PROUDLY INDEPENDENT SINCE 2013

LINUXVOICE

November 2016 FREE SOFTWARE | FREE SPEECH www.linuxvoice.com

Raspberry Pi

Control a Pi with an RFID chip and a bit of Python

PROTECT YOUR PRIVACY

Stop the man from spying on you – thanks to Free Software!

HOW TO MAKE SOFTWARE THAT NORMAL PEOPLE CAN USE P32

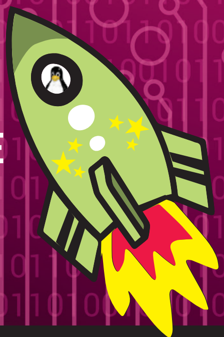
TURN A KINDLE FIRE INTO A CHEAP LINUX TABLET – P70

BURSTING WITH AWESOME TUTORIALS!

IRC Code a robot to shout into the void automatically

DOCKER Spin up container images on Linux the hard way

UBUNTU SNAPPY Finally, there's one package format to rule them all



YOU'VE COME A LONG WAY

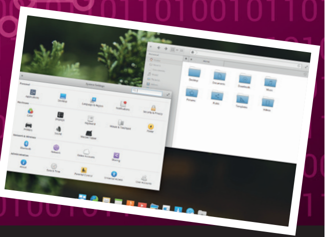
LINUX EVERYWHERE

Linux isn't just on servers – it's kicking arse in space and in science

LINUX DISTRO

ELEMENTARY OS

Pretty isn't just a feature – it's a way of life for this superb system



FREE SOFTWARE | FREE SPEECH
ISSN 2054-3778
November 2016 £5.99 Printed in the UK
9 772054 377001 11

OPENBSD › ARDOUR › WIRESHARK & MORE!

LINUXVOICE

The only Linux magazine available as DRM-free PDFs and ePub

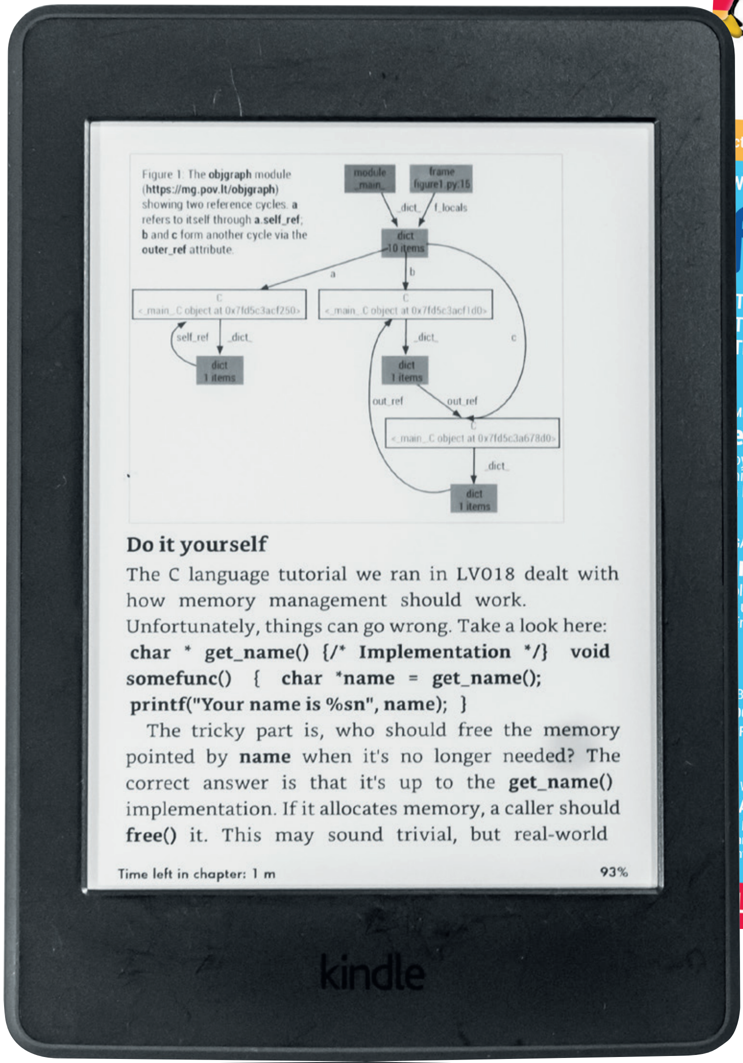


Figure 1 The **objgraph** module (https://mg.pov.lt/objgraph) showing two reference cycles. **a** refers to itself through **a.self_ref**; **b** and **c** form another cycle via the **outer_ref** attribute.

Do it yourself

The C language tutorial we ran in LV018 dealt with how memory management should work. Unfortunately, things can go wrong. Take a look here:

```
char * get_name() {/* Implementation */} void
somefunc() { char *name = get_name();
printf("Your name is %sn", name); }
```

The tricky part is, who should free the memory pointed by **name** when it's no longer needed? The correct answer is that it's up to the **get_name()** implementation. If it allocates memory, a caller should **free()** it. This may sound trivial, but real-world

Time left in chapter: 1 m 93%

PROUDLY INDEPENDENT SINCE 2013
LINUXVOICE
October 2016
FREE SOFTWARE | FREE SPEECH
www.linuxvoice.com
Banish Twitter!
Build your own social network to share daily blatherings
elgg
Why you need to try...
fedora
The best performance
The newest technologies
The most passionate community
BOINC: HELP SCIENTISTS DO SCIENCE BETTER P72
FOSSPICKS – THE BEST FREE SOFTWARE FOR LINUX P64
BURSTING WITH AWESOME TUTORIALS!
RetroPie
LUMO
LLARIUM > AFTERSHOT > OPENVR & MORE!
ISSN 2054-3778

shop.linuxvoice.com

FREE SOFTWARE AHEAD

The November issue

BEN EVERARD

Long-term Linux user and best-selling author Ben is usually found knee-deep in either Python code or a tangle of wires.

We are the generation that will get to decide what digital privacy is. Whatever laws are passed or whatever norms are established in the next five to ten years will become hard to deviate from in the future. If that means a complete erosion of any form of online privacy where your personal data is for sale to the highest bidder, then future generations will have to put up with that. If that means legal protections for our digital rights, then these too will be hard to change even for large multinational corporations with huge war chests.

Those of us who believe in digital privacy aren't just campaigning for our own rights, but those of all the digital citizens who will follow us. It's a matter of profound importance and we shouldn't let it be decided by those who seek to profit from it.

Ben Everard
Editor, Linux Voice

What's hot in LV#032

ANDREW GREGORY

I hated Gnome 3 when it first came out, but it's matured into a really nice desktop. Looking into the useability testing, I can now see that this is the result of hard work, not just random chance.
p32

GRAHAM MORRISON

Mike's IRC tutorial has inspired me to take on a new project using a bot to link IRC servers up to a synthesiser. Whatever is the most typed letter (A–G) each beat, it'll play.
p84

MIKE SAUNDERS

This month we look at Snappy and Docker: two container technologies. I'm starting to wonder if I should move with the times and add containers to my own operating system, MikeOS.
p38 & 84

THE LINUX VOICE TEAM
Editor Ben Everard
ben@linuxvoice.com
Deputy editor Andrew Gregory
andrew@linuxvoice.com
Editor in hiding Graham Morrison
graham@linuxvoice.com
Editor at large Mike Saunders
mike@linuxvoice.com
Games editor Michel Loubet-Jambert
michel@linuxvoice.com
Creative director Stacey Black
stacey@linuxvoice.com
Malign puppetmaster Nick Veitch
nick@linuxvoice.com
Editorial contributors:
Vincent Mealing, Mark Crutch, Jim Hall, Renata Gegaj, Simon Phipps, Les Pounder, Amit Saha, Mayank Sharma, John Lane and Valentine Sinitsyn

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

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

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

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

SUBSCRIBE ON PAGE 60

Contents

Because you need to be prepared when the Internet of Things gains sentience.

Regulars

News 06
PC-BSD has become something different; Vim 8 is finally upon us (after 10 years in development) and the Raspberry Pi has now sold over 10 million units.

Distrohopper 08
A new release of Zonrin OS is upon us, and once again, it's an excellent introduction to Linux for users of Other Systems. Plus we look fondly back on Rebecca Black OS...

Speak your brains 10
Why we're wrong, why we're awesome and what you think we should do more of, plus the now-obligatory priase for Linux Mint.

Subscribe! 12/60
Never miss another issue! Save money on shop prices! Get LV delivered to your door/ inbox every month!

FOSSPicks 64
We persuade Graham to put his sythesizer down for long enough to test some of the finest Free Software that money can buy. If it cost money, that is.

Core Tech 94
Find out what makes your Linux machine tick with a bona-fide doctor – this month, it's job scheduling with *Cron* and *Anacron*.

Geek Desktop 98
Rheumy-eyed Nick pull up his favourite chair and tells us a story, plus: there's Linux in them thar routers.

Cover Feature

14

Online privacy needn't be a contractiction in terms. Follow **Ben Everard**'s guide to privacy software on Linux and protect yourself from The Man.

Interview

40

Rachel Roumeliotis
The puppetmaster in charge of OSCON spills the beans on the move to London.

Feature

22

Linux everywhere
Robots, rockets, sub-atomic particle accelerators – Linux is everywhere, and it's doing a grand job.

FAQ

Ubuntu Snappy 38
Packaging in Linux could be better – let's get behind the latest standard and unite the clans!

Group Test

Hosted storage servers 54
Why let Google Drive look after your precious cat videos when you can do it yourself?

SUBSCRIBE ON PAGE 60

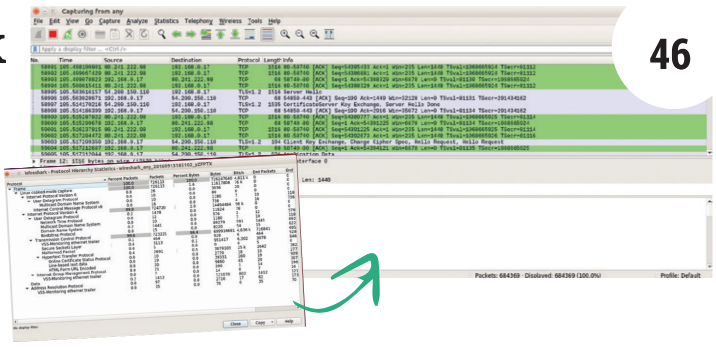
SECRETS OF NEXTCLOUD TURN TO PAGE 26

Feature

Ye Olde Linux
Delve into the archives to see what Linux used to look like – we've come a long way in 25 years.

Reviews

Wireshark 46
Keep tabs on everything to do with your network (including files being transferred over USB keys) with the most comprehensive network analyser we know of.

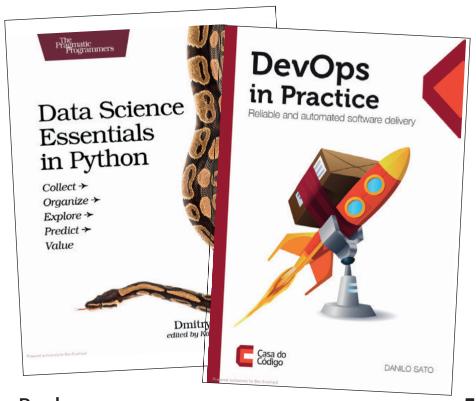

Capturing from any
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Wireshark · Protocol Hierarchy Statistics
Packets: 684369 · Displayed: 684369 (100.0%)
Profile: Default
Close
Copy
Help

OpenBSD 6.0 47
A Unix derivative for the security conscious – try it if your kitten videos are ultra-precious.

Elementary OS: Loki 48
Custom apps, chosen for form and function, all wrapped up in a coherent look and feel. Finally!

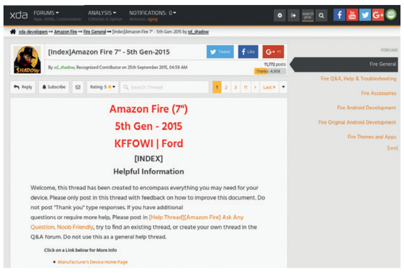
Ardour 5.4 49
This great audio editor is now available for users Windows – they were getting jealous of us...

Gaming on Linux 50
A bumper crop this month of quirky physics, updated nostalgia, and *Life Is Strange* – a gripping story with twists and turns a-plenty.


The Pragmatic Programmers
Data Science Essentials in Python
Collect
Organize
Explore
Predict
Value
Dmitry
edited by
DevOps in Practice
Reliable and automated software delivery
Casa do Código
DANILO SATO

Books 52
Still the best way to get information into your soft, spongy brain; at least, it is until they invent a way to upload knowledge *Matrix*-style.

Tutorials


xda FORUMS ANALYSIS NOTIFICATIONS: 0
xda-developers → Amazon Fire → Fire General → [Index]Amazon Fire 7" - 5th Gen-2015 by sd_shadow
Tweet Like G+1
Reply Subscribe Rating: 5 Search Thread 1 2 3 11 Last
FORUMS
Fire General
Fire Q&A, Help & Troubleshooting
Fire Accessories
Fire Android Development
Fire Original Android Development
Fire Themes and Apps
[Less]

[Index]Amazon Fire 7" - 5th Gen-2015
By sd_shadow, Recognized Contributor on 25th September 2015, 04:59 AM
11,772 posts
Thanks: 4,908

Amazon Fire (7")
5th Gen - 2015
KFFOWI | Ford
[INDEX]
Helpful Information

Welcome, this thread has been created to encompass everything you may need for your device. Please only post in this thread with feedback on how to improve this document. Do not post "Thank you" type responses. If you have additional questions or require more help, Please post in [Help Thread][Amazon Fire] Ask Any Question, Noob Friendly, try to find an existing thread, or create your own thread in the Q&A forum. Do not use this as a general help thread.

Click on a Link below for More Info
- Manufacturer's Device Home Page

Kindle Fire 70
Replace the OS on Amazon's budget device to get a fully functional Linux tablet for under £50. Massive win!

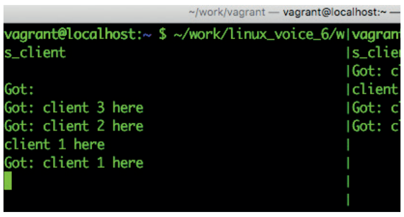
Raspberry Pi and RFID 74
Learn all about RFID key fobs (or cards) that contain a small aerial and chip that can wirelessly interact with readers.

Docker 78
Package a basic Python Flask application – including all of its dependencies – into a container image using the power of Docker.

Coding

```
import socket
import sys
import subprocess

server = "chat.freenode.net"
channel = "#megamiketest"
nickname = "megamikebot9001"

irc = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
irc.connect((server, 6667))
irc.send("USER "+ nickname +" "+ nickname +" "+ nickname +" :Just testing\n")
irc.send("NICK "+ nickname +"\n")
irc.send("JOIN "+ channel +"\n")

while 1:
        text=irc.recv(2040)
        print text

        if text.find('PING') != -1:
                irc.send('PONG ' + text.split() [1] + '\r\n')

        if text.find(':!uptime') != -1:
                output = subprocess.check_output("uptime", shell=True)
```

Coding: IRC bots 84
Build a robots to provide updates on services running on your machines.



```
~/work/vagrant — vagrant@localhost:~ —
vagrant@localhost:~ $ ~/work/linux_voice_6/w|vagran
s_client                                    |s_clie
                                            |Got: c
Got:                                        |client
Got: client 3 here                          |Got: c
Got: client 2 here                          |Got: c
client 1 here                               |
Got: client 1 here                          |
                                            |
                                            |
```

Coding: Crystal 88
Explore a general-purpose language that has a syntax very similar to that of Ruby.

NEWSANALYSIS

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

Opinion

Not neutrality

The powers that be want to control how we access information – we need to fight back!

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

> Right now any actual or perceived crisis makes politicians think any measure attacking net neutrality is acceptable

An important political debate about technology concerns net neutrality. It's a complicated concept that can be interpreted in several ways. The key question is about if, when and how internet service providers (including mobile) should be permitted to prioritise traffic. Some say that ISPs should be allowed to prioritise traffic according to their business priorities, creating for-fee fast lanes for certain kinds of traffic. Others say this prioritisiation should be limited to certain classes of traffic, such as live video streams or emergency information, and that everything else should be treated the same. Others assert that every kind of traffic should get great service but some kinds of traffic, such as partner content, should get even better service (yes, a marketing respin of an earlier argument!). Some say no traffic should ever be artificially degraded or enhanced.

Overlaid on all these arguments are questions of capacity management, bandwidth caps, national interest, copyright licensing and more. It is a profoundly complex space, and I have heard parties supporting every one of those classes of argument claim to be the ones supporting "net neutrality" according to a particulat definition of the term.

The argument is important, but misses the most crucial dimension. Ultimately, the net neutrality debate serves corporate, not citizen interests. There's plenty of discussion of preferential access to connection speeds and interception of traffic for commercial purposes. But there's a dimension of net neutrality that's far more significant.

Cui bono?

Would it be OK for a commercial provider to intercept traffic, analyse it and insert advertising for a commercial product at the behest of a client? If not, why is it OK for a commercial provider to intercept traffic, analyse it and insert different content at the behest of a government agency? Would it be OK for a commercial provider to identify all DNS requests and resolve them with the address of the service most likely to pay them fees for the referral? If not, why would it be OK for an ISP to intercept DNS requests and refuse to resolve addresses alleged to infringe a certain political or moral norm?

That's exactly what mandatory filtering is about. Describing it as "for your benefit and protection" is no different to the language used to describe advertising – "we will send you great offers you'll find interesting".

> Describing filtering as "for your benefit and protection" is no different to the language used to describe advertising

There are no circumstances when it's OK for the traffic passed to me to differ from what was sent to me by the information source at my request? Any interception and manipulation is a breach of net neutrality, and even if it's permitted it should only be as an opt-in case.

Why does the net neutrality discussion not consider the kind that is actually critical to democracy rather than just arbitration between commercial competitors? Part of the problem may be that any policy still has to respect the need for court orders and the like. That's reasonable, but a proper policy would raise a high threshold of proof and need for actions leading to loss of net neutrality. It would offer notification that interception had occurred and offer recourse to have it removed. It would allow use of unfiltered sources, such as Google's 8.8.8.8 DNS, by those willing to accept the risk of legal consequences themselves.

Right now any actual or perceived crisis makes politicians think any measure attacking net neutrality is acceptable, as there are no consequences for them of the loss of freedom they cause all citizens (and not just the ones they want to target). A net neutrality policy has to make it costly to attack net freedoms. It's time to push back against mandatory filters, DNS interception, Internet Connection Records and all the other ill-advised mechanisms our governments are proposing. If net neutrality is a principle that's good enough for commercial interests, it's one that applies to citizen engagements as well.

10,000,000 Pis • Adobe Flash • Vim 8 • Powershell • PC-BSD • LibreOffice

CATCHUP

Summarised: the biggest news stories from the last month

1 Raspberry Pi breaks 10m sales point

Even with our mighty Linux Voice crystal ball, we never would have guessed that the Pi would become so staggeringly popular. Sure, from the start it had all the right ingredients for a great little hacking machine – cheap, silent, no moving parts, and running GNU/Linux – but ten million sales is just mind-boggling. Of course, this continues the Pi's success as the most popular British computer ever made. Now, if only the Pi Zero were a bit easier to get hold of...

2 Adobe Flash lives on in Linux... sort-of

A few years ago, Adobe decided to stop shipping standalone Flash packages for Linux, but now the company has gone back on this (not that anybody missed it): "In the past, we communicated that NPAPI Linux releases would stop in 2017. This is no longer the case, and once we have performed sufficient testing and received community feedback, we will release both NPAPI and PPAPi Linux builds with their major version numbers in sync." **http://tinyurl.com/j8mp6wr**.

3 PC-BSD evolves into TrueOS

PC-BSD was an effort to make FreeBSD more palatable on the desktop, offering a shiny graphical installer and super-simple package management system. But it's no more – future releases will appear under the TrueOS brand. See **www.trueos.org** for all the details.

4 Vim 8 released, after 10 years in development

In recent years, progress on the *Vim* text editor/cult/way of life appeared to have stalled. The codebase was seen as old and full of cruft, while the new *Neovim* fork was doing all the cool work and removing ancient code. Many expected *Neovim* to become the "standard", but it encouraged the *Vim* team to do major work on their own codebase, so now we have *Vim* 8. See **www.vim.org** for a link to the full announcement, and of course the downloads for various platforms.

5 92.3% of kernel developers are professionals

Back in the 1990s, people laughed if you said there was money to be made in developing free software. But a recent statistic has shown that, there's plenty of dough in FOSS. Over 92% of kernel developers are paid for their contributions – ie they work for companies that base products or services on Linux, such as Red Hat. Only 7.7% are part-time hobbyist contributors. So much for Linux being "anti-American", eh Microsoft?

6 PowerShell gets open sourced and runs on Linux

Talking about Microsoft, how about this for news: the company's command line shell has now been made open source and runs on Linux. That's right, if you want to administer your Debian box the Microsoft way (for whatever reason), it's possible. In a blog post, Jeffrey Snover from the company talks about how Microsoft really does love Linux, and this is another example. Read the full announcement on the Microsoft Azure blog at **http://tinyurl.com/hvocafu**.

7 LibreOffice Conference held in Brno

Every year, the *LibreOffice* community gets together to discuss progress in the project and plan new features. This year the event took place in Brno, Czech Republic, with over 100 attendees from around the globe. Many talks were given on development, marketing, translations, UI design and other aspects of the project.

A full report will be posted on **http://blog.documentfoundation.org** – keep an eye on it for news of upcoming *LibreOffice* releases as well.

BRNO 2016 CONFERENCE

8 linuxscreenshots.org closes its doors

Want to see how Mageia has been looking in recent releases? Struggling to remember the theme used by Ubuntu 6.10? One great internet resource for finding screenshots of Linux distribution screenshots was the aptly named **linuxscreenshots.org**. Sadly, this site has announced that it's closing its doors, but the maintainers have made all images freely available. Be warned, though – if you want to download them all, you need almost 50GB of space on your hard drive!

DISTROHOPPER

What's hot and happening in the world of Linux distros (and BSD!).

Bedrock Linux

Addressing distro fragmentation.

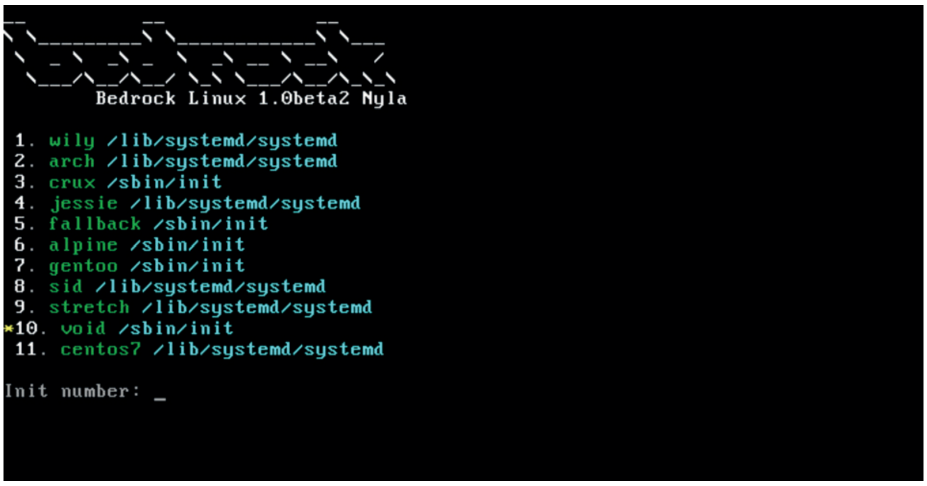
Want a bit Arch, a bit of Ubuntu and a bit of Gentoo in one distro? That's what Bedrock Linux aims to do, not through new package managers or virtual machines, but rather through the use of a virtual filesystem arrangement which enables the different distros' programs to be installed and run alongside one another. The distro is still highly experimental and in heavy development, and if its aims weren't already ambitious enough, Bedrock is written mostly from scratch, not based on any existing distro.

Currently, while taming this multi-distro distro is possible, it's no easy feat. Installing the distro is something that should not be attempted by regular users, with the process involving the compilation of the Bedrock userland from scratch, then installing other distros individually. You can check out the install instructions here: **tinyurl.com/jjlupgj**.

Nevertheless, the project clearly states that it's "not intended as an academic exercise or a purely research project," but rather to solve real issues further down the line when it has matured. The solution has significant advantages over using VMs given the lower overhead from distros interacting directly, also overcoming some of the complexities of using container layers.

With the likes of Snappy, things are moving in a direction where containers may soon have a similar behaviour to that provided by Bedrock, but the solution it provides is still not a wasted effort.

Distro fragmentation has been with Linux almost since its inception, and a variety of solutions have been proposed. With free software adopting and forking different projects, even if the Bedrock solution does not become widespread, it's likely that it will crop up in one form or another down the line.



```
                    Bedrock Linux 1.0beta2 Nyla

 1. wily /lib/systemd/systemd
 2. arch /lib/systemd/systemd
 3. crux /sbin/init
 4. jessie /lib/systemd/systemd
 5. fallback /sbin/init
 6. alpine /sbin/init
 7. gentoo /sbin/init
 8. sid /lib/systemd/systemd
 9. stretch /lib/systemd/systemd
*10. void /sbin/init
 11. centos7 /lib/systemd/systemd

Init number: _
```

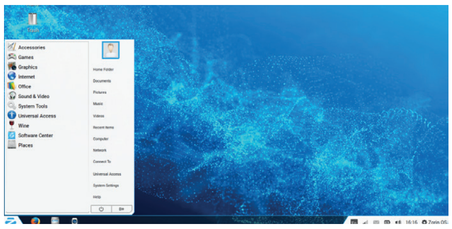
The bedrock one-distro solution to distro fragmentation has advantages, but it's a bit messy.

Zorin OS

Making a big splash.

This distro has been one of Distrowatch's fastest climbers in recent months, having reached 8th place on the website. So what's all the fuss about? The goal of Zorin is to appeal to Windows and OS X users considering jumping ship. Built atop an Ubuntu base with KDE Plasma providing a "familiar Windows-7 like interface," Zorin really goes out of its way to be inviting to potential Linux converts. As such, it does some big FOSS no-nos such as including proprietary software by default, but then there's also little chance of an entry-level newcomer knowing what a "proprietary multimedia codec" is. Not understanding why Flash media isn't playing on the browser would be enough to make many everyday users dismiss Linux before they even give it a chance, so this seems like a good decision.

Among Zorin's exclusive software is a nice theme and layout manager which enables easy switching between more familiar styles like "Windows XP," "Mac OS X" and "Windows 7." The main problem with "Windows transition" distros is that while there is a great FOSS alternative to everything, expecting an entry-level user to ditch the likes of *Photoshop* in favour of learning *Gimp* from scratch seems unrealistic, especially when most tend to have at least one such "must have" application. That said, Zorin does a lot better than most.



Zorin is appealing to existing Linux users, though the *Compiz* effects are a bit silly.

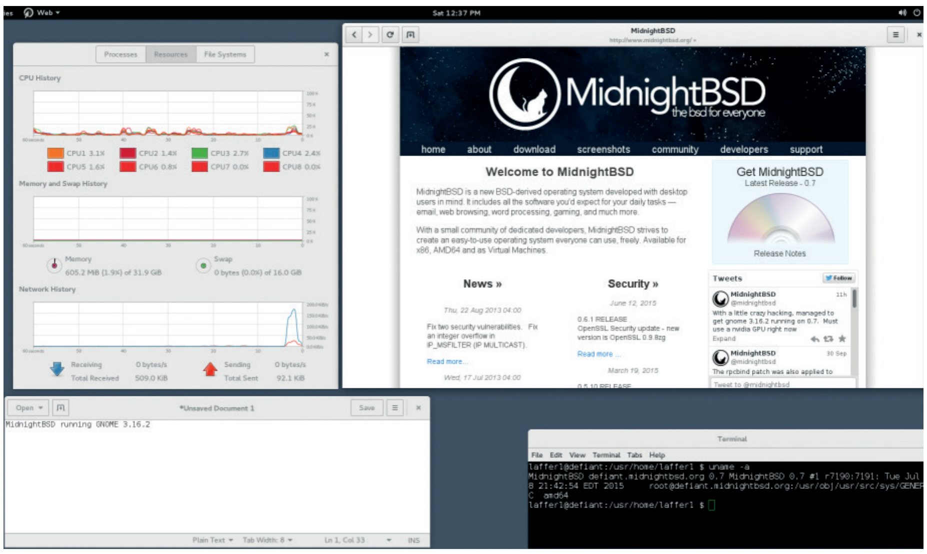
News from the *BSD camps

What's going on in the world of FreeBSD, NetBSD and OpenBSD.

With a slew of bugfixes, FreeBSD has pushed out its first Release Candidate for the much anticipated version 11. On the less positive side of things, FreeBSD experienced a number of potential security vulnerabilities in freebsd-update, portsnap, libarchive, and bspatch after an anonymous post on GitHub made the developers aware of the exploits. These are being ironed out with some patches already available, and the team is recommending that users apply them manually if they don't feel comfortable using the affected freebsd-update.

Also in the FreeBSD space, PC-BSD's Lumina Desktop Environment has hit version 1.0 after four years of development. The lightweight plugin-based desktop environment is also available for other BSDs, as well as Linux. Likewise, the FreeBSD fork MidnightBSD has released version 0.8, marking the ninth release of the system, which has been around since 2006.

DragonFly BSD has seen the release of version 4.6, bringing many new changes. These include updates to the i915 driver to bring it up to the version found on Linux 4.4, adding improved stability for the Broadwell and Skylake architectures in particular.



MidnightBSD is a new BSD-derived operating system developed with desktop users in mind. It includes all the software you'd expect for your daily tasks — email, web browsing, word processing, gaming, and much more.

With a small community of dedicated developers, MidnightBSD strives to create an easy-to-use operating system everyone can use, freely. Available for x86, AMD64 and as Virtual Machines.

```
MidnightBSD running GNOME 3.16.2
```

```
lafferl@defiant:/usr/home/lafferl $ uname -a
MidnightBSD defiant.midnightbsd.org 0.7 MidnightBSD 0.7 #1 r7190:7191: Tue Jul
8 21:42:54 EDT 2015     root@defiant.midnightbsd.org:/usr/obj/usr/src/sys/GENER
C  amd64
lafferl@defiant:/usr/home/lafferl $
```

MidnightBSD aims to focus on usability and stability to appeal more to beginners.

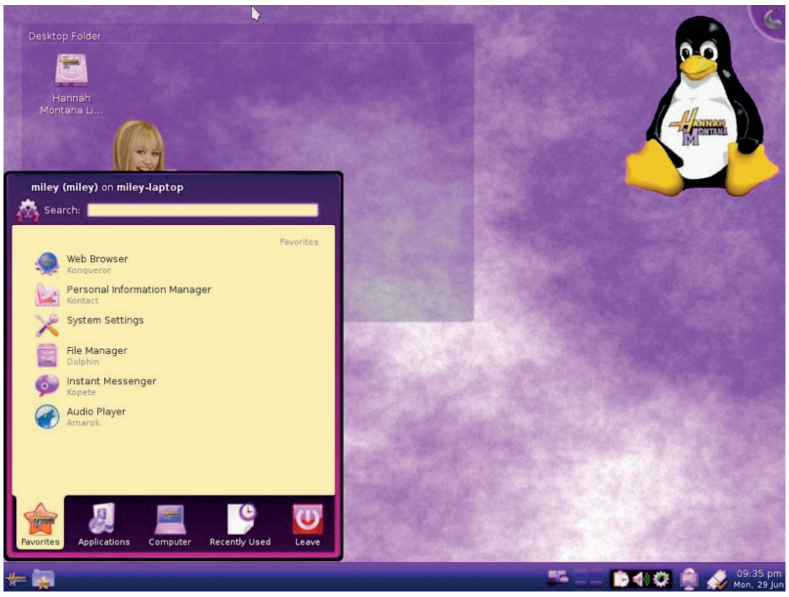
Accelerated video has been improved on both Intel and AMD, with the Radeon driver brought up to the version found on Linux 3.18. Other changes include improved SMP performance and better networking performance under heavy loads. The last bits of Linux emulation have now gone from DragonFly BSD, clearing out some 15–20 thousand lines of code which apparently not many people used. In the OpenBSD camp, the Enlightenment desktop environment has been ported. While there are still a few bugs to work out, most of the work has been done and OpenBSD users are able to install the jazzy user interface. The system has also disabled the tmpfs filesystem due to lack of maintenance, only three years after it arrived on OpenBSD.

Satirical Linux distros

Linux is mostly used for pretty serious stuff, from doing your everyday work to calculating the likelihood of asteroids crashing into earth on supercomputers. However, it can also be used to do very silly things due to the sheer number of distros out there. Not many of these are in active development, but their legacy still has the power to make us laugh (or roll our eyes).

Hannah Montana Linux (basically Ubuntu 9.04, but purple – way ahead of Canonical there) is pretty well known among the Linux community, but there's also a number of other distros like this. Rebecca Black OS (named after the star of the 2011 viral video 'Friday') is also pretty well known, but it may surprise many to know that it's still in active development and was one of the first distributions to include a Wayland server. There's also Biebian, whose project tagline is "It may be Justin Bieber Linux, but it still beats Windows and Mac" – part of the joke is that it's not even based on Debian, but rather Puppy Linux.

Aside from Rebecca Black OS, none of these really provide anything other than some questionable aesthetic choices. However, one fascinating case is Suicide Linux, which plays a kind of twisted game with the user. It uses the *Bash* autocorrect feature (which normally resolves incorrect commands) to automatically resolve any incorrect command to **rm -rf /** thus wiping the user's drive. Awesome. The aim of the game is thus to see how long you can go before you lose absolutely everything.



Joke Linux distros – we tell you about them so you never have to install them.

YOUR LETTERS

Got an idea for the magazine? Or a great discovery? Email us: letters@linuxvoice.com

STAR LETTER

SOME POINTERS

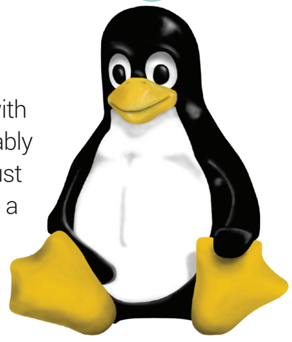
I was reading your four selling points for Linux in the Linux newbie guide, but in my opinion some of them are no selling points at all. Let's look at them one by one.

1 **It saves you money** Not really. If you buy a new PC, you also get a configured operating system. So if you install Linux, or just use the OS that's already there, there is no difference in cost. Even worse, if you want to install Linux, you might need to find someone else to do it for you, because you probably have to create a bootable USB stick. And I am not sure how easy it is these days to boot your PC from a USB stick, with secure boot and UEFI around. But to be fair: if you want to give an old PC a second life, it is cheaper to use Linux. Because you probably don't have a Windows licence lying around that you can use to do a fresh Windows install.

2 **It's open for everyone** That is a good selling point. You are not stuck with a certain company. If you don't like the support of company A, just go to company B. And if you don't trust companies, you can review the source code yourself, if you want to do that.

3 **It's super reliable** Yeah probably. But I dare to say that other OSes are reliable as well these days. I think today Linux tends to crash more, because hardware suppliers don't care about Linux support; you might end up with dodgy drivers to get your hardware running.

4 **It works with your files** Hmmm. I have to admit that I did not try this for a couple of years, but I'm pretty sure that if you work together on an office document, you with *LibreOffice*, and the other one with *Microsoft Office*, the layout will probably be broken. Unless your document just contains text. So that you could use a plain text file as well. Oh wait, **notepad.exe** still does not recognise Linux line endings, does it? But I am optimistic about this. Give it some more years, and office suites might be obsolete, and replaced by web based tools. In that case, the problem with office documents will be gone. Finally :-)



5 **You forgot a selling point** If you want Linux, you need to do a fresh install. And this way, you don't have crapware. As it happens, I just removed all crapware from a new Windows PC that my wife bought. It took me more than an hour. If I just wiped the OS and installed some Linux distro, it would have saved me a lot of time.

So, these are the points I wanted to make. But I also want to thank you for making this great magazine. I recently upgraded my digital subscription to a paper one; it is really worth it.

johanv

Andrew says: Hmm. We use Linux every day, so we're not blind to its faults. But I think you're being a little harsh there (I agree about the crapware...).

PAGING DR SINITSYN

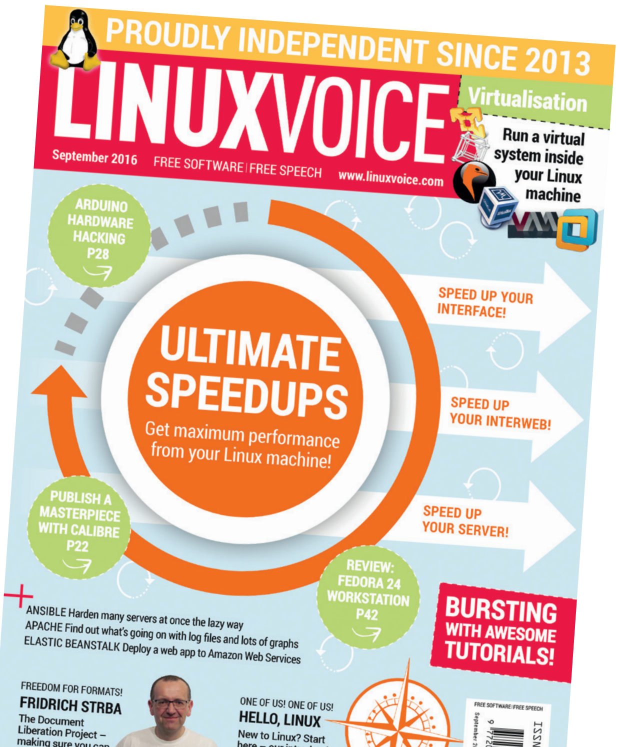
You've probably had loads of people getting in touch to tell you this, but LV030 has a mistake in the pagination – pages 92 and 93 are repeated on pages 94 and 95, so sysadmin is missing the first two pages.

Other than that, it's a top-class issue.

Regards, George Webber

Andrew says: Yes, we did have a fair few emails about that – for anyone who hasn't already seen, here's a link to a PDF of the unmolested Core Tech pages: **www.linuxvoice.com/wp-content/uploads/2016/08/LV30-Core-Tech.pdf**.

We're genuinely baffled as to what went wrong with issue 30. I suspect it was Mike's fault.


PROUDLY INDEPENDENT SINCE 2013
LINUXVOICE
September 2016 FREE SOFTWARE | FREE SPEECH www.linuxvoice.com
Virtualisation
Run a virtual system inside your Linux machine
ARDUINO HARDWARE HACKING P28
ULTIMATE SPEEDUPS
Get maximum performance from your Linux machine!
SPEED UP YOUR INTERFACE!
SPEED UP YOUR INTERWEB!
SPEED UP YOUR SERVER!
PUBLISH A MASTERPIECE WITH CALIBRE P22
REVIEW: FEDORA 24 WORKSTATION P42
ANSIBLE Harden many servers at once the lazy way
APACHE Find out what's going on with log files and lots of graphs
ELASTIC BEANSTALK Deploy a web app to Amazon Web Services
BURSTING WITH AWESOME TUTORIALS!
FREEDOM FOR FORMATS!
FRIDRICH STRBA
The Document Liberation Project
ONE OF US! ONE OF US!
HELLO, LINUX
New to Linux? Start

I LOVE LAMP

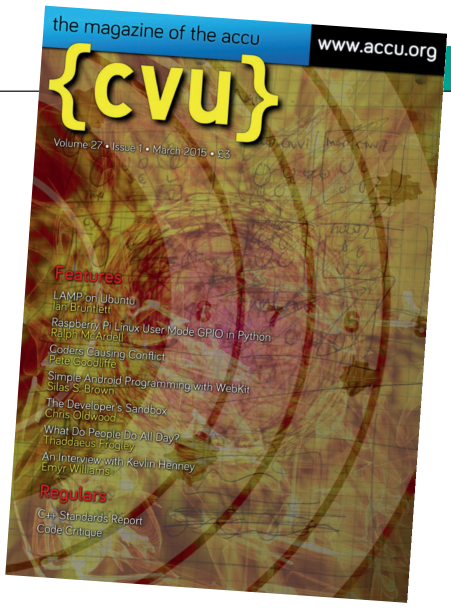
I read Sean Dwyer's letter in LV31's letters page, asking for help with LAMP on Linux.

Well, I'm not an expert but I've dabbled with LAMP on Ubuntu and wrote up my experiences for the ACCU's CVu magazine. I assumed I'd uploaded it to my scribbles website but found out that I'd forgotten to upload it. See the file **CVu-Lamp-on-Ubuntu.pdf** at the bottom of **https://sites.google.com/site/ianbruntlett/home/information-technology**.

I'm currently studying Ruby at the moment. When I wrote that article, PHP 5.x was in use. These days PHP 7.x is in use, something to do with moving to an object-oriented approach.

Ian Bruntlett

Andrew says: Thanks a lot Ian, you've saved us a job there. The journey from dabbler to expert is nowhere near as arduous as that from beginner to dabbler, so I hope Sean heeds and learns from your scribbles. And many thanks to you for pointing out the link to your scribbles!


the magazine of the accu
www.accu.org
{cvu}
Volume 27 • Issue 1 • March 2015 • £3
Features
LAMP on Ubuntu Ian Bruntlett
Raspberry Pi Linux User Mode GPIO in Python Ralph McArdell
Coders Causing Conflict Pete Goodliffe
Simple Android Programming with WebKit Silas S. Brown
The Developer's Sandbox Chris Oldwood
What Do People Do All Day? Thaddaeus Frogley
An Interview with Kevin Henney Emyr Williams
Regulars
C++ Standards Report
Code Critique

ACCU's members get this fine publication.

MY MATE MATE

Thanks for the recent review of Ubuntu Mate, and the interview with the project helmsman this month. I was horrified that you were drinking Red Barrel in a Watney's establishment until I noticed the editor had been punishing a pint of Sharp's Doom Bar.

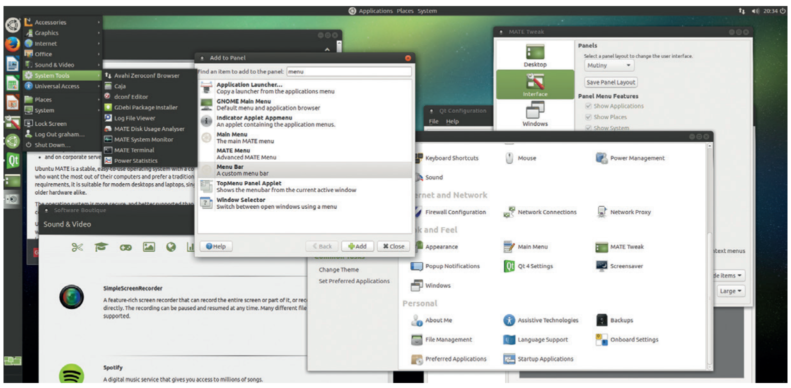
Ubuntu Mate was impressive enough to cause me to send some coin in the direction of the maintainers. It installed nicely on several machines, but baulked when I tried to load it on an Asus eePC 703, complaining that 2GB was insufficient memory.

I upgraded my mentee's netbook with Mate, which he liked better than KDE. He returned to his third year computer engineering at the uni clutching a B-model RPi and a PocketCHIP to go with the refurbished netbook.

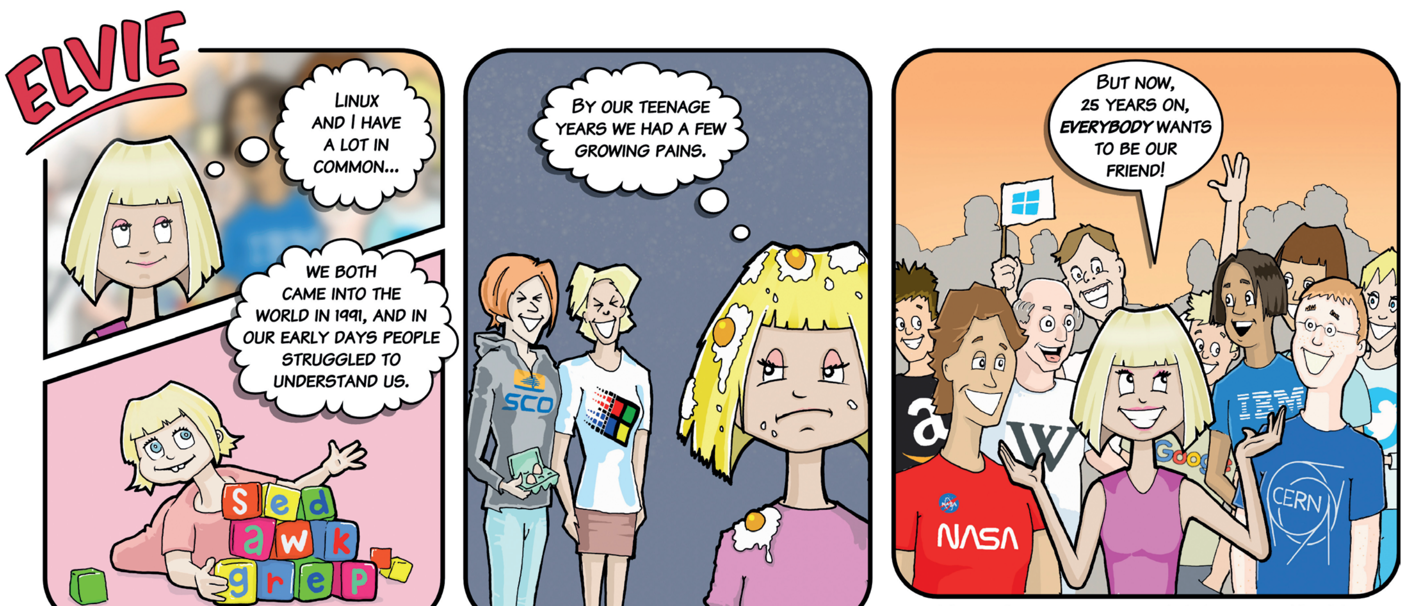
The Raspberry Pi version of Mate works very well on an RPi3 that I got with the intention of donating resources to new projects like MaidSafe and ZeroNet. There's enough on-board electrical power to run the USB keyboard, mouse, and 3TB external disk drive. The Mate installation instructions recommend a fast SD memory card as it shares the USB bus and can become a data bottleneck with slower versions.

Andrew Shead

Andrew says: I think we're all too young to remember Watneys, but if it were put in front of him, I'm sure Ben would drink it regardless.



Mint: better than Watney's Red Barrel (apparently).


ELVIE
LINUX AND I HAVE A LOT IN COMMON...
WE BOTH CAME INTO THE WORLD IN 1991, AND IN OUR EARLY DAYS PEOPLE STRUGGLED TO UNDERSTAND US.
sed awk grep
BY OUR TEENAGE YEARS WE HAD A FEW GROWING PAINS.
SCO
BUT NOW, 25 YEARS ON, EVERYBODY WANTS TO BE OUR FRIEND!
NASA
IBM
CERN

CC-BY-SA WWW.PEPPERTOP.COM

Subscribe

shop.linuxvoice.com

Get your regular dose of **Linux Voice**, the magazine that:

- Gives 50% of its profits back to Free Software
- Licenses its content CC-BY-SA within 9 months

US/Canada subs prices

1-year print & digital: **£95**
12-month digital only: **£38**

Get many pages of tutorials, features, interviews and reviews every month

Access our rapidly growing back-issues archive – all DRM-free and ready to download

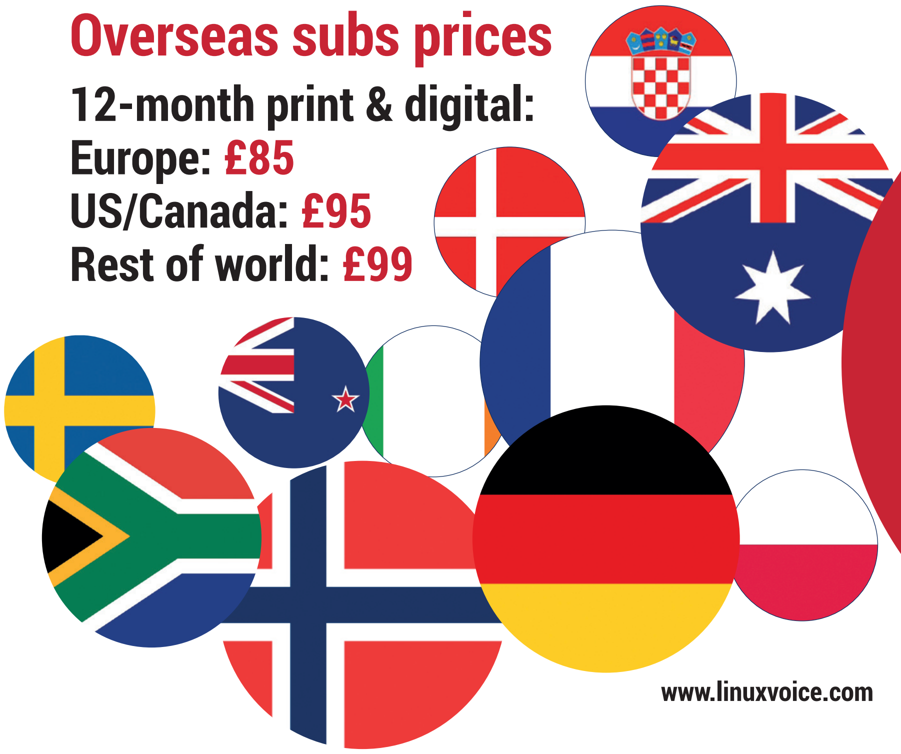
Save money on the shop price and get each issue delivered to your door

Payment is in Pounds Sterling. 12-month subscribers will receive 12 issues of Linux Voice a year. 7-month subscribers will receive 7 issue of Linux Voice. If you are dissatisfied in any way you can write to us to cancel your subscription at subscriptions@linuxvoice.com and we will refund you for all unmailed issues.

All subscribers get access to every single digital back issue – that's about 1,000,000 words of tutorials, reviews and free software hackery at your fingertips

Overseas subs prices

12-month print & digital:
Europe: £85
US/Canada: £95
Rest of world: £99



DIGITAL SUBSCRIPTION*

ONLY £38

*WHEREVER IN THE WORLD YOU ARE – IT'S DIGITAL, SO THERE ARE NO POSTAGE COSTS

PRIVACY ON LINUX

Online privacy isn't a contradiction in terms if you follow **Ben Everard**'s guide.

Privacy is all about having the ability to decide what personal information you want to share, and who you want to share it with. Before the internet, it was you who controlled access to information about you – if you told someone, they knew and if you chose not to, they didn't. That's no longer the case. Personal information is valuable to advertisers and marketers who try to sell more products and to intelligence agencies who are building up profiles of people who have done no wrong.

Unsurprisingly, those who profit from mass invasions of privacy don't see this change as a problem. Google chief executive Eric Schmidt defended the erosion of privacy by saying: "If you have something that you don't want anyone to know, maybe you shouldn't be doing it in the first place." This sentiment is often summed up in the phrase "If you have nothing to hide, you have nothing to fear."

Here at Linux Voice we vehemently disagree with Mr Schmidt. We think that people still have a right to a private life and that information about us is personal property, not just data to be harvested for a profit. Controlling access to personal information isn't just about hiding wrongdoing, but about creating a personal space where we can live our lives without fear of intrusion.

Information about us is personal property, not just data to be harvested for a profit

YOU ARE BEING WATCHED

They lurk behind cookies and web beacons to follow your every step…

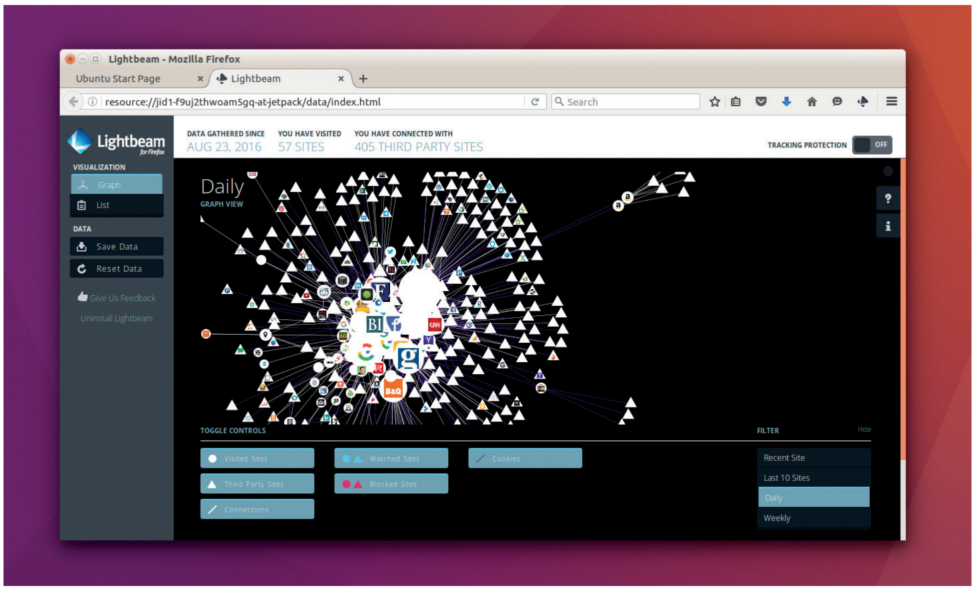
Some of the world's largest companies make money by giving their products away for free. That may sound like an oxymoron, and in a way it is. Let's consider the cases of Google, Facebook and Twitter – they're all hugely valuable global companies that have made many millionaires and a few billionaires, yet almost everyone who uses them does so without paying a penny. This is all possible because these companies use the data you give them to target adverts at you. Google builds up a profile of you based on your web browsing, while Facebook and Twitter work out what products you're likely to buy based on what you post online. Through their free services, they've become far more efficient at parting people with their money than traditional marketers that don't spy on people.

It's not just search engines and social networks – there are many little-known companies that provide services to web hosts in return for placing tracking code on their sites. These services then build up huge amounts of data as they follow us around the web, and this data is then sold to anyone willing to pay.

Somebody's watching me

You can get an idea of just how prevalent this tracking is by using the Lightbeam plugin with *Firefox*. This plugin keeps track of all the different services that monitor you via the websites you visit. In a simple test, we visited eight popular websites and were tracked by 161 different services.

The fundamental problem is that this data is intensely personal. Someone looking at our browsing could reliably deduce our medical history, mental state, sexual preferences (which can have legal repercussions in many countries), and much more that we consider deeply private. Once these trackers have sucked up your personal data, you have no control over what it's used for, how it's processed or who it's sold to. Your most personal details become a tradeable commodity that can be aquired by anyone who can think of a way to monetise them.



The Lightbeam add-on for *Firefox* reveals the truly scary number of web trackers out there.

Your internet service provider is also amassing a trove of information about what you do online. This information is ostensibly to help the government track the most dangerous criminals; however, the safeguards in place in most countries are not enough to prevent abuse.

As well as government agencies overstepping their remit, the system provides an attractive target for hackers. In late 2015, hackers broke into internet provider TalkTalk's data centre and stole data from over 150,000 users. It's only a matter of time before an ISP falls victim to hackers again and the more information they store about their customers, the more there is to steal.

All this is going on right now with little or no oversight. If you want to keep any semblance of privacy, you can't expect it to happen automatically. Unless you take steps to protect your privacy, you won't have any.

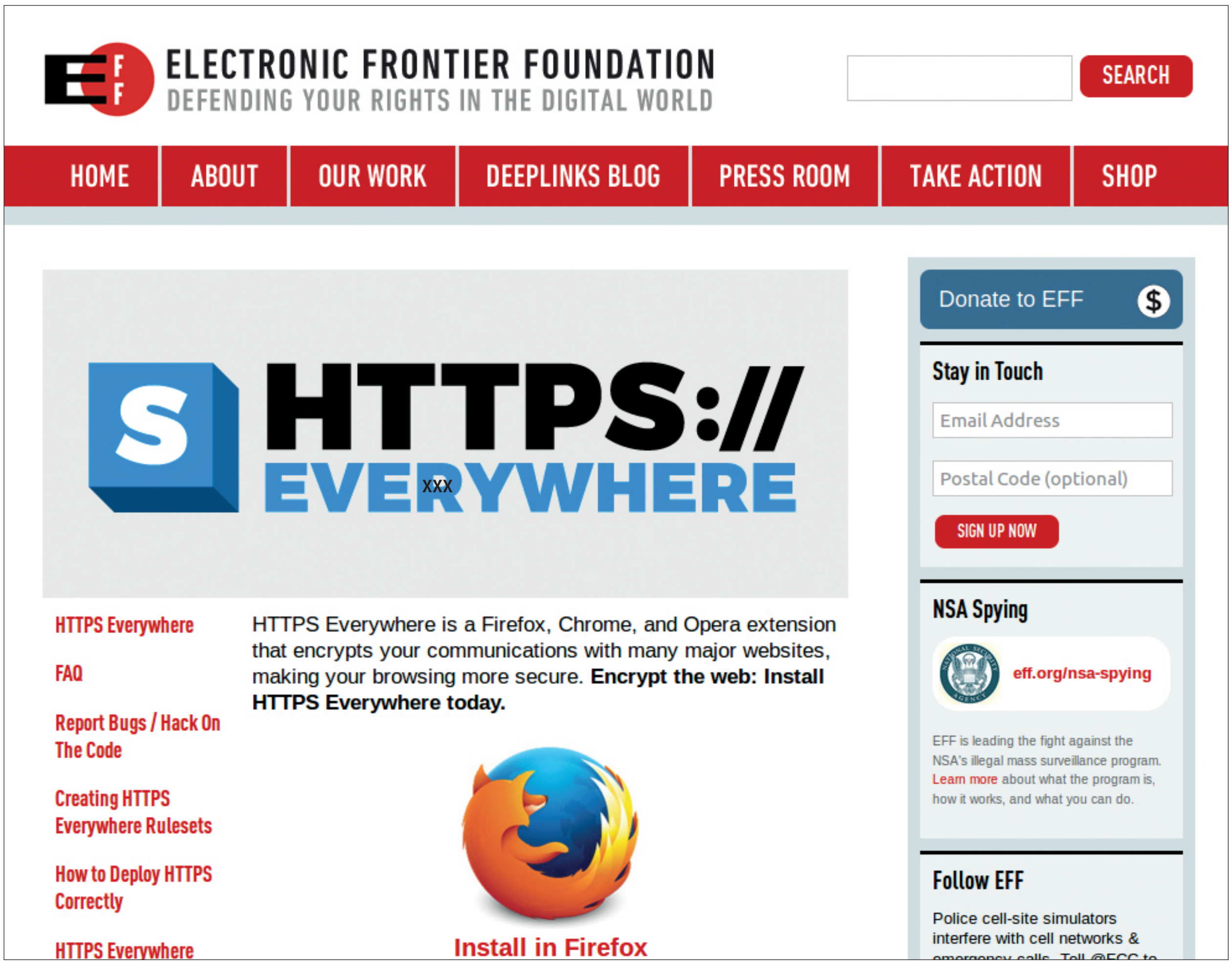
It's only a matter of time before an ISP falls victim to hackers again

SOCIAL NETWORKS

Even if you care about your privacy, it can be tempting to use social networks to stay in contact with people and to find out about social events. The golden rule is to assume that whatever you put on a social network can be seen by the entire world: your parents, children, employer, future girlfriend or boyfriend and anyone else who might take a passing interest in you. Despite the fact that most social networks have privacy settings, they're often obfuscated and prone to change with little or no warning, so it's not prudent to rely on them. If you would be prepared to announce the information loudly in a public place in front of your friends and family, it's fine to post online. If the thought of everyone you know finding something out makes you feel uncomfortable, save it for more private methods of sharing.

WEB PRIVACY

Three options to regain a little personal space.



HTTPS Everywhere is a Firefox, Chrome, and Opera extension that encrypts your communications with many major websites, making your browsing more secure. **Encrypt the web: Install HTTPS Everywhere today.**

EFF is leading the fight against the NSA's illegal mass surveillance program. Learn more about what the program is, how it works, and what you can do.

Police cell-site simulators interfere with cell networks & emergency calls. Tell @FCC to

The HTTPSEverywhere plugin is a simple way to increase your privacy without any negative impact on the rest of your browsing.

Privacy online really comes down to a couple of things: protecting your IP address and stopping tracking between websites. Tracking between websites will be well known to most web users. For example, if you look at a product on a web shop, it's common to then see adverts for this product on other sites you visit. This is an obvious case, but there are far more situations where online advertisers are constantly monitoring your web surfing to try and build up a profile of who you are to better target adverts at you. Intelligence agencies also perform similar actions and use your web browsing to build up profiles about global citizens. Visiting the website of some Linux magazines is enough to get you labelled as an extremist by the NSA!

There's no way of getting around the fact that if you want any semblance of privacy online you need to block adverts. Almost the entire online advertising industry is built around tracking users to find out as much information as possible in order to serve the most appropriate adverts. This is performed by advertising companies placing code in other websites that constantly alert the ad server of websites you visit.

Ad blocking

The simplest thing that you can do to protect your privacy is block adverts (and other tracking data used by advertisers). You can do this in most web browsers using plugins or

CENSORSHIP RESILIENCE

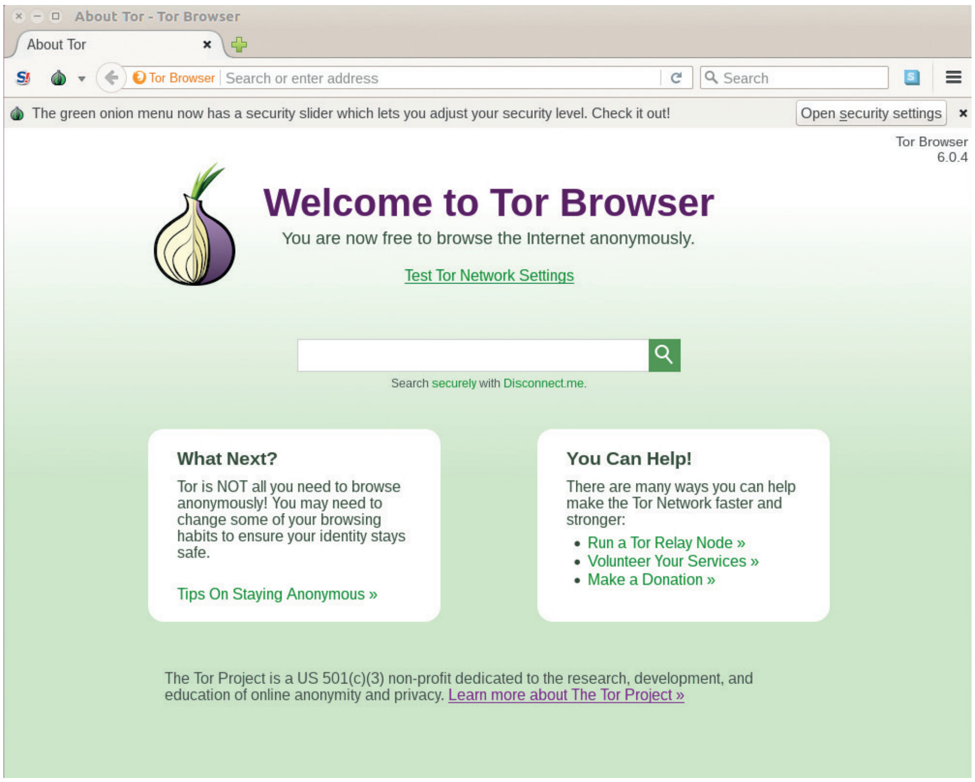
Privacy protection goes hand-in-hand with overcoming online censorship. The Tor Network enables users to get around their country's web blockers as a side effect of masking IP addresses. This is obviously useful in states that heavily censor the web, but can also be necessary elsewhere. In the course of producing this magazine we've needed to use Tor to bypass the UK government's block on Torrent sites to access downloads of unusual Linux distributions and investigate download statistics of different pieces of software.

extensions (see boxout), but perhaps the simplest way of blocking adverts is to use a web browser that does this by default: *Brave*. Within the main menu of this web brower, you can select common-sense and easily understandable options to increase your privacy: Block ads, Block 3rd Party Cookies, Use HTTPS Everywhere and Block Phishing and Malware. Bringing these options into one place makes *Brave* by far the easiest browser to set up for privacy. As an added advantage, these options will speed up your web browsing, as adverts can contribute to a significant proportion of web page load time.

Blocking adverts will stop the most egregious affronts on your online privacy, but it's not perfect. The ad blocker is unlikely to be 100% effective, and it doesn't do anything to stop people intruding on your privacy at a network level. For example, anyone on your local network can see which web servers you're visiting even if you use HTTPS encryption. If you don't use HTTPS, then they can also see all the details of whatever pages you're viewing. Your internet provider can also see this information, and in many countries, they're required to store this and pass it on to the government.

There is one browser that really stands out as the best option for getting a high degree of online privacy: *The Tor Browser Bundle*. This is a single download that contains everything you need to connect to the Tor Network and run a private web browser. The Tor network keeps your physical location secret by passing your data through three separate proxy servers before it gets to the public web. There are three layers of encryption (one for each proxy server) so that only the first proxy knows who you are (but not what you're sending to the web), the second proxy only knows who the other proxies are (but not who you are or what you're sending to the web), and the third proxy knows who the second proxy is but not who you are or who the first proxy is.

This network design means that even if one of the proxies is spying on you, there's no way for it to work out both who you are and what you're doing online.


About Tor - Tor Browser

The green onion menu now has a security slider which lets you adjust your security level. Check it out!

Tor Browser 6.0.4

Welcome to Tor Browser

You are now free to browse the Internet anonymously.

Test Tor Network Settings

Search securely with Disconnect.me.

What Next?

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

Tips On Staying Anonymous »

You Can Help!

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

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

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

When you first start the Tor Browser, it will go green to confirm that you're correctly connected to the anonymising network

The Tor network keeps your physical location secret

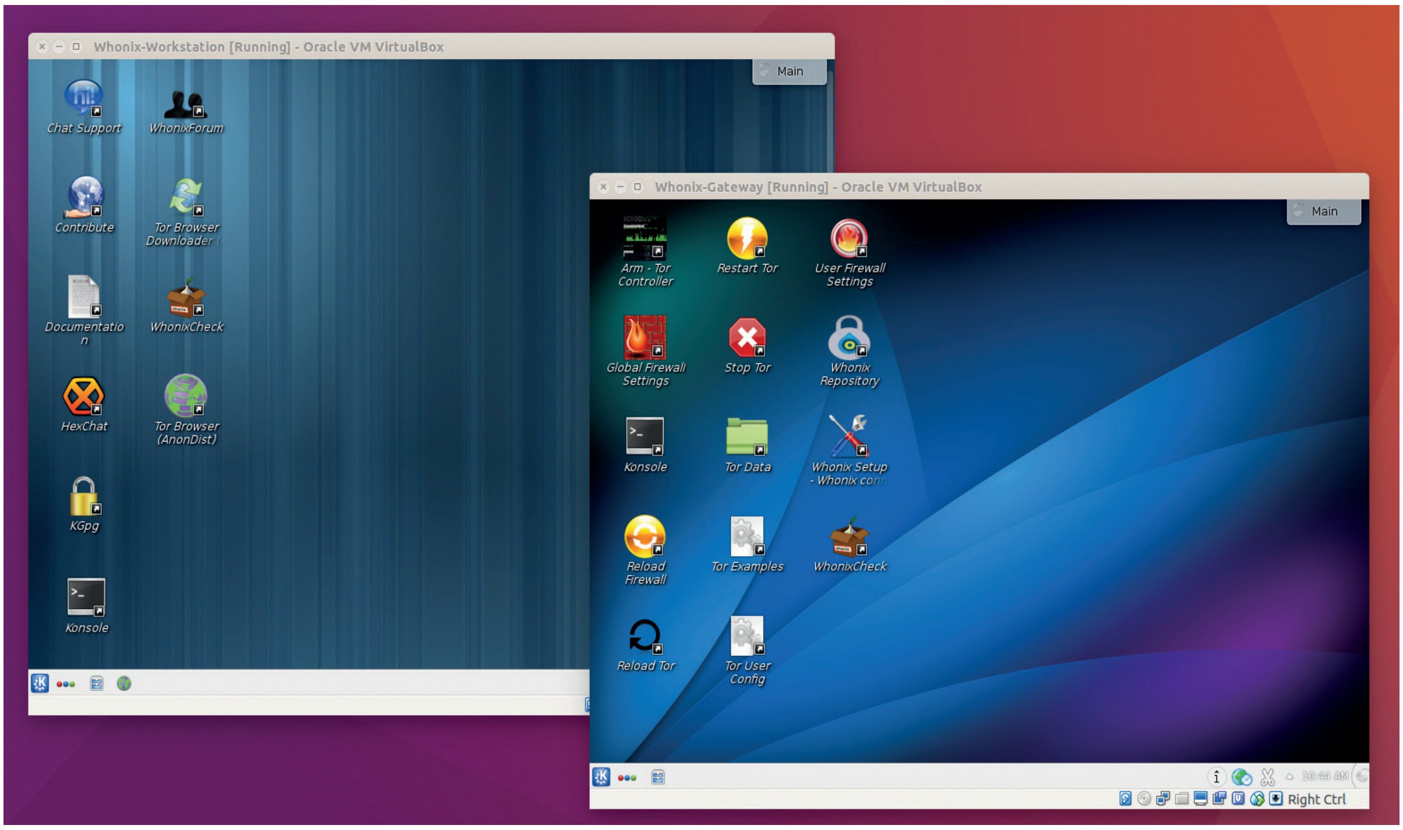
BUILD YOUR OWN PRIVATE BROWSER

There's no perfect solution to online privacy, but you can customise your browser to make the right trade-offs that increase your privacy at the expense of decreasing the amount you can do online. First you'll need to start with either *Firefox* or *Chromium* (*Chome* is too integrated into Google to make this really possible), then consider the following changes:

- **Ad blocker** It's absolutely essential to block ads if you want online privacy. There are several options for all browsers, but not all are equal, because some ad blockers themselves spy on you and sell that data to advertisers. We recommend uBlock (available for *Firefox* and *Chromium*) as it's open source and runs efficiently in our experience.
- **Block cookies** Cookies are bits of data that web servers can store on your computer and retrieve at a later time. There are several legitimate uses for cookies (such as websites remembering your login sessions), but they're also used by web trackers. We recommend blocking third-party cookies, as this only allows cookies from the domain of the website you're on and not any tracking code from third party services. If you want a stronger privacy protection, block all cookies. This is possible in the Options menu of all major web browsers.
- **HTTPS Everywhere** Anything you send over plain HTTP is readable by anyone on the same network as you, and by your ISP. HTTPS encrypts this and makes the situation a little better (though the web server you're visiting is still visible). Some sites are available in both HTTPS and HTTP, and the HTTPS Everywhere extension ensures your browser always uses the HTTPS version where possible.
- **Block scripts** JavaScipt allows websites to run bits of code in your web browser. Mostly this is harmless, and indeed can make browsing the web a more interactive experience. However, some scripts try to maliciously identify your browsing. The NoScript browser extension enables you to control which sites can run scripts and which ones can't. Be aware that some sites won't work without scripts (but you can enable them on a per-site basis).

PRIVACY DISTROS

Preconfigured to keep you safe.



The two Whonix virtual machines (here running in *VirtualBox*) enable you to keep the configuration of Tor separate from the software you want to run.

Using a private web browser will help you stay safe online, but, it isn't a guarantee of ultimate privacy. If you want to make sure that you aren't monitored when using the web, you need a distro that's built for privacy.

The Anonymous Incognito Live System (Tails) is a distro that's designed entirely to help you stay safe by using the Tor Network (see previous page). Tails is used by journalists, activists and other people persecuted around the world. In many countries, it's actively under attack by the regimes from which it seeks to provide privacy, and protection from this is built into the way it works. This protection starts with the moment you try to download the distro. Rather than just having a file that you save to disk, downloading Tails requires you to either install a *Firefox* add-on that will verify that the file has downloaded correctly, or use the BitTorrent download (which will automatically verify the integrity of the file). This way it's much harder for an intrusive regime to intercept the download and replace it with one that doesn't provide the same level of privacy. While this may seem extreme in many western countries, it's a sensible precaution in some parts of the world. Even when it does seem over the top, it's prudent to have more security than you need.

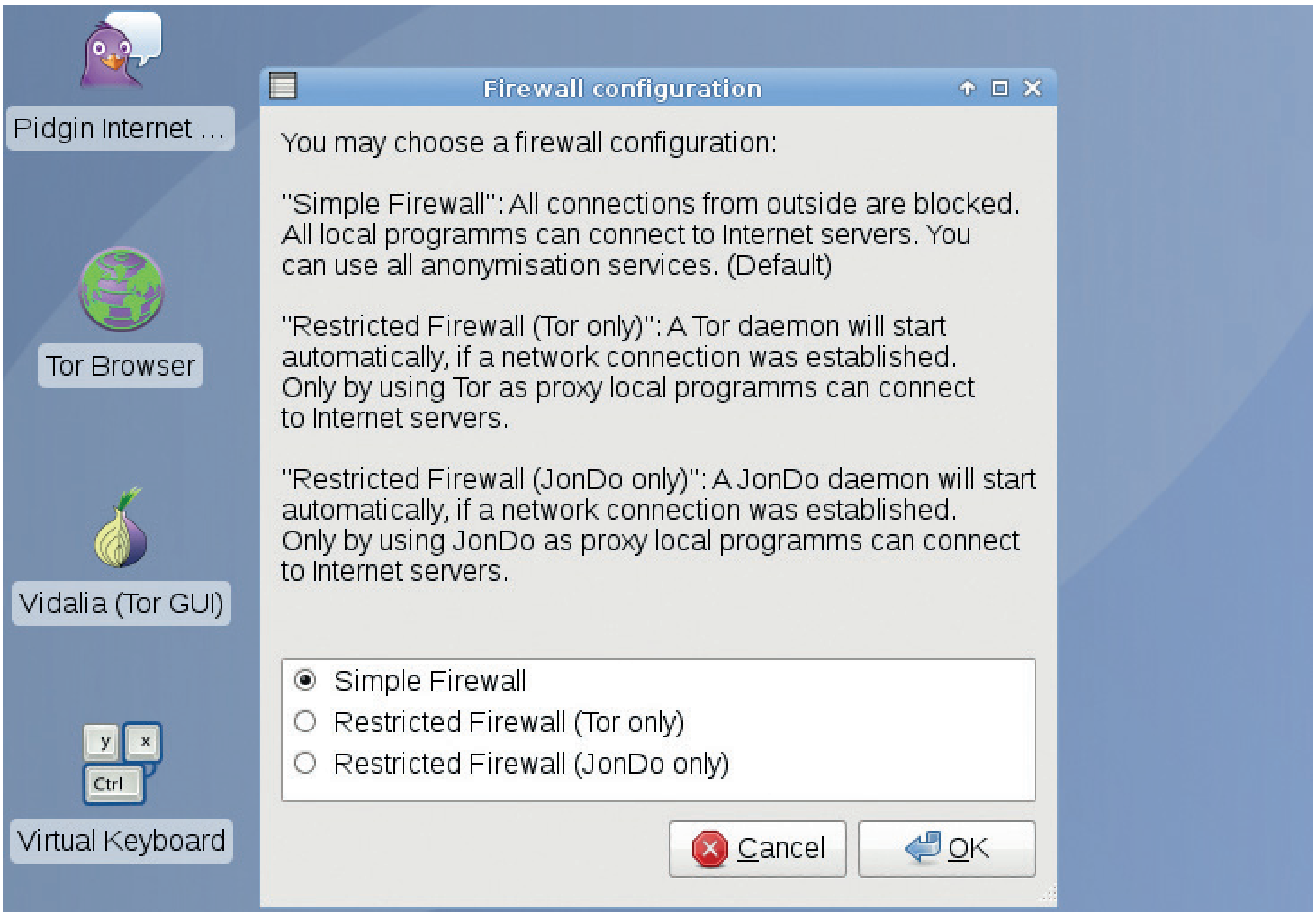
Tails: the anonymising distro

As the name suggests, Tails is a live system. When you boot it, you're presented with a series of options that enable you to set up the system for the circumstances you need. You can spoof the MAC address (as this could be used to identify you), or use one of the specialised methods of connecting to the Tor Network.

Tails hides anything that doesn't need the user's attention. If you follow the recommended boot options, you'll be connected to the Tor Network and have access to the web via the *Tor Browser*. There are a few more applications that you may need if you're trying to stay private online, including a Bitcoin wallet, The IceDove email client (and the OpenPGP applet for encrypting emails both here and in the browser) and Pidgen (with support for OTR).

Basically, if you want the simplest way of setting up a secure computer with Tor, then Tails is the best bet. The biggest problem with Tails is that it's designed to be locked down to a single way of working. If you want more flexibility (which means you have to have the knowledge to protect yourself), then there are a couple of other options you could consider.

If you need to ensure your privacy from a powerful adversary, it's important to ensure that all your network traffic goes through an anonymising network such as Tor. It can be a little tricky to ensure this as Linux (or any other popular operating system) isn't really designed to force all traffic to go through a proxy. Instead, you have to make sure that



You may choose a firewall configuration:

"Simple Firewall": All connections from outside are blocked. All local programms can connect to Internet servers. You can use all anonymisation services. (Default)

"Restricted Firewall (Tor only)": A Tor daemon will start automatically, if a network connection was established. Only by using Tor as proxy local programms can connect to Internet servers.

"Restricted Firewall (JonDo only)": A JonDo daemon will start automatically, if a network connection was established. Only by using JonDo as proxy local programms can connect to Internet servers.

- [✓ Simple Firewall]
- Restricted Firewall (Tor only)
- Restricted Firewall (JonDo only)

JonDo lets you choose between connecting to the internet directly or via an anonymising network such as Tor or JohnDo.

each individual application is behaving properly. An alternative option is to configure this through your network. The Whonix distribution has two separate parts, a workstation and a gateway. Thanks to virtualisation technology these can both be run on the same physical machine.

The Gateway enables you to configure how you want to connect to the Tor Network (including any firewall rules you want to enforce), and this creates a virtual network to ferry all the traffic into the anonymising network. The Workstation then connects to this virtual network, and so all the network data goes via Tor. In theory, this setup is more robust and enables you to perform more actions on the Workstation without risking disclosing your identity. There is still the Tor browser, because this provides more protection than a regular web browser (thanks to script blockers and other privacy-enhancing features), but even with a normal web browser, you'll be running over Tor. The biggest advantage of Whonix over Tails is that it enables you to use all your standard tools – even if they're scripts you've written yourself – and still get the protection of Tor. The biggest downside to Whonix is that while the theory is sound, it's less well tested than Tails. It's also a little more complex to install, though this shouldn't be a challenge for most moderately technical people.

Both Tails and Whonix are based on the Tor network. This provides the best privacy guarantees, but it is a little slow. If you still want privacy, but need speed, the JonDo network may be a better option. Like Tor, this relies of forwarding your data through a series of proxies that obfuscate where the data came from. Unlike Tor, JonDo is a commercial platform. The free tier is publicly available but limits your speed to 50Kbps; only lets you use HTTP and HTTPS ports; and only puts your data through two mixer proxies (compared with Tor's three). The premium service removes the bandwidth and ports limits and lets you use up to three proxies.

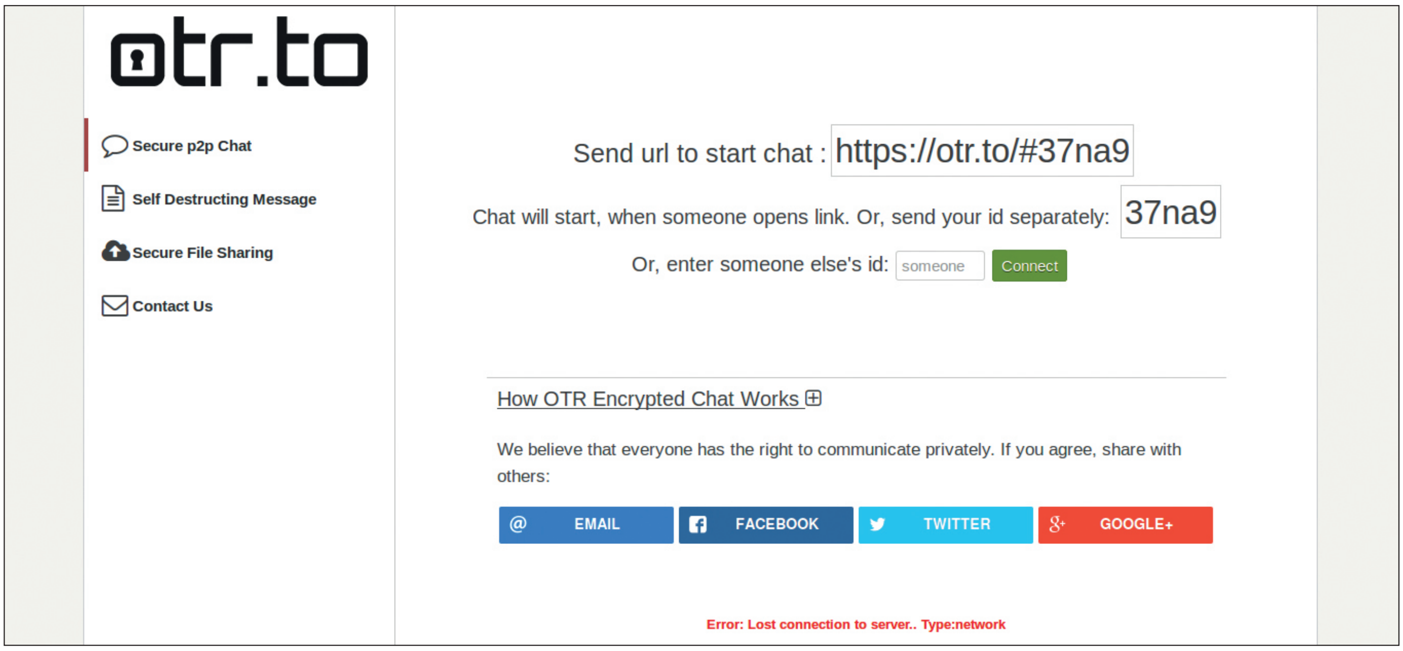
The JonDo network is much smaller than Tor, which would make it easier to fingerprint the traffic, and the Tor protocol has come under more scrutiny, so we're more convinced of its security. That said, JonDo is a viable alternative if you're willing to pay for faster speeds (the free tier, in our opinion, doesn't offer any advantages over Tor).

You can connect to the JonDo network without using the distro, as clients are available for regular Linux distros; however, the live distro offers the best way of trying the service and is a more secure way of using the system.

To simply set up a secure computer with Tor, Tails is the best bet

ONLINE CHAT

Messages are for friends, not snoopers.


otr.to

Secure p2p Chat
Self Destructing Message
Secure File Sharing
Contact Us

Send url to start chat : https://otr.to/#37na9

Chat will start, when someone opens link. Or, send your id separately: 37na9

Or, enter someone else's id: someone Connect

How OTR Encrypted Chat Works ⊞

We believe that everyone has the right to communicate privately. If you agree, share with others:

EMAIL FACEBOOK TWITTER GOOGLE+

Error: Lost connection to server.. Type:network

You can chat securely using *OTR* without installing anything by going to **https://otr.to** and creating a chatroom.

As well as our web browsing, we reveal much about ourselves when we chat online. This was once a niche activity that took place on internet relay chat (IRC) and bulletin boards, but it's increasingly becoming a common way to stay in touch.

There are two key things that we need to know about any messaging system: is it encrypted, and is it end-to-end encrypted. The difference between these is key to understanding the privacy implications of chatting online.

The worst possible case is where nothing is encrypted: you send your messages over the network and anyone can read them, from the people you share your network with to your ISP, to the people running the chat server. Fortunately there aren't many systems like this any more, but it's still something to be aware of if you use an older protocol such as IRC (which can provide encryption but doesn't always). The next best scenario is where the connection between your chat client and the chat server is encrypted. This way, no third party can eavesdrop on your chats, but the chat server can see your conversation as it has to decrypt it to be able to pass it on to the person you're chatting with – this is what most online chat systems do when they claim to be encrypted. Given that many companies providing online chat make money by data-mining their users, it's prudent to assume that your chats are feeding into their information-gathering systems.

End-to-end encryption

The best scenario is where each message is encrypted on your client and can only be decrypted by the person you're chatting with. This is known as end-to-end encryption, and if you wish to retain your online privacy, you should be using a chat system that provides this style of security. For a long time, end-to-end encryption was only available on a few esoteric chat systems that were the preserve of geeks, but recent moves have brought strong privacy to the masses.

Signal can bring best-in-class privacy to non-technical people

ENCRYPTED EMAIL

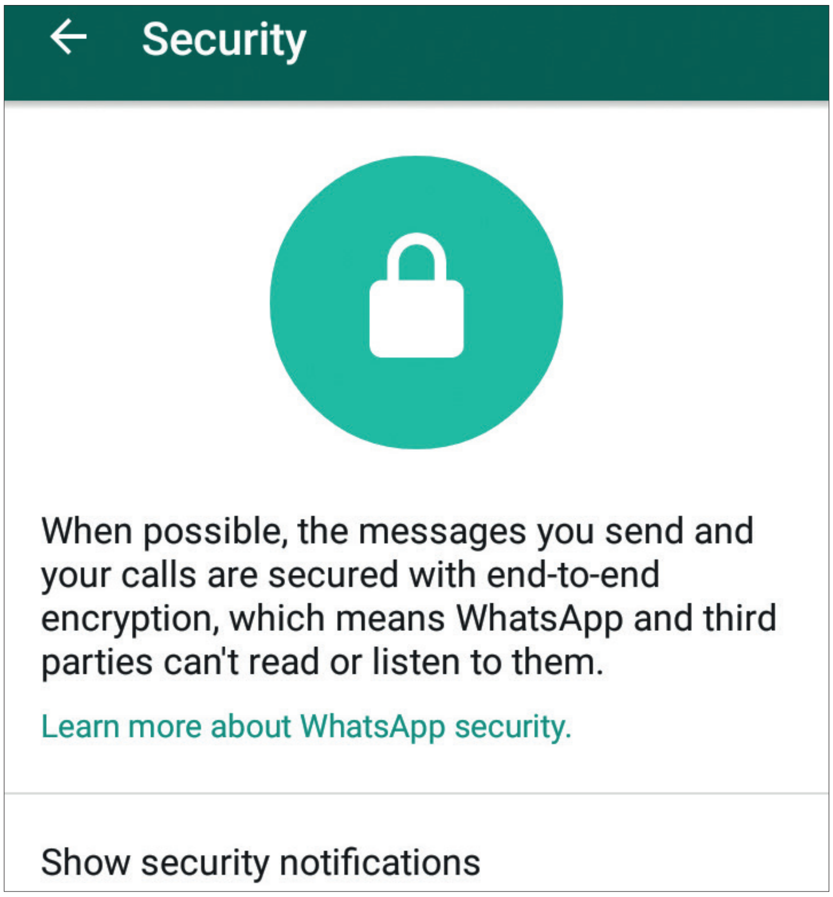
You can use PGP to encrypt and sign emails. PGP is old technology that's never caught on, no doubt in part because it's a little complex to set up. When Edward Snowden first contacted Glenn Greenwald to report on his trove of NSA documents, the reporter was unable to reply because he couldn't work out how to install the necessary software. Fortunately, Laura Poitras was able to help and the documents could change hands in the end.

PGP is an excellent technology that is doomed to a lifetime on the sidelines unless major email providers decide to support it and enable it by default. Given that several major email providers make the majority of their income through advertising, this is unlikely to happen. If you need end-to-end encrypted emails, PGP can provide that – just be aware that few people will be able to correctly receive your mails.

WEB SEARCH

When browsing the web, the search engine you pick has a huge bearing on how you're tracked. Google may be convenient and provide good results, but it also heavily tracks users. Our two favourite search engines that don't spy on users are:

- **Duckduckgo.com** As well as respecting your privacy, Duckduckgo is also a supporter of Free Software and donates to several major open source projects.
- **Startpage.com** If you want Google results without the tracking, this is the option for you. Startpage returns the search results from Google but doesn't pass on any of your personal details, so your privacy remains intact.



When possible, the messages you send and your calls are secured with end-to-end encryption, which means WhatsApp and third parties can't read or listen to them.

Learn more about WhatsApp security.

Show security notifications

To make sure you have end-to-end encryption in *WhatsApp*, go to Settings > Account > Security and you should have this message.

WhatsApp, one of the world's most popular online chat applications, now features end-to-end encryption by default. This development is possibly the largest single step towards online privacy ever, because although there have been encrypted chat tools around for decades, they have always been tricky to set up and use and have never become mainstream. With over a billion users, WhatsApp is definitely mainstream, and as long as you're using the latest version of the software, your messages are secure from prying eyes – even *WhatsApp*'s makers can't access your messages.

While *WhatsApp*'s move to end-to-end encryption is excellent, there are a couple of drawbacks to it: it's mobile-centric with no desktop Linux client, and it's closed source, so we can't verify that it really does what it claims. You can access the *WhatsApp* web client via Linux, but you can't create an account unless you have a smartphone. Security campaigner Moxie Marlinspike helped develop the encryption, and while no endorsement can give the same guarantees as access to the source code, Moxie is a highly regarded encryption expert.

Also from Moxie Marlinspike (and Open Whisper Systems), the *Signal* app for Android and iOS is a highly secure private messaging app. Its big advantage over *WhatsApp* is that it's open source, while its big disadvantage is that it has far fewer users.

Privacy for dummies

The focus on making it easy to use means that *Signal* really can bring best-in-class privacy to people who aren't technical. Edward Snowden's advice on private communication is "Use anything by Open Whisper Systems." If you want to check for yourself that everything's secure, *Signal* is open source, so you can look for anything improper in the code, or compile from source to make sure that no backdoors are added to the code before it's released.

Off The Record (*OTR*) isn't itself a chat system – it's a text-based encryption system that can run on top of any instant messaging protocol. It's most commonly used alongside Jabber, since this is an open source system that enables users to register with one of many servers rather than relying on a single company or organisation that could be profiting from users' data.

If ultimate privacy is important to you, *Off The Record* is the chat system you should use. However, *WhatsApp* provides a good level of privacy and is widely used, so if you want to contact someone, there's a much better chance that you'll be able to reach them on *WhatsApp* than *OTR*. *Signal*, on the other hand, provides strong privacy, and while it isn't widely used, it is easier to get started with than *OTR*.

BEWARE UNTESTED ENCRYPTION

Creating secure encryption is hard. It requires detailed knowledge of complex mathematics to design an algorithm and detailed knowledge of the implications of low-level computing operations to implement it securely. On top of this, it has to run efficiently for it to be widely used without slowing down the computer or draining the battery on a mobile device. By far the best option is to use well-known encryption algorithms implemented by well-tested open-source libraries. Any other option runs a high chance of not providing adequate security.

LINUX EVERYWHERE

Linux isn't just dominant in the server and mobile space – it's doing the grunt work all over the planet. Linux Voice investigates!

In a weird turn of events, Linux is the most successful computer operating system in every market except for the one for which it was originally developed. Linus Torvalds started his kernel in the early 1990s so that he could have a Unix-like operating system on his home desktop – he wasn't thinking about the server and mainframe markets at the time. No, he just wanted something much better than DOS, Windows or Minix, that didn't cost anything and that could be shared around easily.

Fast forward 25 years, and Linux isn't the massive success story on the desktop that many of us had hoped. Not that we're complaining – it's still an excellent desktop operating system and far better than the alternatives in our opinion. But the momentum and mindshare of Windows would always be difficult to overcome. On the other hand, outside of the desktop, Linux pretty much dominates: servers, cloud infrastructure, mobile devices and much more. Linux can be found in almost every industry you can name.

Look around you

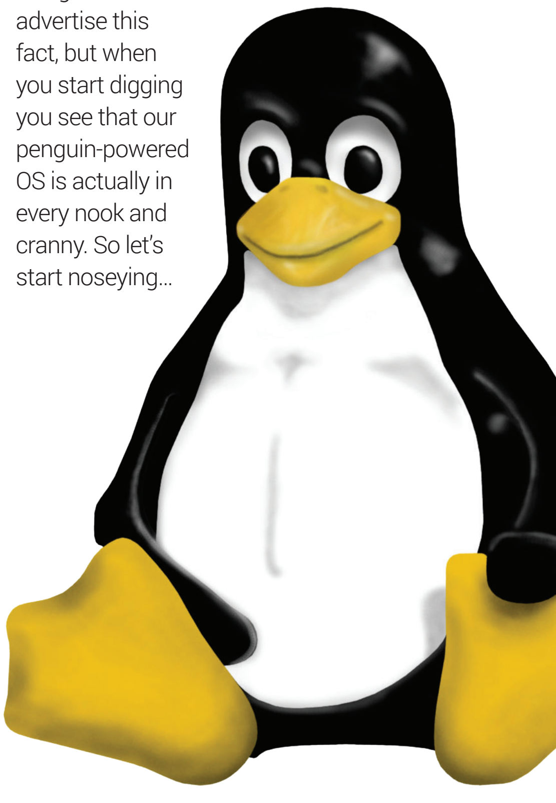
But why is this? Obviously we can run through the usual benefits: Linux is Free and open source software; it has a solid design and architecture; it's portable; it has stable leadership; and it has a very good security track record. Those are all key components in success, but there's one thing we believe is often forgotten: Linux legitimised the idea of Free Software in "serious" business.

When internet service providers – especially the smaller ones – started adopting Linux in the late '90s for their infrastructure, they demonstrated that the OS was capable of handling heavy network traffic. When Linux got established in the server market, it showed it could deal with lots of data and storage.

One thing that gets forgotten is that Linux legitimised the idea of Free Software in "serious" business.

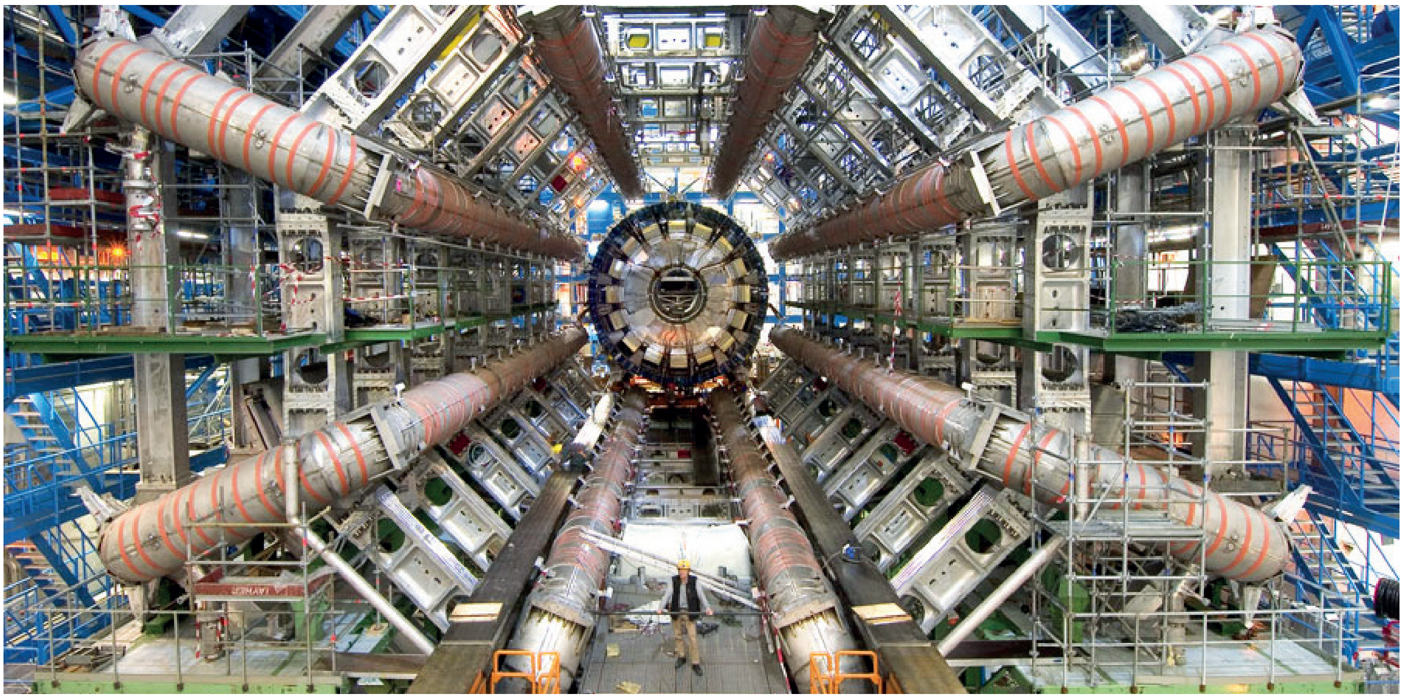
And it goes on... It's more than just about the technology – the perception that Linux is suitable for real-world usage, and with commercial companies providing support, is invaluable to its success.

We thought we'd look at areas where Linux is being used beyond the usual areas. In many cases, the people and companies using Linux don't advertise this fact, but when you start digging you see that our penguin-powered OS is actually in every nook and cranny. So let's start noseying...



LINUX IN SCIENCE

Discovering the mysteries of the universe with FOSS.


The Large Hadron Collider at CERN is, in geek terms, probably the single coolest machine on the planet. (Image: CERN)

We think we have a pretty good grasp of physics, at least when you imagine particles as billiard balls with smaller balls going round them, and a few forces at play. But once you go down to the subatomic level, things start to get rather odd. Quantum mechanics is a fascinating topic – albeit extremely complicated – and Linux is helping to answer some of the questions that it poses.

Take the Large Hadron Collider (LHC), for instance. It's the biggest and most powerful particle accelerator on the planet, straddling the border between France and Switzerland. The LHC is a 27km-long tunnel that took a decade to build with the involvement of scientists from over 100 countries, and since it started operation it has been used to develop theories in particle physics. Specifically, it looks at what happens when certain particles are smashed together at insanely high speeds.

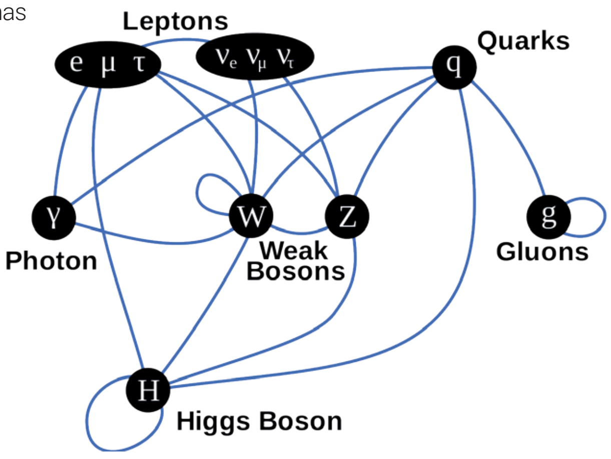
Now, some particles are extremely hard to detect; many have been proposed but never directly observed in action. One such particle is the Higgs boson, which was originally described in the 1960s, but it was an LHC experiment in 2012 that suggested that the particle actually exists. So that's all good and well – but what really interests us is the involvement of Linux. Yes, Linux is very heavily employed in the LHC, and is responsible for processing mind-boggling amounts of data. In 2011 alone, the collider generated a whopping 23 petabytes of data; that's plenty to deal with for CERN, the operator of the LHC, but when hundreds of other scientists want access to the whole data set, it's a big job to shift it all around.

Scientific Linux

Indeed, Linux is so popular among research institutions that an entire distribution has been developed: Scientific Linux. Sponsored by the American Fermi National Accelerator Laboratory, Scientific Linux wasn't created from the ground-up but is essentially a rebuild of Red Hat Enterprise Linux.

Meanwhile, Scientific Linux offers the same benefits of CentOS/RHEL – namely well-tested packages and a very long support lifespan – but with a focus on the scientific community. To that end the distro makes it easier to install cutting-edge research software, and an active community. See the project's website at **www.scientificlinux.org** for all the details.

Another distro with a particular scientific theme is Bio-Linux (**http://environmentalomics.org/bio-linux-download**). As the name suggests, this distro ships with a bunch of software relating to bioinfomatics, such as genome viewers and tools for searching DNA and protein databases. It's based on an older version of Ubuntu (14.04), so it's hardly cutting-edge, but still useful for scientists in that field.



A Higgs boson, and other elementary particles in its class. Linux has been instrumental in (probably) discovering it.

LINUX IN SPACE

Free software powers the ISS and SpaceX's reusable rockets.

The International Space Station (ISS) is a tremendously complicated contraption, with many different modules from different space agencies with different technologies plugged together to make a just-about-working whole. (We're not against the ISS – but given that five of the six astronauts onboard are now just maintaining it and fixing old systems, leaving only one to do real science, we do wonder if it's time to be replaced. Look up Bigelow Aerospace for future options there...)

But anyway. As the most expensive "thing" ever produced, the ISS needs to work reliably, so the choice of software is absolutely critical. You don't want Windows Update to pop up and ask you to wait during a critical re-boost manoeuvre to avoid burning up in the Earth's atmosphere. For this reason, Linux is used throughout the ISS, primarily on laptops used by the astronauts. Prior to 2013, Windows XP was used on the ThinkPad laptops scattered around the various modules of the ISS – but this was changed to Linux because the station "needed an operating system that was stable and reliable".

Debian in orbit

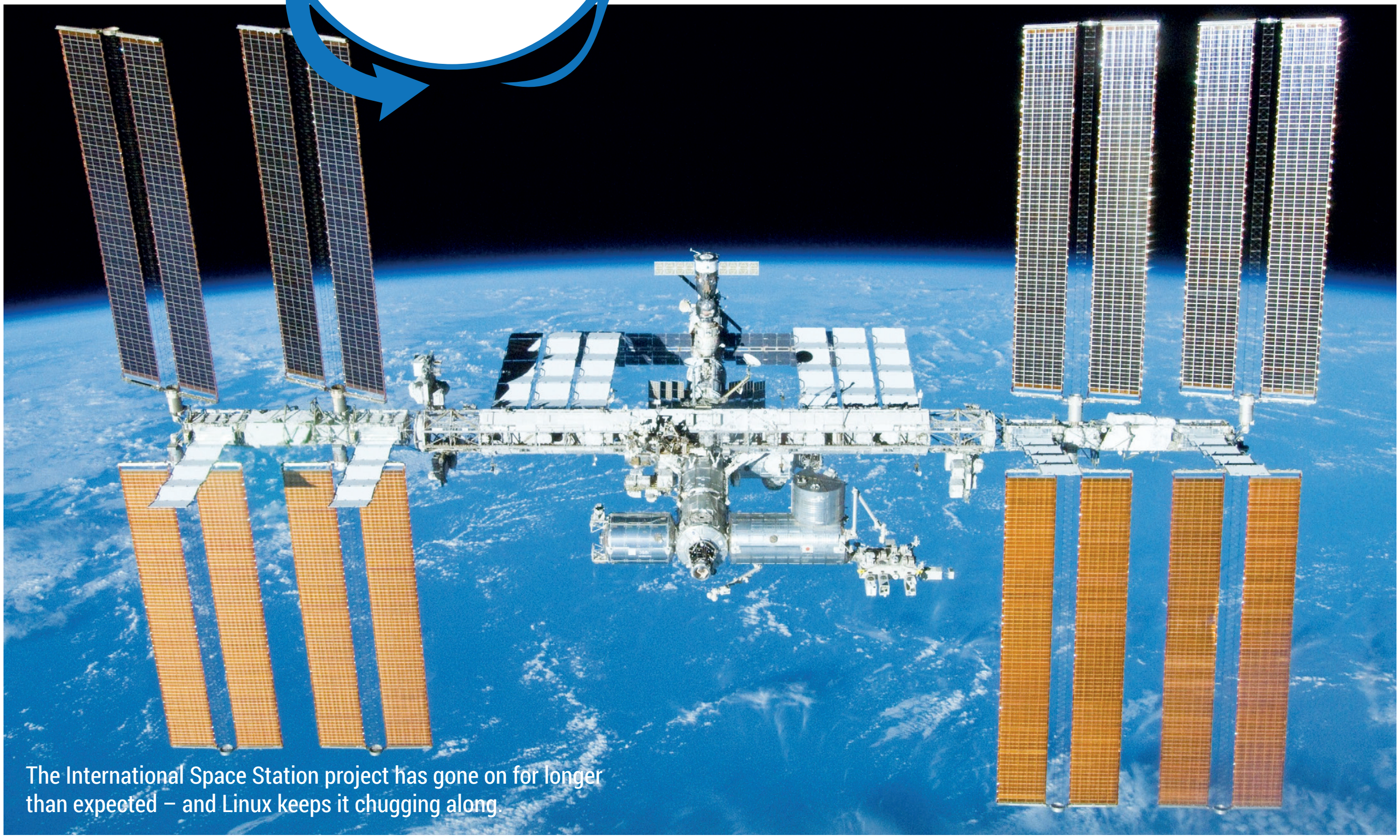
The version of Linux used is Debian, which is fitting as it's hardly cutting edge but one of the most well-tested distros out there. (As an aside, some of the other systems on the ISS run VxWorks, a proprietary real-time operating system from Wind River (now Intel) that has built up a reputation for being extremely robust. The Curiosity Mars rover runs VxWorks, for instance.)

But Linux isn't just powering the computers – it's also running a robot as well. The Robonaut R2 was delivered to the ISS by a Space Shuttle mission in 2011, and it's essentially a robotic torso with arms and fingers. The idea behind the R2 is to explore how a robot can work alongside astronauts, using more dexterity and precision than a simple machine. Don't worry about some kind of robotic take-over in space, though – it's nowhere near enough to develop advanced artificial intelligence. We hope...

For geeks, arguably the most exciting company in the world at the moment is SpaceX. Founded by Elon Musk, who made megabucks from PayPal, SpaceX produces rockets and capsules with the goal of drastically reducing the cost of going to

Linux helps guide the Falcon 9 into space, deliver the payload into orbit, and land the first stage back on Earth

ThinkPads running Linux are the laptops of choice for astronauts on board the ISS.

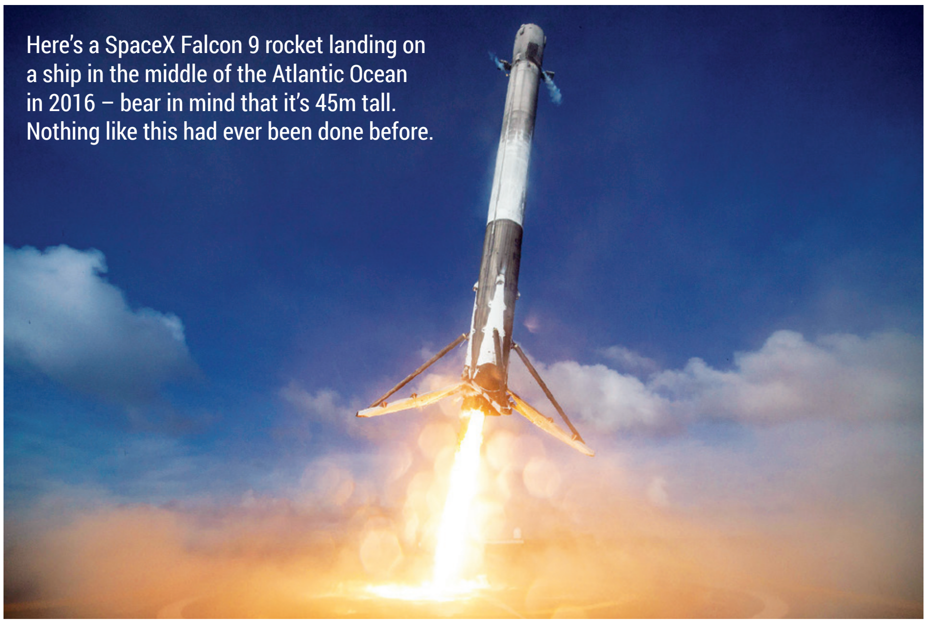

The International Space Station project has gone on for longer than expected – and Linux keeps it chugging along.

SpaceX makes money delivery satellites for customers (and cargo for NASA), and uses that to fund a programme of reusable spacecraft to one day take us to Mars.

space. Right now, a single seat on a Russian Soyuz craft (the only way to get to the ISS) is around $70,000,000 – not cheap.

So why is space so ridiculously expensive? Well, primarily, it's because we throw rockets away after each launch. Because of Earth's gravity, and our available propulsion methods, a rocket can typically get only 2–4% of its overall weight into orbit. So around 97% of the rocket is just fuel, metal and other bits to get the tiny payload into a stable orbit. Once the rocket has done its work, it's out of fuel and falls back into the sea, pretty much completely destroyed.

SpaceX is working hard to make a reusable rocket, and in December 2015 it landed the first stage (main part) of a rocket after performing a successful orbital mission to get some satellites into orbit. So far, SpaceX has landed several rockets; none have been re-flown yet, but this is due before the end of 2016. And if refurbishment costs are as low as hoped, this will be an enormous breakthrough in spaceflight. A brand-new SpaceX rocket costs around $60m, yet the fuel for a mission is only around $300,000. So if SpaceX can start reusing its rockets, maybe 10 or 20 times for each one, it will be able to bring the prices down substantially.


Here's a SpaceX Falcon 9 rocket landing on a ship in the middle of the Atlantic Ocean in 2016 – bear in mind that it's 45m tall. Nothing like this had ever been done before.

To Linux and beyond

Linux is a key component in SpaceX's strategy. Whereas "oldspace" companies would typically contract out the building of a rocket to many different companies, adding complexity and costs, SpaceX strives to do everything in-house with off-the-shelf components and technologies. So its rockets don't run highly customised OSes and software stacks – they use Linux and C++, on triple-redundant computers. Linux helps to guide the Falcon 9 rocket into space, deliver the payload into orbit, and land the first stage back on Earth.

And this is just the beginning. SpaceX has grand plans to establish a colony on Mars in the 2020s, using giant reusable rockets and spaceships – and we can safely assume they'll be running Linux. Yes, our favourite OS will play a key part in the next step of human evolution, going multi-planetary. How awesome is that?

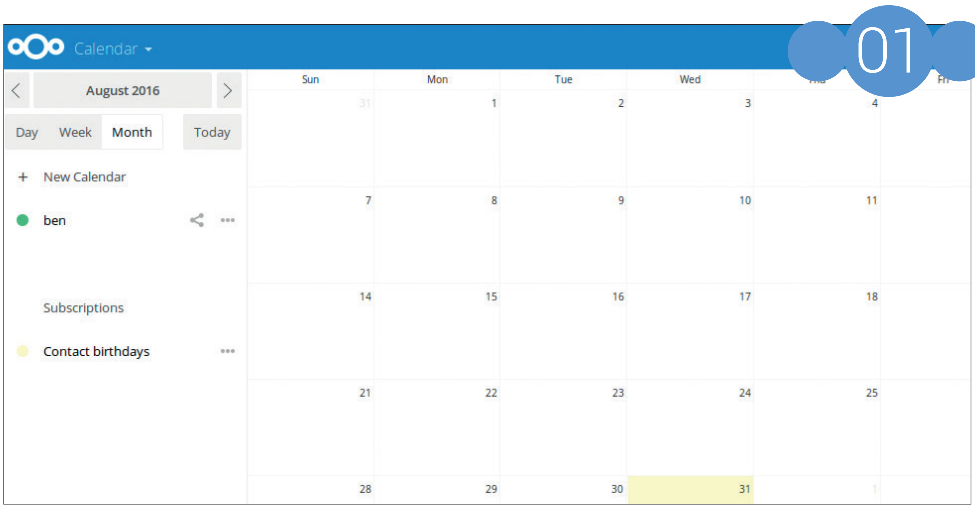
SECRETS OF NEXTCLOUD

Keep your data under control with the newest open source cloud computing offering.

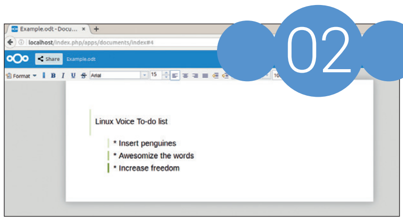
More and more computing is taking place in the web browser. Email, file storage, calendars, and even office suites are now often built in HTML and can be accessed from anywhere on the internet. For users, this means that they are no longer tied to a single computer, but if you're using software running on other people's servers, that means you no longer control your own data.

Fortunately, there is a way of getting all the benefits of web based software without the downside of losing your privacy: self-hosted cloud services. The newest of these is *NextCloud*, a fork of *OwnCloud* with a focus on making development more open with the community. Here are our favourite features of *NextCloud* that can help you get all the benefits of a web-based collaboration system while still retaining control over how your data is handled.

01



02



Linux Voice To-do list

* Insert penguines
* Awesomize the words
* Increase freedom

01 Calendar and email

After storing files, calendar and email are the most useful cloud applications. The Calendar app in *NextCloud* enables you to keep track of your schedule, share it with others and link with other online calendars (via CalDav) so you can see appointments even if they aren't in your calendar. The Email app doesn't include mail servers, but enables you to connect to them via the normal protocols. In this way, it's more like any other mail client (allbeit web based) rather than a full mail server.

02 Documents

While *NextCloud* is most famous for storing and sharing files, you can also edit them directly in the web browser. The Documents plug-in enables you to create and alter ODT, DOC and DOCX format files. If you need more editing power, you can also integrate the Collabora Online office suite, which works better with complex formatting and also enables you to work with spreadsheets and presentations.

03 Collaboration

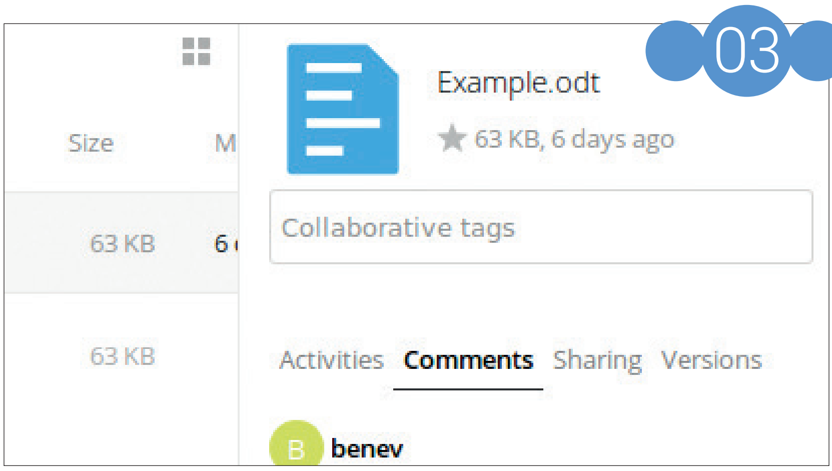
When there's a group of people collaborating on a set of documents, it can be useful to know who's done what and when. *NextCloud* helps with this in a few ways. Firstly, there's the Activity app that shows you the most recent changes made to all the documents you have access to. Secondly there's a versioning system that enables you to view and roll-back any changes to documents. Thirdly there's a comments system where you can discuss a file without changing its contents.

04 Snap

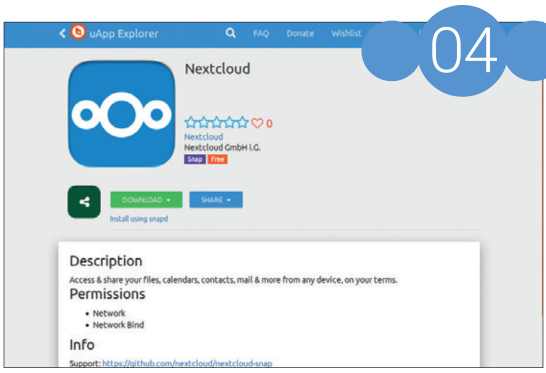
As a web app, installing *NextCloud* takes a little more effort to get running than most software. You'll need a web browser, database and all the associated things – quite a lot if you just want to test it out, and this is enough to put a lot of

> The Calendar app in NextCloud enables you to keep track of your schedule, share it with others and link with other online calendars

03



04



Description

Access & share your files, calendars, contacts, mail & more from any device, on your terms.

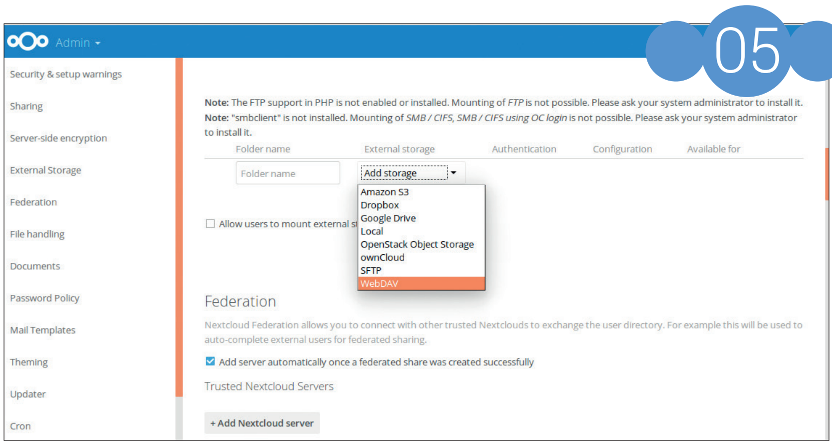
Permissions

- Network
- Network Bind

Info

Support: https://github.com/nextcloud/nextcloud-snap

05



Note: The FTP support in PHP is not enabled or installed. Mounting of *FTP* is not possible. Please ask your system administrator to install it.
Note: "smbclient" is not installed. Mounting of *SMB / CIFS, SMB / CIFS using OC login* is not possible. Please ask your system administrator to install it.

Folder name	External storage	Authentication	Configuration	Available for
Folder name	Add storage			

Amazon S3
Dropbox
Google Drive
Local
OpenStack Object Storage
ownCloud
SFTP
WebDAV

☐ Allow users to mount external st

Federation

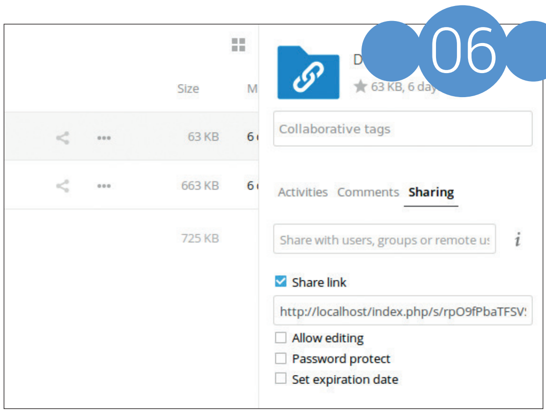
Nextcloud Federation allows you to connect with other trusted Nextclouds to exchange the user directory. For example this will be used to auto-complete external users for federated sharing.

☑ Add server automatically once a federated share was created successfully

Trusted Nextcloud Servers

+ Add Nextcloud server

06



Share with users, groups or remote us

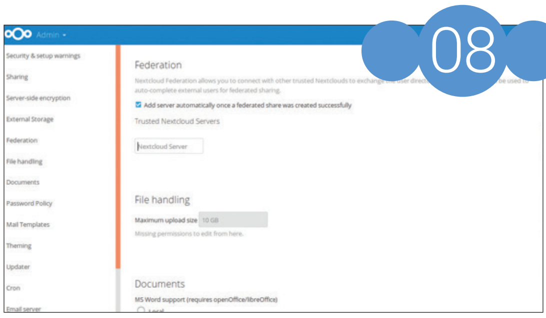
☑ Share link

http://localhost/index.php/s/rpO9fPbaTFSV

☐ Allow editing
☐ Password protect
☐ Set expiration date

07

08



Federation

Nextcloud Federation allows you to connect with other trusted Nextclouds to exchange the user direct[illegible] be used to auto-complete external users for federated sharing.

☑ Add server automatically once a federated share was created successfully

Trusted Nextcloud Servers

Nextcloud Server

File handling

Maximum upload size 10 GB

Missing permissions to edit from here.

Documents

MS Word support (requires openOffice/libreOffice)

Local

people off. Fortunately, it's been packaged up as a Snap, so you can get a working *NextCloud* install with just:

```
sudo snap install nextcloud
```

Once this is done, point your web browser to localhost and you'll be able to create an admin account to complete setup. The Snap is considered beta quality, so it's best to only use it to test out *NextCloud* or for small personal setups.

05 External storage

NextCloud enables users to store, modify and upload files, and all this can end up using a lot of disk space. You can manage this in the same way you would with any other Linux software – get a larger hard drive – but *NextCloud* also allows you to tie in other forms of storage to the *NextCloud* interface. This storage could be on servers you own through Samba, SFTP or WebDav, alternatively, it could be storage from third parties such as Dropbox, Amazon S3 or Google Drive.

06 Sharing and federation

The power of a collaboration platform comes from its ability to help you share information with other people. With *NextCloud*, you can share on a per-file or per-directory basis with named people, the public at large or with a password-protected link.

Federation enables you to join instances of *NextCloud* together. When two instances are linked, you can share information and folders between users on different systems as though they were on the same *NextCloud* server.

07 Mobile

The days when software only had to run on desktop computers are well behind us. The ability to access applications on the go via mobile apps is essential in the modern world, and *NextCloud* makes this easy. You can obviously use the web interface on a mobile phone if you wish, but there are also native apps for Android and iOS that offer a better mobile experience.

08 Open collaboration

The web browser is a universal interface that makes *NextCloud* accessible on almost every internet-connected device. However, there are times when the standard interface just isn't the best option for users, such as when you're creating a customised mobile app or embedding some *NextCloud* features into a different website. For this, there's an external API that conforms to the Open Collaboration Services specification, which enables you to get data from *NextCloud* in XML or JSON format so you can process it in almost any programming language.

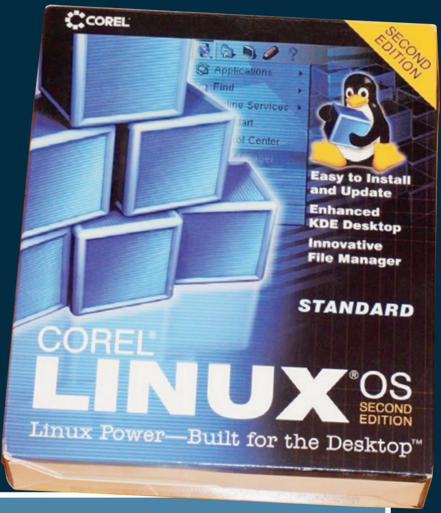
OLD SCHOOL COOL

Mike Saunders looks back at some classic moments in Linux history. And if you're a newbie: yes, Linux used to come in boxes…

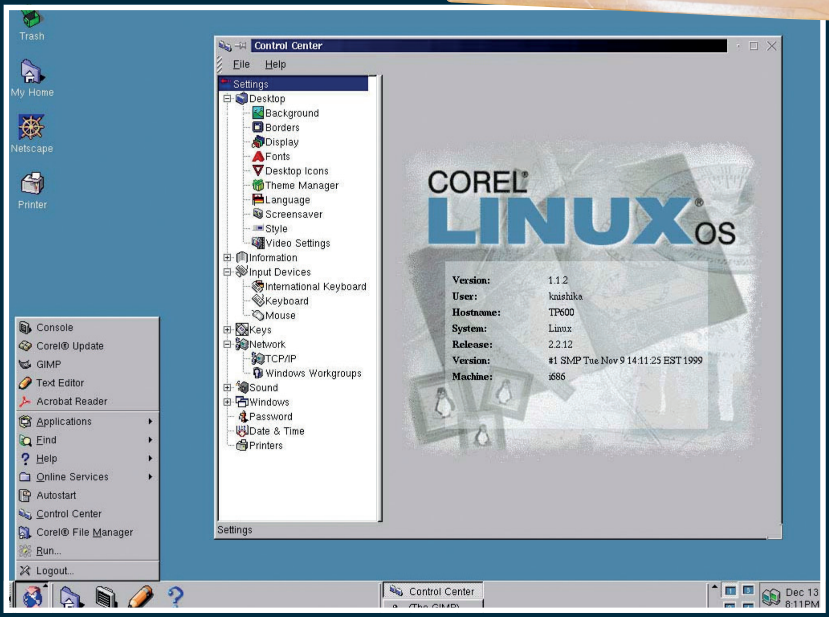
If, like many of us, you got into Linux in the late 1990s or early 2000s, you may have a certain nostalgia for the "good old days". Actually, they weren't very good at all – Microsoft was a bullying monopolist trying to destroy Linux with FUD at every corner, hardware support was iffy, and the desktop had countless rough edges that need to be ironed out. The only serious web browser we had was *Netscape 4.x*, which was a horrendously clunky lump that was famous for locking up, thrashing your hard drive for 30 seconds, and finally giving up the ghost with the totally useless message of "Bus error".

Yet some things had their own charm. For many of us on dialup internet connections, we relied on chunky boxed sets to get Linux onto our machines. (Kids these days with their **pacman -Syu**, eh! They don't know they're born.) We remember placing orders with the old Linux Emporium shop, and twiddling our thumbs waiting for a big lump of SUSE or Red Hat to arrive from the postie a few days later. Yes, even with six CDs the number of packages back then pales in comparison to what's available in repositories today, but it felt completely different – you could really sense the vast range of Free and open source software at your fingertips.

So this issue we thought we'd take a trip down memory lane and look at some of the distros and desktops that the Linux Voice team got started with. If you're a long time Linuxer, prepare for a mighty dose of nostalgia – and if you've only recently converted to the FOSS way, you'll see just what we old 'uns went through to get our Linux fix…

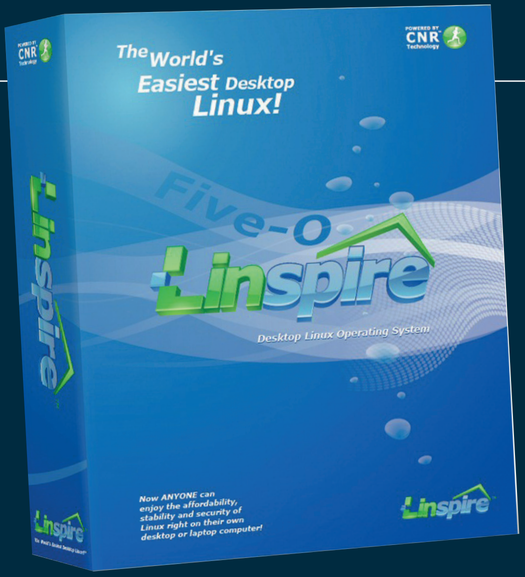


▶ Corel Linux was a short-lived distribution that arrived in 1999. We had high hopes for it back then, as it had the backing of a company known for producing polished end-user software. Corel Linux was pitched directly as an alternative to Windows 9x and Mac OS…



Version: 1.1.2
User: knishika
Hostname: TP600
System: Linux
Release: 2.2.12
Version: #1 SMP Tue Nov 9 14:11:25 EST 1999
Machine: i686

▼ … and here's its desktop. We recall the distro being unbearably slow on our test machines at the time – sure, they only had 64MB of RAM, but that was still plenty for the SUSE and Red Hat versions that were doing the rounds back then. So in all, it's not a distro we really miss.

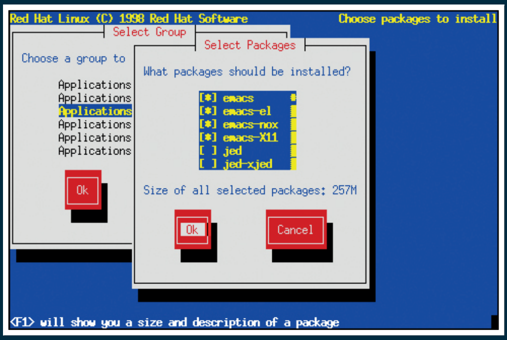

The World's Easiest Desktop Linux!
Five-O
Linspire
Desktop Linux Operating System
Now ANYONE can enjoy the affordability, stability and security of Linux right on their own desktop or laptop computer!
CNR
Linspire

▲

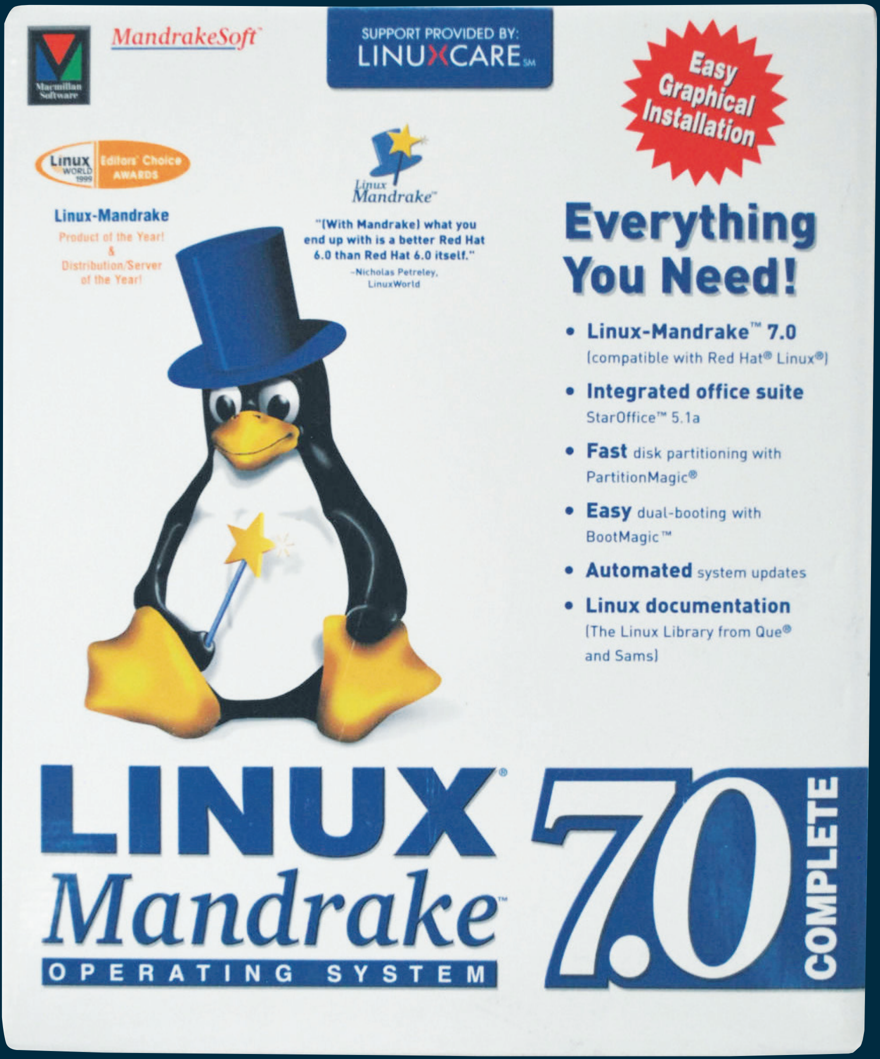
Another distro that aimed to tackle the desktop market, Linspire was originally known as Lindows. Microsoft sued Lindows, Inc. which resulted in a change of name to Linspire, but the distro never really achieved any widespread success.

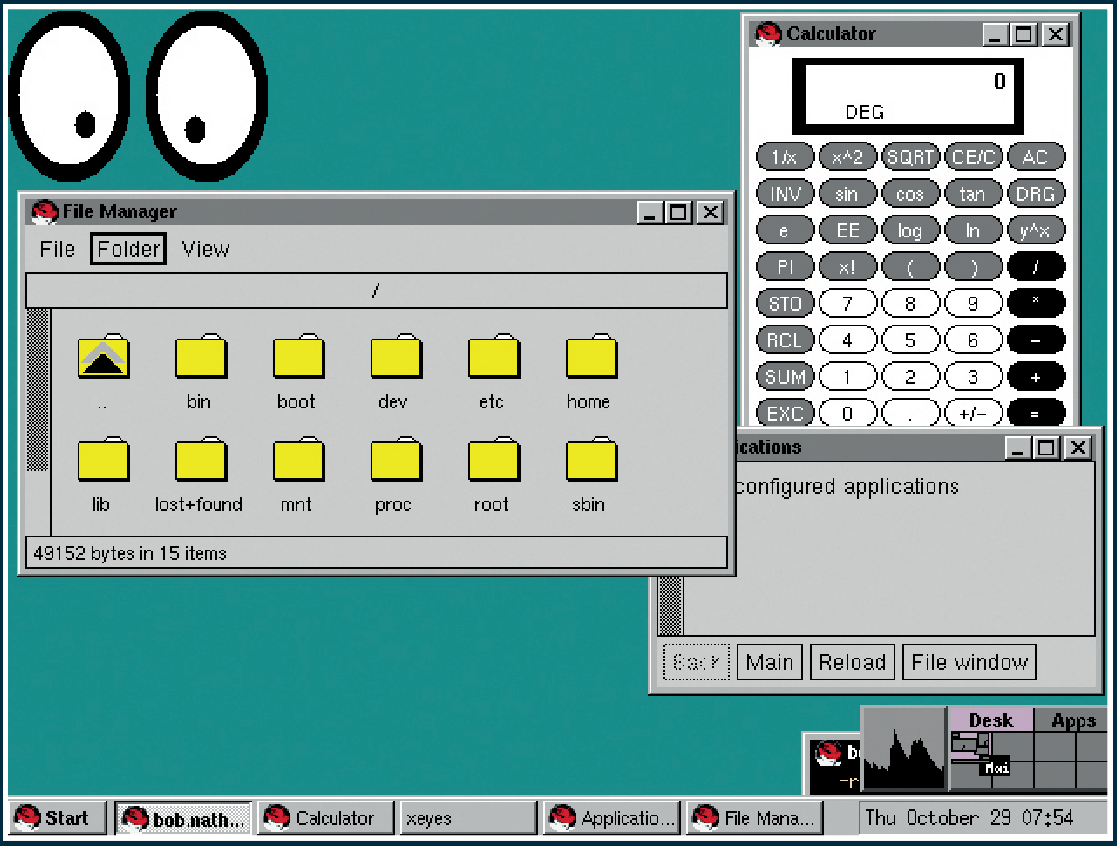
▼

Red Hat's installer back in the late 90s was a text-based tool, but it used dialogs and menus sufficiently that most people could get it installed with a bit of prior knowledge.



```
Red Hat Linux (C) 1998 Red Hat Software                Choose packages to install
        Select Group
                          Select Packages
Choose a group to
                    What packages should be installed?
     Applications
     Applications            [*] emacs
     Applications            [*] emacs-el
     Applications            [*] emacs-nox
     Applications            [*] emacs-X11
     Applications            [ ] jed
                             [ ] jed-xjed
       Ok
                    Size of all selected packages: 257M

                          Ok            Cancel

<F1> will show you a size and description of a package
```


MandrakeSoft
Macmillan Software
SUPPORT PROVIDED BY: LINUXCARE
Easy Graphical Installation
Linux World 1999 Editors' Choice Awards
Linux-Mandrake
Product of the Year! & Distribution/Server of the Year!
Linux Mandrake
"[With Mandrake] what you end up with is a better Red Hat 6.0 than Red Hat 6.0 itself."
–Nicholas Petreley, LinuxWorld
Everything You Need!
• Linux-Mandrake™ 7.0 (compatible with Red Hat® Linux®)
• Integrated office suite StarOffice™ 5.1a
• Fast disk partitioning with PartitionMagic®
• Easy dual-booting with BootMagic™
• Automated system updates
• Linux documentation (The Linux Library from Que® and Sams)
LINUX Mandrake OPERATING SYSTEM 7.0 COMPLETE

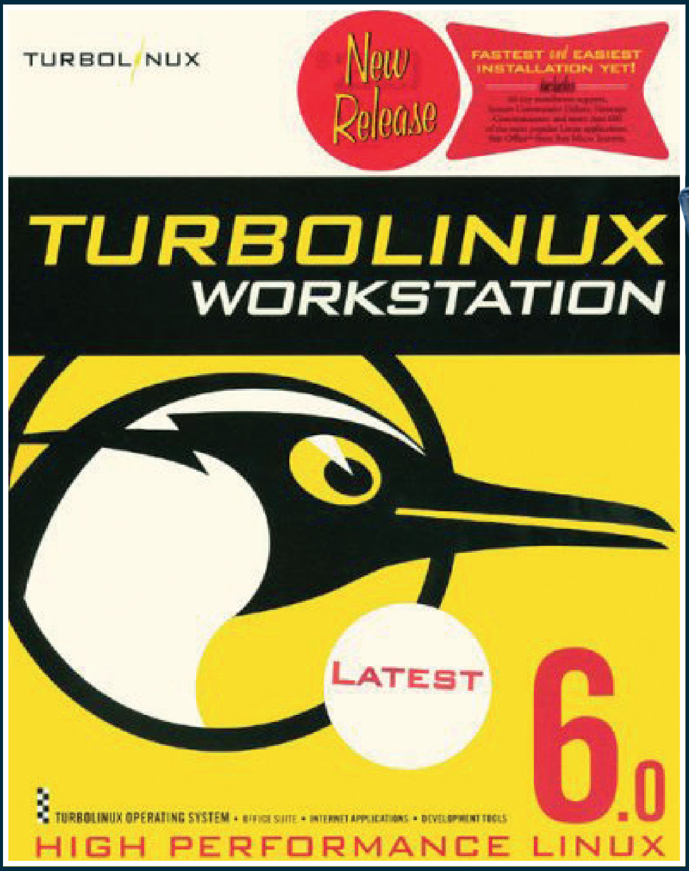


▲

Ahh, Linux Mandrake – arguably the first truly popular newbie distro. This French distro, which came to life in 1998, was little more than a modified Red Hat, but with time it gained its own custom features and was one of the first to sport a fully graphical installation process. Mandrake later became Mandriva, and is now mostly used in its community supported spin-offs Mageia and OpenMandriva.

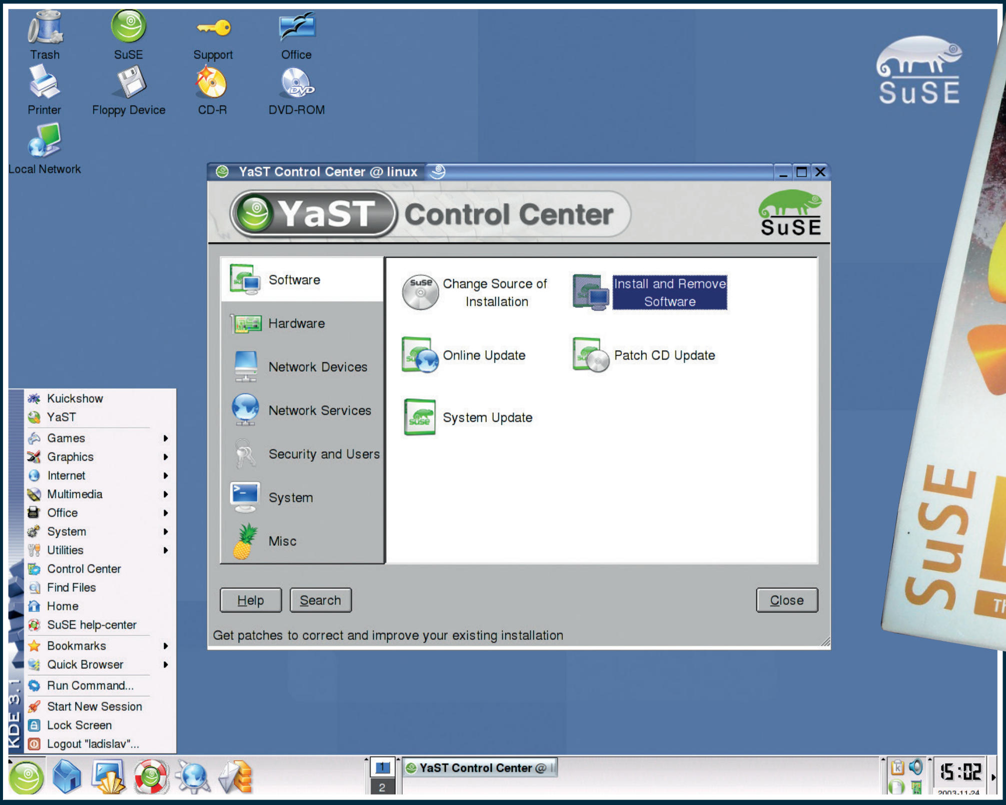
◀

Here's Red Hat Linux 5.0, featuring the FVWM95 window manager in its AnotherLevel configuration. Yes, it's mightily ugly by today's standards, and a lot of fiddling with *XF86Setup* (shudder) was required just to get a usable desktop. And even once we had this set up, we still needed to recompile our kernel to get sound. (Image credit: toastytech.com)


TURBOLINUX
New Release
FASTEST and EASIEST INSTALLATION YET!
TURBOLINUX WORKSTATION
LATEST
6.0
TURBOLINUX OPERATING SYSTEM • OFFICE SUITE • INTERNET APPLICATIONS • DEVELOPMENT TOOLS
HIGH PERFORMANCE LINUX

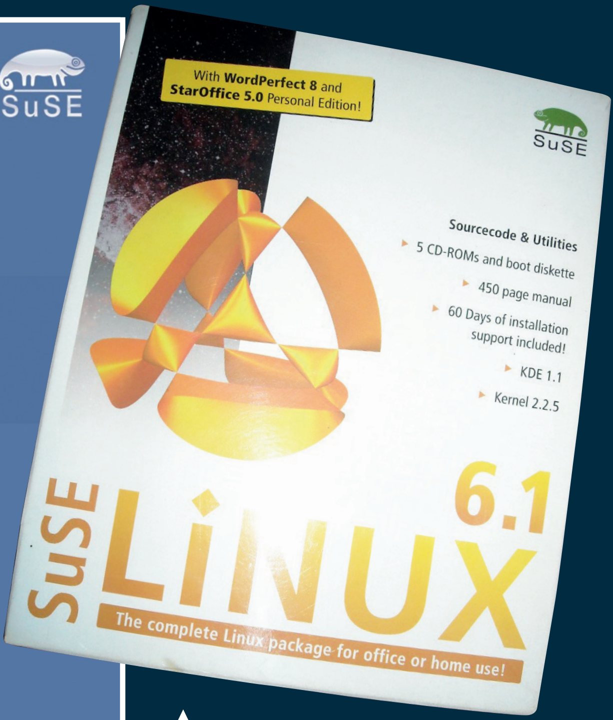
◀ Most of us associate Turbolinux with Japan, but the distribution originally started as a Red Hat spinoff created by developers in Utah, USA. In 2002, Turbolinux was bought up by a Japanese company which focused the distro on its local market.

▲ SUSE's European HQ in Nuremburg has a selection of boxed sets on display. SUSE was one of the first distros to really push its big boxed sets as selling points, bulging with CDs and manuals. Here's the German version of release 4.2, back when the distro was called S.u.S.E. Note how the CD case proudly announces that it uses the ELF format…

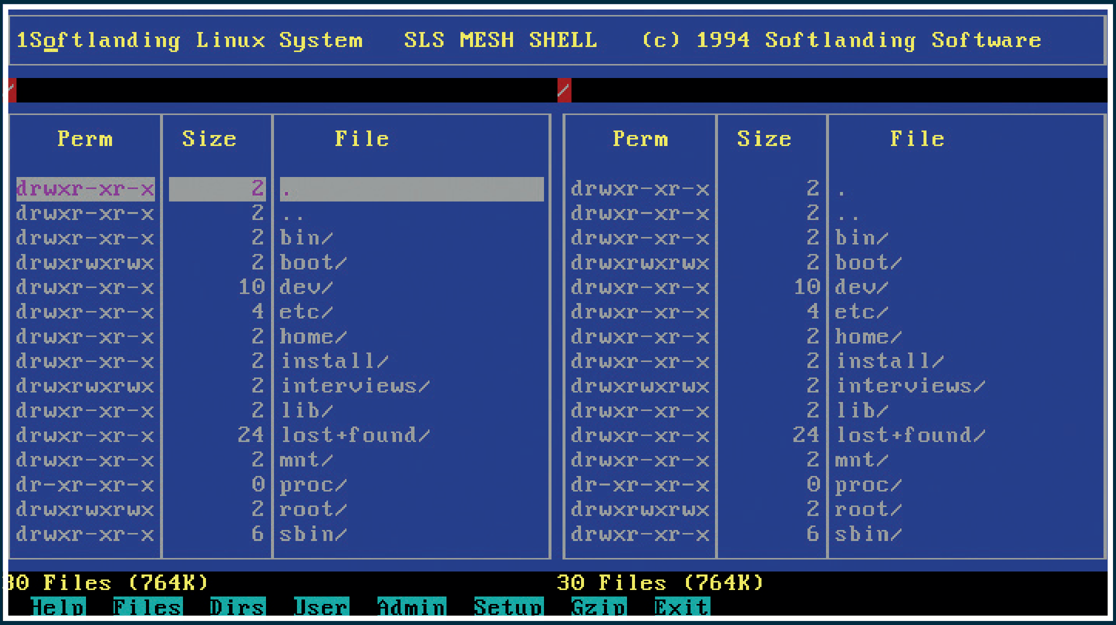


Get patches to correct and improve your existing installation

▲ And here's a shot of SUSE 9.0's desktop – KDE 3.1. Historically, SUSE was one of the best champions of KDE, adding lots of fine touches to the desktop to really make it shine. While KDE was available in other distros like Red Hat, it was often seen as a second-class citizen, whereas in SUSE it was the default for many releases.

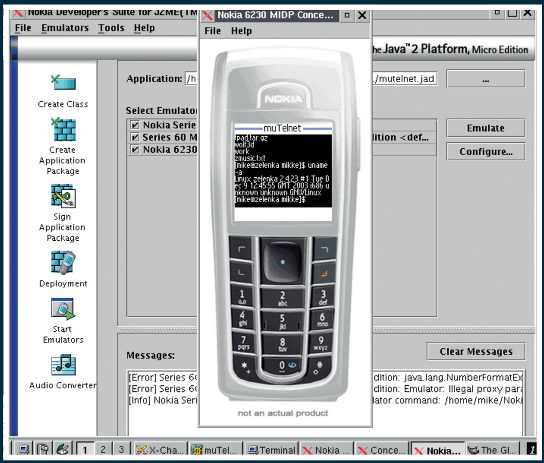

With WordPerfect 8 and StarOffice 5.0 Personal Edition!
SuSE
Sourcecode & Utilities
5 CD-ROMs and boot diskette
450 page manual
60 Days of installation support included!
KDE 1.1
Kernel 2.2.5
SuSE LINUX 6.1
The complete Linux package for office or home use!

▲ By SUSE 6.1, the boxed sets were more polished and the manuals even bigger (they were a superb resource of information on all things Linux at the time, even if they did talk about very SUSE-specific ways of doing things). This release included *StarOffice*, not so long before it morphed into the fully open source office suite *OpenOffice.org*.



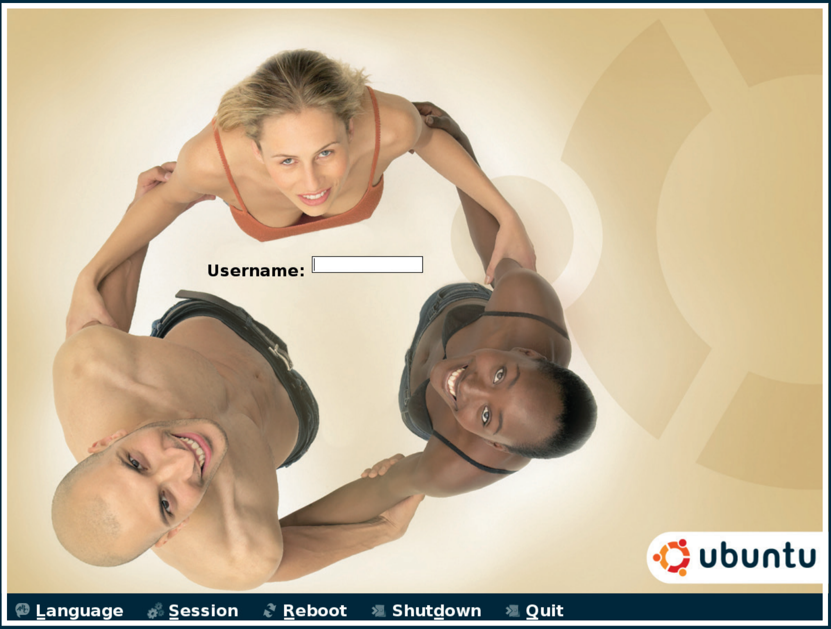
```
1Softlanding Linux System    SLS MESH SHELL    (c) 1994 Softlanding Software

 Perm         Size      File                 Perm         Size      File

drwxr-xr-x       2 .                        drwxr-xr-x       2 .
drwxr-xr-x       2 ..                       drwxr-xr-x       2 ..
drwxr-xr-x       2 bin/                     drwxr-xr-x       2 bin/
drwxrwxrwx       2 boot/                    drwxrwxrwx       2 boot/
drwxr-xr-x      10 dev/                     drwxr-xr-x      10 dev/
drwxr-xr-x       4 etc/                     drwxr-xr-x       4 etc/
drwxr-xr-x       2 home/                    drwxr-xr-x       2 home/
drwxr-xr-x       2 install/                 drwxr-xr-x       2 install/
drwxrwxrwx       2 interviews/              drwxrwxrwx       2 interviews/
drwxr-xr-x       2 lib/                     drwxr-xr-x       2 lib/
drwxr-xr-x      24 lost+found/              drwxr-xr-x      24 lost+found/
drwxr-xr-x       2 mnt/                     drwxr-xr-x       2 mnt/
dr-xr-xr-x       0 proc/                    dr-xr-xr-x       0 proc/
drwxrwxrwx       2 root/                    drwxrwxrwx       2 root/
drwxr-xr-x       6 sbin/                    drwxr-xr-x       6 sbin/

30 Files (764K)                             30 Files (764K)
  Help  Files  Dirs  User  Admin  Setup  Gzip  Exit
```

◀ On the subject of really old distros, here's Soft Landing System, which started in 1992 – just one year after Linus Torvalds announced his very basic kernel. Soft Landing System established the concept of a distribution, including some useful software beyond the bare basics required to get a GNU/Linux system up and running.

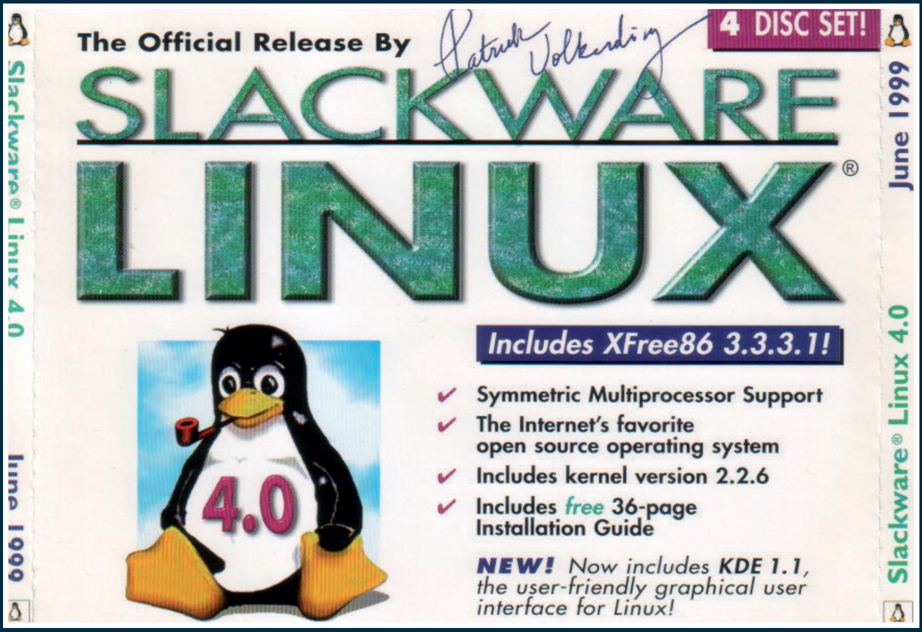

Nokia Developer's Suite for J2ME(TM
File Emulators Tools Help
Nokia 6230 MIDP Conce...
File Help
the Java™ 2 Platform, Micro Edition
Create Class
Create Application Package
Sign Application Package
Deployment
Start Emulators
Audio Converter
Emulate
Configure...
Clear Messages
not an actual product
X-Cha... muTel... Terminal Nokia ... Conce... Nokia... The Gl...

```
Application: /h              ./mutelnet.jad
Select Emulator
☑ Nokia Serie
☑ Series 60 M                ition <def...
☑ Nokia 6230

muTelnet
pad.tar.gz
wolf3d
work
zmusic.txt
[mike@zelenka mikke]$ uname
-a
Linux zelenka 2.4.23 #1 Tue D
ec 9 12:45:55 GMT 2003 i686 u
nknown unknown GNU/Linux
[mike@zelenka mikke]$

Messages:
[Error] Series 6      dition: java.lang.NumberFormatEx
[Error] Series 6      dition: Emulator: Illegal proxy para
[Info] Nokia Seri     lator command: /home/mike/Noki
```

▲ Here's a screenshot from Mike's old archives. This shows a Java-based development environment for old Nokia phones in action, which is running a phone emulator, which in turn is running a Telnet client, which (bear with us) is in turn logging back in to the same Linux box host. Phew...



▲ In the early days of Ubuntu, the distro's developers were somewhat more daring with their marketing materials. Here's one login screen concept, which attempts to highlight the "humanity" in Ubuntu (both in terms of the distro and in the name as a whole). For us, it just gives us vertigo if we look at it for too long.


The Official Release By
Patrick Volkerding
4 DISC SET!
SLACKWARE
LINUX®
June 1999
Slackware® Linux 4.0
4.0
Includes XFree86 3.3.3.1!
✔ Symmetric Multiprocessor Support
✔ The Internet's favorite open source operating system
✔ Includes kernel version 2.2.6
✔ Includes free 36-page Installation Guide
NEW! Now includes KDE 1.1, the user-friendly graphical user interface for Linux!

◀ Slackware is the longest-running Linux distro, and still a favourite among power users who just want a Linux flavour that leaves them alone to do their work. Here's the box for version 4.0 from 1999, gleefully shouting that it includes multiprocessor support and the latest version of XFree86. (Image credit: **http://blog.nielshorn.net**).

We've come a lomg way – think of the olden days when you next power on your Linux box! LV

GNOME USABILITY TESTING

Jim Hall and **Renata Gegaj** reveal the ways in which the Gnome team is making its software more usable for real-life human beings.

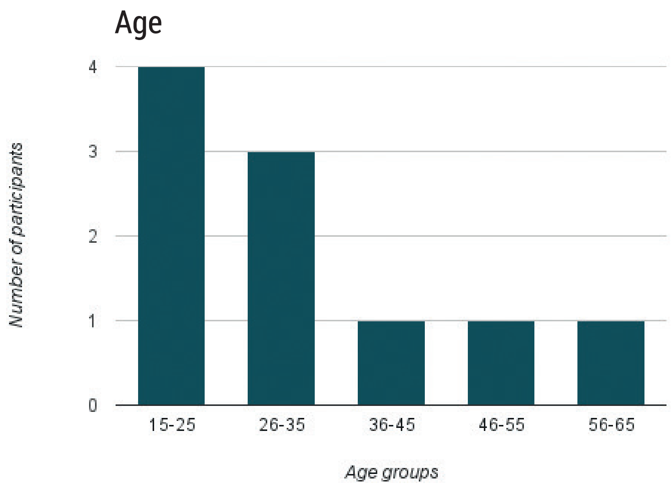
WHY USABILITY TESTING?

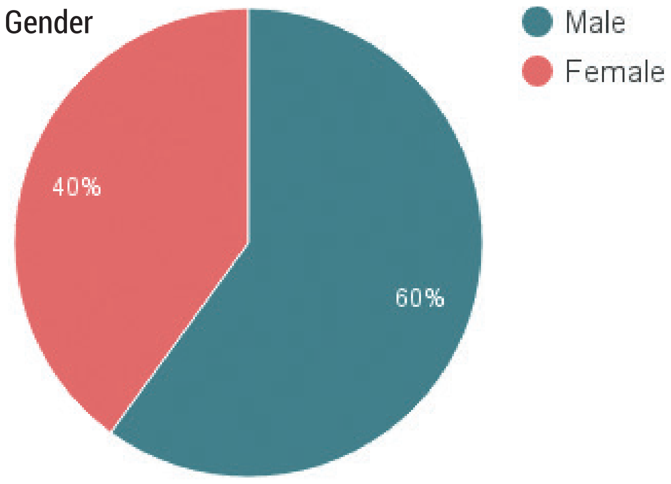
Usability in any program is important if you want other people to use it. Applications need to be easy to use. If an application is too difficult to use, people may switch away from that application and use another similar piece of software that's easier and more obvious. In the sphere of open source software, if an open source software program is too difficult to use, people may stop using it and use a proprietary software program instead. So if we want people to keep using open source software, it's important to get open source software usability right.

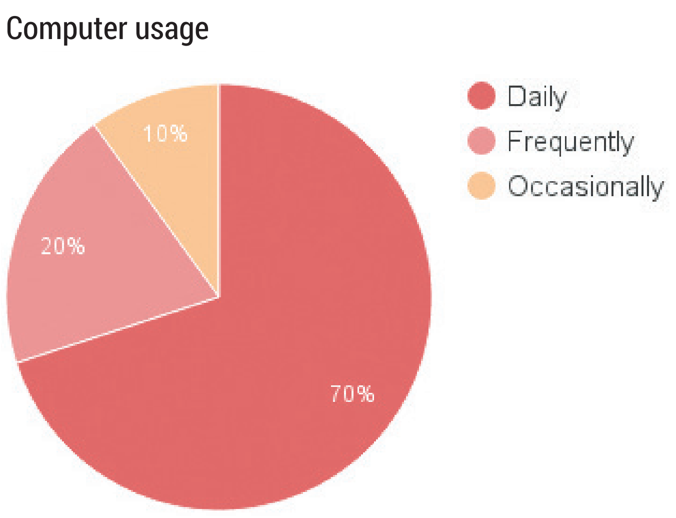
At the same time, it can be difficult for open source software developers to create programs that have good usability. As the authors of our own software, we know how to access all of the program's functionality, and we know how all the menu items work. But can someone else figure out your menus? Can an average user with average knowledge use your software to do their work in a reasonable amount of time? That's the essence of good usability.

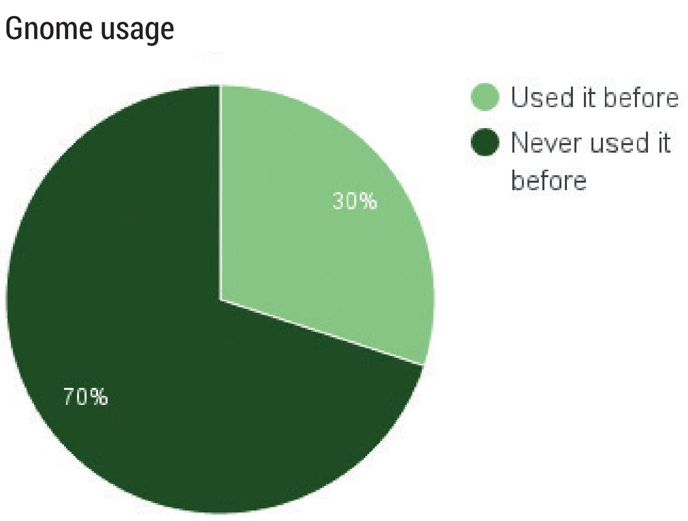
Usability testing involves watching real people use the product to accomplish real tasks, giving you

Below: Most testers were aged 15–35.
Below right: Gender was about equally divided, with slightly more men than women.









Above left: Almost all testers used a computer daily or frequently. **Above:** Most testers had never used Gnome before.

live feedback from users. By doing this iteratively – creating a design, testing it, tweaking the design, then testing it again – you can make your program easy for everyone to use.

Usability testing doesn't require any particular expertise. Anyone can do usability testing with only a little preparation. Let's walk through my usability test as an example.

PREPARING YOUR USABILITY TEST

During Summer 2016, as part of an internship through the Outreachy programme, I [Renata] worked with the Gnome design team to examine the usability of Gnome. Outreachy helps people from underrepresented groups get involved in free and open source software. Jim Hall, Allan Day and Jakub Steiner from the Gnome Design Team coached me on test design and analysis. You can do usability testing using a variety of methods; there is no "one true method" to usability testing. For example, if you're looking at a new design and want to see how users will like it, you can do a usability test using a paper prototype and ask testers to walk through the interface as they respond to different tasks. But if you have already developed a user interface, a more traditional usability test might be more helpful.

My usability test was a traditional usability test, looking at areas of ongoing development in *Gnome Photos* and *Gnome Calendar*. I found 10 volunteers, representing a mix of genders (slightly more men than women) and ages (ages fifteen to sixty-five).

All participants used a computer on a daily basis, and self-identified their computer expertise in a range from "I don't know a lot, I am not a frequent user" to "I am better than most," although most claimed "I am not a frequent user" or "I know some things." Most testers had not used Gnome before; three testers used Gnome on a daily basis, while a few others had heard about Gnome previously but did not use it. For most testers, this was their first time using Gnome.

SCENARIO TASKS

The key to a good usability test is carefully writing the tasks that you ask your testers to do. These are called "scenario tasks" and need to represent what real users would actually do as part of their work. Writing these scenario tasks takes some effort. Start by thinking about what types of people are most likely to use the software, then consider how these users are most likely to use the software to do real work. From that assumption, you can write your own scenario tasks: short assignments you give to each tester, so everyone is doing the same thing during the test.

Scenario tasks should set a brief context, then ask the tester to do something specific. Be careful when writing the scenario tasks; you should not accidentally provide hints or clues to how to complete the task.

For example, a scenario task for a web browser might ask the tester to change the web page's font size. If the menu item to do this is labelled "Font," you should use some other word than "font" to describe the task. One way to write this scenario task is "You don't have your glasses with you, so it's hard to read the text on the web page. Please make the text bigger on the web page." Here are some of the scenario tasks from my usability test:

Photos

1 You just got back from your trip to Thailand. You want to show your friends all the pictures that you took there but you notice that they are all mixed with the other pictures that were previously in *Gnome Photos*. To avoid confusion, you decide to collect the pictures from your trip. Create a group of these photos and name it "Thailand trip."

2 As you are showing the pictures of the album you just created, you notice a picture that you'd like to share on your social media accounts. Please "like" that picture, in order to access it more easily later on.

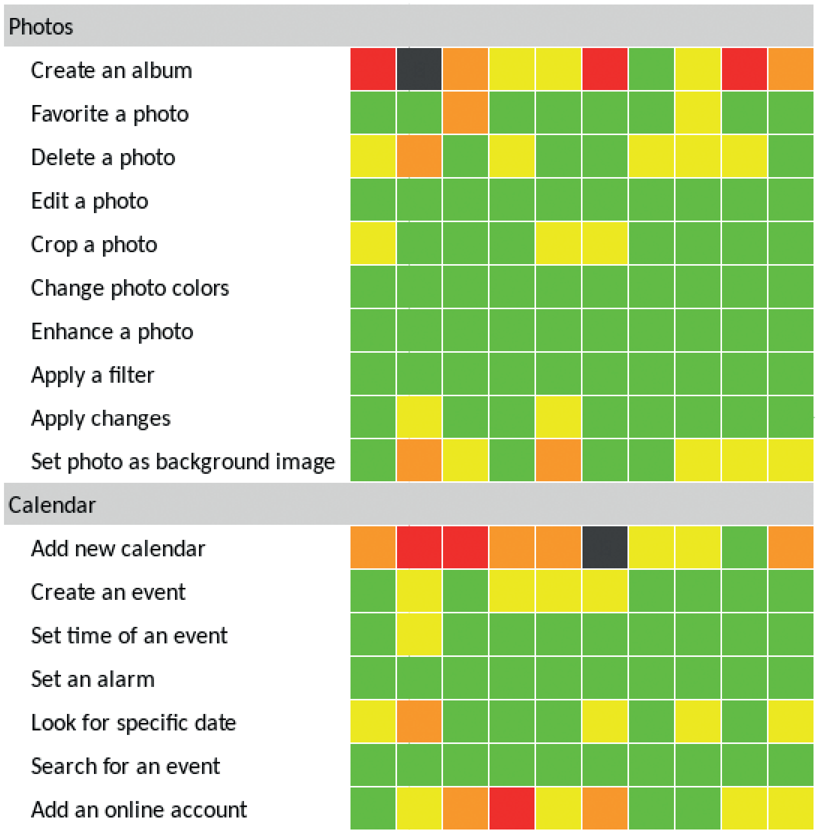
3 While looking through all the photos, you notice that two of the photos look very similar. Please delete one of them.

Calendar

1 You want to have all your work-related activities in your calendar but you don't want them to be mixed with other activities, since you have a lot going on lately. To keep your events organised, you decide to put all your work activities on the same calendar. Make a new calendar, call it "Work," and make the activities appear in purple.

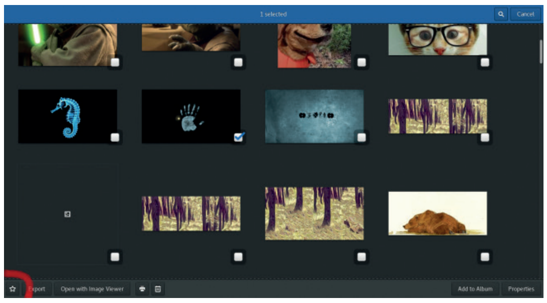
2 You have a meeting with your boss today. Create an item named "Meeting for work" that starts at 2.00pm and goes until 4.00pm. Put it in the "Work" calendar that you created earlier.

3 Set a reminder 10 minutes before the meeting starts, so you don't forget that meeting.

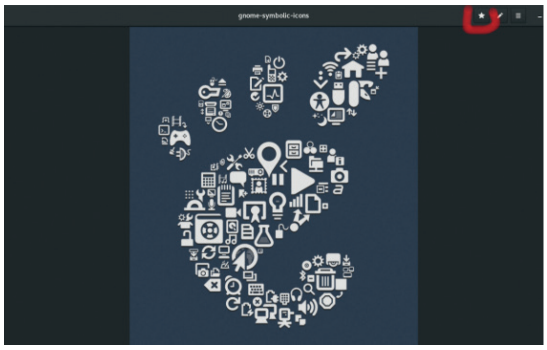


Photos										
Create an album	🟥	⬛	🟧	🟨	🟨	🟥	🟩	🟨	🟥	🟧
Favorite a photo	🟩	🟩	🟧	🟩	🟩	🟩	🟩	🟨	🟩	🟩
Delete a photo	🟨	🟧	🟩	🟨	🟩	🟩	🟨	🟨	🟨	🟩
Edit a photo	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Crop a photo	🟨	🟩	🟩	🟩	🟨	🟨	🟩	🟩	🟩	🟩
Change photo colors	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Enhance a photo	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Apply a filter	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Apply changes	🟩	🟨	🟩	🟩	🟨	🟩	🟩	🟩	🟩	🟩
Set photo as background image	🟩	🟧	🟨	🟩	🟧	🟩	🟩	🟨	🟨	🟨
Calendar										
Add new calendar	🟧	🟥	🟥	🟧	🟧	⬛	🟨	🟨	🟩	🟧
Create an event	🟩	🟨	🟩	🟨	🟨	🟨	🟩	🟩	🟩	🟩
Set time of an event	🟩	🟨	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Set an alarm	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Look for specific date	🟨	🟧	🟩	🟩	🟩	🟨	🟩	🟨	🟩	🟨
Search for an event	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Add an online account	🟩	🟨	🟧	🟥	🟨	🟧	🟩	🟩	🟨	🟨

A heat map is a quick way to show usability test results.

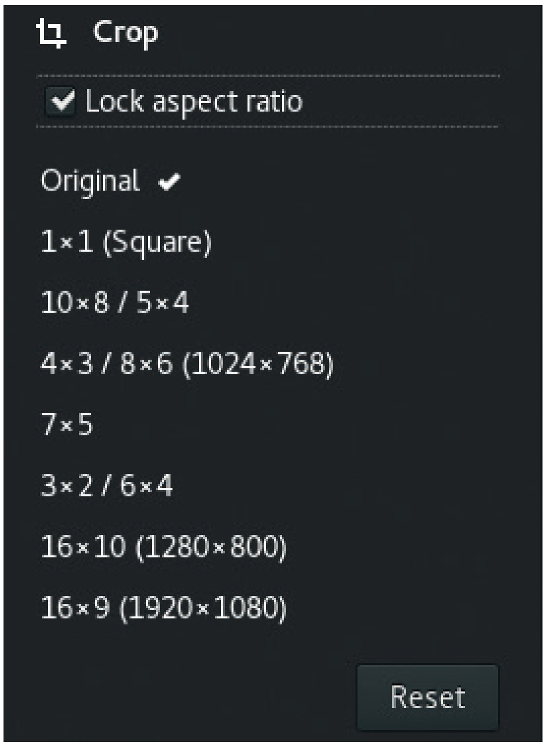


Marking a photo as a "Favourite" from selection mode in Photos.



Using the star icon (here highlighted in the top-right) to mark a photo as a "Favourite" in Photos.

Testers used a dedicated laptop running Fedora 24 with Gnome 3.20, without any modifications or extensions. I used the stock Gnome 3.20 to test *Gnome Photos*, but used the in-development Gnome Continuous Image (running on Gnome Boxes) to test *Gnome Calendar*. Each tester executed their tests using a separate guest account that had been pre-loaded with sample files.



Crop

Lock aspect ratio

Original ✔

1×1 (Square)

10×8 / 5×4

4×3 / 8×6 (1024×768)

7×5

3×2 / 6×4

16×10 (1280×800)

16×9 (1920×1080)

Reset

The "Crop photo" menu.

To conduct our test,wel presented each tester with 16 scenario tasks, which they completed in less than 50 minutes. Throughout the test, we observed and took notes. Afterwards, we thanked them for their time and followed up with a few questions.

ANALYSING THE RESULTS

An easy way to summarise usability test results is with a "heat map." This is a coloured grid that represents each tester in a separate column and each task on a separate row. A coloured cell represents how easy or difficult it was for the participant to accomplish a certain task.

Generally speaking, participants accomplished most of the tasks without difficulties. These tasks included editing a picture, changing the colour, enhancing, and applying a filter in *Photos*; and setting an alarm and searching for an event in *Calendar*.

Participants encountered the most difficulties in creating an album in *Photos*, adding a new calendar in *Calendar*, and adding a new online account in *Calendar*. Two testers were unable to complete two of the tasks, which is a small number considering the number of other tasks that they completed successfully.

Looking at the heat map, you can easily spot the "cool" rows with lots of green and yellow, indicating tasks that the testers found easy to do. You can also identify several "hot" rows with orange and red, representing tasks where testers encountered difficulty. To understand the results, we need to dig deeper than just the heat map. Referring to the notes I captured while observing the testers, we can uncover what happened during the usability test and why testers reacted the way they did. Let's examine several tasks that worked well and others that our testers found more difficult.

WHAT WORKED WELL

Photos

Favourite a picture (task 2): As you are showing the pictures of the album you just created, you notice a picture that you'd like to share on your social media accounts. Please "like" that picture, in order to access it more easily later on.

This task was pretty easy for almost all testers to accomplish. Testers noted that the star icon was very easy to find, which helped them complete the task. You can "Favourite" a photo in several different ways in *Photos*; some participants marked the photo as a Favourite from selection mode, while others used the star icon while viewing a photo to mark it as a Favourite.

Edit a picture (task 4): After showing your friends all the pictures of the "Thailand trip" album, you revisit your "favourite" picture since you want to edit it before sharing on social media.

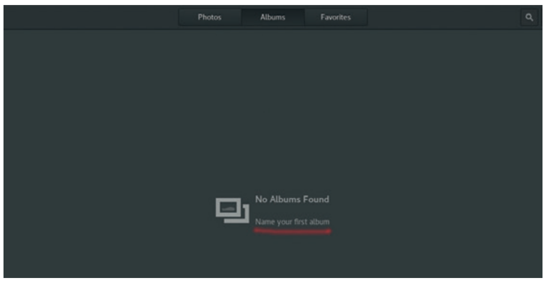
This task turned out to be straightforward and intuitive for all the participants. Testers commented that the "pencil" icon used to edit a photo seemed obvious and familiar to them from other photo editing applications that use a similar icon.

Crop a picture (task 5): You start by cropping the picture to make it look smaller.
Most testers completed this task very easily. You may notice a few yellow boxes on this row in the heat map, showing some testers who experienced a little difficulty. When these few participants went into the Crop action, they first assumed that they could edit the picture using only the ratios in the sidebar, so were initially confused; these testers quickly discovered they could uncheck the Lock Aspect Ratio option to crop the photo the way they wanted.

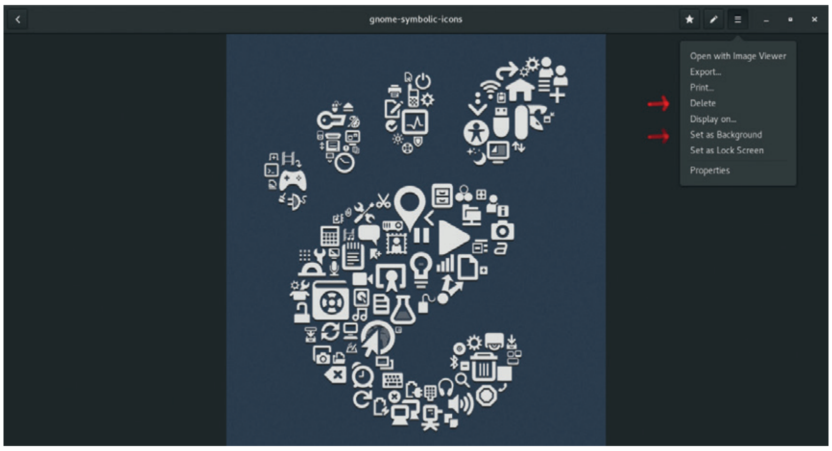
Calendar

Set an alarm (task 3): Set a reminder 10 minutes before the meeting starts, so you don't forget that meeting with your boss.

This test was easy for all participants to complete. Correspondingly, you should notice all green blocks for that row on the heat map. When working on the task, testers quickly found a way to set a reminder. No participant encountered difficulties on this task.



The empty "Albums" view in *Photos*.



The header bar menu in *Photos*.

Search for an event (task 5): You have already created an event named "GUADEC" for the GUADEC conference you are attending next month but you forgot the exact date. Can you please search for it and then tell me the date.

Users assumed that they could edit the picture using only the rations in the sidebar, so were initially confused

This task was also intuitive for all the participants. Again, notice all green blocks for this row in the heat map. All testers completed this task without difficulty. Afterwards, testers noted they found the "magnifying glass" search icon to be very familiar, which helped them in the task.

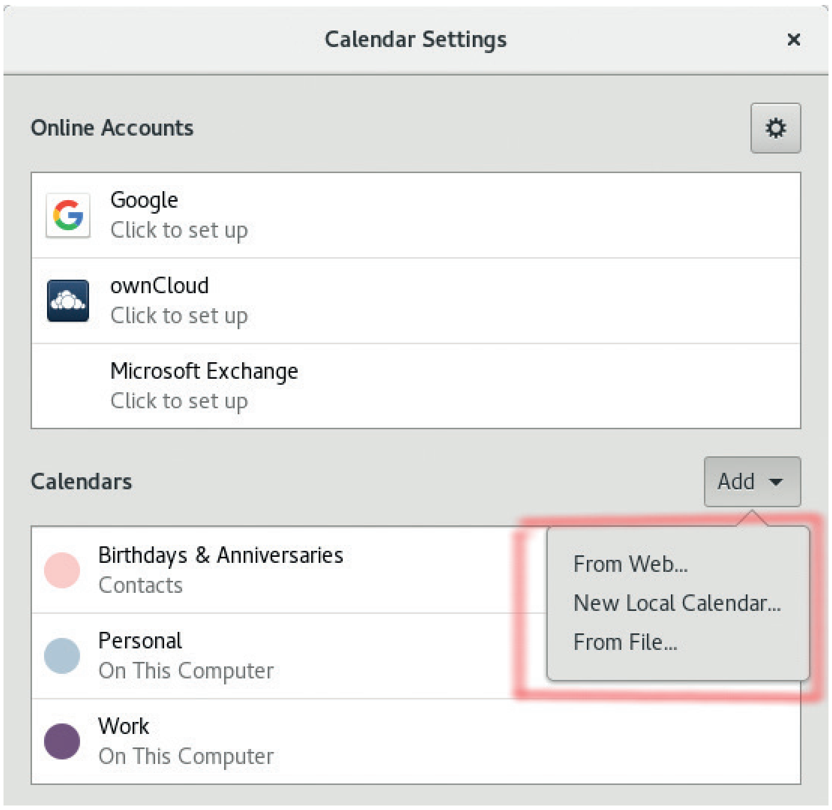
WHAT WERE THE CHALLENGES?

Photos

Create a new album (task 1): You just got back from your trip to Thailand. You want to show your friends all the pictures that you took there but you notice that they are all mixed with the other pictures that were previously in Gnome Photos. To avoid confusion, you decide to collect the pictures from your trip. Create a group of these photos and name it "Thailand trip."

This task was definitely the most challenging for testers to accomplish! Note in the heat map that this is a "hot" row, with more orange, red, and black boxes, representing difficulty and frustration in attempting the task. All participants went through similar steps to complete this one. First, they clicked "Albums" on the view switcher. Finding an empty album, testers then clicked "Name Your First Album," thinking this option would create a new album. However, the "Name Your First Album" text is not a button; instead, it is meant as a hint to the user. Testers found this unclear.

When that didn't work, testers tried to right-click in the screen to bring up a menu, but that didn't do anything either. After that, testers looked for a "Create



Calendar Settings

Online Accounts

Google
Click to set up

ownCloud
Click to set up

Microsoft Exchange
Click to set up

Calendars

Add

Birthdays & Anniversaries
Contacts

Personal
On This Computer

Work
On This Computer

From Web...
New Local Calendar...
From File...

The Calendar Settings menu in *Calendar*.

An Album" option in the Gnome "Application" menu but could not find it. Some participants also searched for "Create An Album" in the search bar.

Finally, testers right-clicked on "Photos" and found the "Add To Album" option on the bottom of the screen.

Delete a picture (task 3): While looking through all the photos, you notice that two of the photos look very similar. Please delete one of them.

and

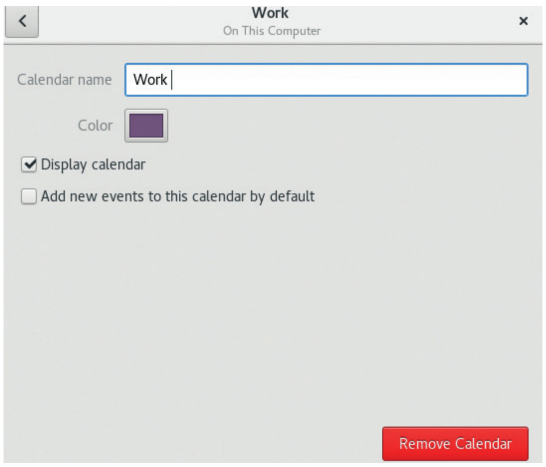
Set a background picture (task 10): You really like the way that the picture turned out and you decide to set it as a "Background picture".

These tasks were somewhat confusing for many testers, as represented in the heat map with yellow and orange boxes. For both of these tasks, most testers right-clicked on the photo and expected to get a menu where they could delete the photo or set it as a wallpaper image. Testers claimed that it would have made more sense to have a special menu that they could use to act on the picture instead of using the header bar menu.

Calendar

Add a new calendar (task 1): You want to have all your work-related activities in your calendar but you don't want them to be mixed with other activities, since you have a lot going on lately. To keep your events organized, you decide to put all your work activities on the same calendar. Make a new calendar, call it "Work," and make the activities appear in purple.

This proved to be the most difficult task in *Calendar*. In general, all participants were confused by the term



Work
On This Computer

Calendar name: Work

Color

☑ Display calendar

☐ Add new events to this calendar by default

Remove Calendar

Renaming a calendar in *Calendar*.

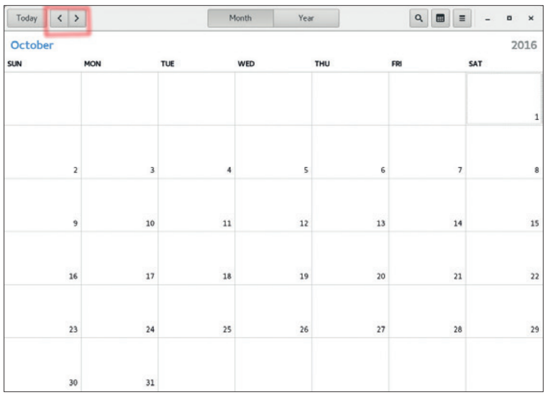
"Add A New Calendar" or "Create A New Calendar" in the menus. Testers expected to find an option like "Add a new Calendar" in the application menu and not under "Calendar Settings." Some participants even created a new event instead of a new calendar. Almost all testers were unsure which option to choose for adding a new calendar.

Testers also expected additional feedback after they named the calendar. They complained about not having an "Apply" button, since they were not quite sure if they added the new album.

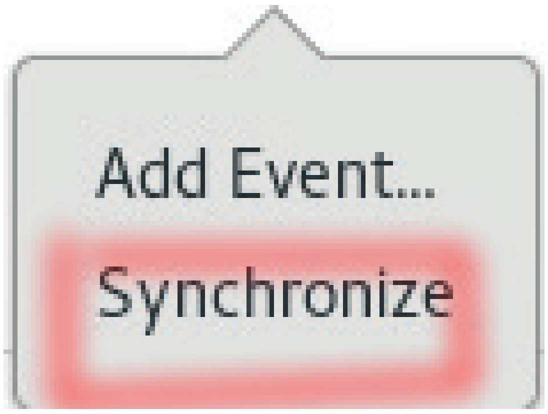
Look for a specific date (task 4): You plan to throw a big celebration party for your birthday next year. You want to check what day of the week will your birthday be, while wishing for Friday so everyone can show up! Check the calendar and tell me what day of the week your birthday will be in 2017.

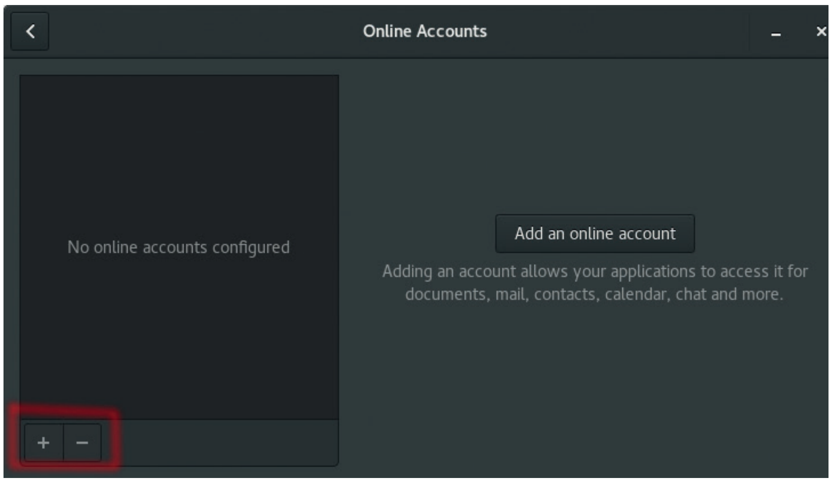
Not many testers used the view switcher to skip ahead to a specific date or year. Instead, most testers used the arrows to change the displayed date in Calendar while others tried to scroll through the months.

Add an online account (task 6): You don't want to move across different calendars, creating same events multiple times. So, you try to connect this calendar with Google Calendar (or any other online account you use).



Using the arrows to change the displayed date in *Gnome Calendar* – our users had no trouble performing this task.





Adding an account allows your applications to access it for documents, mail, contacts, calendar, chat and more.

Above left: The Synchronise option in *Gnome Calendar*.
Above: Adding and removing an online account in *Gnome Calendar*.

This was a difficult task for several testers. The first thing testers often attempted was the "Synchronise" option on header bar menu.

After the testers found a way to add an online account and finished this task successfully, they wanted to remove the account. However, the testers were confused by the "+" and "-" signs, and commented that their meaning wasn't clear. When testers figured out that the "+" and "-" buttons added and removed online accounts, testers said they would instead prefer more obvious labels such as "Add An Account" and "Remove An Account."

TESTING USABILITY IN YOUR OWN PROJECTS

As this test demonstrates, usability testing is simple. Anyone can do it! This usability test of Photos and Calendar shows how working with a few testers, and watching them use the software to do real tasks, can yield useful results.

You can apply usability testing to your own software projects very easily. Start by thinking about who uses your software, and consider why these people use your software. From this set of assumptions, jot down a representative sample of tasks that give a good idea of how your users would actually use the software in a real-world situation.

These are the scenario tasks that you can use for your usability test.

When you have your scenario tasks, ask a few people to sit down with you to do a usability test. Ask each tester to do the scenario tasks, one task at a time. Watch them, and take note of what they do. It's surprising how much you can learn just by watching a few testers use your software. With a few volunteers, you can quickly see what areas of the interface work well, and what parts of the program need more work to become easier to use.

It's surprising how much you can learn just by watching a few testers use your software – watch them and make notes

They key to usability testing is to do it iteratively. Create your design, test it, update your design based on that feedback, then test it again. At each iteration, your program will become easier to use, so anyone can use it. With good usability, everyone wins!

HOW TO CREATE A HEAT MAP

The traditional way to present usability test results is to share a summary of the test itself. What worked well? What were the challenges?

This written summary works well, and it's important to report your findings accurately, but the summary requires a lot of reading on the part of anyone who reviews the results. And it can be difficult to spot problem areas. While a well-written summary should highlight these pain points, the reality is that the reader will need to dig through the report to understand where testers ran into problems, and which areas of the software seemed to be OK.

When presenting my usability test results, I still provide a summary of the findings. But I also include a "heat map." The heat map is a simple information design tool that presents a summary of the test results in a novel way.

When creating your own heat map, follow these three simple rules:

1 Organise scenario tasks (from the usability test) in rows.
2 Arrange test participants (for each tester) in columns.
3 Represent each tester's difficulty in each scenario task with a coloured block.

The colour indicates the relative difficulty of each task for each tester:

- **Green** if the tester easily completed the task. For example, if the tester seemed to know exactly what to do, what menu item to activate or which icon to click, you would code the block in green.
- **Yellow** if the tester experienced some (but not too much) difficulty in the task.
- **Orange** if the tester had some trouble in the task. For example, if the tester had to poke around the menus for a while to find the right option, or had to hunt through toolbars and selection lists to locate the appropriate icon, you would code the block in orange.
- **Red** if the tester experienced severe difficulty in completing the task.
- **Black** if the tester was unable to figure out how to complete the task, and gave up.

The colours borrow from the standard green-yellow-red "stop light" indicators to suggest go-caution-stop. The extra orange and black colours provide gradation to additional difficulty.

The use of colours also lend the heat map its name. The gradient from "cool" colours to "hot" implies increasingly difficulty.

FAQ Snappy

Is distro fragmentation set to be a thing of the past? Let's find out.

MICHEL LOUBET JAMBERT

Q I've seen that name cropping up everywhere, but what actually is snappy? What has it got to do with Ubuntu Core?

A Loosely speaking, Snappy is a package management system, while Snap is the format of the packages. More precisely, Snappy refers to a broader variety of things, including the aforementioned snap packages, package management system, as well as Ubuntu Core, the stripped-down multi-purpose version of Ubuntu. Snappy and Ubuntu Core tend to be used interchangeably by Canonical as "Snappy Ubuntu Core," which causes some confusion. Other than the packages being designed originally for use on the distro, they aren't one and the same – one is a distribution which aims to be used for cloud computing, among other things, while the other is a package management system. We'll be focusing on the latter.

After Ubuntu Core, Snappy appeared in Ubuntu 16.04, and now the packages are making their way to other distros as well. Originally, the installation of packages was only possible from the terminal, but this later became available on the Software Centre (or rather, its Gnome replacement) for those with **xenial-proposed** enabled. For now, there's a far more limited range of software available than through the traditional means, but this is quickly changing with the likes of *VLC* and *LibreOffice* jumping on board.

Q Wait, we already have far too many package managers, why another one? And one with fewer applications for that matter?

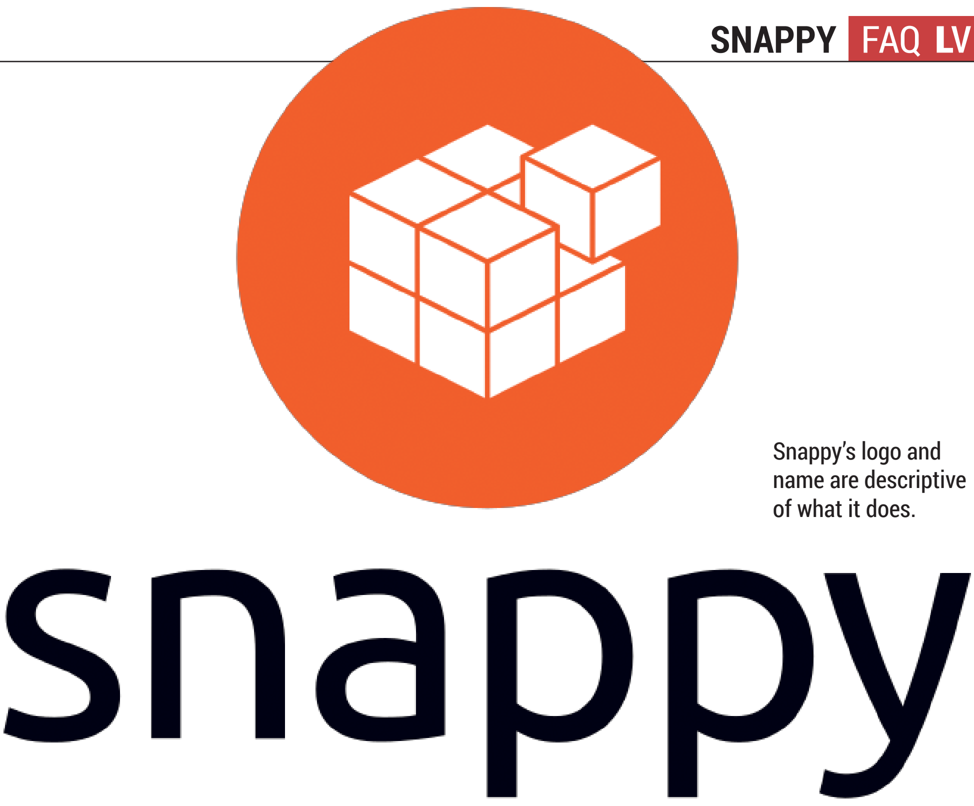
A The lack of software would change overnight if even just one major distro decided to adopt it fully, but many applications are already providing Snap packages anyway. Snappy is very different from most, but not all of these. It's an attempt at creating a cross-platform, self-contained (sandbox) orientated package manager, rather than the single-platform and dependency centred ones. The main difference over traditional package managers is that with a Snap package, everything is self contained. Unlike **apt**, for example, rather than installing a package then all the different dependencies separately, a Snap has everything contained within the one package. This means that if you have a distro with an older version of a library installed than the one needed by the application you wish to install, rather than updating everything and risking the stability of your system (if the dependencies are even compatible), the application installs its own newer version in a self-contained way. Having this kind of isolation design has the added benefit of increased security, since it prevents cross-contamination, as long as the sources are trusted, which could be an issue since there's potential for users downloading Snaps from pretty much anywhere. At the same time, it prevents fragmentation with different version numbers of the same distro, as Snaps themselves can be rolled back easily (preventing downtime) and allow installing multiple versions of the same program on one system.

Snappy is an attempt to create a cross-platform, self-contained package manager, rather than dependency-centred ones

Q I'm sold. Are there any more advantages to this?

A If you've been using Linux for a while, then you could probably think of a few off the top of your head, such as developers being able to control their updates more easily and quickly since there are no repositories in between. One positive implication could be the benefits to developers trying to bring their software over to Linux, but who might have limited experience with the operating system. A good example here would be games, with developers often encountering things like dependency issues and not understanding why a game might work perfectly well on their test systems, but why something like controller support doesn't work on another. Packaging a game in a Snap would do away with many of these kinds of issues and ensure that distro fragmentation (and associated higher support and porting costs) need no longer be barriers towards developing for the OS.

This of course applies to a slew of other areas, and even with FOSS projects, it can simply save a lot of time not having to do as much maintenance, bugfixing and testing. Overhyped "year of the Linux desktop" predictions aside, something like Snappy has the potential to seriously affect Linux adoption and development, not just from the easier and cheaper development, but also the larger amount of software available across distros. The distro developers themselves could also focus more time on core development, rather than application packaging.



Snappy's logo and name are descriptive of what it does.

Q Hurray! The end of distro fragmentation, right?

A Err... not quite. While it looks like the time of package managers being tied to distros might soon be coming to an end, along with the large number of competing ones, if you're a bit of a pessimist, this could also just be the start of a new era where we have a lot of competing cross-platform package managers rather than single-platform ones. Basically, it might be the beginning of the end for one of the causes of distro fragmentation (there are still other things such as display servers), but we might see the start of a new type of fragmentation or just a load of new package managers added to the existing pile. Flatpak (covered here in issue 30) is one such standard competing with Snappy, and Autopackage was also an attempt at cross-platform packages. Though with slightly different aims, AppImage and OrbitalApps are also projects attempting to do similar things, so Snappy is far from alone. Snap packages also don't conflict with existing package managers, so for the time being, it is likely that Snappy will coexist with other package managers on distros as an additional form in which to install software or software versions which are otherwise not supported. If this ended up being the case though, but with just one or two of the cross-platform package managers adopted alongside RPM *et al*, then the situation would probably still be rosier since there would still be increased compatibility. Whatever the case, there are big implications.

Q Wait, aren't there other solutions apart from these?

A Well yes, there are. One of them has been covered in DistroHopper, that being the method used by Bedrock Linux, which installs different distros on virtual drives that are accessed by one central Linux distro. There are various advantages and disadvantages to these solutions, but on the surface, a package manager like Snappy seems like the most elegant and doesn't mean users installing an unfamiliar distro since the advantages can be experienced across different ones. Then there are container solutions like Docker of course, which have a lot of overlap with Snappy but differ in some key areas, like Snappy being able to download changes to an application when updating, rather than downloading the whole thing.

Q I understand about Snappy working alongside other package managers, but are Apt and Yum done for in the long run?

A Hold your horses. These things don't happen overnight, and even if the big distros start adopting Snappy (or alternatives) there would still be plenty of others to hang on to the traditional package managers for whatever reason. At the same time, even if Snaps replace other packages on the application layer, the systems themselves would still be built without using Snaps, and even Ubuntu Core is built using Debs. Also, X.org is a good example of something that has stuck around for years even when Wayland has seemed imminent, due to a number of factors. Multi-distro package managers are an exciting prospect, but it's best not to make bold predictions, though if Snappy does gain ground, there would be a snowball effect.

For the time being, Snappy has to be installed through traditional means on the distros it supports, but once installed, it allows the exact same Snap package to be installed and to run on any distro. If you're eager to try it for yourself, it's already available on a number of distros, so check if yours has it and give it a spin.

"Many of the developers who are new to the community now look at open source as a standard part of software development. That is a big shift.

RACHEL ROUMELIOTIS

Thinking of organising your own conference? We get some tips from the programme chair of one of the biggest – OSCON.

We often write about OSCON, O'Reilly's Open Source Convention. It's a conference that for a long time took place in Portland, Oregon, in the north west of the United States. It was always an expensive conference to take part in, especially for Europeans, but it was also one of the best places for meeting people at the top of their game, and often, the people who helped create many of the technologies we all rely on. Last year, OSCON split into two, with one event being held in Austin, Texas, and another in Europe. Last year's European adventure was held in Amsterdam, and this year it's being held in London between 17 and 20 October.

Rachel Roumeliotis is a strategic content director at O'Reilly Media, where she edits and curates words on everything from enterprise to emerging programming languages. She's also the current programming chair for OSCON. That means she's responsible for putting together the various sessions, tutorials, keynotes and talks that make up the event, a process that starts almost as soon as the previous year's conference stops. We got a chance to ask her about how she's been putting together this year's event, and how things have been going in general for OSCON now that it's no longer bound to North America nor a single event a year.

LV How did the move to Austin go this year?

Rachel Roumeliotis: Moving to Austin was a great adventure. We had so many great years in Portland (I'm sure we will be back at some point) and had a hand in building up Portland's open source community, so it is fun to be able to be a part of an up-and-coming tech hub again. We definitely had a lot of new voices mixed with OSCON favourites, and I expect this year to be more of the same.

"We want as many different perspectives as possible" – that's why OSCON is on the move again.

LV Karen Sandler's OSCON report (https://sfconservancy.org/blog/2016/may/28/oscon-2016/) mentioned how important local outreach had been to the conference's success. How do you balance outreach – perhaps the true origins of OSCON – with the needs of corporate sponsors and proposals from people who are leaders in their fields?
RR: Good question. It all comes down to one main focus – how do we create an outstanding event for attendees? Everyone from the individual who is working on her first proposal to a well known community leader to an enterprise that has five different employees submitting and a giant booth all really want the same thing – engagement with the audience. The job of the chairs and committee is really more one of course correction than balancing. We guide our proposers via our CFP (Call For Participation) towards what we want the program to look like, and on occasion to ensure that content is focused on thought leadership and 'how-to' suggest modifications to a proposal. Getting out in front of the entire audience from community to enterprise and being clear with our intent for the conference really allows for the program to balance out quite nicely by itself.

LV How do you think attendees have changed over the last 17 years? Do you think they reflect the state of open source development?
RR: I've spoken to many different developers about open source and what I've found to be the biggest change over the better part of two decades is how developers come to be a part of the open source community, or rather how open source becomes a part of their lives.

Individuals who attended the first OSCON were often introduced to open source via Linux, which was a revelation letting developers peek under the hood. Many of the developers new to the community now look at open source as a standard part of software development. That is a big shift. What has stayed consistent is that our attendees are committed to open source. Open source needs tending and that hasn't changed since its inception.

LV What do you think the term 'open source' means today?
RR: Today 'open source' means equally innovative and quality software development. Open source allows for many individuals to contribute, which helps move software forward.

LV It feels to us that cloud-based innovation has always been a big part of OSCON, even before cloud was a thing. Do you think the current state of cloud-computing could have been predicted and where do you think it's headed?
RR: Cloud-based innovation is core to the changing world that the software developer finds herself in and so it would follow that OSCON as a reflection of the community would find it at its core as well.

Could it have been predicted? Maybe if one were to have looked at the economics of owning infrastructure and the emergence of scores of startups. Over the last decade or so the choice of tools from software frameworks to servers and now 'Whatever you need as a service' has shown that there is a lot of room for the different types of computing needs across all businesses now that all businesses are software businesses. And maybe the fact that all businesses are now software businesses is the indicator that we are going to see strong cloud computing growth for the next few decades.

Like Austin and Amsterdam, we needed to learn about London

LV How does organising a conference in London differ from Austin, or Amsterdam?
RR: As I mentioned, each OSCON is steeped in the community where it is located, so like Austin and Amsterdam, we needed to learn about London and what software communities were

If you can get your employer to pay for it, go to OSCON – nowhere else has the same buzz of knowledge and opportunity.

focused on in that city. We took into consideration the financial sector, the larger-than-normal Java community, and that fact that the big data industry is strong in London when organizing the program.

LV Do you think the locale of a conference affects the type of proposals and how you edit a conference's content?
RR: Somewhat. We generally get a lot of proposals from the local area, so as I mentioned you will see more prominently the trends of London in a CFP for a London event.

LV How do you ensure a conference has an overall coherence when including huge numbers of subjects?
RR: It is hard. With the connective tissue being open source the content can go far and wide. Choices have to be made about topics, levelling needs to be considered, and it's a constant balancing act to ensure that there are enough specifics vs overarching conceptual talks.

LV What areas of cutting-edge development or technology are you most excited by?
RR: In open source, AI is certainly of interest, as is the blockchain; both are seeing lots of action due to the core bits of technology being open source.

LV Are there any talks/sessions/keynotes you're particularly looking forward to in London (I know this is probably an impossible question to answer).
RR: For our London event, I'm actually quite interested in how the UK government has incorporated open source throughout, and we have a few sessions and keynotes highlighting aspects of that evolution.

LV Do you think technology is changing the role of publishing in ways more subtle than how content is delivered? I'm thinking of the power of 'search', where knowledge is delivered directly and perhaps only recalled for long enough to implement an idea or answer a question. But also, the nature of reading off a screen is changing the way we take in information [see https://www.theguardian.com/books/2014/aug/19/readers-absorb-less-kindles-paper-study-plot-ereader-digitisation].
RR: Ebooks mimic paper books, so as much of an evolution as they were, we are really in the same place in many ways. To this point both digital and print media are generally linear and a reader needs to seek them out. You mention search as an innovation in learning – this is true and has changed the nature of reference books. Do you need a reference book when all knowledge is online? The answer is it depends on the question and the answer.

What is now becoming the central question is how do you curate the information that a person needs; how do you push instead of pull genuinely needed info to an audience so that it is appreciated and not pushed back? Some of that is simple curation, but we are in the middle of figuring out how to be a constant resource that is always there with the right answer rather than a pile of answers one would need to search through for minutes or hours.

LV How do you think edited content can best fight for its position against social media opinion and self-publishing?
RR: I do think that working with a publisher or editor helps to make the content as succinct and clear as possible among all the noise.

LV Do you think print publishing can learn anything from the mistakes made by the music recording industry?
RR: One way we can learn is by listening to our customers and offering a variety of different ways in which to intake content; the music industry seemed to listen too late and then put all its eggs in one basket for a while.

LV Do you think it's now easier or more difficult to find good writers than perhaps 10 years ago?
RR: Good question. Honestly, the same, there are many good technologists, engineers, architects, and developers but to find one who can also express how to do what she does is an added gift and as rare as always. LV

BUY LINUXVOICE MUGS AND T-SHIRTS!

LINUXVOICE
FREE SOFTWARE | FREE SPEECH
"IT'S EASIER TO ASK FORGIVENESS THAN IT IS TO GET PERMISSION"
Rear Admiral Dr Grace Murray Hopper
GNU/
LINUXVOICE
FREE SOFTWARE FREE SPEECH
"PLAYFULLY DOING SOMETHING DIFFICULT, WHETHER USEFUL OR NOT, THAT IS HACKING."
Richard Stallman

You can drink coffee, tea or champagne. It's all about choice.

This mug respects your freedom. Look, it even says GNU/Linux!

shop.linuxvoice.com

REVIEWS

The latest software and hardware, rigorously bashed against a wall by our crack team.

Andrew Gregory

Found his first answer on Stack Overflow the other day. A portal has been opened…

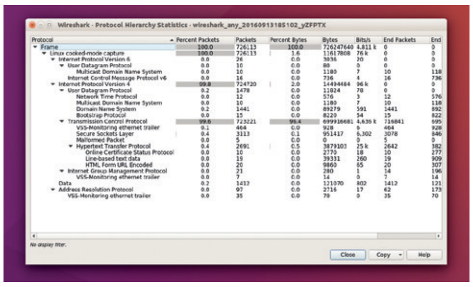
More FUD reaches us this month, brought to light thanks to the leader of HM Opposition, Jeremy Corbyn and his 'Digital Democracy Manifesto'. Some of the ideas therein are already government policy (great – when parties agree on using open formats, there must be some merit in the idea, right?). Some parts are a bit wishy-washy (I've no idea how a digital bill of rights differs from an analogue one – free expression is free expression in my bok, regardless of the medium).

Some bits are refreshingly common sense; such as the recommendation that the UK Government uses more open source. Common sense to those in the know perhaps, but to *The Sun* newspaper, the UK using open source would "let foreign spooks rob UK", elaborating that "Cyber-criminals and foreign intelligence agencies would have a field day." Of course, *The Sun's* own website runs on Wordpress, that well known North Korean spy package.

Fear, uncertainty, doubt; rabid xenophobia and ignorance – everything we expect from that filthy rag. Carry on!

andrew@linuxvoice.com

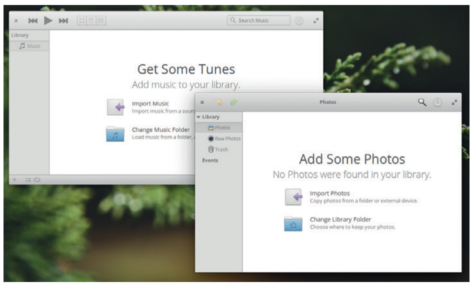
On test this issue . . .



```
Wireshark · Protocol Hierarchy Statistics · wireshark_any_20160913185102_yZFPTX

Protocol                                    Percent Packets  Packets  Percent Bytes  Bytes      Bits/s    End Packets  End
▼ Frame                                     100.0            726113   100.0          726247640  4.811 k   0            0
  ▼ Linux cooked-mode capture               100.0            726113   1.6            11617808   76 k      0            0
    ▼ Internet Protocol Version 6           0.0              26       0.0            3036       20        0            0
      ▼ User Datagram Protocol              0.0              10       0.0            80         0         0            0
          Multicast Domain Name System      0.0              10       0.0            1180       7         10           118
        Internet Control Message Protocol v6 0.0             16       0.0            736        4         16           736
    ▼ Internet Protocol Version 4           99.8             724720   2.0            14494484   96 k      0            0
      ▼ User Datagram Protocol              0.2              1478     0.0            11824      78        0            0
          Network Time Protocol             0.0              12       0.0            576        3         12           576
          Multicast Domain Name System      0.0              10       0.0            1180       7         10           118
          Domain Name System                0.2              1441     0.0            89279      591       1441         892
          Bootstrap Protocol                0.0              15       0.0            8220       54        15           822
      ▼ Transmission Control Protocol       99.6             723221   96.4           699916681  4.636 k   716841       695
          VSS-Monitoring ethernet trailer   0.1              464      0.0            928        6         464          928
          Secure Sockets Layer              0.4              3113     0.1            951417     6.302     3078         846
          Malformed Packet                  0.0              5        0.0            0          0         5            0
        ▼ Hypertext Transfer Protocol       0.4              2691     0.5            3879103    25 k      2642         382
            Online Certificate Status Protocol 0.0           10       0.0            2770       18        10           277
            Line-based text data            0.0              19       0.0            39331      260       19           909
            HTML Form URL Encoded           0.0              20       0.0            9860       65        20           307
      ▼ Internet Group Management Protocol  0.0              21       0.0            280        1         14           196
          VSS-Monitoring ethernet trailer   0.0              7        0.0            14         0         7            14
      Data                                  0.2              1412     0.0            121070     802       1412         121
    ▼ Address Resolution Protocol           0.0              97       0.0            2716       17        62           173
        VSS-Monitoring ethernet trailer     0.0              35       0.0            70         0         35           70

No display filter.
```

Wireshark 2.2 **46**
Keep your electric eye on what's passing over your network. And by network we mean everything!

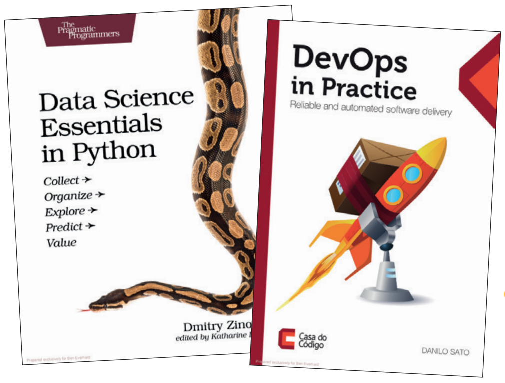
OpenBSD 6.0 **47**
Linux is secure, but this Unix derivative (and the home of OpenSSH) is utterly bomb-proof.



Elementary OS: Loki **48**
Pretty is a feature for some; for Elementary it's an ethos. Newbies, try it today and fall in love.

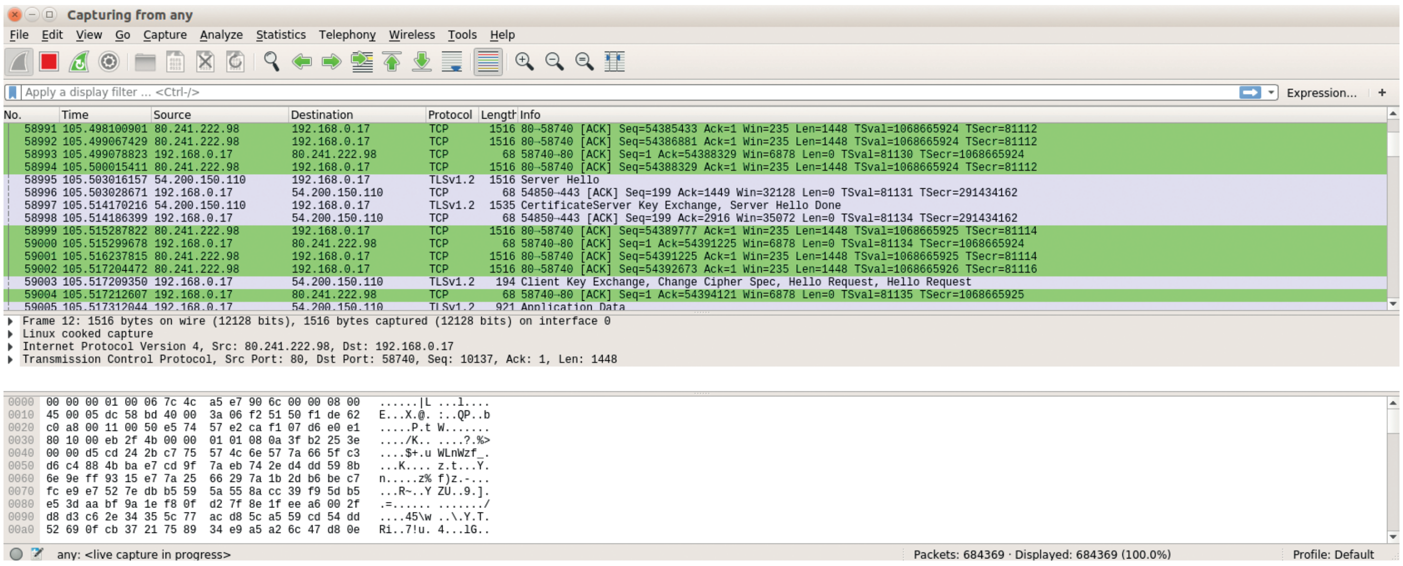
Ardour 5.4 **49**
Now with Windows support, a cleaner interface, and loads more features for audio geeks to play with.

Group test and books



Boooooooooooooooks!!!! **52**
In which Ben Everard wonders what kinds of science don't use data, and benefits from the narrow focus of a DevOps case study.

Group test – Online file storage **54**
WIth *OwnCloud*'s fork, there's suddenly a plethora of platforms all fighting to get hold of your bits. Find the best/most convenient/easiest one for you.


Capturing from any

Apply a display filter ... <Ctrl-/>

```
No.   Time          Source          Destination     Protocol Length Info
58991 105.498100901 80.241.222.98   192.168.0.17    TCP      1516 80→58740 [ACK] Seq=54385433 Ack=1 Win=235 Len=1448 TSval=1068665924 TSecr=81112
58992 105.499067429 80.241.222.98   192.168.0.17    TCP      1516 80→58740 [ACK] Seq=54386881 Ack=1 Win=235 Len=1448 TSval=1068665924 TSecr=81112
58993 105.499078823 192.168.0.17    80.241.222.98   TCP        68 58740→80 [ACK] Seq=1 Ack=54388329 Win=6878 Len=0 TSval=81130 TSecr=1068665924
58994 105.500015411 80.241.222.98   192.168.0.17    TCP      1516 80→58740 [ACK] Seq=54388329 Ack=1 Win=235 Len=1448 TSval=1068665924 TSecr=81112
58995 105.503016157 54.200.150.110  192.168.0.17    TLSv1.2  1516 Server Hello
58996 105.503028671 192.168.0.17    54.200.150.110  TCP        68 54850→443 [ACK] Seq=199 Ack=1449 Win=32128 Len=0 TSval=81131 TSecr=291434162
58997 105.514170216 54.200.150.110  192.168.0.17    TLSv1.2  1535 CertificateServer Key Exchange, Server Hello Done
58998 105.514186399 192.168.0.17    54.200.150.110  TCP        68 54850→443 [ACK] Seq=199 Ack=2916 Win=35072 Len=0 TSval=81134 TSecr=291434162
58999 105.515287822 80.241.222.98   192.168.0.17    TCP      1516 80→58740 [ACK] Seq=54389777 Ack=1 Win=235 Len=1448 TSval=1068665925 TSecr=81114
59000 105.515299678 192.168.0.17    80.241.222.98   TCP        68 58740→80 [ACK] Seq=1 Ack=54391225 Win=6878 Len=0 TSval=81134 TSecr=1068665924
59001 105.516237815 80.241.222.98   192.168.0.17    TCP      1516 80→58740 [ACK] Seq=54391225 Ack=1 Win=235 Len=1448 TSval=1068665925 TSecr=81114
59002 105.517204472 80.241.222.98   192.168.0.17    TCP      1516 80→58740 [ACK] Seq=54392673 Ack=1 Win=235 Len=1448 TSval=1068665926 TSecr=81116
59003 105.517209350 192.168.0.17    54.200.150.110  TLSv1.2   194 Client Key Exchange, Change Cipher Spec, Hello Request, Hello Request
59004 105.517212607 192.168.0.17    80.241.222.98   TCP        68 58740→80 [ACK] Seq=1 Ack=54394121 Win=6878 Len=0 TSval=81135 TSecr=1068665925
59005 105.517312044 192.168.0.17    54.200.150.110  TLSv1.2   921 Application Data

▸ Frame 12: 1516 bytes on wire (12128 bits), 1516 bytes captured (12128 bits) on interface 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 80.241.222.98, Dst: 192.168.0.17
▸ Transmission Control Protocol, Src Port: 80, Dst Port: 58740, Seq: 10137, Ack: 1, Len: 1448

0000  00 00 00 01 00 06 7c 4c  a5 e7 90 6c 00 00 08 00   ......|L ...l....
0010  45 00 05 dc 58 bd 40 00  3a 06 f2 51 50 f1 de 62   E...X.@. :..QP..b
0020  c0 a8 00 11 00 50 e5 74  57 e2 ca f1 07 d6 e0 e1   .....P.t W.......
0030  80 10 00 eb 2f 4b 00 00  01 01 08 0a 3f b2 25 3e   ..../K.. ....?.%>
0040  00 00 d5 cd 24 2b c7 75  57 4c 6e 57 7a 66 5f c3   ....$+.u WLnWzf_.
0050  d6 c4 88 4b ba e7 cd 9f  7a eb 74 2e d4 dd 59 8b   ...K.... z.t...Y.
0060  6e 9e ff 93 15 e7 7a 25  66 29 7a 1b 2d b6 be c7   n.....z% f)z.-...
0070  fc e9 e7 52 7e db b5 59  5a 55 8a cc 39 f9 5d b5   ...R~..Y ZU..9.].
0080  e5 3d aa bf 9a 1e f8 0f  d2 7f 8e 1f ee a6 00 2f   .=...... ......./
0090  d8 d3 c6 2e 34 35 5c 77  ac d8 5c a5 59 cd 54 dd   ....45\w ..\.Y.T.
00a0  52 69 0f cb 37 21 75 89  34 e9 a5 a2 6c 47 d8 0e   Ri..7!u. 4...lG..
```

any: <live capture in progress> Packets: 684369 · Displayed: 684369 (100.0%) Profile: Default

Wireshark 2.2

Thalassophobic **Ben Everard** now fears large marine animals in cat-5 cables.

Web https://www.wireshark.org
Developer Gerald Combs and the Wireshark team
Licence GPL

Wireshark is our tool of choice for investigating anything relating to data on a network. When capturing, it hoovers up every piece of data that passes through your network interfaces and displays them in a list. When we say network, we mean every form of data going into or out of your computer. That includes Ethernet, as you may expect, but also interfaces not commonly thought of as networks, such as USB and Bluetooth. This can make it useful for diagnosing bugs in peripheral connections as well as on networks. There's a command line version (*Tshark*), and graphical versions for both *GTK* (the toolkit used to build the Gnome desktop) and *Qt* (the toolkit used in KDE), so there's a version to fit in with almost every Linux environment. Despite the different user interfaces, they all work in the same way.

> The power of Wireshark is that it can analyse almost any network protocol to see what's actually being sent

The real power of *Wireshark* isn't that it collects network data – there area loads of tools that do this – but that it can analyse almost any network protocol to see what's actually being sent. If your networked application is simply giving an error like "can't connect", *Wireshark* will be able to hunt down exactly where the connection problem is happening. We've also found *Wireshark* useful for finding out exactly what data is being sent from an application. If you're having performance problems with your network, the analysis tools in *Wireshark* can help you track down exactly where this is originating.

Version 2.2 comes with the ability to export directly into Elasticsearch-compatible JSON, which means that you can take advantage of the visualising power of the Kibana data visualisation tool to provide interactive real-time graphing of your networking. There's also improved SSL capabilities and a whole bunch more protocols supported.

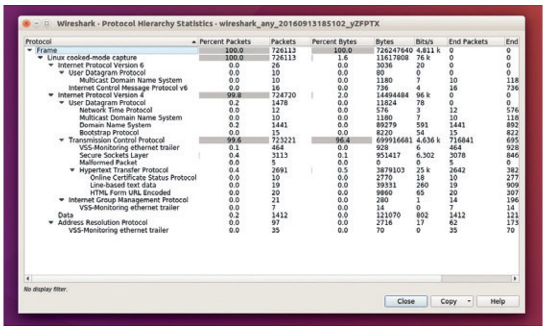
Data overload

While it's a hugely powerful tool, *Wireshark* isn't for the faint-hearted. Unless you understand how the different communication protocols relate to one another, you'll struggle to make sense out of the data it's showing you. That said, if you're interested in learning more about how network protocols work, investigating them with *Wireshark* is a great way to find out more about what's going on.

While version 2.2 may not be the biggest upgrade, *Wireshark* remains our tool of choice for working with network packets.

The best tool for investigating network issues.

★★★★★


Wireshark · Protocol Hierarchy Statistics · wireshark_any_20160913185102_yZFPTX

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End
▾ Frame	100.0	726113	100.0	726247640	4,811 k	0	0
▾ Linux cooked-mode capture	100.0	726113	1.6	11617808	76 k	0	0
▾ Internet Protocol Version 6	0.0	26	0.0	3036	20	0	0
▾ User Datagram Protocol	0.0	10	0.0	80	0	0	0
Multicast Domain Name System	0.0	10	0.0	1180	7	10	118
Internet Control Message Protocol v6	0.0	16	0.0	736	4	16	736
▾ Internet Protocol Version 4	99.8	724720	2.0	14494484	96 k	0	0
▾ User Datagram Protocol	0.2	1478	0.0	11824	78	0	0
Network Time Protocol	0.0	12	0.0	576	3	12	576
Multicast Domain Name System	0.0	10	0.0	1180	7	10	118
Domain Name System	0.2	1441	0.0	89279	591	1441	892
Bootstrap Protocol	0.0	15	0.0	8220	54	15	822
▾ Transmission Control Protocol	99.6	723221	96.4	699916681	4,636 k	716841	695
VSS-Monitoring ethernet trailer	0.1	464	0.0	928	6	464	928
Secure Sockets Layer	0.4	3113	0.1	951417	6,302	3078	846
Malformed Packet	0.0	5	0.0	0	0	5	0
▾ Hypertext Transfer Protocol	0.4	2691	0.5	3879103	25 k	2642	382
Online Certificate Status Protocol	0.0	10	0.0	2770	18	10	277
Line-based text data	0.0	19	0.0	39331	260	19	909
HTML Form URL Encoded	0.0	20	0.0	9860	65	20	307
▾ Internet Group Management Protocol	0.0	21	0.0	280	1	14	196
VSS-Monitoring ethernet trailer	0.0	7	0.0	14	0	7	14
Data	0.2	1412	0.0	121070	802	1412	121
▾ Address Resolution Protocol	0.0	97	0.0	2716	17	62	173
VSS-Monitoring ethernet trailer	0.0	35	0.0	70	0	35	70

No display filter.

The protocol hierarchy statistics show how many packets there are at each level of protocol running across the network.

OpenBSD 6.0

Ultra-secure Mike Saunders investigates the performance of this Unix derivative.

Bit rot is a common problem in software development. You may run an awesome Free Software project, and be tempted to roll in every patch you receive, but who's going to maintain the code in the long run? Sometimes the things you leave out – or indeed the code you remove – is as important as the new features you add. And this is true of the latest release of OpenBSD, the security-focused Unix flavour that's also the home of *OpenSSH* and other well-known tools.

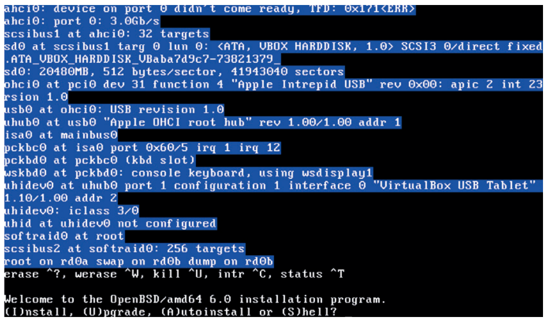
OpenBSD 6.0 drops support for the ancient VAX architecture (you know, those big white machines that are easy to confuse with fridges), along with support for running Linux binaries by translating system calls. Pretty much nobody used the former feature, while the latter was unmaintained and suffering from the aforementioned bit rot. As the OpenBSD team is so heavily focused on security, removing any old and dust-ridden code is part of the process of a release.

Selective improvements

But what's actually new in this version? Well, W^X (write-or-execute) is enabled by default for the base system. This is a security measure that says that a chunk of memory can be written to, or executed, but not both – limiting the attack surface of the OS. If you need to run a program that won't work with W^X enabled, you can set a flag and mount the filesystem that contains it in a specific manner. This may sound like extra hassle, but OpenBSD isn't there to make your life easy – it's to make you secure.

Then there have been improvements to SMP, aka multi-processor support. OpenBSD has historically been weak in this area, especially when compared to Linux or FreeBSD, but progress is being made. The network stack has been improved and for desktop users, web browsing is now a tad smoother.

We use Linux for most things, but we love the strong focus, the attention to detail, and the tools that OpenBSD produces for all OSes. We just hope to see some kind of official binary update system in the next few years.



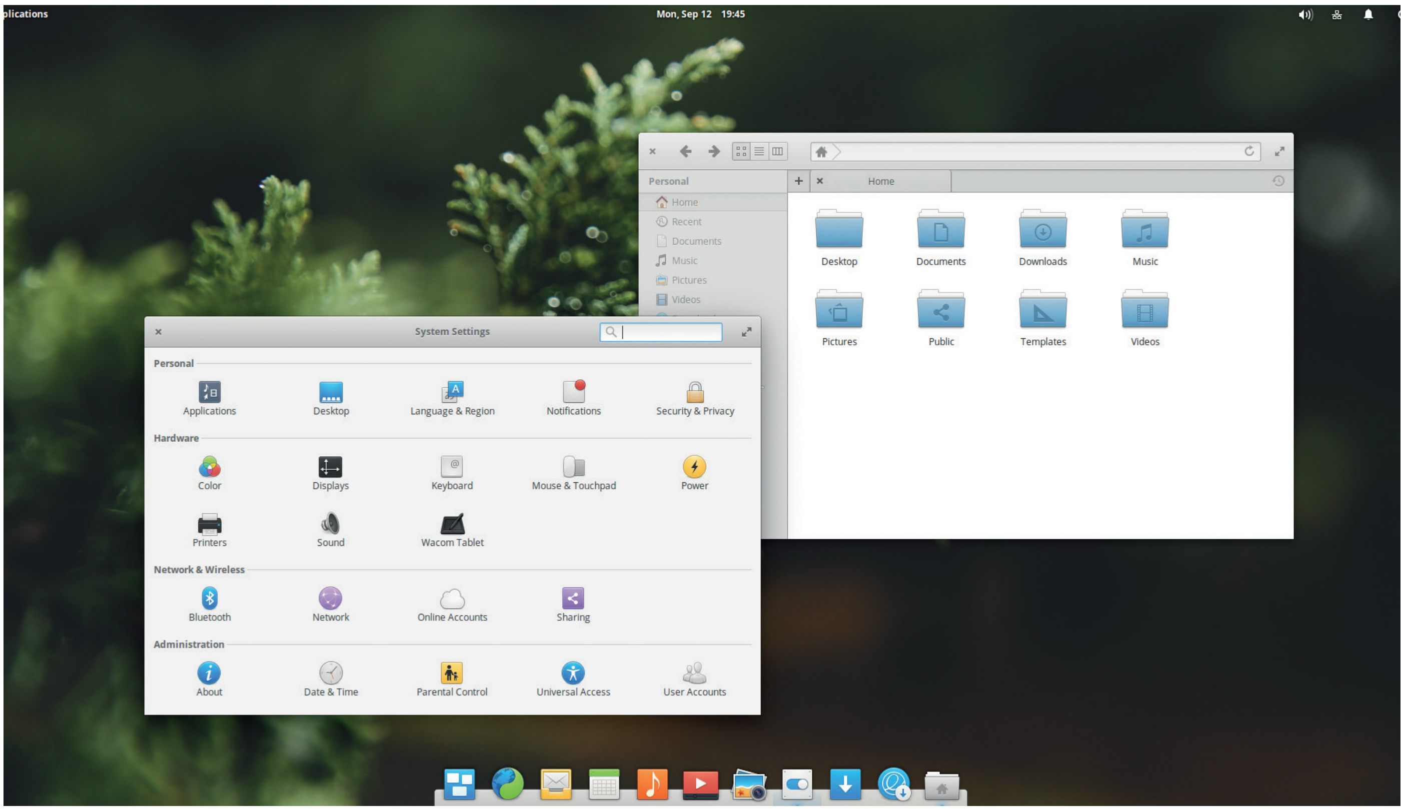
```
ahci0: device on port 0 didn't come ready, TFD: 0x171<ERR>
ahci0: port 0: 3.0Gb/s
scsibus1 at ahci0: 32 targets
sd0 at scsibus1 targ 0 lun 0: <ATA, VBOX HARDDISK, 1.0> SCSI3 0/direct fixed
.ATA_VBOX_HARDDISK_VBaba7d9c7-73821379_
sd0: 20480MB, 512 bytes/sector, 41943040 sectors
ohci0 at pci0 dev 31 function 4 "Apple Intrepid USB" rev 0x00: apic 2 int 23
rsion 1.0
usb0 at ohci0: USB revision 1.0
uhub0 at usb0 "Apple OHCI root hub" rev 1.00/1.00 addr 1
isa0 at mainbus0
pckbc0 at isa0 port 0x60/5 irq 1 irq 12
pckbd0 at pckbc0 (kbd slot)
wskbd0 at pckbd0: console keyboard, using wsdisplay1
uhidev0 at uhub0 port 1 configuration 1 interface 0 "VirtualBox USB Tablet"
1.10/1.00 addr 2
uhidev0: iclass 3/0
uhid at uhidev0 not configured
softraid0 at root
scsibus2 at softraid0: 256 targets
root on rd0a swap on rd0b dump on rd0b
erase ^?, werase ^W, kill ^U, intr ^C, status ^T

Welcome to the OpenBSD/amd64 6.0 installation program.
(I)nstall, (U)pgrade, (A)utoinstall or (S)hell?
```

OpenBSD's installer is a plain text affair, but gets the job done quickly.

Web www.openbsd.org
Platforms x86, amd64, SPARC, ARM, PowerPC
License BSD (some parts GPL)

A spoonful of extra security and a nice glass of performance improvements to wash it down. The release songs are good too.

★★★★☆



Elementary OS: Loki

Creating a beautiful distro? It's [gratuitous pun redacted] my dear **Ben Everard**.

Web https://elementary.io
Developers The Elementary Team
Licence Various open source licences

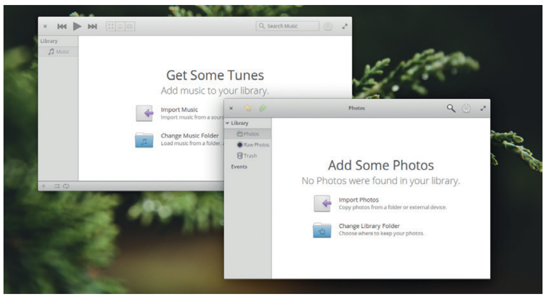
A well designed tool is a joy to use, whether it's a physical item like a perfectly weighted and sharpened chef's knife or a text editor that feels like an extension of your brain. The better the design, the closer the link between your intention and the tool's action. Good user interfaces present the user with all the information they need to use the software, but not be overwhelming. They should also look good, but this is secondary to ease of use. Elementary OS claims to be a design-oriented Linux-based OS, so it's this standard that we will judge them by.

The attention to detail in the design of Elementary will definitely help new users. For example, open the *Music* or *Photos* application and you'll be greeted with the same message telling you how to add your files. This consistency comes from design of the OS as a whole, not just individual components. Our only quibble with the default applications is the web browser. *Epiphany* is a good browser, but it's not in the same league as *Firefox* or *Chrome*.

The designers have made good use of the new features in *GTK 3* particularly the Header Bars that enable developers to put more tools in the top bar of the window. It would be easy to overdo this and end up with cluttered windows, but by focusing on the core uses of the application, the Elementary team have made the basic functions easy to access.

Unity is strength

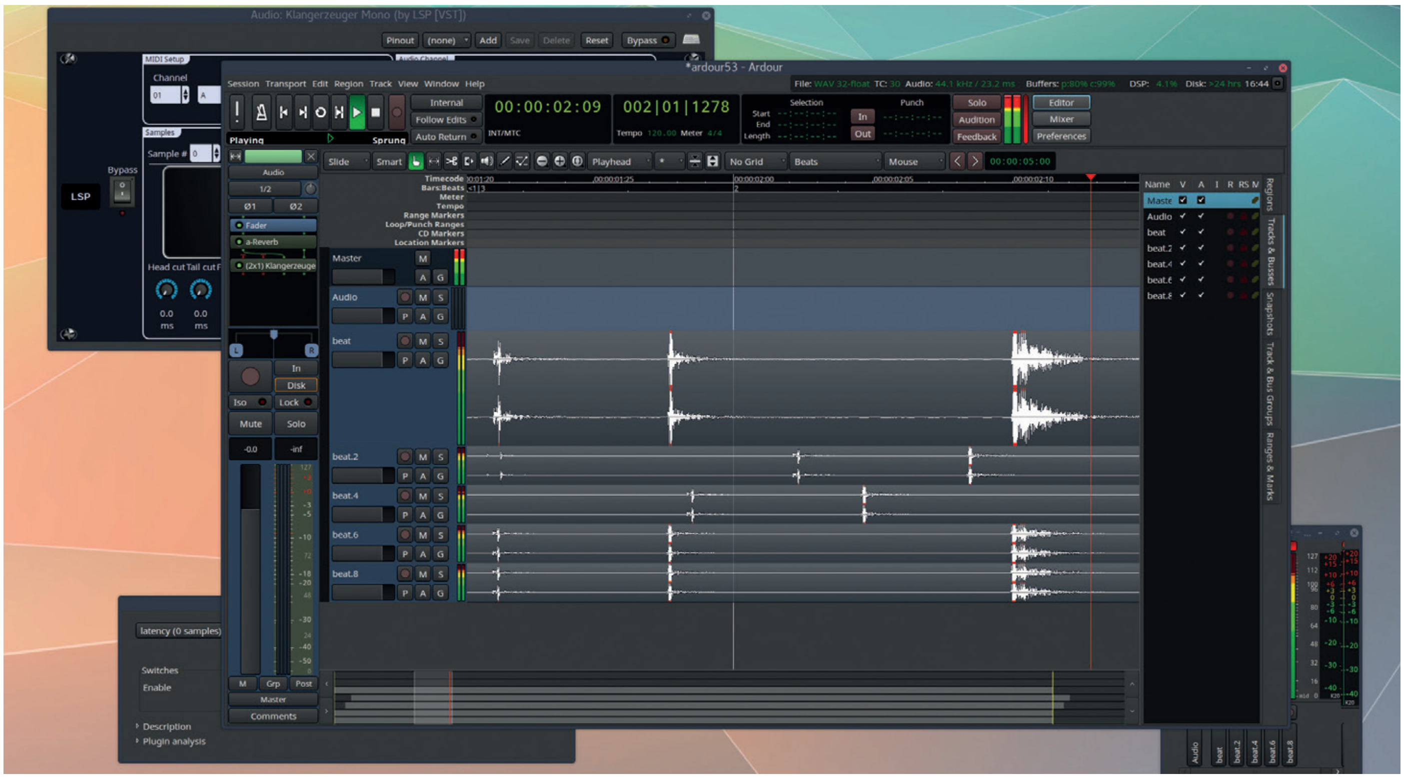
The look and feel of Elementary permeates all of the default applications – this means that if you want to use, for example, a different music player or email application, you quickly lose the smooth feel that comes from having the full *Elementary* experience. If you're set in your ways about which applications you like to use for which tasks, you won't really get the benefits of this distro, but then you're not really the target either – 75% of the downloads of Elementary come from proprietary operating systems, and for these people, new to Linux, Elementary OS is a fantastic introduction to what Linux can do.



The similarity between the pictures and music applications reduces the amount that a new user has to learn.

The design focus has built the most graphically-coherent Linux distribution that looks good and is easy to use.

★★★★★



Ardour 5.3

Graham Morrison goes gaga over the latest update from Paul Davis.

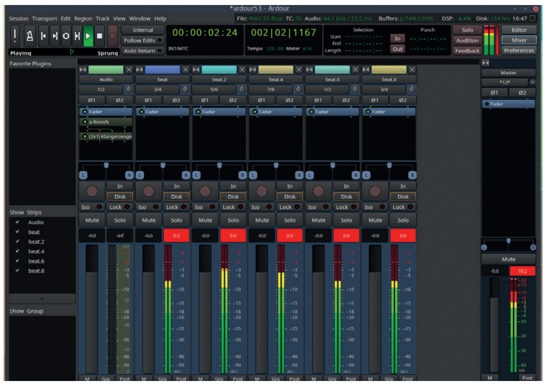
Ardour 5 is the biggest update to our favourite audio workstation software we've seen. Its biggest new feature is support for Microsoft Windows. This might not seem so important to us Linux users, but it will result in more people using *Ardour*, and consequently, more support and development going into future releases for all platforms. This is what happened when OS X support was added, and it's the cross-platform nature of *Ardour* that's been able to help its creator, Paul Davis, fund *Ardour*'s new features, such as the new tabbed interface. You can now switch between the Editor, Mixer and Preferences views while keeping the top panel in place, which makes the application feel more more like a professional DAW (Digital Audio Workstation) on other platforms.

Oh L'Amour

Equally important, and something we've been asking for for years, is the inclusion of some basic plugins. Previously, you needed to install your own plugins before you could do anything with *Ardour*. But the new 4-band parametric EQ, side-chain compressor, delay, reverb and filter effects fit the task perfectly, and will perform 90% of your day-to-day editing duties.

Another major update is a completely revised OSC (Open Sound Control) interface, making it much easier to remotely control *Ardour*, or create profiles for Android tablets and hardware controllers. This would help if you were building your own studio, for example, or wanted to incorporate esoteric touch panels, knobs and sliders into your recordings. Finally, there are five new themes for *Ardour*'s excellent theming engine, helping the application fit into your desktop.

All of which makes this the best update and release of *Ardour* we've seen. If you find *Ardour* as brilliant as we do, don't forget that it's an open source application funded by users who subscribe to updates, and we'd highly recommend doing this if you'd otherwise be buying a costing proprietary application, as it will mean more updates like this in the future.



A major update to *Ardour*'s user interface is the ability to tab between mixer and editor views.

Web https://ardour.org
Platforms Paul Davis
Licence GPLv2

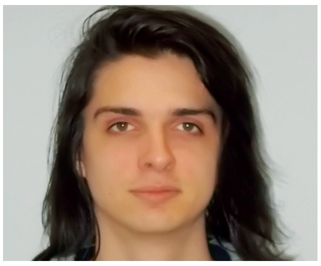
Awesome.

★★★★★

GAMING ON LINUX

The tastiest brain candy to relax those tired neurons

O TEMPORA! O MORES!



Michel Loubet-Jambert is our Games Editor. He hasn't had a decent night's sleep since Steam came out on Linux.

One ongoing trend in the world of games development is that Linux ports that have been promised are just not getting made. Some of the major casualties include *Project CARS* and *Batman: Arkham Knight*, though the latter of these may also be due to the negative critical reception it received. Though there isn't a huge deal to be done about market share (one of the factors to which we can attribute the lack of developer take-up), with porting we can hope that in an increasingly multi-platform world, developers will plan their work around that from the start.

There's more news on the Vulkan API as it continues to expand, with a few open source projects now adopting it. The original *Quake* is one such game to get the Vulkan treatment. There are still some things missing, but it should be up and running for those who want to try it. The same goes for the *Dolphin* emulator, which enables running Gamecube and Wii games on desktop systems. In this case, Vulkan performance is slightly better for Nvidia cards than on OpenGL, and around 25% faster on AMD cards.

Some more nice news is the release of *Godot 2.1*, with the game engine focusing on improving the editor to make game development more user-friendly. More languages have also been added, opening up development to more people who wish to make games entirely on open source software.

Life Is Strange

A gripping story with twists, turns and mystery.

Web http://store.steampowered.com/app/319630
Price £15.99 (all 5 episodes)

This incredibly story-rich episodic adventure game has now made its way onto Linux, being one of the first of the big series of this type get ported. For those not aware, these kinds of episodic story games are a fairly recent phenomenon and are somewhat comparable to the high-budget television miniseries which have become increasingly popular over the years. Their role is to tell a gripping story, using the medium of video games to their advantage, such as player choice and exploration to do things not possible in front of the television. *Life Is Strange* does just that, delivering an excellent story, memorable characters and great voice acting which helps further the immersion.

The game puts the player in the shoes of Max, an 18-year-old photography student who can turn back time. The main premise is that, by using these abilities, she can stop or alter unfortunate events that are going on around her. As such, the game deals with some pretty heavy themes, though manages to do so in a dignified and mature manner. If you like story in games, it's hard to pick too many holes in *Life Is Strange*, with the only obvious caveat being that turning back time and seeing possible outcomes tends to detract from the tension that comes from having less control over a situation. That said, it is still a rollercoaster of a story with a replay value not found in linear narratives.

Rather than spoil the story, it's best to try it out for yourself given that the first episode is now free, which should hook most people into buying the other four – all of which are available now.

Through Max, the player can examine every nook and cranny of Arcadia bay.

The game's characters feel very human, each with their own agendas or issues.

Stardew Valley

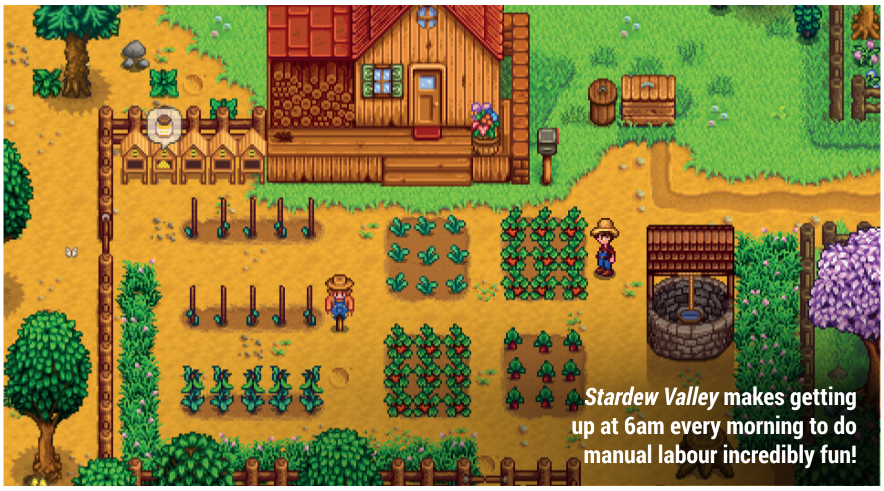
Like Harvest Moon, but bigger and better.

Web http://store.steampowered.com/app/413150
Price £10.99

Harvest Moon is a game many rank among their favourites, but the series lost its way when it made the transition to 3D. *Stardew Valley* addresses that, by providing a spiritual successor which brings the series back to its roots.

The game follows the usual story of inheriting a farm from a relative, but now with the added backstory of the protagonist wasting their life away in a soulless corporation before embarking on a new life full of village folk and cabbages. The farm begins abandoned and the player slowly builds it up, following a mesmerising routine, with side-stories and intrigue thrown in.

Though the formula remains similar, some aspects have been seriously refined. Simply holding a turnip above your head and giving it to your love interest every day until they give up and marry you doesn't work anymore, since superior AI makes things trickier and a lot more rewarding.


Stardew Valley makes getting up at 6am every morning to do manual labour incredibly fun!

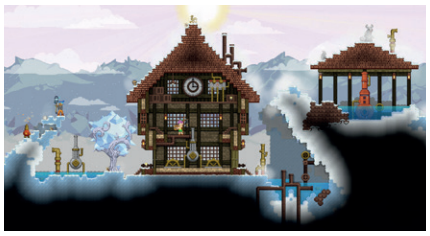
Starbound

A point-and-click classic brought back to life.

Web http://store.steampowered.com/app/211820
Price £11.99

After being in Early Access for just over three years, *Starbound* has now hit 1.0, offering a feature-complete and stable game for those who haven't picked it up. The sandbox/survival/crafting game revolves around a space traveller who has become lost and must land on different planets to gather fuel and supplies for their ship in order to continue the journey through space. There's a good dose of exploration and some brutal *Rogue*-like elements.

A lot has changed since it launched around the same time as Steam came to Linux, so for those who liked it before, chances are you'll like it even more now. For potential newcomers, it's hard to think of those who *Starbound* wouldn't please


Starbound's 16-bit graphics feel out of place at times, but still work well.

given that it has it all: a fleshed-out universe with different races and lore, a satisfying story, retro side-scrolling combat, and the freedom associated with sandbox games.

Where *Starbound* sets itself apart from other sandbox games is with the story and lore, giving a much longer shelf life and replay value to a genre that often gets stale quickly.

ALSO RELEASED…

Human: Fall Flat
This very stylish third person puzzle-platformer combines its wobbly physics and mesmerising dream world to provide an extremely pleasant and minimalistic experience with a charming aesthetic appeal. The controls are extremely unorthodox (even occasionally annoying) and add challenge to the otherwise straightforward, mostly physics-based puzzles.
http://store.steampowered.com/app/477160

Doorways: Holy Mountains of Flesh
The latest installment of the *Doorways* horror game series has come out of Early Access. Set in the Argentine province of Salta, the game is full to the brim of atmosphere through the clever use of sound, ambience and music. It focuses on psychological horror over cheap jump scares (though there are a few) and has a decent story, which is enhanced by playing the previous games in the series.
http://store.steampowered.com/app/383930

Overlord I & II
The *Overlord* series has made its way onto Linux with both the original games and the *Raising Hell* expansion. The series turns the fantasy genre around as the player controls the Overlord, a sort of Sauron-like figure who releases his minions to unleash havoc on and destroy an otherwise peaceful kingdom. *Overlord II* is the better game, but both are worth the price for a good dose of silly fun.
http://store.steampowered.com/app/12810

Data Science Essentials In Python

Ben Everard is still trying to work out what sort of science doesn't use data.

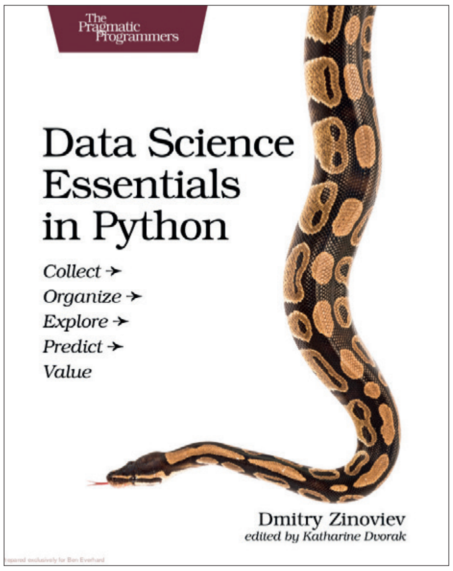
Author Dmitry Zinoviev
Publisher O'Reilly
Price £22.38
ISBN 978-1680501841

Data science – the process of squeezing meaning from large amounts of uncooperative data – became a tech buzzword shortly after Big Data entered the business lexicon.

Data Science Essentials In Python goes through most of the Python features you'll need if you want to analyse data. You'll learn about text processing, working with large amounts of data, tools for manipulating data, ways of making it look pretty and modules that help with statistical analysis. Rather than look at what this book does cover, it's perhaps more instructive to look at two things it doesn't: Data science and Python programming. This book takes you through how to apply data science techniques in Python. It doesn't teach you what techniques you should apply and when you should apply them. If you don't already understand the mathematics behind data science, all this book will teach you is how to generate graphs. Likewise, this book only covers the extra bits of Python that apply to Data Science that regular Python programmers may not know, such as how to use analytic modules. None of this should be taken as a criticism – far from it – by focusing on one aspect, Dmitry Zinoviev has made a book that is not just a useful introduction, but a useful reference guide to flick back to as you're working with data.

Get to grips with the Python you need for data science.

★★★★☆


The Pragmatic Programmers
Data Science Essentials in Python
Collect →
Organize →
Explore →
Predict →
Value
Dmitry Zinoviev
edited by Katharine Dvorak

Pythons only need to eat four or five times per year. This means they can spend almost all their time analysing data rather than hunting for wild pigs.

DevOps in Practice

Ben Everard feels DevOps should be renamed DevTOps. Testing is important too!

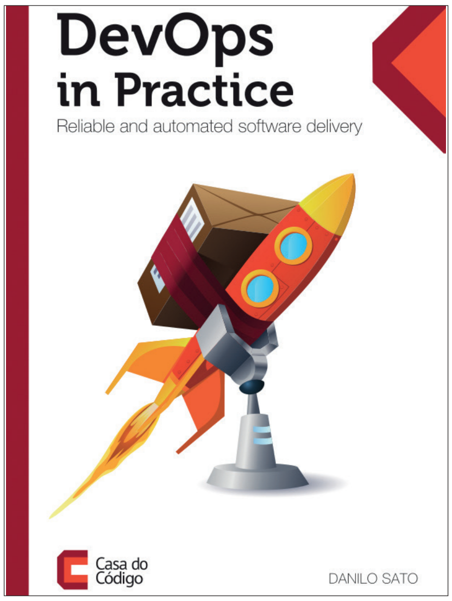
Author Danilo Sato
Publisher Casa do Código
Price £13.99
ISBN none

At the intersection of development and operations lies a brave group of men and women trying to please two opposing sides: operations want servers to keep running smoothly without interruptions; developers want to push out new code that may or may not break the production environment. DevOps achieves this squaring of the circle by carefully applying continuous integration, monitoring and automated provisioning and build systems.

There are a wide range of tools and utilities out there to help, none of which has yet established itself as a *de facto* standard. *DevOps In Practice* selects one chain of tools (Vagrant, Jenkins, Nagios and Puppet), and shows the entire process of setting up and maintaining a production server of a non-trivial example (the *Broadleaf* e-commerce platform). The entire book is built around this case study, which means that it gives a thorough investigation of this problem and it becomes easy to understand how everything fits together, so if you're looking for a book to introduce you to the technical side of this field, *DevOps In Practice* does a great job of showing the end-to-end process. The downside of this is that it focuses quite heavily on these specific tools, so if they're not the ones that your organisation uses, a different book might suit you better.

A case study that shows how DevOps can help an organisation manage its production environment.

★★★★☆



Strapping code to a rocket and firing it off into space is an example of what's not DevOps.

MEDECINS SANS FRONTIERES
DOCTORS WITHOUT BORDERS

The world's A&E Department

Find out more at msf.org.uk

GROUP TEST

Not willing to trust his collection of cat videos to remote storage silos, **Mayank Sharma** looks at options to set up his own.

On test

Cozy

URL https://cozy.io
Licence MIT/X11
Latest release 2.5+
Can it manage your personal data comfortably?

NextCloud

URL https://nextcloud.com
Licence AGPL v3
Latest release 9.0.53
Is this newly forked project any good?

OwnCloud Community Edition

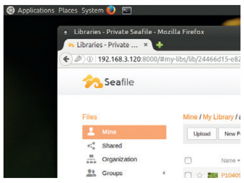
URL www.owncloud.org
Licence AGPL v3
Latest release 9.1
Can it survive past the fork?

Pydio

URL https://pydio.com
Licence AGPL
Latest release 6.4.2
It advertises the users of its Enterprise edition, but is the freely-available edition any good?

Seafile Community Edition

URL www.seafile.com
Licence GPL v2
Latest release 5.1.4
Another option that presents a feature-restricted community edition.



SparkleShare

URL www.sparkleshare.org
Licence GPL v3
Latest release 1.5.0
It's based on Git, but does it get it any advantage?

Hosted storage servers

Online storage services such as Dropbox offer a convenient option for accessing and sharing data anywhere on the planet. Yet the convenience comes at a cost, and the very idea of transferring our files to a remote server, outside of our jurisdiction, seems quaint in the post-Snowden era.

Then there's the matter of cost as well. While many cloud services provide a free basic account, they usually come with limited storage space and only offer a minimum of services. If you want to go beyond these you'll usually have to sign up for a paid subscription, the costs of which can mount up.

This is where personal cloud storage software step in. These tools offer all the conveniences of an omnipresent storage service while keeping you in charge of your private data. The data never leaves the confines of your computer (or more accurately your home network) and yet is accessible through any device connected to the Internet. Many hosted services also offer sync clients for various mobile devices and desktop platforms.

You can use the hosted services to share all kinds of data, including photos and videos without worrying about the costs of uploading additional data. In a professional setup, you can also use these tools to collaborate on documents and use their in-built version control features to track changes and revert to older versions.

In this group test, we'll look at some of the most popular tools for hosting your data and sharing it with others on your terms. Most home users can easily repurpose an old unused computer as the server and some tools even put out server images for the Raspberry Pi.

Most home users can easily repurpose an old unused computer as the server, or even a Raspbperry Pi

Essential features

Some of our hosted storage solutions are easier to set up, some offer fine grained access controls, while others pay more attention to security. However, there are some features that are common to all solutions on test and we'll pay special attention to these when evaluating the available options. Data encryption together with security and access control helps you keep the data under your control. Together they are the foremost reason for going through the trouble of setting up your own storage server. We'll look for the servers that help you bring back accidentally deleted files, and we'll reward the options that offer controls to share and collaborate over the stored data. Tools that offer mobile clients will also be rated higher than those that only offer desktop clients. While scalability is difficult to evaluate without extensive use, we'll make note of solutions that cater to a large number of users just as easily as they can serve a handful.

Make way

Access your storage server from the internet.

By default your storage server will only be accessible from computers within the network it's set up on. But that's not to say that you can't access it from the internet. The trickier solution is to either get a static IP or use a dynamic DNS service and then poke holes in your router's firewall to allow traffic from the internet. The smarter way though is to use a tunnelling service such as PageKite. It uses a Python script to reverse tunnel from your computer to a **subdomain.pagekite.me** address. The service uses a pay-what-you-want model. The minimum payment of $4 (about £3.00) gets you 2GB of transfer quota for a month. Pay more to get more bandwidth for a longer duration and the ability to create additional **.pagekite** addresses. To use PageKite, fire up a terminal and install the *PageKite* software with

```
curl -s https://pagekite.net/pk/ | sudo bash
```

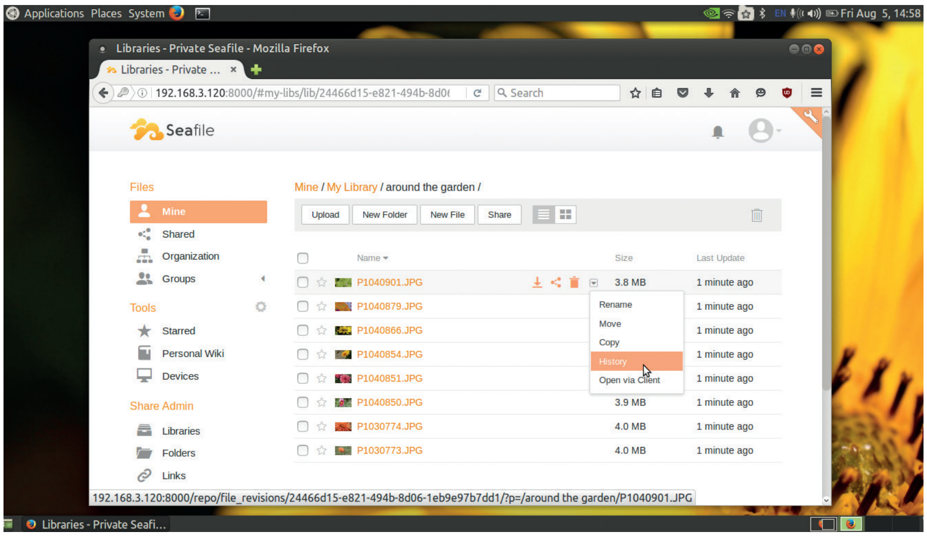
Now assuming your storage server is running on port 80, put it on the internet with

```
pagekite.py 80 mypicturesofkittens.pagekite.me
```

That's it. Your private server is now publicly accessible on **https://mypicturesofkittens.pagekite.me**. Remember to replace **mypicturesofkittens** with any name.

Seafile Community Edition

Sea-full of features.



Name	Size	Last Update
P1040901.JPG	3.8 MB	1 minute ago
P1040879.JPG		1 minute ago
P1040866.JPG		1 minute ago
P1040854.JPG		1 minute ago
P1040851.JPG		1 minute ago
P1040850.JPG	3.9 MB	1 minute ago
P1030774.JPG	4.0 MB	1 minute ago
P1030773.JPG	4.0 MB	1 minute ago

192.168.3.120:8000/repo/file_revisions/24466d15-e821-494b-8d06-1eb9e97b7dd1/?p=/around the garden/P1040901.JPG

Head to https://seacloud.cc to test drive a live Seafile installation.

Setting up Seafile doesn't take much effort. It can use various databases depending on the number of users it'll serve. Simple deployments can use the *SQLite* database, while others can deploy it with existing *MySQL/PostgreSQL* database installations and web servers such as *Nginx* or *Apache*. After creating the database you'll need to manually start the Seafile service and the Seahub administration interface. The first time you start Seahub, the script will prompt you to create an admin account for the Seafile Server.

Seafile's web interface is very verbose. You begin by creating a library, which can optionally be encrypted, and then add files to it from your computer. The service offers flexible sharing options to share libraries or even individual files with individual users or groups of users. While sharing files you can lock them with a password and even set an expiration date for the share. You can also transfer ownership to another user.

You need a client to interact with the server. (Along with Debs and RPMs, Seafile has clients for Windows and Mac OS X as well.) Every Seafile desktop client has a unique private key. When a client and a server connect, they exchange the public key and negotiate a session key. This session key is then used to encrypt the data transfer. You can also create an encrypted library, which is encrypted with AES 256. The desktop client sits in the system tray and displays notifications for sync operations. Seafile also has clients for Android and iOS. The Android client supports client-side encryption for encrypted libraries and the new two-factor authentication feature.

Designed for collaboration

Seafile users on the network can download and create libraries with the client and even share any folders on their desktop by uploading its contents into a shared library. Furthermore, once a library has been downloaded to a client, after any changes, the latest version will be uploaded to the server and then be synced with everyone's computers. The service also includes a **fsck** tool to check the integrity of files.

Seafile also has version control, and although it keeps a full history by default, you can specify for each library a period of time for which you want to keep old files. You can browse the history of a file and restore the file contents to an old version. As the admin, you can also add users and organise them into groups. Users can then share a library with specific contacts or groups and enable read-write or read-only access to different libraries. Members can easily upload, download and edit files online or even download the whole libraries from the cloud. Using the web interface you can see which files are shared with other users.

Seafile's web interface includes a simple editor that converts plain text into valid XHTML documents. It also lets users collaborate on documents and generates a new version of a file after every modification. This helps track changes and also lets you restore the file to a previous version easily. You can view several document types from within the Seafile web interface including *LibreOffice* files, PDF files, JPEGs and PNGs, and various source code formats. You can also optionally enable a wiki module.

While Seafile is very intuitive to operate, it does have detailed documentation on its website to handhold you through all its features and tasks.

Easy to roll out and works pretty much like Dropbox.

★★★★☆

SparkleShare

All that glitters ain't gold.

SparkleShare is one of the easiest and most straightforward solutions to install: all that's required to install the service is to download and run a script. Developers will like the SparkleShare service, since it uses the *Git* version control system under the hood. Besides the server you'll also need to configure clients and hook them up to the server. You can connect multiple clients to the same SparkleShare server instance.

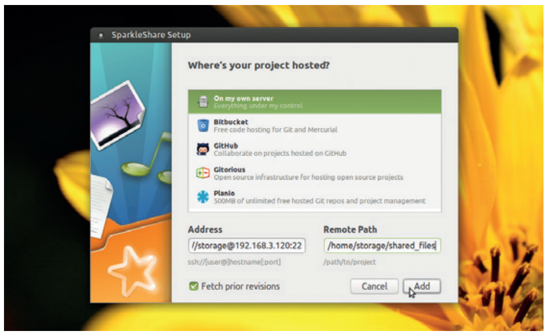
However, sharing a directory with SparkleShare takes some doing. When you create a project SparkleShare spits out the SSH address of the host and the location to the shared directory, which is created under its own **/home/storage** user directory. You need to give these to any client you wish to sync with the server.

You can install the client from your distro's repository or compile it yourself following instructions on the website. When you add clients they'll ask for the server's public SSH key, so you'll have to figure out how to pass them along to remote clients.

A simple plan

SparkleShare only offers file sharing services – nothing else, though you can use it to back up and share data with other users. The service transfers data over SSH channels and you can optionally create encrypted shares as well. But unlike other services, you don't get a web-based user interface for administering the service and it's all done from the command line.

The service has a desktop client with limited features and no mobile client. The desktop client sits in the system tray and displays notifications about the sync operations as well as a list of changes across all shared directories. Additionally, the developers themselves admit that SparkleShare isn't great for storing photos, music collections and large binary files that change often, like video editing projects. Finally, since it's based on *Git*, SparkleShare has version control built-in. Non-seasoned *Git* users will also have trouble hunting for help, which is only dispensed via IRC.


SparkleShare Setup
Where's your project hosted?
On my own server
Everything under my control
Bitbucket
Free code hosting for Git and Mercurial
GitHub
Collaborate on projects hosted on GitHub
Gitorious
Open source infrastructure for hosting open source projects
Planio
500MB of unlimited free hosted Git repos and project management
Address
//storage@192.168.3.120:22
ssh://[user@]hostname[:port]
Remote Path
/home/storage/shared_files
/path/to/project
Fetch prior revisions
Cancel
Add
The SparkleShare client can connect to the server as well as other hosting sites including GitHub and BitBucket.

Easy to set up but offers limited features and flexibility.

★★☆☆☆

Pydio

Space-age file sharing.

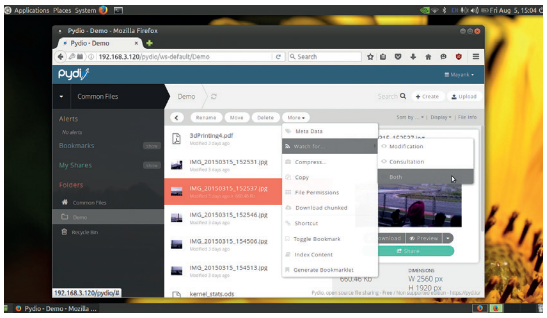
You can install Pydio after adding its repository to your distro. The server's first-run wizard lets you select the database you wish to use depending on the number of users it'll serve. Pydio focuses solely on file hosting, and the platform is geared towards collaborative environments.

It has a feature-rich and intuitive web interface. Files are divided between workspaces that can be kept private or shared with other users. The interface shows a detailed preview of the selected file. You can search for files within workspaces, change the appearance of the files and preview several types of documents including text, PDF, images, and more. You can also download files and add bookmarks to quickly access files.

To share a file you can generate a link and invite collaborators to work on the document. Pydio offers several options to secure your shares. You can password-protect a share and also set an auto-expiration for the link either by date or after a specified number of downloads. You can also set the access permissions for the shares. Similarly you can share complete folders either as independent workspaces or as a public link. When sharing folders you can also specify a layout for the share depending on the content. For example, if the shared folder contains images you can make it appear as a Film Strip.

Store of value

Pydio has desktop and mobile apps for all platforms, which help access and sync documents from the devices to the Pydio server. The desktop app keeps files synchronised across all computers and the mobile app can stream audio and video directly from the Pydio server. Besides the built-in features you can also extend Pydio as per your needs. Pydio has a smart modular design and provides additional functionality via several dozen plugins. For example, you can add LDAP support, Bitly URL shortener, an antivirus scanner, a HTML 5 video player, and more.



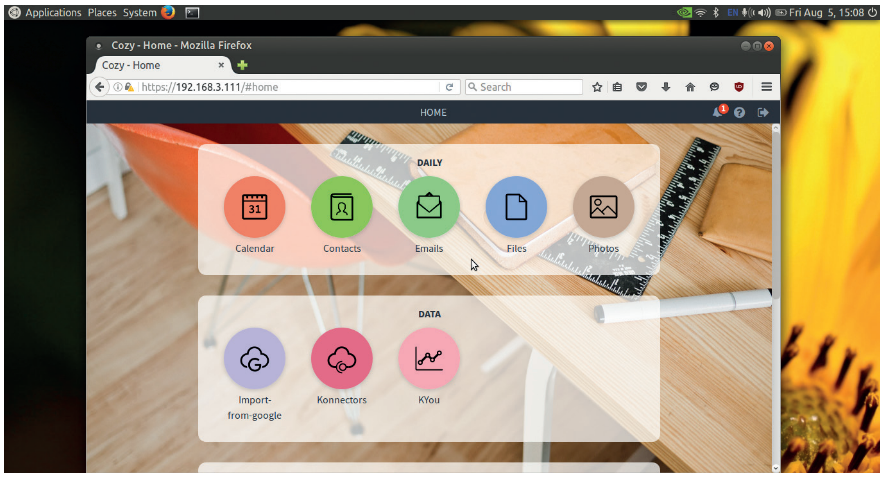
One of the best features of Pydio is that you can set alerts on files to see when they have been modified or viewed.

Ideal platform for collaborating and sharing files with a group of users.

★★★☆☆

Cozy

Get comfy with your data.



Once you've connected Cozy to your various accounts, use the *KYou* app to setup a dashboard that visualises the different kinds of data stored within Cozy.

Cozy is available through a dedicated software repository for Debian and Ubuntu and the project also produces images for the Raspberry Pi 2 as well as for *VirtualBox*. Cozy installs all its required components, including the *CouchDB* database and the *Nginx* web server, and generates its own self-signed SSL certificate (but gives you the option to get one via **letsencrypt.org** as well). A first-run wizard creates the admin user and gives you the option to synchronise your Gmail account and import calendar, contacts and photos from Google.

To store files on your Cozy server, you can upload them to the Files app. Cozy has apps for the desktop, using which you can sync files between your Linux installation and the Cozy server. Similarly the mobile app can sync data between the Cozy server and your Android device. Cozy lets you assign tags to files for easier management. You can share individual files or complete folders with others by generating links. In addition to sending the links manually, you can also share the file or folder with specific users by adding their email addresses in the Sharing dialog, where you can also define access rights for individual email addresses. If you've allowed contacts to add files to the share, you can ask Cozy to notify you when a contact adds a file to the folder.

The default Cozy installation also includes a Photos app that's designed for publishing and sharing photos. From the app you can share photos, mark them as favourites, rotate them, and view them as a slideshow. You can pick photos from the Files app and add them to an album in the Photos app.

More than just files

A default Cozy installation also includes a Calendar app, which supports CalDAV for syncing calendar data across multiple devices and platforms. It offers all essential features for managing events and appointments. Similarly, the Contacts app supports CardDAV and includes features to help manage large address books. Of note is the Email app, which keeps local copies of all email messages as backup and lets you add multiple email accounts via IMAP.

Cozy comes with its own store, which you can use to browse and install over a dozen useful third-party apps. Some useful apps are the Ghost app for rolling out a personal blog; the Music app, which streams music stored on the Cozy server; Kresus, for managing personal finances; and Frost for archiving web pages in Cozy.

Cozy also features a set of connectors for pulling data from external sources – these can collect a wide range of data such as your tweets from Twitter, contacts from LinkedIN and Google, events from Facebook, and even invoices from services like Uber and Virgin Mobile.

Wonderful platform to centrally manage all kinds of files and data.

★★★☆☆

Tahoe-LAFS: A fault-tolerant data store

RAID on steroids.

The Tahoe Least-Authority File System (LAFS) is an open source storage system that pays special attention to the security and redundancy of the data it houses. Unlike other systems that use a single server, Tahoe-LAFS uses a RAID-like mechanism to store files across multiple storage servers. When you tell your client to store a file, it will encrypt the file and then break it up into multiple pieces before spreading them out to multiple servers. Later when you retrieve the file, Tahoe-LAFS will find the necessary pieces, reassemble them before finally decrypting them.

Setting up a Tahoe-LAFS system doesn't take much effort, as it's available in the official repositories of several distributions. On an Ubuntu Server installation,

```
sudo apt-get install tahoe-lafs
```

will fetch the required components.

You'll then need to create an introducer component that communicates with the storage nodes and the client. Type

```
tahoe create-introducer ~/.introducer
```

to create the introducer and

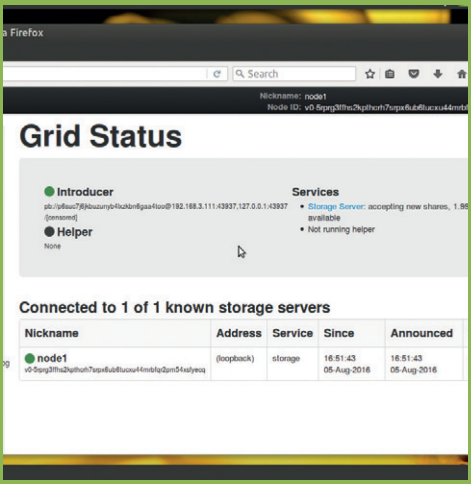
```
tahoe start ~/.introducer
```

to bring it online. The introducer has a unique address listed in the **~/.introducer/private/introducer.furl** file that you need to copy in each of the storage nodes' configuration file (**~/.tahoe/tahoe.cfg**) after creating the storage node with

```
tahoe create-node
```

You'll also need to uncomment the **shares.needed** and **shares.total** parameters, which by default will divide the files into 10 shares and any three can be used to recreate the file. Then save the file and start the server with

```
tahoe start
```



Nickname: node1
Node ID: v0-5rprg3ffhs2kpthorh7srpx6ub6lucxu44mrb

Grid Status

Introducer
pb://p6suc7j6jkbuzunyb4lxzkbn6gaa4too@192.168.3.111:43937,127.0.0.1:43937/[censored]

Helper
None

Services
- Storage Server: accepting new shares, 1.9 available
- Not running helper

Connected to 1 of 1 known storage servers

Nickname	Address	Service	Since	Announced
node1 v0-5rprg3ffhs2kpthorh7srpx6ub6lucxu44mrbfqr2pm54xsfyecq	(loopback)	storage	16:51:43 05-Aug-2016	16:51:43 05-Aug-2016

Head to port 3456 on the Tahoe-LAFS server to manage the installation. You'll need to spend some time with the documentation before you can operate it.

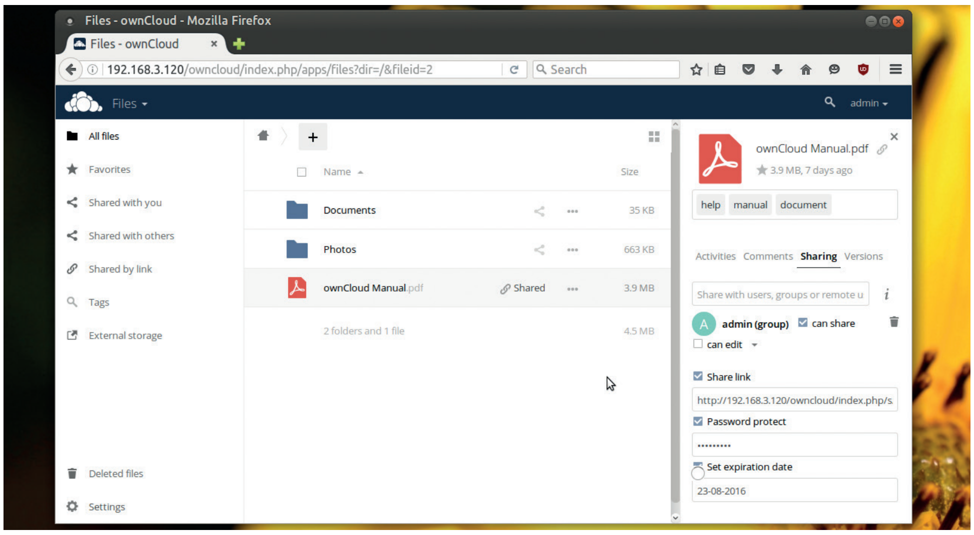
OwnCloud vs NextCloud

Luke, I am your father.

OwnCloud is one of the most widely recognised hosted storage servers. Recently, a majority of its core developers, including the project's founder, forked the code and created NextCloud. By default, both use the *SQLite* database server but can also plug into an existing *MySQL* database and will also work with other web servers including *Nginx* and *Lighttpd*.

Both projects have functional desktop clients that can be accessed from the status bar on your distro and give you a summary of the recent sync activity. You can use the clients to throttle upload and download bandwidth and pause and resume transfers. The clients also let you add local folders and specify patterns for files or directories that shouldn't be synced, and enable you to mount external cloud storage drives, such as Google Drive, Amazon S3, Dropbox, and more, and can seamlessly manage data on these services along with that in your private cloud.

Both solutions are also equally matched in terms of security, supporting server-side encryption and a simple version control mechanism. Unlike encryption, versioning is enabled by default. The versioning feature ensures you never run out of space on your storage server, as it automatically deletes old versions according to a well-defined routine. You can also share a file publicly with a URL and optionally protect it with a password and set an expiration date for the link.


Files - ownCloud - Mozilla Firefox
192.168.3.120/owncloud/index.php/apps/files?dir=/&fileid=2

Name		Size
Documents		35 KB
Photos		663 KB
ownCloud Manual.pdf	Shared	3.9 MB
2 folders and 1 file		4.5 MB

ownCloud Manual.pdf
3.9 MB, 7 days ago
help manual document
admin (group) ☑ can share ☐ can edit
☑ Share link
http://192.168.3.120/owncloud/index.php/s
☑ Password protect
☑ Set expiration date
23-08-2016

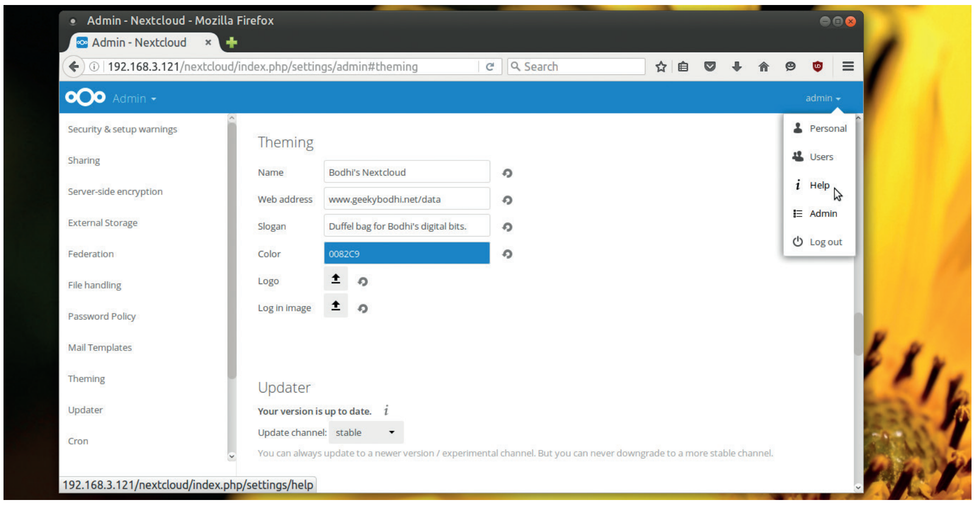
Besides tarballs and virtual machines, OwnCloud offers precompiled binaries and web installers.

What's next?

While it might appear that both OwnCloud and NextCloud are similar to each other, there are differences between the two. For starters, OwnCloud has a Community edition that is open-sourced under the AGPLv3 licence, and an Enterprise edition with additional features available under a commercial licence. In contrast, NextCloud only has one open source version and plans to generate revenue via support and consulting services.

While NextCloud already compares well with OwnCloud in terms of the features in the latter's community edition, it's hard at work offering the features available in the ownCloud Enterprise edition as well. It's already made considerable strides in this aspect and is constantly being updated to address these gaps with features targeted at enterprise users, including the single sign-on capability, theming functionality for custom branding, custom password policy, secure WebRTC conferencing, Collabora Online Office integration and more.

Also, OwnCloud's mobile apps for Android and iOS aren't available as free downloads, nor does it offer an evaluation version. Here again, NextCloud outscores its progenitor by offering apps for both Android and iOS, using which you can work with your files stored on the storage server from the mobile device. Furthermore, you can find a host of add-ons that you can use to flesh out your installation in the NextCloud app store, which surprisingly even at this early stage lists a few more apps than the OwnCloud store.


Admin - Nextcloud - Mozilla Firefox
192.168.3.121/nextcloud/index.php/settings/admin#theming

Theming
Name: Bodhi's Nextcloud
Web address: www.geekybodhi.net/data
Slogan: Duffel bag for Bodhi's digital bits.
Color: 0082C9

Updater
Your version is up to date.
Update channel: stable
You can always update to a newer version / experimental channel. But you can never downgrade to a more stable channel.

192.168.3.121/nextcloud/index.php/settings/help

You can access your remote storage using the file manager on any OS via the WebDAV protocol.

OwnCloud
Offers more features than many of the others but loses out to its fork.
★★★☆☆

NextCloud
The fork of the mature, feature-rich storage server that scales well.
★★★★☆

OUR VERDICT

Hosted storage servers

Selecting a hosted storage service depends on a number of factors, such as the type of data you will be syncing, and your level of expertise working with network software on Linux. In fact, it isn't uncommon for people to use multiple services. For example, if you want a central storage server for your personal use, Cozy presents a very viable option. It's easy to deploy and has useful features and apps for the home user. But while Cozy encrypts passwords and connections, it doesn't encrypt the data it houses. So if you're really concerned about the privacy of your data you can use SparkleShare to collaborate with colleagues and safely pass documents related to confidential projects. The service is very secure since it uses SSH, but isn't intuitive enough for the average desktop user since connecting the clients to the server is a more involved process and you'll have to find a way to securely transfer the public SSH keys from the client to the host.

It's a close call between Pydio and Seafile. Seafile has client-side encryption and versioning, a feature-rich client for the desktop and a functional client for the mobile. The service gets brownie points for making a version especially for the Raspberry Pi. However, Pydio has a much nicer and more intuitive user interface. It also scores for certain administrative features like user roles and its ability to notify you when shared files are viewed or modified.

Taking over

Unlike the runners up, there's no confusion over the top spot, which goes to NextCloud. It scales wonderfully and can be used to share files with family and friends just as easily as you can with colleagues across continents. Besides the core functions you also get a host of additional useful apps. We also like its ability to connect to external storage services.

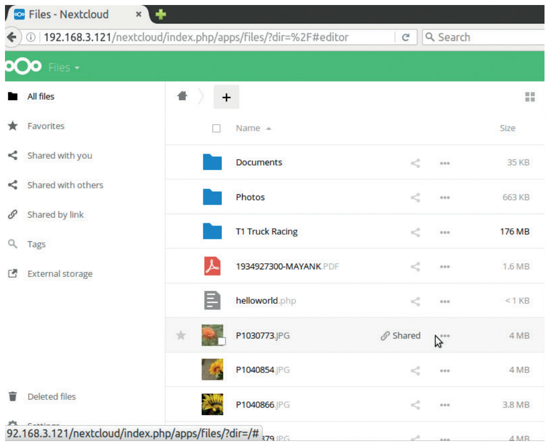
The one victim of NextCloud's success is OwnCloud. In terms of the software, OwnCloud's community edition is still better equipped than the other options on test here, barring the NextCloud fork. But until it publishes an updated version that's more than a bugfix release, we can't bring ourselves to recommend a project that has lost most of its core developers and has had to curb its commercial operation.

NextCloud can be used to share files with friends just as easily as you can with colleagues across continents.

Other options

Besides the storage servers we've tested in this feature, there are lots of options that help you sync data across multiple computers and devices. There's *Git-annex*, which syncs files using *Git*; *Resilio Sync*, which is a proprietary peer-to-peer file synchronisation tool that relies on a modified version of the BitTorrent protocol; and its open source equivalent called *Syncthing*. There's also *StackSync*, which uses a hybrid model to store the metadata inside your network while the actual encrypted data is stored on a public cloud like OpenStack Swift or Amazon S3.

Furthermore, Network Attached Storage solutions are another class of tools that are designed for storing data and making it available across the network. Options such as *OpenMediaVault*, *NAS4Free* and *FreeNAS* help set up a centralised data store that's adept at managing all kinds of data, and can be controlled via a remote browser-based graphical interface. These NAS solutions provide access to the data via multiple protocols including Samba, NFS, FTP and Rsync. You can also install a specialised storage solution like OwnCloud on top on an OpenMediaVault NAS installation.



Name		Size
Documents		35 KB
Photos		663 KB
T1 Truck Racing		176 MB
1934927300-MAYANK.PDF		1.6 MB
helloworld.php		< 1 KB
P1030773.JPG	Shared	4 MB
P1040854.JPG		4 MB
P1040866.JPG		3.8 MB

Because of their shared core, you can easily update your OwnCloud installations to NextCloud.

1 NextCloud

Killer feature Free app store.
URL www.nextcloud.com
Offers everything you can ask for in a hosted storage server.

2 Pydio

Killer feature Intuitive user interface.
URL https://pydio.com
A good option if you don't need advanced enterprise-centric features.

3 Seafile

Killer feature Client-side encryption
URL www.seafile.com
The GPLed community edition offers useful features for a non-Enterprise user.

4 Cozy

Killer feature Email backup
URL www.cozy.io
Easy to deploy and manage if you can live with its lack of support for multiple users.

5 SparkleShare

Killer feature Built around Git.
URL www.sparkleshare.org
If you use Git professionally, you'll enjoy using it to sync your data.

6 OwnCloud

Killer feature Scalability.
URL www.owncloud.org
It's down here only because we're not sure about its future.

Subscribe

shop.linuxvoice.com

PROUDLY INDEPENDENT SINCE 2013
LINUXVOICE
Raspberry Pi
Control a Pi with an RFID chip and a bit of Python
November 2016 FREE SOFTWARE | FREE SPEECH www.linuxvoice.com
PROTECT YOUR PRIVACY
Stop the man fro – thanks to Fr
HOW TO MAKE SOFTWARE THAT NORMAL PEOPLE
SAMSUNG
Banish Twitter!
fedora
BURSTING WITH AWESOME TUTORIALS!
AFTERSHOT > OPENVR & MORE!
ULTIMATE SPEEDUPS
RETROPIE > XATTRS > DIGIKAM 5 & MORE!
OPENVPN > GNEWSENSE > MICROSOFT & MORE

Introducing Linux Voice, the magazine that:

- Gives 50% of its profits back to Free Software
- Licenses its content CC-BY-SA within 9 months

12-month subs prices

UK – £55
Europe – £85
US/Canada – £95
ROW – £99

7-month subs prices

UK – £38
Europe – £53
US/Canada – £57
ROW – £60

DIGITAL SUBSCRIPTION ONLY £38

Get 114 pages of tutorials, features, interviews and reviews every month

Access our rapidly growing back-issues archive – all DRM-free and ready to download

Save money on the shop price and get each issue delivered to your door

Payment is in Pounds Sterling. 12-month subscribers will receive 12 issues of Linux Voice a year. 7-month subscribers will receive 7 issue of Linux Voice. If you are dissatisfied in any way you can write to us to cancel your subscription at subscriptions@linuxvoice.com and we will refund you for all unmailed issues.

NEXT MONTH IN LINUXVOICE

ON SALE 29 OCTOBER

NEW KIDS ON THE BLOCK

The coolest, freshest, smartest Linux distros of 2016 – try them now and find your new favourite Linux flavour.

EVEN MORE AWESOME!

NoSQL

Let's be honest: databases are hard. Which is why you should learn all about NoSQL – you'll be streets ahead of everyone else with only minimal effort!

KDE beast mode

You may know and fear KDE as the grey mess of version 4.x. Fear no more, as Linux Voice presents the ultimate guide to the Best Desktop Environment Ever!

Ansible

Level up your Linux sysadmin skills with Ansible – multi-app, multi-level, multi-system software provisioning that's going to make your CV look awesome.

LINUX VOICE IS BROUGHT TO YOU BY

Editor Ben Everard
ben@linuxvoice.com
Deputy editor Andrew Gregory
andrew@linuxvoice.com
Editor at large Mike Saunders
mike@linuxvoice.com
Editor in hiding Graham Morrison
graham@linuxvoice.com
Creative director Stacey Black
stacey@linuxvoice.com

Editorial consultant Nick Veitch
nick@linuxvoice.com

All code printed in this magazine is licensed under the GNU GPLv3

Printed in the UK by
Acorn Web Offset Ltd

Disclaimer We accept no liability for any loss of data or damage to your hardware through the use of advice in this magazine. Experiment with Linux at your own risk!
Distributed by Marketforce (UK) Ltd, 2nd Floor, 5 Churchill Place, Canary Wharf, London, E14 5HU
Tel: +44 (0) 20 3148 3300

Circulation Marketing by Intermedia Brand Marketing Ltd, registered office North Quay House, Sutton Harbour, Plymouth PL4 0RA
Tel: 01737 852166

Subscribe: **shop.linuxvoice.com**
subscriptions@linuxvoice.com

FOSSpicks

Sparkling gems and new releases from the world of Free and Open Source Software

Our benevolent editorial overlord **Graham Morrison** tears himself away from updating Arch Linux to search for the best new free software.

Video editor

OpenShot 2.1

This is a major update to one of our favourite applications in a desperately needed genre – the humble video editor. In a first for us, we're running the release from an 'AppImage' downloaded from the *OpenShot* site, rather than via the time-consuming source code we typically build our FOSSPicks from. This is the distributable package format that used to be known as Klik, and you simply download the binary (*OpenShot* is a 192MB download), add **+x** to its permissions and run. For comparison, the Arch binary is 84MB and a 160MB installation, which we think compares well considering the supreme convenience of being to run an application from a download.

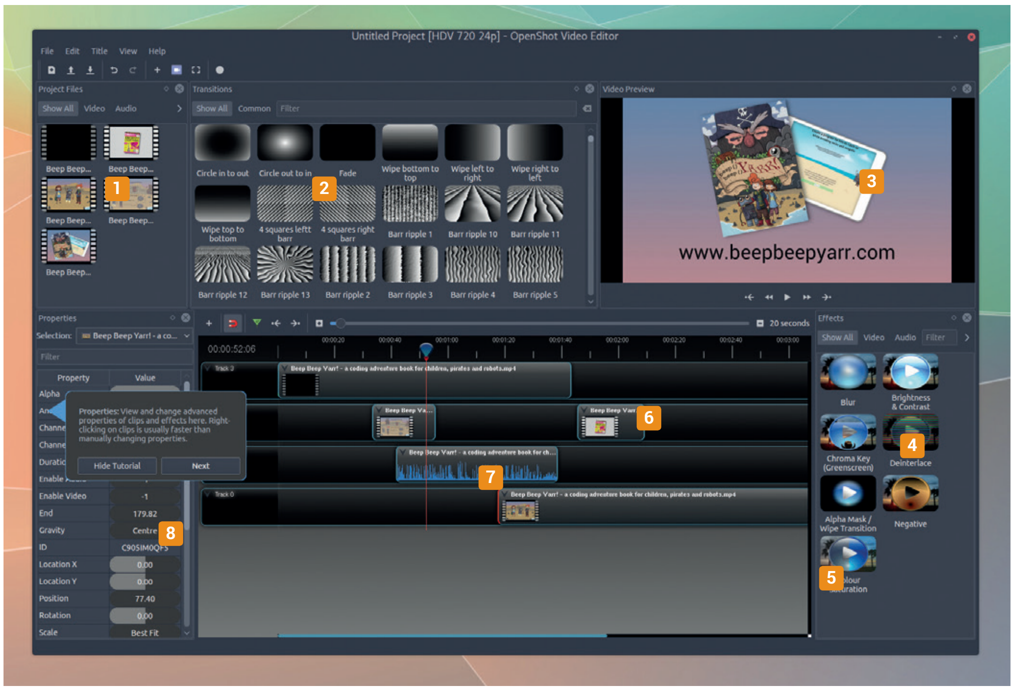
It might seem a little prosaic, but our favourite new feature in this new version is the stability. Video editors are particularly prone to running out of memory, or CPU power, or GPU resources, usually resulting in a silent crash and lost work. Previous versions of *OpenShot* have been susceptible to this, but in our experience, this release has been rock solid. You still need a good machine if you're editing multiple channels of 720p video, and you still need to wait some time for many of the processes to finish, but the application shouldn't crash.

Box of tricks

There are also plenty of new features. There's a new Properties pane, for instance, which works like the properties pane in *OpenShot*'s competitors, enabling you to enter numeric-specific values or slide between attributes when defining anything from the scale of a clip and its location, to its transparency and gravity. It's perfect for creating transitions that fit exactly, and is essential for any large project.

There's also a new audio waveform view, making it easy to line up different clips of video with the same audio, as well as being able to physically see where edits can be made. When enabled, an audio waveform appears beneath the clip, or on its own, and we'd love to see this feature extracted from *OpenShot* into a new audio editor.

We also found the new inline tutorials very useful – especially as one of our worst habits is diving into even the most complex applications and expecting to be able to do something useful. *OpenShot*'s very brief tutorials appear when you first access a feature, such as the main window or the transitions page. In short, if you're into video editing in any way, this latest update to *OpenShot* is the update you've been waiting for.



Properties: View and change advanced properties of clips and effects here. Right-clicking on clips is usually faster than manually changing properties.

1 Clip view Easily import or split longer clips for use within your project. **2 Transitions** Select between simple fades and more complex effects. **3 Video preview** Press Space to play your edits from the cursor. **4 Effects** Blur or adjust image settings over time. **5 Properties** Define exactly how and where your clips are located. **6 Timeline** Place clips and transitions that cut when overlapped. **7 Audio waveform** See the audio amplitude either beneath a clip or on its own. **8 Tutorials** Mini info pages appear when you enter a new section of the application.

Project website
www.openshotvideo.com

Text editor

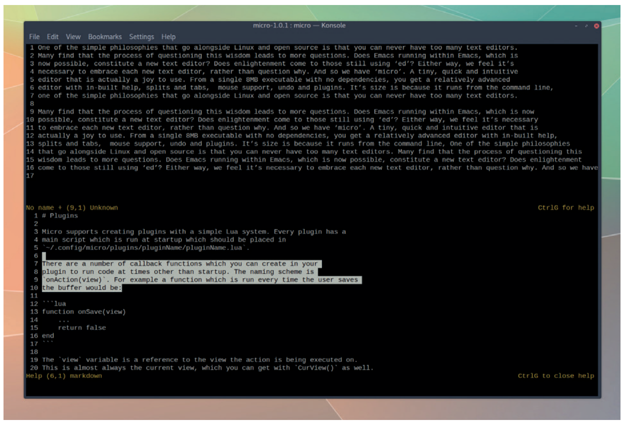
Micro 1.0.1

One of the simple philosophies that go alongside Linux and open source is that you can never have too many text editors. Many find that the process of questioning this wisdom leads to more questions. Does *Emacs* running within *Emacs*, which is now possible, constitute a new text editor? Does enlightenment come to those still use *ed*? Either way, we feel it's necessary to embrace each new text editor, rather than question why. And so we have Micro: a tiny, quick and intuitive editor that runs from the command line and is a joy to use. From a single 8MB executable with no dependencies, you get a relatively advanced editor with in-built help, splits and tabs, mouse support, undo and plugins.

The in-built help is a great way to get started, and opens by splitting the view vertically. This means you can experiment with options and read about how they work at the same time.

Like similar editors, *Micro* relies on keyboard shortcuts and commands typed via a special mode, rather than menus or on-screen prompts as you find with *Nano*. This makes it slightly trickier for a beginner to pick up – but it's easier to use than *Vim*. It's also very capable, with lots of settings and configuration options you can save, and even syntax highlighting support for over 75 languages. And despite being a console-based editor, great mouse support makes it feel like a desktop app, with pane selection and even drag-and-drop implemented by a click of your mouse. Plugins are written in the Lua scripting language, and credit should go to the help text for including the code and the hooks for writing your own. Download it today (it takes about a second).

Micro relies on keyboard shortcuts and commands typed via a special mode


micro-1.0.1 : micro — Konsole
File Edit View Bookmarks Settings Help

~~~
One of the simple philosophies that go alongside Linux and open source is that you can never have too many text editors.
Many find that the process of questioning this wisdom leads to more questions. Does Emacs running within Emacs, which is
now possible, constitute a new text editor? Does enlightenment come to those still using 'ed'? Either way, we feel it's
necessary to embrace each new text editor, rather than question why. And so we have 'micro'. A tiny, quick and intuitive
editor that is actually a joy to use. From a single 8MB executable with no dependencies, you get a relatively advanced
editor with in-built help, splits and tabs,  mouse support, undo and plugins. It's size is because it runs from the command line,
one of the simple philosophies that go alongside Linux and open source is that you can never have too many text editors.

Many find that the process of questioning this wisdom leads to more questions. Does Emacs running within Emacs, which is now
possible, constitute a new text editor? Does enlightenment come to those still using 'ed'? Either way, we feel it's necessary
to embrace each new text editor, rather than question why. And so we have 'micro'. A tiny, quick and intuitive editor that is
actually a joy to use. From a single 8MB executable with no dependencies, you get a relatively advanced editor with in-built help,
splits and tabs,  mouse support, undo and plugins. It's size is because it runs from the command line, One of the simple philosophies
that go alongside Linux and open source is that you can never have too many text editors. Many find that the process of questioning this
wisdom leads to more questions. Does Emacs running within Emacs, which is now possible, constitute a new text editor? Does enlightenment
come to those still using 'ed'? Either way, we feel it's necessary to embrace each new text editor, rather than question why. And so we have


No name + (9,1) Unknown                                                                              CtrlG for help
# Plugins

Micro supports creating plugins with a simple Lua system. Every plugin has a
main script which is run at startup which should be placed in
`~/.config/micro/plugins/pluginName/pluginName.lua`.

There are a number of callback functions which you can create in your
plugin to run code at times other than startup. The naming scheme is
`onAction(view)`. For example a function which is run every time the user saves
the buffer would be:

```lua
function onSave(view)
    ...
    return false
end
```

The `view` variable is a reference to the view the action is being executed on.
This is almost always the current view, which you can get with `CurView()` as well.
Help (6,1) markdown                                                                               CtrlG to close help
~~~

Due to its small binary size, being statically linked and available for several OSs, *Micro* is an ideal editor for a USB stick.

Project website
https://github.com/zyedidia/micro

Reddit reader

rtv

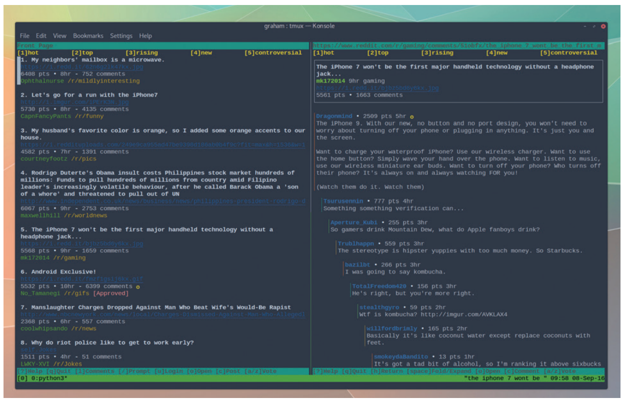
We're reluctant to mention this discovery. This is because it's a command line tool for accessing Reddit. And if you've yet to develop a Reddit habit, we don't want to be responsible for pushing you one step closer. Reddit is wonderful and evil in almost equal measure, with both the good and the bad sides growing in power daily, perfectly balancing themselves while at the same time pooling vast amounts of internet energy. One day it's going to explode.

On the good side, Reddit's breadth is unequalled. This is because the commentary and updates are split into what are called sub-reddits – groups following a certain piece of hardware, or musician, for example. There's likely sub-reddit for everything you can think of and more sub-reddits for things you can't. Sub-reddits can often feel like the early days of the pre-web internet, like Usenet, with a group for every fetish. On the bad side, Reddit is as close to the end of the internet that you can get without delving into its genuinely disturbing underworld. Actually, part of Reddit is a genuinely disturbing underworld, and that's before you start reading the comments.

Don't read the comments

Rtv is a small Python 3 application that gives you access to this world from the command line. It's brilliantly designed, simple to use, and makes Reddit easier to read than via a web browser, complete with user account support, threaded comments, and voting. It's perfect for running from the terminal of your Raspberry Pi located in Nepal. Which we'd suggest is the only location suitable for reading **/r/monkslookingatbeer**.

Rtv is a Python app that gives you access to Reddit from the command line



```
Front Page
[1]hot          [2]top          [3]rising          [4]new          [5]controversial
1. My neighbors' mailbox is a microwave.
https://i.redd.it/62n6g2lk47kx.jpg
6408 pts • 8hr - 752 comments
Ophthalnurse /r/mildlyinteresting

2. Let's go for a run with the iPhone7
http://i.imgur.com/1PErK3N.jpg
5730 pts • 8hr - 4135 comments
CapnFancyPants /r/funny

3. My husband's favorite color is orange, so I added some orange accents to our
house.
https://i.reddituploads.com/249e9ca955ad47be9398d186ab0b4f9c?fit=max&h=1536&w=1
4582 pts • 7hr - 1391 comments
courtneyfootz /r/pics

4. Rodrigo Duterte's Obama insult costs Philippines stock market hundreds of
millions: Funds to pull hundreds of millions from country amid Filipino
leader's increasingly volatile behaviour, after he called Barack Obama a 'son
of a whore' and threatened to pull out of UN
http://www.independent.co.uk/news/business/news/philippines-president-rodrigo-d
6067 pts • 9hr - 2753 comments
maxwellhill /r/worldnews

5. The iPhone 7 won't be the first major handheld technology without a
headphone jack...
https://i.redd.it/bjbz5bd6y6kx.jpg
5568 pts • 9hr - 1659 comments
mk172014 /r/gaming

6. Android Exclusive!
https://i.redd.it/fmzf1gsij6kx.gif
5532 pts • 10hr - 6399 comments o
No_Tamanegi /r/gifs [Approved]

7. Manslaughter Charges Dropped Against Man Who Beat Wife's Would-Be Rapist
http://www.nbcnewyork.com/news/local/Charges-Dismissed-Against-Man-Who-Allegedl
2368 pts • 6hr - 557 comments
coolwhipsando /r/news

8. Why do riot police like to get to work early?
self.Jokes
1511 pts • 4hr - 51 comments
LWKY-XVI /r/Jokes
[?]Help [q]Quit [l]Comments [/]Prompt [u]Login [o]Open [c]Post [a/z]Vote
[0] 0:python3*
```

```
https://www.reddit.com/r/gaming/comments/51obfx/the_iphone_7_wont_be_the_first_m
[1]hot          [2]top          [3]rising          [4]new          [5]controversial
The iPhone 7 won't be the first major handheld technology without a headphone
jack...
mk172014 9hr gaming
https://i.redd.it/bjbz5bd6y6kx.jpg
5561 pts • 1663 comments

Dragonmind • 2509 pts 5hr o
The iPhone 9. With our new, no button and no port design, you won't need to
worry about turning off your phone or plugging in anything. It's just you and
the screen.

Want to charge your waterproof iPhone? Use our wireless charger. Want to use
the home button? Simply wave your hand over the phone. Want to listen to music,
use our wireless miniature ear buds. Want to turn off your phone? Who turns off
their phone? It's always on and always watching FOR you!

(Watch them do it. Watch them)

 Tsurusennin • 777 pts 4hr
 Something something verification can...

  Aperture_Kubi • 255 pts 3hr
  So gamers drink Mountain Dew, what do Apple fanboys drink?

   Trublhappn • 559 pts 3hr
   The stereotype is hipster yuppies with too much money. So Starbucks.

    bazilbt • 266 pts 3hr
    I was going to say kombucha.

     TotalFreedom420 • 156 pts 3hr
     He's right, but you're more right.

      stealthgyro • 59 pts 2hr
      Wtf is kombucha? http://imgur.com/AVKLAX4

       willfordbrimly • 165 pts 2hr
       Basically it's like coconut water except replace coconuts with
       feet.

        smokeydaBandito • 13 pts 1hr
        It's got a tad bit of alcohol, so I'm ranking it above sixbucks
[?]Help [q]Quit [h]Return [space]Fold/Expand [o]Open [c]Comment [a/z]Vote
                                                  "the iphone 7 wont be " 09:58 08-Sep-16
```

If you need Reddit from the command line, you need *rtv*.

Project website
https://github.com/michael-lazar/rtv

File hosting

Seafile 6

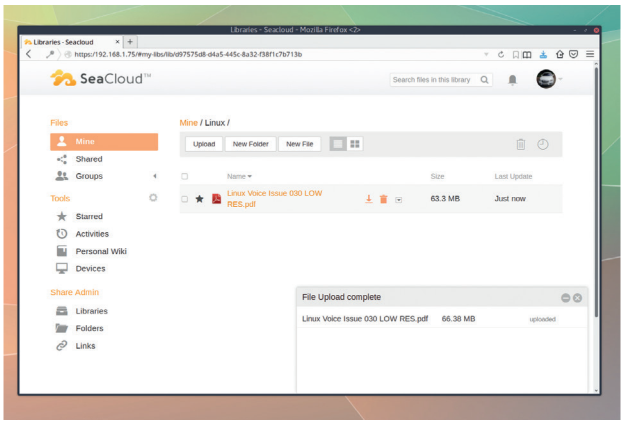
Like many, we've been huge fans of *OwnCloud* and now *NextCloud* for years. It's been central to how we run the magazine since we started. It's great for collaboration, calendars, plugins, contacts and sharing files, but it's overkill if you only need the file hosting/sharing files part.

Despite this, many people still use *NextCloud* just for file hosting, as it's still one of the best replacements for the likes of Google Drive and Dropbox (which just recently admitted to leaking 64m of its users' passwords). The convenience of running a client on your computer or mobile device that automatically uploads and synchronises your files with a remote server is worth the extra overhead of *NextCloud*'s complexity and unused features (see page 54 for more on NextCloud *et al*).

Seafile has always been an alternative that simply provides the file synchronisation, and the combination of a major update and renewed interest from Dropbox deserters make this a good time to take a look. There are some vagaries around the proprietary 'professional' edition of the server, which is obviously a core part of Seafile Ltd's business plan, but the community edition that we've just installed is GPLv2, so there's no reason to hold back.

Installation of the server is via a very simple script that prompts you to install missing packages (such as *SQLite 3*). This script runs as an ordinary user, and you can run the server as an ordinary user, so you don't need to worry about system-wide permissions and access unless you want to. But there are plenty of dependencies still, and it can be difficult to get running – especially if you're using the client.

Installation of Seafile is via a script that prompts you to install missing packages



If you're not too happy that Dropbox leaked 68 million passwords, try *Seafile*, which has just been updated to version 6.

Project website
https://www.seafile.com

Terminal task manager

Taskwarrior

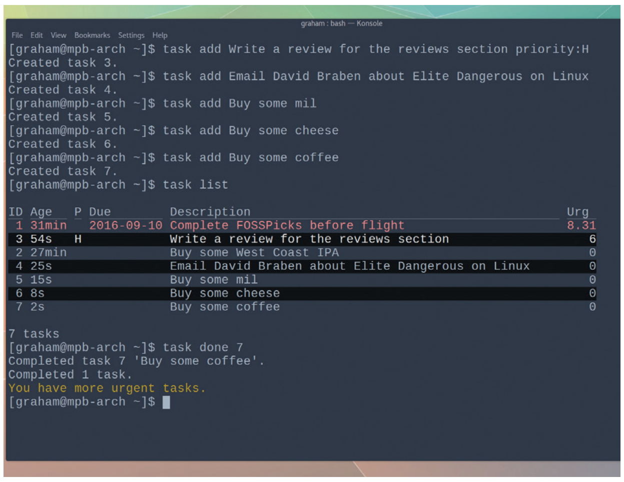
It's not going to be long before we imagine we'll be able to fill this entire section with new discoveries from the command line. As a way of interacting with your computer, it seems the command line is difficult to beat, and getting more popular every month.

Taskwarrior is a to-do list manager, and it's another one of those very neat commands that performs the same job as many GUI applications, only without many of the distractions, transitions, animations, launch screens and notifications, and is another good reason to keep using *Bash*.

In particular, 'task' (as you type it on the terminal), plays to the command line's strengths of modularity and of focusing on small jobs. To create a task, for instance, you type **task add** followed by a description and the time, day or date the task is due, described with the **due:saturday** argument, for example. You don't even need the **due** argument, if you're simply popping to the shops from some milk, but doing so will add an urgency quotient to your list of items, hopefully prodding you to do those urgent things first.

Sort your life out

Typing **task list** will show you the things that need to be done, alongside their urgency and due date, and you can simply tick them off your list by entering **task 1 done**, for example. It's simple and brilliantly effective, especially if you're working from the command line anyway. It's a great way of working through a series of commits or pull requests, for instance. And if you want to sync your tasks across devices, you can even install *Taskserver*, so that each client can independently update a single list and keep you up to date.



```
[graham@mpb-arch ~]$ task add Write a review for the reviews section priority:H
Created task 3.
[graham@mpb-arch ~]$ task add Email David Braben about Elite Dangerous on Linux
Created task 4.
[graham@mpb-arch ~]$ task add Buy some mil
Created task 5.
[graham@mpb-arch ~]$ task add Buy some cheese
Created task 6.
[graham@mpb-arch ~]$ task add Buy some coffee
Created task 7.
[graham@mpb-arch ~]$ task list

ID Age   P Due        Description                                        Urg
 1 31min   2016-09-10 Complete FOSSPicks before flight                   8.31
 3 54s   H            Write a review for the reviews section                6
 2 27min              Buy some West Coast IPA                               0
 4 25s                Email David Braben about Elite Dangerous on Linux     0
 5 15s                Buy some mil                                          0
 6 8s                 Buy some cheese                                       0
 7 2s                 Buy some coffee                                       0

7 tasks
[graham@mpb-arch ~]$ task done 7
Completed task 7 'Buy some coffee'.
Completed 1 task.
You have more urgent tasks.
[graham@mpb-arch ~]$ █
```

Manage your increasingly tangled to-do list from the command line with the wonderful *Taskwarrior*.

Project website
https://taskwarrior.org

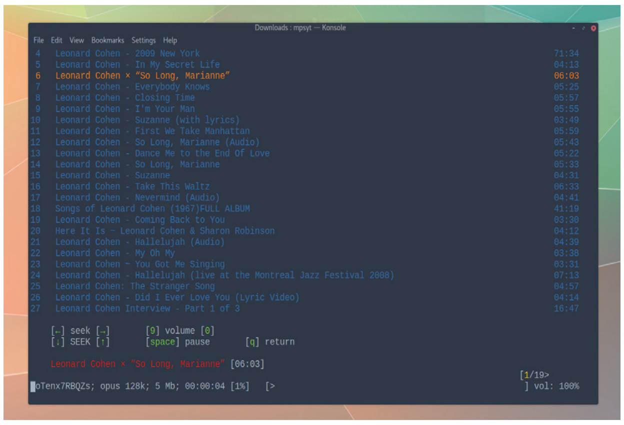
YouTube music player

mps-youtube

Yep, we still can't escape the command line (nor do we want to). Here's a great little utility that simplifies what many of us use YouTube through a browser for – playing music, only without the distraction of the music video that goes along with it.

With **mps-youtube** installed, you run **mpsyt** from the command line and press H to get some help. The most important command is **search**, and you'd use this exactly as you would search YouTube in a browser. Type **search leonard cohen** and you'd get the same results, only with the opportunity to listen in the terminal. You can page through the results and select which video you'd like to hear the audio for. After a brief delay as the file is buffered and converted, you'll hear music just like with any other music player. You can then skip ahead, pause or select another track just like any other playlist. You totally forget that the source for all this music is YouTube, especially as you don't have to endure the increasingly long adverts that YouTube has started to insert into so many of its videos.

You can even create your own local playlists, play YouTube mix playlists, shuffle the playback order for the playlist and encode audio into MP3 and other formats. Music and video can be downloaded, and if you're desperate, you can watch the video in an external player too. If you're even more desperate, you can view the comments from the command line. Turning YouTube into a distraction-free music source is a great feature, and it's reminded us just how much rare and unreleased material exists online, quite apart from the legitimate audio you can subscribe to or pay for from the artists themselves, and **mps-youtube** is one of the best tools we've found for getting to it.


Downloads : mpsyt — Konsole
File Edit View Bookmarks Settings Help

```
4   Leonard Cohen - 2009 New York                                         71:34
5   Leonard Cohen - In My Secret Life                                     04:13
6   Leonard Cohen × "So Long, Marianne"                                   06:03
7   Leonard Cohen - Everybody Knows                                       05:25
8   Leonard Cohen - Closing Time                                          05:57
9   Leonard Cohen - I'm Your Man                                          05:55
10  Leonard Cohen - Suzanne (with lyrics)                                 03:49
11  Leonard Cohen - First We Take Manhattan                               05:59
12  Leonard Cohen - So Long, Marianne (Audio)                             05:43
13  Leonard Cohen - Dance Me to the End Of Love                           05:22
14  Leonard Cohen - So Long, Marianne                                     05:33
15  Leonard Cohen - Suzanne                                               04:31
16  Leonard Cohen - Take This Waltz                                       06:33
17  Leonard Cohen - Nevermind (Audio)                                     04:41
18  Songs of Leonard Cohen (1967)FULL ALBUM                               41:19
19  Leonard Cohen - Coming Back to You                                    03:30
20  Here It Is ~ Leonard Cohen & Sharon Robinson                          04:12
21  Leonard Cohen - Hallelujah (Audio)                                    04:39
22  Leonard Cohen - My Oh My                                              03:38
23  Leonard Cohen ~ You Got Me Singing                                    03:31
24  Leonard Cohen - Hallelujah (live at the Montreal Jazz Festival 2008)  07:13
25  Leonard Cohen: The Stranger Song                                      04:57
26  Leonard Cohen - Did I Ever Love You (Lyric Video)                     04:14
27  Leonard Cohen Interview - Part 1 of 3                                 16:47

    [←] seek [→]       [9] volume [0]
    [↓] SEEK [↑]       [space] pause       [q] return

    Leonard Cohen × "So Long, Marianne" [06:03]
                                                                      [1/19>
oTenx7RBQZs; opus 128k; 5 Mb; 00:00:04 [1%]   [>                     ] vol: 100%
```

Forget the distractions of music videos, and listen to the music instead.

Turning YouTube into a distraction-free music source is a great feature

Project website
github.com/mps-youtube/mps-youtube

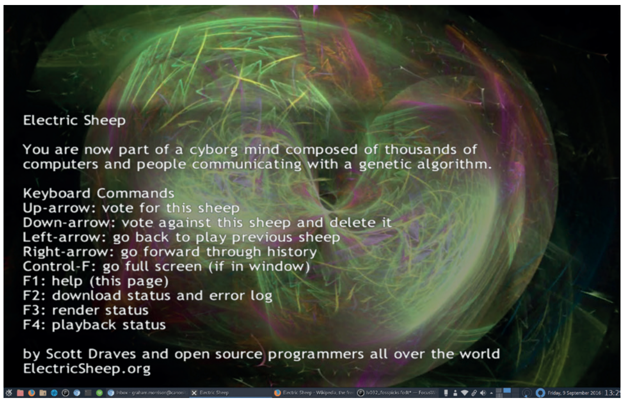
Screensaver

Electric Sheep

This is the first time that we can remember including a screensaver in FOSSPicks. But this isn't your usual piece of passive eye candy, and there are two reasons why we wanted to include it. The first is obviously its name. It would be impossible to ignore a piece of open source software that was obviously inspired by the Philip K. Dick novel that inspired the film *Blade Runner*. The second reason is obviously derived from the first, and it's the way the screensaver is actually a portal to a distributed set of other computers also running the same screensaver. These computers are 'dreaming'.

The screensaver generates complex fractal images that morph and animate. *Electric Sheep* calls these elements 'sheep', and the networked nature of the screensaver means that these sheep are shared across the network as you run the screensaver – a form of farm rendering that turns into collective dreaming for the computers you leave unattended. The animations are therefore amplified and increase in complexity for as long as you leave your computer unattended, and the results are incredibly compelling if you're into Grateful Dead-era psychedelics.

You can also adjust and tinker with the sheep and the algorithms yourself, creating your own fractal files, and user can vote on the output they prefer from within the screensaver. We just hope there's something presumably stopping anyone from farming bitcoins the same way. Either way, the results are fantastic, and the distributed networking nature of this screen saver really makes it worth a look.



Electric Sheep

You are now part of a cyborg mind composed of thousands of computers and people communicating with a genetic algorithm.

Keyboard Commands
Up-arrow: vote for this sheep
Down-arrow: vote against this sheep and delete it
Left-arrow: go back to play previous sheep
Right-arrow: go forward through history
Control-F: go full screen (if in window)
F1: help (this page)
F2: download status and error log
F3: render status
F4: playback status

by Scott Draves and open source programmers all over the world
ElectricSheep.org

"Chew, if only you could see what I've seen with your eyes."

The screensaver generates complex fractals that morph and animate

Project website
www.electricsheep.org

Image conversion

Converseen 0.9.5.1

Our first memory of image conversion, when it meant more than turning the colour of one grid of pixels into another colour, was the proprietary *ImageMagick* application running on the Amiga. This was followed quickly by what might be our first exposure to a desktop open source image tool – *Xv*, also running on an Amiga back in the mid-90s.

Decades have followed, of course, but graphics conversion is still essential, and essentially it hasn't changed that much. The tool that makes conversion quickest, easiest and most efficient wins. Surprisingly, few still manage to get close to these guiding principles.

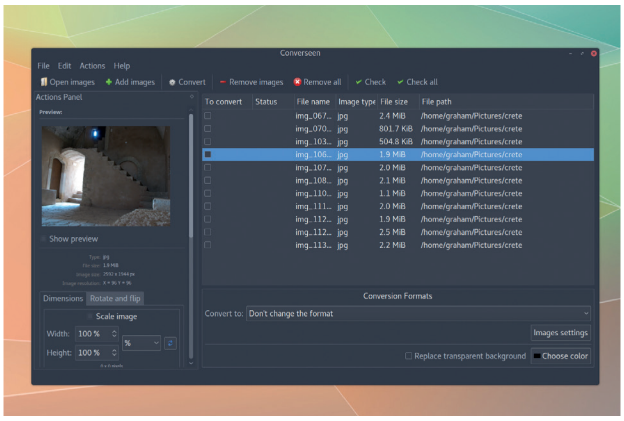
Converseen is one that gets very close, and reminds us of the venerable *Xv*. It's best feature is that it's easy to understand. You drop photos you want to convert into the right file list, change a few things in the Actions Panel and click on Convert. The application will go off and make it happen. It also offers just the right about of control, without overburdening the user with too much choice.

Keep Images Simple, Stupid

You can change the resolution of an image, for example, or the scale, you can rename the output or overwrite the original, and you can rotate/flip and replace the transparent background colour. *Converseen* doesn't do any more than this, but this is what we need a converter for 90% of the time. Most other processes, such as colour or exposure adjustments, typically need human interaction.

Converseen offers the right amount of control, not giving the user too much choice

If you want something to quickly convert your holiday photos for use on a wiki or blog, this is exactly the sort of application you need for doing so.



To convert	Status	File name	Image type	File size	File path
☐		img_067...	jpg	2.4 MiB	/home/graham/Pictures/crete
☐		img_070...	jpg	801.7 KiB	/home/graham/Pictures/crete
☐		img_103...	jpg	504.8 KiB	/home/graham/Pictures/crete
☐		img_106...	jpg	1.9 MiB	/home/graham/Pictures/crete
☐		img_107...	jpg	2.0 MiB	/home/graham/Pictures/crete
☐		img_108...	jpg	2.1 MiB	/home/graham/Pictures/crete
☐		img_110...	jpg	1.1 MiB	/home/graham/Pictures/crete
☐		img_111...	jpg	2.0 MiB	/home/graham/Pictures/crete
☐		img_112...	jpg	1.9 MiB	/home/graham/Pictures/crete
☐		img_112...	jpg	2.5 MiB	/home/graham/Pictures/crete
☐		img_113...	jpg	2.2 MiB	/home/graham/Pictures/crete

Type: jpg
File size: 1.9 MiB
Image size: 2592 x 1944 px
Image resolution: X = 96 Y = 96
Width: 100 %
Height: 100 %
%
Convert to: Don't change the format
☐ Replace transparent background

Banish complexity – *Converseen* is for making quick conversions, not implementing *Photoshop*-like levels of complexity.

Project website
http://converseen.fasterland.net

Embedded web browser

Qt WebBrowser 1.0

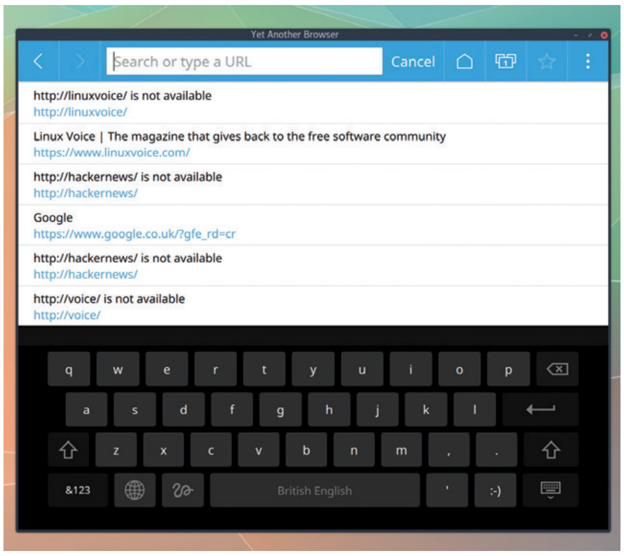
This isn't yet another web browser, but a web browser designed for embedded devices, showing off *Qt*'s rather excellent web rendering engine. For this reason, it isn't going to replace *Firefox* or *Chromium* on your desktop, but it might be able to replace your ageing browser on a low-power device, especially if that device is touch-based.

Qt WebBrowser features its own touch keyboard, which is one of the best we've seen. It quickly and instantly scrolls up whenever you need to enter some characters. You can still type, but if you're cursoring around with a remote control, for example, it is intuitive and works well. The browser itself is also lightning quick, thanks to being unburdened by any other features. There are multiple tabs, complete with lovely QML transitions, private browsing and a favourites system, which is all you need for most browsing. There's even full-screen video and audio playback if you compile with those options enabled.

Perhaps more importantly, the project is a well constructed example of how to access and code for *Qt*'s new WebEngine API, something we can't wait to see as part of *QuteBrowser*, for example (still our favourite browser, after more than six months). The small amount of code, and the use of touch and WebEngine, means this is one of the best projects to help or get started with *Qt* development.

QtWebBrowser features its own touch keyboard, which is one of the best we've seen

Considering it will run on almost anything, we can see lots of us cases for this, making us happy that *Qt* still adheres to GPLv3.



http://linuxvoice/ is not available
http://linuxvoice/

Linux Voice | The magazine that gives back to the free software community
https://www.linuxvoice.com/

http://hackernews/ is not available
http://hackernews/

Google
https://www.google.co.uk/?gfe_rd=cr

http://hackernews/ is not available
http://hackernews/

http://voice/ is not available
http://voice/

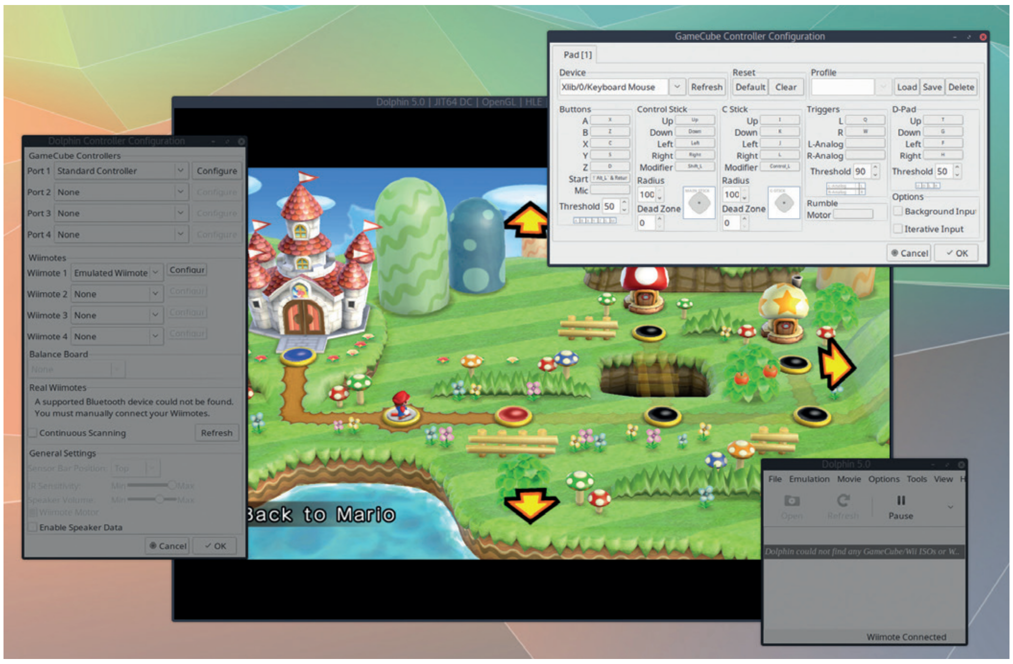
Build yourself a fully fledged web browser with touch input.

Project website
code.qt.io/cgit/qt-apps/qtwebbrowser.git

FOSSpicks Brain Relaxers

Console emulator

Dolphin 5


GameCube Controller Configuration
Dolphin Controller Configuration
Dolphin 5.0 | JIT64 DC | OpenGL | HLE
Back to Mario
Wiimote Connected

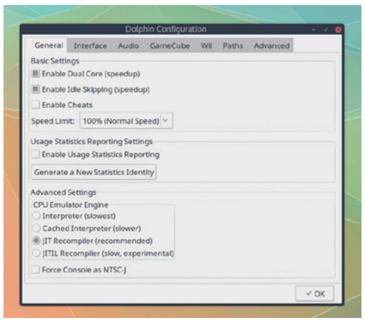
In this image, we're running a game at over three times the resolution of the original, which is one of the best reasons for using *Dolphin*.

This is version 5 of one of the most popular emulators of recent years – *Dolphin*. After being the first emulator to boot Wii games, *Dolphin* has blazed a development trail, doing the seemingly impossible of integrating Nintendo GameCube and Wii emulation, alongside support for some of the Wii's internal channels, such as WiiWare and Virtual Console.

The great thing about *Dolphin* is that it's often better than the real thing. The Wii only had composite output, for instance, and lacked an HDMI port, which means *Dolphin* could be the only way of running old games on modern resolutions, for example, or on modern televisions. Even on our modest laptop with Intel graphics, we were able to run games at native resolution easily, and often at three times the original resolution of the Wii.

Seeing those wonderful colourful graphics on a flat, wide screen, with no need for analogue/digital conversion is wonderful. It's a total transformation that may even lead us to selling our beloved office Nintendo Wii console.

The original controllers even work, although you can use the keyboard or any other controller too,


Dolphin Configuration

General | Interface | Audio | GameCube | Wii | Paths | Advanced

Basic Settings
- [✓] Enable Dual Core (speedup)
- [✓] Enable Idle Skipping (speedup)
- [ ] Enable Cheats
- Speed Limit: 100% (Normal Speed)

Usage Statistics Reporting Settings
- [ ] Enable Usage Statistics Reporting
- Generate a New Statistics Identity

Advanced Settings
- CPU Emulator Engine
  - Interpreter (slowest)
  - Cached Interpreter (slower)
  - [✓ JIT Recompiler (recommended)]
  - JITIL Recompiler (slow, experimental)
- [ ] Force Console as NTSC-J

✓ OK

Options for tweaking the emulation include dozens of graphical effects and the ability to run the emulator at greater than normal speeds.

and you can adjust many different aspect about the emulation. Graphics can be tweaked and changed, textures upgraded and modified. You can even add filtering and effects if your graphics processor has the horsepower.

The best and most esoteric mode enables 3D. If you've got the hardware, such as a 3D television, you'll be able to play some of Nintendo's best games in a way Nintendo hasn't championed since the Virtual Boy 3D headset in 1995. *Dolphin* even has support for the new wave of headsets from Oculus and HTC. We've yet to try this, but we can't wait to play *Metroid Prime* or *Super Mario Sunshine* with our heads literally 'in the game'.

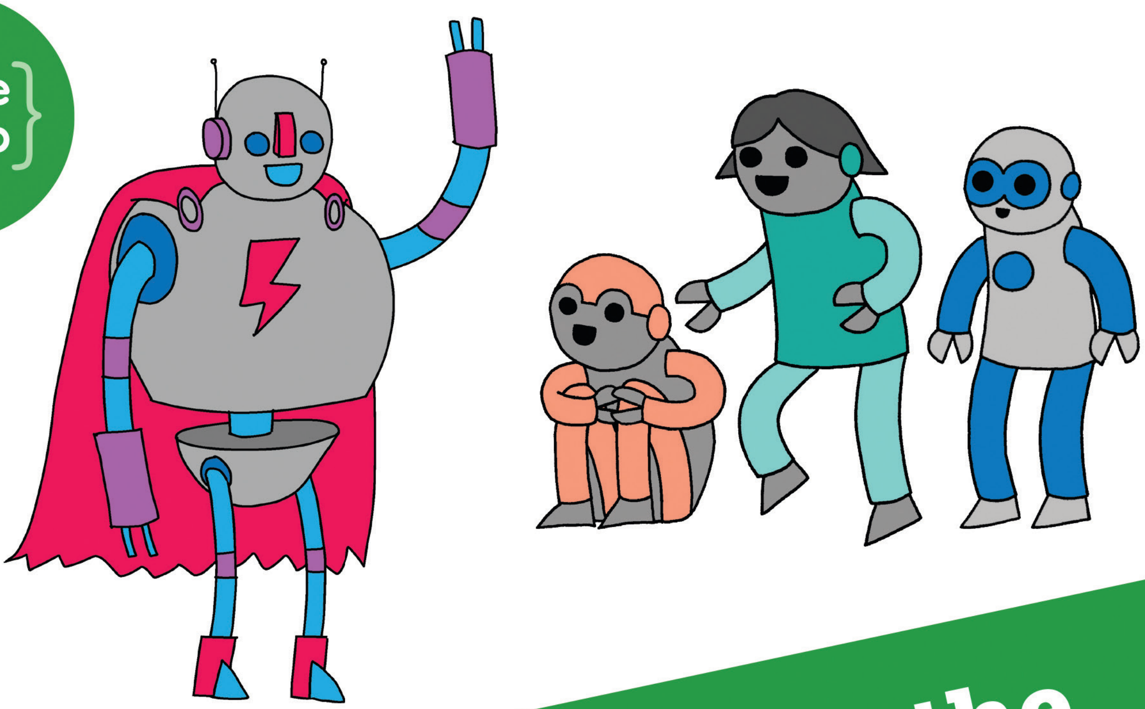
Dolphin Hotel

Version 5 is a huge upgrade that was tested for over a year before release. The texture improvements are massive, and a major bit of hacking has allowed games like *Rogue Squadron II* to run. The CPU emulation is a lot quicker, as shown by the emulator running on our humble laptop. The audio routines have also been rewritten, as have the network routines, which means you can play local games remotely over the internet. There's also the aforementioned 3D support and virtual reality.

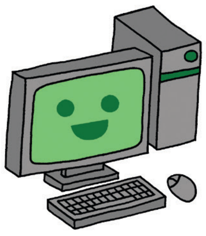
Dolphin is one of the best examples, perhaps alongside a project like *MAME*, of what can be achieved by a dedicated, clever community with a passion for achieving something. Reverse engineering, decoding, running virtual process routines, sometimes even scanning the original chips themselves – this is what emulation often involves. For this to come together into such a highly polished emulator that actually works as advertised is remarkable. We know emulation is often contentious, but this aspect fills us with respect for the developers, and part of us is very happy that many of these classic games will live on, long after the hardware has failed – much like those arcade machines of the 1970s.

Project website
https://dolphin-emu.org

{code club}



Can you help inspire the next generation of coders?



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

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

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

There are loads of ways to get involved!

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

This advertising space was kindly donated to Code Club by Linux Voice

TUTORIALS

Warning: excessive Linux knowledge may lead to fun and more efficient computing.

Mike Saunders
Will one day return home to Lave Station.

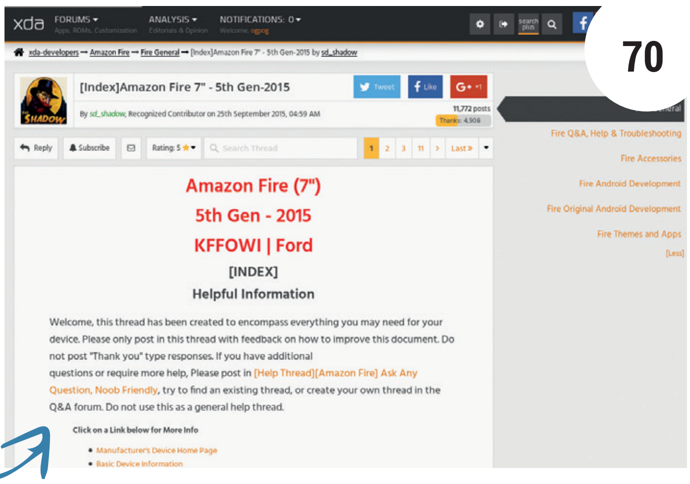
In case you haven't seen it yet, we may have discovered evidence of alien life. OK, it's a big "may", but it's probably the best shot we've ever had. A star called KIC 8462852, around 1,500 light years from Earth, is exhibiting very odd fluctuations in brightness. So odd, in fact, that they can't be explained by normal astronomical processes. Stars often dip in brightness when planets move in front of them – but by a few percent, not up to 20% like KIC 8462852. And there's no way a planet can get that big to block so much light.

So something else is probably orbiting the star and causing these weird dips. A giant cloud of comets? Hmm, maybe, but it's still not a great explanation. The one we all want to hear, of course, is aliens – maybe creating a giant network of panels to generate power. It's a long shot, and probably not the case, but with no other observed stars behaving like this, it's a big mystery. Telescopes are watching it for the next dipping event, though, so we may know the truth soon. If we can handle it, of course – I may need to watch *The Next Generation* again to prepare my tiny mind.
mike@linuxvoice.com

In this issue . . .

Kindle Fire

Linux wins again! Replace the OS on Amazon's low-budget tablet to get a fully functional Android experience for under £50.



[Index]Amazon Fire 7" - 5th Gen-2015

By sd_shadow, Recognized Contributor on 25th September 2015, 04:59 AM

Amazon Fire (7")
5th Gen - 2015
KFFOWI | Ford
[INDEX]
Helpful Information

Welcome, this thread has been created to encompass everything you may need for your device. Please only post in this thread with feedback on how to improve this document. Do not post "Thank you" type responses. If you have additional questions or require more help, Please post in [Help Thread][Amazon Fire] Ask Any Question, Noob Friendly, try to find an existing thread, or create your own thread in the Q&A forum. Do not use this as a general help thread.

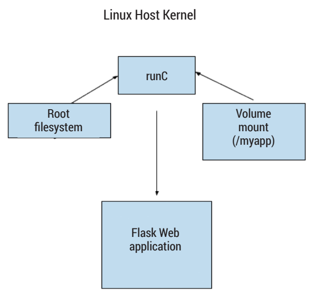
Click on a Link below for More Info
- Manufacturer's Device Home Page
- Basic Device Information

70

74

Build a Pi lockbox with an RFID

Learn all about RFID key fobs (or cards) that contain a small aerial and chip that can wirelessly interact with readers.

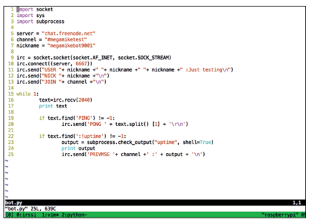


78

Docker: Linux containers done the hard way

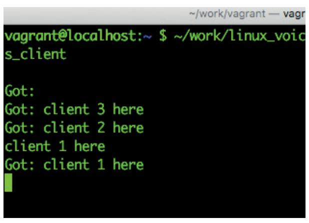
Package a basic Python Flask application – including all of its dependencies – into a container image using the power of Docker.

Coding



```
import socket
import sys
import subprocess

server = "chat.freenode.net"
channel = "#megamiketest"
nickname = "megamikebot9001"

irc = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
irc.connect((server, 6667))
irc.send("USER "+ nickname +" "+ nickname +" "+ nickname +" :Just testing\n")
irc.send("NICK "+ nickname +"\n")
irc.send("JOIN "+ channel +"\n")

while 1:
        text=irc.recv(2040)
        print text

        if text.find('PING') != -1:
                irc.send('PONG ' + text.split() [1] + '\r\n')

        if text.find(':!uptime') != -1:
                output = subprocess.check_output("uptime", shell=True)
                print output
                irc.send('PRIVMSG '+ channel +' :' + output + '\n')
~
~
~
~
~
~
~
bot.py                                                                  1,1
"bot.py" 25L, 639C
[0] 0:irssi  1:vim* 2:python-                                "raspberrypi" 0
```

Coding: IRC bots **84**
Build a community of robots to provide updates on services running on your machines.



```
~/work/vagrant — vagr
vagrant@localhost:~ $ ~/work/linux_voic
s_client

Got:
Got: client 3 here
Got: client 2 here
client 1 here
Got: client 1 here
```

Coding: Crystal **88**
Explore a general-purpose, OO language that has a syntax very similar to that of Ruby.

Get access to every Linux Voice tutorial ever published in our digital library of back-issues available exclusively to subscribers – turn to page p60 to join.

AMAZON FIRE AND CYANOGENMOD

Flash Amazon's low-budget adware-crippled tablet to get a fully functional Android experience for less than 50 notes…

JOHN LANE

Why do this?

- Get a 7-inch tablet with a 1024x600 display, 1.3GHz processor and 1GB of RAM for only £50.
- Install Linux (well, Android) on All The Things.
- Get rid of Amazon's spyware.

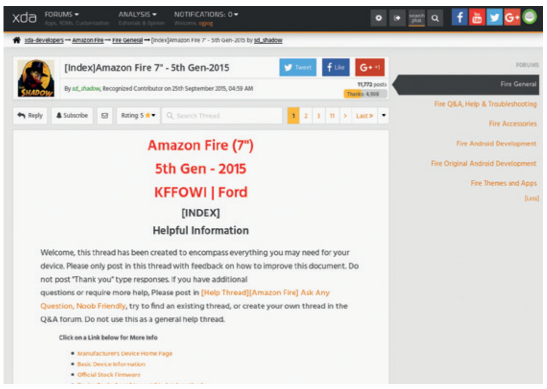
The Amazon Fire tablet was released in September 2015. It runs on Amazon's Android Lollipop-based Fire OS 5 that's designed to provide easy access to Amazon's online features and suck you into its proprietary ecosystem (and, therefore, lacks features usually found on Android such as the Google Play Store). It has a 7-inch 1024x600 display, a 1.3GHz quad-core processor, 1GB of RAM, 8GB internal storage, microSD support and a 2MP camera.

If you're buying a new tablet with the intention of replacing its firmware, save yourself some money by opting for the "with special offers" version – it's £10 cheaper and you'll be wiping the Amazon firmware anyway. (The 'special offers' refers to a screensaver that plies you with advertisements.)

Before beginning, it's worth familiarising youself with the device. There are three buttons along the top edge: the power on/off button is on its own on the right-hand side. On the left-hand side there are a pair of buttons primarily used to raise and lower the speaker volume, with volume down to the left and volume up on the right. You should also be aware of what firmware version it has, as we'll demonstrate, because this can affect its ability to be replaced.

If your device is new and has not yet been powered on, you can prevent it from updating its firmware (and avoid any unforeseen difficulties that may bring) by not connecting it to Wi-Fi.

After powering on for the first time, proceed through the country selection and then decline Wi-Fi. You should reach the home screen shortly afterwards.



[Index]Amazon Fire 7" - 5th Gen-2015

By sd_shadow, Recognized Contributor on 25th September 2015, 04:59 AM

11,772 posts
Thanks 4,909

Amazon Fire (7")
5th Gen - 2015
KFFOWI | Ford
[INDEX]
Helpful Information

Welcome, this thread has been created to encompass everything you may need for your device. Please only post in this thread with feedback on how to improve this document. Do not post "Thank you" type responses. If you have additional questions or require more help, Please post in [Help Thread][Amazon Fire] Ask Any Question, Noob Friendly, try to find an existing thread, or create your own thread in the Q&A forum. Do not use this as a general help thread.

Click on a Link below for More Info

- Manufacturers Device Home Page
- Basic Device Information
- Official Stock Firmware

The XDA Developers have done the hard work for you. This valuable resource should be your first port of call when looking for information about flashing your fire: **forum.xda-developers.com/amazon-fire/general/index-amazon-fire-2015.**

With your Fire tablet powered-on and at its home screen, check its firmware: swipe down from the top of the screen and select Settings followed by Device Options and System Updates to reveal the current firmware version.

We're using a brand-new device for this tutorial and its firmware is Fire OS 5.1.3 with a build date and time of 01:26 on 31 March 2016. We'll assume you have a firmware between 5.1.4 and 5.1.2 and that you heed the following advice:

- Firmware 5.3.1 (the latest at the time of writing) cannot be downgraded.
- Firmwares 5.1.4, 5.1.3, 5.1.2.1 can be downgraded to 5.1.2 but not to 5.1.1 or lower.
- Attempting to downgrade lower than 5.1.2 will brick your device (render it useless).
- Do not proceed unless you have firmware 5.1.4, 5.1.3, 5.1.2.1 or 5.1.2.

Once you know what firmware you have, press and hold the power button to turn the device off.

Collect the things

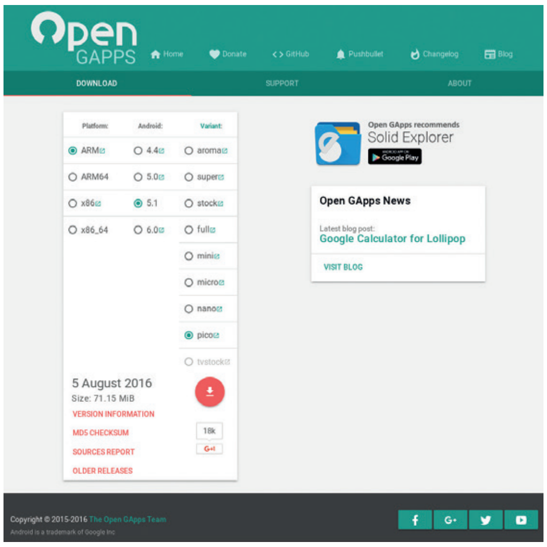
Android devices have a bootloader and a so-called "Recovery" mode that can often be replaced by a savvy user, and this is usually a first step towards running custom firmware. But in the Fire's case, both bootloader and recovery are locked and cannot be replaced (this wasn't the case with early firmwares up to 5.0.1), and the recovery mode will only accept official firmware that has been signed by Amazon.

This means that, in addition to the custom firmware, you will need root access and an alternative utility capable of loading firmware onto the device. We will download everything required to a convenient Linux box, so create a directory to put them in:

```
$ mkdir firefiles
$ cd firefiles
```

The first prerequisite is that your device has a rootable firmware (one that can be modified to make root access possible), so check its version: only version 5.1.2 can be rooted, but versions 5.1.2.1, 5.1.3 and 5.1.4 can be downgraded.

If you need to downgrade your device's firmware (which is most likely given the Fire's desire to upgrade automatically), and so long as its current version can



Platform: [✓ ARM] ARM64 x86 x86_64
Android: 4.4 5.0 [✓ 5.1] 6.0
Variant: aroma super stock full mini micro nano [✓ pico] tvstock

5 August 2016
Size: 71.15 MiB
VERSION INFORMATION
MD5 CHECKSUM
SOURCES REPORT
OLDER RELEASES

Open GApps News
Latest blog post:
Google Calculator for Lollipop

Copyright © 2015-2016 The Open GApps Team
Android is a trademark of Google Inc

Get as much or as little Google as you want from Open GApps.

be downgraded, you should download the official 5.1.2 firmware directly from Amazon. This official image has been signed by Amazon and can be installed using the device's standard recovery mode. Download the 609MB image:

```
$ wget https://kindle-fire-updates.s3.amazonaws.com/
kacn8mJNu53VtzS1JUbcF5Oh4r/update-kindle-
global-37.5.4.2_user_542168620.bin
```

To be on the safe side, check the download's hash:

```
$ sha1sum update-kindle-global-37.5.4.2_
user_542168620.bin
66b5423725b79ceb0d5866fa32ff414a99a4b50a
```

Root access can be obtained relatively easily with a tool called *KingRoot* (see **www.kingroot.net**). Download the Android version from its website and save it in your **firefiles** directory.

```
$ wget http://king.myapp.com/myapp/kdown/img/
NewKingrootV4.9.6_C151_B299_en_
release_2016_08_04_105203.apk
$ sha1sum NewKingroot*apk
bd2944bf0d5d862deef6ec167c3d0c3cdf124a9f
```

The utility that can install custom firmware is called *FlashFire* (see **flashfire.chainfire.eu** and download from **www.apkmirror.com/apk/chainfire/flashfire**). We downloaded version 0.5.2 which has an MD5 hash of 638b0ea6ec0e17f38538eee69cb99a6a.

FlashFire expects to find a root tool called *SuperSU*, so we will need to install this in place of *KingRoot* after it gains root access. There's a small utility to do this that you can download from **http://www.w0lfdroid.com/2015/05/How-to-Remove-Replace-KingUser-KingRoot-with-SuperSU.html** by following the link at the bottom of the page. You may compare the hash of your download with ours:

```
$ sha1sum Replace_Kinguser_with_SuperSU-v2.4.zip
a16b81d71c1f55ba0ec5e7f3001e482c4dfae01f
```

The last thing you need is some new firmware and you can choose between the various firmwares that support your device. Do your research and choose the one you'd like. Don't worry – you can re-flash whenever you want so you can change your mind later. You do need to make sure that the firmware image you use is for your device and this will limit your choice, restricting you to firmware based on Android 5.1 Lollipop.

One such firmware is CyanogenMod and that's what we'll use for this tutorial. There is an unofficial build of version 12.1 for the Fire 2015 tablet that you can get via **http://forum.xda-developers.com/amazon-fire/orig-development/rom-cm-12-1-2015-11-15-t3249416** where the download link is within the first forum post and leads to a download page with several mirrors (there is no official image for the Amazon Fire). The site presents the MD5 hash of the file, which should have a name similar to **cm-12.1-20160710-UNOFFICIAL-ford.zip**, so that you can verify your download.

CyanogenMod

Custom firmwares such as CyanogenMod are based on the open source version of Android, which doesn't include any of the proprietary Google applications usually found on Android devices, but you can install them yourself thanks to the Open GApps Project (**opengapps.org**).

This arrangement gives you control over what Google experience (if any) you'd like. Use the Open GApps website to download an image according to your needs. Select the ARM platform and Android version 5.1 (because these work with the Fire tablet). The "variant" determines how much (or little) Googleness you get. We went with "Pico" because it is designed for users who want the absolute minimum GApps installation available. Use your browser to download the ZIP and MD5 files from the website and check the hash (your filename will differ):

```
$ md5sum -c open_gapps-arm-5.1-pico-20160614.zip.
md5
open_gapps-arm-5.1-pico-20160614.zip: OK
```

By now everything you need will be in your **firefiles** directory and you're ready to begin:

- Amazon 5.1.2 firmware (**update-kindle-global-37.5.4.2_user_542168620.bin**)
- The FlashFire APK file
- Kingroot (eg **NewKingrootV4.9.6_C151_B299_en_release_2016_08_04_105203.apk**)
- SuperSU (eg **Replace_Kinguser_with_SuperSU-v2.4.zip**)

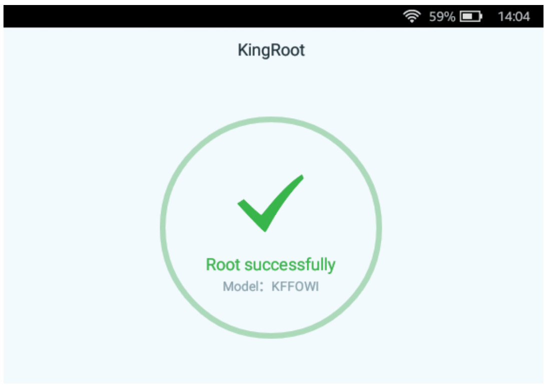
PRO TIP

There are links to the various firmwares at http://forum.xda-developers.com/showpost.php?p=62986665.

Alternative Firmwares

There are alternatives to CyanogenMod should you want to try a different firmware. You can look at **forum.xda-developers.com/amazon-fire/orig-development**, which has information about Nexus, Android Ice Cold Project (AICP), SlimLP (SlimRoms – **https://slimroms.org**) and Resurrection Remix (**www.resurrectionremix.com**), which contains the best from CyanogenMod and others.

KingRoot may take a few attempts but should eventually get root access on your Fire.



Disable FireOS OTA Updates

Should you decide to stick with FireOS but would prefer that Amazon did not update your device without your knowledge then you can use **adb** to disable Amazon's Over-the-air updates (but you need root to do so).

```
$ adb shell
$ su
$ cd /system/priv-app/DeviceSoftwareOTA
$ mount -o remount,rw /system
$ mv DeviceSoftwareOTA.apk DeviceSoftwareOTA.apkx
$ exit # from su
$ reboot
```

- CyanogenMod (eg **cm-12.1-20160710-UNOFFICIAL-ford.zip**)
- Google Apps (eg **open_gapps-arm-5.1-pico-20160614.zip**)

But, before you do, make sure the device's battery is fully charged, because power loss while writing firmware could render your device unusable.

Over the bridge

The first step, if necessary, is to downgrade FireOS so that it can be rooted. We put the device into its recovery mode and flash FireOS from a Linux box using the *Android Debug Bridge*. This is a command-line tool that lets you communicate with your device over USB; you can install it from your distro's repository on Arch this is with:

```
$ pacman -S android-{tools,udev}
$ gpasswd -a myuser adbusers
```

Now boot your device into recovery mode (from the Fire's powered-off state, press Volume Down and Power at the same time and hold until it starts). You'll see a basic text screen showing a menu entitled Amazon System Recovery and a warning may be displayed if the battery level is too low.

PRO TIP

If adb doesn't appear to work, use 'adb kill-server' to kill its daemon. This forces it to start a new one when you re-issue your adb command.

Select Apply Update From ADB (use the Volume Up/Down to navigate and Power to select). The display will show

```
Now send the package you want to apply
to the device with "adb sideload <filename>"...
```

Connect a USB cable to your computer and then to the device. Confirm the connection from the computer (with **sudo** or as **root** to ensure sufficient permissions):

```
$ sudo adb devices
```

USB debugging

To enable USB debugging on FireOS, swipe downwards from the home screen and select Settings. Now choose Device Options and find the serial number entry. Tab that seven times to unlock a new Developer Options item. Choose that and set Enable ADB to ON.

On CynaogenMod you select 'About Tablet' and then tap 'Build number' seven times to enable the developer options, which you'll then find on the Settings screen.

```
G0K0H41873351RDH            sideload
```

You can now send the firmware to the device:

```
$ sudo adb sideload update-kindle-global-37.5.4.2_
user_542168620.bin
serving: 'update-kindle-global-37.5.4.2_user_542168620.
bin'
```

During the upload, **adb** displays upload progress and the device's screen gives additional feedback:

The device automatically applies the firmware once the upload is complete and then displays the recovery menu, where you should select **reboot system now**. You can check the firmware version is the expected 5.1.2 after the system reboots (it will spend some time optimising system storage and applications). Ours shows a build date of 27 February 2016, 02:18.

With the device running FireOS 5.1.2, we can root it and then flash CyanogenMod. We'll use **adb** to install packages from Linux, so we must first enable the USB Debugging option and then permit your computer to connect. Start by testing your connection with **adb** (reconnect the USB cable if you unplugged it):

```
$ adb devices
List of devices attached
G0K0H41873351RDH            unauthorized
```

You'll see **unauthorized** the first time you do this because your tablet must permit your computer to access it (a pop-up will appear on it asking you for this permission). You also need to permit apps from unknown sources so that you can sideload application packages (the option is on the Settings screen, under Security). After all that, you'll see **device** and you can install *KingRoot*:

```
$ adb devices
List of devices attached
G0K0H41873351RDH            device
$ adb install firefiles/NewKingrootV4.9.6*.apk
6848 KB/s (13183365 bytes in 1.879s)
Success
```

You can detach the USB cable and then look on the device's home screen for the new *KingRoot* icon. Tap the icon to launch the app and follow the first few screens to start the rooting process. You will need to enable Wi-Fi so that it can download a rooting strategy. The device may reboot during the process and you may need to try a few times before being successful. As a precaution, disable Wi-Fi after rooting to prevent Amazon from updating the device.

Without ADB

Instead of using "adb" to install packages, you can use a spare SD card that is sufficiently large copy the package files to the device and install them, but you'll need a file browser to access the uploaded files.

The first step is to copy the necessary files onto it and there are two ways to do that - you can either transfer them using a spare SD card that is large or you can use "adb" to copy them over USB.

package file to a suitable place such as the "Download" directory:

$ adb push firefiles/NewKingrootV4.9.6* /storage/sdcard0/Download/

The next step is to install the *SuperSU* root app that FlashFire expects to be present. For this we once again connect the device to the computer with the USB cable and then use the utility that we downloaded previously:

```
$ cd firefiles
$ unzip Replace_Kinguser_with_SuperSU-v2.4.zip
$ cd mrw
$ adb shell mkdir /storage/emulated/legacy/mrw
$ adb push busybox /storage/emulated/legacy/mrw/busybox
$ adb push root.sh /storage/emulated/legacy/mrw/root.sh
$ adb push su /storage/emulated/legacy/mrw/su
$ adb push Superuser.apk /storage/emulated/legacy/mrw/Superuser.apk
$ adb shell
shell@ford:/ $ su
root@ford:/ # sh /storage/emulated/legacy/mrw/root.sh
```

This launches the installation of *SuperSU* on the device. Work through it by selecting the 'normal' install method, allowing it to uninstall other superuser apps and, finally, rebooting. The command-line output may show errors (eg No Such File Or Directory), which can be safely ignored.

The home screen should now show a *SuperSU* icon and the *KingRoot* one should have gone. You can perform a quick test from **adb** to acquire root privileges – the device should pop up a *SuperSU* screen to allow you to grant the root access:

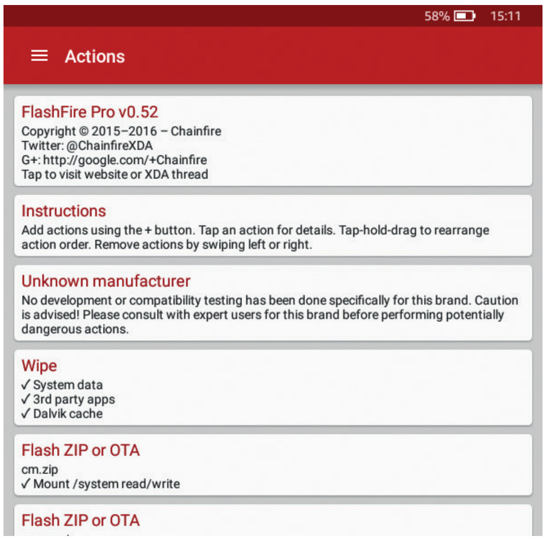
```
$ adb shell
shell@ford:/ $ su
root@ford:/ # exit
shell@ford:/ $ exit
```

Now install *FlashFire*:

```
$ adb install firefiles/'eu.chainfire.flash_0.52-52_minAPI17(armeabi-v7a,x86)(nodpi)_apkmirror.com.apk'
6993 KB/s (11921938 bytes in 1.664s)
```

FlashFire will look for firmware in the device's **Download** directory. You can use **adb** to push CyanogenMod and Open GApps to the device, but **adb** cannot push to the **Download** directory due to insufficient permissions, so we must push elsewhere and then move them as root:

```
$ adb push cm-12.1-20160710-UNOFFICIAL-ford.zip /storage/emulated/legacy/cm.zip
$ adb push open_gapps-arm-5.1-pico-20160614.zip /storage/emulated/legacy/gapps.zip
$ adb shell
shell@ford:/ $ su
root@ford:/ # mv /storage/emulated/legacy/{cm,gapps}.zip /data/media/0/Download
root@ford:/ # exit
shell@ford:/ $ exit
```


58% 15:11
Actions

FlashFire Pro v0.52
Copyright © 2015–2016 – Chainfire
Twitter: @ChainfireXDA
G+: http://google.com/+Chainfire
Tap to visit website or XDA thread

Instructions
Add actions using the + button. Tap an action for details. Tap-hold-drag to rearrange action order. Remove actions by swiping left or right.

Unknown manufacturer
No development or compatibility testing has been done specifically for this brand. Caution is advised! Please consult with expert users for this brand before performing potentially dangerous actions.

Wipe
✓ System data
✓ 3rd party apps
✓ Dalvik cache

Flash ZIP or OTA
cm.zip
✓ Mount /system read/write

Flash ZIP or OTA

FlashFire makes it possible to install custom firmware on Amazon's locked-down device.

Now we can run *FlashFire* and finally get rid of FireOS. Locate the *FlashFire* icon on the home screen and tap it to get started. Grant root access when requested and page through the following pop-ups until the main screen appears where you configure the tasks that it should perform. Use the red plus-sign icon at the bottom right-hand corner of the screen to add tasks as follows:

- Wipe, check system data, 3rd party apps and Dalvik cache.
- Flash ZIP or OTA, select **cm.zip** from **Download** and accept the default options.
- Flash ZIP or OTA, select **gapps.zip** from **Download** and accept the default options.

To complete adding each task, tap the tick-mark that you'll find in the top-right corner of the screen. After adding the three tasks, drag the wipe task so it appears before the two flashing tasks. Then hold your breath and press the big "FLASH" button.

You should see lots of command line output on the device's screen and, after a while, it should reboot. You'll see the CyanogenMod logo for a while and, eventually, the device will show that Android is starting (while it optimises apps). You'll shortly be delivered into the CyanogenMod setup screen, which you can follow through yourself to select your country and connect to Wi-Fi and your Google account before being presented with the home screen. Congratulations, you've flashed your device!

PRO TIP

Android optimises installed apps into and stores them in its Dalvik Cache area so they load quicker. Dalvik is the Java Virtual Machine used by Android. The cache should be wiped when updating or replacing firmware so that the cache is rebuilt.

John Lane provides technical solutions to business problems. He has yet to find something that Linux can't solve.

CREATING A LOCKBOX USING RASPBERRY PI

Les Pounder is creates a locked box to protect his precious choccy biccies.

LES POUNDER

Why do this?

- Use a new input method
- Keep yourbiscuits safe

You will need

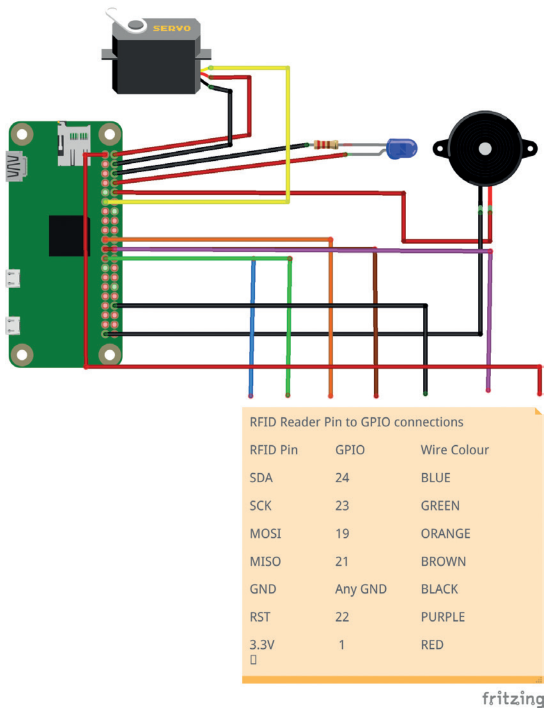
- A Raspberry Pi Zero
- An RFID-RC522 RFID reader
- An LED
- A 220Ω resistor
- A buzzer
- Lots of female to female jumper wires
- A USB battery pack

RFID (Radio Frequency Identification), key fobs or cards contain a small aerial and a chip that can wirelessly interact with readers. They are commonly used for security passes and contactless payment cards such as the Oyster cards used on London's public transport. Here we shall be using an RFID card and reader to trigger a servo motor. The servo motor will be used as a lock for box full of stuff, and we'll also add an LED and a buzzer to alert us to intruders.

We used a Raspberry Pi Zero and a USB battery pack for this project, giving us a truly small portable system, and a Dremel tool to cut a hole in our plastic.

Our RFID reader is an RFID-RC522, which can be found on eBay for around £5. The RFID reader uses SPI, a specialist interface and protocol. On the Raspberry Pi some of the GPIO pins double-up as SPI pins, and we can connect our reader as follows

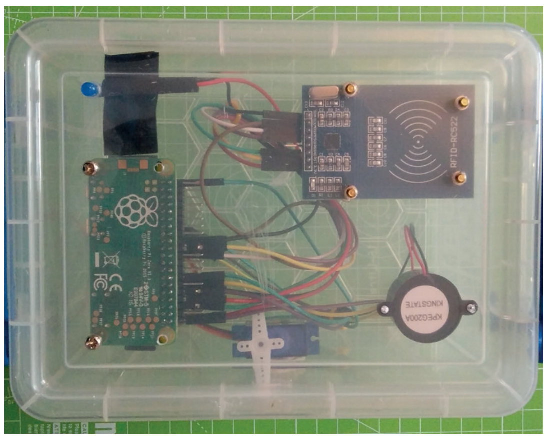
SDA → 24
SCK → 23
MOSI → 19
MISO → 21
IRQ → NO CONNECTION



RFID Reader Pin to GPIO connections

RFID Pin	GPIO	Wire Colour
SDA	24	BLUE
SCK	23	GREEN
MOSI	19	ORANGE
MISO	21	BROWN
GND	Any GND	BLACK
RST	22	PURPLE
3.3V	1	RED

The circuit for this project is simple, but involves a lot of wiring. Take your time – build and test one section at a time.


Our completed project resides in a clear plastic box, enabling anyone to see how it works. For your version, transparency is optional.

GND → Any GND Connection
RST → 22
3.3V → 1

For our servo, LED and buzzer connections please refer to the diagram, available as a download along with the code for this project.

Software setup

Using the Raspberry Pi Configuration tool, found in the Preferences menu, you can easily enable the SPI interface via the Interfaces tab. You will need to reboot your Raspberry Pi for the changes to take effect.

Next we're going to install some software, so we will first need to ensure that our Raspberry Pi is connected to the internet and that the list of software available for our Raspberry Pi, commonly referred to as a repository, is up to date. We will then install the Python developer tools. Open a Terminal and enter the following:

```
$ sudo apt update && sudo apt upgrade -y
$ sudo apt install python-dev
```

Once this has been completed we shall now set up the software that will enable our project to talk to the SPI interface. **SPI-Py** is a Python library that enables a connection between the SPI interface and Python. To install the latest version, in a terminal type the following, pressing Enter to run the command:

```
$ git clone https://github.com/mab5vot9us9a/SPI-Py
```

With the download complete, we now need to navigate into the directory and run the setup tool.

```
$ cd SPI-Py
$ sudo python setup.py install
```

With the installation complete, we need to return to the Home directory and download the software that will enable our RFID reader to talk to our project.

```
$ cd ..
$ git clone https://github.com/mxgxw/MFRC522-python
```

Once the download is complete, it prudent to reboot to ensure that all of the changes take effect.

Once rebooted, open a terminal and navigate to the directory containing the code for our RFID reader.

```
$ cd MFRC522-python
```

Now let's test that our reader is working correctly. In the terminal type the following to run a read test. It will read your RFID cards/fobs and tell you their details.

```
sudo python Read.py
```

You should now see information scrolling up the screen, which will look similar to this.

```
Welcome to the MFRC522 data read example
Press Ctrl-C to stop.
Card detected
Card read UID: 133,48,217,101
Size: 8
Sector 8 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

We have successfully installed the RFID reader and can now continue with our project.

Coding the project

For this project we shall be coding in Python 2, as this is necessary for the libraries used to control the RFID reader. On the Raspbian desktop go to the main menu at the top-left of the screen. Navigate to Programming and select Python 2. Once the Python 2 editor loads, click on File > New to open a blank document. Immediately save the new window by clicking on File > Save and name the file **lockbox.py**. Save your work to the **MFRC522-python** directory so that we can use that library. Subsequent saves will now be much quicker – remember to save often!

We start our code by including a special line of code that we will use to instruct Python where to look for our Python executable. This will enable us to call the application from the command line and use the lock box project as an embedded project.

```
#! /usr/bin/env python
```

Our first major section of code is our importing a number of libraries that will extend the functionality of our project. Our first import is the GPIO library necessary for working with the GPIO pins. Next we import the **MFRC522** RFID library. Finally we import the **sleep** function from the **time** library; we shall use this to pace our project.

```
import RPi.GPIO as GPIO
import MFRC522
from time import sleep
```

Next we set up our GPIO to use a logical pin numbering layout. Odd numbers are the pins nearest the CPU; even numbers are nearest to the edge of the Pi. This numbering system is required for the RFID library.

```
GPIO.setmode(GPIO.BOARD)
```

Our next section of code is to create two variables used to store the location of the LED and buzzer, which will be used to raise the alarm should an intruder try to gain access.

```
LED = 8
BUZZER = 10
```

Next we'll set up three GPIO pins as outputs. Our first is pin 11, used to control the servo. Our other pins are LED and BUZZER, which we created variables for.

```
GPIO.setup(11,GPIO.OUT)
GPIO.setup(LED,GPIO.OUT)
GPIO.setup(BUZZER,GPIO.OUT)
```

An extra configuration detail is to create an object, a variable called **pwm**, that will store the pin connected to the servo, and also the frequency required to correctly communicate with the servo.

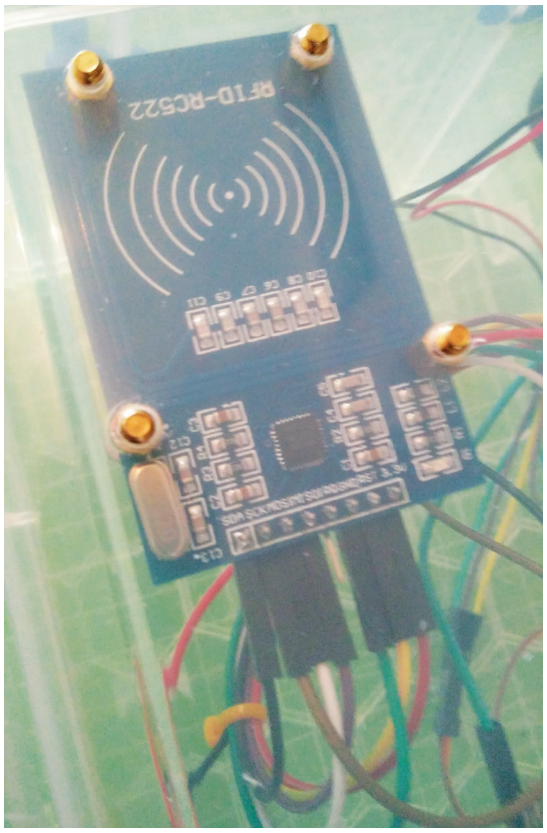
```
pwm=GPIO.PWM(11,50)
```

Next we create three more variables. The first two variables are used to store two states. **Continue_reading** is an instruction to check that the code should still be running, which means True. The **locked** state that the lockbox should default to being locked. The third variable creates an object using the **MFRC522** library.

```
continue_reading = True
locked = True
MIFAREReader = MFRC522.MFRC522()
```

We now move to the main body of code. We start by using a **try...except** construction. This is used to enable a method to exit the project cleanly should any errors occur.

The RFID-RC522 is a rather old reader, but it can be picked up rather cheaply via eBay. You will need to solder the pins to the board.



Mounting our components in the box is simple – using a drill we made holes to hold components. A slot on the edge is also required for the servo lock to operate.

```
try:
```

Our next line of code is indented, either one tab or four spaces. Here we ensure that our lock is in position by sending a pulse to instruct the servo to move to the left, forcing the arm of the servo inbetween the plastic of the case.

```
    pwm.start(5)
```

Inside our 'while' loop we create a conditional test that will check that the card has been read correctly

We next create a **for** loop, which shall be used to sound our buzzer twice, indicating that the code is working and ready to go. We use the **GPIO.output** function to turn on the buzzer, then wait for 0.1 seconds, then we turn off the buzzer and sleep for a further 0.1 seconds.

```
    for i in range(2):
        GPIO.output(BUZZER, True)
        sleep(0.1)
        GPIO.output(BUZZER, False)
        sleep(0.1)
```

We now move out of the **for** loop, but remain inside the main loop. In order to debug that our code is working, we add a **print** function that will advise the user to present their card to the reader.

```
    print("Please present your RFID card to the reader -
this can only be seen in debug")
```

We now create a new loop inside our main loop. This loop checks the value of the variable **continue_reading** and if it is True, then the code will continue.

```
    while continue_reading == True:
```

Inside this new loop we first turn on the LED before we start to scan for RFID cards or keyfobs. We then pause for half a second to save taxing the CPU too much.

```
        GPIO.output(LED, True)
        (status,TagType) = MIFAREReader.MFRC522_
Request(MIFAREReader.PICC_REQIDL)
        sleep(0.5)
```

Still inside our **while** loop, we now create a conditional test. This test will check that the card has been read correctly, and if so the indented code is executed. In this case it prints to the Python shell that a card has been detected. Then it reads the UID (Unique Identification) number, and stores it as a variable called **uid**, which we then print to the shell for debug.

```
        if status == MIFAREReader.MI_OK:
            print("Card detected")
            (status,uid) = MIFAREReader.MFRC522_Anticoll()
            print(uid)
```

We are still inside the conditional test that checks the card has been read correctly, and now we create another conditional test. This time the test compares the UID of the RFID card/fob against what we expect the UID to be. The expected UID is a list of five numbers seperated by commas. When the RFID card/fob is read, the UID is stored as a list called **uid**, which is then compared to this hard-coded list. If the card is correct, it prints that a key has been detected.

```
            if uid == [195, 87, 55, 213, 118]:
                print("KEY DETECTED")
```

We now use another conditional test. This test checks the status of the lock, our servo. It works for both unlocking and locking the box. If the lock is on, we change the status of the **locked** variable to **False**, and then turn the servo to the unlock position. The test also checks to see if the box is unlocked. If

Servos are really cheap on eBay, and can be picked up for around £5. This type is the most common and works well for most applications.

unlocked then the variable **locked** is changed to **True** and the servo returns to the locked position.

```
    if locked == True:
      locked = False
      pwm.start(5)
    elif locked == False:
      locked = True
      pwm.ChangeDutyCycle(10)
```

We now come out of the conditional lock test and return to the test that ensures the correct card has been presented. We use an **else** condition, which requires no test, which is triggered when an unknown card is presented. So when an unknown card is presented we print an alert to the Python shell, again for debugging purposes. We next use a **for** loop that has a range, used to control the number of times the loop will iterate for. In the **for** loop we turn the LED and buzzer on, then wait for 0.2 seconds, before we then turn them both off and wait for a further 0.2 seconds. This will give us a flashing LED and beeping buzzer.

```
  else:
    print("ALERT ALERT ALERT")
    for i in range(3):
      GPIO.output(LED, True)
      GPIO.output(BUZZER, True)
      sleep(0.2)
      GPIO.output(LED, False)
      GPIO.output(BUZZER, False)
      sleep(0.2)
```

We now exit out of all the conditional tests and return to the **try..except** construction that we're using to handle exiting the application. Our **except** section uses a **KeyboardInterrupt**, in this case Ctrl+C, to exit the application by changing the **continue_reading** variable to **False**; then we clean up the GPIO pins, returning all of them to their original state before we started the application. Finally we print that the application is exiting. We introduce **\n**, which is Python for introducing a blank line before the text is written.

```
except KeyboardInterrupt:
  continue_reading = False
  GPIO.cleanup()
  print("\nExiting the application")
```

With the code completed, now is the time to test before we assemble the project. In *Idle* ensure your work is saved, and then click on Run > Run Module. Place your RFID card/fob in front of the reader and check that it works. To stop the code, press Ctrl+C.

Now that our project works, it is time for us to enable it to run autonomously. Our first step is to make our code executable, and we can do this using a command called **chmod**. On your Raspberry Pi, open a terminal. In the terminal, navigate to where your project code is located. Type in the following code.

```
sudo chmod +x lockbox.py
```

In the terminal, try executing the code by typing.

```
./lockbox.py
```

To stop the code, press Ctrl+C.

We now have an executable Python script, but now we need to automatically run the script on boot. For this we stay in the terminal and use a command called **crontab**. **Crontab** is an easy-to-use editor that enables a user to specify what code to run at a given time or circumstance. To launch **crontab**, type:

```
crontab -e
```

If you see a prompt asking you to choose an editor, pick *Nano*, the default option.

Crontab is really a text file, so in the text file navigate to the bottom of the file using the arrow keys.

We need to add a line that instructs **crontab** to run our code on reboot. So we need to tell **crontab** where our code is, for example our code is in **/home/pi/MFRC522-python**. Add a new line as follows

```
@reboot /home/piMFRC522-python/lockbox.py
```

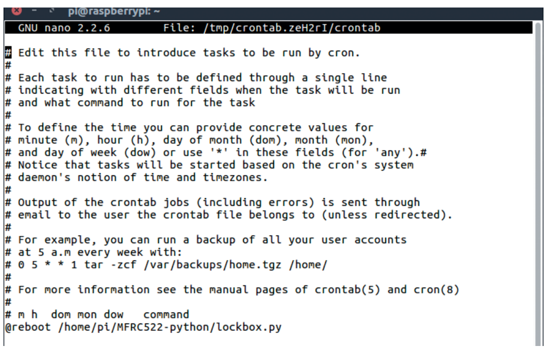
Next, press Ctrl+X to exit **crontab**. You will be prompted to save, then Enter to confirm the filename.

Now we're ready to perform a final test before fitting the hardware into a box. Reboot your Raspberry Pi; you can do this in the terminal by typing.

```
sudo reboot
```

Once your Pi reboots, your code should auto run – now present your RFID card to the reader and watch the box unlock. It's like magic!

Les Pounder makes things, breaks things, and spends the rest of his time teaching teachers about the new IT curriculum.



```
pi@raspberrypi: ~
  GNU nano 2.2.6          File: /tmp/crontab.zeH2rI/crontab

# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
@reboot /home/pi/MFRC522-python/lockbox.py
```

Editing **crontab** is easy, requiring only one extra line be added to the end of the file, readying our Pi to run our code on boot.

Choice of motors

For this tutorial we used a servo motor to control the lock of our lockbox. Servos are tricky to use, requiring the user to perform a little maths to find the correct duty cycle in order to rotate them to the desired orientation. So what other motors could we have used?

A stepper motor is another highly precise motor. Typically stepper motors can be found in printers, scanners and DVD drives. A stepper has multiple coils organised into groups. As we provide power to each group in turn, the motor will move one step, slowly but with plenty of torque. Each stepper motor requires six connections to the GPIO, for power, Ground, and a GPIO pin for each group of coils. Our final motor is the humble DC motor used to power robots. DC motors are instant-on and come in a range of speeds and torques. To precisely control a DC motor we can use Pulse Width Modulation, PWM, which we have used with our servo in this project.

A word of caution: DC motors should not be directly attached to the GPIO as they can cause damage to the pins. Instead, use a motor control board such as an L298D or L9110S to act as a buffer to the pins.

DOCKER: CREATE LINUX CONTAINER IMAGES THE

Containers are easy – so make it harder with Docker Build and learn how it's done.

AMIT SAHA

Why do this?
- Easily package programs into self-contained operating images
- Buzzwordify yourself with Docker, containers and runC

In this article, we will package a basic Python *Flask* application. *Flask* is a micro web framework for Python, and while our application will be small, it will be close to an application that you would usually package up as a container image. The image will have all the dependencies that your application will need and hence we should be able to essentially hand it over to the container runtime such as *Docker* or runC (discussed later). In addition, we want to have two separate images – one we use for testing and the other for deployment. The reason is that the image for testing will have additional dependencies that we don't need to have in the image we use for deployment. You can find the application in the **myapp** directory at **https://github.com/amitsaha/linux_voice_7**. When we use the second image for deployment, we expect that we will have our application listening for incoming HTTP requests on port 5000. Now, it's worth noting that a container image is a self-contained Linux distro – albeit running on the same kernel as the host distro. Hence, in addition to our *Flask* application, we will also need to include everything that's needed to run our application. This includes the Python interpreter (CPython 3) and everything else that it needs to run, such as the entire standard Python library and the third-party dependencies we depend on, ie *Flask* and its dependencies. This is all of course in addition to the system shared libraries that Python itself needs to run (Figure 1). To sum up, the first container image we will be building will include our simple *Flask* application as well as everything that we need to run it.

> The first container image we will build will include our Flask application and everything we need to run it

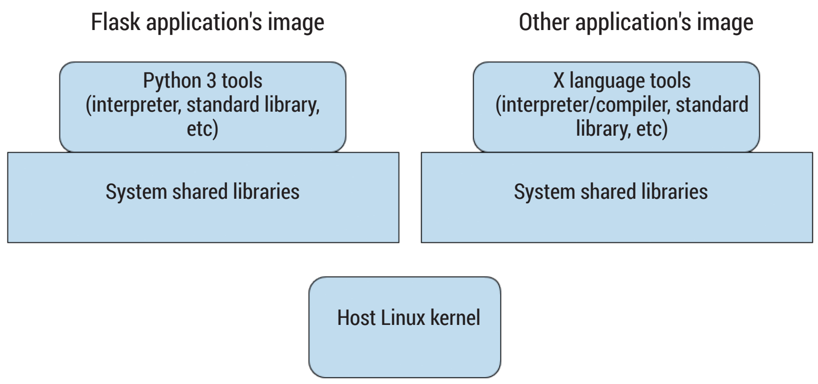

Flask application's image
Python 3 tools (interpreter, standard library, etc)
System shared libraries
Other application's image
X language tools (interpreter/compiler, standard library, etc)
System shared libraries
Host Linux kernel

Figure 1: Bundling software for running in a container.

Building our first image

Before we do anything, let's create a new directory, **linux_voice_7** somewhere in our home directory, which we will use as the working directory here. Next we want to choose our base Linux distribution in which we will run our Flask application. I am currently on Fedora 24 and I will use Fedora 24 as the base Linux distro too. The following command will create a directory structure you will find similar to any Linux system, and also install Python 3 development tools such as Python 3 itself and **pip**:

```
$ sudo dnf --installroot=`pwd`/rootfs --disablerepo=*
--enablerepo=fedora --releasever=24 install python3 &&
sudo dnf --installroot=`pwd`/rootfs --enablerepo=fedora
clean all
```

Now that we have the base system for our *Flask* application, let's install flask in it:

```
$ cd rootfs/usr/bin
$ sudo ./python3 pip3 install flask
```

At this stage, here's our current directory structure:

```
$ pwd
linux_voice_7
$ tree -L 1
rootfs
myapp
```

Now we have our Linux image ready that is capable of running our *Flask* application. We will use this image as the root filesystem from which we will bootstrap our Linux container. Next, we need a way to run a Linux container using our root filesystem. We will do so using runC – a Linux container runtime similar to *Docker*. It is also the Open Container Initiative's (OCI) reference implementation of a software container runtime.

Installing runC

runC (**https://github.com/opencontainers/runc**) is written in Golang, and next we will download the source, build and install it.

The steps below are tested on Fedora 24, but it should work on another distro. If you don't have the Go tools (compiler and other tools) installed, you can either use the distro's package manager (on Fedora 24, you can use **sudo dnf -y install golang**) to install them or download the Linux binary and follow the instructions on the install page at **https://golang.org/doc/install**.

Once the installation steps are completed, open your favourite terminal emulator and type **go version** and it should print a message similar to below:

```
$ go version
go version go1.6 linux/amd64
```

We next need to set up our workspace. If you already have **GOPATH** set up, you may skip ahead. Create a **golang** directory in your home directory (**/home/<user>**) and a sub-directory, **src** inside it. The directory tree for your workspace should look as follows:

```
golang/
`-- src
```

The Go compiler and other tools expects the **GOPATH** environment variable to point to the workspace directory, so set the following to your **.bashrc** or the file relevant to your shell, so that it is always set when you start a new terminal session (Replace **<user>** with your username):

```
export GOPATH=/home/<user>/golang
```

Once you have set the above, start a new terminal session and type in **go env GOPATH**:

```
$ go env GOPATH
/home/<user>/golang
```

If you see your **GOPATH** printed correctly similar to the above, we're all set to get the source for runC, build and install it.

```
$ sudo dnf -y install libseccomp-devel # for Fedora
$ mkdir -p ~/golang/github.com/opencontainers
$ cd ~/golang/github.com/opencontainers
$ git clone https://github.com/opencontainers/runc.git
$ cd runc
$ make
$ sudo make install
```

At this stage you have **runc** installed and accessible via the **runc** command. Just typing it in and pressing E displays all the different **runc** sub-commands. The **--version** flag describes the version:

```
$ runc --version
runc version 1.0.0-rc1
commit: 99c683a84fcfb0675883dc98eaa8c0bc3311e436
spec: 1.0.0-rc1
```

The **version** is the version of **runc**; **commit** is the Git commit hash that we build **runc** from; and **spec** is the version of the OCI specification that the runtime complies with.

Flask application in a container

Before we can run our *Flask* application in a container using runC, we will have to generate a runC configuration or a spec for the container. This configuration specifies, among various other things, the process that we want to run in the container. We can generate this spec using **runc spec**, but it doesn't allow any customisation and hence we will need to resort to hand-editing JSON files. Instead, we will use ocitools (**https://github.com/opencontainers/ocitools**). Let's get the source, build and install it:

```
$ cd $GOPATH/src/github.com/opencontainers/
$ git clone https://github.com/opencontainers/ocitools
$ cd ocitools
$ sudo dnf -y install golang-github-cpuguy83-go-
md2man # this install go-md2man
$ sudo make install
$ ocitools --version
oci version 0.0.1
```

Let's generate the configuration:

```
$ pwd
/home/vagrant/work/linux_voice_7
$ ocitools generate --os=linux --bind myapp:/myapp:ro
--args "python3" --args="/myapp/app.py" --output
config.json
```

You will now see that you have a **config.json** file in the current working directory. We can use the **jq** program (JSON command line processor) to output the most interesting parts of this configuration (To install **jq**, use **sudo dnf -y install jq**):

```
$ cat config.json | jq 'keys'
[
 "hooks",
 "hostname",
 "linux",
 "mounts",
 "ociVersion",
 "platform",
 "process",
 "root"
]
```

The configuration has the above seven configuration keys. The **hostname** key specifies the hostname for the container and defaults to **mrsdalloway**; the **linux** key specifies the Linux specific container configuration; the **mounts** key specifies

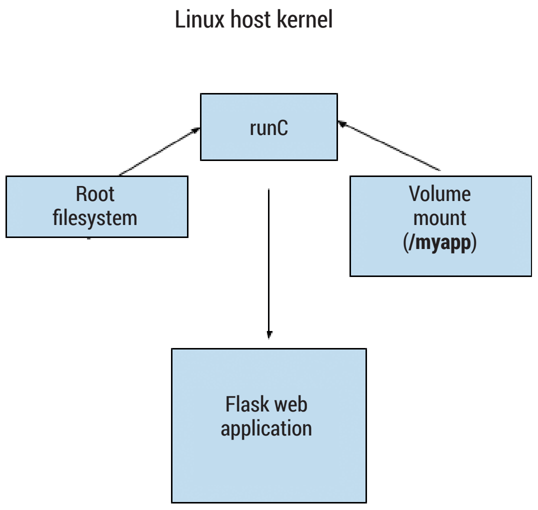


Figure 2: Our *Flask* application running under runC.

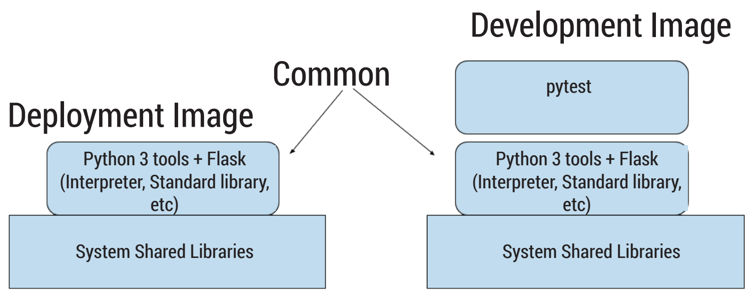

Development Image
Common
Deployment Image
pytest
Python 3 tools + Flask (Interpreter, Standard library, etc)
System Shared Libraries
Python 3 tools + Flask (Interpreter, Standard library, etc)
System Shared Libraries

Figure 3: Comparison of the two images

paths to be mounted inside the container in addition to the root filesystem. This is where we also specify any "bind mounts" – that is, if we want to make a file/directory from our host system available inside the container, this is where we would do it. Note that when we generated the runC configuration above, we specified **--bind myapp:/myapp:ro**. This meant that we want to bind mount our host directory, **myapp**, which has our *Flask* application inside our container at **/myapp** and we want to mount it as read-only. If we check the mounts key of the generated **config.json**, we will see that there is an entry for this:

```
$ cat config.json | jq '.mounts'
..
..
{
  "destination": "/myapp",
  "type": "bind",
  "source": "myapp",
  "options": [
    "bind",
    "ro"
  ]
}
```

PRO TIP
Think of choosing our base distro as choosing a Docker base image that you specify in the FROM command in the Dockerfile.

The **process** key and specifically the **args** key inside it describes the process that we will run inside the container:

```
$ cat config.json | jq '.process.args'
[
 "python3",
 "/myapp/app.py"
]
```

The **root** key specifies the path of the root filesystem, which defaults to **rootfs**:

```
$ cat config.json | jq '.root'
{
 "path": "rootfs"
}
```

OK, now we are all set to run our *Flask* application in a container:

```
$ pwd
/home/vagrant/work/linux_voice_7
$ tree -L 1
.
└─ config.json
└─ myapp
└─ rootfs
$ sudo /usr/local/sbin/runc run flask-app-1
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

We now have our *Flask* application running inside the container and listening on port 5000 (Figure 2). However, our container runs in its own network namespace, which means that to be able to access our application, we must be "inside" the container itself. Let's do that next.

The argument **flask-app-1** to **runc run** is the container ID of our container and must be unique for currently running containers on the system. We can use **runc list** to show the currently running containers:

```
$ sudo /usr/local/sbin/runc list
ID          PID      STATUS     BUNDLE          CREATED
flask-app-1  12101    running   /home/vagrant/work/
linux_voice_7  2016-09-03T10:32:12.521065663Z
```

Now, we want to execute a *Bash* shell inside this container, which we can do using the **runc exec** command from another terminal:

```
$ sudo /usr/local/sbin/runc exec -t flask-app-1 bash
bash-4.3# python3
Python 3.5.1 (default, Mar  4 2016, 15:21:15)
[GCC 6.0.0 20160302 (Red Hat 6.0.0-0.14)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import http.client
>>> conn = http.client.HTTPConnection("127.0.0.1:5000")
>>> conn.request("GET", "/")
>>> r1 = conn.getresponse()
>>> r1.read()
b'Hello World!'
>>>
```

The **-t** switch allocates a new pseudo TTY for us, and then we start the Python 3 interpreter inside it and make a request to our web application running on port 5000. We get a "Hello World" back from our web application as a response. We can exit out of the container here. We can stop or kill our container and our *Flask* application using Ctrl+C.

Before we proceed, let's move our current **rootfs** and **config.json** into a new sub-directory – **deployment** such that our current **linux_voice_7** directory looks like this:

```
$ tree -L 2
.
└─ deployment
|  └─ config.json
|  └─ rootfs
└─ myapp
|  └─ app.py
|  └─ test_app.py
└─ README.md
```

We will also have to change the source for **myapp** to "source": "../myapp", in the **config.json** file now. Now, to start a container serving our *Flask* application, we will first have to **cd** into the **deployment** directory and then execute **runc run flask-app-1**.

A second container image

We have seen that our *Flask* application works; next we'll want to run tests before we deploy it and of course we want our tests to be also run in a container.

The test runner that we want to use is **pytest** (**http://doc.pytest.org/en/latest**). This means that we will need to install **pytest** in our root filesystem before we run the testing container. We could create a second image like we did the first and have two different images on our disk. However, that means that we are literally storing two copies of the first image (Figure 3).

It would be better if we could use the first image as a base image and then just have the **pytest**-relevant files "applied" over it to get our second image and start the testing container. Note that we don't want to change the first image's root filesystem in any way. We can do so using the Overlayfs filesystem on Linux. The concept of Overlayfs is simple: it enables us to have what is known as "union filesystems" ie combine one or more directories and merge them as if they were one giant directory (Figure 4). Let's try it out:

```
$ pwd
/home/vagrant/work/linux_voice_7
$ mkdir pytest
$ mkdir merged
$ mkdir workdir.overlay
$ sudo mount -t overlay overlay -o lowerdir=./
deployment/rootfs,upperdir=./pytest,workdir=./workdir.
overlay ./merged
```

We created three sub-directories: **pytest**, **merged** and **workdir.overlay**. The **merged** directory will have the merged contents of our lower and upper directories. The **workdir.overlay** directory is needed by Overlayfs for its own purposes.

Finally, we use the **mount** command to create an overlay filesystem specifying the **lowerdir** (lower directory), **upperdir** (upper directory), **workdir** (working directory) and finally the mount point, **merged**.

If we now **cd** into the **merged** directory, we will see the contents of the **./deployment/rootfs** directory:

```
$ cd merged/
vagrant@localhost:~/work/linux_voice_7/merged $ ls
total 64
lrwxrwxrwx.  1 root root    7 Feb  4  2016 bin -> usr/bin
dr-xr-xr-x.  2 root root 4096 Feb  4  2016 boot
drwxr-xr-x.  2 root root 4096 Sep 11 01:18 dev
drwxr-xr-x. 25 root root 4096 Sep 11 01:18 etc
...
...
```

Let's now install **pytest** in this merged filesystem:

```
$ cd merged/usr/bin
$ sudo ./python3 ./pip3 install pytest
..
```

Let's now unmount **merged**:

```
$ sudo umount ./merged
```

Now, let's see what we have in the **pytest** directory, which was originally empty:

```
$ ls pytest/
total 4
drwxr-xr-x. 4 root root 4096 Sep 11 01:18 usr
$ du -sh pytest/
2.1M    pytest/
```

We see that there is a new sub-directory **usr** in this directory, which has the **pytest** package in it. In other words, these are the additional files that were written to the **merged** directory when we installed **pytest**. The lower image (**./deployment/rootfs**) was not modifed. The key thing to note here is that the **pytest** directory now has the difference between our deployment image and what we need to run our testing container. We will next see how we can combine them to get our second image.

We will create a **testing** sub-directory and two new sub-directories in it: **rootfs** and **rootfs.empty**:

```
$ pwd
/home/vagrant/work/linux_voice_7/
$ mkdir -p testing/rootfs -p testing/rootfs.empty
```

Now, we'll create an overlay filesystem combining the deployment root filesystem (which has *Flask* installed) and our **pytest** directory (which has only **pytest** installed in it) to give us a new root filesystem in **testing/rootfs**:

```
$ sudo mount -t overlay overlay -o lowerdir=./pytest:./
deployment/rootfs,upperdir=./testing/rootfs.
empty,workdir=./workdir.overlay ./testing/rootfs
```

The above command is an example of combining multiple lower filesystems using Overlayfs. When multiple directories are specified separated with a colon (**:**) the first directory specified will be the topmost layer followed by the others in order.

At this stage the **testing/rootfs** now has **pytest** as well as *Flask* installed in it, and hence we are now ready to configure our testing container:

```
$ cd testing
$ cp ../deployment/config.json .
$ pwd
/home/vagrant/work/linux_voice_7/testing
$ tree -L 1
.
└── config.json
└── rootfs
└── rootfs.empty

2 directories, 1 file
```

We modify the **config.json** to run **pytest** on startup and also change the current working directory via the

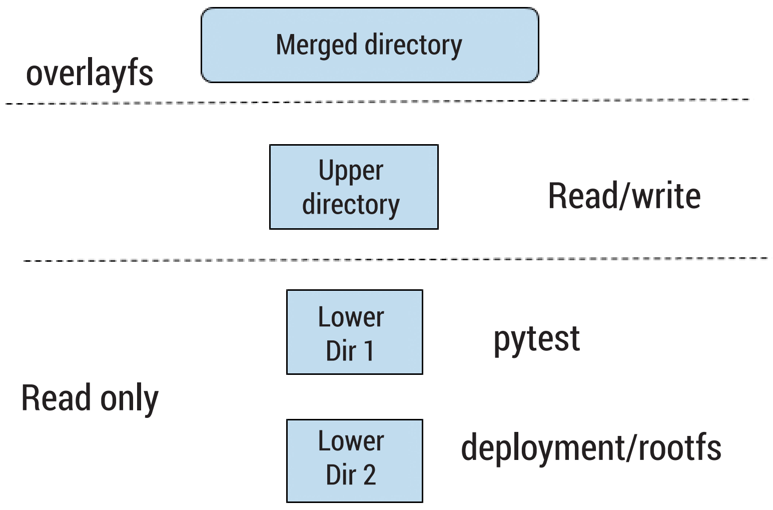


Figure 4: Demonstration of Overlayfs.

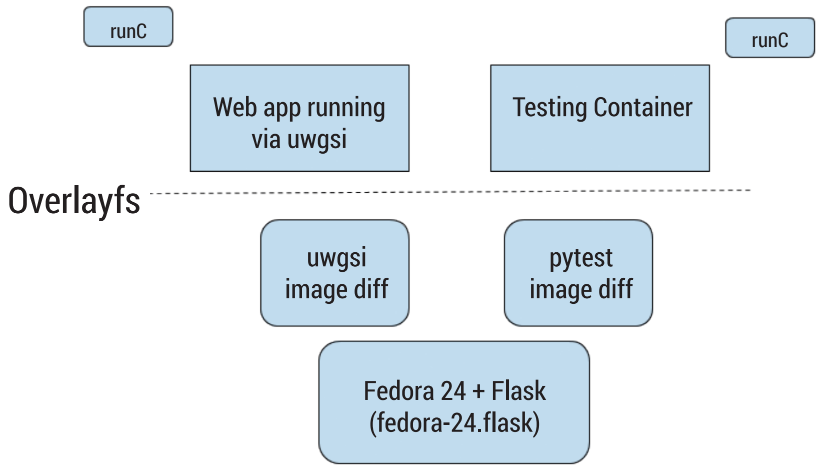


Figure 5: Using our **fedora-24.flask** image as the base image for multiple container images.

cwd key in the **process** object:

```
$ cat config.json | jq ".process.args,.process.cwd"
[
 "/usr/bin/python3",
 "/usr/bin/pytest",
 "-p",
 "no:cacheprovider"
]
"/myapp"
```

Let's run our testing container now

```
$ pwd
/home/vagrant/work/linux_voice_7/testing
$ sudo /usr/local/sbin/runc run flask-app-testing
================ test session starts ================
platform linux -- Python 3.5.1, pytest-3.0.2, py-1.4.31,
pluggy-0.3.1
rootdir: /myapp, inifile:
collected 1 items
test_app.py .
============= 1 passed in 0.20 seconds ==============
```

Our **flask-app-testing** container was created and then **pytest** ran and found the one test we had, ran it and the container exited. We can unmount the rootfs now using **$ sudo umount ./rootfs**.

Generalising this approach

We can clean up our workflow a fair bit here by having our base image in the top-level directory, and then keeping two separate directories – one for our Flask app container and the other for our testing container such that the entire directory looks as follows:

```
$ tree -L 1
.
└─ deployment
└─ fedora-24.flask
└─ myapp
└─ testing
```

The **fedora-24.flask** directory above is what was our base root filesystem. Now, it's moved out of the deployment directory. The **deployment** and **testing** directory each have a shell script that creates the overlay filesystem making use of this base image and starts runC appropriately. The shell script in the **testing** directory is as follows:

```
$ cat testing/test_flask_app.sh
#!/bin/bash
set -e
mkdir -p rootfs rootfs.empty overlay.workdir
mount -t overlay overlay -o lowerdir=./pytest:../
fedora-24.flask,upperdir=./rootfs.empty,workdir=./
overlay.workdir ./rootfs
/usr/local/sbin/runc run flask-app-testing
umount rootfs
# Clean up
rm -r rootfs rootfs.empty overlay.workdir
```

We create the three directories – **rootfs** (where we will have the merged filesystem), **rootfs.empty** (our upper directory, which will mean the container can write to it – temporary files for example), and the **overlay.workdir** directory. Next, we create the overlay filesystem combining the **pytest** directory from earlier with our base image and then use runC to run the container. Once the tests are run, the rootfs is unmounted and the intermediate directories removed.

Another advantage to this approach is that our base image, **fedora-24.flask**, will now be immune to any accidental changes, since overlayfs mounts the lower directory as read-only. Now, if you want to create another container image for serving your web application via **uwsgi**, you could create another directory, and follow an approach similar to our testing image (Figure 5).

A peak into the OCI image specification

We wanted to build container images the hard way and we have been successful in that quest. However, we have adopted a fairly *ad-hoc* workflow, and our images don't follow any standardised image format. The Open Container Initiative's Image Format project (**https://github.com/opencontainers/image-spec**) aims to standardise how container images should be formatted. A key component of the specification is to specify what layers an image is made up of. This is similar to how we built our second image.

Currently there are no tools to build a container image from scratch, which follows the specifications. It can be done by hand but this involved a very convoluted process and so we didn't attempt to do so here. However, we can briefly see what an OCI-compatible image looks like by converting an existing container image to this format.

Skopeo (**https://github.com/projectatomic/skopeo**) can convert existing *Docker* images from *Docker* hub to OCI-compatible images. First of all, let's install it:

```
$ sudo dnf -y install gpgme-devel libassuan-devel
$ cd $GOPATH/src/github.com
$ mkdir projectatomic && cd projectatomic
$ git clone https://github.com/projectatomic/skopeo.git
$ cd skopeo && make binary-local
```

We should have a *Skopeo* binary as **./skopeo**. Let's convert the Fedora 24 Docker hub image into an OCI image:

```
$ pwd
```

```
/home/vagrant/work/linux_voice_7
$ $GOPATH/src/github.com/projectatomic/skopeo/
skopeo copy docker://fedora:24 oci:fedora-24-oci
$ tree -L 3 fedora-24-oci/
fedora-24-oci/
└── blobs
|  └── sha256
|     └──
0d891a5bdae9b701e4b4017ce0a56de05acf8bc828a5c
67e53bfe086546215fd
|     └── 11a5107645d4ecb36e75d933576f5cdb52358bef385
eac2c2d2a91af44ad4ad7
|     └── 2bf01635e2a0f7ed3800c8cb3effc5ff46adc6b9b8
6f0e80743c956371efe553
└── oci-layout
└── refs
  └── latest
3 directories, 5 files
```

The **fedora-24-oci** sub-directory has our Fedora 24 image in the OCI format. In brief, the **blobs/sha256** sub-directory has the operating system image itself as gzip-compressed data, and two other specification files describing the image. One of these specifications, the image manifest, has the layers comprising the image:

```
$ cat fedora-24-oci/blobs/sha256/0d891a5bdae9b701e4b
4017ce0a56de05acf8bc828a5c67e53bfe086546215fd | jq
".layers"
[
 {
  "mediaType": "application/vnd.oci.image.layer.tar+gz
ip",
  "digest": "sha256:2bf01635e2a0f7ed3800c8cb3effc5ff4
6adc6b9b86f0e80743c956371efe553",
  "size": 72881216
 }
]
```

Note that these blobs are all content-addressable blob objects – roughly meaning their filenames are the same as the sha256 digest of their contents. The **refs** directory is roughly analagous to the image tags that *Docker* users are familiar with and specifies the image manifest that it points to. The **oci-layout** file specifies the OCI layout version.

What can we do with this OCI image? We can convert this into a root filesystem using **oci-image-tool** and use runC to create a container from it (Figure 6). First we will install the **oci-image-tool** (**https://github.com/opencontainers/image-spec/tree/master/cmd/oci-image-tool**):

```
$ cd $GOPATH/src/opencontainers/
$ git clone https://github.com/opencontainers/image
-spec
$ cd image-spec
$ make oci-image-tool
go build ./cmd/oci-image-tool
```

Next, let's create the runtime bundle for runC:

```
$ pwd
/home/vagrant/work/linux_voice_7
$ mkdir fedora-24-runtime-bundle
```

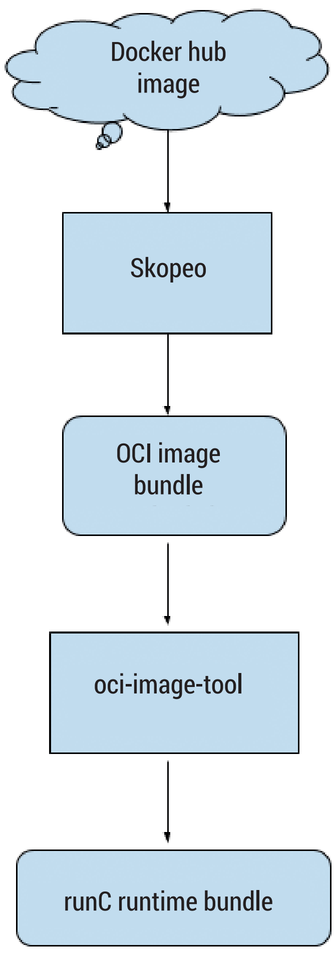


Figure 6: Converting a *Docker* hub image to an OCI-compatible image and then using with runC.

```
$ $GOPATH/src/github.com/opencontainers/image-spec/
oci-image-tool create-runtime-bundle --ref latest
fedora-24-oci/ fedora-24-runtime-bundle
$ tree -L 1 fedora-24-runtime-bundle/
fedora-24-runtime-bundle/
└── config.json
└── rootfs
```

We can go ahead and start a Fedora 24 container from the runtime bundle using runC as we did earlier.

Conclusion

We hand-crafted container images in this article and then used overlayfs to implement a rudimentary layer-based container images solution. Overlayfs is just one approach to implement such a mechanism as is exemplified by the many supported *Docker* storage drivers. We also learned about the OCI's image format standardisation efforts and used the **oci-image-tool** to assemble layers to a single container image.

It's an exciting time for software containers, and we hope this article has got you thinking about how images for software containers can be built. You can find all the code we discussed in this article at **https://github.com/amitsaha/linux_voice_7** in addition to a set of resources to explore next.

Amit Saha is the author of Doing Math with Python (No Starch Press) and a software engineer. He blogs at **https://echorand.me**, tweets @echorand and can be reached via email at **amitsaha.in@gmail.com**

IRC: BUILD YOUR OWN CUSTOM CHAT BOT

Use Python to create an IRC bot to entertain yourself – or provide useful information.

MIKE SAUNDERS

Why do this?

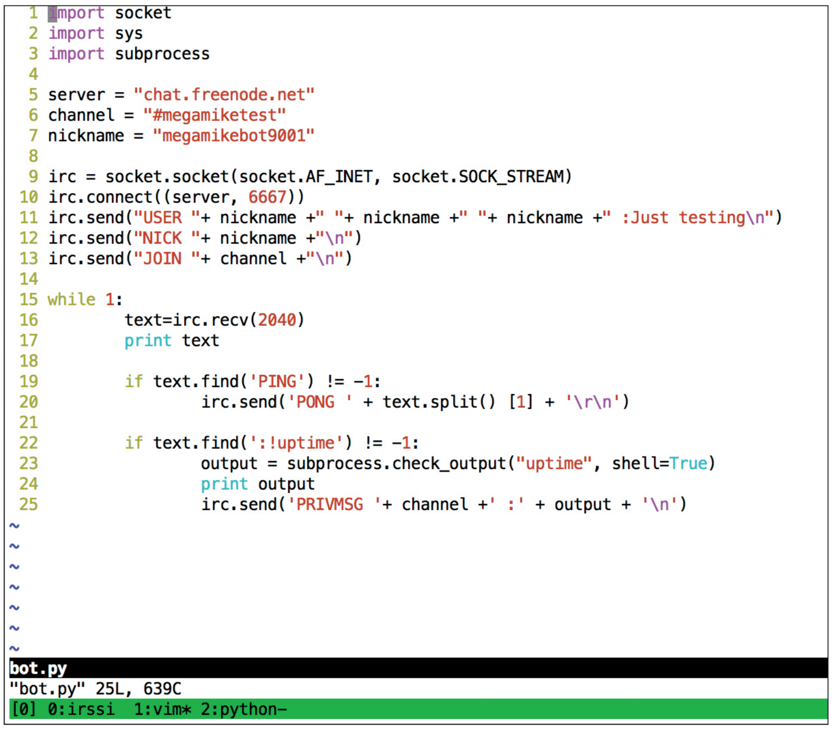
- Explore network connections in Python
- Monitor servers via IRC channels
- Get your mad scientist career started

How worried are you about artificial intelligence (AI) taking over the world? It all sounds very sci-fi, very Skynet, but some public figures with geeky backgrounds are talking about it. SpaceX and Tesla head honcho Elon Musk describes AI as "our biggest existential threat", for instance, while Steven Hawking has also said it's something we need to be wary of. We at Linux Voice don't live in fear of being enslaved by our robotic overlords in the near future, but with the rate of progress, we certainly should be careful with what we create.

Now, one example of AI is a chat bot. This is a program with which you communicate by sending text. The chat bot analyses the text for information and context, and responds accordingly. Really good chat bots are often hard to distinguish from real people – at least, when talking about very mundane topics. In this tutorial, we'll show you how to create a very simple chat bot that talks on IRC and responds to commands. You could elaborate on it to make it talk in more depth, or turn it into a practical tool to provide information from remote machines. It's up to you!

First let's recap what IRC is, in case you're not familiar with it. Internet Relay Chat is an open protocol for text-based communication across the net. Users connect to IRC servers – many of which can be linked together to form a network – and then join "channels" that cover specific topics. For instance, you could install an IRC client (such as *X-Chat*) via your package manager, connect to the **chat.freenode.net** network, and then join the #linux channel to talk about our favourite operating system.

IRC is a simple, text-based protocol, so you don't need to do a lot of grunt work to write an IRC client from scratch. All you need to do is connect on a certain network port, send text messages to it, and respond accordingly. And that's what we'll do here – using Python. There are many specialised Python chat bots that you can use and customise, offering all kinds of advanced features, but we think it's more fun and informative to start from the ground up, so you can see exactly how IRC works. So let's get started!



```
import socket
import sys
import subprocess

server = "chat.freenode.net"
channel = "#megamiketest"
nickname = "megamikebot9001"

irc = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
irc.connect((server, 6667))
irc.send("USER "+ nickname +" "+ nickname +" "+ nickname +" :Just testing\n")
irc.send("NICK "+ nickname +"\n")
irc.send("JOIN "+ channel +"\n")

while 1:
        text=irc.recv(2040)
        print text

        if text.find('PING') != -1:
                irc.send('PONG ' + text.split() [1] + '\r\n')

        if text.find(':!uptime') != -1:
                output = subprocess.check_output("uptime", shell=True)
                print output
                irc.send('PRIVMSG '+ channel +' :' + output + '\n')
~
~
~
~
~
~
~
bot.py
"bot.py" 25L, 639C
[0] 0:irssi  1:vim* 2:python-
```

Here's our code, nicely syntax highlighted in *Vim*. We've said it before, but it really is worth learning an advanced editor in depth.

Show me the code

For this bot we're going to use Python 2. Yes, it's an old version of the language, but it's the default shipped in Debian and therefore available out of the box in Raspbian on the Raspberry Pi (which is a great little machine for running bots – you can have bots online 24x7 without using much electricity). In true Linux Voice fashion, we're going to show you the code and let you try it first to see what it does, before going through it bit-by-bit.

So, open up your favourite text editor, enter the following code, and save it in a file called **bot.py**. (Or alternatively, copy the code from our website at **www.linuxvoice.com/wp-content/uploads/2016/09/bot.txt**)

```
import socket
import subprocess

server = "chat.freenode.net"
channel = "#megalvtest"
nickname = "megalvbot9001"

```

```
irc = socket.socket(socket.AF_INET, socket.SOCK_
STREAM)
irc.connect((server, 6667))
irc.send("USER "+ nickname +" "+ nickname +" "+
nickname +" :Just testing\n")
irc.send("NICK "+ nickname +"\n")
irc.send("JOIN "+ channel +"\n")

while 1:
  text = irc.recv(2040)
  print text

  if text.find('PING') != -1:
    irc.send('PONG ' + text.split() [1] + '\r\n')
  if text.find(':!uptime') != -1:
    output = subprocess.check_output("uptime",
shell=True)
    irc.send('PRIVMSG '+ channel +' :' + output + '\n')
```

So there we have it: an IRC bot in just 23 lines of code (or even fewer if you remove the blank lines). Not bad, eh? Note that if you're new to Python, the indentation added by tabs here is very important in the language. If you try to run the code and see error messages, make triple sure that your code is indented just like ours.

Once you have the code saved in **bot.py**, open your IRC client, join the **chat.freenode.net** server and then the #megalvtest channel. Unless some other Linux Voice readers happen to be around testing their bots, the channel should be empty. In a terminal window, run your bot like so:

```
python2 bot.py
```

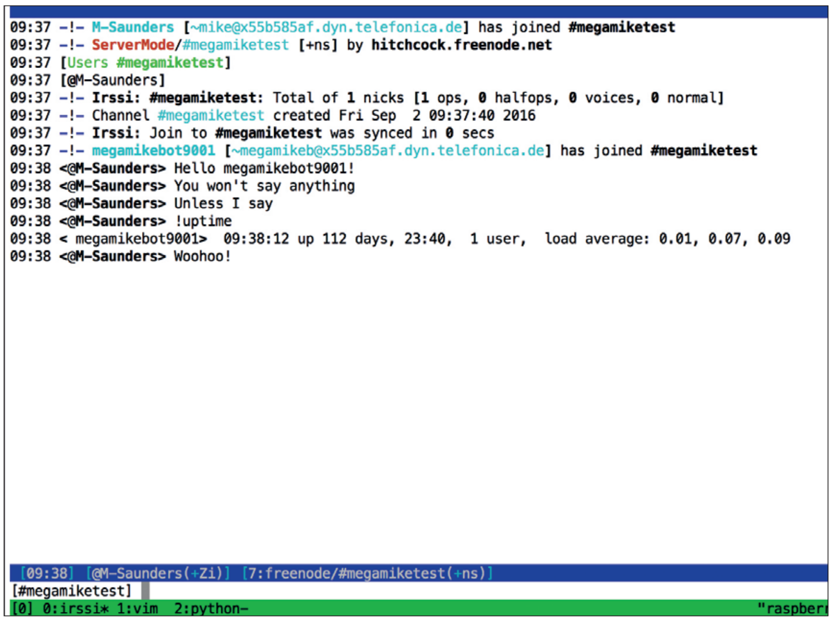
After a few seconds, the bot will establish its connection to the server and join #megalvtest. You'll notice that it doesn't say any kind of greeting when it joins the channel, nor does it respond if you talk to it directly.

"Well, not much of a bot then, is it?" you may say at this point. Hold your horses! In the IRC channel, try entering **!uptime** in your IRC client. The bot should respond with the output of the **uptime** command from the machine on which it's running. Now you see what it does! (The exclamation mark at the start is just common practice with IRC bots, to make it clear to other users in the channel that you're trying to issue a command to a bot.) To terminate the bot, use Ctrl+C in the terminal window where you're running it.

A quick guide to IRC

If you've never used IRC before, getting started is easy. Install a client like *X-Chat* (graphical) or *Irssi* (command line) from your package manager and start it. Enter **/server chat.freenode.net** to join the Freenode network – after a few seconds, a bunch of messages will appear, and then you'll be prompted for your next action. Enter **/nick MyNickName** to set your nickname – choose something appropriate, and if you get a message saying it's already in use, use a variant.

Then join a channel, which is typically prefixed by a hash mark, like so **/join #megalvtest**. You can be connected to multiple IRC servers and channels at the same time, making it a very useful tool for real-time communication, especially in open source projects. For more commands, see **www.irchelp.org/irchelp/new2irc.html**.



```
09:37 -!- M-Saunders [~mike@x55b585af.dyn.telefonica.de] has joined #megamiketest
09:37 -!- ServerMode/#megamiketest [+ns] by hitchcock.freenode.net
09:37 [Users #megamiketest]
09:37 [@M-Saunders]
09:37 -!- Irssi: #megamiketest: Total of 1 nicks [1 ops, 0 halfops, 0 voices, 0 normal]
09:37 -!- Channel #megamiketest created Fri Sep  2 09:37:40 2016
09:37 -!- Irssi: Join to #megamiketest was synced in 0 secs
09:37 -!- megamikebot9001 [~megamikeb@x55b585af.dyn.telefonica.de] has joined #megamiketest
09:38 <@M-Saunders> Hello megamikebot9001!
09:38 <@M-Saunders> You won't say anything
09:38 <@M-Saunders> Unless I say
09:38 <@M-Saunders> !uptime
09:38 < megamikebot9001>  09:38:12 up 112 days, 23:40,  1 user,  load average: 0.01, 0.07, 0.09
09:38 <@M-Saunders> Woohoo!

 [09:38] [@M-Saunders(+Zi)] [7:freenode/#megamiketest(+ns)]
[#megamiketest]
[0] 0:irssi* 1:vim  2:python-                                          "raspberr
```

Here we join the #megamiketest channel, and then our bot (megamikebot9001) does shortly after. Then we have a jolly nice chat with it.

> IRC is a simple, text-based protocol, so you don't need to do a lot of grunt work to write an IRC client from scratch

Juicy details

Now, the magic that makes this happen is all in the final three lines of this code, but the rest of it is equally important to set things up. So let's go through it in detail. The first two lines tell Python that we want to use some of its supplied modules for extra features:

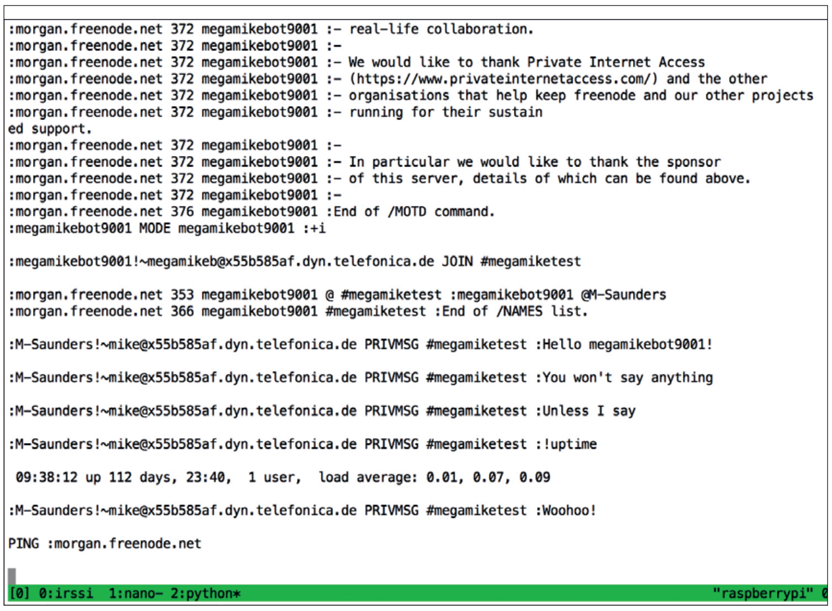
```
import socket
import subprocess
```

Python on its own can't do much, but by importing bolt-on modules into your program you can use more functionality. In this case, we import the socket (for network communication) and subprocess (to run command-line programs inside our program) modules for use in a moment.

Next up, we set up three string (text) variables:

```
server = "chat.freenode.net"
channel = "#megalvtest"
nickname = "megalvbot9001"
```

This is pretty obvious – they contain the server to which the bot will connect, the channel it will join, and the nickname it will use. If your bot doesn't run correctly, maybe another Linux Voice reader is also playing around with the bot, so try changing its nickname to avoid clashes!



```
:morgan.freenode.net 372 megamikebot9001 :- real-life collaboration.
:morgan.freenode.net 372 megamikebot9001 :-
:morgan.freenode.net 372 megamikebot9001 :- We would like to thank Private Internet Access
:morgan.freenode.net 372 megamikebot9001 :- (https://www.privateinternetaccess.com/) and the other
:morgan.freenode.net 372 megamikebot9001 :- organisations that help keep freenode and our other projects
:morgan.freenode.net 372 megamikebot9001 :- running for their sustain
ed support.
:morgan.freenode.net 372 megamikebot9001 :-
:morgan.freenode.net 372 megamikebot9001 :- In particular we would like to thank the sponsor
:morgan.freenode.net 372 megamikebot9001 :- of this server, details of which can be found above.
:morgan.freenode.net 372 megamikebot9001 :-
:morgan.freenode.net 376 megamikebot9001 :End of /MOTD command.
:megamikebot9001 MODE megamikebot9001 :+i

:megamikebot9001!~megamikeb@x55b585af.dyn.telefonica.de JOIN #megamiketest

:morgan.freenode.net 353 megamikebot9001 @ #megamiketest :megamikebot9001 @M-Saunders
:morgan.freenode.net 366 megamikebot9001 #megamiketest :End of /NAMES list.

:M-Saunders!~mike@x55b585af.dyn.telefonica.de PRIVMSG #megamiketest :Hello megamikebot9001!

:M-Saunders!~mike@x55b585af.dyn.telefonica.de PRIVMSG #megamiketest :You won't say anything

:M-Saunders!~mike@x55b585af.dyn.telefonica.de PRIVMSG #megamiketest :Unless I say

:M-Saunders!~mike@x55b585af.dyn.telefonica.de PRIVMSG #megamiketest :!uptime

 09:38:12 up 112 days, 23:40,  1 user,  load average: 0.01, 0.07, 0.09

:M-Saunders!~mike@x55b585af.dyn.telefonica.de PRIVMSG #megamiketest :Woohoo!

PING :morgan.freenode.net

[0] 0:irssi  1:nano- 2:python*                                              "raspberrypi" 
```

Meanwhile, the bot spits out everything it receives, such as messages from the IRC server and individual users.

Now the real fun begins. We establish a connection to the IRC server, on port 6667:

```
irc = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
irc.connect((server, 6667))
```

Note that we're not using any fancy IRC modules or libraries to do this; we're simply connecting to the server on a specified port, so we can send textual data to it. That's the beauty and simplicity of IRC. Once

We're not using any fancy IRC modules to do this; we're simply connecting to the server on a specified port

the connection has been established, we tell the IRC server who we are by sending it a few messages:

```
irc.send("USER "+ nickname +" "+ nickname +" "+ nickname +" :Just testing\n")
irc.send("NICK "+ nickname +"\n")
irc.send("JOIN "+ channel +"\n")
```

The **"USER"** information is made up of multiple parts – that doesn't interest us here, so we just put the bot's nickname for everything. Then we tell the IRC server what nickname we want to use, followed by the channel we want to join. Note the use of **"\n"** newline characters at the end, to make sure the messages get through to the server.

And then we're in! Our bot is connected to the IRC server, has identified itself and joined a channel. But there's one thing it needs to do to stay alive. You see, the IRC server will periodically send a **PING** message to the bot to see if it's still running – and if the bot doesn't respond with a **PONG** message, the IRC server will assume the connection has dropped or the bot has been terminated, and remove the bot from the channel accordingly. So we set up an infinite loop:

```
while 1:
```

And then we wait to receive a bunch of data from the IRC server. When the data arrives, we also print it out to the terminal window, for debugging purposes:

```
        text = irc.recv(2040)
        print text
```

Now we need to check if the data sent by the IRC server was the aforementioned **PING** request, and if so, we send **PONG** back:

```
  if text.find('PING') != -1:
    irc.send('PONG ' + text.split() [1] + '\r\n')
```

Note the use of **text.split()** to pluck out the name of the IRC server and respond directly to it. (Even though the bot has joined the **chat.freenode.net** network, it may have been redirected to another server on Freenode like **niven.freenode.net**. Don't worry about that now – it's just so you know what's going on.

Then we have the final three lines of code, which check to see if someone has said **!uptime** in the channel, and then respond accordingly:

```
  if text.find(':!uptime') != -1:
    output = subprocess.check_output("uptime", shell=True)
    irc.send('PRIVMSG '+ channel +' :' + output + '\n')
```

The **subprocess.check_output()** routine lets us run any program on the Linux box that's hosting the bot, and store its output in a string. We then send that output to the channel by prefacing it with **PRIVMSG**, the channel name, and a colon – plus a newline character at the end.

So there we have it! If you have multiple servers to monitor, you could create an IRC channel, run bots on each server, and customise them to respond to different commands so you can see their status (RAM usage, free disk space etc) without having to SSH in to each one.

But! We have one last trick. At the moment, anyone can issue a command to the bot and it will execute it – the bot doesn't check to see who said the command. This is obviously a potential security issue, so let's change the final three lines to only respond when a certain person issues the command:

```
 if text.find(':!uptime') != -1:
     if text.split()[0].find(':MyNickName!') != -1:
         output = subprocess.check_output("uptime", shell=True)
         irc.send('PRIVMSG '+ channel +' :' + output + '\n')
```

Here, we've added a line (the second one) that searches for **MyNickName** (change to yours accordingly), surrounded by a colon and exclamation mark, as that's how the nickname of a chatter is supplied in messages from the server. So in this case, the bot will only respond if **MyNickName** says **!uptime**. And to be extra secure, register your nickname with the IRC services, add a password, and nobody else will be able to pretend to be you and control your bot!

Mike Saunders is not a bot, honest. KILL ALL HUMANS... *cough*, sorry, ignore that.

LISTEN TO THE PODCAST

LINUXVOICE

WWW.LINUXVOICE.COM

FREE SOFTWARE | FREE SPEECH

WRITE A WEB APPLICATION IN CRYSTAL

Try a new-ish programming language that reads like Ruby and is as fast as C.

AMIT SAHA

Why do this?

- Because life's too short to specify your variable types

The Crystal programming language tools (compiler and others) are available via a single command: **crystal**, with the latest release at the time of writing being 0.19. We can either download the pre-built binaries or install using the distro's package manager. We will follow the latter approach even though it involves adding a third party-repository. The following will install Crystal on Fedora 24:

```
$ curl https://dist.crystal-lang.org/rpm/setup.sh | sudo bash
$ sudo dnf -y install crystal
```

Note that if you don't have *GCC* installed, you will have to install it before you can compile and run Crystal programs. For a complete list of installation options, see the installation documentation at **https://crystal-lang.org/docs/installation/index.html**. Once the installation steps are completed, open your favorite terminal emulator and type **crystal version** and it should print a message as below:

```
$ crystal version
Crystal 0.19.0 [dcfb2b6] (2016-09-02)
```

Before we start to type in entire programs, let's take a quick look at the **eval** command. Type in **crystal eval**, type in the code snippet below and press Ctrl+D to send the EOF signal:

```
$ crystal eval
msg = "Hello World"
puts msg
puts(1+2)
```

The line **msg = "Hello World"** creates a variable, **msg** and assigns it the string **"Hello World"**. Note how we didn't need to declare the variable type here. Next, we print this string, using **puts msg**. **puts** is a Crystal method used to print to the standard output. In the next line, we again call the **puts** method with **1+2** as the argument. Once you press Ctrl+D above to signal EOF, you should see the following output:

```
Hello World
3
```

The **eval** command is useful for quickly verifying snippets of code, but it's worth noting that Crystal is a compiled language, so even when you are using **eval**, it is compiling your code in the background and then executing it

Let's now write our first Crystal program save it into the file **hello.cr**:

```
# hello.cr
puts "Hello from Crystal. Please enter a line of text: "
s = gets()
puts("You entered: #{s}")
puts("Type of input: #{typeof(s)}")
if s
  length = s.size
end
puts("Type of length: #{typeof(length)}")
puts("Number of characters you entered: #{length}")
```

Let's now compile the program and execute it:

```
$ crystal build hello.cr
$ ./hello
Hello from Crystal. Please enter a line of text:
Hi Crystal
You entered: Hi Crystal
Type of input: (String | Nil)
Type of length: (Int32 | Nil)
Number of characters you entered: 11
```

The first line in the above program is a comment, beginning with the **#** character. Crystal doesn't yet support multi-line comments.

The next two lines displays an input prompt and waits for the user to enter a line of input. This input line is stored in the variable, **s**. We have to use the **if** condition to check whether the variable **s** has a non-nil value, since the size method is only defined for the String type.

Next, we use the **puts()** function and use string interpolation to embed the input message in the output string. That is, **You entered: #{s}** creates a new string as **You Entered: Hi Crystal**. Next, we use the **typeof** expression to print the type of the input string, **s**. We can see from the output that the type is **(String | Nil)**. This indicates a union type in Crystal and is read as **String or Nil**. The Nil data type can have one value only, **nil**, and is used to indicate the absence of a value.

Next, we use an **if** condition to check if **s** has a non-nil value, and if so we find the number of characters in the string using the **size** method, which we finally print. Note that the type of the length variable is another union type **(Int32 | Nil)**.

The union type is a compilation-time property. At runtime, a variable or an expression of course has a single type. The runtime type can be found by invoking the class method. For example:

```
s = gets()
puts("Compile time type: #{typeof(s)}")
puts("Runtime type: #{s.class}")
```

When a line of input is entered, we will see that the runtime type is reported as **String**:

```
A line of input
Compile time type: (String | Nil)
Runtime type: String
```

The compile time type still remains as **String|Nil**.

Methods

We can define a method (or function) in Crystal using **def <method-name>** specifying the parameters in parentheses. Our next program shows how we can define methods including overloaded methods:

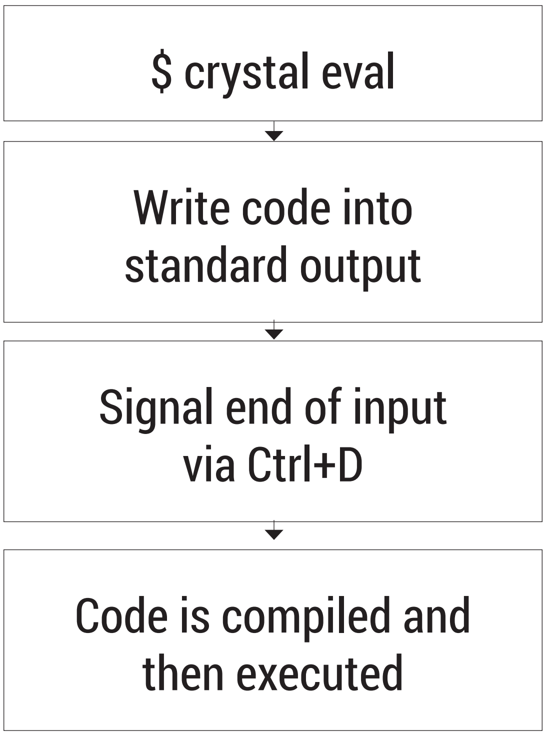
```
# Example of defining method including overloaded
methods
# method_demo.cr
def power(number)
 number**1
end
def power(number, p)
 number ** p
end
print("Enter a number to square: ")
number = gets()
print("Enter the power to raise it to: ")
p = gets()
if number
 if p
  p = p.to_i
  result = power(number.to_f, p)
 else
  result = power(number.to_f)
 end
 puts("\nResult: #{result}")
end
```

We define two methods **power()** of the same name. One accepts a single parameter (a number), and the other accepts accepts two numbers as parameters, the second one being the power to which to raise the first number.

Then, we ask the user to input a number and the power that we want to raise it to. If the second input is not entered, we call the **power** method with just the first number and hence the first **power** method gets called. When the power is also specified, the second power method gets called. This is an example of overloaded methods in Crystal.

Let's compile and run the above program:

```
$ crystal build method_demo.cr
$ ./method_demo
Enter a number to square: 2
Enter the power to raise it to:
Result: 2.0
$ ./method_demo
Enter a number to square: 3.2
Enter the power to raise it to: 2
Result: 10.24
```


$ crystal eval
Write code into standard output
Signal end of input via Ctrl+D
Code is compiled and then executed

Functionality of **crystal eval**.

In the above program, note how we converted the inputs to a floating point number and integer using the **to_f** and **to_i** methods respectively.

Let's see what happens if we give the second input (power) as a non-integer:

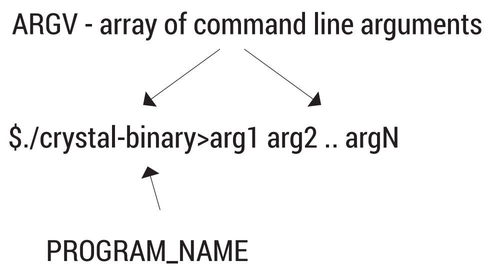
```
./method_demo
Enter a number to square: 3.4
Enter the power to raise it to: 1.2
invalid Int32: 1.2
 (ArgumentError)
<traceback>
```

The **to_i** method couldn't convert the string 1.2 to an integer and prints us a traceback, which is not how we want to report an error. We will now see an example of Crystal's exception handling and report a friendly error message and exit. The **begin..rescue..else..end** block is used to handle exceptions in Crystal. The **begin** block has code that may raise an exception, which we want to rescue (or catch) in the **rescue** block. We may want some code to be executed only when there is no exception and we put that code in the **else** block. Below, I have shown only the part that has changed from our previous program:

```
# For the entire program, see method_demo_exception_
handling.cr
if number
 begin
  number = number.to_f
 rescue ex : ArgumentError
  puts("Error converting number to floating point
number: #{ex.message}")
  exit(1)
 else
```

PRO TIP

In Crystal, we can pass arguments to methods with or without parentheses. For example, puts "Hello" and puts("Hello") are both valid.


ARGV - array of command line arguments
$./crystal-binary>arg1 arg2 .. argN
PROGRAM_NAME

ARGV and **PROGRAM_NAME** for a Crystal binary program

```
  if p
   begin
    p = p.to_i
   rescue ex : ArgumentError
    puts("Error converting power to integer: #{ex.
message}")
    exit(1)
   else
    result = power(number, p)
   end
  else
   result = power(number)
  end
 puts("\nResult: #{result}")
 end
end
```

PRO TIP

Crystal programs end with the ".cr" extension. For the purpose of this article, we will put all our programs in a sub-directory "linux_voice_6" under the $HOME directory.

The first **begin..rescue..end** block handles the conversion of our first input to a floating point variable. We specify that we want to rescue the **ArgumentError** exception. If we get one, we print a message and exit out of the program with an exit status of 1. We do the same for the second input. Now, if we build and run this program, we will see the following upon bad input:

```
$ ./method_demo_exception_handling
Enter a number to square: 4
Enter the power to raise it to: 2.1
Error converting power to integer: invalid Int32: 2.1
$ ./method_demo_exception_handling
Enter a number to square: 1a
Enter the power to raise it to: 2
Error converting number to floating point number:
Invalid Float64: 1a
```

Arrays and command line arguments

Arrays can be used to store items of a single type, such as integers or multiple types such as integers and strings. We can create an array and initialise it with the members as follows:

```
mixed_arr = [1, 1.2, "Hello", nil]
puts typeof(mixed_arr)
```

The type of **mixed_array** will be printed as: **Array(Float64 | Int32 | String | Nil)** indicating that the type is a union type.

When we want to create an array without initialising it at the same time, we have to declare the type as follows:

```
mixed_arr = [] of Int32|String|Nil
mixed_arr << 1
mixed_arr << "Hii!"
puts mixed_arr[0]
puts mixed_arr[1]
puts "Size of mixed_arr: #{mixed_arr.size}"
```

In the above code, we declare that **mixed_arr** is an array of union type **Int32|String|Nil**, then we append items to the array using the **<<** operator. We can then refer to the fifth element using **mixed_arr[4]** (the first element has the index 0) and the **size** method can be used to find the number of elements in the array. The output of the above code is:

```
1
Hii!
Size of mixed_arr: 2
```

A common operation with an array is to iterate over the elements and do something with each element. We would do so using the following construct:

```
mixed_arr.each{|item| puts item}
```

The above code will call the **puts** method with **item** where **item** is each element in the array.

A common task when writing command line applications is to be able to do some basic command line argument handling. In Crystal the global array **ARGV** stores the command line arguments passed to a program when executing it. Unlike, most programming languages out there however, **ARGV[0]** – that is the first element of this array is not the program name, but the first command line argument. Another global, **PROGRAM_NAME** stores the program name. Let's see a quick demo of these globals:

```
# cmdline_args_demo.cr
puts "PROGRAM_NAME: #{PROGRAM_NAME}"
puts "ARGV: #{ARGV} Size: #{ARGV.size}"
```

If we build and run this program now passing command line arguments to it, we will see:

```
$ crystal build cmdline_args_demo.cr
$ ./cmdline_args_demo hello world
PROGRAM_NAME: ./cmdline_args_demo
ARGV: ["hello", "world"] Size: 2
```

In the GitHub repository for this article, you can see a more useful example of using the command line arguments in the program **basic_ls.cr**.

Concurrent execution with Fibers

Fibers are lightweight threads (more commonly referred to as coroutines) and provide the mechanism via which we can have concurrent threads of execution in a Crystal program. Any Crystal program runs as a fiber and its execution is managed by Crystal's own runtime scheduler. To create a fiber in our program and hence be able to have two concurrent units of execution, we use **spawn**. The next listing shows a simple example:

```
spawn do
  puts("I am in second fiber")
end
puts("In main fiber")
```

```
sleep 1.second
```

When we create a fiber using **spawn**, it doesn't start running immediately. Our main fiber continues executing; only when we go to sleep for a second does our secondary fiber get a chance. If we run the code above, we will get:

```
./fiber_demo
In main fiber
I am in second fiber
```

WebSocket powered web application with Kemal

Kemal (**http://kemalcr.com**) is a micro web framework for Crystal that comes with inbuilt support for websockets. We will use it to create our websocket-based chat application.

We will first create a Crystal project skeleton using **crystal init app <name>**:

```
$ crystal init app broadcaster
   create  broadcaster/.gitignore
   create  broadcaster/LICENSE
   create  broadcaster/README.md
   create  broadcaster/.travis.yml
   create  broadcaster/shard.yml
   create  broadcaster/src/broadcaster.cr
   create  broadcaster/src/broadcaster/version.cr
   create  broadcaster/spec/spec_helper.cr
   create  broadcaster/spec/broadcaster_spec.cr
Initialized empty Git repository in /home/vagrant/work/
linux_voice_6/temp/broadcaster/.git/
```

We have given the name of our application as **broadcaster**. Crystal generates a number of files for us in a new subdirectory **broadcaster** and also makes the directory a *Git* repository.

The **.travis.yml** file is created for us so that we can easily have the Travis software testing service build our projects (**https://docs.travis-ci.com/user/languages/crystal**).

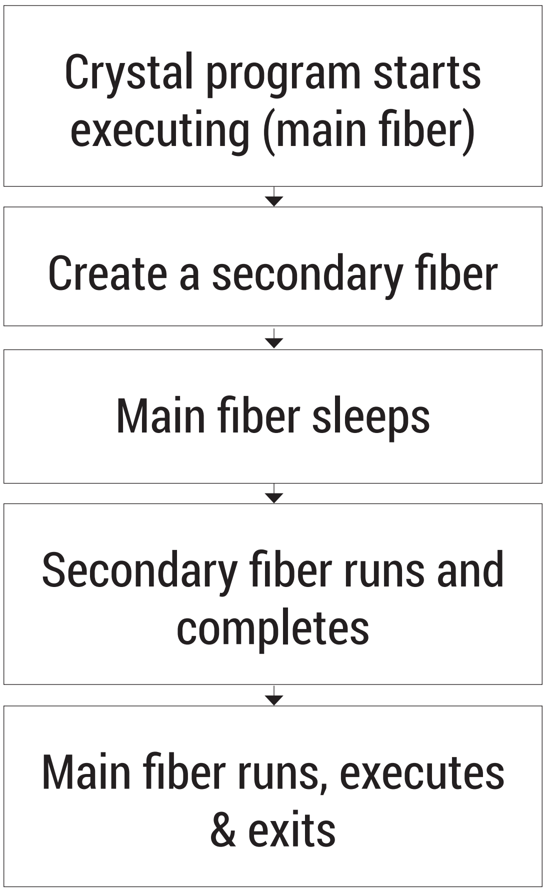
The **shard.yml** file is where we will add our third-party dependency as follows:

```
name: broadcaster
version: 0.1.0
dependencies:
 kemal:
  github: sdogruyol/kemal
  version: 0.15.1
authors:
 - Amit Saha <amitsaha.in@gmail.com>
license: MIT
```

We specify that our application depends on the "kemal" which we want fetched from github and we want the specific version 0.15.1 (the most recently released) version.

Next, we will save the following code for our web application in the file broadcaster/src/broadcaster.cr:

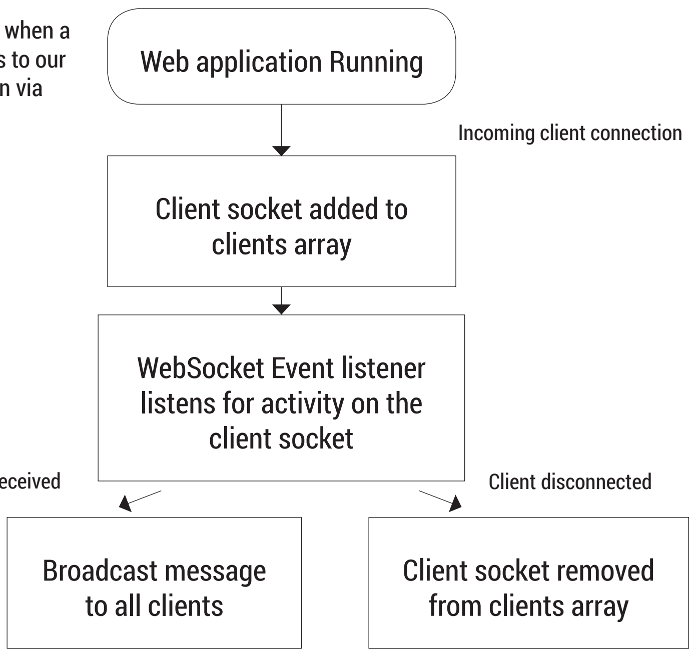
```
require "./broadcaster/*"
require "kemal"
# Array, clients to store the incoming
# connections
clients = [] of HTTP::WebSocket
```


Crystal program starts executing (main fiber)
Create a secondary fiber
Main fiber sleeps
Secondary fiber runs and completes
Main fiber runs, executes & exits
Program with one or more fibers spawned

```
get "/" do
 "Hello client, connect to /socket for websocket chat!"
end
ws "/socket" do |socket|
 # Add to the clients array
 clients << socket
 # On recieving a message from a client, echo it
 # back to all other clients
 socket.on_message do |message|
  clients.each {|socket| socket.send message}
 end
 # If a client disconnects, delete it
 # from the array
 socket.on_close do
  clients.delete socket
 end
end
Kemal.run
```

Note how I have removed the **module** definition from the generated code but have kept the first **require**, so that you if we decide to extend this application, we can add our code to a file in the **broadcaster** directory and it will be automatically available to us for use here.

require is used to bring in type definitions, methods or essentially bring in code from other files. Here, when we specify **require "kemal"**, Crystal looks for a file named **kemal.cr** in the standard library, and then in the **libs/** sub-directory relative to the current working directory. If it isn't found in any of these locations,


Web application Running
Incoming client connection
Client socket added to clients array
WebSocket Event listener listens for activity on the client socket
Message received
Client disconnected
Broadcast message to all clients
Client socket removed from clients array

Flow of events when a client connects to our web application via web socket.

crystal looks for a directory named **kemal** in the present working directory that contains a file **kemal.cr** directly underneath it. If this isn't found, a compile time error is reported.

At the top of the file, we have an array, clients of type WebSocket defined in the HTTP module. When a new client connects to our web socket, we will store it in this array and delete it upon disconnection. Next, we set up a handler for any **GET** request to the root of the application, **"/"**. Here, we just return a message.

Next, we set up our WebSocket endpoint as **ws "/socket"**. We first add the incoming client connection socket to the **clients** array. Then we register a handler for the **on_message** event. When we get a message from the client, we echo back to all the clients, hence the name **broadcaster**. We use the **send** method to send this message.

Next, we register the **on_close** handler to handle client disconnections. When a client disconnects, we want to remove its socket from the **clients** array, so that we don't try to send a message to it. We do so using the **delete** method. Finally, we have **Kemal.run**, which starts our web application.

PRO TIP

The standard library of Crystal uses overloaded methods extensively. One example is the gets() method in the io.cr file.

Before we can run our application, we will first have to install our dependency – **kemal**(you will need to install the **openssel-devel** package):

```
$ shards  install
Updating https://github.com/sdogruyol/kemal.git
Updating https://github.com/luislavena/radix.git
Updating https://github.com/jeromegn/kilt.git
Installing kemal (0.15.1)
Installing radix (0.3.1)
Installing kilt (0.3.3)
```

We are now all set to start our web application. From our project directory, let's build our application and run it:

```
$ crystal build src/broadcaster.cr
$ ./broadcaster
[development] Kemal is ready to lead at
http://0.0.0.0:3000
```

From another terminal, we can use **curl** to send a request to the root of our application:

```
$ curl 127.0.0.1:3000
Hello client, connect to /socket for websocket
chat!vagrant@localhost
```

Now that our web application is ready, we will now write the websocket client to talk to it. For our client, we will not have to use any third-party library, as Crystal comes with inbuilt support for it in the **http** module. The listing below presents the client:

```
# ws_client.cr
require "http"
if ARGV.size == 1
  server = ARGV[0]
else
  server = "ws://localhost:3000/socket"
end
ws = HTTP::WebSocket.new(URI.parse(server))
# This fiber will run concurrently waiting for a line of
input
# and then writing to the socket
spawn do
  while true
    msg = gets()
    if msg
      ws.send(msg)
    end
  end
end
# Display any message we get from the server
ws.on_message do |message|
  puts "Got: #{message}"
end
# Start the ws loop
ws.run
```

If we build and run the client with our server still running, we can interact with our server via websockets:

```
$ ./ws_client
Hello broadcaster
Got: Hello broadcaster
```

We will leave this client running and start another client in a different window:

```
$ ./ws_client
Hello, I am client 2
Got: Hello, I am client 2
```

If we go back to our first client window, we'll see this:

```
$ ./ws_client
Hello broadcaster
Got: Hello broadcaster
Got: Hello, I am client 2
```

It would be a good exercise to try our client with any WebSocket server and make any changes to it to make it into a generic command line client for interaction with WebSockets.

Next, we will add tests to our web application so that we don't have to execute a separate command line client by hand to make sure that our web application is working as we expect it to. With a fast-developing language like Crystal, this is even more useful. When **crystal init** generated the app skeleton for us earlier, it also created a **spec** directory where it created two files for us: **spec/broadcaster_spec.cr** and **spec/spec_helper.cr**. We will define our tests in **spec/broadcaster_spec.cr** and put any helper code including other files in **spec/spec_helper.cr**. Let's run **crystal spec** now from the application's root directory (ie the **broadcaster** directory) and see what happens (Note that you have to install the **libxml2** development package before we can run this successfully – on Fedora, **sudo dnf -y install libxml2-devel**):

```
$ crystal spec
[development] Kemal is ready to lead at
http://0.0.0.0:3000
^CKemal is going to take a rest!
Finished in 12.55 seconds
0 examples, 0 failures, 0 errors, 0 pending
```

Tests for your web application

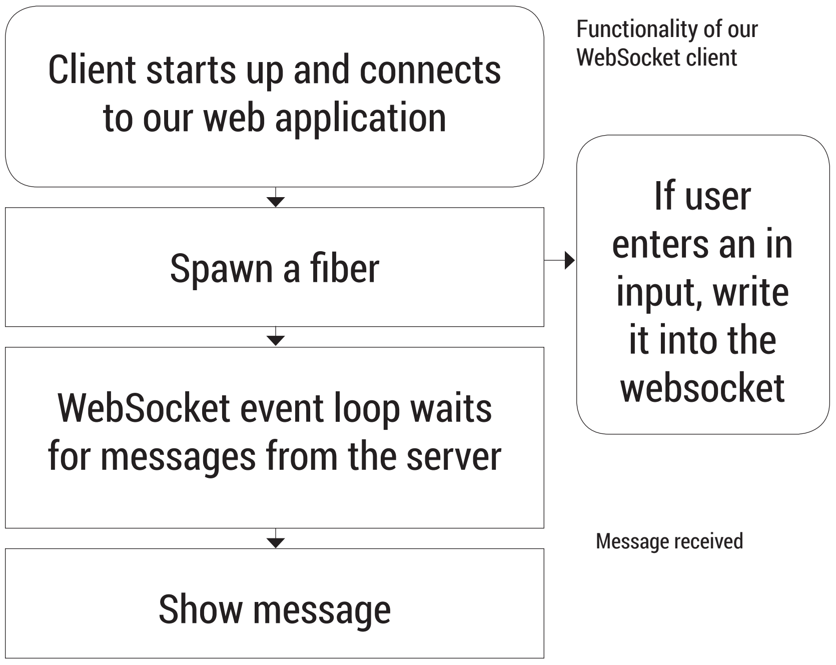
Our web application was started because the spec code calls our application, and nothing useful happened. I stopped the application using Ctrl+C. Now, let's write our tests. We will use another third-party library, **spec-kemal** (**https://github.com/sdogruyol/spec-kemal**), which provides helpers for testing a Kemal web application. We will modify the **spec/spec_helper.cr** file as follows:

```
$ cat spec/spec_helper.cr
require "spec-kemal"
require "../src/broadcaster"
```

We will modify the "spec/broadcaster_spec.cr" to be as follows:

```
require "./spec_helper"
require "http"
describe Broadcaster do
 # Start our kemal web application
 start
 it "can call /" do
  get "/"
  response.body.should contain "Hello client"
 end
 it "can talk to websocket /socket" do
  server = "#{APP_URL}/socket"
  ws = HTTP::WebSocket.new(URI.parse(server))
  ws.send("Hello from client")
end
 # Stop our web application
 stop
end
```

The first test checks that sending a request to **/** returns a friendly message and the second test just sends a message to the web socket. The **APP_URL** global variable is exposed by **spec-kemal** to be that of the currently running server, so we use that. As of


Client starts up and connects to our web application
Spawn a fiber
WebSocket event loop waits for messages from the server
Show message
If user enters an in input, write it into the websocket
Message received
Functionality of our WebSocket client

this writing, **spec-kemal** doesn't have any helper for testing websocket-based applications and hence, we have copied over a part of our command line client code we wrote earlier to test connecting to the websocket and sending a message. Note also that we don't check the reply from the server for simplicity.

Finally, we will need to add the dependency to our **shard.yml** file as:

```
spec-kemal:
 github: sdogruyol/spec-kemal
 branch: master
```

Note that here we specify the branch that we want the code from, rather than a version. Run **shards install** again and then we run the spec tests:

```
$ crystal spec --verbose
Broadcaster
 can call /
 can talk to websocket /socket
Finished in 2.69 milliseconds
2 examples, 0 failures, 0 errors, 0 pending
```

The **--verbose** flag tells the test scenarios being run and we see that it succesfully ran both the tests for us without any failures.

Conclusion

We started off with the absolute basics of the Crystal programming language and built a WebSocket-powered web application and a command line client to talk to it by the end of this. Along the way, we learned about using external packages and how to write tests for our Crystal application as well. You can find all the code listings for this article at **https://github.com/amitsaha/linux_voice_6**.

Amit Saha is the author of *Doing Math with Python* (No Starch Press) and a software engineer. He blogs at **https://echorand.me**, tweets @echorand and can be reached via email at amitsaha.in@gmail.com

Valentine Sinitsyn develops high-loaded services and teaches students completely unrelated subjects. He also has a KDE developer account that he's never really used.

CORE TECHNOLOGY

Prise the back off Linux and find out what really makes it tick

Job scheduling

We all live in a four-dimensional world. One of these dimensions is time, that is, a sequence of recurring events that we can count and measure. Many things in our lives are periodic as well: we wake up and go to bed, take our meds and meals, and so on. No wonder we designed computers to operate in almost the same way.

Periodic events lie at the very heart of CPU operation, but today we'll be concerned with a higher-level picture. If you are a system administrator, you may want to check for updates once per week, and rotate logs daily. On the other hand, jobs like planned maintenance usually occur at specific times. Linux comes with all sorts of tools to help you scheduling that, and they will be our focus today.

Crontabs explained

Crafting your very own periodic job in Linux is really a piece of cake:

```
while true; do
  # yes, do something
  sleep $some_time
done
```

Run this script in the background with **nohup** to make it last longer than your interactive session, and you are set. Of course, reinventing the wheel each time you want a recurring task is not the way things work in Linux. There are several daemons for this role. Despite the variations, often we call them just *cron*. The name originates from "chronous", the Greek word for time. Rumours are that *cron* first appeared in Unix V6, co-authored by Dennis Ritchie and Ken Thompson back in the mid-seventies.

In a nutshell, the *cron* daemon is just a slightly more elaborate version of the above script. It wakes up once per minute and checks config files to see if it has anything to do. If yes, *cron* executes the respective commands. Otherwise, it goes back to sleep until the next minute.

The main config file for *cron* is **/etc/crontab**. There are also per-user crontabs, as we'll see shortly. Each line in the crontab is either an environmental setting or a *cron* command. Settings look similar to shell variables (LV027) except no expansions are made to the values. The values themselves can be single- or double-quoted, and **~** works as expected, since the shell, not *cron*, handles this symbol.

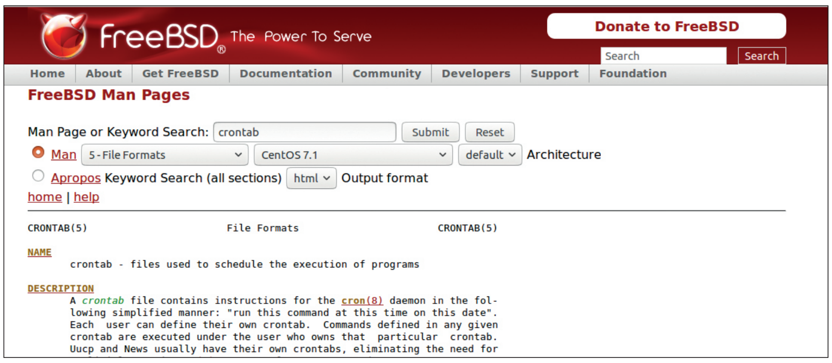
Cron defines several environment variables automatically. This includes **SHELL** (defaults to **/bin/sh**), **PATH `/usr/bin:/bin**), **HOME** and **LOGNAME**. The latter is the name of the user for which *cron* executes the command. Note that the default **PATH** setting is often different from what you get used to in the shell:

```
$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/
bin:/usr/games:/usr/local/games
```

This is a common source of errors when you first try the command in an interactive session, then copy and paste it into the crontab. Either use full paths or set **PATH** appropriately.

Another predefined setting is **MAILTO**. By default, *cron* captures commands outputs and mails them to the crontab owner. You can set **MAILTO** to a comma-separated list of addresses (including local usernames) to receive this mail instead. In real-world deployments, *cron* jobs often execute under a dedicated non-human user account, or just root. Either case, there is no real person behind the owner's mailbox, so **MAILTO** is used to redirect *cron* messages to some mailing list instead.

Cron commands contain several blank-separated fields (see the figure). They set the schedule and the content (that is, shell commands) for the job to execute. In the system crontab we're discussing now,



FreeBSD Man Pages

Man Page or Keyword Search: crontab Submit Reset
[✓ Man] 5 - File Formats CentOS 7.1 default Architecture
Apropos Keyword Search (all sections) html Output format
home | help

```
CRONTAB(5)                 File Formats                  CRONTAB(5)

NAME
       crontab - files used to schedule the execution of programs

DESCRIPTION
       A crontab file contains instructions for the cron(8) daemon in the fol-
       lowing simplified manner: "run this command at this time on this date".
       Each  user can define their own crontab.  Commands defined in any given
       crontab are executed under the user who owns that  particular  crontab.
       Uucp and News usually have their own crontabs, eliminating the need for
```

The FreeBSD project archive all manpages for most versions of BSD and Linux on their website, so head there to find out how to use any version of cron there is usually a field in between saying the user name to run the command as.

The first five fields of a *cron* command set the schedule. These fields are (in order) the minute, hour, the day of a month, month and the day of a week. All are expressed as integers; 1 means January and 0 or 7 is Sunday. You can also use abbreviated English month and day names ("Jan", "Wed" etc), which are case-insensitive. For integer values, ranges (1–5) are supported and treated as inclusive. A slash after the range defines the step: 0–6/3 expands to 0, 3, and 6. Individual values come comma-separated (1,2,3), and you may use an asterisk (*) as a wildcard matching any value. Period names, such as `@weekly`, are also recognised; see **crontab(5)** for details.

The command itself is an ordinary shell expression, with a few things to note. First, it should be valid for whatever interpreter **SHELL** points to. That is, you won't be able to use *Bash* constructs unless you change the defaults. Another common pattern is to silence the standard output with **>/dev/null**. This way, *cron* will mail you only if there is anything in **stderr**, which presumably means that things went wrong. A percent sign **%** is translated to a newline unless escaped (**\%**). Everything up to the first **%** is a command, and everything after that is fed to its standard input.

Flexible schedules

Now, consider this example:

```
MAILTO=cron-messages@mydomain.tld
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/
usr/bin

# Sample cron schedules
*/5 * * * * root logger "I'm a frequent job"
0 1 * * * root logger "I'm a midnight wanderer"
*/30 8-20 * * * root logger "I work when you do"
* * * * * root logger "You can't do without me"
```

Crontabs may contain blank lines and comments. Be careful not to append comments to *cron* commands, however, as they will be sent to the shell.

The first *cron* command runs once every 5 minutes. The second runs once per day, at 1.00am. The third runs twice per business hour (8am–8pm). The final command executes at each *cron* iteration, or roughly once per minute. *Cron* considers time relative to the server's time zone, and there is no way to redefine it per crontab.

You may want to add these lines to your system's crontab and see how this really works. The **logger** command (LV025) puts a message in the system logs. Each log entry includes a timestamp, so you can check that the schedule works as intended.

In fact, there is little need to touch **/etc/crontab** these days. Most packages install their own crontabs under **/etc/cron.d**, and *cron* stitches them together. As creating and removing files is much simpler than editing text configs, this makes package maintenance easier. This is what you'll find in this directory on a typical Ubuntu system:

```
$ ls /etc/cron.d
anacron  php5
```

Moreover, most (if not all) distributions configure *cron* to execute scripts in **/etc/cron.{hourly,daily,weekly,monthly}** every hour, day, week or month, respectively:

```
17 * * * * root cd / && run-parts --report /etc/cron.hourly
25 6 * * * root test -x /usr/sbin/anacron || ( cd / &&
run-parts --report /etc/cron.daily )
...
```

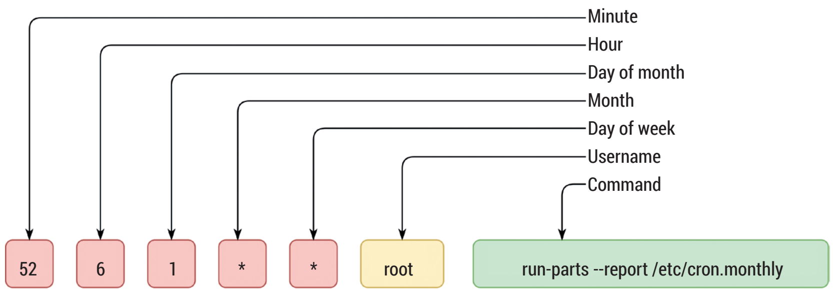
The excerpt above comes from an Ubuntu 14.04 box. Note that hourly jobs run at the 17th minute, and daily ones execute at 6.25am. You may think the 15th minute or 6.30am look prettier; that's true, and that's the reason not to use these times in your schedules. Many people may decide to run their jobs at the noon or at the midnight, or every quarter of an hour. So a bunch of scripts will start at once, drawing unwelcome peaks in the system load graphs. "Randomizing" the schedule helps to keep system load more uniform.

The *run-parts* tool is employed to run all executable scripts in a directory the proper way. We'll cover anacron shortly. To sum up: for typical schedules, just wrap your job as an executable script under **/etc/cron.***. For custom schedules, put crontab in **/etc/cron.d**. Either way, *cron* detects your changes automatically, and there is no need for explicit reload.

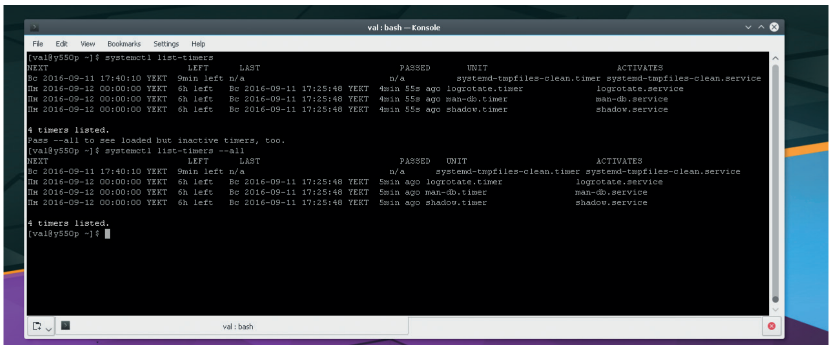
Your crontab, your way

So far we dealt with system-level crontabs. They are good for jobs that affect the host as a whole, like rotating log files or fetching package updates. Ordinary users (including **root**) may also want to have their own jobs. Imagine you run a website on shared hosting and need to aggregate some stats. It would be a breach to grant every user in a system an access to **/etc/{crontab,cron.d}**, so Linux has another mechanism – per-user crontabs.

These crontabs live in **/var/spool/cron/crontabs**, and they follow the same syntax, except that *cron* commands have no username field, for obvious reasons. You don't edit these files directly. Instead, you call the crontab tool, which runs $EDITOR of your choice (that's *Vim*, isn't it?), validates the crontab, and stores the file in the proper place. crontab facilitates editing (**-e**), removing (**-r**) and listing (**-l**) crontab files. You may also instruct it to operate on someone else's crontab with **-u**. It's a good idea to use this switch



The *cron* command dissected. The sixth field is only present in system crontabs. Fields are blank-separated, and shouldn't include comments.



```
[val@y550p ~]$ systemctl list-timers
NEXT                         LEFT      LAST                              PASSED       UNIT                         ACTIVATES
Вс 2016-09-11 17:40:10 YEKT  9min left n/a                               n/a          systemd-tmpfiles-clean.timer systemd-tmpfiles-clean.service
Пн 2016-09-12 00:00:00 YEKT  6h left   Вс 2016-09-11 17:25:48 YEKT  4min 55s ago logrotate.timer              logrotate.service
Пн 2016-09-12 00:00:00 YEKT  6h left   Вс 2016-09-11 17:25:48 YEKT  4min 55s ago man-db.timer                 man-db.service
Пн 2016-09-12 00:00:00 YEKT  6h left   Вс 2016-09-11 17:25:48 YEKT  4min 55s ago shadow.timer                 shadow.service

4 timers listed.
Pass --all to see loaded but inactive timers, too.
[val@y550p ~]$ systemctl list-timers --all
NEXT                         LEFT      LAST                              PASSED   UNIT                         ACTIVATES
Вс 2016-09-11 17:40:10 YEKT  9min left n/a                               n/a      systemd-tmpfiles-clean.timer systemd-tmpfiles-clean.service
Пн 2016-09-12 00:00:00 YEKT  6h left   Вс 2016-09-11 17:25:48 YEKT  5min ago logrotate.timer              logrotate.service
Пн 2016-09-12 00:00:00 YEKT  6h left   Вс 2016-09-11 17:25:48 YEKT  5min ago man-db.timer                 man-db.service
Пн 2016-09-12 00:00:00 YEKT  6h left   Вс 2016-09-11 17:25:48 YEKT  5min ago shadow.timer                 shadow.service

4 timers listed.
[val@y550p ~]$
```

There is no equivalent to **/etc/crontab** for *Systemd* timers. However, **systemctl list-timers** output is a close analogue.

for your own crontabs if you run crontab via **su**. Of course, you must be a superuser to make use of the **-u** switch. By the way, root may also have a per-user crontab, and no, it's not the same as **/etc/crontab**. Changes you make to per-user crontabs are applied automatically, and you don't have to restart the daemon.

You may restrict per-user crontabs with **/etc/cron.allow** and **/etc/cron.deny**. Both files contain usernames (one per line). If **/etc/cron.allow** exists, only users explicitly listed are eligible to run crontab and have private *cron* schedules. **/etc/cron.deny** works the same way, albeit it's a blacklist, not a whitelist. If both files exist, **/etc/cron.allow** takes precedence. That is, the user must be listed in **/etc/cron.allow** to be able to run crontab.

While you were sleeping

Cron works just fine, as long as your computer is switched on. But in *cron*, there is no such thing as an overdue task. If a job has missed the deadline for whatever reason, it has to wait until the next time. Depending on nature of the job, this could be a problem. For frequent jobs, waiting for another five minutes or half an hour usually don't hurt. Missing a daily package update is also okay, but not rotating logs may overfill your **/var** partition.

Anacron ("anachronistic cron") was designed to overcome this problem. It's not a daemon, but a tool which relies on something else (usually *cron*) to run itself in a timely manner. *Anacron* tracks jobs with day-level granularities. When it runs, it checks which jobs are overdue, executes them and exits.

More specifically, *Anacron* reads **/etc/anacrontab**. This file also has a crontab-like syntax. The usual environment settings such as **LOGNAME**, **PATH** and **MAILTO** are recognised, but job descriptions are a bit different:

```
1 5 cron.daily run-parts --report /etc/cron.daily
7 10 cron.weekly run-parts --report /etc/cron.weekly
@monthly 15 cron.monthly run-parts --report /etc/cron.
monthly
```

The first number is a job's period, measured in days. *Anacron* understands some period names as well, but what exactly it recognises depends on the flavour. In Debian, you can use **@monthly** and **@yearly**, albeit the man page mentions the former only. Red Hat's *cronie*, which also includes *Anacron*, adds **@daily** and **@weekly** to the set.

The second field is a job's delay, measured in minutes. The idea is not to spawn many jobs simultaneously, so when *Anacron* decides to run a job, it delays execution for the given number of minutes. *Cronie* improves uniformity even further with the **RANDOM_DELAY** environment setting (see **anacrontab(5)**).

The next field is a job's identifier. Any non-blank character except a slash is permitted. This identifier is used in the messages that *Anacron* produces, and for internal book-keeping. For instance, the **anacron** command receives a **job** argument, which is an ID (actually, a shell glob pattern) of the job to run. The final part of the line is a shell command to execute. Again, here's a **run-parts** invocation for daily, weekly and monthly *cron* jobs. If you compare this to the previous snippet, you'll see that *cron* simply offloads these jobs to *Anacron* if the latter is installed (**test -x /usr/sbin/anacron || ...**). This is not an option for hourly *cron* jobs, as *Anacron* doesn't support sub-day granularities.

When the system starts, init usually spawns *Anacron* to handle jobs that are currently overdue. *Cron* is also started, and it typically has *Anacron* somewhere in the system crontab:

```
$ cat /etc/cron.d/anacron
SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/
usr/bin
```

Cron flavors

Cron is not a single program. Like syslog, it is more of a standard for which numerous implementations exist in Linux (and BSD). Perhaps the most popular *cron* variant in Linux is *Vixie cron*. Paul Vixie authored it for 4BSD, but now it comes as a standard in many Linux distributions, including Ubuntu. As we know, *Vixie cron* doesn't handle overdue tasks, so Christian Schwarz introduced *Anacron* as a Perl script. Later, Itai Tzur reimplemented it in C. In 2007, Red Hat forked *Vixie cron* as *cronie* (**https://fedorahosted.org/cronie**), adding some security and configuration enhancements. In 2009, *cronie* embraced *Anacron*, adding *cron* period names (**@weekly**), among other things. *Cronie* is the default *cron* implementation for Red Hat distributions and Arch Linux.

And this isn't a whole story. Matthew Dillon, the author of DragonflyBSD, created *dcron*, and Arch Linux includes it as well. There is also *fcron* (**http://fcron.free.fr**), which was designed as a *Vixie cron* replacement. *Fcron* doesn't rely on anything external for asynchronous job scheduling, and it may run jobs depending on system uptime, akin to *OnBootSec* in *Systemd*. With *fcron*, one can set nice values for jobs and inhibit running them when the system is under heavy load, as in *at*. *Fcron* isn't compatible with *Vixie cron*, and none of the major Linux distributions ship it by default. However, Arch Linux has *fcron* in the official repositories.

With this variety of options, you can easily find the one to fit your needs, especially if you are on Arch Linux.

```
30 7 * * * root            start -q anacron || :
```

If your system happens to be awake at 7.30am, *cron* will spawn *Anacron* to run daily (and also weekly and monthly) jobs. If not, init will do it on the next startup. Alternatively, you could use a dedicated **@reboot** schedule for the **anacron** command in **/etc/crontab**. Such commands run only once when the *cron* daemon is started.

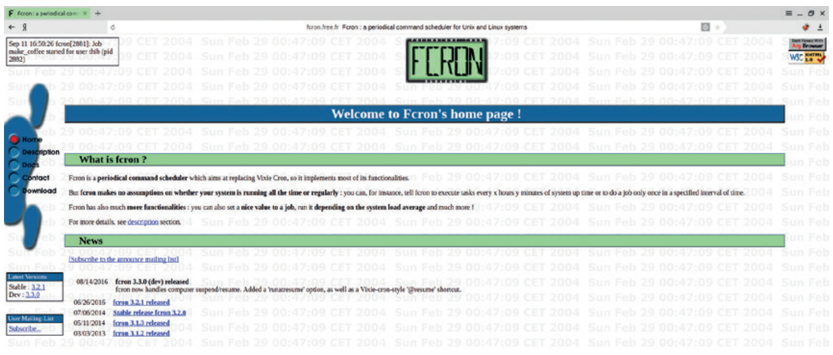
Systemd route

Systemd is your best anything in modern Linux. No wonder it provides its own job scheduler, which can be used as a *cron*/*Anacron* surrogate.

More specifically, *Systemd* has a notion of "timers". A timer is just a unit file that carries the **.timer** suffix. Timers control service units, so for any scheduled job you'd need two unit files: one **.service** and one **.timer**. As a rule, they carry the same name (except for the suffix), but that's configurable. Having a separate unit for a service eases debugging a bit, as you can activate it manually. It is possible to bind a timer to a calendar or to some other event, and *Systemd* can run overdue tasks without any external tools involved. The most important section in the timer unit file is **[Timer]**:

```
[Unit]
Description=Sample timer
[Timer]
OnBootSec=10min
OnActiveSec=1h
RandomizedDelaySec=60
[Install]
WantedBy=timers.target
```

This defines a so-called "monotonic timer". It triggers 10 minutes after the system is booted, and every hour since that, regardless of whether the service was started manually or not in between. If an **OnBootSec** timer is overdue, it fires immediately. **RandomizedDelaySec** is again a way to stretch the load across some activation interval.



fcron.free.fr Fcron : a periodical command scheduler for Unix and Linux systems

Sep 11 16:50:26 fcron[2881]: Job make_coffee started for user thib (pid 2882)

Welcome to Fcron's home page !

What is fcron ?

Fcron is a **periodical command scheduler** which aims at replacing Vixie Cron, so it implements most of its functionalities.

But **fcron makes no assumptions on whether your system is running all the time or regularly** : you can, for instance, tell fcron to execute tasks every x hours y minutes of system up time or to do a job only once in a specified interval of time.

Fcron has also much **more functionalities** : you can also set a **nice value to a job**, run it **depending on the system load average** and much more !

For more details, see description section.

News

[Subscribe to the announce mailing list]

Latest Versions
Stable : 3.2.1
Dev : 3.3.0

User Mailing List
Subscribe...

08/14/2016 **fcron 3.3.0 (dev) released**
fcron now handles computer suspend/resume. Added a 'runatresume' option, as well as a Vixie-cron-style '@resume' shortcut.

06/26/2016 fcron 3.2.1 released
07/06/2014 Stable release fcron 3.2.0
05/11/2014 fcron 3.1.3 released
03/03/2013 fcron 3.1.2 released

fcron is an all-mighty, yet backwards-incompatible *Vixie cron* replacement. Perhaps that's the reason you don't encounter it too often.

For other jobs, such as planned maintenance, you may want a real-time (wall clock) scheduling. *Systemd* supports it via calendar events timers. The corresponding **[Timer]** section may look like this:

```
[Timer]
OnCalendar=daily
Persistent=true
```

Persistent is important for *Anacron*-like functionality. It tells *Systemd* to store the last invocation's timestamp on the disk, and fire a calendar event immediately if the timer is already overdue. For **OnCalendar**, all other period names such as **weekly** are recognised. Moreover, you may use expressions such as **2016-*-1 17:00:00**. See **systemd.time(7)** for details.

While *Systemd* timers are similar in spirit to *cron*, they are not equivalents. There is no crontab, and no out-of-the-box way to send an email about failed jobs. The *Systemd-cron* project aims to bridge both worlds. It translates between crontabs and *Systemd* timer units, and handles failure email notifications, among other things. This way, you can continue using the crontab interface on top of *Systemd*, although Lennart would argue it's not a good idea. Details are here: **https://github.com/systemd-cron/systemd-cron-next**. Just keep in mind that nothing stops you from running original *cron* on any *Systemd*-managed system as well.

Command of the month: at

Cron is great for recurring tasks, but what if you want your job to run only once? You need the **at** command:

```
$ at 00:10
```

First, you tell **at** when to run a job. Time specification is rather flexible, with both absolute (16:00) and relative (now + 12 hours or 16:00 + 2 days) specs supported. If the time specified has already passed, and no date was specified, **at** assumes you meant tomorrow.

If everything is OK, **at** opens a shell where you enter commands to run. They are fed to **/bin/sh**, and **at** mails their output back to you (press Ctrl+D to exit).

```
warning: commands will be executed using /bin/sh
at> ...
at> <EOT>
```

Alternatively, you could store commands in a file and supply it via **at -f**. Either way, **at** will tell you the job's identifier and when it is supposed to run:

```
job 1 at Sat Sep 10 00:10:00 2016
```

The daemon named **atd** is responsible for actual execution. You can look what's in its queues with **atq**:

```
$ atq
1              Sat Sep 10 00:10:00 2016 a val
```

a is the name of the queue. **a–z** and **A–Z** are valid queue names, and you can tell **at** which queue to use with the **-q** command line argument. Higher queue identifiers mean increasing nice values for the job. Uppercase letters work as if **batch**, not **at**, was used to place a job. The difference is that **batch** jobs aren't executed if the system load is high.

If you changed your mind after placing a job, use **atrm** to remove it by the identifier, eg **atrm 1**.

LINUX INSIDE: FRITZ!BOX 7490

This router runs Fritz!OS, a specially built distribution of Linux. Actually, we could have picked from any of a huge number of home routers where the power of the Linux networking stack combined with its ability to run on inexpensive hardware makes it the dominant operating system in the market. Most of the time these routers are pretty locked down, but some – including the Fritz!Box – let you make changes.

If you can make judicious use of the often-limited hardware, this gives you an always-on Linux server at the heart of your home network. It can share files, block adverts, or do almost anything else you want. If you want to get started unleashing the full power of your router, head to https://www.openwrt.org, one of the most active router-hacking sites on the web. It costs more than the free router you get from your ISP, but by jimminy it's worth it.

Nick Veitch was the original editor of Linux Format, a role he played until he got bored and went to work at Canonical instead. Splitter!

"Didn't you used to be famous?" is something I was asked recently (it depends – back in the 90s, I *was* in a very famous magazine). The discussion led on to why I no longer worked at the coalface of media, and didn't I miss it etc.

The answers are, because like actual coal faces, that place doesn't exist anymore. Sure I miss it, it really was the best of times in a lot of ways (as well as the worst in others), but the question is moot – I can't (until my son completes his time-machine) go back. The magazine industry is different now. Some things have progressed quite a way – I used to have to sift through sacks of actual mail; moving files from place to place relied on copying them onto unreliable media and physically transporting it somewhere; the list of the lost is a long one. But it has changed.

Misty-eyed nostalgia

I was reminded of this again recently when I upgraded my distro. My old, favoured IRC client (OK, *some* things never change) was no longer available. It was unmaintained, so I had to move to a newer fork of the original. Of course I spent about two weeks complaining about things being in the wrong place and the default config being in the wrong place. Objectively, it is better, but like most humans, there is comfort in the familiar. As the sensible chaps who host the Linux Luddites podcast quite correctly say, not all change is progress.

But I persevered. I have never been afraid of change. Change is afraid of me. Change is inevitable – sometimes it is good, sometimes not so good, but in general, in most things we have to hope it leads in the right direction. Otherwise we would all still be running CP/M.

LV