GROUPTEST: TOP BACKUP TOOLS

www.linuxuser.co.uk

Linux User & Developer™

THE ESSENTIAL MAGAZINE FOR THE GNU GENERATION

BUILD THE PERFECT NETWORK

Plan & Design ✔ Set up & Configure ✔ Monitor & Manage ✔

IN-DEPTH GUIDE TO
Qubes OS
The most secure distro in the world

SERVAL PROJECT
Life-saving disaster response networks

EXPERT PI PROJECTS
> pyCLI: Build a Pi controller
> PiServer: Manage many Pis

INTERVIEW
Fixing complexity
Puppet's plan to simplify the toolchain for DevOps

51 PAGES OF GUIDES
> Python: Parallel programming
> Security tools: Build an arsenal
> Ubuntu Server: Lockdown Linux

New Netrunner
Smart and snappy with a custom Plasma desktop

R programming primer
Make beautiful data simulations with this matrix-based language

ALSO INSIDE
» Pi rocket panel
» Kernel update
» GNU Make tips

FUTURE

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

Linux User & Developer
THE MAGAZINE FOR THE GNU GENERATION

Future PLC Quay House, The Ambury, Bath BA1 1UA

Editorial
Editor **Chris Thornett**
chris.thornett@futurenet.com
01202 442244
Designer **Rosie Webber**
Production Editor **Ed Ricketts**
Editor in Chief, Tech **Graham Barlow**
Senior Art Editor **Jo Gulliver**

Contributors
Joey Bernard, Neil Bothwick, Christian Cawley, Alex Cox, Nate Drake, John Gowers, Toni Castillo Girona, Jon Masters, Paul O'Brien, Arsenijs Picugins, Calvin Robinson, Mayank Sharma

Advertising
Media packs are available on request
Commercial Director **Clare Dove**
clare.dove@futurenet.com
Advertising Director **Richard Hemmings**
richard.hemmings@futurenet.com
☎01225 687615
Account Director **Andrew Tilbury**
andrew.tilbury@futurenet.com
☎01225 687144
Account Director **Crispin Moller**
crispin.moller@futurenet.com
☎01225 687335

International
Linux User & Developer is available for licensing. Contact the International department to discuss partnership opportunities
International Licensing Director **Matt Ellis**
matt.ellis@futurenet.com

Subscriptions
Email enquiries **contact@myfavouritemagazines.co.uk**
UK orderline & enquiries **0888 888 8888**
Overseas order line and enquiries **+44 (0)8888 888888**
Online orders & enquiries **www.myfavouritemagazines.co.uk**
Head of subscriptions **Sharon Todd**

Circulation
Head of Newstrade **Tim Mathers**

Production
Head of Production US & UK **Mark Constance**
Production Project Manager **Clare Scott**
Advertising Production Manager **Joanne Crosby**
Digital Editions Controller **Jason Hudson**
Production Manager **Nola Cokely**

Management
Managing Director **Aaron Asadi**
Editorial Director **Paul Newman**
Art & Design Director **Ross Andrews**
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
ISSN 2041-3270

We are committed to only using magazine paper which is derived from responsibly managed, certified forestry and chlorine-free manufacture. The paper in this magazine was sourced and produced from sustainable managed forests, conforming to strict environmental and socioeconomic standards. The manufacturing paper mill holds full FSC (Forest Stewardship Council) certification and accreditation

Welcome

to issue 189 of Linux User & Developer

In this issue

Welcome to the UK and North America's favourite Linux and FOSS magazine. Somewhat appropriately, I experienced an earthquake while writing up this month's inspiring open source piece on open source mesh networks being used for disaster relief (p34). Fortunately, it was only 4.3 on the Richter Scale.

But this month there are plenty of other things to get excited about as we devote 14 pages to setting up a network suitable for advanced home or small-business use (p18).

If security and privacy are more your concern, you'll enjoy our guide to building Qubes OS from scratch, with its bare-metal hypervisor-based compartmentalisation measures (p60).

Meanwhile, in tutorials we have a standalone primer on the wonders of statistical programming with R; server hardening with Nate Drake; and in Pi tutorials we have 'Mr. ZeroPhone' himself, Arsenijs Picugins, with a guide to monitoring your Pi's system parameters with a press of button with pyLCI. Enjoy!

Chris Thornett, Editor

Contents

OpenSource

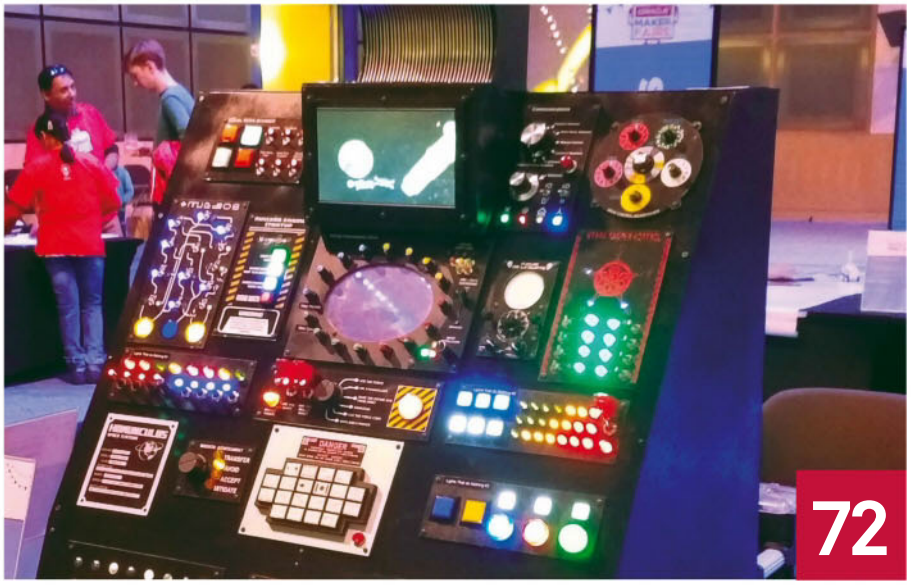
InspireOS

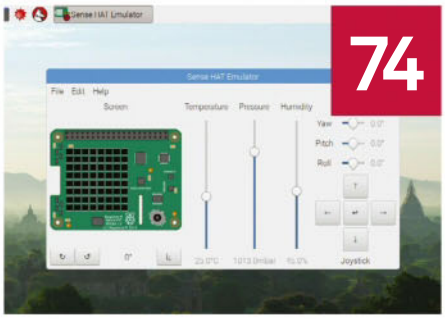
Features

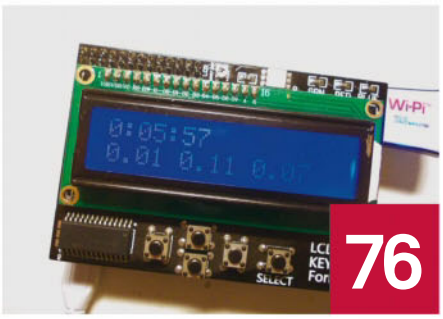
Tutorials

LinuxUser & Developer
Issue 189
March 2018
facebook.com/LinuxUserUK
Twitter: @linuxusermag

FileSilo
94 Free downloads
We've uploaded a host of new free and open source software this month





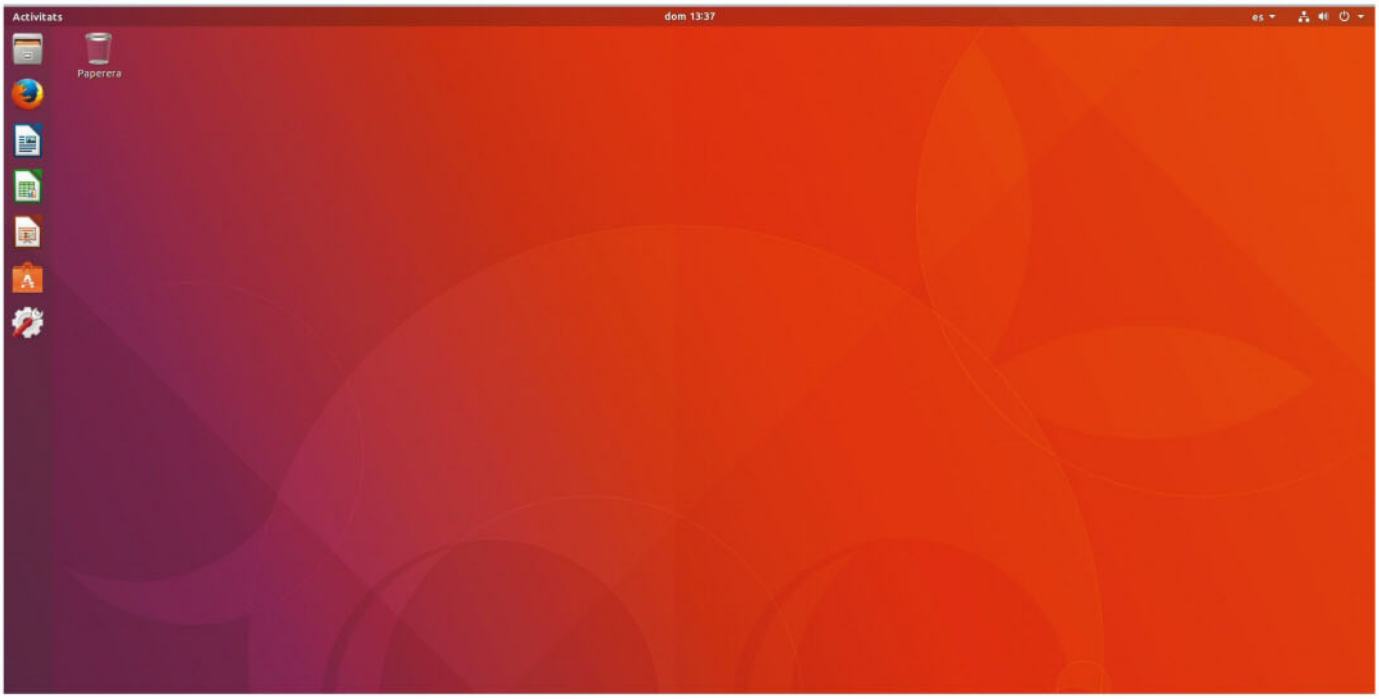


Practical Pi

Reviews

Back page

OpenSource



UBUNTU

Ubuntu: Wayland dropped as default

With complaints over stability and screen sharing in Wayland, it seems X.Org's just better suited for the job

Forthcoming Ubuntu release 18.04 LTS 'Bionic Beaver' will abandon the Wayland display server and revert to X.Org, it has been announced. A blog post from Ubuntu desktop engineering manager Will Cooke revealed that both the traditional X.Org graphics stack and the Wayland-based stack will be included. However, in a massive reversal, it was underlined that X.Org will be the default – for several reasons.

Bearing in mind that the LTS has a five-year support life, and therefore needs to be working out of the box, Cooke cited three key deciders: screen sharing on services such as WebRTC, Google Hangouts and Skype; remote desktop with RDP and VNC; and recovery from shell crashes. These all work better with X.Org than with Wayland. Given that Canonical increasingly sees Ubuntu as a business-ready operating system, it makes sense that they would want to ensure its stability with these functions.

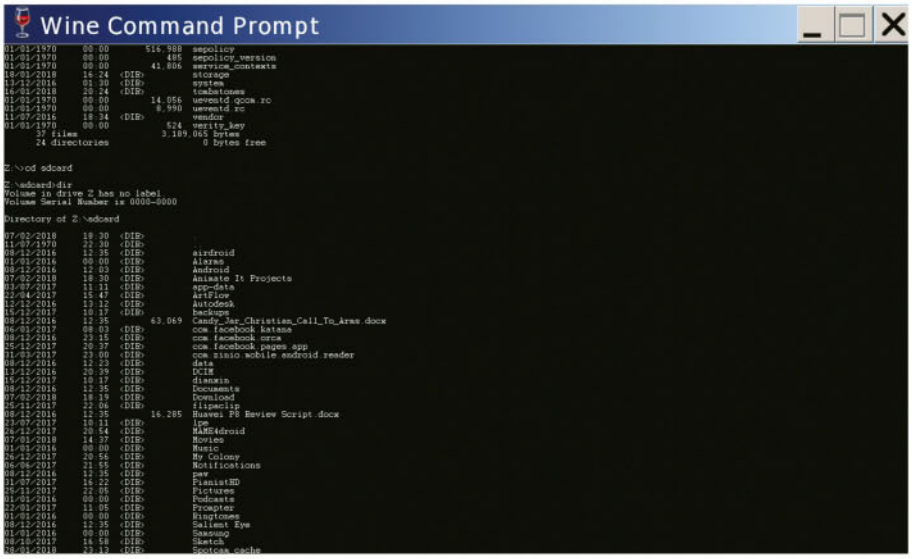
Explaining the first two, Cooke observed that while the Wayland/GNOME plan is to use Pipewire for screen sharing, further development is required. "Until that happens, Xorg [sic] is necessary for people who need screen sharing features," he wrote.

With regards to recoverability, and the fact that a GNOME Shell crash can "end your whole session, killing running applications and returning you to the login screen," he clarified: "When using Xorg, the shell can restart independently of the display server and running applications. This means that once the shell is restarted, you can pretty much pick up your session from where you left off, with your applications still running."

There is also some question over the stability of Wayland when it comes to games. While open source drivers work fine with Wayland for standard desktop activities, proprietary drivers are less reliable and, unfortunately, they offer far better graphics performance in terms of frame rate, textures and so on. The solution to this problem is to switch to X.Org and use proprietary drivers.

But it's not the end for Wayland, Cooke added. "There are two solutions to this problem when using Wayland: make sure the shell doesn't crash or change the architecture. Both of these are work[s] in progress and we continue to contribute to this work upstream. GNOME Shell 4 will bring a new architecture... In short, we remain committed to GNOME and the GNOME stack and will continue to actively contribute to Wayland by adding features and fixing bugs."

As Cooke noted: " [...] the Ubuntu experience needs to be stable and provide the features [users] have come to expect and use in daily life [...] For 18.10 we will re-evaluate Wayland as the default." Ultimately, reverting to X.Org for 18.04 LTS is little more than a hiccup on the road to Wayland's eventual adoption by all major Linux distros.



SOFTWARE

Wine updated with support for Direct3D 10/11

Windows compatibility layer also adds Android graphics driver

The Wine team has announced the release of Wine 3.0, the favoured option for running Windows apps and games on Linux, which brings with it a number of long-awaited improvements. Among these are Direct3D 10 and 11 support, the Direct3D command stream, the Android graphics driver, and improved DirectWrite and Direct2D support.

Noting the lack of some other hoped-for additions, the Wine team clarified that "because of the annual release schedule, a number of features that are being worked on have been deferred to the next development cycle. This includes in particular Direct3D 12 and Vulkan support, as well as OpenGL ES support to enable Direct3D on Android."

Most significant for PC users, particularly gamers, is the Direct3D 10 and 11 support, which includes compute shaders, tessellation shaders, depth bias, multi-threaded command streams, support for more graphics cards, and improvements to OpenGL. There are also improvements to HiDPi scaling and memory management.

All of this means that you can expect improvements in the performance of Windows software running on Linux, as well as a wider selection of apps and games. Wider support for more recent games is another benefit, something that will certainly attract anyone hanging onto a Windows dual-boot partition. Wine 3.0 is available for free now, although improved support can be enjoyed if you opt to wait for CrossOver to adopt and polish it.

As important as the release of Wine 3.0 is for gamers, it's also quite a big deal for Android. The inclusion of the Android graphics driver means that an APK version has been made available from **www.winehq.org**. Available for ARM and Intel-based Android devices, Wine 3.0 can be easily installed on Android as long as 'unknown sources' is enabled. At this stage only a small number of tools work, with wider support on x86/x64 devices than on ARM. However, the possibility of running legacy Windows software on Android tablets is fascinating.

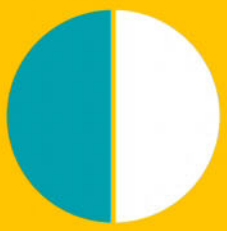
DISTRO FEED

Top 10

(Average hits per day, 30 days to 11 Feb 2018)

1.	Manjaro	▼	3438
2.	Mint	▼	2955
3.	Ubuntu	▲	1690
4.	Debian	▲	1675
5.	elementary	▲	1285
6.	Solus	▲	1249
7.	Antergos	▼	1088
8.	TrueOS	▲	1081
9.	Fedora	■	967
10.	openSUSE	■	906

This month



In development (5)
Stable releases (5)

The usual suspects occupy the top 10 list, although it's worth noting that ArchLinux has dropped to 12th place; rival Manjaro remains at number 1.

Highlights

ArchLinux
One of the most well-known alternatives to the Debian family, ArchLinux has its own package manager and is available for x86, x64 and ARM devices.

Manjaro Linux
Manjaro is a more powerful interpretation of Arch, with automatic hardware detection, multiple kernel support and desktop configurability.

Archman GNU/Linux
Although based on Arch, this version eschews pacman in favour of Octopi, with the intention of making software installation easier. It's lightweight, too.

HARDWARE

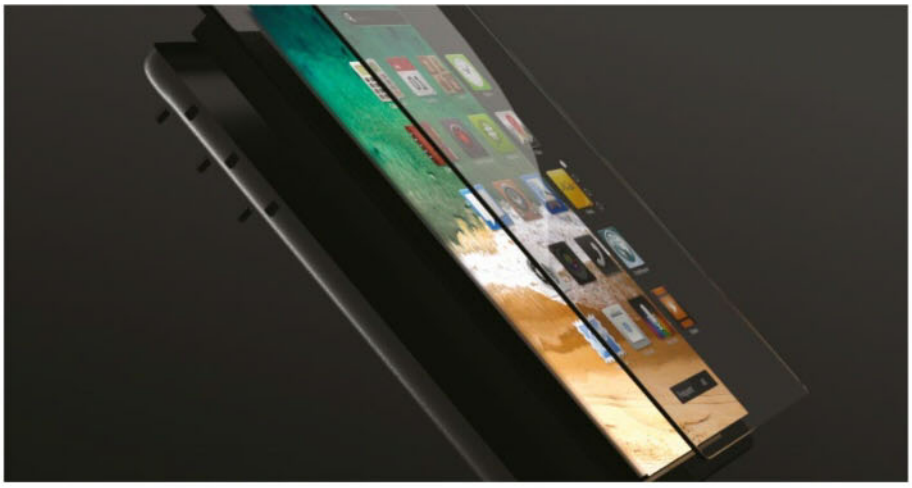
Purism breaks silence on the Librem 5

Regular updates promised for the phone project

Following the announcement of Purism's new security and privacy-focused Linux smartphone, Librem 5, in August 2017, news about its progress has largely disappeared. That's about to change. In a blog post in late February, Mobile Development Lead Nicole Faerber revealed that the team has been expanded, with 15 new roles filled as of January 2018. Some volunteer roles are also expected. More importantly, weekly blog posts will provide progress reports, an alternating weekly focus on the hardware and the UI. But what has been happening to prompt the Librem 5 team's radio silence?

Most significant is that the NXP i.MX 6 system-on-chip (SoC), used in early evaluation for the project, is unsuitable for use in a phone. "The most important feature of the i.MX6 was that it is one of only a handful of SOCs supported by a highly functional free software GPU driver set, the Etnaviv driver," Faerber wrote. "However, work with the i.MX6 showed us that it still uses quite a lot of power so when put under load it would drain a battery quickly, as well as warm up the device."

> "User interface and user experience work is also underway"



Above The Librem 5 promises to revolutionise smartphone security

Although disappointing, it seems that the team has identified a suitable replacement, the NXP i.MX 8M SoC, which Faerber describes as "currently the most likely candidate." Meanwhile, AARCH64 architecture – also known as ARM64, the 64-bit version of ARM – is planned, and development without the intended SoC continues with the i.MX 6 Quad Plus board. User interface and user experience work is also underway, with the display expanded to be between 5-5.5 inches with full HD resolution (1920x1080p). Keep an eye on **https://puri.sm/posts** for more.

SOFTWARE

Softmaker Office adds Microsoft-style ribbon

2018 version touts extra features, but still isn't open source

Writing and managing documents, spreadsheets and presentations on Linux can be tricky for anyone who has recently migrated from Windows. Microsoft Office won't run without Wine; LibreOffice, for all its strengths, doesn't look like Microsoft Office.

One solution is to try a different office suite, and Softmaker Office – the 2018 version of which has just been released – could be the answer for many people. As well as adopting the DOCX, XLSX and PPTX file formats by default, Softmaker Office 2018 also introduces a Microsoft Office-style ribbon menu. First seen in Microsoft Office 2007, the ribbon is intended to ease access of rarely used tools. In Softmaker Office it has a secondary purpose, however: to 'ease in' newcomers to office tasks on Linux.

Available in 32-bit and 64-bit versions, Softmaker Office 2018 features the TextMaker word processor, PlanMaker spreadsheet, Presentations for presentation design, and add-ons for Mozilla Thunderbird, which is also included.

Various other enhancements are also featured. These include a tabbed user interface, enabling the use of multiple documents within the same app window. For presentations, 2D and 3D slide transitions can be used. Integration with Mozilla Thunderbird brings some productivity boosts, such as enhanced calendar management. The address book can also be imported into TextMaker.

You'll find a 30-day trial version at **www.softmaker.com**, while the standard package costs £60 for three household devices. A 'professional' suite can be bought for £90, with five home licences included.

OPINION

Security: it's time to make the shift left

The merging of development and operations has left security concerns lagging behind – **Paul Farrington** suggests a number of ways to bring it back in line

The staggering growth of the application economy has put unprecedented pressure on development teams. Missed delivery deadlines can result in lost revenues, while poor functionality can impact customer loyalty and retention. To meet the demands of a faster time to market, many organisation are adopting DevOps. This burgeoning philosophy shifts left the responsibility for ensuring stability and security of an application throughout the entire lifecycle – including production and customer usage – to include developers.

Evolution of 'shift left'

The shift left into development has not only impacted the role and responsibility of developers, but has transformed the entire software life cycle.

In the more traditional approach to software development, Waterfall, there would be handoffs at each stage of the software development life cycle (SDLC): from planning, development, quality assurance and operations. With knowledge lost during the silos, operations issues would never be fed back down to developers; to the same end, business intent would never make it up the chain to operations.

Agile development further aligned business intent and application knowledge with the product owner, developers and quality assurance on the same team. However, it was still someone else's problem to operate the software. DevSecOps now supports continuity across the spectrum; the team now must be responsible for what is written and deployed. As a result, developers are starting to think differently about what they're building and how they're building it. The result for software delivery has been lower waste, fewer errors through automation and greater empathy for demands on the teams across the SDLC.

The benefits of 'shift left' testing have been understood by the software development industry for a long time, including higher confidence in the delivered product, higher customer satisfaction and reduced product developments costs, but security has been a latecomer to the 'shift left' methodology. However, with software and application-related data breaches and cyberattacks heavily plaguing the industry over that last five years, it is clear that security can no longer be viewed as an inconvenient bottleneck, and must become an integral part of the build process. To shift left, it is crucial to bring development, operations and security together with a culture of quality improvement. Here are a few suggestions to help DevOps teams make the shift.

Fail quickly, using automation Build testing directly into the DevOps process through automation. Save time by failing tests as early in the DevOps pipeline as possible.

Integrate app security into your dev tools Integrating security helps reduce friction. Ensure your security assessments integrate with your IDE and build and ticketing systems that automatically test code and coordinate remediation.

Fix flaws as you go Give developers the tools to find and fix coding errors as they write code, such as developer sandboxes, 'as-you-type' static testing, and eLearning.

Build security champions Make those developers with an interest in security your security champions, who can help reduce culture conflict between development and security by promoting the security message on a peer-to-peer level.

Paul Farrington

Paul is a Manager for the EMEA Solution Architects at Veracode.

> "Give developers the tools to find and fix coding errors as they write code, such as developer sandboxes"

Don't stop for false alarms Don't put up with application security solutions with a high false-positive rate – especially as false alarms could prevent a critical business function from being deployed.

Extend application security into production Application security can't stop after deployment. Similar to other aspects of DevOps, a well-engineered solution must enable closed-loop feedback from production for any subsequent security incidents.

Provide operational visibility DevOps promotes team autonomy, but make sure operations and security have the required visibility to measure and assess teams for compliance and risk.

Start making the shift By taking the decision to ensure that security is integrated across the entire SDLC from the outset, you will enable your team to deliver more secure software – and faster.

COMMENT

Your letters

Questions and opinions about the mag, Linux and open source

TOP TWEET

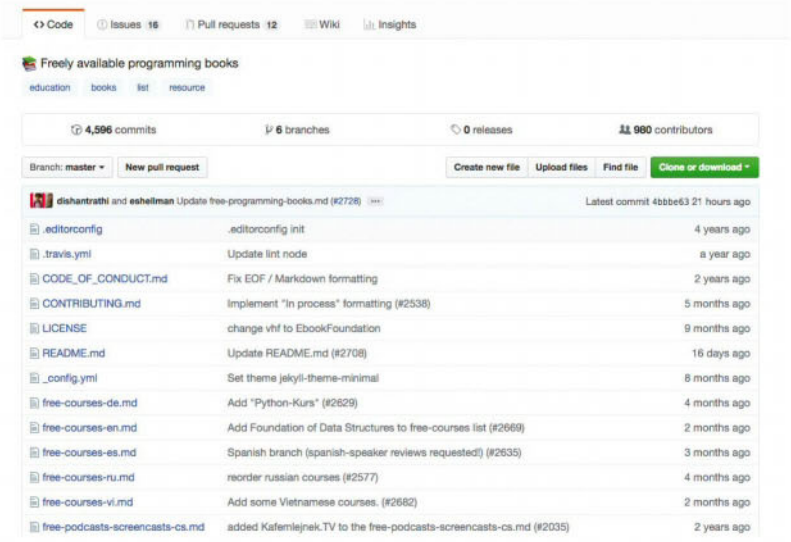
A nice message from @arribada_i: A wonderful write up by @LinuxUserMag looking at the story behind the Arribada Initiative's drive towards open conservation technology. Shout outs to @institute_irnas @AudioMoth @OpenAcoustics and the Principe Trust too.

I heart pcDuino

Dear LU&D, You've asked readers about their interests so I thought I'd mention my own. I'm running Linux Mint having given Windows 10 'the flick' as it was gobbling up all my meagre bandwidth with updates every time I connected to the internet. I'm also interested in Knoppix and really small SBCs like the pc3Duino Nano, having had success with the Arduino UNO. Unfortunately, it looks like the Raspberry Pi is going to win out in the popularity stakes and I will have to switch over, which is a shame because the Duino has on-board analogue/digital converters. Data logging from sensors is what I'm working on, so low power consumption and an 'all in one' standalone black box is my aim.

Tom Hartley

Chris: Thanks, Tom. You're in good company with Linux Mint. It seems that Mint has taken over as the mainstream distro of choice. I think our small board computer round-up (See Features, p60, **LU&D187**) demonstrated that there's a big ol' world out there for the adventurous. I must admit I've been mucking about with a Pi myself making a radio and weather station with the youngster since the Christmas break. I've also got the Particle Electron 3G set aside for some IoT playtime in the near future. Its hardware design is entirely open source (**https://github.com/particle-iot/electron**).



Above If you can't afford to buy a programming book, there are plenty available for free – and nice people like the Free Ebook Foundation curate them for you on GitHub

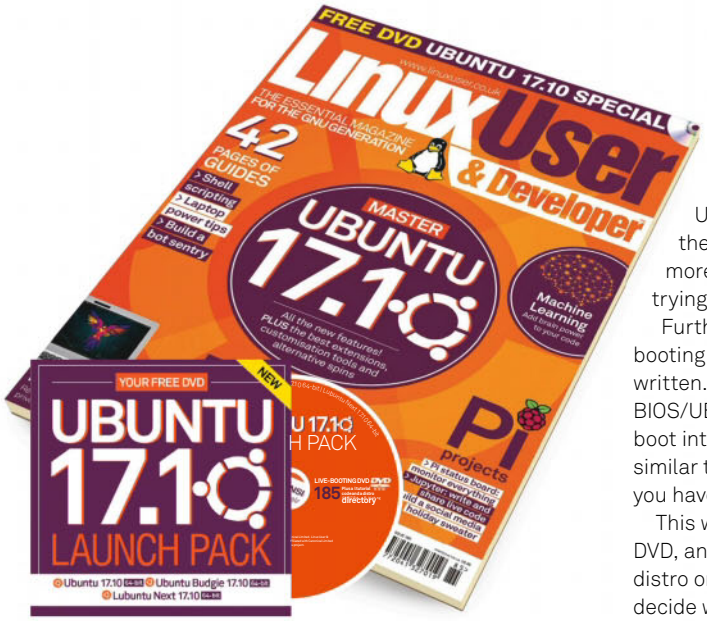
Above The Electron Particle 3G comes with its own SIM card and a plug-in system for extensions

Developing dev skills

Dear LU&D, I'm a new Linux user and I just want to create a simple application and I don't know where to find tutorials. I'm looking to design an application such as a music player or calculator for Linux.

Freeman Moyo

Chris: It's an interesting idea, isn't it? How about a 'Developer's 101' series on building a Linux desktop app from scratch and covering all the decisions that you have to make along the way? Questions like what language to write it in? Of course, a lot of people will say Python, but it doesn't have to be as you're likely to find an interpreter or compiler that will enable you to run most languages on Linux (and particularly Ubuntu). As I was thinking about this I was reminded that Jon 'Kernel Column' Masters co-wrote a useful book that covers this subject called Professional Linux Programming (1st Edition), published by Wrox. Not surprisingly, it dealt with development for the Linux kernel, but also development for the desktop and web. It looked at techniques for integrating an app with the OS. I believe it's out of print now, which prompts the question: is it time for a second edition, Jon? Of course, outside of magazines there are plenty of introductory books or bookazines (try **www.**



Above Even if you are trying to install Ubuntu in Windows (tip: don't), it's probably better to download the latest version of Ubuntu 17.10, given that Canonical had to suspend and resupply its ISO because of Intel SPI driver issues

myfavouritemagazines.co.uk/tech-and-gadgets-guides-and-specials) geared towards beginners. Some are released for free on the web and there's a GitHub (**https://github.com/EbookFoundation/free-programming-books**) that tracks lots of them.

Ubuntu ills

Dear LU&D, I bought the Ubuntu 17.10 edition (**LU&D185**) with the disc, and a modest (inexpensive) SSD to load it onto. I cannot get it installed properly as the mouse doesn't function to finish loading. I tried several tactics to bypass the problem, but none have worked. The screen for [selecting the] keyboard was as far as I ever got. Is there a workaround that I missed? I have tried four mice, two wired and two wireless, into every USB port with my PC. I have a late motherboard dated 6-17, with a modest AMD CPU (4-core in AMD3+ socket). Any suggestions? **Frank B**

Chris: Hey Frank, sorry to hear you've had some problems. What have you tried so far?
This kind of thing for the mouse, for instance?

```
sudo rmmod usbhid, sudo modprobe usbhid
```

I would say that given that Ubuntu has reissued 17.10 recently because of problems with the Intel SPI driver bricking certain laptops, it might be better to download the latest version from the Ubuntu website rather than use the version on the disc now. [Note: we queried the problem a little more with Frank via email and discovered he was trying to install Ubuntu from Windows.]

Further to our discussion, it doesn't sound like you are booting from the DVD, unless I've misread what you've written. If you pick your CD drive to boot from in your BIOS/UEFI, you'll get a GRUB menu with the option to boot into a number of distros including Ubuntu. (GRUB is similar to the Windows boot-up menu which appears if you have two versions of it installed).

This will load up Ubuntu's desktop running from the DVD, and you'll see an option to physically install the distro on the desktop – or you can have a play around to decide where you want to install it at all. But basically, you shouldn't be booting into Windows at all to install the Linux distros.

COMPETITION

This month's winner is: Tom Hartley

Email the **Linux User and Developer** editorial team at **linuxuser@futurenet.com** and bag yourself the chance to win an iStorage datAshur Pro flash drive worth £60! These drives are a great piece of added security for the data in your pocket and include a PIN access panel with XTS-AES 256-bit hardware encryption and handy features such as auto-lock when unplugged. For more details head to **https://istorage-uk.com**.

INTERVIEW PUPPET

Puppet: the complexity fixer

The renowned configuration management business explains its plans to automate everything from commits to software deployments and beyond

Marianne Calder
Marianne is VP and MD of Europe, Middle East and Africa. Prior to joining Puppet in 2016, she was MD of Collaboration Sales at Cisco and has worked for 21 years in international roles throughout the company.

Rahul Singh
Rahul is VP of Engineering at Puppet. His academic background is in robotics and he was one of the first engineers at Amazon AWS, where he worked for nine years before founding Distelli.

When Luke Kanies founded Puppet in 2005, the project, along with the likes of Chef, rapidly took IT automation mainstream.

Puppet was Kanies's way of exorcising the fears he had as a sysadmin that he and his colleagues would be stuck dealing with repetitive tasks with the potential to go wrong – while titans of the internet, such as Google and Amazon, had the 'magic source' to automatically configure vast numbers of systems.

The rest of the industry was struggling, so Kanies licensed the first Puppet release under GPL (and later Apache 2.0). It used a model-based approach built around a simple declarative language that enabled sysadmins to create 'manifests' to declare how each system should be configured to do its job.

Now in its thirteenth year, Puppet is spreading its wings. It boasts over 40 open source projects (we covered Puppet 5 in Tutorials, p42, **LU&D180**) that sit alongside its commercial Puppet Enterprise edition which is free to use for up to 10 nodes. Puppet has been busy, introducing new product lines and recently acquiring Distelli, a continuous delivery automation software company, which all reflect Puppet's new focus on automating more areas of the software life cycle.

We caught up with both Marianne Calder, VP and MD of EMEA, and Rahul Singh, former owner of Distelli and new VP of Engineering at Puppet.

Marianne Calder: We're really invested from Puppet, signing to our global extension service. I joined just over a year ago, and we've also extended quite heavily into Asia-Pacific. We now have a new office in Seattle and one in Australia. We've very recently come out with a Puppet Enterprise in Japanese as well; we're doubling down on entering the Japanese market. With companies moving and driving their digital transformation, all companies really are becoming software companies. And with that drive, there's huge demand for DevOps and agile methodologies, and therefore the ability to move faster on software, both in development and operation as well.

Above Distelli was acquired by Puppet in late 2017 and its VM Dashboard has since been released as Puppet Pipelines, providing automation for IT and engineering

Above Toolchain complexity is the number one issue for DevOps wrestling to get automation working – hence the strong focus on simplification at the latest PuppetConf

That's where Puppet really comes in, in terms of helping our customers drive software better and faster. We're seeing growth that's very well in line, and also faster than, with what we see generally in the market, based on the information that we're getting from different analysts we meet.

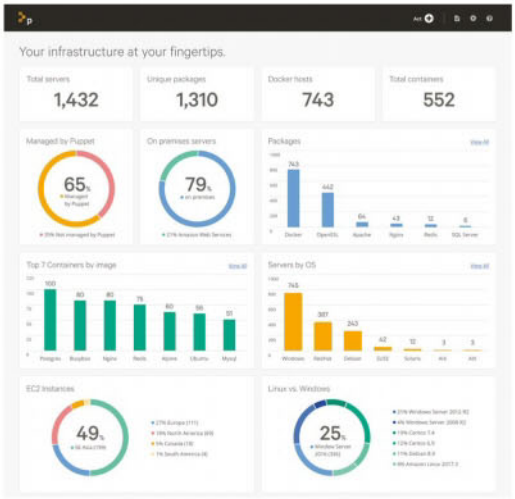
Our expectation is that that's going to continue. And actually, since you met with Puppet in May last year, we have moved from being a one-product company to a company with a product portfolio. So we're able to serve our customers much better, in a much more integrated way, that allows them to indeed move faster and move more pervasively to a fully automated environment.

Rahul Singh: Puppet's always had best-of-breed automation in the infrastructure, and operation space, and the configuration management for large fleets of servers, whether they are virtualised, bare metal, physical on-premises, cloud, hybrid and so on. The Distelli acquisition allows us to now bring automation further up the stack – beyond just

QUICK GUIDE

Puppet Enterprise

As an alternative to the open source version, Puppet Enterprise (which is free for managing up to ten nodes) can be used for larger or more complex installations. We've found this especially useful for managing heterogeneous infrastructure using Puppet across the board. As well as Linux servers, you might have to look after AIX or F5 kit, for example, or have a need to hook into an existing Active Directory setup to enable proper role-based authentication access. For teams that use Puppet but who also need to need to align themselves with corporate processes, reporting and compliance needs, the built-in functions of the Enterprise version are well worth looking into.



Above Puppet Enterprise enables companies to adopt DevOps practices such as version control and code review

configuration management and also into the areas of continuous integration, continuous delivery, applications delivery, the ARA (Application Release Automation) space, container management.

The Pipeline suite of products enables our customers to automate the application delivery, all the way from source control to servers, including continuous integration, continuous testing, continuous deployment, notifications, security integration – while still being able to leverage seamless integration with Puppet Enterprise, which has always provided best-of-breed automation for operations teams.

So I'd say we're empowering not just operations teams but we're now moving forward and empowering development teams, as well as operations teams. And that's a huge value proposition for all our customers.

So you're spreading your focus from going from configuration to development and testing?
RS: Yeah, exactly, we're expanding our capabilities as a multi-product and multi-portfolio company now.

Where do you feel the market is going? We're hearing that hybrid systems are gaining in popularity, but I'm curious to hear from you where you think companies are going in terms of how they do development.
RS: Definitely. There's been a shift from the old role of bare physical rack-and-stack to virtualisation, and then containers, and then eventually containers in the cloud. We are definitely seeing that our customers are using a variety of different approaches depending on their use case. We don't see that a one-size-fits-all is typical. We're definitely seeing the cloud being a big part of a strategy for our customers, but also multi-cloud and hybrid-cloud infrastructure with some remaining on-premises and some going into the cloud. Also, we're seeing a mix of containers and traditional applications coexisting, and a lot of uptake on some specific platforms such as Kubernetes.

As we look at the market and see where our customers are deploying their software and how they're managing the software through the life cycle, we are partnering with our customers to continue to evolve our automation to cover more and more use cases – with breadth as well as depth in automation, with not just the Dev, but the Ops as well.

The Distelli acquisition makes a lot of sense if you've got so many different types of deployment. Having to deal with all that variety and somehow keep an eye on that visually.
MC: Yes, one of the things that we continue to hear from our customers here is that there is an explosion of tools. How do you actually have an easier approach across the whole toolchain that allows you to move much faster? There have been so many tools developed. It's keeping up with integration and separate models – it's just really difficult for those customers. So having a much broader footprint, we believe, is going to be absolutely helpful for the large enterprises that we primarily support here.

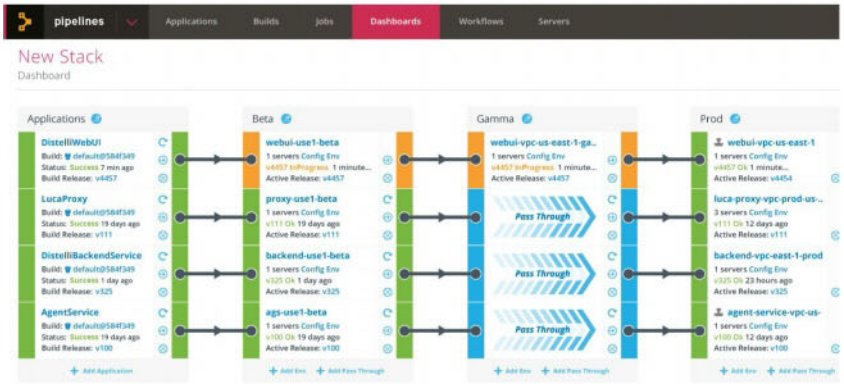
Could you give us an overview of the Puppet Pipelines offering?
RS: There are three primary products. There's Pipelines for Applications, which is a continuous integration, continuous delivery ARA (Application Release Automation) product developed specifically to allow enterprises to ship traditional applications onto virtual machines, or VMs, regardless of where

> "We're now moving forward and empowering development teams, as well as operations teams"

the VMs live: on premise, in the cloud, multi-cloud, public cloud, private cloud or hybrid cloud; regardless of whether you're shipping software from GitHub or Bitbucket, Mercurial or Git repositories and integrating with all the chat options.

We also have another product in the pipeline space, which is a Pipeline for Containers. It brings a similar flexible, yet powerful, automation for containers that are built and deployed across multiple Kubernetes clusters. Again, we're targeting Kubernetes clusters running on on-prem, public cloud, private cloud, hybrid cloud, and still having deep integration with all the other tools in automation that developers and development teams use today.

The third product that we offer is a Container Registry which allows enterprises to build and ship containers to a repository that lives on their premises, but also then start to pipeline those container images to third-party registries, regardless of the cloud. What that really gives our customers is freedom of choice and freedom from lock-in, and allows them to use container images they've built within any cloud with zero friction.



Below The aim of Pipeline for Applications is to simplify app delivery and provide continuous delivery and release automation

Am I right in thinking that the registries are an open source project?
RS: The Puppet Container Registry has several different editions. It is based on the open source Europa project that we started building, but there are commercial versions of the Container Registry as well, for use with an enterprise.

Are any of the other pipeline products based on open source products?
RS: The Pipeline for Applications, the Pipeline for Container products are completely commercial products. The Puppet Container Registry is the only one that has an open source version available. All three products were built and developed at Distelli, and are now at Puppet. The Puppet Container Registry follows a similar model to Puppet Enterprises, with an open source version but also a commercial version.

Is that presumably because of the scale involved – are they of less interest to the community side?
RS: Yes. As we talked to customers, at least in this space, they are looking for deep integration, a wider breadth of connections, other tools and other automations. They also want different deployment models. So a Puppet Pipeline for Containers and Applications is available even as a free service day they sell on Puppet, but it's also available on enterprise offer, that you can run on-prem. So for a variety of reasons, as we've talked to customers, we've had a lot of interest in the commercial versions. We found the right model for that space.

QUICK GUIDE
Puppet plans

Puppet has been expanding rapidly, opening new offices in Singapore and Seattle, and with the acquisition of Distelli last year it's added a host of products to extend automation to more of the software life cycle. According to Puppet, deployment and infrastructure automation are the two sweet spots that companies look at when considering where to start with automation. Part of the appeal of Distelli, as a constituent of Puppet, is that it offers continuous integration and continuous delivery technology that enables companies to gain a clear picture of an entire environment as well as track systems. This is achieved through:

Puppet Pipelines for Applications
Provides automation from commits to deployments.
Puppet Pipelines for Containers This can build Docker images from a source repository and deploy them to a Kubernetes cluster.
Puppet Container Registry
An open source project, which, Puppet says, is an easy way for software teams to host Docker images within their infrastructure and get a unified view of images stored in local and remote repositories.

Along with products stemming from Distelli, Puppet has introduced Puppet Tasks for running configuration management tasks on specific machines. This includes Puppet Bolt, an open source task runner. This, Puppet says, is generally designed to be used for troubleshooting or deploying one-off changes, distributing scripts to run across your infrastructure, or automating changes that need to happen in a particular order as part of an application deployment. A standalone product called Puppet Discovery enables companies to uncover what they have running in their infrastructure. This actually applies Lumogon, the open source project announced last year, on a larger scale to cover both traditional and cloud.

What's the vision, the road map, for this year? Now that I assume you've integrated and everything's merged and everything's running smoothly, what's the plan?

RS: You're absolutely right. We have integrated, and we are continuing to integrate. But I think the message we'd like to give to customers is that the innovations don't stop. We are going to continue to innovate. We are going to continue to add capabilities and feature sets to all these products at an unprecedented pace because that's what customers are expecting.

MC: We will go live early February. Then for the Discovery products, that will be in a Q2 timeframe, which is what we're expecting right now.

So Puppet Discovery [Puppet's tools for visualising what's going on in a hybrid infrastructure, for example file changes in your containers since deployment and so on] will allow customers to know what they have and know where to start with the automation process; and Puppet Enterprise continues to be developed in order to make sure that customers can start simple and then scale out, but keeping it very compliant; and then into Puppet Pipelines, which provides that bridge between Dev and Ops and the automation side of the tech.

The next six months is going to be really key from a go-to market perspective, and the timeframe is to really move Pipelines to the market early Feb [it will be available by the time you read this], and then Discovery in a May timeframe.

MC: The next six months are going to be quite transformational for Puppet in terms of the evolution of the company, the go-to market, and our approach with our customers as well.

Are the products that Distelli brings to Puppet something customers have been asking for?

MC: I would definitely say so. In fact, we just had our Technical Advisory Board here in Europe yesterday. They said that it's something that customers, not just here in Europe, but globally have been asking for. We run a Technical Advisory Board in each of the larger regions – America, Europe and also Asia-Pacific. That's been very consistent across the board. Is that resonating with what you might be hearing as well?

Generally, from what I've heard, the complexity – managing the complexity – seems to be the ongoing issue. We ran a feature on Microsoft's open source journey recently and it's astonishing to see how fast it was supporting different products in its [Azure] cloud platforms.

MC: Actually, Rahul came from Amazon, and you can maybe share your experiences from there.

RS: Yeah. As I might have mentioned earlier, prior to starting my own company, I was at Amazon for almost nine years. I spent all my time on AWS. In fact, I was the fourth engineer on the AWS team. So really, I had the opportunity to start from the ground up and watch that business grow – just being in that massive growth.

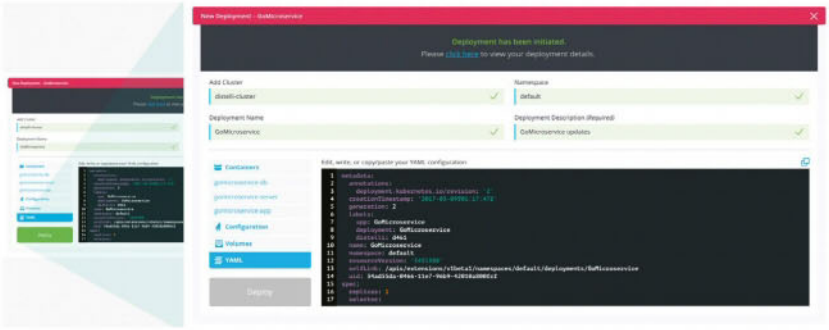
I think that I've learned, and also customers have been talking to us about this, how do we move faster? How do we move faster in the face of all of this complexity? I mean, there was a time when it was just rack-and-stack bare metal, but then there's virtualisation, and as we moved into a virtualised world, the technology moved on with containers, and now we're talking about the multi-cloud and things like server-OS. So the rate of change in the industry has gone up exponentially. The thing I learned after leaving Amazon is that the automation, at least in the case of the application delivery stage, has not kept up. That was one of my missions when I started my company – to build great automation, have the same scale and sophistication that Puppet had done for infrastructure and configuration management, and to be able to offer customers something in the application delivery space that's at a par with something they were used to in the configuration management space.

That's resonating really well with customers. We've had the Pipelines series of products in general use, in general availability at Distelli with large Fortune 500 customers, and we've had good success at scaling them up.

Has the situation with Meltdown and Spectre changed any of your approaches? Has it made you think about things differently?

RS: Meltdown and Spectre are both good examples that reinforce what we've always believed at Puppet and that is you've got to have that superb automation, like doing things in silos; doing things piecemeal or ad hoc is no longer sufficient.

Automation really has to grow and evolve with the challenges in the industry, and it has to go broad and deep at the same time. It has to be seamless. It has to be sophisticated. It has to be intuitive and powerful. I think that's the message we're sharing across the board. ■



Above Pipelines for Containers is touted as the DevOps dashboard for Kubernetes and uses the built-in YAML "human-friendly data serialisation standard' language

> "The next six months is going to be really key from a go-to market perspective"

OPINION

The kernel column

With Spectre and Meltdown still causing aftershocks, the development of 4.16 has been surprisingly trouble-free – though there's lots of work still to be done

Jon Masters is a Linux-kernel hacker who has been working on Linux for more than 22 years, since he first attended university at the age of 13. Jon lives in Cambridge, Massachusetts, and works for a large enterprise Linux vendor, where he is driving the creation of standards for energy-efficient ARM-powered servers.

Linus Torvalds announced Linux 4.16-rc2 in February, noting that things were fairly calm and that he took "the fairly quiet rc [as] a good sign for 4.16". Indeed, he had previously noted that "things certainly look a lot better than with 4.15. We have no (known) nasty surprises pending, and there were no huge issues during the merge window". That came, of course, in stark contrast to the tail-end of the 4.15 development cycle, which had coincided with the public disclosure of significant security vulnerabilities that impact many modern microprocessors.

In the end, 4.15 had required nine RCs (Release Candidates), compared with the usual 7, and close to the record of 10. Since each RC happens on a weekly cycle, this threw off advanced planning made by a number of members of the kernel community for travel or much needed downtime. As a result, Linus had noted that he would batch-up early submissions for 4.16.

The 4.16 kernel includes a number of interesting new features that were pulled into Linus's tree during the traditional two-week 'merge window' prior to RC1. These include initial support for the 'Jailhouse' hypervisor on x86 systems, Memory Protection Keys on PowerPC, and of course many more security fixes for the Meltdown and Spectre vulnerabilties (of which more in a bit).

Memory Protection Keys enable applications to limit their own access to certain regions of their address space (memory). There are many reasons why software might want to do this, but one example is to improve security. By using updated interfaces, such as changes to mprotect, an application can explicitly restrict its own access to, for example, memory containing sensitive crypto keys, except for those times when it actually needs to have access to them. If an exploit is later found in the application, this may make it harder to trick it into leaking the crypto key because an unintend access will trigger a fault.

The Jailhouse hypervisor is known as a 'static partitioning hypervisor' in that it's not intended to be used to build a cloud-computing service or to run generic operating system images, such as Windows guest VMs. Instead, Jailhouse is intended to explicitly cooperate with a running Linux kernel to isolate physical hardware resources without hiding its own existence from the applications running on the machine. Typically, Jailhouse is used in areas such as autonomous vehicles, where hypervisor isolation is good for overall security, while being lightweight enough for real-time requirements is important as well.

Petr Mladek is working to upstream patches previously created by Jason Baron that allow for atomic replacement of kernel livepatches. A 'livepatch' is a mechanism through which a running kernel can be updated without rebooting by applying carefully prepared binary patches to it. Until now, these live patches could be applied and removed only in order, so updating one patch out of many would necessitate removing and reapplying potentially numerous others. The new approach will allow individual atomic replacement of older patches.

Taras Kondratiuk is working on adding support for extended attributes to the CPIO archive format used by kernel initramfs images. This will (among other things) allow for SELinux labels to be used on files contained within an initramfs image. In addition to adding this, his proposed patches also address y2038 date-handling issues, and were generally well received. When the question turned to whether the official CPIO standard and tools can be updated, or if this would be specific to tools such as toybox (which already has some support for newcx), it seemed the latter was more likely.

Meltdown and Spectre

The fallout and resulting clean-up from the public disclosure of the Meltdown and Spectre microprocessor security vulnerabilities continued into a second month, with the release of many additional patches designed to aid in mitigating (rather than fixing) against potential exploits. As we explained in last month's extra-length column on the topic, true fixes to these vulnerabilities will necessitate hardware changes over time as newer silicon comes to market. Until then, we're faced with having to choose a small loss in performance that comes from working around certain flawed processor optimisations to protect systems from potential exploitation.

In the case of the x86 architecture, Linux 4.15 contained most of the pieces required for a bare-bones mitigation of Meltdown, and the worst of Spectre, using an upstream kernel out of the box. As Linus explained in

his 4.15 announcement message, users can use the `cat` command to read the files contained within the `/sys/devices/system/cpu/vulnerabilities` directory, with each file documenting a potential security issue and how it is being mitigated. These include mitigations for variant 2 of Spectre (attacks against the CPU branch-prediction hardware), as well as Meltdown (the ability to dump arbitrary memory on an unprotected system), but not (as of 4.15) any means to address variant 1 of Spectre (bounds-check bypass). Various distributions of course carried their own fixes, some of which may initially be more comprehensive.

Upstream kernels address variant 2 of Spectre using a novel approach invented by Google and released in a technical whitepaper last month. The attack on branch-prediction hardware typically targets indirect branches, program code jumps within the kernel to various (controllable) locations. When the branch predictor is 'poisoned' (by carefully crafted code, as explained last issue), the microprocessor may speculatively execute code other than that intended by controlling these indirect jumps. Such pieces of code are termed 'gadgets' when abused in this fashion, and the traditional fix for them is to disable part of the branch predictor that handles indirect branches.

The Google alternate, known as a 'retpoline', does something different. It replaces indirect jumps with function returns by carefully manipulating the kernel stack so that the CPU thinks it's returning from a function call. A jump is still made as part of this code sequence, but in a careful manner, such that any speculated code is intentionally forced to be a harmless infinite loop sequence. Retpolines require compiler assistance, since GCC (or LLVM) must generate these sequences in place of indirect branches. In the case of the kernel, what actually happens is that the compiler generates calls to retpoline code provided by the kernel, so both the kernel and compiler must have been updated for this to work. While the very recent upstream GCC has been updated, the vast majority of users don't yet have compilers available that can actually build retpoline kernels. Contrast this with the distro kernels, which are now often being built with a special GCC.

> "Upstream kernels address variant 2 of Spectre using a novel approach released in a whitepaper last month"

Retpolines are good, but they're not the only piece required to mitigate Spectre variant 2. Another piece is a barrier operation on the process (task) context switch that flushes the branch predictor state. On x86, this is known as IBPB (Indirect Branch Prediction Barrier) and is provided by updated CPU microcode that must be loaded separately. IBPB support went in to 4.16 (and was modified to happen only in certain cases, such as when switching into a non-dumpable process, including skipping it on switches back to the same process following idle), through a patch from Thomas Gleixner. He also included an "initial set of spectre V1 mitigations". These are small changes to specific places in the kernel where an array or other variable-bounds check might be bypassed, and they're fixed by modifying affected source code to use new 'nospec' macros around data access.

Meanwhile, ongoing debate of the various patches included a lengthy back and forth about how the IBPB code might itself contain an indirect branch without careful handling. David Woodhouse amusingly explained to Borislav Petkov how he should "Wait until the wind changes" if he wanted to rely on the compiler always generating optimised code that didn't happen to contain an errant indirect branch. Still further work focused around handling virtual machine guests, in particular those using primarily the IBRS (Indirect Branch Restrict Speculation) microcode-provided alternative to retpolines. IBRS is more expensive in terms of CPU time than retpolines, but is used by some other operating systems that need to be supported as VMs under Linux. A special case in which Linux itself would prefer to use IBRS for certain x86 CPUs is still under debate upstream, with the code not yet merged by Linus.

BUILD THE PERFECT NETWORK

Networking is everywhere – that's the point of it. **Neil Bothwick** looks at what you need to set up an advanced home or small business network

AT A GLANCE

Where to find what you're looking for

Many years ago, Sun Systems began using the phrase 'The network is the computer'. Nowadays this is so true that we don't even think about it. We all use other people's networks, whether that's your ISP, a Wi-Fi hotspot or a friend's home Wi-Fi, but someone has to set up those networks. While we won't be showing you how to set up an ISP in your front room – although that has been done – we will go through setting up a basic network suitable for more advanced home or small-business use. Wherever possible, this will be scalable, because networks always end up larger than you expected. We'll look at the topology of a standard network (if such an animal exists), the options you have regarding physical layout and some of the hardware you may need for this.

There are a lot of servers you could run on your network, most of which have been covered in great detail in **Linux User & Developer** in the past. There are some basics of running any network, however – such as DNS, routing and file sharing – that we will be covering. We'll also look at how you manage an ever-growing network: more computers means more administration and more bits to go wrong. This means that administration and backup tools need to be able to handle the extra work for you, without expecting you to deal with each computer in isolation – this is a network after all.

> "Installing a proxy server will save on your bandwidth and speed up access for your users"

More computers also mean more opportunities for attack, especially if you let others connect to your network, so we will consider how you can protect yourself from the actions of guests, whether those actions result from malicious intent or simple careless and ignorance. Remember, allowing a device unwittingly carrying a trojan to connect to your network could result in private information being taken from your computers.

More computers on your network also means more bandwidth usage. Installing a proxy server will save on your bandwidth and speed up access for its users too, so we'll look at how to set that up too.

You may be happy with your Wi-Fi-enabled router, a laptop and a phone – but you probably wouldn't be reading this magazine if that were the case. Setting up a network would normally involve some hardware investment, but it's possible to try things out using virtual machines, although your networking hardware itself may need upgrading. Routers supplied by ISPs tend to be fairly locked-down, as well as sometimes limiting the services you can offer, so a small investment in hardware that can run open source software is usually more than worthwhile.

Design your network

How you build your network is as important as what you include in it

1 Draytek Vigor 2925 Dual-WAN Router Firewall

A router like this DrayTek Vigor 2925 unit with dual WAN ports means you can connect to two providers for increased bandwidth and a failsafe connection.

Price: £185

Website: http://bit.ly/vigor2925

2 FRITZ!Box 3490

If your provider allows you to use your own modem, something like this Fritz!Box is a good choice because it offers dual wireless networks, which is ideal for giving guest access.

Price: £160

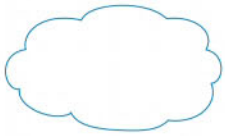
Website: http://bit.ly/FritzBox3490

3 Zyxel GS2210-24LP

A basic network switch simply routes traffic to the right device. A much more expensive managed switch allows you to control traffic.

Price: £160 (for 8 port), £300 (for 24 port)

Website: http://bit.ly/GS2210



TOPOLOGY EXPLAINED

A typical physical network layout. Not every network will contain all of these items, and some will contain more, or have one item performing multiple functions.

4 Zyxel 802.11ac Dual-Radio Ceiling Mount PoE Access Point

If you need to provide extra wireless access points, these ceiling mount units are neat and unobtrusive, looking more like a smoke alarm than networking hardware.

Price: £138

Website: http://bit.ly/ZyXelNWA1123

Many home and small office networks tend to evolve organically, and somewhat chaotically, as devices are added when needed. This is one of those areas where a little planning goes a long way, particularly in regard to siting the various bits of hardware. One location you may have little control over is where your internet connection enters the building. This may be determined by your provider, as may the hardware at this point.

If you are using ADSL, you can often change the supplied modem for any other. Users of fibre broadband or cable services, at least here in the UK, are generally stuck with the modem they are given. For fibre, this has an Ethernet port for you to connect your own router. Cable services such as Virgin provide a wireless-enabled router, but you can put this in modem mode and use it simply as an Ethernet interface to the network, ready to accept your own router.

So all you need to do is pick a router, plug it into the modem and connect your computers to the router, right? Well, maybe – provided they are all within cable range of where you site the router (most types of Ethernet cable have a 100m limit) and the router has enough ports. Otherwise you will need a network switch. A switch connects several network devices together and takes care of routing traffic appropriately. You can have more than one switch on a network but generally you shouldn't chain them; any device should have no more than one switch between itself and the central router, which is itself also a switch.

Splitting the network

Normally you want to restrict access from outside your network to a minimum, but what if you want a publicly accessible server for, say, mail or web? If someone were able to compromise that server, they could access the whole of your network. The solution is a DMZ (from the term 'de-militarised zone'). A DMZ is a separate segment of your network that is open to the internet and to the local network, but does not have access back into your network. If a computer in the DMZ is compromised, the harm is limited to that machine and any others on the same DMZ. Another type of network segment is a virtual LAN, or VLAN. It is virtual because, although it behaves as a separate network segment, it uses the same physical network. A VLAN can be used to isolate part of the local network from the rest, for example, to allow guests to use your network to connect to the internet without being able to access your computers.

> "Any device should have no more than one switch between itself and the central router, which is itself also a switch"

5 PowerEdge T30 Mini Tower Server

Any computer can be used as a server; generally the software it's running is more important than the hardware. Having said that, some hardware is more suitable than others, like this Dell T30. You don't need fancy graphics hardware – what's important is CPU power, storage space and, especially, plenty of memory.

Price: starting at £400 (Intel Xeon E3-1225 3.3GHz, 1x 8GB DDR4, 1TB HDD, DVDRW)

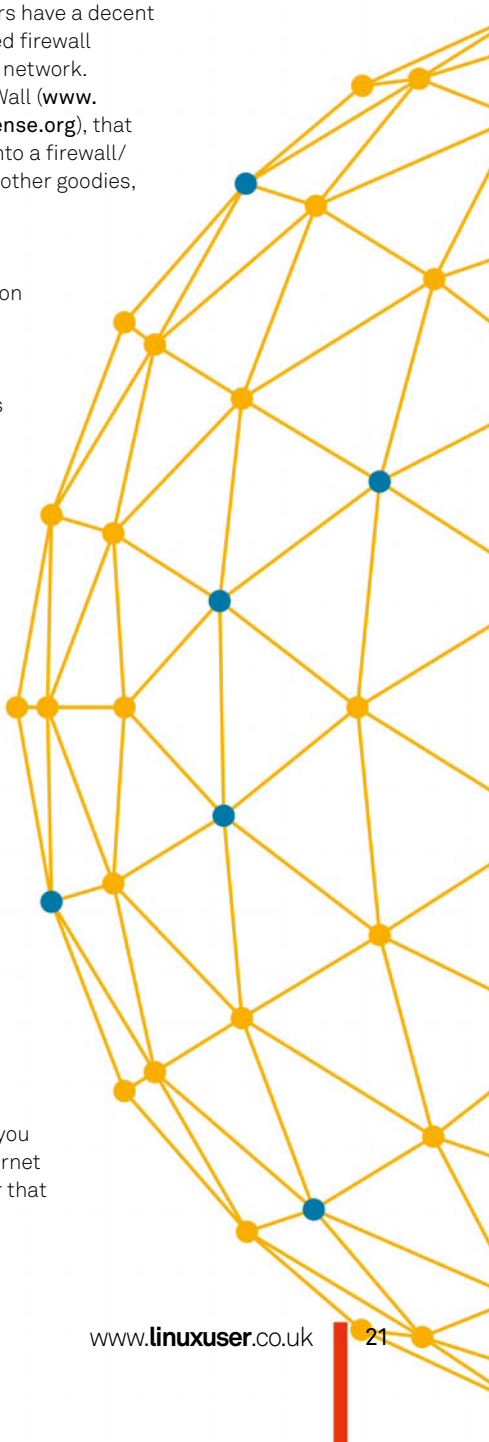
Website: http://bit.ly/DellT30Xeon

We haven't mentioned one of the most important components yet: the firewall. Most routers have a decent firewall built in, or you can run a dedicated firewall between your modem and the rest of the network. There are distributions, such as SmoothWall (**www.smoothwall.org**) and pfSense (**www.pfsense.org**), that are designed to turn an older computer into a firewall/gateway complete with DMZ, VLANs and other goodies, all controlled from a web interface.

Connecting it up

The diagram (see p20) shows an illustration of these elements. A router feeds a DMZ containing a server, a VLAN containing a wireless access point for guest users, some nearby devices including an access point and a connection to a switch for devices in another part of the building. In some cases two or more of these functions may be provided by a single device, such as a router that is also a wireless access point. Another component is in some ways the most important: the bits of wire joining everything together. There are only two real choices of cable spec, category 5e and category 6, generally known as Cat5e and Cat6. Both are capable of handling gigabit speeds, and both can handle power over ethernet (PoE) where devices can receive power as well as data from a suitable switch. Cat6 has a shorter maximum length of 55m, but Cat6a restores this to 100m.

If internet connectivity is critical for you, you should consider a backup connection. Some routers have an option to add a USB 4G dongle as a fallback, or you could go the whole hog and have two internet connections with a load-balancing router that uses both at once.



QUICK TIP
Releasing leases
If dnsmasq still gives the same address from the dynamic range to a computer after you have set up a specific configuration, use dhcp_release to get rid of the old lease details.

Essential services

Set up the basic services that a network needs to function, such as DNS and DHCP, and avoid common pitfalls

Once you have more than one device on a network, you need them to be able to talk to one another. That means you need to be able to give each host an IP address and let it know the IP addresses of the other hosts. This can be done manually by giving each device a static address and adding it to the hosts file of each of the other devices, but this soon gets out of hand – and what happens when a guest wants to connect their phone to your wireless network?

The solutions are DHCP and DNS. A DHCP server will allocate an address to any device that requests one, along with other information such as the gateway and DNS addresses. The DNS server converts hostnames to IP addresses. On most networks it is best to use a forwarding server for this. It passes the request to a server on the internet and passes the result back, remembering it for the next time. Most routers have DNS forwarding and DHCP services built in, but you can get more control if you run your own, with dnsmasq being the preferred choice for many. Some routers, especially those that run a version of DD-WRT (see the wireless section for more on this), use dnamasq by default, but we'll look at running it on a computer on your network. The computer does need to be permanently turned on, but it's easier to monitor and debug your configuration this way. Once you have it working, you could copy the configuration to a router that supports it if you prefer to run it there.

> "Most routers have DNS forwarding and DHCP services built in, but you can get more control if you run your own"

Install dmnmasq from your distro's repositories and edit its configuration file to set it up. The main configuration at `/etc/dnsmasq.conf` is huge, but well-commented, fortunately. To keep things simple, uncomment the line that says `conf-dir=/etc/dnsmasq.d`. Now create your own configuration files in that directory:

```
domain-needed
domain=your.domain
local=/your.domain/
dhcp-range=192.168.1.128,192.168.1.192
dhcp-option=option:router,192.168.1.99
log-facility=/var/log/dnsmasq.log
```

This tells the server the range of addresses to use, the name of the local domain and the address of your internet gateway. Two other files are used: `/etc/resolv.conf` contains the addresses of the upstream name servers, although you can also specify these in the config file with `server=8.8.8.8`. The other standard file is `/etc/hosts`, where your statically addressed hosts are listed. You need at least one statically addressed host, the one running your DHCP server. Speaking of static addresses, you can ensure dnamasq always gives the same address to each computer with a line like this:

```
dhcp-host=00:fd:45:fc:92:20,192.168.1.8,desiato
```

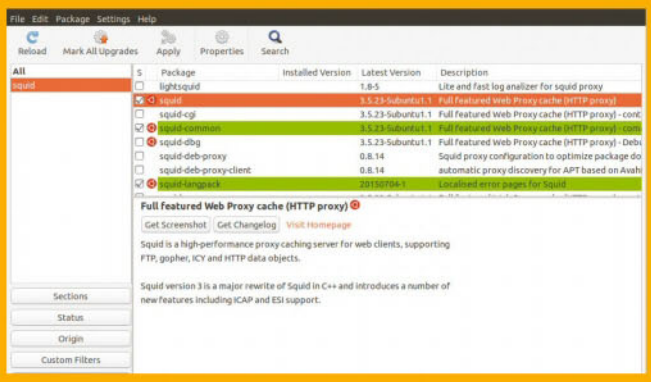
The first field is the MAC address of the host, the second is the IP address to give it and the third is the hostname. The address should be outside of the `dhcp-range`. If dnamasq has already allocated an IP address to this computer, you may need to use `dhcp_release` to clear the lease. This is a convenient way to give static addresses to servers on your network without having to maintain separate configs.

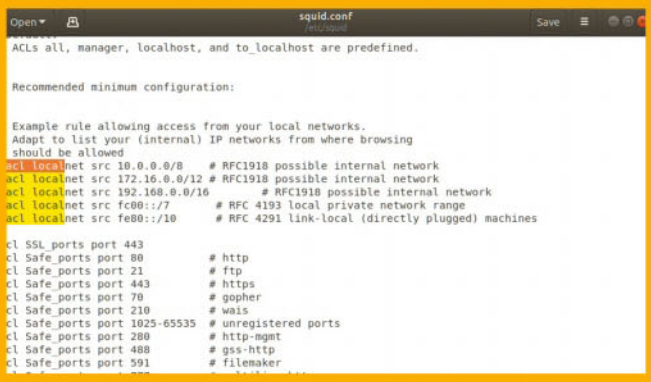
When you're ready to fire up dnsmasq, first turn off the DHCP service in your router. Having two DHCP servers on a network is like trying to drive with two satnavs: confusing as hell to all involved. Now set one of your computers to

QUICK GUIDE
Reverse proxy servers
A reverse proxy server is a rather different animal to a normal proxy that basically forwards requests from one server to another. The idea is that if you have several domain names pointing to your public IP address, you can use a reverse proxy to redirect requests to the correct server based on the domain name (or other criteria). For example, you could forward requests to mail.yourdomain.com to your mail server and www.yourdomain.com to your web server, even though all requests come in on port 80, which your router forwards to the computer running the reverse proxy. This function is often performed by a web browser.

HOW TO

Set up a proxy server



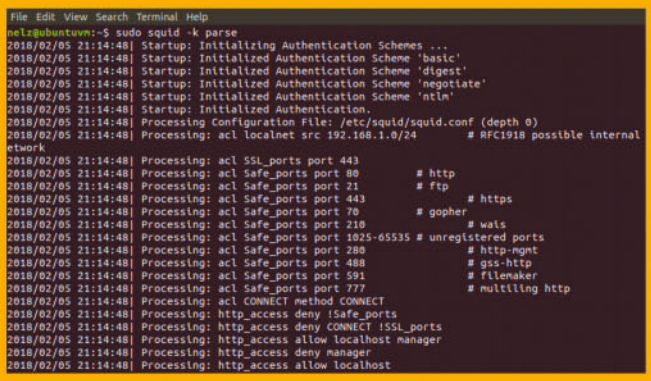


1 Install the squid caching proxy server

A recent version of squid should be in your distro's software manager, or you can compile from the source code on the DVD using the standard `./configure && make && make install` invocation, which should only be necessary if your distro is old or you really need the latest version.

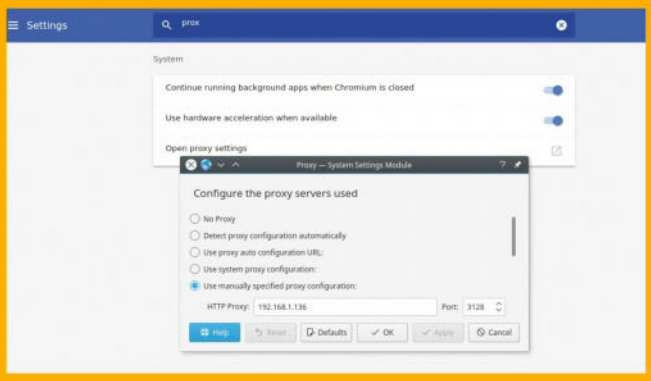
2 Configuration basics

Edit `/etc/squid/squid.conf` to suit your network. You should set one `acl localnet` line to point to your local network and comment out the other examples. Then check the `http_access` lines, which determine who can use the server. The defaults may well be suitable for you, but check anyway.



3 Test your configuration

Configuration is important for any program, but more so for network servers where a mistake can open you up to abuse. squid has an option to test the configuration before it goes live. Run `$ sudo squid -k parse` to see the configuration in action.



4 Start/restart the proxy server

Some distros start a server when you install them, which isn't always a good idea. After testing the settings, restart the server with `$ systemctl restart squid`. This may take a while on first run. Then set your browser to use your server address and port 3128.

use DHCP, if it doesn't already, and reboot it. You should see the request and response appear in `/var/log/dnsmasq.log` on the server. There are programs that will test for DHCP responses without rebooting or messing up your existing configuration. If you have dhcpcd installed as your DHCP client, use

```
dhcpcd --ipv4only --test
```

otherwise you can use nmap; both will give information about the server's response.

```
sudo nmap --script
broadcast-dhcp-discover
```

So that's DNS and DHCP sorted out. Now you can connect any device to your network and it should be able to find its way around, and be found. The next step you may want to take is to add a proxy server to reduce the load on your internet connection. The standard proxy server is squid (see above), but this may be overkill for a small-to-medium sized network, in which case you can use Tinyproxy.

Tinyproxy is unusual in that the defaults are all many need, although there is scope for tweaking. Just tell your browser to use port 8888 and the hostname of the server and it will forward all requests through tinyproxy.

QUICK TIP

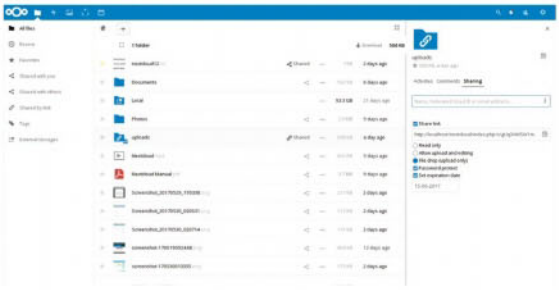
Add debug logging

Add `log-dhcp` to your dnsmasq configuration to increase the amount of information logged while you are testing.

QUICK TIP
Control users' rights
If running a file server, you may want to restrict what people can do with the files. You don't want one user deleting another's work.

File and media servers

Allow others on your network, and elsewhere, to access files and media from your servers



Right Nextcloud is a good option if you want to provide cloud-style file storage, accessible through a browser on your network

Once you start using multiple devices on your network you run into a particular problem: how to access a file on a different device. There are two possible solutions. One is to use a synchronisation program such as Syncthing, to make sure all important directories are copied onto all devices that need them. This works for a small number of devices but soon becomes cumbersome. The other option is to use a file server, so that everything is stored on one device but accessible from all of them. You can implement a file server using one of the standard directory-sharing protocols and let the individual computers sort things out, or you can set up a media server that provides not only files but associated metadata and probably a nice interface too.

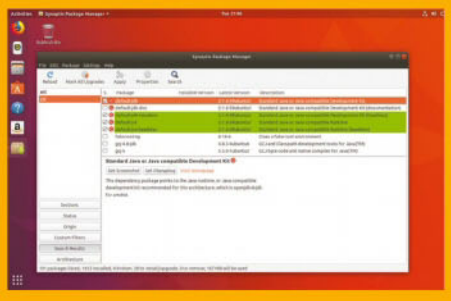
Examples of the former are NFS, Samba or even FTP, while the latter is covered by the likes of Airsonic (for audio files) or Emby and Plex (for video content). The other alternative is to use a dedicated file server, a NAS (Network Attached Storage), running a specialist operating system like FreeNAS. This is easy to set up and manage but has the considerable drawback of needing a separate computer.

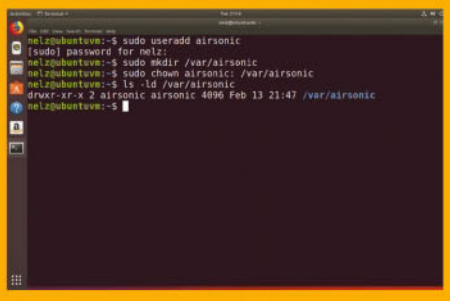
Samba is an implementation of the Microsoft CIFS protocol (previously called SMB, hence the name Samba) and is more widely supported than NFS, so we'll concentrate on that. Install Samba from your distro's repositories in the usual way and set it up by editing the main configuration file at `/etc/samba/smb.conf`. The file uses the INI format, where each section starts with a name in square brackets followed by one or more lines setting parameters, and ends at the next set of brackets. The first part is called `[globals]` and, not surprisingly, contains global settings for the server. At a minimum it will contain:

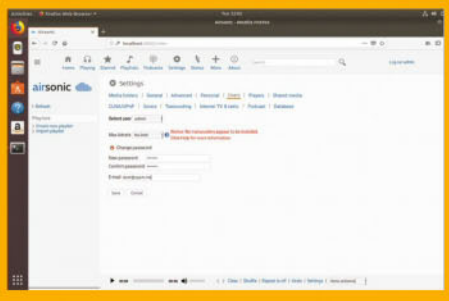
```
[global]
   workgroup = MYGROUP
   netbios aliases = MYHOSTNAME
```

This sets the workgroup that all computers will see and the NetBIOS name by which the server is addressed, which you will usually want to set to the hostname of the computer for convenience. Next comes have a section for each share you want to set up, for example:

HOW TO
Install Airsonic music server







1 Install dependencies
Airsonic is a Java web application, so you need to install a Java JDK, such as OpenJDK, Java JDK or IcedTea. Airsonic also uses Tomcat. If you already have Apache/Tomcat you can use that, but we'll use the standalone Airsonic package.

2 Prepare the location
Create a directory at /var/airsonic and set its owner to the user that will run Airsonic. Then download the WAR file from **https://airsonic.github.io/download** and run it with `$ sudo -u airsonic java -jar airsonic.war`

3 First login
Point your browser to **http://localhost:8080** and login as admin with password admin; you'll be asked to set a new password. If you see a warning about no transcoders being installed, pop back to your package manager and install `ffmpeg`.

```
[Music]
   comment = Music folder
   path = /mnt/music
   read only = no
```

These parameters are all pretty self-explanatory. In order to mount this share, you will need to supply a username and password. It is possible to set up anonymous logins on Samba, but not wise, especially if you are allowing write access. Samba users can be created with the **smbpasswd** command:

```
$ smbpasswd -a username
```

You will be prompted for a password. For this to work, the user must already exist in the system **/etc/passwd** file. If the user is only to have access to the Samba server, you can create the user without a home directory and then prevent other logins with:

```
$ useradd -M username
$ usermod -L username
```

Then use **smbpasswd** to create their Samba login. However, if you want them to be able to log in, for example via SSH, and have created a normal user account for them, you don't need **smbpasswd** at all. Instead, add a line to **/etc/samba/smbusers**:

```
linuxuser = sambauser
```

and any attempt to mount a share as **sambauser** will use the credentials for **linuxuser**. The two user names can be, and often are, the same.

Now start the **smbd** service on the server; you can mount it on another computer on the network with the following command:

QUICK GUIDE

A step further

An alternative way of making files available is to run your own cloud server, using something like Nextcloud (see Features, p56, **LU&D** 180 for a complete tutorial). Nextcloud gives you a similar experience to DropBox or one of the other cloud-storage services, but under your own control. Not only do you keep hold of all your data but it is also served at LAN speeds making it much faster for local users. This may be a suitable solution for less technical users, although it is more work for you to set up.

```
$ sudo mount.cifs //myhostname/share /some/
directory -o username=xxxx,password=yyyy
```

Assuming that works, you can now add this to **/etc/fstab** to mount the share at boot-up, for example:

```
//myhostname/Music /mnt/music cifs
username=user,password=pass
```

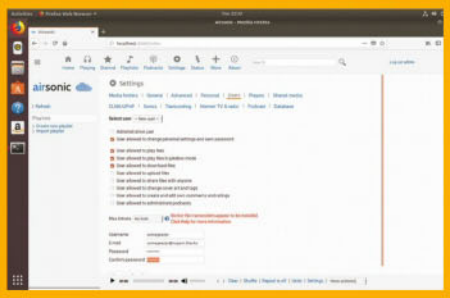
However, **/etc/fstab** is globally readable, so you may prefer this approach:

```
//myhostname/Music /mnt/music cifs
credentials=/etc/samba/musiclogin
```

where **/etc/samba/musiclogin** is readable only by root and contains:

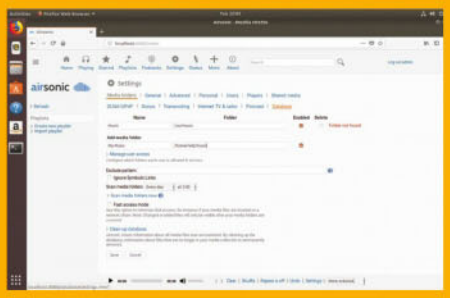
```
username=youruser
password=yourpassword
```

In the walkthrough below, we'll look at how to install the previously mentioned Airsonic, a media server designed specifically for music and other audio files. The likes of Plex and Emby work in the same way.



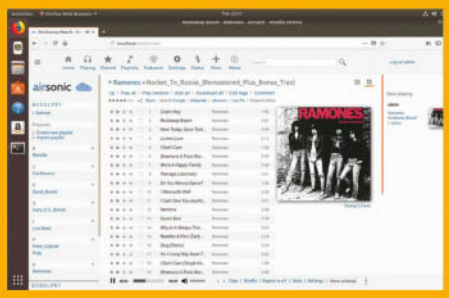
4 Add another user

As the admin user, you have the ability to screw up anything you touch. You probably don't want to share that power so create another user with less permissive rights. You can have as many users as you want, with different rights.



5 Add some music

A music server isn't much use without some music. Airsonic can play music from anywhere, so add a folder or two, Save and then click 'Scan'. By default Airsonic scans the library for new additions at 3 AM.



6 Good to go

You can now browse and play music from any web browser on your network. If you want to be able to access your music from outside of your network, look at the reverse proxy documentation on the Airsonic website.

QUICK TIP
SSH the smart way
When using SSH, especially with multiple terminals, use screen or tmux to give a detachable shell in which you can run commands and then return later.

Automate systems

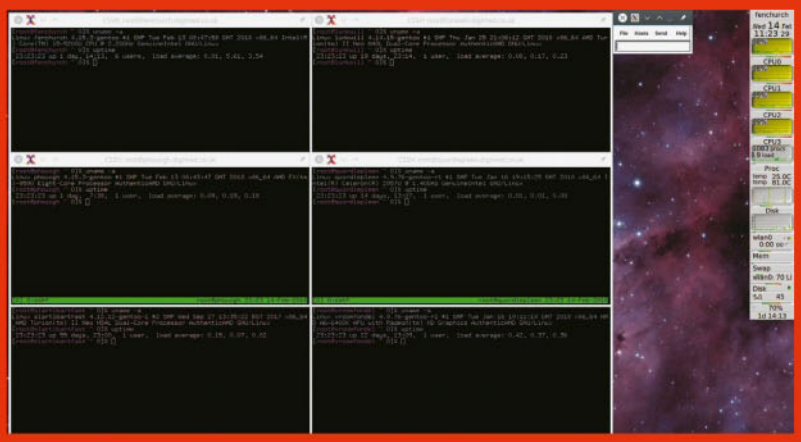
Multiple computers multiply your admin workload, but there are many tools available to stop this getting out of hand

Adding computers to your network also adds work in terms of maintaining them and keeping them up to date. You don't want to be hopping from one keyboard to the next to do things, and your servers may not even have keyboards or monitors, so the first thing to do is set up SSH. Most distros have two SSH packages, the client and server. The client package is often installed by default so you just need to add the server to each machine you want to control remotely. On Ubuntu, the package is called openssh-server. After installing and starting the service, you should be able to connect from another computer with `$ ssh user@hostname`. The first thing you may want to do is remove the need to use a password when logging in as the same user. For this you will need an SSH key on your local computer, which you can create with: `$ ssh-keygen -t ed25519`. The `-t` option specifies the type of public-key signature to use; Ed25519 is currently considered safest. When prompted for a passphrase, you can press Enter to create a key pair with no passphrase, but you should use one if there is a possibility that others may have access to your keys, or they could be stolen.

> "You can set up cron jobs on each computer to do the boring stuff"

Now you send the public part of the key to the remote server using:

```
$ ssh-copy-id -i ~/.ssh/id_ed25519.pub user hostname
```



Below Cluster SSH connected to six computers. Each one can be controlled individually, but you can type a command in the small window at the top right and it will run on all of them at once

QUICK GUIDE
Backups

Making backups becomes increasingly important as your network, and the amount on data on it, grows. There are three key rules about backups: do them frequently, do them automatically and test them regularly. The first one is obvious, the second fairly obvious, but the third is critical.

You need to be certain that you can restore data from your backups in the event of a loss, and just after that event is not the time to find out whether this is true. You need something that can be run unattended from a cron job. With a network you have two main choices: a backup server that pulls backups from other computers, as BackupPC does (**http://backuppc.sourceforge.net**), or a cron job running on each computer sending backups to a central point, such as your file server.

In this case your file server's integrity becomes critically important, so use at least a RAID 1 setup on the server. Our preferred method is to backup to a file server from each computer and then sync the backups with a cloud service so there is always a safe, off-site copy.

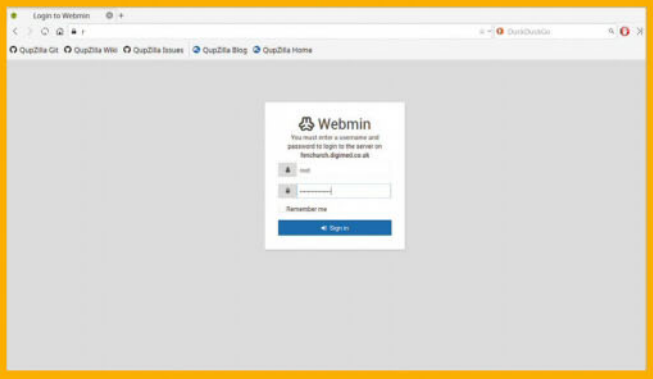
You will be asked for the password one more time, but from now on you can SSH in to that account without it. Once you have done this, and tested that it works, you can disable passworded logins on the remote computer. SSH into it and run:

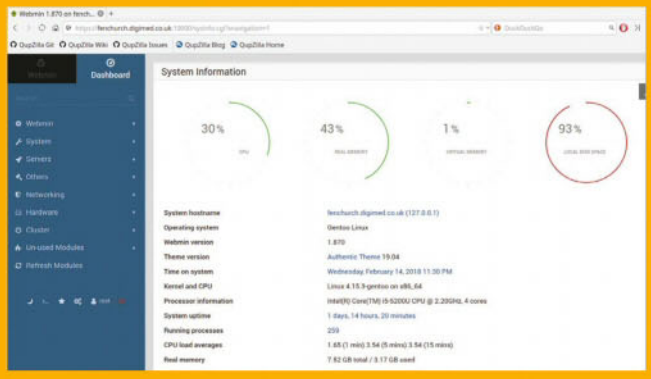
```
$ sudo nano /etc/ssh/sshd_config
```

and change the setting for PasswordAuthentication to no. Make sure you have either local access or key-based login before you do this, or you could lock yourself out. By default, root logins with password are blocked. If you want to be able to do this, change PermitRootLogin in the sshd_config file to yes, restart the server, copy your key file as above and then change the PermitRootLogin back to prohibit-password then restart the server. Exposing the root user to password attempts is never a good idea.

With a growing number of computers on your network the task of keeping them all up to date grows at least as quickly. There are a number of professional standard solutions for managing multiple computers on a network, including Ansible, Chef and Puppet. Each one has its benefits and each one

HOW TO
Monitor computers with Webmin



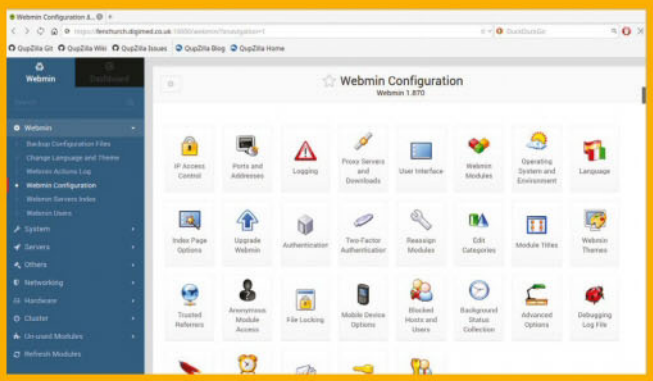


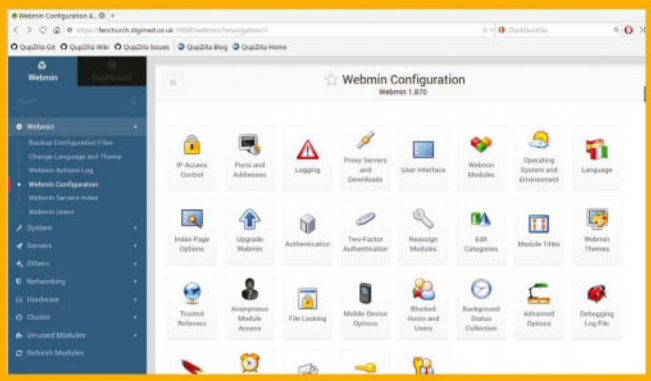
1 Webmin graphical remote admin

Webmin (**www.webmin.com**) provides web-based administration for many functions of a computer system. After installation, connect to **http://localhost:10000** and log in. This is a normal Linux user login and needs to be root.

2 A warm welcome

The home page shows a dashboard summary of the system running Webmin, with continually updated figures, just like a decent desktop system monitor. The menu on the left lists the many and various categories of admin possible.





3 Lock it down

Webmin's default is to allow access from anywhere, possibly rather unwisely. Go to the Webmin configuration menu and select IP Access Control to restrict the IP addresses that are allowed to connect.

4 Modules galore

Webmin comes with many configuration modules, with many more available as official add-ons, most of which are pre-installed, or third party add-ons. See the website for details of the various modules and themes available.

is deserving of an article in its own right. However, there are also ways to use simple SSH, especially if you have a number of computers running similar operating systems. For example, if several are Debian-based, you can update them all by running `apt-get update && apt-get upgrade` on all of them at once. One way of doing this is with Cluster SSH. This program opens multiple xterm windows on different hosts; if you type into the additional control window rather than the standard host window, what you type goes into every xterm. The hosts to run on are defined in `/etc/clusters` or `~/.clusterssh/clusters` like this:

```
deb ubuntu1 ubuntu2 debian3
rpm suse1 fedora2
all deb rpm
```

The first item on a line is a group name and the rest are hostnames or groups that belong to that group. So you can open xterms for all your Debian-based machines with `$ cssh deb`, then run `apt-get update`. The `all` group is useful for non-distro specific operations.

Another important tool in reducing your workload is cron. You can set up cron jobs on each computer to do the boring stuff, such as running backups and checking for updates; or you can use key-based SSH to run commands on other computers from a set of cron jobs on one computer. While this is slightly more complex to set up, and requires connectivity, it does mean you can manage everything from one computer. ▸

Monitor & manage

Monitor what is going on with your network, and keep an eye on the health of each computer in it

Adding more computers to a network not only adds to the maintenance workload, you also need to keep an eye on them for problems. There are plenty of monitoring programs available which broadly fall into two categories: those that simply provide information on the current and historical behaviour of the systems, and those that watch for specific problems and either attempt to tackle them or send you an alert.

Cacti (**www.cacti.net**) can produce beautiful graphs of the status of many computers. It's a PHP app so installing it can be tricky. First, unpack it into your web server – we will assume Apache here – **DocumentRoot** and **cd** into the **cacti** directory. Create a suitable database and user:

```
$ mysqladmin --user=root -p create cacti
$ mysql -p cacti <cacti.sql
$ mysql --user=root mysql
> GRANT ALL ON cacti.* TO cactiuser@
localhost IDENTIFIED BY 'apassword';
> grant SELECT on mysql.time_zone_name TO
cactiuser@localhost IDENTIFIED BY 'apassword';
> flush privileges;
> quit
```

Make sure the web server user can write to directories:

```
chown -R apache: rra/ log/
```

Add this line to **/etc/crontab** to have Cacti poll for information every five minutes, adjusting the path to suit:

```
*/5 * * * * apache /usr/bin/php /var/www/
htdocs/cacti/poller.php >/dev/null 2>&1
```

After installing cacti, go to **http://localhost/cacti** (or wherever you installed it on your web server) and the installer will start. This mainly consists of system configuration checks, so correct anything highlighted and proceed until it is fully installed. Then you will be asked to log in; the default user is **admin** and so is the password, which you will be asked to change immediately. After the somewhat lengthy set up, you can now add your first device, by clicking 'Create devices' and then the + icon at the top right. Just set the description and hostname/IP address (you may as well start with **localhost**) for now; the defaults will do for everything else, although you may want to change the Device Template setting. Now you have an option to add one or more graphs for this device and click Save. Many of the queries for information use SNMP (see the boxout, right).

Add some graphs for your device by clicking New Graphs in the Create menu. Select your device and it will show the data options you added when you created it; tick the ones you want to graph and press Create. If you go back to your device's information page it should now show the graph templates as being graphed. Click the graphs tab at the top left and the preview tab at top right to see your graphs.

Monitor and react

Monit (**https://mmonit.com/monit**) is a more proactive option that can take action when situations occur. It is run locally on each computer you want to monitor, but can sent notifications to a central location. All configuration is done in the main configuration file at **~/.monitrc** or **/etc/monit** if that does not exist. As this file may contain sensitive information, it is wise to restrict read privileges (**chmod 600**). Start monit from your service manager and point your browser to **http://localhost:2812** – the default login is **admin:monit**. You will see some basic system information, so let's add some more. First we tell monit how to alert you by email; **/etc/monitrc** has commented examples, so uncomment the set **mailserver** and set **alert lines** and add your own mail server and address. Here's our first test:

```
check filesystem rootfs with path /
   if changed fsflags then alert
   if space usage > 90% then alert
```

Each check starts with the word **check** followed by the type of check, a unique label and the item to check.

QUICK TIP

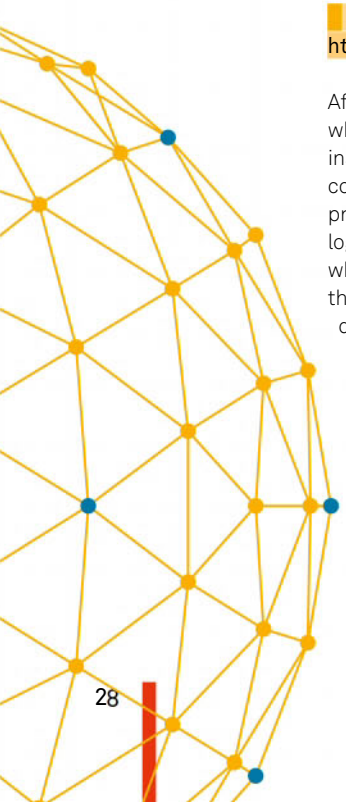
Diagnostic tools to use

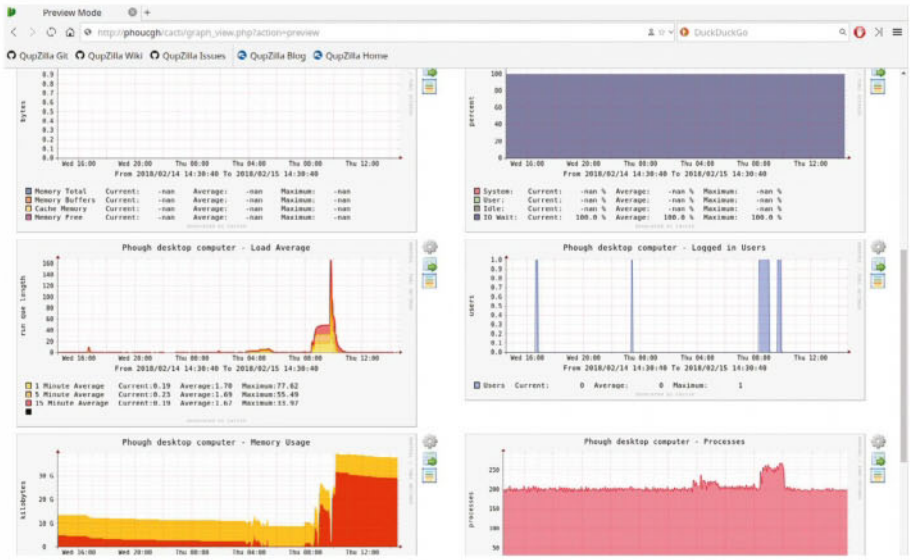
While the graphs give a nice overview, sometimes it is better to SSH into a computer that you suspect may have a problem and use standard diagnostic tools like top or free.

QUICK GUIDE

SNMP

Cacti uses SNMP (Simple Network Management Protocol) to gather information from the computers it is monitoring, so each computer need the **snmp** package installed and the **snmpd** service running. On the computer running cacti you should be able to install and forget, but the default setting is to only listen on **localhost**, so you will need to change the configuration on the other computers. Any other computers you want to monitor do not need Cacti installed, as long as **snmpd** is installed and suitably configured. Edit **/etc/snmp/snmpd.conf**, find the **agentAddress** setting and set it to something like **agentAddress udp:161** to listen on port 161 (the default) on all IPv4 addresses. Try not to expose port 161 on any computer to the outside of your network, or else read the manual sections on SNMP V3, which adds authentication.





Above Cacti produces graphs from information it harvests from computers on your network for a graphical

In this case we're checking the root file system. The **check** line is followed by a couple of tests. The first checks whether the file system mount flags have changed, which could be caused by an error making the file system read-only, and the second checks for space usage. In each case the action to carry out is to send an alert. If you look at the commented-out **system check** in **/etc/monitrc**, you'll see it performs similar checks for things like CPU load, memory and swap usage. An increase in any of these could indicate a runaway program – or maybe that you're transcoding a 4K video! If you *are* going to do something that you know will trigger a monit test, you can turn off a monitor while you do it:

> "You can use exec to run any program you want on any event"

```
$ monit unmonitor rootfs
```

Use **monitor** to start it again. If you run monit with no arguments it will immediately run its checks, or run it with a reload if you have changed the config file. These commands all communicate with the running instance of monit.

Monit will only send an alert the first time a test fails. If you want it to nag you about an important failure, add a reminder:

```
if space usage > 95 % then alert
with reminder every 10 cycles
```

The default polling interval is 30 seconds, so this will remind you every five minutes when your file system is critically full.

As nice as it is to get emails, sometimes you just want the issue fixed, for example if a daemon stops running. The most reliable way of doing this is to use the pid file created when many daemons start up:

```
check process mysqld with pidfile
/run/mysqld/mysqld.pid
    start "/etc/init.d/mysqld start"
    stop "/etc/init.d/mysqld stop"
```

Now monit knows how to start the program if it stops. However, there may be a reason it stopped that would cause it to stop again and you don't want monit to get into a loop of repeatedly restarting the daemon, so add

```
if 2 restarts within 3 cycles then
unmonitor
```

You can also check for the availability of a remote computer with

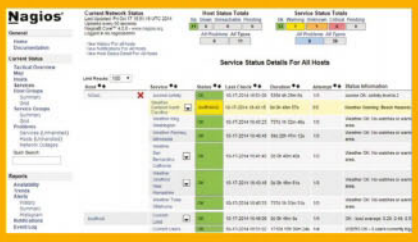
```
check host rpi with address
192.168.1.168
    if failed ping then alert
```

As well as the alert and start options, you can use exec to run any program you want on any event.

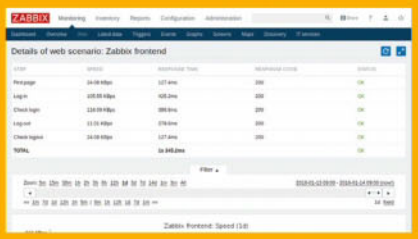
PRODUCTS

Other monitoring apps

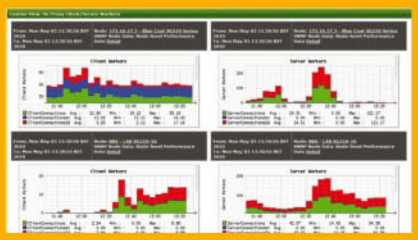
There are many other products available to monitor a network of computers



1 Nagios
www.nagios.org
This is the main heavyweight contender in this field. If you want to monitor it, there's bound to be a Nagios plugin to do it for you. It's probably overkill for a SoHo-type network but worth considering for larger installations.



2 Zabbix
www.zabbix.com
Zabbix is another enterprise-level system, mostly aimed at anyone needing to watch over large numbers of computers and services, but it may also fit the needs of smaller networks in some cases. It works with Windows too.



3 OpenNMS
www.opennms.org/en
OpenNMS is another product describing itself as "carrier-grade". It uses an event-driven architecture where actions are taken when services stop or criteria are exceeded, but again is mostly likely to be overkill for small networks.

QUICK TIP
Password politics
You may want a fairly simple password for guest wireless access, but choose a much stronger, and completely different, one for internal network access.

Other types of access

It's not all about computers plugged into Ethernet switches – what if you or others want to connect wirelessly?

When it comes to adding wireless access to your network, this event raises a whole lot of new questions and the answers bear some serious consideration before diving in:

- Do you want to give wireless access to visitors?
- Should you have a separate network for guests that gives access to the internet but not local computers?
- Where should I place the access point?
- Do I need multiple access points?
- Will different antennas help?
- Which access point should I use?

It is likely that your ISP-provided router includes a wireless access point (AP), but that may not be ideal. Aside from the privacy concerns of your ISP having remote access to your device, it may not provide the features you want, or it may not be located in the best place for coverage. Standalone APs are available, usually with far more features than the bundled ones. There are even a couple of Linux distros for APs, DD-WRT and OpenWrt – some APs already use these. This means that you can run some of the services for your network on the AP, such as DNS or VPN, reducing the need for an always-on computer to do this.

Positioning your AP

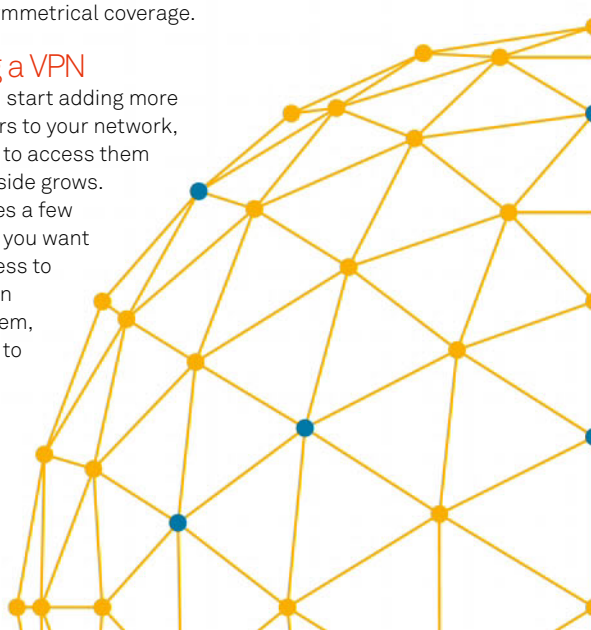
Access point placement is important, and is easier to do with a separate AP as you are not tied to where the internet connection enters the premises. A signal-strength app on a mobile phone will help you decide the best placement for overall coverage. If the building is too large, or the walls too thick, to cover everywhere, you have a number of options: different antennae, more than one AP or a wireless repeater. The last is the easiest to use but because it is making two wireless connections, one to your device and one to the AP, throughput is halved. Multiple APs work well and if you use the same SSID and password on each, devices should switch automatically as you move around the building, always giving you a decent signal. While you have your signal-strength app running, check which channels are in use by nearby APs and set your APs to use channels well away from those and each other. The better APs also have removable antennae. Changing the antenna will not increase the output power but it will reshape the coverage. A higher-gain, usually longer, antenna will increase horizontal coverage (assuming the antenna is positioned vertically) at the expense of vertical coverage. This is useful in a single-storey building, or if you have an AP on each floor. If you are unable to position your AP centrally, you may be able to fit a directional antenna that gives asymmetrical coverage.

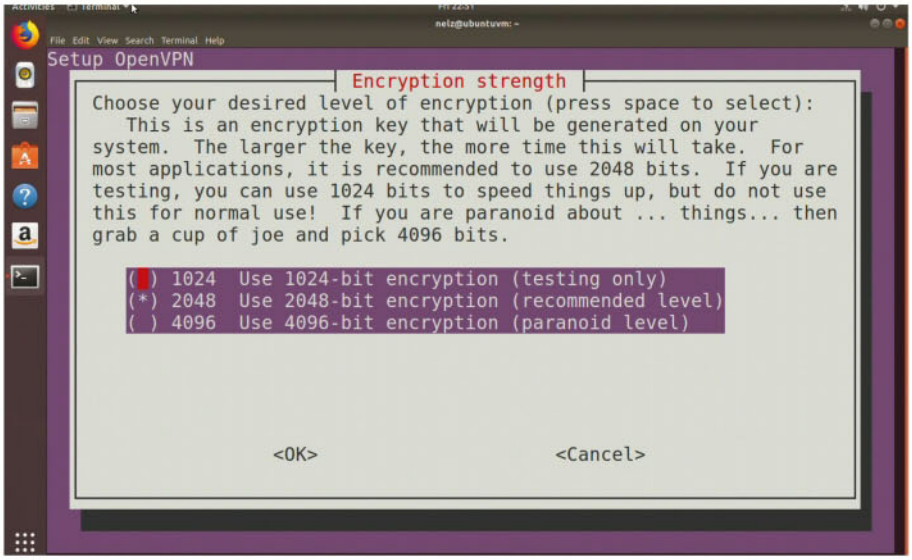
> "Access point placement is important, and is easier to do with a separate AP as you are not tied to where the internet connection enters the premises"

QUICK GUIDE
Guest access
You need to consider exactly who you're giving access to your wireless network. Connecting to the AP not only gives them the internet but also your local network. Some access points offer VLANs (Virtual LANs) so you can provide one password for the guest VLAN and another for your own, with the guest VLAN being separated from your own. The exact details of this vary according to your router's operating system, but DD-WRT offers it. The other approach is to create the VLANs in your router and connect a separate AP for each one. Which is best depends on the options offered by your router and APs. Doing it at the router could also give you the option of using traffic-shaping to prevent guest users sucking up all your bandwidth. Some routers refer to this as hotspot mode. Don't be tempted to make your guest access open; it still needs WPA.

Adding a VPN

Once you start adding more computers to your network, the need to access them from outside grows. This raises a few issues. If you want SSH access to more than one of them, you have to run SSH on non-





Above PiVPN is the easy way to set up OpenVPN; the eagle-eyed among you will have noticed that it's running on Ubuntu here, not a Raspberry Pi, despite the name

standard ports, as a NAT router can only forward each port to one computer (there are ways around this with reverse proxies, but it starts to get complicated). Another issue is that the more ports you open up to the outside world, the more opportunities you provide for attack. So it would be best to connect to your network from outside in a way that is secure, but allows you to do everything you can locally.

The answer is a VPN (Virtual Private Network). This provides a secure tunnel though your internet connection, allowing you to join the network from outside. OpenVPN is based on SSH, a trusted and secure way of connecting. It should be noted that this is not quite the same as the VPN services you can purchase online. There you are paying to join someone else's network and then go online, to disguise your location. Here we are joining our own private network from outside.

It can be a little tricky to set up but there is help in the form of PiVPN (**www.pivpn.io**). From the name you may have guessed that this is designed for the Raspberry Pi, and one of these tiny boards tucked away on your network can act as a gateway, but it also runs on most Ubuntu and Debian distros. To set up an OpenVPN server using PiVPN, run:

```
$ curl -L https://install.pivpn.io | bash
```

This will download the latest script and run it. It will ask you a series of questions, generate keys for the server, install and configure it. Next you need to set up each client to be able to connect, and PiVPN does this for you too:

```
pivpn add
```

This asks you for a name for the client and a password for the connection (not the same as the user's password on the computer). PiVPN will generate a .ovpn file that contains both configuration and a certificate. This can then be loaded into a VPN client to establish a connection.

On Android devices, for instance, we use OpenVPN for Android, which imports .ovpn files directly and creates a connection you can just touch to open. To use it with Linux, copy the file to **/etc/openvpn/client/name.conf**: the **.conf** extension is important. Then you can start it with

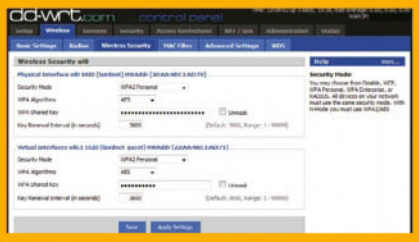
```
$ systemctl start openvpn-client@name
```

Using this method you can have several different client configs for connecting to different networks and just supply the name in the command above. ■

PRODUCTS

Access point firmware

Replace your access point or router's firmware with an open source alternative

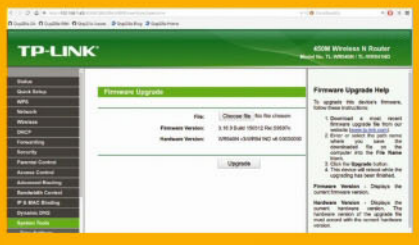


1 DD-WRT
www.dd-wrt.com

DD-WRT is the most well-known of the choices. Its name comes from the fact that it was originally developed for the classic WRT54G access point, but now it covers hundreds of models, listed at the website.

2 Openwrt

Openwrt is a fork of DD-WRT that is more flexible, but at the expense of more effort on your part – think Arch vs Ubuntu on routers. With Openwrt, for example, you can add extra packages to increase the capabilities of your router.



3 Installing the firmware

Installing your own OS on your router is surprisingly simple, especially with DD-DRT. You simply upload a firmware file as if you were doing a factory-supplied update. Just be careful to follow the instructions to avoid bricking your router.

Serval Project: saving lives with mesh telephony

Chris Thornett talks to the founder of an open source project that wants to make mobile communication available to everyone

Dr Paul Gardner-Stephen

Paul is a senior lecturer at the College of Science and Engineering at Flinders University, Adelaide, Australia. He's also the founder of the Serval Project.

KEY INFO

The Serval Project is a humanitarian endeavour to make mobile telecoms available to all people, regardless of income, location or disaster, by allowing off-the-shelf mobile phones to communicate directly without needing cellular towers.

What inspires Paul?

"I would say it's Purism (**https://puri.sm**), the guys making the fully open laptops and phones, because this is so necessary. We also know some of the folks doing Qubes OS to really provide people with as secure an operating environment as they can."

Above The newly manufacturable Serval Mesh Extenders are used to connect communities together without the need for a cellular tower

I still remember hearing about the Haiti earthquake," says Dr Paul Gardner-Stephen. Uncharacteristically for Paul, who tends to ride a bike everywhere, he was driving a car to work that day. "I had the radio on so I was hearing about the earthquake as it was happening." Paul's day job is a senior lecturer for the College of Science and Engineering at Flinders University, Adelaide, Australia, but for the last eight years he's also been running the Serval Project, a not-for-profit that has created an open source mesh mobile telephony system for disaster zones. We're talking over a pixelated VoIP call, but his warm personality comes through the poorly rendered view of his office. He cracks humorous asides during our conversation, but as he recalls how he felt about Haiti he says it still feels raw: "I knew instinctively that within about three days, without law and order being able to coordinate their actions, that this leads to lawlessness." The earthquake in Haiti ultimately affected an estimated three million people and killed as many as 160,000, but it was also the humanitarian crisis that followed that shocked both Paul and the world.

The airport in Haiti was severely damaged, and every road had been destroyed by the earthquake. The final option left, it seemed to Paul, was for aid to come via sea from the Dominican Republic. He knew it would take about two days to sail around to Port-au-Prince. It'd be tight, he felt, but law and order would be restored and the people would get everything they needed. But then the woman on the radio said the harbour in Port-au-Prince had collapsed. At that point Paul realised that things were going to go very, very badly for the Haitian people. "Unfortunately, history shows that that was what happened. There were rape gangs going around, militias and all sorts of nasty stuff because the situation was just so desperate and so uncontrolled," says Paul.

Paul's main reaction was that this should never happen again to people in such dire situations. As a Christian, he

also felt strongly that he should be using his gifts to help other people: "I realised at that moment that I had the right background to make telecommunication systems that could work in disaster zones. No one else was bothering to do it... It was a bit like the story of Esther in the Old Testament: Perhaps you've come to this position for such a time as this? And that really set me thinking – how can we do this? What can we do?"

Some of Paul's initial ideas he admits were a little big and crazy, including "transformer-like shipping containers" that turned into phone towers on the ground after being airdropped in. But then it dawned on him, not without a little anger, how everyone in Haiti had had the hardware they needed to communicate: "It's called a mobile phone. You can talk to a phone tower with an aerial in the right place 20 kilometres away and they are little computers," explains Paul. "They can do funky stuff, computing and network, input and output device." However, the overriding commercial model for telephony is the exact opposite of this: "Make sure phones can't work without phone towers" is the name of the game, Paul believes. Because of the sunk capital, management shuts down anything that will undermine the monopoly power that the phone tower networks create – a flawed model that is beginning to change.

That set Paul thinking about what he could do. Then the first commercially released Android phone was released, the T-mobile G1 (also called the HTC Dream),

> "We wanted to basically make phones communicate with one another"

and when they started being available on the second-hand market, Paul thought: "This is the answer. It's a programmable mobile phone. We can't reprogram the main radio, but we can, at least, reprogram the Wi-Fi radio and prove to the world that this is possible to do. That way we *can* make mobile phones communicate without towers and that it doesn't require any additional expensive hardware. That is purely a software thing that needs to happen."

In what was to be the start of endlessly applying for grants, Paul took six minutes to write a 250-word summary of his idea to the Awesome Foundation (**www.awesomefoundation.org**) based in Boston. The foundation supports 'awesome' ideas to the tune of $1,000: "We said we want to basically make phones communicate with one another. We need 1,000 bucks to buy some G1 phones so we can use those as the experimental basis for it, " says Paul. Six days later, after clarifying he was going to make the whole project open source, the money was on its way. That was the beginning of the Serval Project, so with a French exchange

SPOTLIGHT

Helping Red Cross

Above A Serval Mesh fly-away kit for humanitarian use by the New Zealand Red Cross (NZRC). Paul says the cases can hold 8 or 16 units per case. (Human not included.)

Disaster zones are often characterised by a lack of usable communications infrastructure, and resort to 2G networks (with or without data) or use satellite communications. This creates data transfer issues: "The nice apps for field data produce monstrous XML files," says Paul. "It's a lovely exchange format. I love it in that regard, but unfortunately when it takes 100 kilobytes to house the answers for 10 yes or no questions, it's not ideal from a data transport perspective."

Being able to quickly collect data and analyse it easily after a disaster is crucial to any aid effort. The New Zealand Red Cross's (NZRC) Emergency Response Unit (ERU) for IT and Telecommunication has been gradually moving towards digitalising data collection and Serval has been helping by developing the Succinct Data concept, a low-cost critical data collection method. Paul explains "Red Cross have been looking to capture that data much more rapidly, more accurately, avoid transcription delays and errors and to act on it. Succinct Data was really designed around that."

Serval takes the data from the likes of ODK (Open Data Kit, **https://opendatakit.org**) and assesses the XML and XML stylesheet: "Basically we go 'What really matters is these 26 bytes of information' and we do crazy data compression on that." Paul says that the short-text message compression scheme used for the NZRC and Serval is the best there is: "We were able to match or beat all of the claims of closed source products and to easily beat the only other open source example we found. We now believe that SMAC, our Short Message Arithmetic Compressor, (**https://github.com/servalproject/smac**), is the best-in-class – open source, closed source or otherwise in that space."

QUICK FACT

Early warning system

Serval has also started work on prototyping a satellite-broadcast bridge as part of a Humanitarian Innovation Fund (HIF) grant. This will be used as a "proof-of-concept low-cost tsunami/cyclone early warning system" that, Paul says, will also provide weather forecasts and other useful information via Serval all year round.

student with a background in telecommunications and applications, Paul spent a hectic four months working at what amounted to two full-time jobs to try to bring the project together: "There were a whole number of fun problems that we had to solve on the way software-wise, because we were basically porting Asterisk to Android." This is an open source framework for building communications applications (see **www.asterisk.org**). "We had a complete PBX running in the phone. A SIP [Session Initiation Protocol] client to talk to the local Asterix instance and ignore the phone network's speed and work out what codec would work. Then our missing glue that we created, the first piece of the Serval project, was something called Distributed Numbering Architecture [DNA]." Essentially, this is an app for phone numbers on a mesh network. "We had this mostly cobbled together, not quite working, and we thought, how could we accelerate this process from here?" recalls Paul. The team felt that academic publishing was not going to cut the mustard, but that general media publicity would be the best way to do it. Given Paul's previous media experience as the inventor of a shoe phone in 2007, he decided to contact people he knew at the national broadcaster, the ABC.

> "I think we got the software working six hours before we were about to climb into the aeroplane at 4am"

Paul felt they needed to do something "visually impressive and that carried the point." So not only did he tell them they were trying to make phones work without phone towers for disaster use, but that they were going to test the system in the outback where there are no phone towers for hundreds of kilometres. ABC were delighted but that left one problem – they software didn't work: "I think we got the software working six hours before we were about to climb into the aeroplane at 4am the following morning. Everything was really dependent on that day succeeding, and fortunately it did."

The team managed to get some fantastic footage of the system running, and the TV publicity led to Serval being discovered by New Zealand Red Cross's Emergency Response Unit (ERU) for IT and Telecommunication. "That's one of five globally that do humanitarian telecommunications support for Red Cross and allied organisations," says Paul. "The lead of the ERU at the time, Matthew Lloyd, was a really key figure, as he had operational experience in disaster response. I had great ideas and wanted to make a difference and together we were able to really guide the development of Serval over the years" (see Helping Red Cross, p27).

The Serval Project has been gaining momentum from that point onwards, although not greatly helped by the web giant, Google. Paul says the corporation has made it practically impossible to do ad hoc Wi-Fi on Android phones since Gingerbread onwards (Android 2.3–2.3.7). This led Serval to create the Mesh Extender device to do the relaying between the phones. "In retrospect this was a really good idea," says Paul. "It has kilometres of range and lots of people can share a unit in a location, and the power consumption is much lower than the phones trying to mesh with one another. We can have the apps to download from the device. There's a whole lot of nice things we can do around that." Last year also saw the Serval Project achieve manufacturable mesh-extended devices, including injection-molded housing and custom circuit boards.

2017 was also the year in which Serval was asked to run a pilot scheme in Vanuatu, an island nation in the South Pacific, with the support of the Australian Department of Foreign Affairs and Trade (DFAT). Vanuatu's 65 inhabited islands have suffered greatly from natural disasters. In fact, the United Nations classifies Vanuatu as having the highest natural disaster risk of all the countries that it has assessed. Paul explained: "It has a number of active volcanoes and it's in a very exposed position in the Pacific Basin for tsunamis. They can get wildfires during the dry season, they can get floods and landslides – you name it. I'm not sure they can get tornadoes or they've had them recorded, but that seems to be the only thing they don't seem to get."

In March 2015, Vanuatu was hit by its worst natural disaster to date, Cyclone Pam. The United Nations recorded 16 deaths and UNESCO estimated a recovery and rehabilitation bill of $268.4 million dollars. "The signs and damage of Cyclone Pam are still everywhere," says Paul. For instance, almost three years later a five-storey office building in downtown Port Vila, the capital of Vanuatu, still has a tarpaulin for a roof. Head out to the villages where the Serval Project is working and the houses are temporary dwellings. They are made of wood and bamboo, Paul tells us, because their more permanent dwellings that were made of concrete slabs and masonry construction were blown away by the cyclone. "In one of the villages, they have a mark where

Below Installing Serval Mesh Extenders in Epau, one of the two villages targeted on the island of Efaté

Above The alpha of the Serval client app is a little clunky, but the project is hoping to get funding to improve the experience.

the storm surge came during Cyclone Pam. The ocean was 50 metres into their village."

Critically, communication across the country was crippled in the aftermath, with only one cellular tower in Port Vila left in operation. Officials also struggled to contact outlying islands where there was poor infrastructure, so Serval's work is vital and very much in demand in the villages.

After Paul's last visit, there are now a couple of villages that have a few of mesh extenders each: "We're going to set up a link between those villages that are 9km apart, so a good two- or three-hour walk between them, with the hope that people will use the technology on a day-to-day basis, so that when a disaster does strike they are already familiar with how to use it," says Paul.

Serval is talking to networks and other parties to create transparent mesh-to-SMS gateways, so it can extend communications coverage beyond the edge of some of the networks. "In Vanuatu you maybe have 100km between islands, but we can bridge that all using open source technologies, which to me is just tremendously exciting. And if we haven't already, we will, in due course, save livelihoods as well as lives, which is also key."

Serval isn't about plopping down networks for villagers and walking away; for Paul, the dream is that in the people will deploy it themselves. "One of the really nice things about open source software is that it lets people provide for their needs in a permissionless manner. They don't need to ask someone else to solve their problems. So we're trying to complement that in the communications space so if a village lacks communications to the outside world or to the nearby villages, they should be able to solve it themselves."

"We're getting messages sent to the whole community and the software that we're deploying there is the alpha version of Serval Chat, which is the follow-on from the Serval Mesh app, so it has SMS-like text messaging which we call MeshMS and Twitter-like microblogging that we call MeshMB. We've shown people how to briefly use those and came back a couple of days later and there's public MeshMB messages sent to the community by people that we'd never met. So they'd worked out how to get the app onto their phone without there being a signal whatsoever."

"It's really satisfying to know we've created something of value to people to the point where they are actively seeking us out to get it. They know there's a cost involved as, unfortunately, we can't make it for free, but there is this felt need around communications and that's around the world. It's such a deep human need. That's why it's such a joy, a motivation and a privilege that we can try to solve this problem for people." ■

ROAD MAP

The future of Serval

In the short term, Serval's plans are to make an SMS-to-MeshMS gateway. "We're actively working with the carriers in Vanuatu to come up with a plan around that and to seek funding to get that implemented," says Paul. Fiji and a number of the other pacific nations has also expressed strong interest in expanding the Vanuatu pilot into their respective countries, but first, Paul says, they have a list of software and a couple of hardware issues to deal with. "There's every indication that once we've corrected those issues, we'll have a minimum viable product that will provide message delivery over kilometres and multi-hop with acceptable delivery timeframes," says Paul. Once that's complete, he hopes to keep expanding and get the funding for the alpha client: "We need to really improve the end-user software so that it's a delight to use."

Paul also mentioned that businesses and startups are beginning to consider Serval as a platform for which they can build services that are economically valuable. "Our hope is that people will set up a Serval Alliance – like you have the Wi-Fi Alliance and the Bluetooth Consortium and these kinds of groups – based around Serval Mesh. That will be a vehicle where conventional businesses can support the ongoing development and testing of Serval with their products for everyone's benefit and, of course, that shares the cost down and at that point we probably truly have financial stability and sustainability."

John Gowers

John is a university tutor in Programming and Computer Science. He likes to install Linux on every device he can get his hands on, and uses terminal commands and shell scripts on a daily basis.

Resources

A terminal running the Bash shell
Standard on any Linux distribution

GNU Make
Included on most Linux distributions – can be downloaded from https://www.gnu.org/software/make

PART THREE

Build programs with GNU Make: multiple directories

When putting together a project that spans multiple directories, there are a number of tricks we can use

Almost all large software projects end up spanning a large number of directories. Projects use directories in order to organise a project into coherent self-contained pieces so that individual developers can work on particular parts of them. Indeed in some languages, such as Java, some sort of directory structure is actually enforced by the language. When it comes to creating a makefile in order to build a project, it can be hard to keep track of all the files that occur everywhere in a project, and developers use a number of techniques to organise their Make builds and make them reflect their projects' directory structure.

We'll be examining a couple of commonly used techniques. A classic way of organising builds for multi-directory projects is to have a separate makefile inside each directory, and to have Make call these makefile recursively; that is, include the `make` command itself as part of the recipe for a rule. This allows us to split up our build across multiple makefile, each of which has responsibility for its own directory.

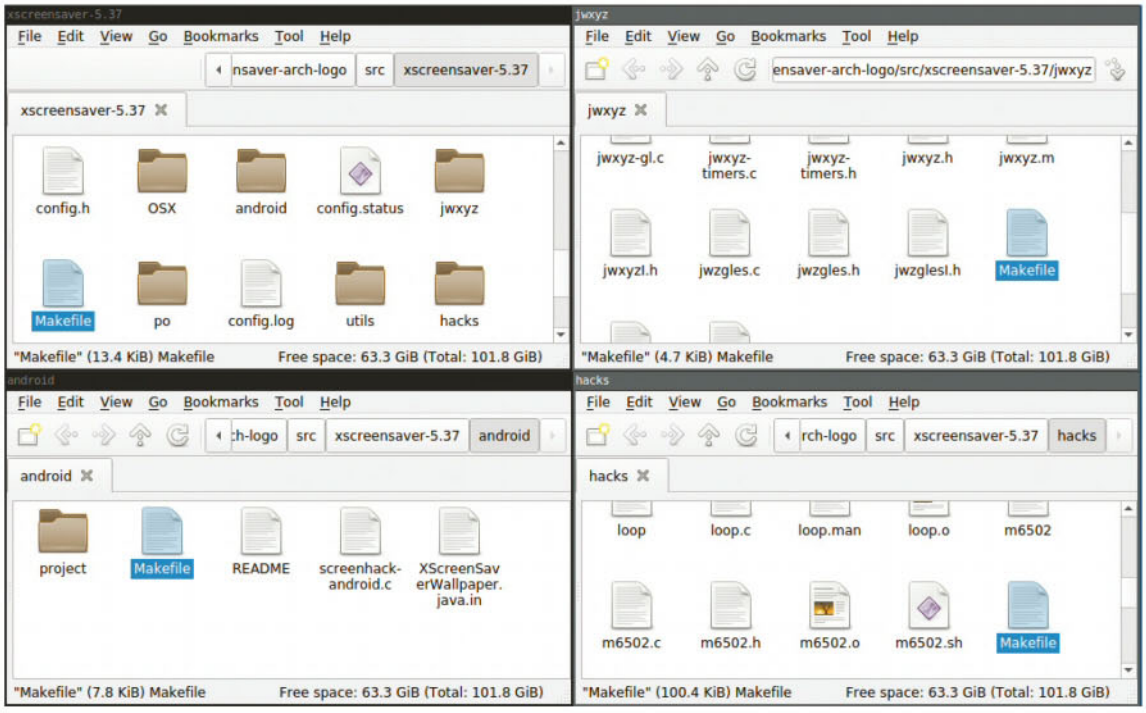
As we'll see, this approach has drawbacks as soon as we start to get non-trivial dependencies between different directories, so we'll also look at an alternative.

Create directories automatically

When we build a multi-directory project, not only the source files live in separate directories; the output target files also need to be organised in some way. So we need to ensure that when we create a new output file, the directory it is going to live in exists.

We could create all these directories manually and distribute them along with our source files, but this would be a bit difficult to maintain. Furthermore, we might want to allow the user to configure the directory structure that we use for output files. It's better, then, to let the makefile create these directories itself.

This is easy enough to do: we can add the directory as a prerequisite of the output file, as in **Figure 1**. Here, the rule for the program `player` makes sure that the directory `target` exists before compiling the files and moving the



compiled program into that directory. If `target` does not exist, Make calls the recipe `mkdir target` to create it. However, there is a problem with this approach. To see why, we need to remember that adding a rule A as a prerequisite to another rule B does two separate things:

1. It makes sure that the target for rule A has been built before running the recipe for rule B.
2. If the file corresponding to rule A is out of date, it triggers a re-run of the recipe for rule B.

In the case of the rule `target`, however, we are only interested in number 1. That is, we want to make sure that we run `mkdir target` before we try to move any files into that directory, but we certainly don't want to trigger a rebuild of `player` every time the directory `target` is flagged as being out of date. In Linux, a directory's time stamp changes every time we modify a file inside the directory: so if we modified just one file inside `target`, Make would rebuild every file inside that directory.

Luckily, Make provides a mechanism for telling the makefile that a prerequisite should only fulfil function 1. Such a prerequisite is called an order-only prerequisite, because it only affects the order in which rules are carried out, and does not prompt rebuilds if it goes out of date. To show that a prerequisite is an order-only prerequisite, we put it on the right-hand side of a pipe character `|` like so:

```
$(TARGET_DIR)/player: player.o dvd_read.o screen.o | $(TARGET_DIR)
	$(CC) -o $@ $^
```

We can have multiple normal prerequisites and multiple order-only prerequisites for a rule. The syntax for specifying them is that all the normal prerequisites go on the left side of the pipe and all the order-only prerequisites go on the right side. For example, in

```
rule_name : normal prerequisites | order only
```

`normal` and `prerequisites` are normal prerequisites, while `order` and `only` are order-only prerequisites. If a rule has only normal prerequisites, we can omit the pipe character (as we have been doing in this series of tutorials up until now).

Calling Make recursively

With a multi-directory project, it can be hard to keep track of everything if we use a single makefile to try to build everything. One common pattern is to create a separate makefile in each source directory, and to tell each makefile to call the makefile in its subdirectories. For example, suppose that our program relies on some image files, in a directory called `pictures`, that are generated from code in some way. We could write a makefile to explain how to make the images and put it into the directory `pictures`. We could then add the following into the makefile in the root directory:

Figure 1

```
TARGET_DIR = "target"

$(TARGET_DIR)/player : player.o dvd_read.o screen.o $(TARGET_DIR)
	$(CC) -o player player.o dvd_read.o screen.o
	mv player $(TARGET_DIR)

$(TARGET_DIR) :
  mkdir $(TARGET_DIR)
```

```
IMAGE_DIR = "pictures"
TARGET_DIR = "target"

.PHONY: images

images :
  cd $(IMAGE_DIR) && $(MAKE)
```

Above Adding the directory where an output file will live as a dependency of that file can cause unnecessary rebuilds of that output file

The important part of this is the line `cd $(IMAGE_DIR) && $(MAKE)$`, which runs the makefile used to create the image files. The variable `MAKE` is another special automatic variable used by Make, and you should always use it rather than calling `make` directly – as in `cd $(IMAGE_DIR) && make`. One advantage of using `$(MAKE)` rather than writing `make` is that if we have multiple possible executables called `make` (for example, multiple different versions of the Make program), `$(MAKE)` will always expand to give the path to the executable that was used to call the original makefile. If we call `make` with any

> "To show a prerequisite is an order-only type, we put it on the right-hand side of a pipe character"

command-line flags, these will be passed on to all our recursive invocations of `make`.

Another alternative to `cd directory && $(MAKE)` is to use the `-C` flag to `make`, which causes Make to run the makefile in a separate directory. For example:

```
$(MAKE) -C $(IMAGE_DIR)
```

The typical usage is to declare some phony rule (`images` in our case) that will call the makefile in the subdirectory. Then, if we run the rule `images`, we cause Make to call the makefile in the `pictures` directory and build the pictures.

Another way to do this that works with multiple subdirectories is to add the directories themselves as phony rules, as in **Figure 2**. In this case, we have one rule, `build_subdirs`, that will be responsible for building all the subdirectories. This rule has no recipe, but lists

Make variables for commands

Sometimes in this article we've used the make variable $(CC) to stand in for the C compiler cc or gcc. The idea behind this is to make the Makefile as portable as possible. Although we know a compiler will be used, we cannot assume that the user will always have the same compiler as us, even if it is something standard such as gcc – so Make provides a number of variables, such as $(CC), which automatically provides the appropriate programs on that particular system.

Figure 2

```
SUBDIRECTORIES = images stylesheets audio

.PHONY build_subdirs $(SUBDIRECTORIES)

build_subdirs : $(SUBDIRECTORIES)

$(SUBDIRECTORIES)
      $(MAKE) -C $@
```

Right By marking the rules for building each subdirectory as .PHONY, we can ensure that they are always run, even though they share a name with directories that might not be out of date

each of the subdirectories as prerequisites. We have a separate rule for each subdirectory, which calls the makefile residing at that location. We need to declare these rules to be .PHONY too, otherwise Make will assume that they refer to the actual directories themselves, and might not run them (since the directories might not be out of date, as far as this makefile is concerned).

Pass variables between instances

With a fairly complicated makefile where we have called Make recursively, we might well want to pass the values of variables between the different invocations of Make. For example, we might want to use the images in the **pictures** folder throughout our project, and we don't want to have to respecify the variable **IMAGE_DIR** in every single makefile. Unfortunately, if we define **IMAGE_DIR** inside one makefile, it won't be accessible in the others.

The solution to this problem uses the fact that there is a close correspondence between Make variables and environment variables in the shell that Make uses to run. For every environment variable in the shell, Make creates its own variables with the same names. So we could get around the problem by declaring the variables that we want to be accessible in all makefiles in the shell, before running Make:

```
$ export IMAGE_DIR=pictures
$ make
```

Then we can access this variable using **$(IMAGE_DIR)** from inside any makefile. However, we want a Make-only solution, particularly since we might want to modify the value of the variable inside our makefile, or set it using special Make-specific functions such as **patsubst**. Luckily, Make allows us to turn a Make variable into an environment variable, using the **export** keyword:

```
export IMAGE_DIR = pictures
```

When we write this, we ensure that the variable **IMAGE_DIR** will be accessible to all invocations of Make. We can use the keywords **export** and **unexport** to change the scope of any Make variable:

```
TARGET_DIR = target

export TARGET_DIR
   # Accessible to all invocations of Make

unexport TARGET_DIR
   # Can only be accessed within this makefile
```

The 'multiple makefile' approach is a simple way to separate the build process into separate parts. If the different parts of a project are truly separate, then it can work very well. However, there are a number of drawbacks to separating out parts of the build into completely separate makefiles.

The main problem is that the separate invocations of Make are *completely* separate. They can share variables, as we saw in the previous section, but they do not share the underlying model. In particular, we can't really specify a rule from one makefile as a dependency of another.

> "For every environment variable in the shell, Make creates its own variables with the same names"

To see how the problem arises, let's go back to our images example. Suppose that we are building a program, **gamepad**, whose build relies on the existence of header files called **ArrowButton.h** and **Screen.h**. These header files live inside a directory, **include**, and are not written by a human, but are automatically generated using some separate process. This process is carried out inside a separate makefile living in the **include** directory. We should add the image files as dependencies:

```
.PHONY: include
include :
      $(MAKE) -C $@
gamepad : gamepad.c ArrowButton.h Screen.h
      gcc -o gamepad.c
```

Since **ArrowButton.h** and **Screen.h** are built by the makefile inside the **include** directory, they are not listed as rules within this makefile. That means they are treated as source files, and Make will refuse to continue if they're not present. As long as we remember to run **make include** before we run **make gamepad**, then, we should be okay. As an improvement to this, we might add a rule that will ensure files inside the **include** directory always get built using the makefile in that directory:

```
include/* :
      $(MAKE) -C include $@
```

The problem with this is that our existing makefile knows nothing about what the prerequisites are for, say, **Screen.h**, inside the second makefile. Of course, we could always add more information to the first makefile

Using shell for loops

Some Makefiles you will see in the real world, such as the one shown in **Figure 3**, use a shell for loop in order to call the Makefiles in a number of subdirectories. That is, they have a rule that uses the for loop from the shell to iterate over a list of subdirectories and call $(MAKE) in each of them. The approach we have outlined is better, though; for example, it lets Make take advantage of parallel execution.

to explain this, but then we end up with a large amount of repeated code, and we start to lose the benefits of having multiple makefiles in the first place.

The only real approach to the problems is to use a single invocation of Make, rather than calling it recursively. Luckily, this approach does not force us to use a single, giant makefile, since Make allows us to include makefiles inside one another, using `include`.

This suggests a strategy for dealing with large, multi-directory projects. We can still put a makefile in each directory, but we should `include` these makefiles into our base makefile, rather than call them recursively. For example, the following code will insert the makefile for the directories `pictures` and `include` into our makefile.

```
include pictures/makefile
include include/makefile
```

When adopting this approach, we should make sure that none of the variable names in the `include`d makefile coincide with any of the variable names in the base makefile, since `include` does nothing more than include the text from the given file directly.

Automatic building rules

Suppose we are building a large C program. We might have many rules of the form

```
component.o: component.c some.h header.h files.h
```

where the header files are those included in `component.c` using the `#include` directive. Since we might be adding `.h` files to and from this `.c` file all the time, we don't want to have to modify the makefile every time.

Luckily, the `gcc` compiler has some command-line flags that allow us to automatically generate a Make prerequisite list for a given `.c` file:

```
$ gcc -MM -MG player.c
player.o: player.c player.h screen.h dvd read.h
```

The nice thing about that is that we can pipe the output of this command into a file, and then use an `include` keyword to put it into our makefile. Even better, we can have the makefile itself automate this process!

The traditional name for one of these so-called 'dependency makefiles' is formed by replacing the `.c` on the end of the filename with `.d`. Thus, we could have a pattern rule

```
%.d : %.c
	$(CC) -MM -MG $(CFLAGS) $* > $@
```

which will run the command `gcc -MM -MG` (plus any additional options to `gcc` specified in the `CFLAGS` variable) over a given `.c` file, producing a `.d` file containing the rule and the appropriate list of prerequisites. For example, running `make player.d` would produce a file `player.d`

Figure 3 = tar

```
21 MAKE_SUBDIR  = for dir in $(SUBDIRS); do (cd $$dir; $(MAKE)
   $@) || exit 5; done
22 MAKE_SUBDIR2 = for dir in $(SUBDIRS2); do (cd $$dir; $(MAKE)
    $@) || exit 5; done
23
24 default::
25   @+$(MAKE_SUBDIR)
26 all::
27   @+$(MAKE_SUBDIR)
28 install::
29   @+$(MAKE_SUBDIR)
30 install-program::
31   @+$(MAKE_SUBDIR)
32 install-man::
33   @+$(MAKE_SUBDIR)
34 install-strip::
Makefile                                34,15          4%
```

containing the output `player.o: player.c player.h screen.h dvd_read.h`.. We can then include all of the `.d` files at once:

```
SOURCES = player.c screen.c dvd_read.c

include $(SOURCES:.c=.d)
```

This demonstrates a special behaviour of the `include` keyword: if it can't find the file that you tell it to include, it will try to create it using an appropriate rule. Including these `.d` files will insert the list of prerequisites for the `.o` files, but not the recipe. We will have to provide that ourselves, or let Make deduce it automatically.

This is almost working as we want, but we still need a way to tell Make to rebuild the `.d` files if one of the `.c` files changes. To do this, change the contents of the `.d` files so that they include themselves as targets. That is, instead of the rule `player.o : player.c ...`, we want the rule

```
player.o player.d : player.c player.h screen.h dvd_read.h
```

A typical way of doing this is by using a `sed` command to convert the output from `gcc` into the form that we want. For example:

```
%.d : %.c
	$(CC) -MM -MG $(CFLAGS) $* |
	sed -E 's/(.*)\.o/\1\.o \1\.d/' > $@
```

This works as long as all of our files are in one directory. If this is not the case, the only possible problem is that `gcc` automatically produces a rule that assumes that the `.o` file will be in the current directory. If we want to put the `.o` files in the same directory as their `.c` files, we need to modify the `sed` command slightly to:

```
sed -E 's/(.*)\.o/'$(dirname $*)'\/\1\.o '$(dirname$*)'\/\1\.d'
```

or something similar. By using all these techniques, multi-directory projects are much easier to handle. ■

Above It's quite common to see developers use a shell for loop in order to recursively call Make in multiple subdirectories, but the approach we've outlined in this tutorial is better

Toni Castillo Girona

Toni holds a degree in Software Engineering and a MSc in Computer Security and works as an ICT research support expert in a public university in Catalonia (Spain). Read his blog at **http://disbauxes.upc.es**.

Resources

- **Kali Linux Tools Listing** http://bit.ly/lud_kalitools
- **FOCA** http://bit.ly/lud_foca
- **BetterCAP** www.bettercap.org
- **Coffee Miner PoC** http://bit.ly/lud_coffee
- **mitmproxy API** http://bit.ly/lud_mitm

Security tools: Build your own InfoSec arsenal

Learn to select and use security tools that best suit your pen-testing work needs

Welcome to the InfoSec Arsenal, where you will learn how to arm yourself to the teeth! As a penetration tester, or a security enthusiast, you need tools to speed up your security engagements and tests. We are talking about software, of course, but a good pen-testing drop box wouldn't hurt either (See Column: Add some hardware to your arsenal, p47). From simple yet precise web brute-forcing utilities such as `wfuzz` to complex and powerful reverse engineering frameworks such as `radare2`, there are plenty of tools to suit everyone's taste and engagement needs.

One good thing about pen-testing is that most of these tools ship out-of-the-box with well-known pen-testing GNU/Linux distros, such as Kali Linux or Parrot Security, so go ahead and pick one (or both). You will find that some tools are really old, but they are as useful today as they were ten years ago. DirBuster, for example, is still widely used (we do love it!) because it's fast and gets the job done. It's no longer supported, but it's still included in Kali and Parrot and works (almost) flawlessly. So take a seat, fasten your seatbelt and let's see what's out there...

Meet the arsenal

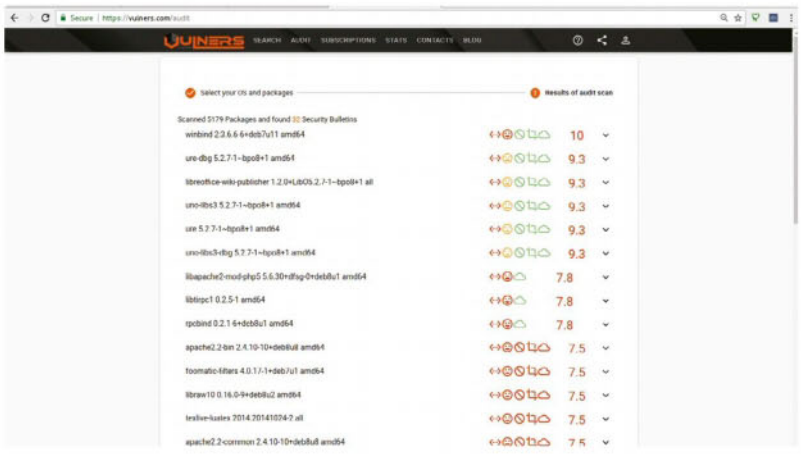
So you have to start pen-testing, and there are hundreds of tools and frameworks out there to choose from and you may feel overwhelmed by all that. Don't panic! First, use the Kali Linux Tools Listing (see Resources) as a reference. Say you need a password brute-forcing utility for a Secure Shell server; look it up on that list, and narrow the possibilities. Not all the good tools are there, of course. Below, we present a list of powerful tools and frameworks we hold dear. We have been using them for quite some time now, and they always deliver.

You can install Kali Linux or Parrot Security on a new virtual machine and you will have most of these tools pre-installed and ready to use. It is worth mentioning that FOCA is a great tool capable of extracting metadata from all publicly accessible documents for a given domain. It does this by using the Google, Bing and DuckDuckGo search engines to look for certain file types (for example `ext:pdf` and `ext:doc`).

After that, it processes all the data gathered and presents a fancy graph to the user with extracted hosts, software used and even feasible usernames. The catch? It was written with Windows in mind. It is now open source though (see Resources) and although it may work with Wine, we use a Windows VM to execute it. We know, this sounds like heresy, but as a pen-tester sometimes you have no choice...

Discovering new tools

There are a bunch of great public sources of information concerning hosts and companies out there, such as Censys (**https://censys.io**), Shodan (**www.shodan.io**), Bing, Google and DuckDuckGo (**https://duckduckgo.com**). Thanks to their APIs, plenty of Passive Discovery tools have been adding support for these services, like the amazing Recon-ng. Recon-ng needs the right API keys for the services you want to use. Navigate to **https://censys.io/register**, create a new account and then get your API keys from **https://censys.io/account/api**. Open a new terminal and run `recon-ng`. Within the tool's CLI, you can enumerate all the available services that need an API key with `keys list`. Let's add your Censys key ID now: `keys add censys_id YOUR_API_ID`. Do the same with your API secret: `keys add censys_secret YOUR_API_SECRET`. If you want to use more services, perform the same steps for all of them (register, get your key, add it to Recon-ng). Now, let's search which module can be used to query Censys: `search censys`. Create a new



Above Who said hackers can't enjoy beautiful and functional GUIs?

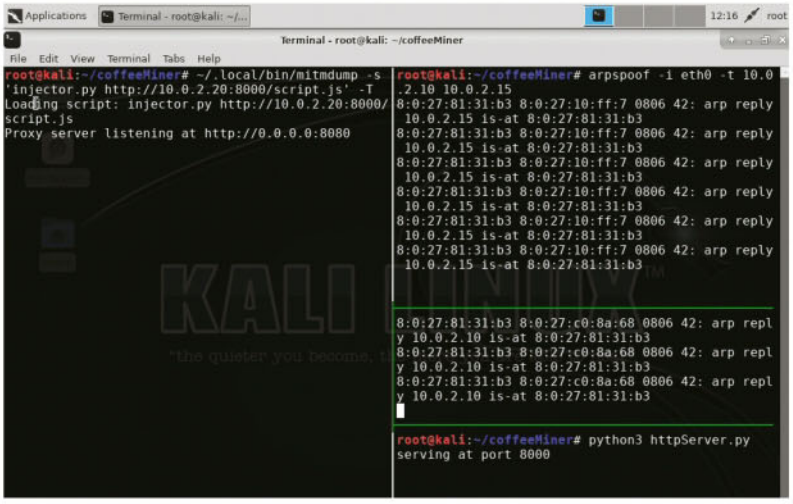
workspace within recon-ng, and add a new domain to it: `workspaces add TEST / add domains TARGETDOMAIN`. Add the company's name now: `add companies NAME~DESCRIPTION`. Then you need to start populating Recon-ng database with hosts from `TARGETDOMAIN`. You can use the Google website for that; first load the module: `use recon/domains-hosts/google_site_web`. Execute it: `run`. After a while you will have a list of hosts that belong to `TARGETDOMAIN`. Resolve all these host names to IP addresses now: `use recon/hosts-hosts/resolve`. Execute it: `run`. Don't forget to reverse-resolve the IP address of each host, in case there are additional hostnames hosted there: `use recon/hosts-hosts/reverse_resolve; run`.

Finally, let's make use of the `censys` module to gather all the open ports for these hosts: `use recon/netblocks-ports/censys`. Because we want to use the IP addresses we have just gathered, set its `SOURCE` option to the result of querying the table 'hosts' like this: `set SOURCE query SELECT DISTINCT ip_address || '/32' FROM hosts WHERE ip_address IS NOT NULL`. Finally, execute the module and wait for it to finish: `run`.

QUICK GUIDE

Essential pen-testing tools

Passive discovery	Recon-ng, The Harvester, FOCA
Active discovery	masscan, Nmap, WhatWeb
Vulnerability searching	SearchSploit, Vulners, Rapid7 Vulnerability & Exploit Database, Lynis, LinEnum.sh
Brute-forcing	Hydra, wfuzz, DirBuster
Password cracking	John the Ripper, hashcat
Word lists	FuzzDB, SecLists, CrackStation
Web hacking	OWASP ZAP, mitmproxy, WPScan, sqlmap
Network attacks	arpspoof, BetterCAP, Wireshark
Reversing	GDB, radare2
Binary emulation	Radare2 (ESIL), angr
Firmware and binary analysis	Binwalk, VirusTotal, Firmalyzer
Frameworks	Metasploit, Veil Framework, OWASP ZSC, WordPress Exploit Framework



ArnauCode 2018 (http://arnaucode.com/blog)

Left Creativity and tool-leveraging are key for a pen-tester. Oh, and some coffee...

You can get all the open ports now by querying Recon-ng database: `show ports`. Are you interested in Telnet ports only? Execute: `query SELECT * FROM ports WHERE port=23`. Feel free to explore a bit more; Recon-ng modules are written in Python, so you can easily extend its functionalities.

At some point, you will have no choice but to actively engage with the systems you are testing. After gathering all those hosts using Recon-ng, you can execute `masscan` to actively port-scan a whole IP range using the CIDR notation in no time. Even if you have different IP ranges, you can scan them all too. Enumerate all the IP addresses from Recon-ng: `show hosts`. Make sure to drop packets destined to a TCP port of your choice (say, 6666): `iptables -P INPUT 1 -i ethX -j DROP` and then feed masscan with the CIDR notation of all the IP ranges you want to port-scan along with the chosen port that will receive the packets: `masscan 192.168.1.0/24,192.168.2.0/24,... -p21,22,23,80,443,3389,... -e ethX adapter-port 6666 banners -oJ results.json`. Feel free to add more IP ranges or ports to scan. We use masscan a lot: it can port-scan the whole TCP port range

> FuzzDB, SecLists and CrackStation word lists are all ideal companions to any brute-force utility

(-p1-65535) of a single host in minutes. Once the huge scan is complete, fine-tune your findings by executing `Nmap` against those hosts/services you are after. Nmap enables you to perform vulnerability assessment and exploitation as well thanks to its powerful LUA script engine. Let's say you have opened the JSON file generated by masscan and there's a web server running on your network. You want to test it against ShellShock: `nmap -p80 -script=http-shellshock SERVERIP`.

Leverage everything

As a pen-tester, take advantage of everything that's accessible and use it during engagements. For instance, GitHub released a vulnerability scanner in November last year, capable of flagging vulnerable Ruby and NPM dependencies in public repositories: see **http://bit.ly/lud_vuln**.

AI applied to security

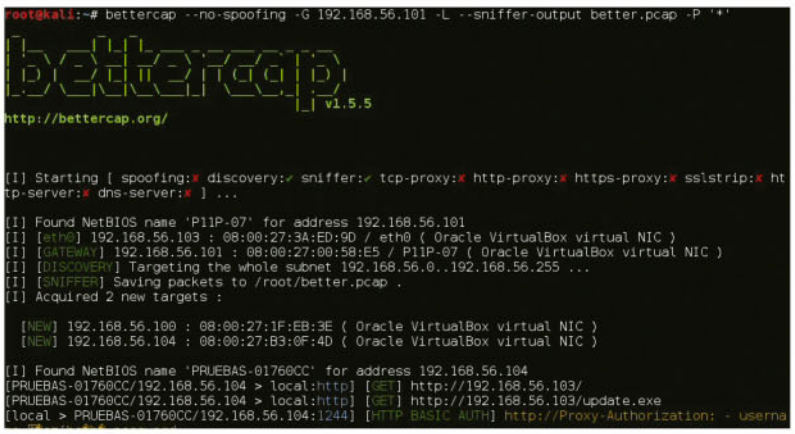
Since TensorFlow made its appearance, some open source projects have been using it to develop partially automated analysers that can flag malicious incoming requests to applications, or perform network-packet forensics. One such project is SecuritAI (see **http://bit.ly/lud_securitai**), born from Keras, a high-level neural-network API than can run on top of TensorFlow.

Nmap ships with a lot of useful scripts – see **https://nmap.org/nsedoc** for a list of them.

We prefer looking for vulnerabilities avoiding, whenever possible, active engagements with our targets. So we make heavy use of Exploit-DB (**www.exploit-db.com**), Vulners (**https://vulners.com**) and Rapid7 Vulnerability & Exploit Database (**www.rapid7.com/db**). One amazing thing about Exploit-DB is its bash script to query the database: SearchSploit (see Tutorials, p44, **LU&D187**). If you already have access to a particular computer (maybe you are assessing the security of one of your own systems), you can use the Vulners Audit Tool to look for potential flaws. Try it: navigate to **https://vulners.com/audit**, choose your OS and then execute the suggested command to list all the OS packages. Paste the results into the 'Paste list of Packages here' text area, type your OS version in the 'OS version' text box (get it by running `lsb_release -rs|cut -d"." -f1`) and click Next. To help you copy/paste the package list, install `xclip`: `apt-get install xclip`. Then, pipe the two commands together; so for Debian-based distros: `dpkg-query -W -f='${Package} ${Version} ${Architecture}\n' |xclip -i -selection clipboard`. Now go back to Vulners and press **Ctrl+V** to paste the output.

All the vulnerable packages will be shown along with their CVSS score. If you click them, you will see all the references to their security advisories, possible public exploits, and so on. These sources are great once you have set the first foothold into the system too. As a non-privileged user, you always need to find ways of escalating privileges, so if you happen to find a potentially vulnerable package installed on the system thanks to Vulners, you can pat yourself on the back! Apart from Vulners Audit, you can use Lynis (**https://cisofy.com/lynis**), LinEnum.sh (**https://github.com/rebootuser/LinEnum**) or even vulners-scanner (**https://github.com/vulnersCom/vulners-scanner**), but these must be copied to the target system and executed there, something that is not always feasible or desired. LinEnum is a great standalone Bash script that does not require an internet connection and it will help you find weak spots on the target to escalate privileges (such as poorly set-up **cron** jobs or **sudo** entries). We'll cover privilege escalation in a tutorial soon.

Below We are so sorry, Ettercap, but it's time for us to move on...



BetterCAP

Brute-forcing your way

When it comes to brute-forcing services, Hydra is the tool of our choice. If it's about brute-forcing web files and directories, wfuzz and DirBuster are the winners. Anyway, you will need a bunch of good word-lists or you will get nothing out of these tools. FuzzDB and SecLists are two huge collections of predictable paths, usernames, passwords, and well-known injection points; they are all ideal companions to any brute-force utility. We suggest you clone their repositories under `/usr/share/wordlists` on your Parrot VM just for coherency: `cd /usr/share/wordlists; git clone https://github.com/fuzzdb-project/fuzzdb.git; git clone https://github.com/danielmiessler/SecLists.git`. Some of the word lists

> As a pen-tester, you should take advantage of everything that's accessible and use it during your engagements

in FuzzDB can be found in SecLists too. From time to time, make sure these lists are up-to-date by running `git pull` within their respective directories. CrackStation (**https://crackstation.net**) is another amazing source of word lists, dictionaries and password database leaks.

As a pen-tester, you will have to create your own word lists sometimes. We've been in a bunch of real engagements and CTFs so far, and there's one thing that many less savvy sysadmins tend to do when it comes to their websites: they make copies of some files right there on the production server, which means you can always get that copy and see what's in there with a simple HTTP GET request!

If a particular web server is running some well-known open source project like WordPress or pfSense, you can brute-force those juicy files with different extensions such as `.old`, `.bck`, `.swp`, `.~` and so on. Why don't you try it? Execute `wfuzz` with two different word lists; the first one will hold hundreds of common filenames whereas the second one will hold a bunch of well-known backup extensions. The tool will create their Cartesian product automatically for you:

```
wfuzz -t 50 -c -w /usr/share/
wordlists/fuzzdb/discovery/predictable-
filepaths/KitchensinkDirectories.
txt -w  /usr/share/wordlists/fuzzdb/
discovery/predictable-filepaths/
filename-dirname-bruteforce/
Extensions.Backup.txt --hc 401,403,404
http://WEBSERVERFUZZ.FUZ2Z
```

wfuzz will replace the first **FUZZ** string with words from the first file, and **FUZ2Z** with words from the second file.

Should there be a third word list, add FUZ3Z and so on. wfuzz is amazing because it can be used for a lot more than just brute-forcing web files and directories (see **http://wfuzz.readthedocs.io/en/latest**).

Password cracking has already been covered in the magazine (see Tutorials, p40, **LU&D182**), but suffice it to say that John the Ripper and hashcat are the tools you must have and use whenever you're trying to crack any hash. One annoying thing about hashcat, though, is that if you have an older GPU and want to use newer hashcat versions, you probably won't be able to because the hardware won't be supported.

In that case, you could upgrade your GPU – which is a good idea in theory, but with the current craze for cryptocurrency mining pushing up the cost of hardware in this area to a ridiculous degree, probably isn't the most cost-effective solution. Otherwise, you'll have to stick with an outdated hashcat version. If your old hashcat binary does not run any more because it says `This copy of cudaHashcat is outdated`, read **http://bit.ly/lud_outdated_hashcat** for a way of patching the code to remove this restriction.

Intercepting packets

As a pen-tester, the ability to intercept network packets, extract valuable information from them, manipulate them and reply to them is key. So you must perform man-in-the-middle (MITM) attacks from time to time. Ettercap has been around for years and was once the de facto tool for this; almost every single book and tutorial talks about it to perform these sort of attacks. But it's old, and some of its functionalities do not work properly any more. Enter BetterCAP (see Resources). The old arpspoof, on the other hand, is still a good option if you want to combine different tools to achieve your goal. As an example, Arnaucode (@Arnaucode) has developed a collaborative (MITM) crypto-currency mining pool in Wi-Fi networks, wittily naming it Coffee Miner (see Resources). It uses arpspoof to perform the initial MITM attack, and mitmproxy and its API to intercept HTML traffic and then inject a simple line of HTML to the client. We'll cover these techniques in a tutorial soon.

> "Security tools do have security vulnerabilities too, so don't rely on them too much"

Reversing

Being fluent in reversing and exploit development always pays off. The best debugger for us is the good old GDB. When it comes to exploit development, PEDA was a good option back in the day, but now it's all about GEF (**https://github.com/hugsy/gef**) because it's built around an architecture abstraction layer, unlike PEDA. radare2 is an incredible reverse engineering framework that can help you with those IoT binaries extracted with binwalk or with any other sort of binary. Thanks to its ESIL language, you will be able to emulate some parts of a binary, thus saving you the burden of reverse-engineering those obfuscated functions (see Tutorials, p44, **LU&D185**). We've been experimenting with angr too (see Tutorials, p38, **LU&D180**) and we have used them for some CTFs already. The learning curve of these tools is extremely steep, but they're worth the effort!

Frameworks

The good thing about frameworks is that you don't have to reinvent the wheel to perform common tasks while pen-testing. When it comes to reverse-shells, process migration and so on, Metasploit Framework with its incredible payload, Meterpreter, is always a winner (see Tutorials, p44, **LU&D188**). Sometimes uploading meterpreter to a target computer will prove difficult: for these cases, use Veil (**https://github.com/Veil-Framework/Veil**), another amazing framework capable of obfuscating Metasploit payloads (on one particular pen-testing engagement, we had to use Veil to bypass Windows Defender). If you tend to assess WordPress installations, combine WPScan with the WordPress Exploitation Framework (**http://bit.ly/lud_wpexploit**), which includes an amazing assortment of WordPress exploits. OWASP has its own framework too, OWASP ZSC – although it's mainly focussed on generating shell-code and obfuscating both shell-code and scripts (see Tutorials, p44, **LU&D187**).

There are more amazing tools out there that we haven't mentioned here, so feel free to explore them. Use the Kali Linux Tools Listing as a reference. Security tools do have security vulnerabilities too, so don't rely on them too much and make sure you always use a well-protected and isolated computer whenever pen-testing. Happy hacking!

WHAT NEXT?

Add hardware to your arsenal

1 Hak5 drop boxes

Hak5 (**https://hakshop.com**) is a well-known and reputable company providing pen-testers with a wide variety of hardware devices for performing MITM attacks, packet sniffing, keystroke injection and data exfiltration. It has everything you need to drop a box and start engaging: from pocket-sized wireless devices to network implants disguised as USB-to-Ethernet adaptors.

2 iStorage Encrypted drives

You need to securely store all your information somewhere. Apart from encrypting your laptop's hard disk using LUKS, you can also rely on datAshur and diskAshur devices from iStorage. It sells a wide range of USB hard disks, desktop hard disks and USB flash drives with AES 256-bit hardware encryption. These have been designed to be tamper-evident and -resistant. See **https://istorage-uk.com**.

3 Build yourself a pen-testing drop box

Why not build your own customised, pocket-sized pen-testing drop box? There are a lot of cheap devices to choose from. As for software to use with it, try S.W.O.R.D – see **http://bit.ly/lud_sword**.

Nate Drake

Nate is a technology journalist who specialises in information security.

Resources

- **Prevent DoS with iptables** http://bit.ly/lud_dos1
- **Cloudflare: protect against DDoS attack** http://bit.ly/lud_dos2
- **DDoS protection with iptables** http://bit.ly/lud_dos3
- **iptables wiki** http://bit.ly/lud_iptables
- **Linux Audit** https://linux-audit.com/lynis

Secure your system with server hardening

Batten down the hatches and lock down your Linux server by following these handy instructions

If you own or operate your own Linux server, you can congratulate yourself on being in good company: Linux accounts for almost 97 per cent of public server operating systems. This is partly because Linux is free of charge but also due to the fact that popular server distros such as Debian and Ubuntu are open source and highly customisable.

Unfortunately Linux's relative popularity also means that servers are a popular target for attacks, such as Mirai, the staggering denial of service attack in 2016, which harnessed IoT (Internet of Things) devices to overwhelm DNS hosting provider Dyn.

In this guide, you'll discover some ways to 'harden' your server to protect against this and other types of attacks. If you haven't set up your server yet, our distro of choice is Ubuntu Server 16.04.3 LTS as it's one of the most popular and stable on the market today. Support is guaranteed until 2021.

We've assumed that you are comfortable with the steps for installing a server and creating a user account, as well as routine tasks such as how to use **apt-get** to install new programs and have already enabled the UFW firewall. If this is your first time, we encourage you to read the Ubuntu server guide (**https://help.ubuntu.com/lts/serverguide**). Make sure to backup your data before following the steps below.

Secure drives and memory

When setting up your server, you should have been given the option to encrypt your partitions, including the swap space, using LUKS. If you chose a guided installation this is done for your automatically using AES-256 encryption. This means if your server is seized while it's powered off, it would be extremely difficult to access your data.

If you're using Ubuntu Server you also have the option to encrypt the **home** folders for each user account as it's created. If you didn't select this, you can use the **ecryptfs-migrate-home** tool to create a new encrypted **home** folder and move your data there. To do this, simply log in as root and run **secryptfs-migrate-home -u yourusername**. Follow the steps on screen. Once it's complete you must log into your account before the server next restarts. If your swap space is unencrypted, an adversary can also gather data on running processes, so make sure it's encrypted with **sudo ecryptfs-setup-swap**.

You can also encrypt any external drives to which the server is connected using **cryptsetup**. For instance, to encrypt the device **/dev/sda5**, run sudo cryptsetup luksFormat **/dev/sda5**. The utility will walk you through the process of entering a password and formatting the drive. Remember that this means any data already on the drive will be lost.

The server's shared memory space is another vulnerable point. This area is designed to allow programs to exchange data easily, but a malicious program could potentially execute processes you don't want, or change the UID of legitimate programs. Set the shared memory space to read-only firstly by running **sudo nano /etc/**

> "The greatest advantage of unattended-upgrade is its greatest weakness"

fstab then adding the text **none /run/shm tmpfs defaults,ro 0 0** to the end of the file. Although this will hugely improve your server security, remember that certain server applications may need to write to **/run/shm**. Reboot your machine and run **sudo mount -a**.

Automatic updates

Admins often have to strike a balance between running critical updates and keeping the server running. The 'unattended-upgrades' package, which comes preinstalled in Ubuntu Server, offers an elegant compromise. During the setup process Ubuntu will ask if you wish to enable automatic security updates. If you're unsure if you enabled this, you can view your current unattended-upgrades configuration by running **sudo**

nano /etc/apt/apt.conf.d/50unattended-upgrades. Remove the '//' at the start of any line to enable other kinds of update if you want.

Security updates usually require a restart. You can configure the server to do this automatically by uncommenting the line starting Upgrade:Automatic-Reboot and changing false to true. If you are worried about too much server downtime you can also schedule restarts by changing the Automatic-Reboot-Time value. For example: Unattended-Upgrade::Automatic-Reboot-Time "05:30";

You can fine-tune your update schedule even further by running sudo nano /etc/apt/apt.conf.d/10periodic, for instance to:

```
APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Download-Upgradeable-Packages "1";
APT::Periodic::AutocleanInterval "7";
APT::Periodic::Unattended-Upgrade "1";
```

This configuration would update the package lists, then download and apply updates every day. The local download archive is cleaned once a week. Feel free to change these values to a schedule that suits you.

The greatest advantage of unattended-upgrade is also its greatest weakness, in that you won't be there if there are any issues with your configuration. Fortunately you can test your current settings using sudo unattended-upgrade --dry-run -d. Note that these settings won't upgrade the kernel itself. However you can do this in the usual way via sudo apt-get dist-upgrade.

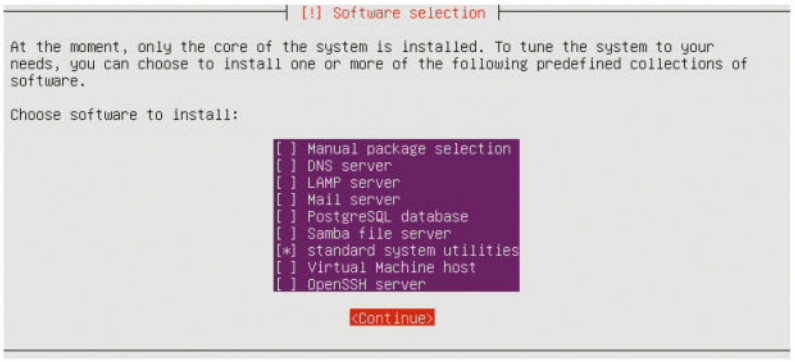
If you run a mail server you can also configure unattended-upgrades to email admins about package upgade issues. See http://bit.ly/lud_autoupdate for more information.

Armour up!

One of the easiest ways to secure your server is to reduce your attack surface. In plain English this involves removing any programs and services your server doesn't need. For example, a simple DNS server doesn't need to run Dovecot, which is used for email.

If you're setting up Ubuntu Server, this is very easy as you can choose 'Manual package selection', then only install the programs you need. If you decided to stick with the default packages that came with your server of choice, run the command netstat -tulpn and sudo lsof -i to determine which processes are listening on your server's ports. You can use systemctl disable <name> to disable services, but this only applies the next time you reboot. Make sure to run a full backup before removing any packages.

Ubuntu also employs AppArmor, a kernel enhancement used to restrict the resources used by programs, for instance which files they can access and whether networking is allowed. The specific rules for each program are governed through the use of profiles. These are small text files and are located in /etc/apparmor.d/. Some programs you install will have their own AppArmor profile and in fact Ubuntu ships more and more with each new release; you can create other profiles yourself. If there's no profile for your chosen program visit https://wiki.ubuntu.com/AppArmor for an example profile.



Above Ubuntu Server lets you choose server-specific software during setup

The most restricted mode for AppArmor is when a profile is 'enforced'. This means that any unauthorised actions are blocked and logged. The easiest way to work with profiles is to install apparmor-utils. You can then choose to enforce specific profiles via the commmand aa-enforce. For instance:

```
sudo aa-enforce /etc/apparmor.d/usr.lib.dovecot.imap
```

You can also run AppArmor in 'complain' mode, where policy violations are logged but not enforced. See http://bit.ly/lud_complain for more information.

Lock down SSH

As an administrator, you'll probably need to log into your server via SSH to run updates and make changes. Unfortunately this also leaves your server at the mercy of bots, which routinely probe vulnerable ports on servers. The SSH port is one of the most targeted.

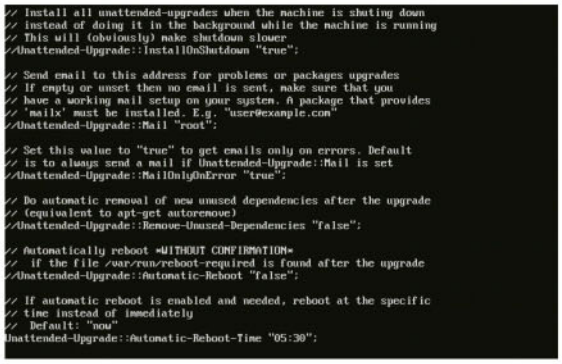
Run sudo nano /etc/ssh/sshd_config to view your current SSH configuration. Your first step should be to change the default port from 22 to another unused one on your system, such as 38757. Use the command sudo netstat -lat to list any ports currently in use to avoid conflicts. By default the root user can log in via SSH. You can change this by altering the value for PermitRootLogin to no.

If you routinely use SSH from a certain location such as another machine on your own network, you can also restrict sshd to certain IP addresses using ListenAddress. Simply uncomment the corresponding line, for example ListenAddress 192.168.1.24. Add other IP addresses (such as those for a remote machine) on a new line. To restrict SSH access to certain users on your server, add AllowUsers on a new line: AllowUsers tom dick harry.

For extra peace of mind, consider installing fail2ban, a handy program which manages your iptables configuration to restrict login attempts via SSH. Once installed, switch to the fail2ban directory with cd /etc/fail2ban. Create your own copy of the configuration file

UFW basics

Uncomplicated Firewall (ufw) is a firewall configuration tool for iptables, and comes preinstalled in Ubuntu Server. You can activate UFW at any time using sudo ufw enable. Use deny to block traffic from a specific IP address: sudo ufw deny from 185.92.25.220. You can also enable incoming traffic with allow: sudo ufw allow 22 or sudo ufw allow ssh.



Right Uncomment lines in unattended upgrades to schedule automatic updates and reboots

by running `sudo cp jail.conf jail.local`. You can then edit this via nano. The `bantime` value reflects how long an IP address will be blocked when an incorrect login attempt is made. By default this is 10 minutes (600 seconds) but feel free to change this.

The `findtime` and `maxretry` values determine the maximum number of failed login attempts allowed during a certain period before a client is banned. By default clients are allowed up to five login attempts every 10 minutes. Run `sudo service fail2ban restart` to apply your changes.

Detect rootkits

Rootkits are a collection of programs designed to hack into your system while masquerading as legitimate software, for example by replacing a legitimate program with one of the rootkit's own. Your best defence against this is to practise good physical security and maintain your firewall, but you can also install a rootkit scanner.

One of the most popular scanners is chkrootkit, which is configured to detect many of the most popular rootkits and worms. By default it does this simply by looking for known signatures in various programs, but you can also run it in expert mode (command line option `-x`) to detect suspicious strings inside binary programs.

chkrootkit makes use of a number of system commands itself, such as find and netstat. Naturally this causes concern if you're using said commands to check for rootkits, as an attacker may already have modified them. Fortunately chkrootkit allows you to specify alternative binaries using the path (`-p`) option. For instance you can insert a Linux CD and run

Below AppArmor controls the files that programs can access by using profiles

`./chkrootkit -p /media/cdrom/bin`. Alternatively, you can mount your server's hard drive on a trusted machine and scan it from there using the `rootdir` option: `/chkrootkit -r /mnt/disk1`.

Another reason we're fond of chkrootkit is that it's easy to perform regular scans. Simply edit the configuration file using `sudo nano /etc/chkrootkit.conf` and change `RUN_DAILY=` from `false` to `true`.

Certain types of rootkit such as SucKIT work by patching the Linux kernel itself. Given how deeply these kind of rootkits entrench themselves into your OS, even hardened penetration testers have difficulty ensuring they're fully removed, even when accessing a drive from a Live CD or another machine. If chkrootkit finds an infection, the best move is to restore your system from a previous backup, then run the scanner again to check that no infected files are found.

Limit DoS attacks

The easiest and best way to protect against DoS (Denial of Service) attacks is to host your domain with a provider which has the infrastructure for load balancing and built-in protections against DOS attacks, such as Cloudflare. If you're managing your own server, though, you will need to work harder to limit excessive traffic.

There are various types of DoS attacks. A 'SYN flood' involves malicious clients sending a number of connection (SYNchronise) messages to a server, but then failing to return the corresponding ACK code until the number of half-open connections is more than your server can bear. Fortunately you can mitigate these by modifying `iptables`. For instance, to limit traffic on port 80 you would use:

```
sudo iptables -A INPUT -p tcp --dport 80 -m state --state NEW -m limit --limit 50/minute --limit-burst 200 -j ACCEPT
```

The `limit-burst` option here is used to allow up to 200 connections before enforcing a limit of 50 per minute. You can also use `iptables` to enforce rules for established traffic on all your ports – for instance:

```
sudo iptables -A INPUT -m state --state RELATED,ESTABLISHED -m limit --limit 50/second --limit-burst 50 -j ACCEPT
```

The above rule allows 50 connections before setting a limit of 50 per second. Modify these values as you like.

If you've used iptables before, you may be aware that it consists of five tables, each of which are made up of chains. These are rules that are listed in order. The above rules involve using the `INPUT` chain and the default `filter` table. The `INPUT` chain is only processed after the `PREROUTING` and `FORWARD` chains. The extra resources for this are minimal in themselves, but will add up during a DoS attack.

For best results, you can use the `mangle` table, which works with the `PREROUTING` chain to process and block as many packets as possible. To block all non-SYN packets

and those which don't belong to a valid TCP connection, run the following:

```
sudo iptables -t mangle -A PREROUTING -m conntrack --ctstate INVALID -j DROP
iptables -t mangle -A PREROUTING -p tcp ! --syn -m conntrack --ctstate NEW -j DROP
```

Oh honey honey

If you've already secured your server against SSH attacks as outlined above, you can set up your own 'honeypot' to log attempted brute-force attacks. Kippo and its more recent fork Cowrie are handy Python programs designed to fool less dedicated hackers into believing they have connected to your server.

By default, the honeypot listens for connections on port 2222, although you can use port-forwarding to make it reachable via port 22 if you wish, for instance via running `sudo iptables -t nat -A PREROUTING -p tcp --dport 22 -j REDIRECT --to-port 2222`. Make sure to follow the steps above to change your server's own default SSH port if you do this.

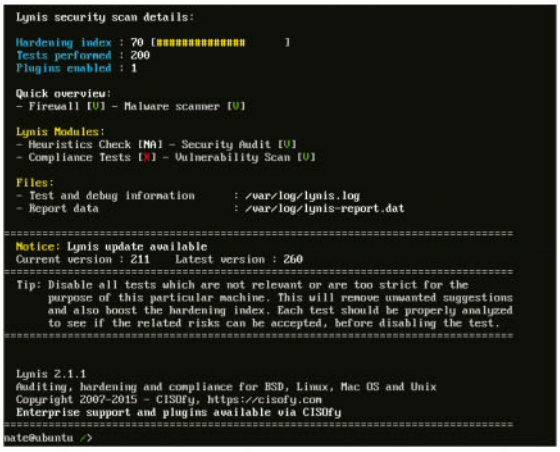
Hackers who succesfully probe and break into the honeypot will be rewarded with access to a fake file system designed to resemble Debian 5.0 (Lenny). The honeypot stores log files to allow you to track what commands are run on the system. Anything hackers download is stored in its own directory (`/home/user/kippo/dl`).

> Hackers who break into the honeypot will be rewarded with access to a fake file system designed to resemble Debian 5.0

The rather dated version of Debian, and the fact that the default passwords are **root** and **123456**, would make most hackers suspicious, but the honeypot may thwart some automated attacks. The Kippo GitHub page recommends running your honeypot on a dedicated, firewalled virtual machine for safety reasons. Both programs will run quite happily on a Raspberry Pi too.

To get started with Kippo, visit **https://github.com/desaster/kippo/wiki/Running-Kippo**. You need to have both MySQL and Apache2 installed on the target machine. If you need in-depth data about hacking attempts, you can also install Kippo Graph (**http://bruteforcelab.com/kippo-graph**). This displays information about hacking attempts broken down by IP address, country and so on.

We can't emphasise strongly enough that honeypots are only for experienced administrators. If you aren't sure what you're doing, you could risk making your server more vulnerable.



Left Lynis runs security audits, then gives your server a score out of 100

Security audits

No matter how safe you believe your server to be, it will always benefit from regular security audits. One of the most popular tools for this is Lynis.

Lynis won't automatically make any changes to your server, but it will perform a number of tests to check for common vulnerabilities. Once installed via the Debian or Ubuntu repositories, you can run an audit immediately with `sudo lynis audit system`. This should take no more than a couple of minutes. Lynis helpfully displays an overall score of 'hardening points' to grade your server security, or lack of it.

For instance, if the audit detects the presence of a malware scanner such as chkrootkit, or you forbid logging in as root via SSH, then your score will increase by 2. Your points are then ranked out of an overall index of 100; during our tests we found that following the steps above resulted in an index of around 70. Lynis records results of audits in `/var/log/lynis.log`. Scroll through these to read any warnings – about needing to set a password age limit, for example.

In total, Lynis runs hundreds of tests and the log file can be hard to navigate. You can make life easier for yourself by narrowing down tests to categories such as Authentication or SSH with the `--tests-category` option. For instance:

```
sudo lynis audit system -tests-category "SSH"
```

One of they key recommendations from Lynis is to install a HIDS (host-based intrusion detection system) such as Tripwire, which is available via the Ubuntu repositories. Tripwire works by gathering information about your server configuration and file system, then storing it. It can then compare this against your current system state at any later point to detect tampering.

This is most effective if you install Tripwire when first setting up your server, as otherwise the program itself may have been compromised and might therefore be of no use. Visit **http://bit.ly/hud_tripwire** for help with initialising the Tripwire database and running system integrity checks.

Password policies

Longer passwords are much harder for hackers to brute-force. You can enforce a minimum password length for users on your server with the PAM component `libpam-pwquality`. Once that's installed, open the file `/etc/pam.d/common-password` and specify a minimum password length of 12 characters by inserting `minlen=12` just before `retry=3`.

Dask: how to do parallel programming the easy way

Parallel computing can be difficult to do well, especially with large amounts of data – but meet Dask

Joey Bernard

In his day job, Joey helps researchers and students at the university level in designing and running HPC projects on supercomputing clusters.

Parallel programming can be one of the messiest techniques to implement. After all, the whole point is to try to run multiple processes concurrently and therefore speed up your computations. Once you start having more than one thing at a time happening, it can get confusing as to what process caused the result you're seeing. Debugging reaches a whole new level of difficulty at this point. Also, in many data analysis tasks you may have very large sets of data that need to be coordinated across all of these parallel processes. You need to be aware of the possibility of corruption of results or repetition of calculations. This can happen if the multiple threads are not properly synchronized when they read or write the data involved.

All of this can be achieved using lower-level functions available within the Python standard library, but there's no reason to do all the work yourself. The Dask library has been developed to help manage all of the complexities involved when you move to using more than one CPU core. It provides both the functionality to create and schedule tasks, as well as several classes to manage data in a way that is aware of parallel task execution.

Install Dask

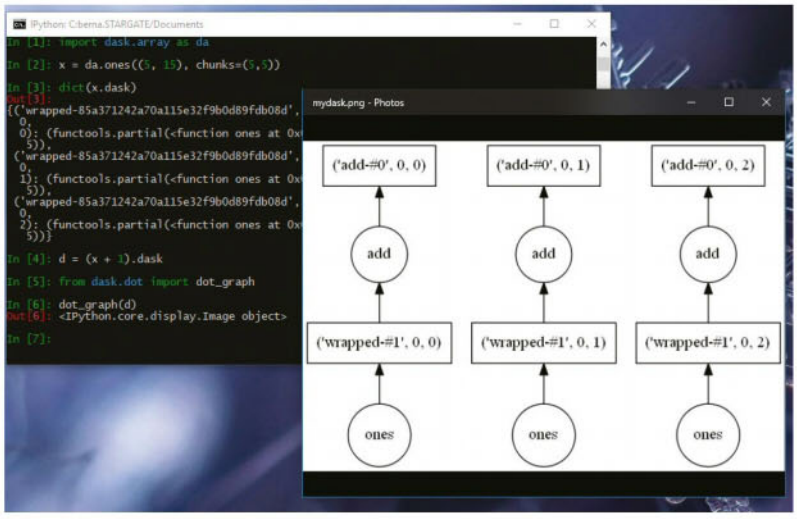
Before you can get started, you need to have Dask installed. Because there are so many parts to it, the development team has broken up the installation into several options; this way, you can install only the parts you need. As with most Python modules, you can install Dask using the `pip` command; the available options are shown below.

```
pip install dask[complete]: Install everything
pip install dask[array]: Install dask and numpy
pip install dask[bag]: Install dask and cloudpickle
pip install dask[dataframe]: Install dask, numpy, and pandas
pip install dask: Install only dask, giving you the task schedulers
```

If you want to avoid the hassle of installation yourself, you can always use Anaconda, as it includes all the installation options for Dask. If you're using a Raspberry Pi, try Berryconda instead. Then, if you do need the absolute latest and greatest feature in Dask, you can always download and install from source. We'll leave it as an exercise for the reader to find all of the requirements for compiling your own personalised version of Dask.

Set up Dask

If you're just setting up a development environment, things are pretty simple. Out of the box, Dask is already configured to be run on a single machine. This allows you to start developing and running code right away. Dask uses schedulers to manage multiple tasks and ensure they behave as intended.



When you install Dask on a single machine, the default scheduler is set up to use threads and processes on the system. This scheduler requires no additional code and is used by the new objects introduced by Dask, such as Bag and DataFrame. This means you can get better performance with your code right away. You will need to get a handle to your scheduler of choice in order to give it work to do, with code like this:

```
thread_sched = dask.threaded.get()
proc_sched = dask.multiprocessing.get()
```

Resources

Dask
http://bit.ly/lud_dask

Dask documentation
http://bit.ly/lud_daskdocs

Tutorial files available:
filesilo.co.uk

Use scheduling

Once you have some code developed, you will likely want to move to an environment where you can take advantage of even higher levels of parallelism – maybe you've built yourself a Raspberry Pi cluster, or you're working in an HPC environment at your university. In these cases, you will want to move to the distributed scheduler. You can also use this scheduler on a single machine, which means that you can do all of your development work locally before moving to a larger production system. In this case, you just need to create a new Client object with no options, as shown below.

```
from dask.distributed import Client
client = Client()
```

This sets up a local cluster of a scheduler and workers, as in **Figure 1**. When you move to multiple machines, you will need to set up this same infrastructure across your network of machines. The first step is to start a scheduler on the main machine, using `dask-scheduler`:

```
$ dask-scheduler
Scheduler at:    tcp://192.0.0.100:8786
```

This starts the scheduler process on this machine, listening for incoming connection requests from worker processes. You do this with the command `dask-worker`, giving it the IP address for the machine hosting the scheduler process.

```
$ dask-worker tcp://192.0.0.100:8786
Start worker at:   tcp://192.0.0.1:12345
Registered to:     tcp://192.0.0.100:8786
```

If you want to be able to manage the worker processes from your scheduler machine, you can use the `dask-ssh` command to start up worker processes on the remote machines over SSH. To use this command, however, you'll need to make sure that the Python module `paramiko` is installed, as it enables SSH connectivity within Python.

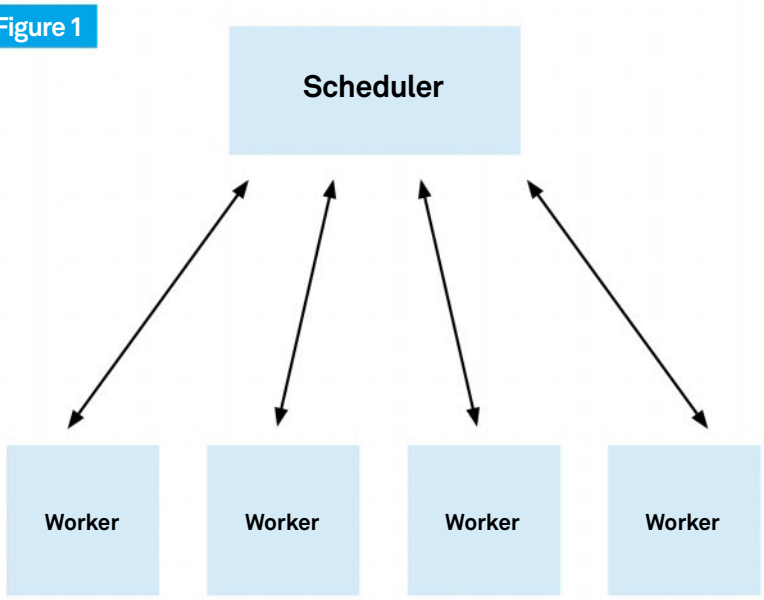
First run

Once you have a network of threads and/or processes set up, how do you use them? The most direct way is to create a new Client object that uses this network, and hand in the location of the scheduler process. In the above example, you would use `192.0.0.100:8786` as the parameter to `Client()`. You can then call `map` functions to distribute work across the cluster, and the `gather` function to pull the results back again. As an example, say you wanted to get the square of the first 100 numbers on your cluster; you'd use something like this:

```
def square(x):
    return x**2

answer = client.map(square, range(100))
client.gather(answer)
```

Figure 1



Above Dask uses a central scheduler and a set of workers that do the actual computational work

First run

Once you start introducing more machines and spreading out the work out among them, you open up a huge number of options in terms of how to configure the system to try to optimise it for your own particular needs. The first step is to find out what you have to work with. You can use the `scheduler_info()` method of the Client object to get the details of the current configuration but you might need to collect even more information. If so, you can get access to the scheduler logs by executing `get_scheduler_logs()` and the worker logs with `get_worker_logs()`. This gives you details on how the scheduling has been handled in the past.

> "Anaconda includes all of the installation options for Dask. If you are using a Raspberry Pi, try Berryconda instead"

But what if you need to know what is happening right now, in terms of how your computations are being distributed? Simple enough: use the `processing()` method to see which task is running on which worker. From this information, you may find that some workers are being overloaded with tasks to carry out. In these cases, you can use the `rebalance()` method to redistribute data to workers that are more lightly loaded. If all else fails and you find the results are now worse, you can stop everything and get back to an original state with the `restart()` method.

Debugging Dask code

One major issue is debugging Dask code. The standard debugger, pdb, doesn't know how to communicate across multiple machines. If you do need to use this, you may need to simplify your job so that it can be loaded and run on a single machine. This way, everything runs within a single process and you can use pdb to access all of the information about your Dask objects.

Using delayed

Another new object which Dask provides is called `delayed`. This new object provides a lightweight way for you to create customised algorithms that can use the underlying Dask infrastructure. A function called `delayed()` wraps other functions and objects and pauses their execution until they are farmed out to workers.

Simplify with data objects

So far, we've looked at setting up our cluster and then send work to be done explicitly out to the network. This is not the way most people use Dask, however. Many use-cases take advantage of the new data objects provided by dask in order to simplify the parallelisation of data-processing tasks. The three main objects introduced are the Array, the Bag and the DataFrame; they each provide functionality that is optimised for a particular workflow. See **Figure 2** for how they interact with the rest of the Dask library.

Arrays are modelled on NumPy arrays, and simplify the task of distributing the processing of this type of data across the network. A Dask Array is made up of a grid of smaller NumPy arrays. By default, Dask Arrays use the threaded scheduler; they can use the distributed scheduler, however, with no loss in performance. Because Dask Arrays are so closely associated with NumPy arrays, there are actually helper functions to translate back and forth between these data types.

> "Many use-cases take advantage of the new data objects provided by Dask in order to simplify the parallelisation of data-processing tasks"

A Bag is a very loose collection of Python objects that can be distributed across your network of machines. There are several functions, such as `maps`, `folds` and `filters`, which are available to apply computations to the data stored in a Bag. One major limitation to such a data structure is that calculations can't depend on the results of other calculations happening elsewhere within the Bag, because that introduces cyclic dependencies which aren't supported in such a loose collection. To help enforce this, Bag contents are immutable and can't be changed. This way, one thread can't corrupt data that may be used by other threads. Creating a Bag can be as simple as using the function `from_sequence()` and handing in a sequence of Python objects.

A DataFrame is a more structured data object modelled on Pandas DataFrames. As with arrays, a Dask DataFrame is created from a set of smaller Pandas DataFrames. The biggest advantage of this data object is the ability to manage very large data sets. This way, you can handle more data than can be loaded on a single machine. Even if your data fits on a single machine, using Dask DataFrames efficiently divides your data among multiple CPU cores so that you can process multiple chunks of data in parallel. You can create Dask DataFrames from Pandas DataFrames with the `from_pandas()` function. However, since this is a very common use-case for Dask, there are several helper functions, such as `read_csv()` and `read_hdf()`, which can read in data directly from files.

Generate graphs

All the work done by Dask is actually implemented by modelling your code as a graph and executing this graph. Dask takes each task that needs to be executed in your code and treats them as a set of nodes, and the edges connect two nodes together if one node depends on the output of a previous node. Once the graph has been generated, the job of the schedulers is to execute this graph and do the actual work. Dask actually defines a full specification on how to describe these graphs, using standard Python objects like dictionaries, tuples and functions. This means that custom schedulers can be written if you need Dask to manage your task flow in a very particular way.

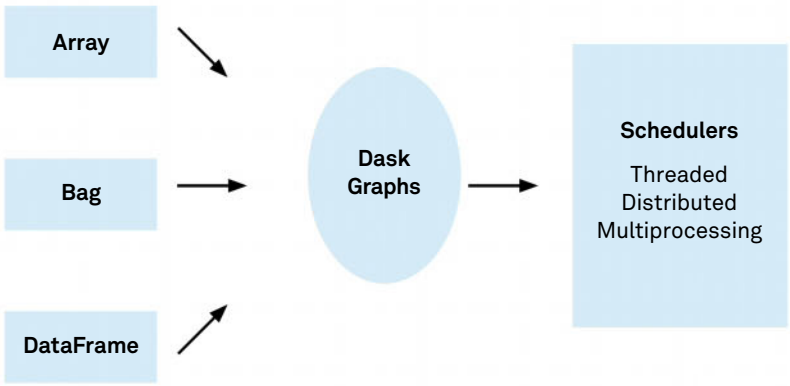
This also works in the other direction. The new data objects introduced by dask (Array, Bag and DataFrame) generate graphs based on their properties that can be handed to a scheduler. Your project may require a very particular data structure in order to map to tasks in an optimised way. This is where intimate knowledge of your particular problem can help to find a solution that the automatic processes in Dask may miss. If this is the case, you can create a new data object of your own that outputs a Dask graph that can be handed in to a scheduler to be executed.

Make further optimisation

Because of this intermediate layer between your code and the scheduler, you have an opportunity to look at and potentially optimise things before your code executes. There are a set of of functions under `dask.optimize` that provide a starting set of optimisation steps that you can use further improve performance.

For example, you could call `dask.optimize.cull()` to remove any unnecessary tasks from the Dask graph. You can even do optimisations that are common when compiling code, such as inlining. The function `dask.optimize.inline()` can inline common variables to avoid access bottlenecks, while the function `dask.optimize`.

Figure 2



Below Dask divides up the functionality into three broad layers: data objects, execution graphs and schedulers

`inline_functions()` takes expensive function calls and inlines them to the dependent tasks.

In the more general case, you can use your knowledge of the problem to provide changes to the functions being executed to try to optimise the computations being done. There are two functions, `RewriteRule()` and `RuleSet()`, which enable you to provide a set of translations from one function to a more efficient one and then apply these translations to a Dask graph before handing it over to a scheduler.

If none of these built-in functions provide enough control, you can define your own optimisation functions for various data objects. You can then use the function `dask.set_options()` to define what optimizations are to be applied for various data objects.

Look inside Dask objects

As was mentioned at the beginning of this article, parallel programs can get very messy and hard to debug – and since programs are written by human beings, there will be issues that need to be debugged.

Luckily, Dask does provide some tools to try to make this process at least somewhat easier. The first step is to be able to look into Dask objects to see what state they're in. For the new data objects provided by Dask, there is a new property available to query that provides this internal state information. Below is an example of what a Dask Array looks like.

```
import dask.array as da
x = da.ones((5, 15), chunks=(5, 5))
dict(x.dask)
{('wrapped_1', 0, 0): (ones, (5, 5)),
 ('wrapped_1', 0, 1): (ones, (5, 5)),
  ('wrapped_1', 0, 2): (ones, (5, 5))}
```

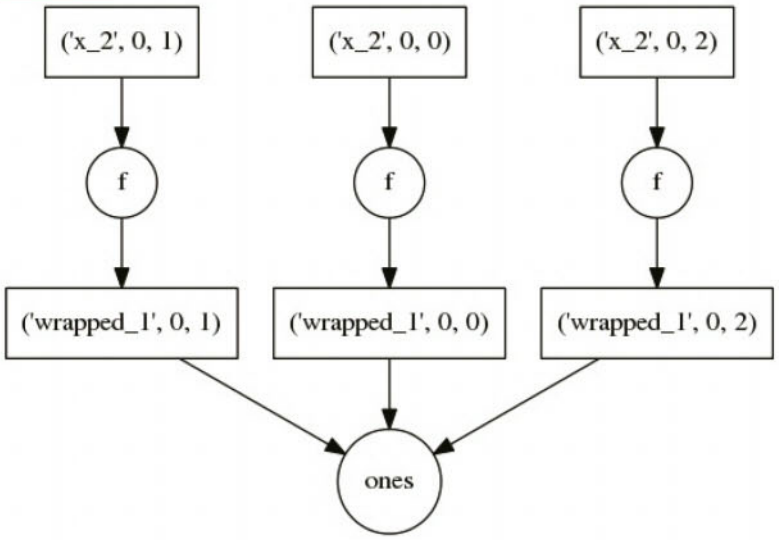
While this provides information on the nodes of the Dask graph, we still need to get data on the edges. Dask provides a function called `dot_graph()` which can generate a visualisation of a Dask graph. You need to import it from the submodule `dask.dot` before using it. Once you call it, it will generate a PDF file in the current working directory.

Using this functionality needs `graphviz` and the associated Python `graphviz` module to be installed, which may not be there by default in your environment. If it isn't there, you should be able to install it using either `conda`, or your operating system's package management software. As an example, we could start with the example above and look at the graph generated if we do some work with that array.

```
d = (x + 1).dask
from dask.dot import dot_graph
dot_graph(d)
Writing graph to mydask.pdf
```

You can see in **Figure 3** what such a graph might look like. There are also other information tools available under the submodule `dask.diagnostics` to help you figure out what is happening in your program. For example, you could display a progress bar with the following code.

Figure 3



Above The DOT functionality enables you to visualise the entire network of your Dask run, identifying the tasks (nodes) and their relationships (edges)

```
from dask.diagnostics import ProgressBar
a = da.random.normal(size=(10000, 10000),
chunks=(1000, 1000))
res = a.dot(a.T).mean(axis=0)
 with ProgressBar():
...      out = res.compute()
[########################################] |
100% Completed | 17.1 s
```

This lets you know that there is still progress happening, which is especially useful for longer runs. There are also specialised profilers that can give you usage information across the entire network. The first is the ResourceProfiler, which looks at time usage, memory usage and CPU percentage used. The second is the CacheProfiler, which looks at what is happening at the scheduler cache level. You can get key, task, size metric, cache entry time and cache exit time. With this information, hopefully you can find your problem spots before you completely lose hope.

Things to do next

Hopefully, this article has provided enough of a starting point that you might look at adding parallel computations to your own work. Dask is particularly well suited to parallelising data analysis tasks, since it has several new data objects that wrap the complexity of spreading data and its analysis across many different machines to get better throughput.

However, keep in mind that there are several other modules available to do parallel tasks, with each one tailored to a particular class of problems. If Dask doesn't fit your particular problem, you should be able to find a better fit with a quick Google search. ■

Learn statistical programming with R

Master the basics behind using the R programming language to analyse data and perform simulations

John Gowers

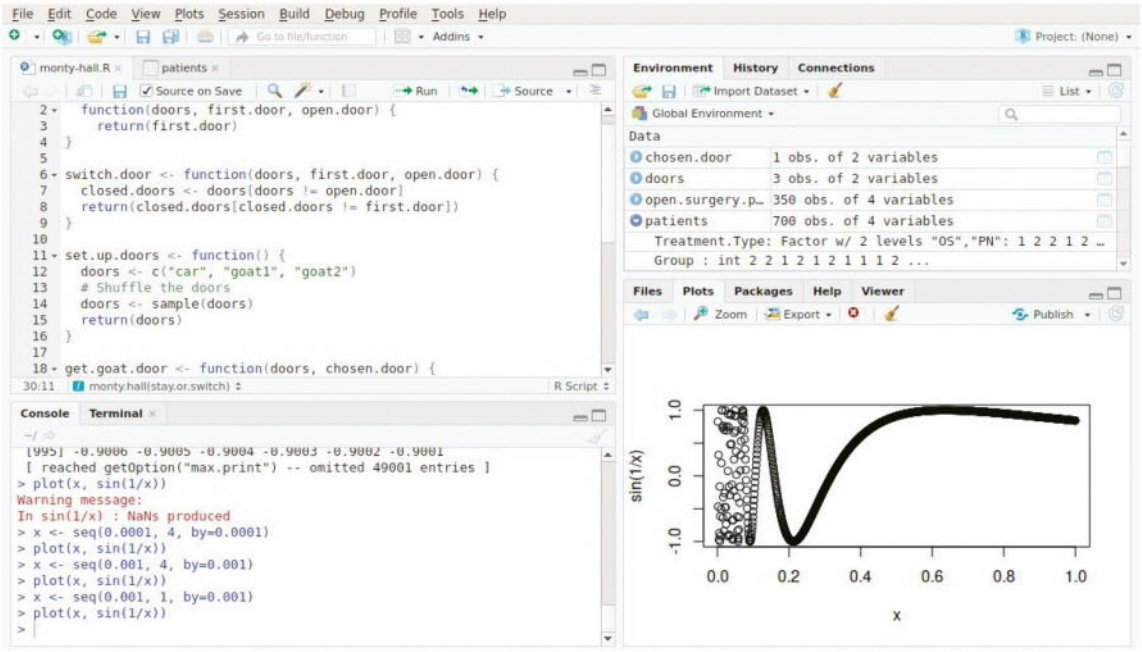
John is a university tutor in Programming and Computer Science. He likes to install Linux on every device he can get his hands on. He has used R both in an academic setting and in public-sector industry.

Resources

R
See your package manager, or download at **www.r-project.org**

RStudio
Optional; see your package manager or download from **www.rstudio.com**

Tutorial files available:
filesilo.co.uk



Above R is invaluable for analysing and solving real-world statistically based problems

Since its inception in 1993, R has evolved to become one of the most widely used programming packages in the world, and is the most popular language specifically designed for statistics. It is one of the great success stories of open source development, eclipsing its non-free predecessor, S, and building on its large developer community to build a rich library with many important external packages.

R is a matrix-based language like MATLAB or Octave, and is designed using a functional-style paradigm. The result is a highly flexible language optimised for statistical analysis.

We can't hope to cover everything that R has to offer in this article, but we'll try to give an idea of the colour of the language and the different operations it supports. We'll be covering a couple of case studies based on classic statistics topics, Simpson's paradox and the Monty Hall Problem, which will help us to develop an understanding of some of the paradigms that R supports.

Once you've understood the basics of programming in R, there are numerous online resources which you can search out in order to help with your own specific statistical problems.

Get started

The first step to working with R is to open up the REPL (read-execute-print loop) that we use for running commands in real time. If you have the RStudio IDE, we recommend that you fire it up. Otherwise, you can type

```
$ R
```

at the command line to access a more limited version.

We'll start with a classic statistics topic: Simpson's paradox. The particular example we will use is famous, and is centred around two treatments for kidney stones. The first, and more invasive, treatment is open surgery (OS), which often involves making a large incision in the patient's flank, followed by an incision into the kidney itself. The second treatment is percutaneous nephrolithotomy (PN), which involves making a much smaller incision into the patient, and passing a wire through into the kidney, allowing the stone to be removed that way. If the stone is large, it may be crushed using ultrasound probes.

We are going to be using the datasheet `kidney-stones.csv`, which is based on real data from a paper

by Chirag *et al*. discussing the effectiveness of these two treatments. The CSV file is divided into columns indicating the type of treatment performed (OS or PN), the sex of the patient, whether or not the treatment was successful and also a 'group', which indicates how big the kidney stone was. Patients whose stone was less than 2cm in diameter are placed into Group 1, while patients whose stone was at least 2cm in diameter are placed into Group 2.

The first step is to load this CSV file into R. Once we have downloaded the file somewhere on our system (`/path/to/kidney-stones.csv` in the example below), we use the built-in function `read.csv` to do this:

```
> patients <- read.csv('/path/to/kidney-stones
csv')
```

R uses the `<-` operator to assign values to variables. Here, we have created a new variable `patients` and populated it using the `read.csv` function. To get a feel for what the `patients` variable contains, we can type its name at the REPL, as in **Figure 1**. Since our CSV file has 700 entries in it, the output from this command is much too large to fit on a single screen. If you are using the RStudio IDE, you can get a friendlier look at the contents of the `patients` variable in the 'Environment' pane, as in **Figure 2**. Clicking names in the Environment pane brings them up in a table form, as in **Figure 3**.

> Matrix indexing is one of the most powerful syntactic features in R

`read.csv` takes information out of a CSV file and places it into a matrix form that R can understand. A matrix is like a 2D array of values. `read.csv` automatically interprets the first line of the CSV file as a list of headings, which we can use to get individual columns of the matrix. For example, `patients$Group` is a vector (1D array) containing all of the 'Group' values from the CSV file:

```
> patients$Group
  [1] 2 2 1 2 1 2 1 1 1 2 1 1 2 2 1
 ...
```

Since a matrix is a two-dimensional array, we can get individual cells of it using indexing notation. For example:

```
> patients[1, 4]
[1] N
Levels: N Y
```

Here, R is telling us that patient number 1's treatment was unsuccessful. The number `4` refers to column `4` of the matrix, the 'Success' column, so another way of getting the same result would be to type `patients$Success[1]`.

Figure 1

```
> patients
    Treatment.Type Group Sex Success
1               OS     2   M       N
2               PN     2   M       Y
3               PN     1   F       Y
4               OS     2   M       Y
5               PN     1   M       Y
....
```

Above R natively displays data frames using a tabular format at the REPL

In R, vector and matrix indexing starts at `1`, not `0`. Matrix indexing is one of the most powerful syntactic features in R. For example, we sometimes want to get an entire column or an entire row of the matrix. In that case, we can leave the other coordinate blank.

```
> patients[7, ]
   Treatment.Type Group Sex Success
7              PN     1   M       Y
```

This is the record for patient number 7, who is male and underwent a successful PN procedure. Leaving the second coordinate blank gives us an entire row of the matrix. Similarly, if we type `patients[, 2]`, then we get the same result that we had for `patients$Group`.

Use conditional indexing

A natural thing to do with this data is to try to work out the success rate for each type of treatment. The first step is to separate out the data by treatment. A command that we can use to get a matrix of all those patients who underwent open surgery is as follows.

```
> open.surgery.patients <-
+    patients[patients$Treatment.Type == "OS",
]
```

What's going on here? If we run the single command

```
patients$Treatment.Type == "OS"
```

then we get a long list of `TRUE` and `FALSE` values, each specifying whether the patient in question underwent open surgery. This is an example of a vectorised operation that is applied to every row of the matrix and returns a vector whose elements are the results from each row. Now we get to the indexing part. If, instead of a number, we place a vector of `TRUE` and `FALSE` values as an index to a vector or matrix, then the result is the sub-vector or sub-matrix of entries corresponding to `TRUE` values.

```
> x <- c(1, 2, 3, 4)
> x[c(TRUE, FALSE, FALSE, TRUE)]
[1] 1 4
```

Here we've used the `c` function, which constructs a vector holding some given values. The vector `x` holds

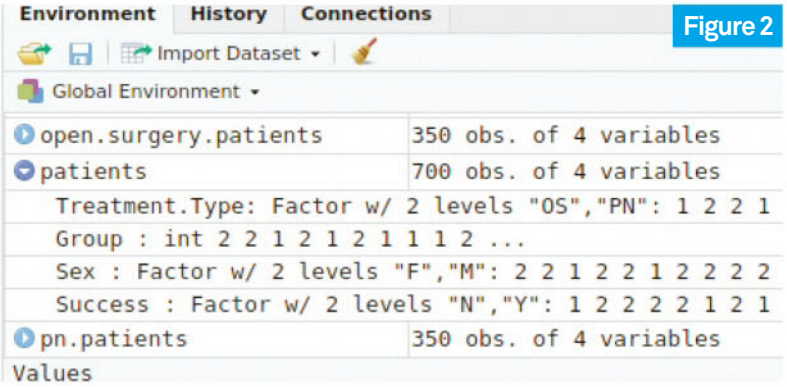


Figure 2

Above The RStudio IDE is useful for viewing complicated data structures in a manageable way

Right RStudio can display data frames as scrollable tables to enable quicker examination of data

the numbers 1 to 4, and we use the vector TRUE FALSE FALSE TRUE to pick out the first and fourth elements.

In this particular case, then, `patients$Treatment.Type == "OS"` gives us a vector of TRUE or FALSE values: true if the corresponding patient underwent open surgery, and false if they underwent PN. Then the indexing `patients[patients$Treatment.Type == "OS", ]` picks out precisely those rows corresponding to open surgery patients, returning all columns. The result is a submatrix of the original matrix, consisting of exactly those rows corresponding to open surgery patients.

As well as TRUE/FALSE lists for indexing, R accepts a list of indices. For example, typing `patients[c(1:5), ]` gives us the first five patients.

Write functions in R

Once we have a list of all the open surgery patients, we want to find out how many of them underwent the procedure successfully.

A useful command to use for this is the `which` command, which takes a TRUE/FALSE list and returns the indices of all the TRUE values.

How to use packages

People have written many packages for R, some of which are very useful. Many of them are designed for specific statistical techniques, of course, but some are useful for general-purpose programming. In order to install a package, we run `install.packages` from within R. For example, running `install.packages("plyr")` installs the plyr package, so it's available for us every time we want to use it. To load a package that we have installed, we use `library` – for example, `library("plyr")`. We need to do this separately for each session where we use the package (or include it in a script file).

If you're working on a machine where you haven't got root access, you can still install packages locally. Create a directory like `~/install/RPackages`, and then create a file called `.Renviron` in your home directory containing the line `R_LIBS=~/install/RPackages` (or whatever your directory was). Then restart R, and you can then run `install.packages`.

	Treatment.Type	Group	Sex	Success
1	OS	2	M	N
2	PN	2	M	Y
3	PN	1	F	Y
4	OS	2	M	Y
5	PN	1	M	Y
6	OS	2	F	N
7	PN	1	M	Y
8	PN	1	M	N
9	PN	1	M	Y
10	OS	2	M	N
11	OS	1	M	Y
12	PN	1	M	Y
13	OS	2	F	Y
14	OS	2	F	N
15	OS	1	F	Y
16	OS	2	M	Y
17	PN	1	F	N
18	OS	2	F	Y
19	PN	1	F	Y
20	PN	1	M	Y

Figure 3

```
> which(patients$Sex == "F")
  [1]   3   6  13  14  15  17  18  19
21  22  25  26
 ....
```

This is not very useful for indexing purposes, since we can use the TRUE/FALSE list directly, but it is a great way to count the number of entries in a matrix or vector for which a particular condition holds. For example, `length(which(patients$Sex == "F"))` returns 289, the total number of female patients in the study. This makes it easy to work out the success rate of open surgery:

```
> length(which(open.surgery.patients$Success
== "Y")) /
+       nrow(open.surgery.patients)
[1] 0.78
```

Here, the function `nrow` returns the number of rows in a matrix. There is also a function `ncol`, which gets the number of columns. An alternative to `length(which(something))` is `sum(something)`. `sum` adds up the elements of a numerical vector and returns the result; since R treats TRUE as 1 and FALSE as 0, this gives the same result as counting up all the TRUE values.

We want to do the same thing for the PN patients. Rather than repeating the same code, let's write a function that will do it for us.

```
> print.success.rate <- function (patient.
list) {
```

```
+    successes <- which(patient.list$Success
== "Y")
+    length(successes) / nrow(patient.list)
+ }
```

Let's look at the syntax here. In order to define a function, we use the special keyword **function**, followed by a list of arguments in round brackets **(...)**. The body of the function follows, in curly brackets **{ ... }**. Here, **patient.list** is a parameter, which is passed into the function. The function computes the success rate, and then prints it out.

```
> print.success.rate(open.surgery.patients)
[1] 0.78
> pn.patients <- patients[patients$Treatment.
Type == "PN", ]
> print.success.rate(pn.patients)
[1] 0.8257143
```

It appears that PN has a higher success rate than open surgery, at least for this particular sample.

Return values from functions

Let's see how things work out when we break things down by groups. It might turn out, for example, that open surgery is actually better for the patients in Group 2 who had the larger stones. We can use exactly the same indexing tools in order to break down **open.surgery.patients** and **pn.patients** by group, but we should write a function to do it for us rather than repeat code.

```
> get.group <- function(patient.list, group.
number) {
+    group <-
+      patient.list[patient.list$Group ==
group.number, ]
+    return(group)
+ }
```

R uses the keyword **return** to return values from functions, just as in languages such as C. We can now get a whole group at once:

```
> print.success.rate(get.group(open.surgery.
```

Figure 4

```
> nrow(get.group(open.surgery.patients,
1))
[1] 87
> nrow(get.group(open.surgery.patients,
2))
[1] 263
> nrow(get.group(pn.patients, 1))
[1] 270
> nrow(get.group(pn.patients, 2))
[1] 80
```

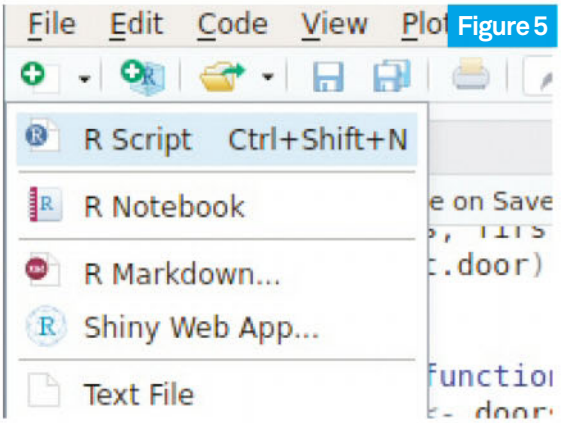


Figure 5

Above It's easy to create new R scripts from within RStudio

```
patients, 1))
[1] 0.931034
> print.success.rate(get.group(pn.patients, 1))
0.8666667
```

So we see that open surgery is actually more effective than PN for the smaller stones in group 1. We can repeat this for group 2.

```
> print.success.rate(get.group(open.surgery.
patients, 2))
[1] 0.730038
> print.success.rate(get.group(pn.patients, 2))
[1] 0.6875
```

Now we see that, in fact, open surgery was more effective than PN for both groups of patients, even though PN appeared to be more successful overall. This seemingly paradoxical situation (known as Simpson's paradox) can be explained by looking at the sizes of the different groups, as we do in **Figure 4**.

It is clear from looking at these what is actually going on: open surgery is a more invasive and slightly more expensive procedure, so it tends to be used mainly for stones of size 2cm or greater. For a given operation, open surgery will have a higher chance of success than PN, as our results have shown. However, when we look at all the patients as a group, PN appears to have a higher success rate, because it tends to be used for the 'easier' cases when the stone has diameter under 2cm.

Write R scripts

As well as the REPL, R enables us to write scripts that we can load into the REPL. Scripts typically contain function and variable definitions, although they may also contain statements that print things out. This helps us keep our code organised and saves us having to repeat ourselves.

R scripts are normal files (typically ending with the extension **.R**) that contain R code. If you are working from the command line, you'll need to create the files using your usual text exitor. If you are using RStudio, you can select 'New > R Script' as in **Figure 5**, which will allow you to edit the script directly in the IDE.

R naming patterns

You might get a bit confused when you see a name such as read.csv if you are used to languages like C or Java where '.' is used as a member access operator. In R, we use $ for member access, and '.' is the symbol used to separate words in variable or function names, just as the underscore character '_' is used in the snake_case naming convention.

Left Percutaneous nephrolithotomy was used more often for patients with smaller stones, which was why it appeared to have a higher success rate

Clear out the workspace

When you declare a function or a variable in the REPL, it's loaded into R's workspace and stays there. If you're using the RStudio IDE, this can clutter up the Environment pane, and could cause unexpected behaviour if you forget about a previous definition. You can use the `rm` function to delete the definition of a particular function or variable, which is very useful if you misspell the name the first time. The brush button you can see in **Figure 2** deletes all the definitions from the workspace.

Let's create a script to save what we've done so far. Using either of the methods we've just mentioned, create a new script called `kidney-stones.R`, and copy the definitions of the functions `print.success.rate` and `get.group` into it. You can add any other function or variable definitions that you think might be helpful. The best way to print out the definition of a function is by typing its name at the R REPL.

```
> print.success.rate
function (patient.list) {
  successes <- which(patient.list$Success
== "Y")
  length(successes) / nrow(patient.list)
}
```

We can then copy and paste this straight into our script, making sure to add `print.success.rate <-` before the definition. In order to load a script into R, we use the `source` function. So if we type

```
> source("/path/to/kidney-stones.R")
```

into an R session, we will have access to all our functions from the script. If you're working in RStudio, you have the access to automate loading the source. Ticking the box 'source on save' causes R to run the `source` command to source the script every time we save it.

The other way to save what you have done is by saving the entire R workspace. To quit R, we use the command `q()`, which automatically prompts us to save what we have done:

```
> q()
Save workspace image to ~/.RData? [y/n/c]:
```

If we type `y` and press Return, R will save all the variable and function definitions that we have entered at the REPL into a file `~/.RData`. Then, when we start R again, it will automatically reload all our previous work.

Sampling and randomness

Our next example is the famous 'Monty Hall problem', illustrated in **Figure 6**. The setup for the problem is as follows: you are on an old gameshow called Let's Make A Deal, hosted by a chap called Monty Hall (this was a real show which aired in the US). He shows you three doors, and explains that there is a new car behind one of the doors, and goats behind the other two. He asks you to choose a door and, when you do so, he opens one of the other two doors, revealing a goat. He then offers you a choice: you can either stick with your original door, or you can switch to the third door (that is, the one that is still closed and that you did not choose the first time). The question is, should you stick with your first choice, or switch to the other door?

Figure 6

Right Intuitively, it seems as if there would be no difference between sticking with the original door or switching to a new one

This question can be solved mathematically, but it is well known for being confusing: intuitively, it seems as though it should make no difference which of the two doors you choose. Instead, let's write an R script to experiment for us.

Create a new file called `monty-hall.R` to contain the functions we are about to write. We'll start off with a function that sets up the three doors in the first place:

```
set.up.doors <- function() {
  doors <- c("car", "goat1", "goat2")
    doors <- sample(doors)
    return(doors)
}
```

The only new thing here is the function `sample`. This function is used to randomly choose a sample from a given vector. For example, in the first part, if our population size had been much bigger, it might not have been feasible to check every single patient. Using `sample`, we could have chosen some suitable subset. For example, `patient.sample <- sample(patients, 1000)` would have returned a sample of size 1,000.

By default, `sample` is non-replacing – that is, it's guaranteed to never return the same element twice. If we omit the sample size argument, `sample` uses the size of the original vector. So `sample(doors)` is the same as `sample(doors, length(doors))`, which is the same as `sample(doors, 3)`. Since elements are not replaced, this has the effect of shuffling the original vector. We can test this in the REPL.

```
> set.up.doors()
[1] "goat1" "goat2" "car"
```

Figure 7

```
monty.hall <- function(pick.final.door) {
  doors <- set.up.doors()
  # Pick door 1 to start with.
  first.door <- doors[1]
  open.door <- get.goat.door(doors,
first.door)
  final.door <-
    pick.final.door(doors, first.door,
open.door)
  return(final.door)
}
```

Above The function that runs the simulation. R treats any line starting with a # as a comment

What about for loops?

R supports loops of the form `for (x in v) { ... }`, where v is a vector and x iterates over its elements. From a performance point of view, however, it is usually much better to use vectorised operations, such as apply and lapply, which can be used to apply a function to every element of a matrix or list, returning a new matrix/list of the results. For more complicated applications, we might need to use the plyr library instead.

Of course, all these operations are implemented using for loops under the bonnet, but these are mostly written in C and run with very little overhead. By contrast, when we run an R for loop, R runs all its error-checking code and other overheads – which run on every command – for every iteration. For this reason, it makes sense to use vectorised operations wherever possible.

We recommend putting in the effort as early as possible to learn how to do things in a vectorised way so that you can reap the performance benefits.

Figure 8

```
stick.with.first.door <-
   function(doors, first.door, open.door) {
        return(first.door)
}

switch.door <-
   function(doors, first.door, open.door) {
      return(doors[(doors != first.door) &
                   (doors != open.door)])
}
```

The next function we'll write is the one used by the game to choose a door with a goat behind it. There are two constraints then on the door that Monty Hall opens: it's never the first door we picked, and it must have a goat behind it. If we happened to choose the door with the car behind it, Monty has two possible choices of which door to open; otherwise, there is only one door he can choose. We use `sample` to pick a door, in case there is a choice.

```
get.goat.door <- function(doors, chosen.door)
{
   unchosen.doors <- doors[doors != chosen.
door]
      possible.goat.doors <-
         unchosen.doors[unchosen.doors !=
"car"]
      return(sample(possible.goat.doors, 1))
}
```

The main function that will run the simulation is shown in **Figure 7**. It takes in a function, `pick.final.door`, as an argument, which it will use to pick the final door at the very end. R has a very functional style, based on that of Scheme, and it is very easy to pass functions as arguments to other functions.

After setting up the doors, the function assumes that we choose door 1 to start with. There is no loss of generality in doing this, since the doors have been randomly shuffled already. It then calls our `get.goat.door` function to open one of the other doors. Lastly, it calls the function `pick.final.door` in order to choose which door to end up with, and returns whatever is behind that door.

Performing multiple trials

The last piece of the puzzle is the functions that we'll use to choose the final door. These are shown in **Figure 8**. The first one is easy: choose the door we started with. For the second one, we use vector indexing to choose the only door that is not the one we chose at the start and that has not been opened. We can now try out each of the different strategies in turn.

```
> monty.hall(stick.with.first.door)
[1] "car"
> monty.hall(switch.door)
[1] "goat2"
```

Left These are the functions that we will pass in as parameters to the main simulation. R uses & for its logical conjunction operator

Your answers may differ from these, since the initial configuration of the three doors is chosen randomly. In order to see which strategy is better, we want to test them out multiple times. To do this, we will use the `replicate` function from R, which runs a function a given number of times and then puts all the results into a vector.

You can see our results in **Figure 9**, which show that we get a car 317 out of 1,000 times when sticking with the original door, and 650 out of 1,000 times when switching doors. Your results will be different from these, but they bear out the mathematics, which says that you will get the car approximately two thirds of the time when switching doors, and one third of the time when sticking with the original door. Again, this is not something you would immediately intuit, but statistics don't lie – as R has demonstrated for us.

Figure 9

```
> first.door.trials <-
+      replicate(1000,
+                monty.hall(stick.with.
first.door))
> sum(first.door.trials == "car")
317
> switch.trials <-
+    replicate(1000, monty.hall(switch.
door))
> sum(switch.trials == "car")
650
```

Left The `replicate` function is very useful for carrying out multiple trials of a nondeterministic computation

ULTIMATE PRIVACY

QUBES OS FROM SCRATCH

Its tagline may be 'reasonably secure operating system', but **Paul O'Brien** thinks you'll find that bare-metal hypervisor-based compartmentalisation means it takes privacy *very* seriously

AT A GLANCE

Where to find what you're looking for

Qubes OS is touted by its developers as 'a reasonably secure' operating system but in reality, the distribution offers huge benefits over standard distros from a security perspective – provided you're willing to put in a little bit of effort. It's advocated as 'the best OS available today' by none other than exiled whistle-blower Edward Snowden (a quote proudly displayed on the Qubes OS home page), and this level of praise is the norm rather than the exception among privacy experts.

At the core of the Qubes OS model is the concept of compartmentalisation. The OS assumes that there is no such thing as a perfect, bug-free desktop environment; and because, without Qubes OS, a bug in any part of a distribution can lead to the takeover of a system, the solution is to limit what can be accessed if such an environment is compromised. The free, open source distro focuses on splitting all of the activities you carry out in daily computer use into securely isolated compartments known as 'qubes'. Fundamentally, the easiest way to think about each qube is as a regular virtual machine, which makes sense – the underlying technology is the bare-metal hypervisor technology 'Xen'. A Type 1 hypervisor, Xen is far more secure than Type 2 (hosted) hypervisors typically used by virtualisation tools, as they are separated at a much lower level. Not every app you launch will run in its own qube, due to the underlying concept of security domains. Out-of-the-box domains are configured for work, personal and untrusted, but you can create as many as you like – you might want domains for gaming, banking, shopping and other similar activities.

Qubes vs physical machines

Thanks to its bare-metal hypervisor, Qubes OS works a lot like separate, physical machines – so what are the pros and cons of using the OS compared to multiple devices? As you'd expect, the main pros of using Qubes OS are relative cost, physical simplicity and more straightforward (and secure) data transfer between qubes via a built-in secure inter-VM file transfer system. The cons of using qubes are that ultimately, all qubes are at risk if the hypervisor system were to be compromised (although this is very unlikely), and the fact that it prevents use of physical security techniques – for example, leaving a secure laptop locked away while you take an insecure laptop out with you.

> "Without Qubes OS, a bug in any part of a distribution can lead to the takeover of a system"

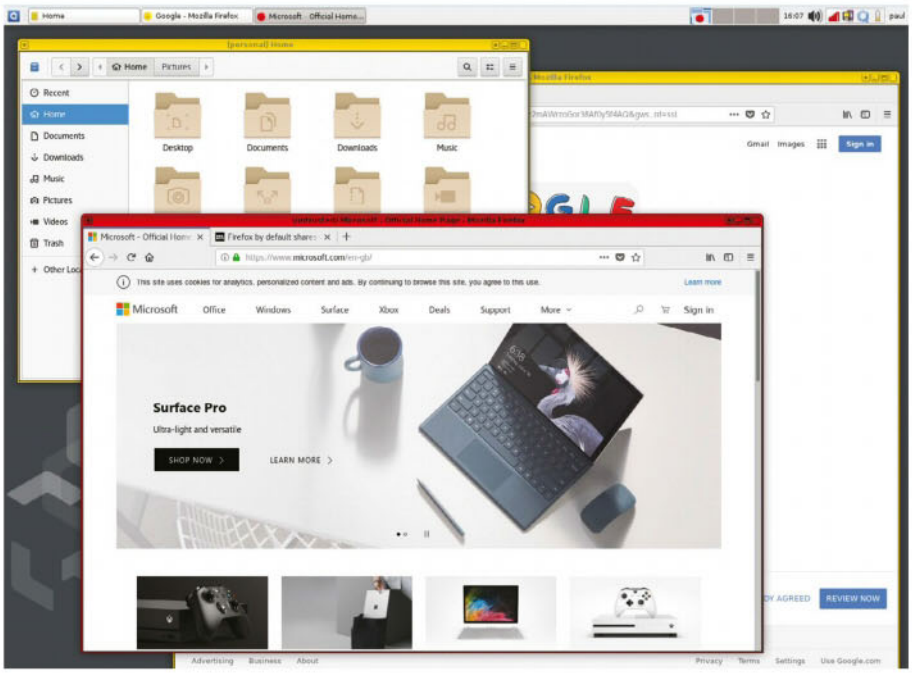
While the level of security on offer makes Qubes OS initially appealing, tools of this nature can easily become too unwieldy for everyday use because of the limitations imposed by virtualisation. Qubes OS attempts to overcome this with helpful features such as transparent full-disk encryption on installation, secure copy and paste between qubes, secure file copying, secure networking, hardware isolation and coloured window borders to indicate the configured trust level of a VM.

Qubes OS uses a template system known as TemplateVMs to simplify the software installation needed to power a qube. The default base template is based on Fedora, but the distribution's developer, ITL, also provides a Debian-based template. Community-supported templates are available including Whonix (for using Tor), Ubuntu and Arch Linux. Because it's a bare-metal hypervisor, it's fundamentally possible to run anything in this way.

Starting an installation of Qubes OS is much the same as with any other Linux distribution. The installation ISO can be downloaded from **www.qubes-os.org**, either directly from host **mirrors.kernel.org** or via a torrent. For a distribution of this nature, verifying both authenticity and integrity is vitally important and for each download a signature, digests and a PGP key are provided to you. After all, there's no point installing a secure OS if it's been compromised en route!

After the initial boot, you'll first be prompted to choose your language and set your keyboard layout. Make sure you pick the right settings at this point, because the layout is used to enter the initial disk encryption password and the keyboard settings can't be changed that early in the boot process. The default options install the base OS with Xfce4, an optional Debian 9 Stretch template (Fedora 26 is always included) and optional Whonix gateway and workstation templates.

With the default options, an install of Qubes OS requires a total of 25.42GB of space: 18.2GB for the software itself and 7.22GB for swap. Partitioning can be carried out manually or automatically; as well as installing to the main disk, installation can also be carried out to external drives/memory sticks or even SD cards. A number of credentials are entered during install: the aforementioned passphrase for full-disk encryption, a root account password (if you enable the root account, it's disabled by default) and of course the username and password for the main user account.



Above Qubes OS looks much like any other Linux distribution, which makes sense as it's based on Xfce4

> "For each download of Qubes OS, a signature, digests and a PGP key are provided to you"

First-boot options

After the initial setup completes, your machine will restart and prompt you to complete the installation as part of the first-boot sequence. This includes basic settings such as enabling the default system qubes (sys-net, sys-firewall, default DispVM) and the main application qubes (personal, work, untrusted and vault). A tickbox provides the option to use Tor via Whonix for system and template updates. USB access to the machine can be disabled completely (providing you're not using USB peripherals, of course) to limit the scope for attacks using this method.

Above Starting images from different domains (qubes) is easy as they all share the same menu

After confirming your choices on this screen, the final installation tasks will complete, you'll arrive at the login prompt

TIMELINE: QUBES OS DEVELOPMENT

APRIL 2010

An initial proof of concept ('alpha') version of Qubes OS arrives after six months of work, which is 'usable if you're determined'. An unexpectedly enthusiastic reception cements Qubes OS's future.

SEPTEMBER 2012

Invisible Things Lab (ITL) announces the first release of QubesOS, version 1.0, based on Fedora and KDE. The Warsaw-based team emphasises the 'reasonably secure' message from the outset.

SEPTEMBER 2014

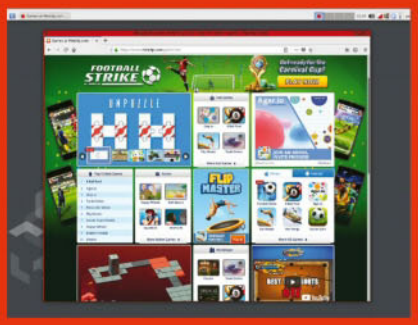
Qubes OS 2.0 is released, with support for Windows-based AppVMs and improved audio support among other things. In addition, Xfce replaces KDE as the included default desktop environment.

QUICK GUIDE

Disposable domains for further increased security

Although the different domains configured in Qubes OS allow isolation of the various parts of your digital life, there is still the risk of something undesirable infecting a domain without you being aware. Of course, with the right setup of domains there shouldn't be any risk of confidential data being leaked because of a lower security-level site, but for an additional level of protection Qubes OS offers the concept of disposable domains (aka DispVMs). As the name suggest, disposable domains have a very short lifetime; after launching, whatever you do within that domain is discarded when you close the domain. This includes any user-generated data and any apps installed within that domain – basically, any changes over and above the base TemplateVM. The disposable VM's default template can be edited as required.

A common use for a disposable domain is for hosting a single application such as a viewer, editor or web browser. It is possible to select a file within one of your regular VMs and pass it to a disposable VM for editing or viewing. Changes made to the file are passed back to the originating VM, but this means you can safely work with files from untrusted sources without risk of compromising your VM as a whole. When viewing a file in a domain's file manager, you'll notice an 'Open in Disposable VM' option on the context menu. Use this to automatically launch the viewer/editor and to close it again afterwards.



and then land in Qubes OS itself. Your initial impressions are likely to be that it all feels very normal, particularly if you are an existing Xfce4 user. Xfce4 is a more bare-bones desktop environment compared to alternatives such as Gnome and this does manifest itself in a number of ways. If you're using a HiDPI device, the scaling options in Xfce are very limited – you can increase the font size and some icon sizes, but it's not a great experience. For many users, of course, this is not a problem, but it's worth bearing in mind if you have such high-end hardware.

Assuming you selected the default options, you'll now see your qubes in the application menu. A good place to start is by firing up a few of the standard apps in the different domains to get a feel for how things work.

Now that your Qubes OS install is up and running, it's worth understanding some of the main concepts. Click the icon at the top left of the screen to open the application menu. At the top right of the screen you'll see your workspaces (as with any standard Xfce install, although you'll notice that the overviews appear with the appropriate domain border colour); the clock and the volume icon (all very much business as usual so far); and your icon tray.

> "Your initial impressions are likely to be that it all feels very normal"

The icon tray is a little bit different to a normal distro in that icons can display colours or a border which indicates the qube that icon is associated with. Out of the box you'll see a network menu (for your NetVM – note the red border), an icon to assign hardware devices to VMs, a memory usage overview and a menu for battery status, display brightness and so on. The user menu for locking, suspending and shutting down appears as normal. In the application menu on the top left you'll see the standard Xfce menu with the run program and terminal emulator options.

Both of these run inside dom0, which is the lowest level within Qubes OS. There is no network access inside here so exactly what you can do will be limited. This is actually a good thing, because you shouldn't be doing much in here anyway – wherever possible you should be doing everything within your qubes.

QUICK TIP

Read the frackin' manual

The online documentation for Qubes OS, available at **www.qubes-os.org**, is second to none. Extensive guides in text, screenshot and video form are provided for both installation and use. Detailed configuration guides are particularly valuable.

TIMELINE CONTINUED...

FEBRUARY 2015

Qubes OS is selected as a finalist for the Access Innovation Prize 2014 for Endpoint Security alongside Tails and Open Whisper Systems, ultimately finishing as a runner-up (Tails took the prize).

OCTOBER 2015

Qubes OS 3.0 arrives, introducing new Debian templates, an updated version of Xen and a new Hypervisor Abstraction Layer allowing potential use of alternative hypervisors in the future.

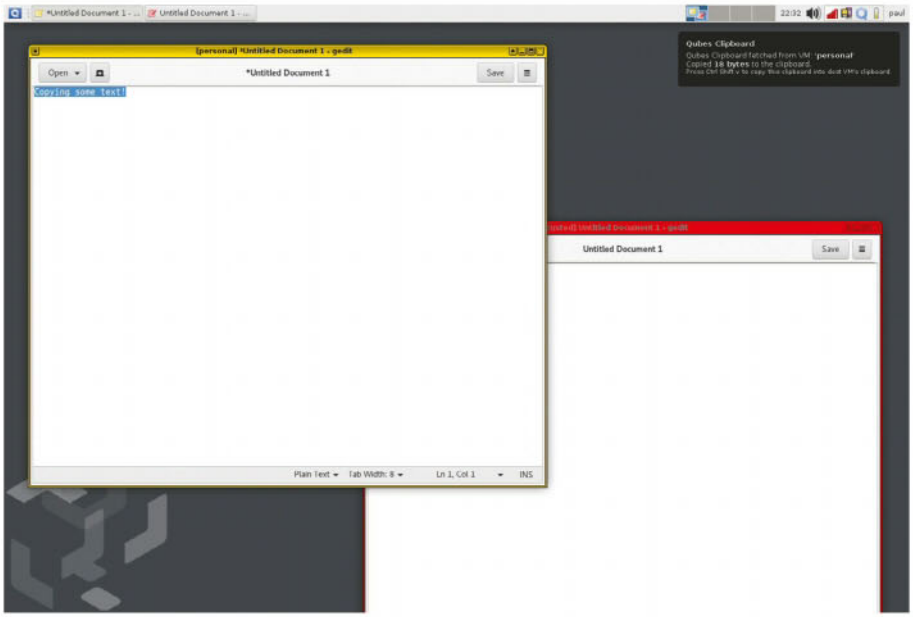
FEBRUARY 2018

Qubes 4.0 is ready for release, with a huge number of iterative updates included, a new admin API and protection from Meltdown and Spectre vulnerability attacks built into the OS.

The System Tools option in the menu configures system-wide settings – things such as appearance, keyboard config and so on, everything you'd normally see in Xfce4. You'll also notice a couple of Qubes-specific options in here: Qube Manager is the GUI option for managing your qubes, Qubes Global Settings configures the default settings used when creating or updating qubes, and the Backup Qubes and Restore Backup options provide the ability to backup and restore your qubes. (Note that the format changed significantly in version 4 of Qubes OS, which switched encryption and integrity checking to scrypt).

Qubes can also be backed up using the Qube Manager utility – or the command line, of course. Below, you'll see an option to create a Qubes VM and next in the menu come the qubes themselves, also referred to as AppVMs.

At the top of the list you'll find your disposable domains. Disposable domains are exactly as the name suggests; after launching, any changes within that domain are discarded when you close the domain. You'll see a disposable domain for the main Fedora install, as well as one for Whonix if you elected to install that option. Out of the box, installed software is minimal within domains, mostly limited to Firefox, XTerm and Qube Settings. Which apps are 'in' and 'out' of the qube can be changed in the Qube Settings app, where you can also configure a host of other options including networking, storage limits, memory limits, firewall rules, hardware device access and configured services.



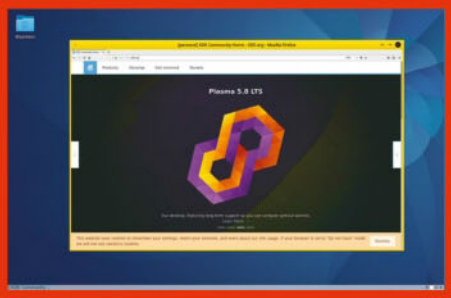
Above What makes Qubes OS unique among similar solutions is exactly how easy it is to live with on a daily basis. Secure copy and paste is a great example of functionality that smooths inter-qube use

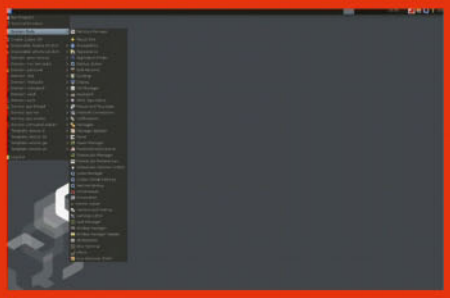
> "Limiting hardware device access only to hardware that will be used within that qube is a sensible approach"

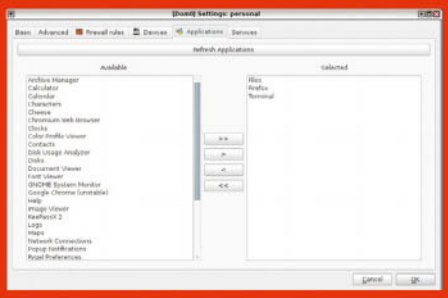
Limiting hardware device access only to hardware that will be used within that qube is a sensible approach. As you'd expect, everything is unselected by default and only needs to be selected when you specifically need the functionality (say for a webcam if you are running a video calling app within the qube). Changes to these settings can also be made later in the global Qube Manager app.

Below the disposable domains are your main domains: 'personal',

HOW TO
Tweak your Qubes OS install







1 Use the KDE environment

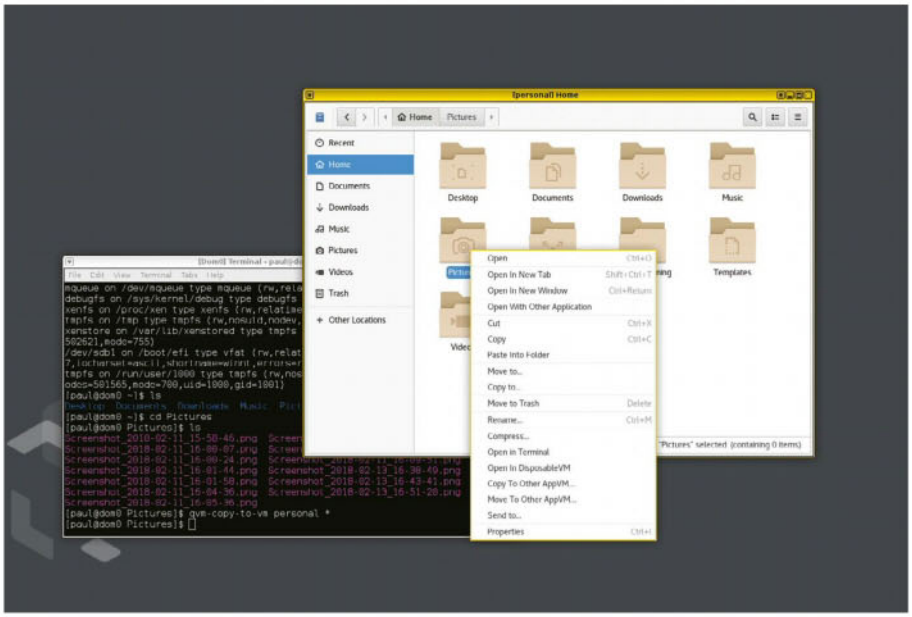
The command `sudo qubes-dom0-update @kde-desktop-qubes` switches to the KDE desktop. Change the default login manager to KDE's sddm with `sudo systemctl disable lightdm` and `sudo systemctl enable sddm`. Reboot.

2 Use dark KDE in dom0

By default, KDE uses a light theme, but you can switch to a dark one. From the main menu, open System Tools > System Settings > Workspace Appearance. In Desktop Theme select Oxygen, then apply the theme.

3 Add items to application menu

After you've added an application to your TemplateVM, you'll want it to appear in your domain app menus. To make that happen, launch Qube Settings for that domain, select the Applications tab and move items to the right column.



Above As well as copying and pasting between qubes, secure file-copying functionality is also included. This never happens unprompted, so you're always in complete control

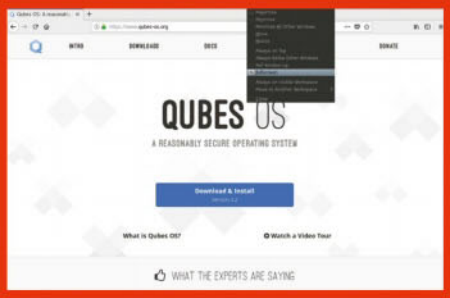
'untrusted', 'vault' and 'work' out of the box, plus optionally the Whonix domain. This is a good place to start to get a feel for what Qubes is like to use; start by launching Firefox in your 'personal' domain. You'll see the qube spin up and you'll be presented with a yellow border on the window.

Do the same in the 'untrusted' domain and you'll notice you have a red window. This is Qubes OS isolation and identification in action! After you launch a qube, you'll see a notification message at the top right of the screen indicating that it is starting. When that is complete, your window will open. Note that closing a window doesn't automatically shut down the qube. This is to ensure that if, during the session, you launch another application that resides in that qube, the startup will be quicker.

If you have specific qubes that you always use, you can set them to automatically start on boot – you can specify this in the main settings panel for the qube.

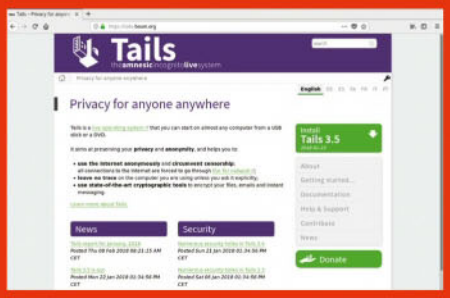
QUICK GUIDE
Manage the OS using the command line

Qubes' command-line utilities are split into two sections. The first are for dom0 (the privileged domain) and the others are for domU (a term used to refer to the unprivileged domain, which is where all domains except dom0 live). Within the dom0 set, 'qubes-' commands are provided to update dom0 or view system-wide Qubes OS settings, while the 'qvm-' set deals specifically with the virtual machines themselves: creation, cloning, removal, backup and restore, state management and more. In the domU set you'll find 'qvm-' commands to copy files to a VM, open a file in a VM or disposable VM, and run a command in a specified VM. The commands themselves are documented on device, but there is also extensive documentation online, as well as a cheat sheet created by a third party (search for qubes-cheatsheet) which is invaluable. Why would you use the command line rather than just firing up Qube Manager? In day-to-day use of Qubes OS, you might be spinning up new qubes frequently, changing configuration often (for example the networking config on your templates), and the CLI provides a super-quick way to do these things.



4 Full-screen mode in Qubes OS

Qubes deliberately restricts VMs from going full-screen, because without the window borders, you can't easily tell which domain you are in. You can force full-screen by right-clicking a title bar and selecting 'fullscreen'.



5 Use Tails from within Qubes

Tails is a live-boot distro that aims to preserve privacy and anonymity by leaving no trace on the host machine. Using Tails can be frustrating as it requires a reboot every time it's used. To mitigate this, use Tails inside Qubes in HVM mode.

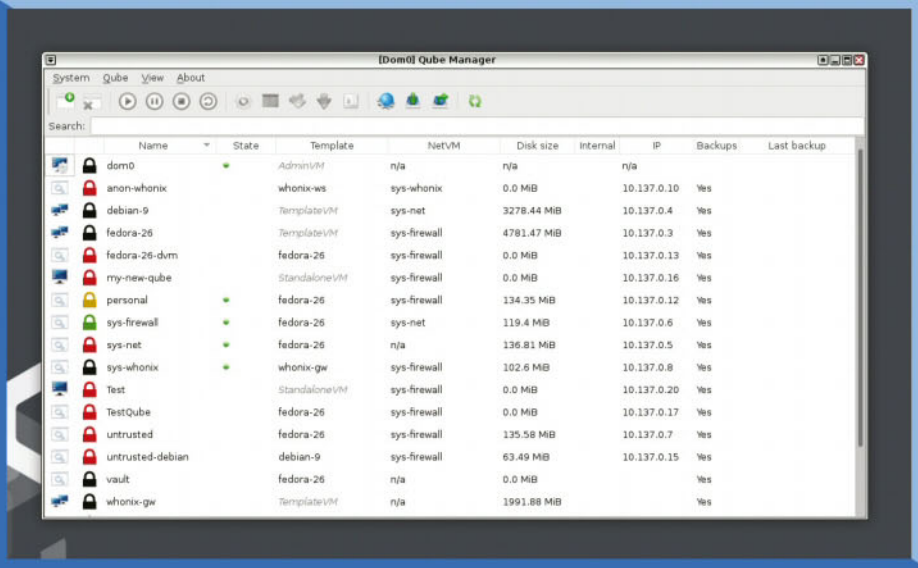
6 Use a ProxyVM for VPN

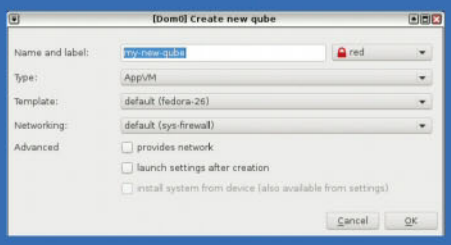
Qubes include a special type of VM called a ProxyVM. Qubes see this as a config-selectable netVM. Configure a VPN in a ProxyVM between your qube and the netVM to separate your VPN credentials from your qubes data.

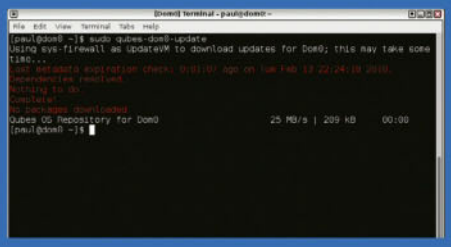
Below your own domains, you'll notice there are a couple of 'sys-' system domains, specifically 'sys-firewall' and 'sys-net'. sys-net manages your network devices and has direct internet access. sys-firewall is a trusted firewall that grants internet access to other qubes. Note that for domains granted access to the internet, it is always via sys-firewall. Domains can be specifically isolated from the internet of course, as is the case for the default 'vault' domain.

At the end of the list you'll see your templates; fedora-26 is always included and you'll see debian-9 here if you selected it during the setup, likewise for whonix-gw (gateway) and whonix-ws (workstation). Although you can specifically run applications within a template, you shouldn't – templates serve only as a basis for your own domains. The install domains are based on a TemplateVM, which means that rather than being stored on disk as an entire machine image, their root filesystem is instead derived from their parent template. As you'd imagine, this saves a huge amount of disk space, as the only storage space needed is for the user's files (the home directory).

What this does mean is that any changes to the root file system within your qube are either not made or won't be saved upon a reboot. With this in mind, it's easy to understand that if you want to install additional software, you shouldn't do it directly within your domains, but in the parent TemplateVM instead.







Top Qube Manager provides a GUI-based interface for managing your qubes. Press the Qubes tray icon to open it Above Left The Create New Qube menu enables you to select basic information about your AppVM Above Right Use sudo qubes-dom0-update from the command line to ensure that you stay up to date

Installing programs

The process for installing apps is simple. Start your TemplateVM, start either a terminal session or your software installation tool, and install the software as normal (you'll need to ensure your TemplateVM has network access if required). When finished, shut down your TemplateVM. This can be achieved by right-clicking the 'Q' menu in the tray at the top of the screen (or, using the Qubes OS CLI, issuing the qvm-shutdown command). You'll now notice in Qube Manager that all of your qubes based on this template are marked as 'outdated'. This is because their file systems have not yet been updated to match the template; in order to do that, you must restart each VM. Your qubes don't need to be restarted immediately – you can instead do it as and when it's convenient.

> "As well as support for copy and paste, Qubes makes it straightforward to move files around between qubes"

What if you want to update your apps? It's easy to fall into the trap of updating them in your qubes, but don't! As with installation, update within your TemplateVM and reboot qubes to ensure the changes are picked up by the appropriate qubes.

So, we know how to launch our own domains, add, update and remove software, but how about creating a new domain from scratch? This can be achieved by using either the Create Qubes VM option in the main menu or from within Qubes Manager. You'll be prompted for: a name and label; a window border colour; a type (typically qubes, but if desired you can also create template- or non-template-based standalone VMs, which then become completely independent of the TemplateVMs); the networking interface to be used (this would almost always be sys-firewall); whether your new qube provides network access (almost certainly not); and whether you'd like to open settings after creation (almost certainly).

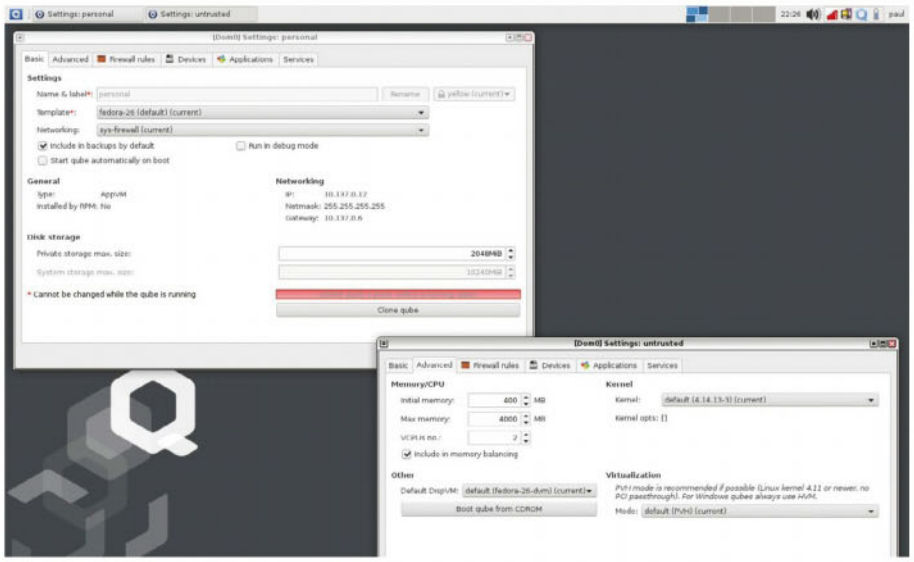
In the settings screen you'll see all the options we discussed previously. After completing this, your new qube will be displayed in the domain list and you can get started with it.

QUICK TIP

No live mode?

It's not possible to run the latest Qubes release as a live distro, but an alpha release of the now outdated Qubes 3.1 release is still downloadable and can be booted in live mode.

Although the real benefit of Qubes OS is its isolation of applications, there are going to be occasions where you'll want to share data between domains. At the most basic level, this means the ability to copy and paste, and secure copy and paste is fully supported out of the box in Qubes OS. In the source app, copy the content as normal using the appropriate hotkey (for example, **Ctrl+C**). While the app is still in focus, press **Ctrl+Shift+C**. This tells Qubes that we want to globally copy between domains. It's now simply a case of switching to the target app and pressing **Ctrl+Shift+V** followed by **Ctrl+V** to paste into that domain. Note that only plain-text copy and paste is supported. If desired, specific qubes such as 'vault' can be configured to never accept copy and paste by editing `/etc/qubes-rpc/policy/qubes.ClipboardPaste`.



Above More advanced config settings are found in the Qube Settings app, including memory and storage limits. Networking options can also be changed on the fly, for temporarily granting access

Moving files around

As well as support for copy and paste, Qubes OS makes it straightforward to move files around between qubes. The easiest way is to use the GUI; simply open the file manager in the source qube, select the appropriate file and from the right-click menu select the option 'Copy to another AppVM'. You'll then be prompted for the name of the destination. A confirmation box from dom0 appears (this is impossible to fake) and after confirming, the file copy takes place. Inbound files arrive in `/home/user/QubesIncoming/<source>`.

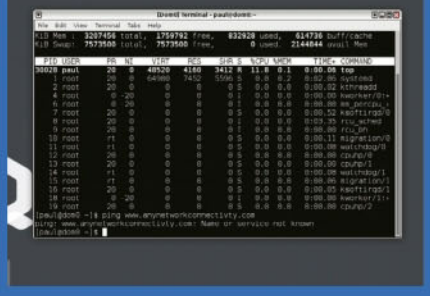
> "Qubes OS includes TemplateVMs for Fedora and Debian, with a number of additional templates based on other Linux distributions"

As with all Qubes OS operations, file copying can also be carried out from the command line. In Qubes 4, use `qvm-copy` or `qvm-move`; for earlier versions you need to use `qvm-copy-to-vm`.

Out of the box, Qubes OS includes TemplateVMs for Fedora and Debian. However there are a number of additional templates based on other Linux distributions. There are also templates available with or without certain software preinstalled. A small number of templates are available in ready-to-use binary form, but the majority of them are available only as source code, which can be built using the Qubes Builder tool. The tool is well documented on the Qubes OS website, but using it is a fairly involved process and isn't for the faint-hearted.

The most popular third-party templates are for firm favourites ArchLinux (in both binary and builder form) and Ubuntu (in only builder form). One point to note is that by using templates which are provided by third parties rather than ITL, you are trusting the distribution's maintainers. ITL also notes that third-party templates haven't undergone the same rigorous level of testing as it applies to the included templates.

Qubes OS is popular as a tool to aid in penetration testing, and the documentation includes details on how to install BlackArch, Kali and the PenTester Framework (PTF) if you want to use it for this purpose.



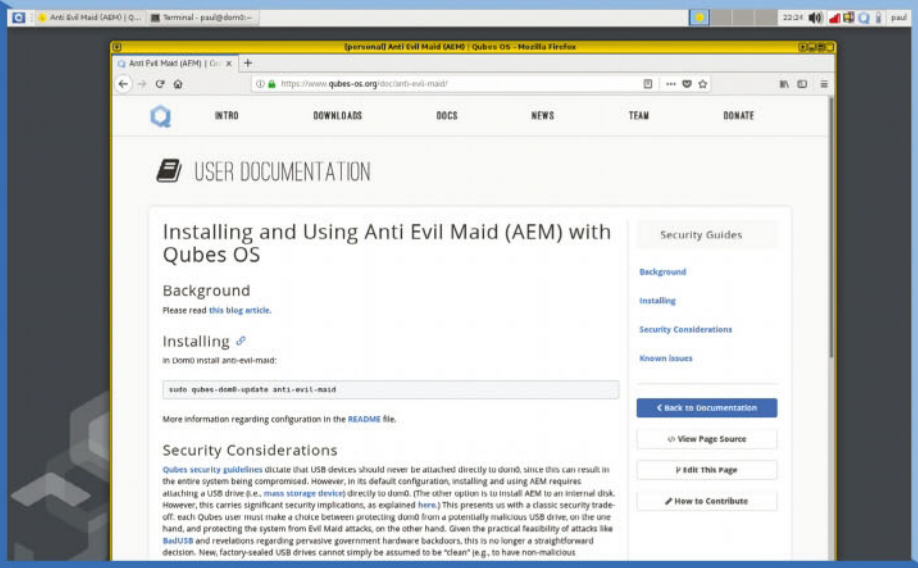
QUICK GUIDE

The core of Qubes OS and the importance of dom0

In addition to your qubes and TemplateVMs, there's an additional special domain called dom0, which is where the Window Manager and Desktop Manager run. This is also where you login to the system. dom0 is more trusted than any other domain, including TemplateVMs and black-labelled qubes. If dom0 were ever to be compromised, the entire system would effectively be compromised. Due to the importance of this not happening, dom0 has no network connectivity and is used only for these very specific purposes. You shouldn't use dom0 for anything else. In particular, you should never run user applications in dom0; run absolutely everything in your qubes, that's what they're there for!

If the idea of building or using a third-party TemplateVM feels too difficult or just like too much work, another option provided by Qubes OS is HVM (Hardware VM) domains. in contrast to PV (paravirtualized) domains, HVMs enable the creation of domains based on any OS, provided you have an installation ISO. This even allows installation of Windows-based VMs in Qubes.

To create a HVM-based Qube, select the option to create a new qube as normal in Qube Manager, but when prompted for the type, select 'Standalone qube not based on a template'. As usual you will need to specify the network (sys-firewall) and you will now have an additional option to 'Install system from device'. This launches an additional dialogue box after you hit OK, where you can specify the source of the installation ISO – either a connected physical disk or a file located in an existing qube. After selecting the appropriate source, your qube opens and your installation begins.



Above Anti Evil Maid provides important extra protection, warning of any changes to the BIOS or boot partition of your disk. Changes of this nature are likely to have been made via physical access to your PC

> "HVMs enable the creation of domains based on any OS, if you have the ISO"

If you want to install Windows in a HVM qube, be aware of a few limitations. The most recent supported version is Windows 7. Only an emulated SVGA GPU is supported, although there have been reports of working GPU passthrough. There is currently no audio support for Windows HVMs. On the plus side, Qubes Windows Tools (QWT) can be installed which provides a host of bonus features for Windows users: a seamless GUI mode that integrates app windows with the common Qubes trusted desktop; support for secure clipboard copy and paste; support for secure file exchange between the Windows VM and other AppVMs; support for `qvm-run` and generic `qrexec` for the Windows VM, providing the ability to run custom services within or from the Windows VM; and Xen PV drivers for Windows that increase performance compared to QEMU-emulated devices. Unfortunately, there are no official Qubes OS tools for other operating systems (such as other Linux distros), but some third-party attempts to get this working may be helpful.

Ensuring security

The Qubes OS website provides a number of tips for ensuring that your installation is, and remains, secure, which is vitally important because that's why you're using the OS in the first place! Aside from the aforementioned validation of the initial install package, there are some key things to remember.

The first is making yourself ultra-aware of security contexts. The coloured window borders are your guide to remaining secure. This is particularly valid for password prompts; if you see a prompt with a differently coloured border to the window behind it, simply drag it out of the window boundary to ensure it's not being faked. Another key tip is with regards to software installation vs execution. Although you will frequently be installing software into your TemplateVMs, you should never launch them within that VM. Switch to one of your qubes based on that template first.

Finally, always try and stay up to date with the latest fixes and features, by running `sudo qubes-dom0-update` to update dom0 and `sudo dnf update` to update templates or standalone VMs.

If there is a risk that somebody could gain unauthorised physical access to your computer, or if you use Qubes in dual-boot mode, you may also want to install AEM (Anti Evil Maid). AEM will inform you of any modifications which have been made to your BIOS or boot partition. If AEM *does* alert you of an attack... well, that's really bad news, because there is no true fix in this situation. If you are really serious about security, you'll just have to buy a new laptop and reinstall Qubes from a trusted ISO. ■

QUICK GUIDE

Optimal hardware for Qubes using the compatibility list

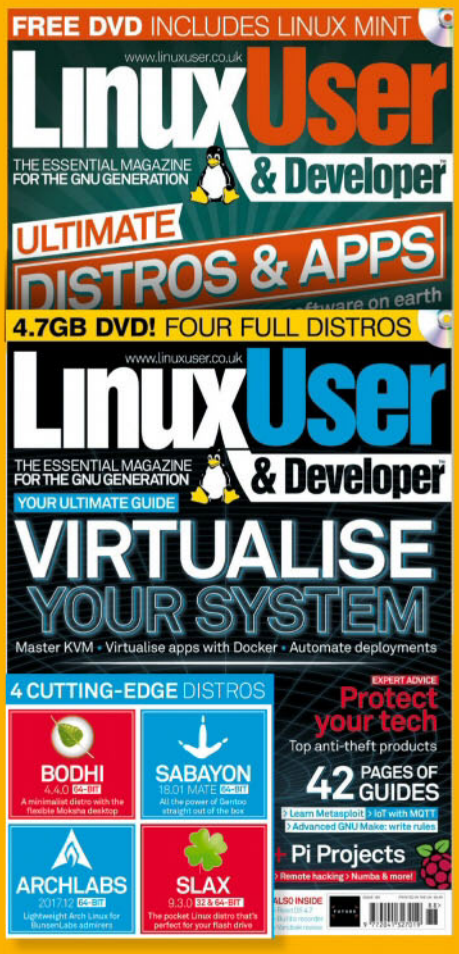
As you might expect, due to the fact that it is effectively spinning up multiple OSes, Qubes OS has fairly demanding hardware requirements. The Qubes guide mandates a 64-bit processor with Intel VT-x with EPT or AMD-V with RVI, Intel VT-d or AMD-Vi (aka AMD IOMMU), 4GB RAM and at least 32GB of disk space. A fast SSD is strongly recommended, as is an Intel IGP and a TPM with proper BIOS support. Nvidia GPUs are particularly problematic; AMD GPUs are better, but the best place to start checking if your system will support Qubes is the extensive Hardware Compatibility List, to which users can also contribute. Certified hardware is also becoming available.

SPECIAL **USA** OFFER

OFFER ENDS **APRIL 30 2018!**

SUBSCRIBE

& GET 6 ISSUES FREE*



FREE RESOURCES! FOR SUBSCRIBERS See p.96

ORDER ONLINE & SAVE

www.myfavouritemagazines.co.uk/sublud

OR CALL 0344 848 2852

* This is a US subscription offer. '6 issues free' refers to the USA newsstand price of $16.99 for 13 issues being $220.87, compared to $112.23 for a subscription. You will receive 13 issues in a year. You can write us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14 day cancellation period unless exceptional circumstances apply. Your statutory rights are not affected. Prices correct at point of print and subject to change. Full details of the Direct Debit guarantee are available upon request. UK calls will cost the same as other standard fixed line numbers (starting 01 or 02) are included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit: **bit.ly/magtandc Offer ends April 30 2018**

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

PRACTICAL Raspberry Pi

72 "I hit a lot of roadblocks putting this together"

Contents

Rick Perotti

Rick is a demo architect at Oracle Corporation, currently working on IoT and Chatbot projects. Outside of work he loves using Raspberry Pis and Arduinos.

Like it?

You can see Rick's mood board for the project on his Pinterest page (**www.pinterest.co.uk/r570sv/switches-and-dials**). For more of Rick's tinkering follow him on Instagram (**www.instagram.com/rick.perotti**).

Further reading

Rick's project adventures are recorded at **www.instructables.com/member/r570sv** and include a mechanical fortune teller that uses the open source Jasper for always-on, voice-controlled apps (https://jasperproject.github.io) and a Pi.

Retro rocket ship panel

A Pi- and Arduino-powered panel that mashes up everything from old kids' TV shows to sci-fi films

As a kid, Rick Perotti was fascinated with switches and dials on control panels of all kinds – on dashboards, spaceships and aircraft cockpits. As an adult, that led him to build a nostalgically fuelled rocket ship control panel. Little did he know how hard it was going to be. Among many other things, he had to learn laser-cutting, but the end result is a cornucopia of dials and switches worthy of any child's imagination.

This year-long project has been quite a learning experience for you. What's been the biggest challenge?
Figuring out how to make the individual panels. I tried quite a few different techniques and finally settled on using a laser cutter and CorelDRAW. I didn't own either when I started the project. Solving this problem took up the bulk of the time for the project.

> "I worked on this project part-time for 14 months"

You weren't impressed with the free software that came with your laser cutter and ended up using Corel Draw. Is there a software gap here that needs filling?
The cutter is typically called the 'eBay laser cutter' or the '40W laser cutter'. You can find it on eBay, Amazon and AliExpress to name a few places. I had thought I'd done my homework checking an endless number of videos on YouTube about this tool, and decided to buy it. The free software was useless for any serious work... [But] the printer does support a CorelDRAW Add-On that lets you drive it and enables you to engrave or cut.

There is an alternative and that is to add additional boards so the cutter can accept G-code. It becomes a two-axis CNC laser cutter; I believe some of the kits are Arduino-powered. I decided to go the CorelDRAW route because Corel offers its software as a service – I'm renting it and pay an annual licence, the least expensive route. The G-code conversion costs at least as much as the original printer and then I still need to find/purchase/learn some CAD software that can create the drawings and generate G-code. I decided to go the least/cheapest route so I could continue to make progress on the project.

Tell us about the maker event you're involved in.
Oracle puts on its own annual Maker Faire and this was its second year. I built this project specifically for the event. My last year's project was Homunculus, the Mechanical Mystical Oracle – Oracle Magazine made a video about it (**http://bit.ly/RicksHomunculus**).

What would you do differently after your long journey with this project?
I worked on this project part-time for 14 months in total. Know ahead of time that not everything is going to work out the way you expect it to! Try not to get discouraged or give up. I hit a lot of roadblocks during the year putting this together that I beat my head on the table trying to figure out. I learned that I need to stop banging my head and work on another part of the project for a while. Getting away from a problem and coming back some time later can give you a new outlook on things.

Now you're a master at speed soldering, do you have any tips for our readers?
Yes – I use a variable-power soldering iron with the smallest tip I could find. You will never be able to solder anything to a Pi or Arduino with the typical bulky off-the-shelf soldering irons that DIY stores sell. You will either melt or destroy what you are trying to work with.

The scale of this project is incredible. Have you counted the number of lights, dials and switches you've included? How many movie references are there?
No, I never counted them – I'm just a happy camper when I plug it in and they all come on! Here are all the references movies used in the project [see if you can guess them before reading]: Star Wars, The Empire Strikes Back, The Fifth Element, The Jack Benny Show (the "creepy smoking guy"), Moon, Alien, Aliens, Batman, Lost in Space, The Time Tunnel, The Adventures of Buckaroo Banzai Across the 8th Dimension, Dr Strangelove, Dark Star, Star Trek and Marvel's Agents of S.H.I.E.L.D.

What are you thinking about for your next project, or are you thinking of adding some more to this one?
I've already started! What I didn't include on the project were any dials, such as pressure, temperature and so on. I knew I could make them, but I didn't want to take the time to perfect *how* to make them.

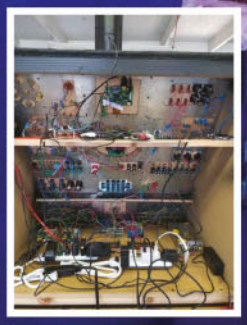
I'm starting to work on a Pi-driven digital instrument console for a sandrail that we own [a lightweight off-road buggy]. I'm also going to see how far I can take the Pi Game API to build something Tony Stark would use. My current Pi projects are a Tony Stark/Iron Man-inspired automotive gauge cluster using Raspberry Pi's Pi Game API; making a 2.0 version of Homunculus; and some Pi and Arduino-powered games that can fit into old cigar boxes.

Perfecting panels

After some experimentation, Rick settled on using 3/16-inch MDF for the panels. He first drew the panel in CorelDRAW, making sure beforehand he'd identified the dials, switches and so on that he wanted to use. Each drawing had two layers: one for the cutting of the panel and one for the panel print-out, for which he used photo paper covered with a clear protective sheet. Making sure all the holes were aligned, he used 3M SprayMount to glue the print-outs to the laser-cut panels, cut out the holes, and used a leather hole-puncher for the four corner-mounting holes.

Dials, lots of dials

All the parts came from AliExpress, Amazon and eBay. It's not always easy to buy what you need, Rick says; you'll need to know the difference between SPDT and DPDT (Single-Pole and Double-Pole, Double-Throw) switches and momentary vs latch. A latch toggles on and off; momentary are only on while pressed. "Lights will be in the same format, and be for 12V, 24V, 120V or 220V, so read carefully."



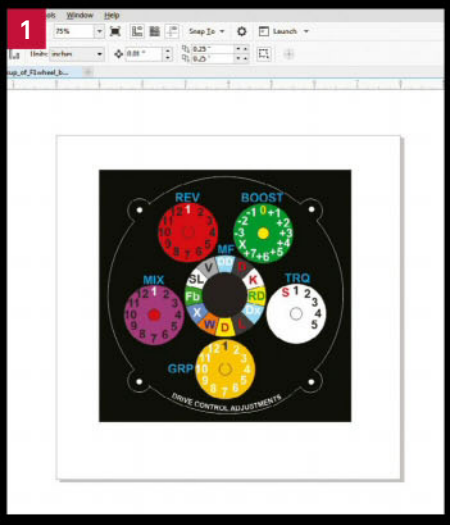
Light show

To power the panels at the bottom, Rick used an Arduino driving a four-relay board. He added marbles with plastic tubing lit by conventional 3.3V Christmas lights. To light them all, he had to use use two 3.3V power supplies: "The lights were wired up in parallel, in a random order, so that when each relay is pulled a random pattern of lights is lit."

Wiring web

Rick describes his wiring as a "get-'er-done approach", as the plan was always to display the completed panel at Oracle's Maker Faire. "I had to keep adding 12V power supplies to the project because there were so many incandescent lights!"

1



Drawing it out

Rick used a 40W CO2 laser cutter and designed his panels using CorelDRAW as he found the free software supplied with the cutter "useless". Fortunately, the cutter supported a CorelDRAW Add-On. "I had never used this program before, but I was kind of stuck," says Rick, so he bought a year's licence for it.

2

Switch to visual display

A monitor on the panel displays films. It's a 9-inch TFT LCD Display Module HDMI+VGA+2AV Driver Board and can be used with any Pi model. Rick used a Python script that runs and kills the command-line OMXPlayer when a knob is turned. It uses 13 GPIO lines for rotary switches on the panel that catch the knob's movements.

Create a Raspberry Pi network with PiServer Tool

Use PiServer to easily set up a network of Pis connected to a central server, which you control

Nate Drake

Nate is a technology journalist specialising in cybersecurity and Raspberry Pi.

Resources

- **PiServer** www.raspberrypi.org/blog/piserver

The good people of the Raspberry Pi Foundation have outdone themselves once again with the release of PiServer, a tool which enables you to easily set up PXE (Pixie) network booting. In plain English, this means you can set up a network of Raspberry Pis, each of which are connected to a single server. The Pis can boot over the network without any need for microSD cards and you, as the server admin, can control user accounts and accessible files.

The fact that every Pi, as a network 'client', uses the same accessible files makes PiServer perfect for classroom environments or workplaces where you may want to use Raspberry Pis for specific purposes such as teaching or operating machinery.

You can create as many users as you need during setup. These users don't have root privileges, meaning they cannot install new software; this makes for a much better and safer learning environment.

In order to get started setting up your network, you'll need at least one Raspberry Pi 3 and an Ethernet cable to connect each one to the router or network hub. The PiServer tool is currently available via Debian with Raspberry Pi Desktop. Follow the steps in the guide below to install it on a dedicated PC for best results, although you can also set up your server using a virtual machine or even a Raspberry Pi (see below).

01 Enable network boot

Each Raspberry Pi you want to use with PiServer must have network boot enabled. To do this you'll need a microSD card with Raspbian or Raspbian Lite preinstalled. To enable network boot, just add the line `program_usb_boot_mode=1` to the file `config.txt` in `/boot`. You can do this manually on your computer using a card reader, or by opening a terminal on the Pi itself and running the command `echo program_usb_boot_mode=1 | sudo tee -a /boot/config.txt`. Power off the Pi and remove the microSD once this is done.

02 Set up your server

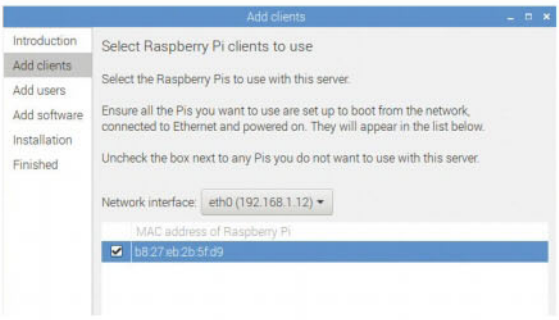
In order to set up a server for your Pis, you'll need either a dedicated x86 computer or a virtual machine onto which you can install Debian with Raspberry Pi Desktop, which includes the PiServer tool. To get started, visit **www.raspberrypi.org/downloads/raspberry-pi-desktop** and download the ISO file. While you can run the OS in Live mode, we recommend you opt for a 'Graphical Install' to the hard disk. Make sure you have at least 16GB of free space on the drive to contain the Pi's file system and user data.

03 Customise the server display

By default, Debian with Raspberry Pi Desktop uses a restrictive 640x480 resolution. To fix this, restart the machine and hold C while it boots to enter the GRUB prompt. Enter the command `videoinfo` and note which resolutions are supported by your graphics card. Boot to the Debian desktop and open the Terminal. Run `sudo nano /etc/default/grub`. Remove the # at the start of the line `#GRUB_GFXMODE` and change it to your desired resolution, for example 1024x768. Press **Ctrl+X**, followed by **Y** and then **Enter** to save and exit. Next, run `update-grub` and restart to apply your new resolution.

04 Register Pis with the server

Connect your Raspberry Pi(s) to your network router or hub via Ethernet and power it on. Return to the Debian desktop and go to Preferences > PiServer. Read through the introduction, then click 'Next'. From this screen you can select the client(s) to add; if you don't see all of your Pis, check that the listed 'network interface'



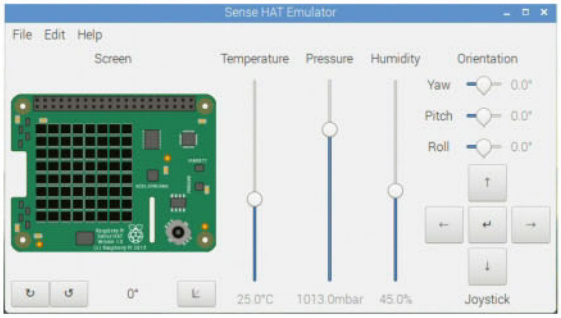
is correct. Failing this, repeat the first step to enable network boot. If it is enabled, the LEDs both on the Pi and connector should be active. Click 'Next' to continue.

05 Create user accounts

You must create at least one user account so that people using your Pis can log in. Remember that this account is created on the server itself and will use the files stored there. This means, for instance, that if you create the user 'Alice' with the password 'password123', she can log in using these credentials on any of the Pi's connected to your server and will see the same files. You can add and remove user accounts after setup if you want. Fill in the fields, then click 'Next' to continue.

06 Install the operating system

Use the Add Software screen to choose which OS to install. At the time of writing only Raspbian and Raspbian Lite are available, although the Raspberry Team is confident other operating systems will be supported in future. PiServer also supports installing an OS via local files at a URL, although it's not clear if any other Pi-compatible systems currently work with it. If you're handy at programming, head to `/var/lib/piserver/scripts` to see how these operating system images are created. Wait for the software to install before clicking 'Next'.

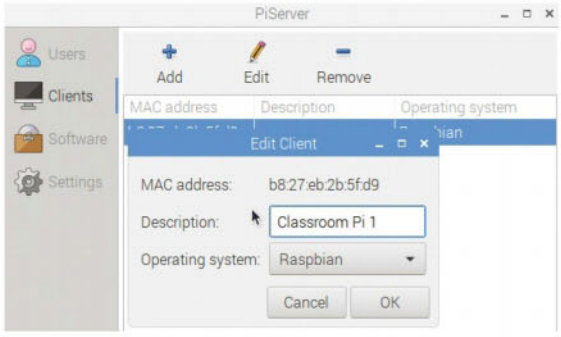


07 Start your Pis

Once installation is complete, restart each Pi connected to your network and make sure that the mouse, monitor and keyboard is connected to each one. Remember that the keyboard must have the same layout as the one connected to your server. Users will need to log in using the credentials you created earlier. They can use preinstalled software, change the background, mount drives and connect to the internet. Root access is disabled, however, so they cannot make system-wide changes such as installing software.

08 Edit users and clients

The PiServer tool enables you to add new users and/or Pis to your network as necessary. To edit user accounts, simply click the Users tab on the left side and click either Add or Remove. As the server admin you can also reset the password for users from here. Use the Clients tab to add more Pis to your network; the layout here is the same as in **Step 4**. Use the Edit button to give each Pi a description, for example `Classroom Pi 1`.



09 Install software

As the Pi users don't have root privileges, they can't update the system nor install new programs. Click the Software tab in PiServer to remedy this. From here you can choose to remove or add a new OS altogether. Alternatively click Shell to be taken to the command prompt, where as the root user you can perform operations such as downloading new programs. Use the shell to run `apt-get update` and `apt-get upgrade` so that users have the most recent version of Raspbian.

10 Change DHCP server settings

In most cases your server is connected to a router, so you'll want it to act as a proxy DHCP server. In other words, if you have a router which assigns IP addresses to devices on the network, your Debian server will not interfere with this in any way while interacting with the Raspberry Pis. However, you can select 'Act as a Standalone DHCP Server' if you want. Check that your desired range of IPs, netmask and gateway addresses are correct and click Save. ■

No server? No problem!

If you don't have a spare computer to act as your Pi Server, you can set up Debian with Raspberry Pi Desktop in a virtual machine using VirtualBox or similar. If you choose to do this, make sure that your virtual hard disk is large enough to contain the filesystem and users (with at least 16GB free). Your VM also needs to connect directly to the Pis, so ensure you enable bridged networking too. In VirtualBox you can do this by clicking Settings on your virtual machine, then Network. In the drop-down menu labelled 'Attached to:', choose 'Bridged Adapter'. You can also use a Raspberry Pi as the server, but file performance will obviously be much slower than using a dedicated server or VM.

Arsenijs Picugins

Arsenijs is a maker that uses Raspberry Pi extensively, along with Python, and currently works on ZeroPhone, a Pi-based phone.

Resources

- **pyLCI Source** https://github.com/CRImier/pyLCI
- **pyLCI documentation** https://pylci.rtfd.io

Tutorial files available: **filesilo.co.uk**

Control your Pi with pyLCI

Stop using all those weird console commands – this is an easier way to control your Pi

pyLCI is a simple hardware interface you can use on your Pi to control and configure it without the usual requirement of having a HDMI monitor, Ethernet connection or a connecting a USB-UART dongle.

Mainly, it enables you quickly configure the most important settings, such as connecting the Pi to a wired or wireless network and getting its IP address. This makes your Pi accessible – whether you are in the comfort of your home, somewhere outdoors, or maybe even working with your Pi on a long bus ride! All you need is a simple shield with a character display (16x2, for instance) and some buttons; there are many suitable shields which you can get, some of them as cheap as £5.

pyLCI stands for 'Python-based Linux Control Interface' and it's a user-friendly interface, too – you don't need to remember commands or spend time connecting things, and even people that are not technically literate can use it. In general, it saves you a lot of time when setting up another way to access your Pi (SSH, say), or monitoring it. It does not replace the command-line completely; rather, it helps you get through the initial hassle of needing to know the IP address which could otherwise waste a lot of your time, and it allows you to make your Pi projects considerably more user-friendly.

Where pyLCI is useful

There are places where pyLCI can be particularly useful – in educational workshops, for example. When you have multiple Pis on a table and students connect to them through SSH, you might need the IP address for "the Pi to the left of that guy"; with pyLCI, it takes five seconds.

Likewise, it helps on hackathons. If, at the beginning, you have to spend time searching for a HDMI-capable monitor or a cable, you'll have less time (and energy) to actually work on your idea. With pyLCI, you can concentrate on what's important, and you can even make the inevitable debugging easier by writing a simple pyLCI plugin for monitoring your software.

Also, it helps you transport your Pi projects safely. Say you want to bring your latest Pi-based project to work and show it to your co-workers – there's a chance that it'll stop working once you've transported it, and the more parts are involved in your project, the bigger that chance is. Whether the culprit is the lack of an internet connection, a loose wire, a hard-coded IP address or just a command failing to run on startup, you can easily connect to your Pi and figure out what the problem is.

One more kind of project with which pyLCI is helpful is Pi-based home servers. Whether you're running PiHole,

Above With a couple of button presses, you can monitor your system parameters with pyLCI

a small webserver or using the Pi as a router, having a tool that lets you take a quick peek under the bonnet is useful, and it's quicker since you don't have to use SSH or access a web interface. You can check whether your Pi is actually running and the OS hasn't crashed, restart services, unmount mounted drives and run scripts – and you can also run shell commands from the interface (entering them is fairly slow, but there's a command history so you don't have to repeat yourself).

Most popular 'character display and buttons' Pi shields are supported out-of-the-box. For example, Adafruit's 16x2 Character LCD + Keypad for Raspberry Pi, which has a cool character display, with RGB backlight; you can get it from Adafruit (**www.adafruit.com**) or one of its distributors. There's also Piface Control and Display 2 with three more buttons and an IR receiver, available from Element14 (**www.element14.com**). There are also the no-name LCD RGB KEYPAD ForRPi shields, which are clones of the Adafruit shield with minor hardware changes (and a separate RGB LED instead of RGB LCD backlight), available on eBay and similar sites.

If you've already got a different shield which does have an LCD and buttons, it's likely that it can work with pyLCI; you'll just need to enter the GPIOs used by the display and the screen into a config file (see `OTHER_SHIELDS.md` in the pyLCI source for instructions)

> "Most popular 'character display and buttons' Pi shields are supported out-of-the-box by pyCLI"

As for the software, you can install pyLCI on any Pi that runs Raspbian Jessie or later. Check whether it's compatible by running:

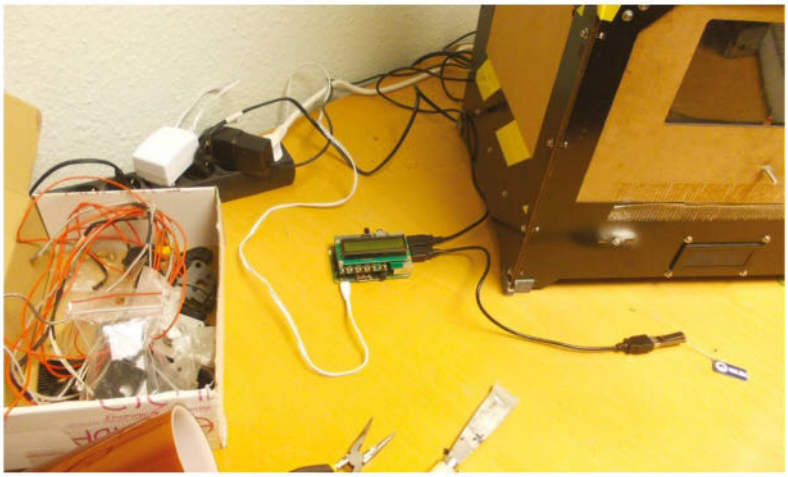
```
cat /etc/os-release
```

If your `VERSION_ID` is 8 or more, you can install pyLCI; if not, a system upgrade might be in order!

Installation and usage

To install pyLCI, download it from GitHub, install its dependencies, run the configuration script and pick your shield – pyLCI will configure itself for that shield. Then run pyLCI manually to make sure your hardware works. Once you've confirmed that it does, sync your pyLCI code to the global pyLCI installation (which runs on system startup). The sync mechanism is there so that your local changes won't break pyLCI until you sync. Here's some code to do all these things:

```
apt-get install git
git clone https://github.com/CRImier/pyLCI
./setup.sh
./config.sh
```



```
sudo python main.py
./update.sh
```

Above A Pi using Octoprint - pyLCI helps with webcam and flash drive operations

Your screen should light up, and "Welcome to pyLCI" should appear on it. Right after, you'll see a menu: use the Up and Down buttons to navigate, and Enter to go into submenus. To go back, use the Left key. The Right key will occasionally be used for context menus or character input. PiFace Control and Display has the Left and Right as two leftmost keys in the bottom row, with the other keys unused.

Capabilities

You can easily monitor your uptime and CPU load from pyLCI; go to System > System info > Uptime&load. It's handy for monitoring your Raspberry Pi creations, when you can glance at the display and see if your Pi is okay and hasn't rebooted when you didn't expect it to. It is even more useful if you end up compiling something on a Pi, or, say, streaming video from it – you can check the CPU load and see if the compiling process that you've launched has finished, which is very convenient because you no longer have to check the terminal all the time.

Additionally, if your HAT has a RGB screen backlight (like the Adafruit HAT) or an RGB LED of some sorts, the CPU monitor will make use of it, lighting it red when CPU load is nearing 100%, green when it's below 30%, and blue/violet for values in between – making for an even more convenient monitoring tool!

Most importantly, you can connect to Wi-Fi using pyLCI, by going to Networking > Wireless. If a wireless card is found, you'll see a menu of possible actions. Pick Scan to make the Wi-Fi card re-scan the networks; wait for five seconds, then go to Networks to see the networks that are available. Pick the network you want to connect to; if it has a password set, it will request you to enter the password. You can do this using the arrow keys: use Up or Down to change the character you're on, and Left or Right to change the character itself.

Entering a password this way is tiring, but thankfully, you shouldn't have to do it twice! To make it even quicker, move up for letters (both lowercase and uppercase)

Catch-22 banished

pyLCI solves the chicken-and-egg problem of connecting a Pi to a Wi-Fi network, when you need to use SSH to enter the Wi-Fi password, but you can't use SSH until you're connected to a network. Using just the screen and five buttons, you can pick a network and enter its password, then see the IP address that you've received from the router.

pyLCI can save you

If you ever were in a situation where you couldn't connect to your Raspberry Pi, it's likely pyLCI could have solved that problem. You do need to install it beforehand, but you only need to do it once - after installing, it will be there for you, waiting to save your day...

and down for numbers and special characters. If you mess it up and enter too many characters, you can also clear your last character using backspace (move down once to see it). You can also check your IP address. Go to Networking > Interfaces and select your interface (for example, `eth0`). Scroll down a little bit to see the IP address (if you received one, that is).

You can also run scripts and commands from pyLCI. This enables you to automate parts of your life; for example, you might have a flash drive with important information that you don't want to lose, and you want to regularly back it up on some kind of device. You can write a script that detects your device, mounts it and uses `rsync` to synchronize the flash drive's contents. Then, when you come home to your Pi, go to Scripts > Script by path, find your script on the Pi and click Enter on it – it'll run your script (and you can even give it command-line arguments before running).

> "In more complicated apps, you can use RPC so that your app code isn't contained in pyLCI"

An example app

All the functionality in pyLCI is provided by Python 'apps'. It's easy to write one and add it to your pyLCI install. Here's a pyLCI 'Hello world':

```
# Caption: main.py
menu_name = "Hello world"
from ui import Printer
i = None # An input device object
o = None # An output device object
def callback(): # This will be called when the app is selected
    Printer("Hello, world!", i, o, 5)
```

This code prints "Hello, world!" on screen for five seconds once the "Hello world" app is selected in the main menu Simply saving this file somewhere is not enough – you have to store it inside pyLCI. Go to the directory where you downloaded pyLCI into (most likely `/home/pi/pyLCI`). All apps are stored in the **apps** folder; create a new folder inside **apps** and name it something like **hello_world**, then place the `main.py` file inside that folder. Also create a new empty file in that folder, named `__init__.py`. Here are the commands you can use to do this, assuming you downloaded pyLCI into `/home/pi/pyLCI`:

```
cd /home/pi/pyLCI
mkdir apps/hello_world
nano apps/hello_world/main.py # Paste the code there
touch apps/conncheck/__init__.py
```

The last command opens an editor, pastes your code into it and exits. That's all you need to create your own pyLCI app. To test it, you can run pyLCI manually. First, run **sudo systemctl stop pylci** to stop the global pyLCI process (so that it doesn't try to grab the screen and buttons you're using). Then run **sudo python main.py**, use the buttons to scroll down to your application and press the Enter key. "Hello, world!" should appear on the screen.

Writing your own app

Of course, an app is only truly useful once you can somehow interact with the outside world. Let's take a real-world situation: there's something wrong with your network and you want to know where the problem lies – is it your computer, your router or your ISP? Let's write a simple pyLCI app for that.

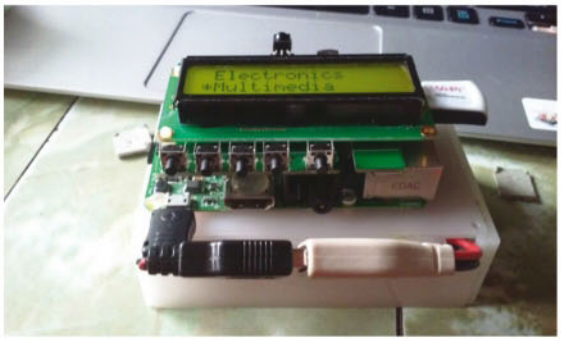
First, what do we check for? There are four typical places for connectivity problems to appear: the computer you're using, your router, the DNS or your internet provider. To cover most of those, we can: 1) check whether we can ping an IP address that belongs to Google, ruling out ISP problems; 2) try to ping a website, referring to it by its domain name – making sure our problem is not a DNS problem.

There's a great Python library that lets us do both of these tasks easily – it's called **pyping**. Let's install it:

```
sudo pip install pyping
```

Note that due to the way ping works on Linux, you need to use the **pyping** library as root – that is, using **sudo**. As pyLCI has to run as root anyway, this is not a problem for our app, but it's something to remember while experimenting.

The pyping module has a **ping** function, which accepts a string: either an IP address or a domain name. It raises an exception if the name can't be resolved – and if we're trying to resolve a well-known name such as **google.com**, this would likely mean that DNS isn't working for us. Otherwise, it returns a response object, with a **ret_code** property; if the destination was reached, it's set to 0, otherwise it's set to 1. Perfect! So let's write some simple code that uses this to check whether an internet connection is accessible. First, let's check the internet connection in general, then whether the DNS works:



Right A portable battery-powered Raspberry Pi, with pyLCI greatly improving the portability

```
menu_name = "Connection test"

import pyping # Our library of choice
from ui import Printer

i = None; o = None

def callback():
     success = pyping.ping("8.8.8.8").ret_code
     if success == 0:
          Printer("Connection working!", i, o)
     else:
          Printer("Connection failed!", i, o)
     # Then, check DNS
     try:
          pyping.ping("google.com")
     except:
          Printer("DNS failed!", i, o)
     else:
          Printer("DNS working!", i, o)
```

This is all we need – our proof-of-concept app is done. Let's put it with the other networking-related apps in pyLCI, in the **apps/network_apps** directory:

```
mkdir apps/network_apps/conncheck
nano apps/network_apps/conncheck/main.py
# Again, paste our code in the editor
touch apps/hello_world/__init__.py
```

Adding features

There's one more important feature we can add to our new app: 'captive portal' detection. When you connect to certain Wi-Fi hotspots, such as those in coffee shops, they won't let you access the internet straight away; instead, they redirect you to a page that requires you to agree with their Terms of Service, or even create a login and password.

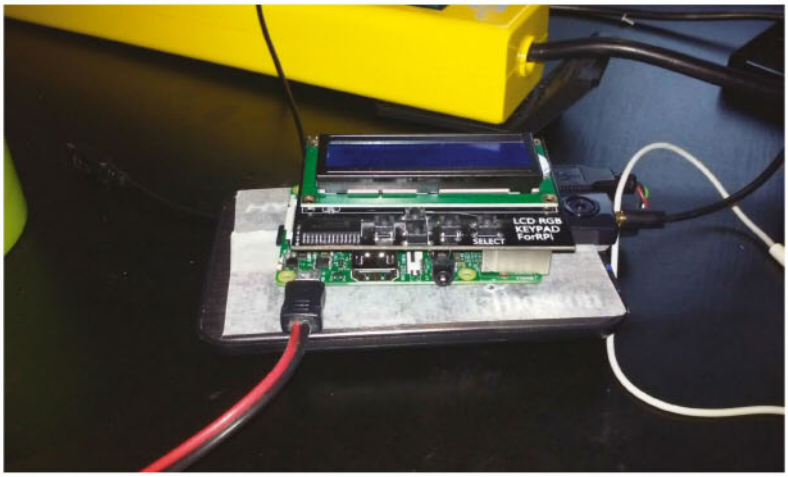
This means that your Pi won't be able to access the internet straight away and, in many cases, will also mean that you won't be able to SSH into it. Now, we can't yet do much in these kind of situations, but we can at least detect it, so that you spend less time on debugging your network connection.

To detect if a captive portal is redirecting us somewhere else, we can fetch a known page and check its contents. One example of such a known page is IcanHazIP, **http://icanhazip.com**. If you visit this page, you'll see your external IP on a blank page, and that's it. So we can trivially check if we're getting an IP address, or if we're being redirected to somewhere else – and we can get our external IP address, too. At the top of **main.py**, with all the other imports, let's import the **requests** library – we can use it to download the webpage:

```
import requests
```

Then let's add a function for captive portal check:

```
def is_captive_portal():
```



Above A spectrum monitoring station with an RTL-SDR dongle and an SSD; pyLCI shows system status

```
     try:
          r = requests.get("http://icanhazip.
com")
          assert (r.status_code == 200)
```

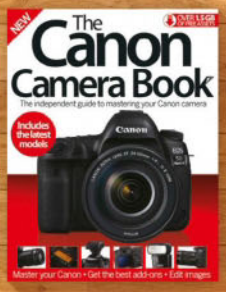
At this point, if the status code is not 200 we've been redirected somewhere – which shouldn't happen with this website. So if the code is not 200, an AssertionError will be raised and we'll go straight to the **except AssertionError** block. However, that might not be enough; let's parse the text we received from the page, to make sure that what got is actually an IP address – in other words, it's four numbers separated by three dots:

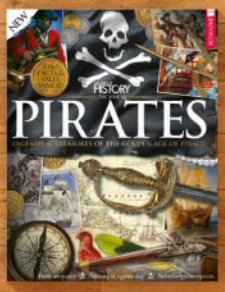
```
          ip = r.text.strip()
          assert (ip.count(".") == 3)
          parts = ip.split(".")
          assert all([part.isdigit() for part
in parts])
     except requests.ConnectionError:
          # Internet connection problem?
          return False
     except AssertionError:
          return True
     else:
          return False
```

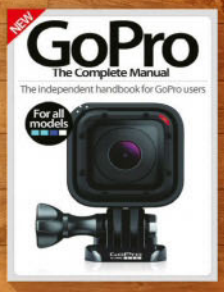
In the callback, let's add one more stage of checks:

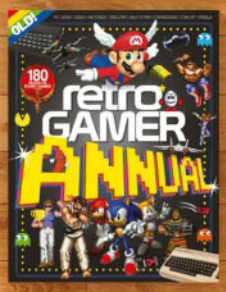
```
      [...]
          Printer("DNS working!", i, o, 3)
      if is_captive_portal() == True:
           Printer("Captive portal
detected!", i, o)
      else:
           Printer("No captive portal
found!", i, o)
```

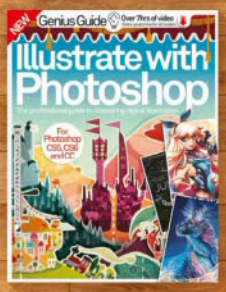
As you can see, it's easy to add your own functions to pyLCI. In more complicated apps, you can use RPC so that your app code isn't contained in pyLCI, and pyLCI has helpers functions for this, too. Happy hacking!

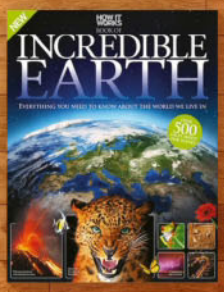


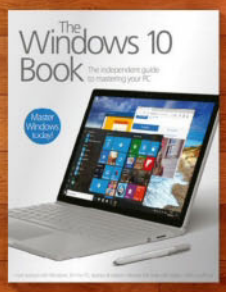






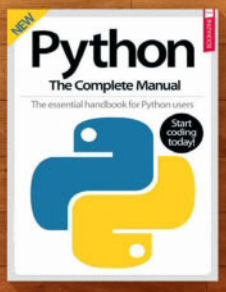


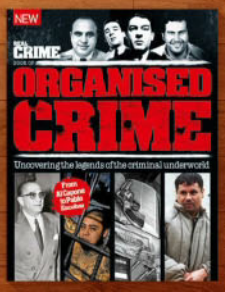


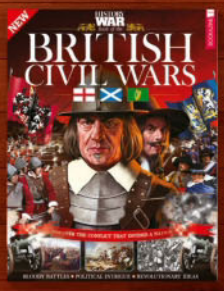


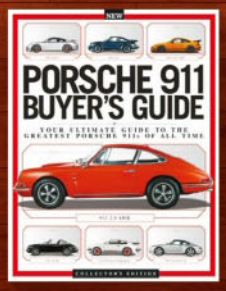
Discover another of our great bookazines

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.

Reviews

GROUP TEST

Backup tools

Use one of these comprehensive backup tools to minimise downtime and sail through any data disaster without losing sleep or data

Back In Time

Inspired by Mac OS X's Time Machine backup app, Back In Time is built around the versatile rsync utility, and offers advanced features to create local and network backups via a graphical interface. It enables multiple backup profiles and saves space thanks to hard links.

http://bit.ly/lud_backintime

Duplicati

In addition to local and network backups, Duplicati can also house backups online. Indeed, one of its best features is support for several popular cloud-storage services. Its features are powered by a collection of command-line open source utilities such as GPG and AESCrypt.

www.duplicati.com

fwbackups

Don't let fwbackups' simple interface fool you – it packs in quite a useful set of features, including all the usual functions you'd expect from a modern backup solution. You can backup a folder or an entire computer to a local or a network destination with relative ease.

http://bit.ly/lud_fwbackups

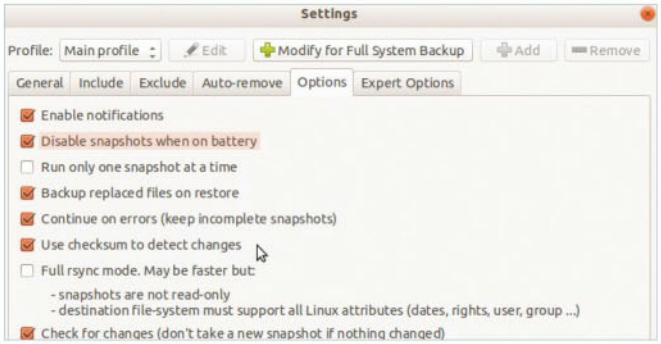
Grsync

As the name suggests, Grsync is a graphical frontend to rsync, which is one of the most frequently used backup command-line utilities on Linux. The graphical front-end does a nice job of exposing the CLI utility's large number of features for local and network backups.

www.opbyte.it/grsync

Back In Time

A general backup tool that does a little bit of everything



There are an impressive number of useful and expert options available, such as using file checksums to detect changes

Backup control

The tool allows you to fine-tune a backup job by specifying files you don't want to back up. It contains a list of common patterns for files to be excluded, such as temporary or hidden files, and you can also manually specify your own patterns, files and folders, or exclude files that exceed a particular size. Besides local backups, the app can also backup files remotely via SSH.

User experience

Back In Time installs two versions. One runs with root permissions to access and back up system files. The non-root version is meant for backing up personal files. On first launch you're asked to define a backup within the default profile using the multi-tabbed settings window. You can also create as many profiles with different backup settings as you need.

Compression and encryption

For security, Back In Time can encrypt snapshots using EncFS. You can use its interface to create a locally encrypted or an SSH-encrypted profile. However, there's no option to compress the snapshots to save both backup time and disk space.

Restoration

Back In Time creates snapshot directories, which means it copies a directory's entire contents into the backup if the contents have changed (you can specify if you want to use a file's checksum to detect changes, which saves some time). Using the tool you can restore individual files and complete directories to either their original location or to a custom one.

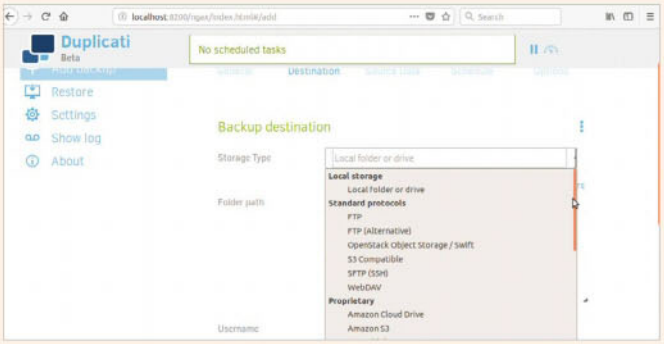
Overall

Back In Time doesn't support multiple network protocols or encryption standards, nor can it compress snapshots. However it still manages to offer a good number of configurable parameters.

7

Duplicati

Easy to use backup application that excels at everything it does



Duplicati automatically verifies backups stored on an online service periodically to ensure their integrity

Backup control

The program enables you to exclude directories or files from a backup by either manually defining filters or toggling one of the predefined options to exclude certain types of files. There's also an option to exclude files that are larger than a particular size. Furthermore you can use the Advanced options pull-down menu to add and customise a host of other options.

User experience

There's a browser-based graphical interface which is clean and intuitive and can be locked with a password. Duplicati breaks down critical tasks into wizards; for example, the app runs through a five-step wizard to help configure a backup task. It also exposes just the right number of features for the job in hand, while advanced options are just a pull-down menu away.

Compression and encryption

Files are encrypted with AES-256 by default and optionally you can also use GPG before transmitting the backups to their destination. Duplicati also compresses all data before it is encrypted and transferred. It supports Zip and 7Z compression, and can detect and skip compression of already compressed files such as .mp3 and .jpg.

Restoration

The Restore Files wizard lists all backups, but also gives you the option to restore from backup files not listed or from a cloud service. Once you've selected a backup, Duplicity enables you search for and selectively restore individual files or the entire archive. Files can be restored to their original location or a different destination.

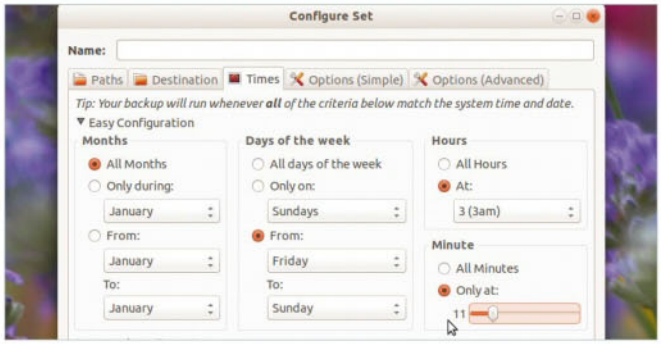
Overall

Duplicity offers all the best features of its peers and excels at virtually all backup and restoration tasks. Its intuitive interface offers just the right number of options at any given point.

9

fwbackups

An easy-to-use program that's mediocre in all areas except usability



The graphical crontab configuration makes scheduling automatic backups a piece of cake

Backup control

There are a variety of backup modes and formats, including an archive and clone copy designed to help recover data from damaged disks. It has an intuitive interface to help create recurring backups as well as one-time backups. fwbackups can also exclude files that match a pattern, but you'll have to manually specify the pattern or the file to exclude yourself.

User experience

The multi-tabbed interface neatly organises the options required to create a backup set. It's also easy to configure the scheduler for users not familiar with the ways of cron, which is what fwbackups uses for scheduling tasks. Optional parameters are divided into simple and advanced tabs and should be tinkered with only after leafing through the docs.

Compression and encryption

By default, the program backs up files into a tar archive that can optionally be compressed using one of two supported options: gzip, which compresses faster, or bzip2 which tends to create smaller archives. However there's no option to encrypt the backed-up archives in any way.

Restoration

fwbackups does both incremental and differential backups, but the default archive backups aren't incremental. The app enables you to select a backup from which to restore, along with a list of dates. It also has several features for organising stored backups, including the ability to remove expired ones.

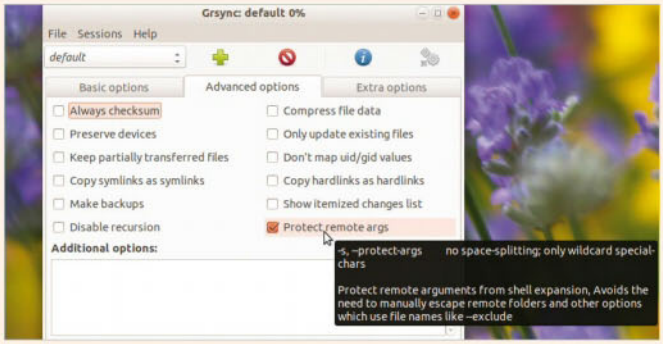
Overall

fwbackups does its job as advertised, but lack of encryption is one of its biggest drawbacks. On the other hand, the app's intuitive interface and the easy scheduler are two of its strong points.

6

Gsync

Essentially a straightforward front-end to the venerable rsync



You can hover over any of the tickboxes to get a brief description about the option and what it does

Backup control

Grsync offers the malleability of the rsync backend, but many of its options have to specified manually. For example, to exclude files you'll have to manually specify them with the `--exclude` option. That said, you still get quite a few options under the basic, advanced and extra options tabs.

User experience

The program might seem unintuitive to first-timers and those unfamiliar with rsync, since it lacks the wizard-like interface of many of its peers. Also, the sheer number of tickbox options to ponder might seem daunting, though the app does select reasonable defaults. The option to simulate backups is very handy because rsync is fairly powerful and a misconfiguration, such as a seemingly harmless trailing slash, can wreak havoc.

Compression and encryption

There's an option to compress the backup archive before transferring it to the destination, though for some reason the program suggests this is only useful if you're backing up data to or from a remote network location. Because the app uses SSH to connect with remote locations, the transfers are automatically encrypted.

Restoration

Thanks to the features in rsync, you can create incremental backups. However, Grsync doesn't offer a dedicated option to restore backups. Instead you need to switch the values in the source and destination fields using the menu to restore files to their original location, which seems a somewhat fiddly way to do things.

Overall

Grsync helps ease rsync's learning curve. It exposes some of the most frequently used options and comes with enough defaults out of the box to avoid overwhelming new users.

6

In brief: compare and contrast our verdicts

	Back In Time		Duplicati		Fwbackups		Grsync	
Backup control	Can exclude files that match a pattern or which exceed a particular size	8	Offers a large number of configurable parameters to customise backups	9	Offers various backup modes but some functions require manual input	6	Exposes a number of rsync's features but many have to be specified manually	7
User experience	Fairly intuitive app, with support for creating multiple backup profiles	8	The important tasks are broken down into wizards for easier navigation	9	Offers lots of configurable parameters and as well as a wonderful scheduler	7	Isn't very appealing but the option to simulate backups is very useful	5
Compression and encryption	Can encrypt with EncFS, but offers no option to compress backups	5	Offers a couple of options for both encrypting and compressing backups	8	The app's TAR archives can be compressed, but they can't be encrypted	5	Can compress backups before transferring them via SSH to a remote location	6
Restoration	This app can restore individual files to their original or custom location	8	Enables you to search inside backups and restore individual files to custom locations	9	Does both incremental and differential backups and auto-removes expired backups	6	Can restore files by reversing the backup source and destination locations	6
Overall	**Offers a fairly good number of options but is short on some useful ones**	**7**	**Present the greatest number useful features of any program here via an easy-to-use interface**	**9**	**Its intuitive interface doesn't make up for a lack of important features like encryption**	**6**	**Takes some pain out of using the CLI-only rsync utility, but perhaps not enough**	**6**

AND THE WINNER IS...

Duplicati

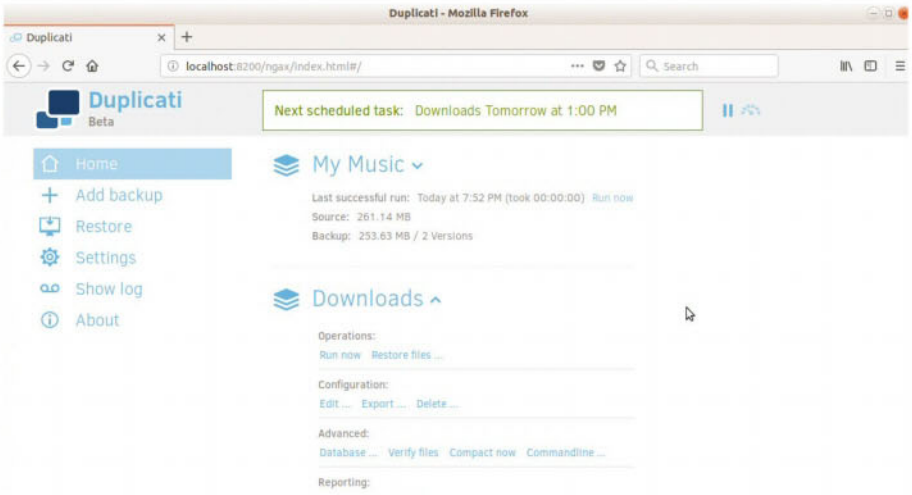
The number of actively maintained open source backup programs has been in steady decline over the years. Many of those that are regularly updated are designed for large-scale deployments and are too elaborate for most users and the average-sized network. The top-tier desktop distributions all ship with their own backup app that's well integrated into their respective environment, which further eliminates the need for a third-party backup program in most cases.

Of those we tested, Duplicati is head and shoulders above the others, both in terms of features as well as usability. The program offers all the best functions of its peers and yet doesn't inundate its users with an unending list of buttons and toggles. It's easily installable on virtually all Linux distributions and its browser-based interface does a wonderful job of catering to both first-timers and experienced campaigners. To top it all, Duplicati is preconfigured to work with a wide range of online storage services.

You can easily use Duplicati on the desktop to backup your personal files, or across the network to backup a bunch of computers to a central backup server. The app has very useful illustrated guides to hand-hold first-timers, and its command-line interface will help advanced users to script backups.

Duplicati is cross-platform and licensed under LGPL, so requires Mono. The 2.x series of the app is still in beta but is considered stable enough for production environments. Ultimately, Duplicati manages to strike the right balance between form and functionality. It's a wonderful backup tool that's dexterous enough to suit all kinds of users and use cases, and well worth adding to your day-to-day operations.

Mayank Sharma



Duplicati comes with an updater that downloads and installs newer versions of the app

The source for tech buying advice

techradar.com

Above You have to give kudos to the designers for at least trying to make the FRITZ!Box look different to most standard routers

HARDWARE

FRITZ!Box 7590

Price
£225
Website
https://en.avm.de
Specs
Wireless Connectivity Dual band 802.11a/b/g/ac/n, 5GHz and 2.4GHz; 4x4 MU-MIMO; DECT base station
Processor Dual quad-core MIPS CPU @ 2GHz
Memory 512MB RAM
Storage 512MB NAND flash
Ports 1x Gigabit WAN, 4x Gigabit Ethernet LAN, 2x USB 3.0,VDSL/ADSL2+ Modem (Annex B), 2x Analogue Phone, 1x ISDN

The do-it-all router that does even more than that

Whatever you might think about the slightly startling name, the latest FRITZ!Box does more than its jaunty logo and wacky, dated styling might suggest. In fact, it seems it's absolutely insistent on doing everything it possibly can. It routes wireless traffic, it routes wired traffic, it routes DSL and analogue phone lines, it manages DECT phones, printers and network storage.

Plug in a 4G dongle and it'll even provide fallback connectivity for your network in case of a main connection outage. You are entirely unlikely to need at least half of its features, but if you're craving a router with flexibility you've definitely found it.

The real question is, though, do we want this in our house? Does that flexibility warrant the price? Are we prepared for visitors to ask what a FRITZ!Box does?

Those questions are answered not just by the FRITZ!Box 7590's wide range of almost enterprise-level features, but by the fact that it almost keeps pace with similarly priced competitors in terms of performance and power.

The German-made FRITZ!Box 7590 is £225. That's a whole lot of money for a router, particularly if you're not looking for more than its base features, but if you're administering a SoHo situation, or you're just really serious about your home connection, its myriad ports and possibilities might just make that a reasonable outlay.

We may have muttered facetiously about the number of features the FRITZ!Box 7590 squashes in to its small frame, but AVM has at least done a fabulous job of putting them in a small space.

Above One thing you won't be short of is connectivity options – but you'd expect that at this price

> FRITZ!OS is a great firmware front-end packed with wizards and diagnostic tools that make config easy

The new-look design, which adopts a couple of styling features from its classic predecessors but chucks out the retro-futuristic fins, is sleek, offers good airflow underneath, and can be wall-mounted if you choose. It's also visually very different from most routing hardware, opting for a grey/dark grey/dark red combo that's uniquely FRITZ!Box.

This is a great router for VDSL users – including BT Infinity subscribers – or at least it will be, thanks to its support for VDSL 35b super vectoring. Essentially this is a way to tease up to 300Mbit/s out of twisted-pair cabling, presuming you're on a short enough hop to the cabinet. No UK ISP supports it at this time, but it's good to be prepared for the future.

The 7950's rather unique DECT phone support is also a pretty big deal, particularly when tied in to AVM's excellent FRITZ!OS firmware and FRITZ!App phone application; support for Fon means that you can route your calls as you wish, and even make landline calls, after a fashion, through your mobile.

That's not the only app that can manage the router's features – there are four or five on offer covering a host of different aspects.

All of those features plus world-beating performance might have been too much to ask, and so it proved, as the FRITZ!Box 7590 didn't blow away our performance tests. It's not bad, by any means, but middling Wi-Fi performance puts all those additional extras (and the 7590's pretty high price) into perspective if you're not going to take full advantage of them.

Without access to VDSL we weren't able to test the super vectoring performance, but realistically you should be able to take the theoretical maximum of 300Mbit/s with the same pinch of salt applied to most internet speed ratings – it's all going to depend on the distance to the cabinet, the quality of the cabling used on your local loop and your ISP's temperament on any given day. All things that the FRITZ!Box has no control over.

Control, though, is its real strong point. FRITZ!OS is a great firmware front-end packed with wizards and diagnostic tools that make configuration easy, and it continues the enterprise-aping feature set with some pro-level tweaks. Fancy dropping the Ethernet ports from gigabit to 100Mbit to save a bit of power? Disabling Wi-Fi on a schedule? Blocking pings, filtering ports, and everything in between?

It's an immensely capable router with a huge number of features, comprehensive firmware, some slightly dodgy external design, and a price that's just a little too high. What home doesn't need its landline turned into something actually useful? Why not centralise storage, printers, and the like? This changes our view of what a router should be – the only problem is that it's not, in its price bracket, quite as good at the bits that really matter.

Alex Cox

Pros

It's great to have so many options and the FRITZ!Box 7590 offers more than you could ever truly need. The app management and comprehensive firmware put those controls at your fingertips.

Cons

That raft of features, outside of very specific use cases, just isn't required, so paying a little extra for mid-range Wi-Fi power might sting.

Summary

The FRITZ!Box 7590 is a bit of a surprise. It's not the feature set, or the striking look, or the comprehensive firmware that does it – it's that this is an inherently likeable, capable and versatile router that makes us want to experiment.

8

Above Netrunner Rolling enables you to switch kernels and use proprietary drivers

DISTRO

Netrunner Rolling 2018.01

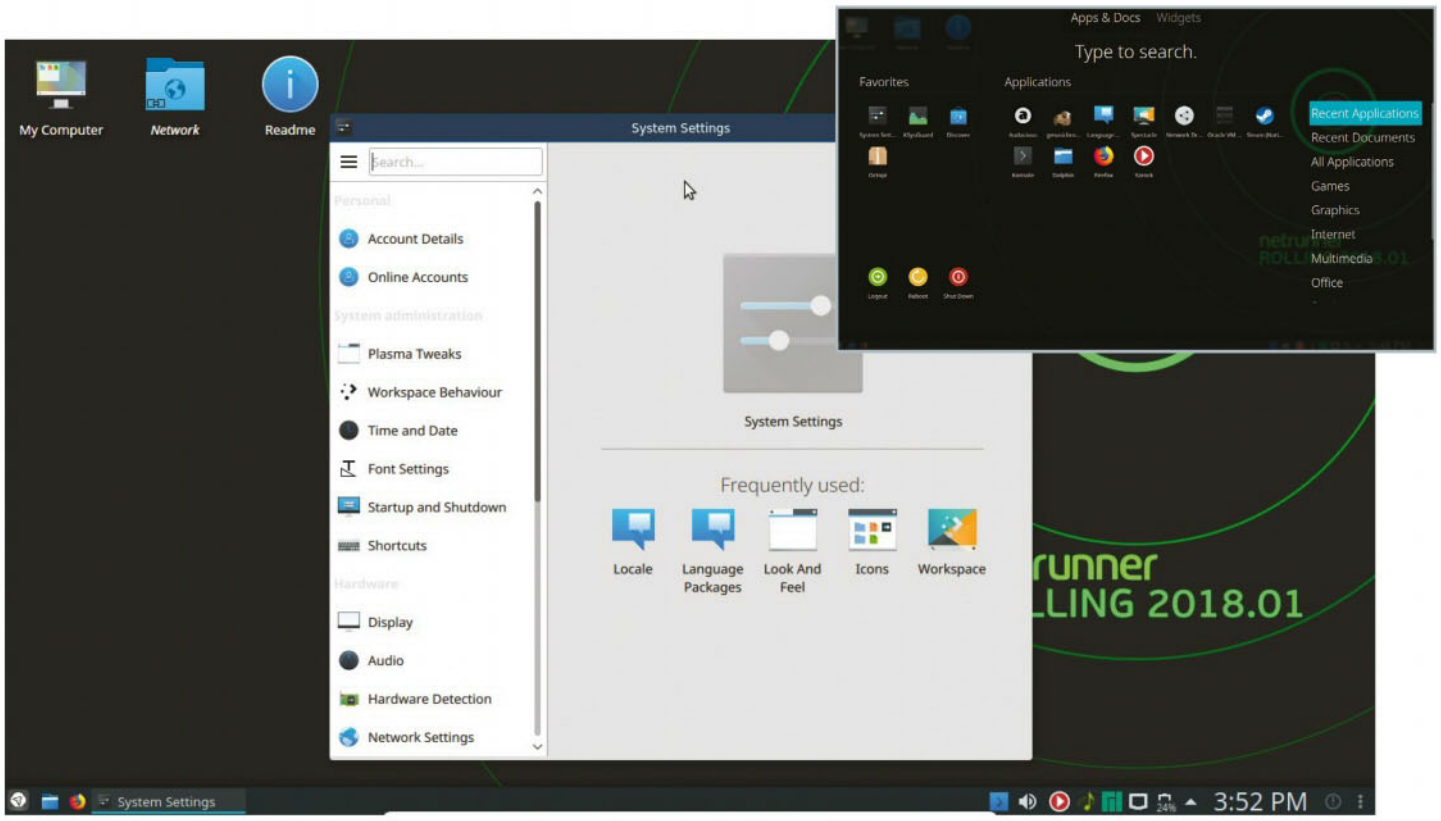
RAM
1GB
Storage
10GB
Specs
1.6 GHz processor, 64-bit only
Available from
www.netrunner.com

There's a seemingly endless list of Arch-based KDE distributions. Does this one manage to set itself apart from the others?

Netrunner is a product of Blue Systems, a German IT company which is one of the most prominent supporters of the KDE project and which employs quite a few core project developers. Netrunner has multiple KDE-centric versions. There's the headline Desktop release, based on Debian, and this rolling release based on Manjaro Linux – one of the most popular distributions built atop Arch Linux. Given its close ties to KDE, Netrunner ensures its distributions offers the best KDE experience.

Netrunner Rolling doesn't try to mask the fact that it's based on Manjaro. For example, if you've used Manjaro you'll notice Netrunner Rolling uses the same boot menu. The unique point of this boot menu is that it allows users to select whether they wish to boot the Live environment with either the free or proprietary drivers. The one very prominent aspect in which Netrunner Rolling distinguishes itself from Manjaro is its heavily customised Plasma desktop environment. The distribution has also worked on its usability by reorganising the various modules in the KDE System Settings manager. For example, a new section named Plasma Tweaks houses all the settings related to the look and feel of the KDE desktop and its various elements such as icons, window decoration and so on.

Unlike stock KDE desktops, the one in Netrunner Rolling doesn't have the Folder View widget; instead, it's completely blank aside from three icons. Command-line warriors will appreciate the inclusion of Yakuake, a utility that enables them to pull down a terminal window at any time with the F12 key.



Above You can anchor the distribution to your hard disk using the intuitive distribution-agnostic Calamares installer

> The one very prominent aspect in which Netrunner Rolling distinguishes itself from Manjaro is its heavily customised Plasma desktop environment

The distribution has also done a wonderful job of integrating the items borrowed from the Manjaro Settings Manager into KDE System Settings. For example, the MSM Language and Language Packages components are found in Netrunner's KDE System Settings > Account Details component.

Netrunner Rolling's smart-looking desktop uses KDE's full-screen Dashboard application launcher by default. The only issue with the launcher is incomplete labels for several menu items. In the absence of tooltips, this will not only confuse and irritate first-time users, but also looks very unprofessional. You can, however, switch to any of the three other included app launchers.

One strong suit of the distribution is its list of installed apps: Firefox Quantum 57, for instance, launches almost instantly thanks to a pre-load daemon. It's also equipped with a couple of add-ons such as uBlock Origin for filtering content. Similarly, KDE's default file manager Dolphin ships with several extra extensions enabled by default, including the ability to manage Git, SVN, Mercurial and Bazaar repositories.

There's also a host of marquee KDE apps such as Kdenlive, Krita, KDE Marble and more. Surprisingly, Netrunner Rolling includes the mainstream LibreOffice instead of KDE's almost-as-good Calligra Suite, perhaps because of the former's improved usability. In the same vein, the distribution also rolls in a couple of proprietary apps, including Skype and the Steam installer.

To appeal to a wide variety of users, Netrunner Rolling includes a handful of niche apps such as vokoscreen for screencasting, Handbrake transcoder, GRUB Customiser and VirtualBox. Multimedia duties are handled by gmusicbrowser, Audacious, and SMPlayer. There's also the Yarock music player, which is visually appealing and includes a bunch of online radio-streaming services.

For package management, the distribution includes Octopi, which is a front-end to Arch's Pacman package manager, and also Plasma Discover, which is KDE's app store. You can use either of the store apps to track and install updates, which is a nice touch.

Mayank Sharma

Pros

Aesthetically pleasing desktop with a string of usability customisations and a wide range of apps.

Cons

Rearranged modules in KDE System Settings might adversely affect usability for KDE users.

Summary

Netrunner Rolling is an aesthetically pleasing distribution built atop a solid foundation. It makes good use of its base distro by nicely incorporating its key components. The distro does a good job of catering to a wide variety of users, from first-timers to experienced.

8

WEB BROWSER

QupZilla 2.2.5

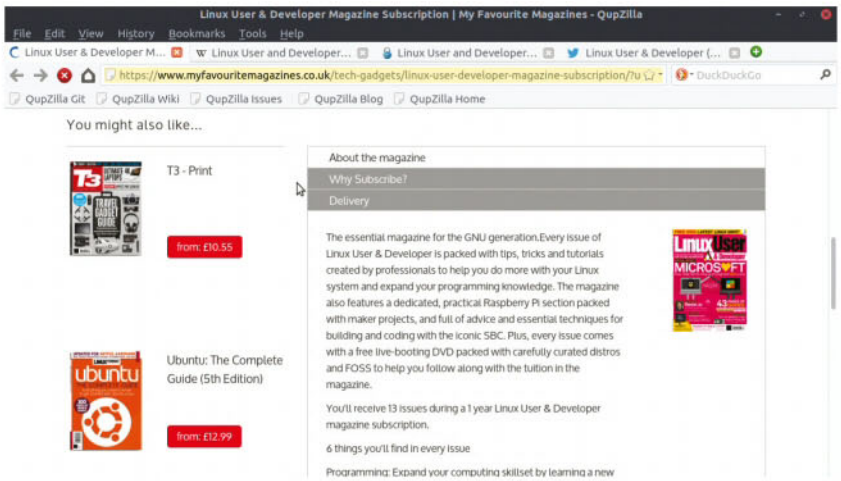
A lightweight browser that functions like any mainstream one

If the web browser is slowing down your computer, try switching to QupZilla – a feature-rich, lightweight alternative that uses QtWebEngine. It has all the standard functions that you'd expect from a modern web browser, including bookmarks, history and tabs. Additionally, the browser has an RSS feed reader and includes a password manager.

QupZilla also has several interesting features, such as the ability to change its user-agent to be more appealing to certain websites, and is capable of taking a screenshot of the entire web page. You can also tie common actions to mouse gestures for convenient execution, plus there's a built-in ad blocker.

QupZilla is very tweakable and has an extensive Preferences window, which is neatly organised. From here you can switch to six bundled themes and change the behaviour of the tabs and address bar, and access the password manager. QupZilla also allows you to switch between native Linux system and OSD notifications.

Overall, QupZilla manages to deliver the closest experience to mainstream browsers while taking up only a fraction of the resources in terms of processing power.



Above The latest release brings back the AdBlock and GreaseMonkey icons in the status bar, and can display remaining time for downloads in the download manager

Pros

A standards-compliant browser that offers all the useful features in a lightweight application.

Cons

The Qt5 baggage on a non-KDE environment, and the limited number of add-ons and plug-ins.

Great for...

Conserving resources on an underpowered or struggling machine.

https://qupzilla.com

METADATA MODIFIER

ExifTool 10.77

Read and manipulate all kinds of image metadata

ExifTool is a flexible utility for working with photographic metadata. Photo management apps such as digiKam, Geeqie, and gThumb enable you to view and edit metadata for an image or two, but the command-line ExifTool lets you process dozens of images in a fraction of time. The program doesn't need to be installed – you can simply download and extract the tool's tarball and run it directly.

When you point ExifTool to one or more image files it generates a long list of its metadata; besides JPG, a wide range of other image formats is supported, as well as raw files from DSLRs and video files. You can head to the project's website to view the ever-expanding full list of supported formats. ExifTool can perform other useful actions besides merely reading metadata. For example, the tool can determine the photo's creation date and time and then use the obtained values to rename the file. There's support for a wide range of parameters and switches, and you should take time to read through its documentation to familiarise yourself with the options to edit or add metadata; this is perhaps one area where GUI users might feel a bit lost.

ExifTool also comes in handy to help analyse images using metadata. You can, for example, easily extract the lens and focal length information from photos and arrange these in a tabular format. Privacy-conscious users can also use the tool to strip all metadata from images before sharing them on social media, and the latest release improves its ability to read new types of metadata.

Pros

Can read and write all kinds of metadata in video files and images from all types of cameras.

Cons

It's an extensive utility with a seemingly unending stream of command-line switches and options.

Great for...

Batch-processing metadata for hundreds of images with just a single (complicated) command.

www.sno.phy.queensu.ca/~phil/exiftool

VIDEO DOWNLOADER

youtube-dl 2018.01.27

A command-line tool to grab online videos

There's no dearth of graphical tools that can download content from video sharing websites. What sets youtube-dl apart from the rest is that it's a command-line utility, and despite its name it can fetch videos from dozens of video sharing services including Vimeo, Udemy, Khan Academy, Ted, PBS, BBC, Archive.org and a lot more.

The utility is written in Python and you can install it with Python's PIP package manager using `sudo pip install youtube_dl`. Despite being a CLI utility, using it is fairly straightforward: just specify the link to the video to download it. But don't confuse simplicity of use with a paucity of features, because the program is very dexterous. You can, for example, use the `-F` switch to fetch a list of available resolutions for a video and then download the version you want. Similarly, the utility enables you to download subtitles for videos, and even entire playlists with a single command.

You can also use it to extract audio from a video and save it in a particular format and bitrate, followed up with various post-processing if you have ffmpeg installed. For example, it can encode downloaded videos to another format, add metadata, embed subtitles and so on.

youtube-dl is also a very capable download manager in its own right and supports resuming interrupted downloads, as well as being able to limit download rates. You can forget about downloading from commercial sites such as Netflix, though, as youtube-dl won't touch anything DRM-protected.

Pros

Straightforward to use and Chock-full of features and supports a wide range of video-sharing websites.

Cons

Limited in what it can download and you'll need to read through its man page before you can use it productively.

Great for...

Automatically downloading a whole bunch of videos.

http://rg3.github.io/youtube-dl

MUSIC PLAYER

BallroomDJ 3.18.4

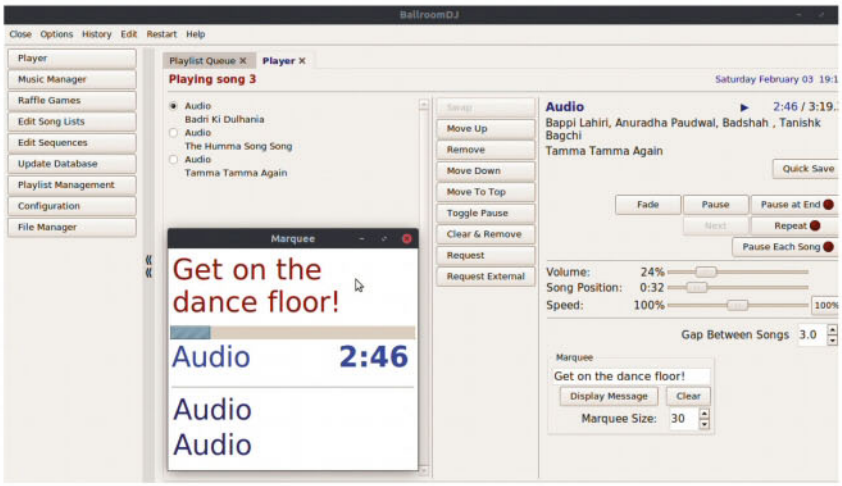
Dish out non-stop music all through the evening

If you're planning an event and want to stream music throughout without any intervention, you can delegate the task to BallroomDJ. This graphical app is designed for, well, ballroom events and dance studios that require uninterrupted music playback.

The app includes a music manager to help organise your media library. It also includes features such as automatic playlist generation and flexible playlist control, while other relevant functions include the ability to define maximum playback time and customise a marquee display.

You also have the option to trim the start and end of songs and adjust its playback speed and pitch without modifying the audio file. While the app will play music without any human intervention, it does give users the ability to interact with the play queue by adding, removing and moving tracks.

To install the app you simply download and extract the tarball from its website. Then run the command-line installer with the `./INSTALLER` command and follow the prompts to specify an installation directory, and download any additional components if required. The app relies on a variety of open source apps such as the VLC media player and the PulseAudio sound server.



Above Before you can put BallroomDJ to use you'll need to spend some time acclimatising to the app's Tcl/Tk-based graphical user interface

Pros

An extensive app with features and functions to enable uninterrupted music playback.

Cons

A cumbersome and esoteric user interface that takes some getting used to.

Great for...

Dishing out a constant stream of music throughout an event.

https://ballroomdj.org

RECOMMENDED

Hosting listings

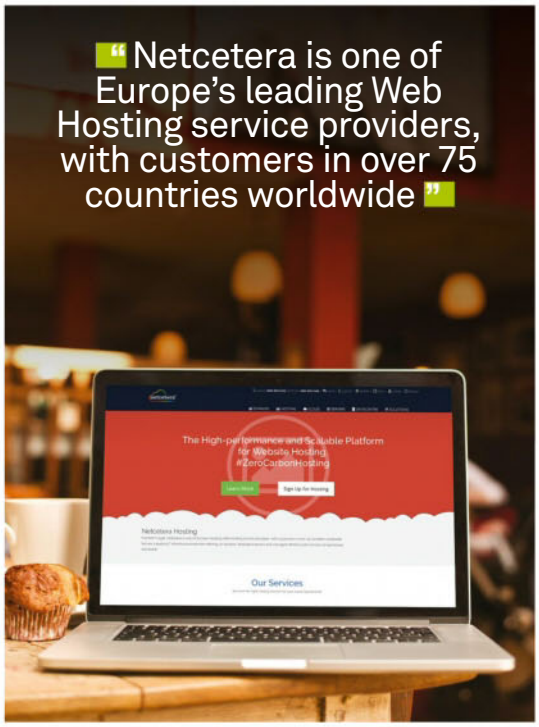
Featured host:
www.netcetera.co.uk
0800 808 5450

About us

Formed in 1996, Netcetera is one of Europe's leading web hosting service providers, with customers in over 75 countries worldwide. It is a leading IT infrastructure provider offering co-location, dedicated servers and managed infrastructure services to businesses worldwide.

What we offer

- Managed Hosting
  A full range of solutions for a cost-effective, reliable, secure host
- Dedicated Servers
  Single server through to a full racks with FREE setup and a generous bandwidth allowance
- Cloud Hosting
  Linux, Windows, hybrid and private cloud solutions with support and scaleability features
- Datacentre co-location from quad-core up to smart servers, with quick setup and full customisation



Five tips from the pros

01 Optimise your website images
When uploading your website to the internet, make sure all of your images are optimised for the web. Try using **jpegmini.com** software; or if using WordPress, install the EWWW Image Optimizer plugin.

02 Host your website in the UK
Make sure your website is hosted in the UK, and not just for legal reasons. If your server is located overseas, you may be missing out on search engine rankings on google.co.uk – you can check where your site is based on **www.check-host.net**.

03 Do you make regular backups?
How would it affect your business if you lost your website today? It's vital to always make your own backups; even if your host offers you a backup solution, it's important to take responsibility for your own data and protect it.

04 Trying to rank on Google?
Google made some changes in 2015. If you're struggling to rank on Google, make sure that your website is mobile-responsive. Plus, Google now prefers secure (HTTPS) websites. Contact your host to set up and force HTTPS on your website.

05 Avoid cheap hosting
We're sure you've seen those TV adverts for domain and hosting for £1! Think about the logic... for £1, how many clients will be jam-packed onto that server? Surely they would use cheap £20 drives rather than £1k+ enterprise SSDs? Remember: you do get what you pay for.

Testimonials

David Brewer
"I bought an SSL certificate. Purchasing is painless, and only takes a few minutes. My difficulty is installing the certificate, which is something I can never do. However, I simply raise a trouble ticket and the support team are quickly on the case. Within ten minutes I hear from the certificate signing authority, and approve. The support team then installed the certificate for me."

Tracy Hops
"We have several servers from Netcetera and the network connectivity is top-notch – great uptime and speed is never an issue. Tech support is knowledge and quick in replying – which is a bonus. We would highly recommend Netcetera."

J Edwards
"After trying out lots of other hosting companies, you seem to have the best customer service by a long way, and all the features I need. Shared hosting is very fast, and the control panel is comprehensive..."

Supreme hosting

CWCS MANAGED HOSTING — SUPREME HOSTING. SUPREME SUPPORT.

www.cwcs.co.uk
0800 1 777 000

CWCS Managed Hosting is the UK's leading hosting specialist. It offers a fully comprehensive range of hosting products, services and support. Its highly trained staff are not only hosting experts, it's also committed to delivering a great customer experience and is passionate about what it does.

- Colocation hosting
- VPS
- 100% Network uptime

Value hosting

elastichosts

elastichosts.co.uk
02071 838250

ElasticHosts offers simple, flexible and cost-effective cloud services with high performance, availability and scalability for businesses worldwide. Its team of engineers provide excellent support around the clock over the phone, email and ticketing system.

- Cloud servers on any OS
- Linux OS containers
- World-class 24/7 support

Small business host

HOSTPAPA

www.hostpapa.co.uk
0800 051 7126

HostPapa is an award-winning web hosting service and a leader in green hosting. It offers one of the most fully featured hosting packages on the market, along with 24/7 customer support, learning resources and outstanding reliability.

- Website builder
- Budget prices
- Unlimited databases

Enterprise hosting:

2020Media www.2020media.com

www.2020media.com | 0800 035 6364

WordPress comes pre-installed for new users or with free managed migration. The managed WordPress service is completely free for the first year.

We are known for our "Knowledgeable and excellent service" and we serve agencies, designers, developers and small businesses across the UK.

Budget hosting:

HETZNER ONLINE

www.hetzner.de/us | +49 (0)9831 5050

Hetzner Online is a professional web hosting provider and experienced data-centre operator. Since 1997 the company has provided private and business clients with high-performance hosting products, as well as the necessary infrastructure for the efficient operation of websites. A combination of stable technology, attractive pricing and flexible support and services has enabled Hetzner Online to continuously strengthen its market position both nationally and internationally.

- Dedicated and shared hosting
- Colocation racks
- Internet domains and SSL certificates
- Storage boxes

SSD web hosting

bargainhost

www.bargainhost.co.uk
0843 289 2681

Since 2001, Bargain Host has campaigned to offer the lowest-priced possible hosting in the UK. It has achieved this goal successfully and built up a large client database which includes many repeat customers. It has also won several awards for providing an outstanding hosting service.

- Shared hosting
- Cloud servers
- Domain names

Value Linux hosting

PATCHMAN WEB HOSTING

patchman-hosting.co.uk
01642 424 237

Linux hosting is a great solution for home users, business users and web designers looking for cost-effective and powerful hosting. Whether you are building a single-page portfolio, or you are running a database-driven ecommerce website, there is a Linux hosting solution for you.

- Student hosting deals
- Site designer
- Domain names

Fast, reliable hosting

:BYTEMARK

www.bytemark.co.uk
01904 890 890

Founded in 2002, Bytemark are "the UK experts in cloud & dedicated hosting". Its manifesto includes in-house expertise, transparent pricing, free software support, keeping promises made by support staff and top-quality hosting hardware at fair prices.

- Managed hosting
- UK cloud hosting
- Linux hosting

Free Resources

WHAT IS IT?

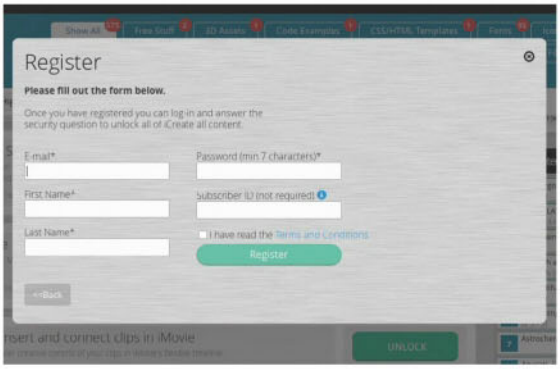
Every time you see this symbol in the magazine, there is free online content that's waiting to be unlocked on FileSilo.

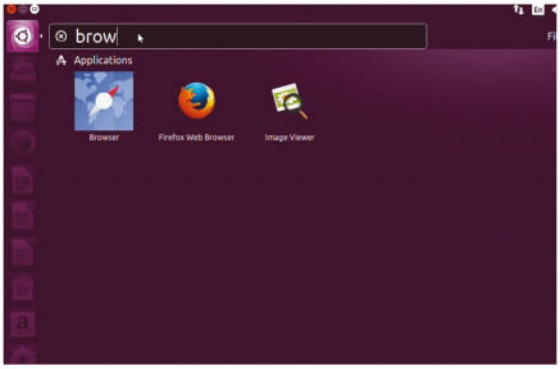
WHY REGISTER?

- Secure and safe online access, from anywhere
- Free access for every reader, print and digital
- Download only the files you want, when you want
- All your gifts, from all your issues, all in one place

Welcome to Filesilo!

Download the best distros, essential FOSS and all our tutorial project files from your FileSilo account





1. UNLOCK YOUR CONTENT

Go to **www.filesilo.co.uk/linuxuser** and follow the instructions on screen to create an account with our secure FileSilo system. When your issue arrives or you download your digital edition, log into your account and unlock individual issues by answering a simple question based on the pages of the magazine for instant access to the extras. Simple!

2. ENJOY THE RESOURCES

You can access FileSilo on any computer, tablet or smartphone device using any popular browser. However, we recommend that you use a computer to download content, as you may not be able to download files to other devices. If you have any problems with accessing content on FileSilo, take a look at the FAQs online or email our team at **filesilohelp@futurenet.com**.

Free for digital readers too!
Read on your tablet, download on your computer

Log in to **www.filesilo.co.uk/linuxuser**

SOFTWARE
BetterCAP 1.6.2

SOFTWARE
FOCA

DISTRO
Qubes R3.2

SOFTWARE
Grsync 1.2.6

Subscribe and get instant access

Get access to our entire library of resources with a money-saving subscription to the magazine – subscribe today!

This month find...

DISTROS

One excellent security conscious Linux distribution for you to enjoy: Qubes OS 3.2. The distro of choice for Edward Snowden while he hides in a Kremlin cupboard.

SOFTWARE

Two bundles for you to try. First we have a network bundle, including proxy servers, monitors and more followed by our backup tools from this month's grouptest, which includes the impressive Duplicati.

TUTORIAL CODE

Sample code for tutorials in this issue, including how to program with R, the statistical language, and a selection of tools for your InfoSec arsenal.

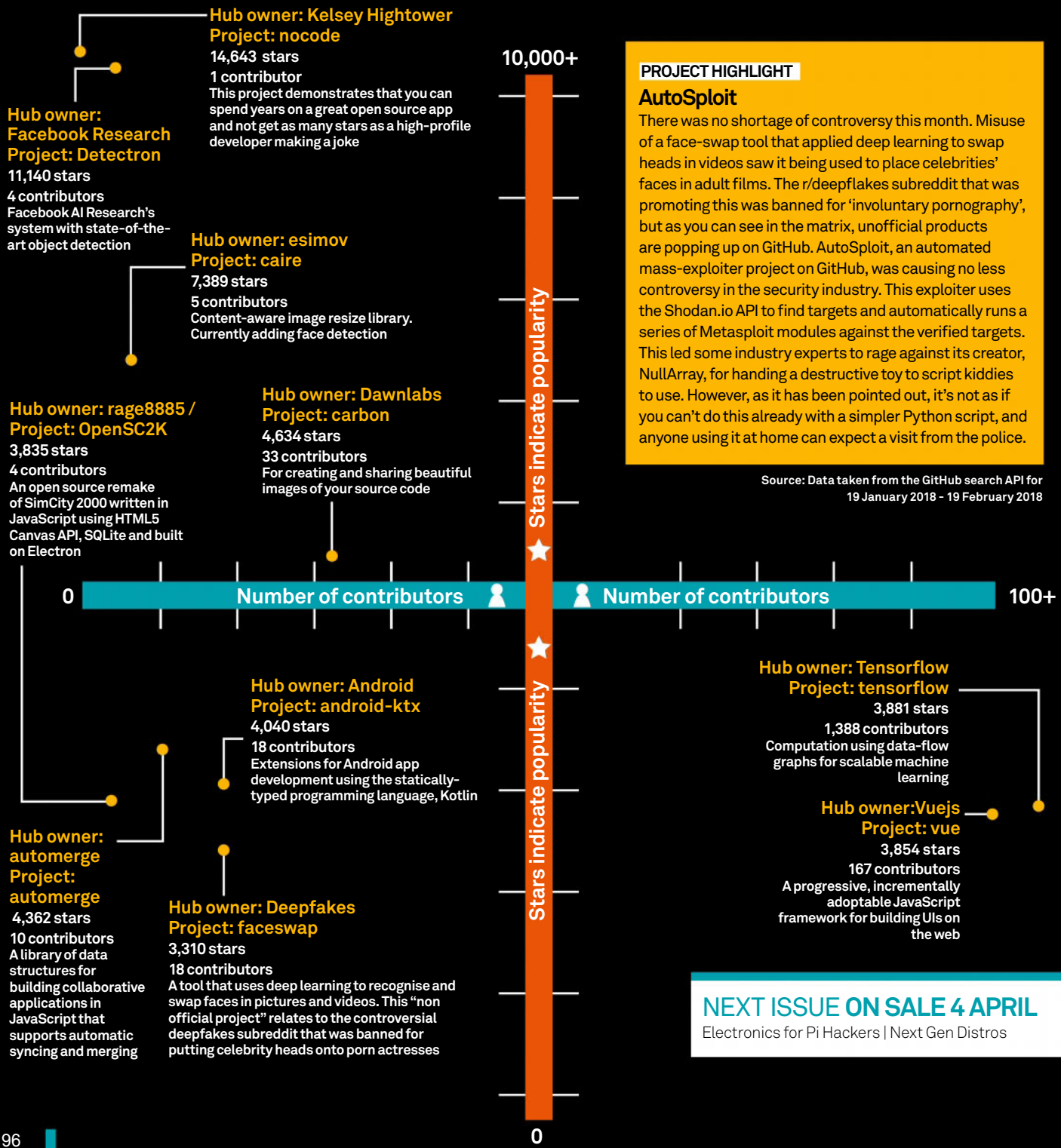
Subscribe & save!

See all the details on how to subscribe on **page 32**

MATRIX

Top open source projects on GitHub

The hottest software on the planet



PROJECT HIGHLIGHT

AutoSploit

There was no shortage of controversy this month. Misuse of a face-swap tool that applied deep learning to swap heads in videos saw it being used to place celebrities' faces in adult films. The r/deepflakes subreddit that was promoting this was banned for 'involuntary pornography', but as you can see in the matrix, unofficial products are popping up on GitHub. AutoSploit, an automated mass-exploiter project on GitHub, was causing no less controversy in the security industry. This exploiter uses the Shodan.io API to find targets and automatically runs a series of Metasploit modules against the verified targets. This led some industry experts to rage against its creator, NullArray, for handing a destructive toy to script kiddies to use. However, as it has been pointed out, it's not as if you can't do this already with a simpler Python script, and anyone using it at home can expect a visit from the police.

Source: Data taken from the GitHub search API for 19 January 2018 - 19 February 2018

Future

GIZMODO | UK

Not your average technology website

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

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

---

www.gizmodo.co.uk

twitter.com/GizmodoUK facebook.com/GizmodoUK

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

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

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

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

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

TWO SIDES