavours elected to continue producing 32-bit media for 18.04, and well-known Ubuntu derivatives (most notably Linux Mint) continue to cater to the 32-bit crowd. With the release of Ubuntu 18.10 the 32-bit tide ebbed a little further away, with the Kubuntu, Mate and Budgie flavours no longer catering to these older CPUs, but let's focus on Bodhi.

With its stylish Moksha desktop, Bodhi has long been a popular choice for older hardware, and continues to be available in 32-bit form and will be supported until 2023. The legacy release of Bodhi uses the older 4.9 kernel, which will be just fine for older hardware. It's been marked a long-term release, so is being treated to Spectre & co fixes as they are discovered.

Moksha is a fork of the Enlightenment desktop, which despite having been around since 1997 is often overlooked, in part because Enlightenment releases are pretty disparate. Moksha is based on Enlightenment 17 (aka E17), which was in development for 12 years. It's predecessor, E16, continues to be developed and inspire the current version, E22. Bodhi and Enlightenment parted ways around the time E18 came about, which was seen as too resource intensive. Moksha retains the lightness and customisability of E17. It comes with its own lightweight update manager (eepdater) and the web-based Appcenter instead of a native App Store like Gnome Software. Terminology, Moksha's stylish terminal features two-dimensional splitting and an attractive visual bell.



Besides Cool Retro Term, Terminology is one of the most enjoyable terminal emulators around.



The Midori web browser is pleasingly simple to operate.

Bodhi has the honour of being our distro of choice for Netbook installs, thanks to its tiny footprint and thanks to Moksha working so well on low- resolution displays. The install option is lurking away in Applications> Preferences. Note that this distro may not install from the disc in UEFI mode (which you're unlikely to be using on a 32-bit machine), so either switch to classic BIOS mode, or write the Bodhi ISO to an optical medium or USB stick.

IMPORTANT NOTICE!

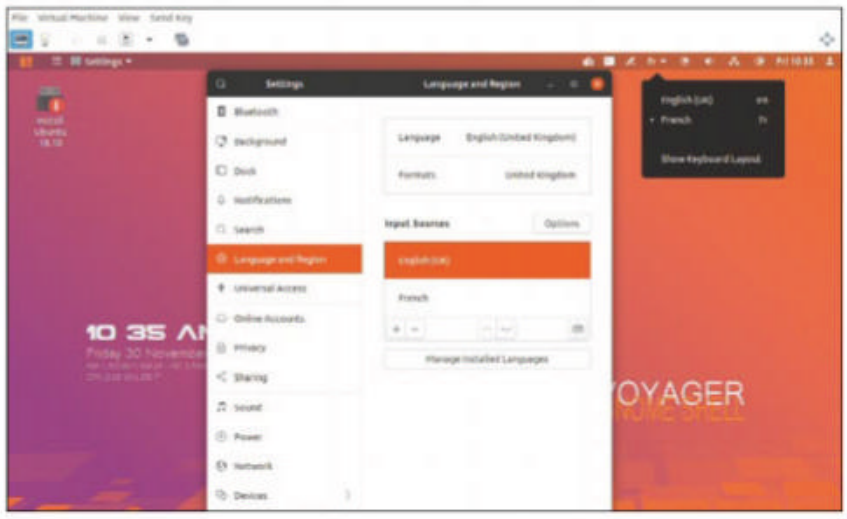
DEFECTIVE DISCS: For basic help on running the disc or in the unlikely event of your *Linux Format* coverdisc being in any way defective, please visit our support site at www.linuxformat.com/dvdsupport. Unfortunately, we're unable to offer advice on using the applications, your hardware or the operating system itself.

EURO DISTRO

Voyager 18.10 GE

64-bit

Voyager is a French distribution that comes in a variety of flavours. We featured the Debian release back in **LXF227**, which is its most conservative offering. There's also a release that's based on Xubuntu 18.04 LTS, a special GS edition of this customised for gaming, and this particular GE offering, based on Ubuntu 18.10 and the latest Gnome 3.30. Basing on an intermediate release of Ubuntu gives Voyager GE a short (nine month) lifespan, but does avail it of all the fixes that took place since Gnome 3.28, which features in the LTS release. In particular, the new Gnome should have be much more considerate about resource usage. When Ubuntu 18.10 reaches EOL, you should be able to seamlessly upgrade Voyager GE to a 19.04 base.



Voyager enables you to marvel at Ubuntu's new Yaru theme and improvements to Gnome's resource usage.

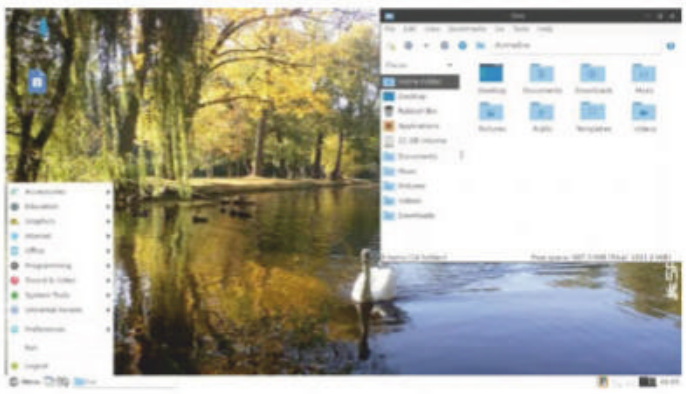
Despite Voyager shipping with several locales (regional settings), which can be chosen from the More Languages boot menu, when Gnome starts the keyboard layout will still be set to French. The localisation options at boot set the console keymap and font, but unless you like your Qs and As, and your Zs and Ws transposed, you'll want to go to Settings>Language and Region>Input Sources and add your preferred keymap. You can then activate it from the applet that appears in the corner.

STRONG AND STABLE

Sparky Linux 4.9

64-bit

Sparky is a regular on our DVDs, and keen-eyed readers may wonder why we're putting version 4.9 on the disc when we ran version 5.3 back in **LXF239**. The answer is because the 4.x series is based on Debian Stable and uses the venerable LXDE desktop, whereas the 5.x series are rolling releases based on Testing and use the LXQt desktop. Version 4.9 was released back in mid-November, which is when we wrote these pages, so it seemed a good idea to include it this time around. The rock-solid Debian 9 foundation reduces the chance of you running into bugs, at the cost of having slightly older packages.



If you don't care about new-fangled software, Sparky 4.9 is a great choice for older machines.

Sparky is an ideal distro for older systems. LXDE is easy on system resources, partly because it's built on the older GTK2 toolkit, partly because of the Openbox window manger, but mainly because it doesn't try to be fancy. It's a great distro for beginners too, so if you want to get into Linux and have an old machine lying around, why not Sparky it up? **Note:** that Sparky can't be installed from our DVD. To install it, burn the ISO file in the Sparky directory to a DVD or write it to a USB stick. LXF

NEW TO LINUX? START HERE

We answer your questions, insert the DVD and open index.html

- **What is Linux? How do I install it?**
- **Is there an equivalent of *MS Office*?**
- **What's this command line all about?**

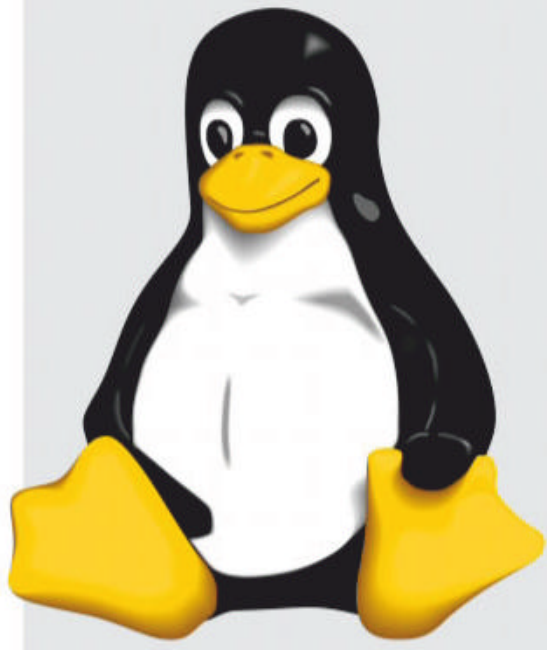
DOWNLOAD YOUR DVD IMAGES!

Get code and DVD images at:
www.linuxformat.com/archives

AND MORE!

READING MATTER

- **Advanced Bash Scripting Guide**
  Go further with shell scripting.
- **Bash Guide for Beginners**
  Get to grips with the basics of Bash scripting.
- **Bourne Shell Scripting**
  First steps in shell scripting.
- **The Cathedral and the Bazaar**
  Eric S Raymond's classic text explains the advantages of open development.
- **The Debian Book**
  Essential guide for sysadmins.
- **Dive Into Python**
  Everything you need to know.
- **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**
  Overview of GNU tools.
- **GNU Emacs Manual**
  Six hundred pages of essential information!
- **Producing Open Source Software**
  Everything you need to know.
- **Programming from the Ground Up**
  Take your first steps.



Rock-hard Linux Mint 19.1

We take you through installing and hardening the latest release of world's favourite green distro.

Open source photography

We explore the best tools to use, to process shots taken on everything from your D-SLR to your LineageOS phone.

Cryptography explained!

Get inside the software, technology and maths behind the smarts that secure the web (until governments break it).

Easy game development

Create your own games with GDevelop, a games engine development suite. Are you ready, player one?

Home banking

It's time to balance your books the FOSS way, as we get you back in the black using the best Linux home finance software.

Contents of future issues subject to change – we'll still be stockpiling potatoes for Brexit.

LINUX FORMAT
The #1 open source mag

Future Publishing Limited,
Quay House, The Ambury, Bath, BA1 1UA
Email linuxformat@futurenet.com

EDITORIAL
Editor Neil Mohr
Editor in Scotland Jonni Bidwell
Art editor Efrain Hernandez-Mendoza
Operations editor Cliff 'splitter' Hope
Group editor in chief Graham Barlow
Senior art editor Jo Gulliver
Editorial contributors
Mats Tage Axelsson, Michael Bedford, Neil Bothwick, Christian Cawley, Matthew Hanson, Keith Edmunds, John Lane, Jon Masters, John Knight, Jamie Munro, Nick Peers, Lee Philips, Les Pounder, Mayank Sharma, Shashank Sharma, Valantin Sinitsyn, Alexander Tolstoy, Mihalis Tsoukalos
Cartoons Shane Collinge
Cover illustration magictorch.com

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 Licensing team to discuss partnership opportunities.
Head of Print Licensing Rachel Shaw
licensing@futurenet.com

SUBSCRIPTIONS & BACK ISSUES
Web www.myfavouritemagazines.co.uk
Email *linuxformat@myfavouritemagazines.co.uk*
UK 0344 848 2852
International +44 (0) 344 848 2852

CIRCULATION
Head of newstrade Tim Mathers

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
Chief content officer Aaron Asadi
Editorial director William Gannon
Brand director Andy Clough
Head of art & design Rodney Dive
Commercial finance director Dan Jotcham
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 9001

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