ARCHIVE DVD

HUGE SAVINGS! $39.90 VALUE!

DVD INSIDE! ALL 214 ISSUES ON A SEARCHABLE DISC!

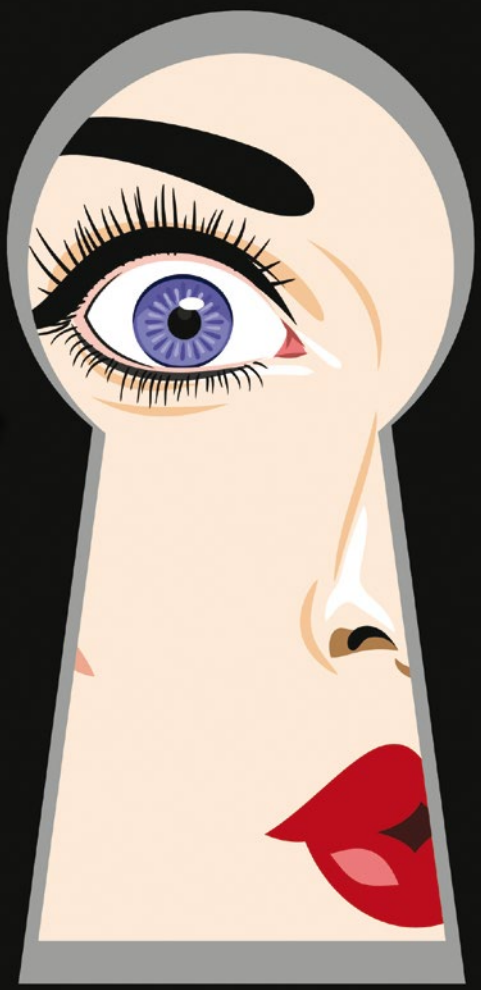
PRIVACY
STOP SNOOPERS AND PROTECT YOUR IDENTITY

LINUX PRO MAGAZINE

OCTOBER 2018

PRIVACY

Stop snoopers and protect your identity



Rockbox
Liberate your music player with free firmware

RSS Readers
View and organize all your favorite news sources

PiXtend
More interfaces for your Rasp Pi

Rasp Pi Sailboat
Ride the winds with the tiny Pi Zero

Ready to Go?
Smart queries in the powerful Go language

LINUXVOICE

- Secrets of data compression
- maddog: Life of Linux
- Pass: Protect your accounts with this free password manager

FOSSPicks
- cheat.sh
- HyperRogue

Tutorials
- Metadata
- Docker 101 for End Users

Issue 215
Oct 2018
US$ 19.99
CAN$ 24.99

HETZNER ONLINE

ULTIMATE PERFORMANCE
COMPLETE FLEXIBILITY

BRAND NAME HARDWARE FROM DELL

Dedicated Root Server DX152

- ✔ Dell PowerEdge™ R640
- ✔ 2 x Intel® Xeon® Silver 4114 10-Core (Skylake-SP)
- ✔ Incl. Hyper-Threading-Technology
- ✔ 64 GB DDR4 ECC RAM
  max. 768 GB at additional cost
- ✔ Up to 10 drives at additional cost
- ✔ No minimum contract
- ✔ Setup Fee $187

monthly from $ **187**

Dedicated Root Server DX292

- ✔ Dell PowerEdge™ R640
- ✔ 2 x Intel® Xeon® Gold 6130 16-Core (Skylake-SP)
- ✔ Incl. Hyper-Threading-Technology
- ✔ 64 GB DDR4 ECC RAM
  max. 768 GB at additional cost
- ✔ Up to 10 drives at additional cost
- ✔ No minimum contract
- ✔ Setup Fee $257

monthly from $ **257**

www.hetzner.com

FREE AS IN VOTE

Dear Reader,

I get this familiar feeling whenever an election year rolls around. I guess it is kind of like despair mixed with something more proactive, like maybe *annoyance*. I'm not talking about politics exactly, although I will admit that politics get pretty annoying. What really concerns me now is the backward nature of voting technology and the sense that nothing ever gets done about it.

The vote-counting fiasco of the US 2000 election was 18 years ago. Since then, numerous studies have shown that our voting machines are insecure, and we have uncovered evidence of foreign powers attempting to hack our voting systems. If you're wondering "why isn't this problem fixed yet?" you're not alone.

It is fair to say that some (though not all) of the very worst machines have been retired in recent years, but other systems that have some pretty severe problems are still in active use. Many voting machines use software from the 1990s – including obsolete OpenSSL implementations and unpatched versions of Windows XP. And because these systems are all proprietary and closed source, the world has no way to audit them and see how broken they really are. At the recent DEF CON conference in Las Vegas [1], testers revealed numerous security issues with voting machines in use today. One had an SSL certificate that was five years old. Another had an easily accessible memory card, which an attacker with physical access could swap out, exploiting software vulnerabilities to get control and change vote totals. These stories come back every year, typically before an election, and everyone gets shocked; then after the election, the problem floats back down to the end of the priority queue.

In the midst of all this grim news, one very interesting and hopeful development is the ongoing work of the TrustTheVote project [2] and its parent organization, the Open Source Election Technology (OSET) Institute [3]. TrustTheVote is an effort to design a complete framework for the voting process, including registration, voting, and counting, that is logical, unified, sensible, and secure. OSET's Election Technology Framework will be based on open standards, so everyone in the world will know how it works, and the powerful crowd-sourcing capabilities of the open source development model will provide universal auditing and feedback to ensure that the system remains secure and up-to-date.

OSET and TrustTheVote are interested in the engineering. Instead of serving as just another voice in the room, they want to build the system that the other voices are talking about. As they state on their website:

- No lofty academic research papers
- No congressional testimony
- No reliance on bureaucracy
- No endless public debates
- No TV news talking heads

Their focus is on "designing, developing, testing, and making available real production-ready and more trustworthy election administration software."

OSET and TrustTheVote have no intention of acting as vendors or distributors of voting machines. Their mission is to build a software framework that is then available to any vendor who wants to use it.

The voting machine industry in the US is mostly controlled by three companies, and those companies have changed very little over the years. In spite of all the negative publicity, no market forces have actually caused them to stop selling their cryptic, invisible systems to non-technical election officials. But open source software has great potential for disruption.

If the TrustTheVote project succeeds in bringing their framework into the discussion, voting machine vendors will have to make a choice. They can cooperate with TrustTheVote, integrating the universal election platform into their systems, and probably achieve vast savings in development costs, but they will need to give up some of their capacity for secrecy and competitive obfuscation.

On the other hand, if they insist on continuing to market their archaic, black-box systems, they will risk losing business to mainstream vendors such as HP, Oracle, and IBM, who will be perfectly happy to integrate TrustTheVote's framework rather than having to develop their own.

The Election Technology Framework is still a work in progress. According to the website, the current timeline calls for TrustTheVote to deliver "… production candidate election management systems plus ballot casting and counting devices for test and evaluation with the goal of being ready for deployment in the 2020 election cycle." Much has already been accomplished, but much work remains.

The TrustTheVote project could use more volunteers, especially volunteers who understand the importance of open standards and open source software. Of course, they also welcome donations, but another gift you can give to the TrustTheVote project is your awareness. Visit their website, send the link to your friends, and let the world know that we really do have a chance for better election security if concerned citizens tune in.

Joe Casad, Editor in Chief

Info

[1] US Voting Systems: Full of Holes, Loaded with Pop Music, and Hacked by an 11-Year-Old: *https://www.theregister.co.uk/2018/08/13/defcon_election_vote_hacking/*

[2] TrustTheVote: *https://trustthevote.org/*

[3] Open Source Election Technology Institute: *http://www.osetfoundation.org/*

LINUX MAGAZINE

WHAT'S INSIDE

This month we study some tricks for improving privacy on your Linux system. You'll learn how to turn off the webcam and mic when they are not in use, and we'll show you some extensions and advance configuration settings for dialing up your privacy in Firefox. Other highlights include:

- **RSS Readers** – With all the problems associated with getting news from social media, old-school RSS aggregators are making a comeback. We show you some leading contenders for organizing your news feeds (page 34).
- **Pi Zero Sailboat** – The Node-RED dashboard tool lets you build a simple user interface for your IoT creations (page 58).

Also, check out MakerSpace for a look at how to build web-based controls for a Raspberry Pi sailboat, and read on to LinuxVoice for an introduction to memory compression and a tutorial on getting useful information from LibreOffice metadata.

SERVICE

3 Comment
6 DVD
96 Featured Events
97 Call for Papers
98 Preview

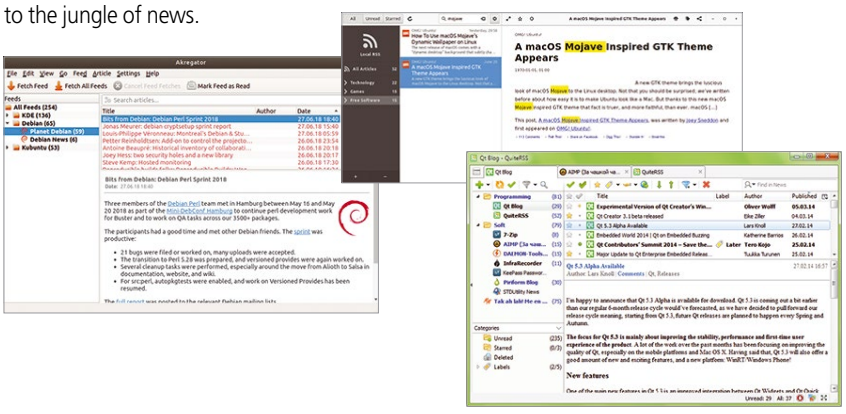
NEWS

08 News
- Chromebooks Support Debian Applications
- Opera Embraces Snap for Linux
- Canonical Fixes Boot Failure Issues in Ubuntu
- Weird Unofficial LibreOffice Version Shows Up in the Microsoft Store
- New Version of the Spectre Vulnerability Allows Attack from the Network
- SUSE Sold for $2.5 Billion

12 Kernel News
- New GNSS GPS Subsystem
- New LoRaWAN Subsystem
- Tracking Compiler Dependencies at Config Time
- Uninlining for Debugging

REVIEW

32 RSS Readers
RSS feed readers bring order and clarity to the jungle of news.



COVER STORIES

16 Privacy in Firefox
The Firefox browser is not so private under its default settings, but several add-ons and configuration settings will help you keep the spies in the dark.

22 Privacy Hacks
You don't have to dig deep into your toolbox to protect your privacy: With a few simple tricks, you can disable the webcam and microphone and permanently delete data from your hard disk.

28 Cryptomator
Cryptomator adds encryption to the cloud storage environment.

IN-DEPTH

40 Freeing Your Music Player with Rockbox
Turn your music player into open hardware with Rockbox's free firmware.

44 Command Line – zstd
In an effort to meet modern computing needs, zstd offers a greater degree of compression at a faster rate, with unique options to enhance performance.

48 Charly's Column – SSH Tunnel
Charly draws attention to a widely unknown weather phenomenon: The instability of rarely used tunnels leading to a Raspberry Pi. Read on for greater insights.

50 Programming Snapshot – Go
To find files quickly in the deeply nested subdirectories of his home directory, Mike Schilli whips up a Go program to index them in an SQLite database.

MAKERSPACE

56 PiXtend v2
The PiXtend board extends the Raspberry Pi with many useful interfaces and functions for new target groups.

58 Pi Zero Sailboat
Use the Node-RED programming tool to create a web dashboard for controlling a toy sailboat.

62 Open Hardware – Prosthetic Hand
In true open hardware spirit, Social Hardware looks to produce a development kit for prosthetic hands to help rural amputees in India.

LINUXVOICE

65 Welcome
This month in Linux Voice.

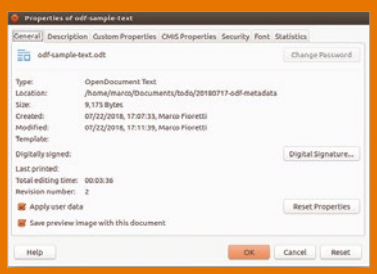
67 Doghouse – Linux History
"maddog" takes us on a brief tour of Linux history.

68 Memory Compression
Data compression costs virtually no computing power today. Why not save some space by putting data compression techniques to work on RAM and cache memory?

72 Pass
This simple shell script helps you manage and synchronize passwords using Git.

76 FOSSPicks
This month Graham looks at Brackets, Browsh, Borderlands, Timekpr, Fractal, HyperRogue, and much more!

82 Tutorial – Metadata
ODF files contain useful metadata that is very easy to read or modify.



90 Tutorial – Docker 101
The Docker container management system isn't just for sys admins. Here's how to get started implementing your own Docker container environment.

On the DVD

Linux Magazine Archive DVD

This month's DVD includes the 2018 edition of the Linux Magazine Archive DVD – every previous issue of Linux Magazine on a single, searchable disc. Browse the pages of every article we've ever published, and experience our special brand of technical yet accessible how-to insights.

If you want to learn about a common tool in the open source space, search this disc – we've probably written about it more than once over the last 18 years. Catch all the articles you missed – on scripting, system administration, security, containers, cloud computing, Raspberry Pi, and more! Discover desktop applications that will save you time and simplify your life, and relive important moments in the history of Linux: the SCO case, the Novell/Microsoft pact, the birth of Git, the mobile revolution, ...

All the issues on this disc would cost you about $3,400 in our shop – if they all still existed, but many older issues are out of print: **The only way to experience them is through the Linux Magazine Archive DVD!**

Defective discs will be replaced. Please send an email to subs@linux-magazine.com.

openSUSE

NEWS

Updates on technologies, trends, and tools

THIS MONTH'S NEWS

08
- Chromebooks Support Debian Applications
- Opera Embraces Snap for Linux

09
- Canonical Fixes Boot Failure Issues in Ubuntu
- Weird Unofficial LibreOffice Version Shows Up in the Microsoft Store
- More Online

10
- New Version of the Spectre Vulnerability Allows Attack from the Network
- SUSE Sold for $2.5 Billion

Chromebooks Support Debian Applications

Google is finally bringing the ability to install and run traditional Linux apps in Chrome OS. The company announced Project Crostini back in May during the Google I/O event. Initially, it was announced for Google Pixel, but support for Linux started landing on supported devices recently.

Chrome Unboxed, a site that covers Chrome OS, reported that they have managed to install Debian apps on Chromebook (*https://chromeunboxed.com/news/chrome-os-linux-debian-packages-chromebook-crostini*).

If you are running the dev channel of Chrome OS, you can easily enable support for Linux on Chromebooks. All you need to do is go to *Settings | About Chrome OS | Detailed build information* and change the channel from *stable* to *dev*. It will ask you to power wash your device, which means deleting all data and reformatting the machine. Once the device is power washed, you would be running the latest dev branch of Chrome OS.

Users running the dev channel will notice an option to enable Linux apps under the *Settings | Device* option. Once you enable Linux, it will download and install the terminal app, which runs Debian with custom packages.

Users can simply run `apt-get` to update Debian on Chromebook and install desired apps. Of course, it's just the beginning and things need to be ironed out.

Opera Embraces Snap for Linux

Ubuntu's Snap is gaining popularity. After Microsoft, now Opera is backing the Snap packaging format to distribute their apps to the Linux platform. Opera may not be one of the most popular browsers today, but they did a lot of innovation in the past, including tabs, saved sessions, pop-up blocking, and speed dial. Opera and Canonical, the parent company of Ubuntu, worked together to bring Opera web browser to Linux and Snap (*https://www.operasoftware.com/press/releases/desktop/2018-08-02*).

"The addition of Opera to the Snap Store enables users of all major Linux distributions to benefit from the auto-updating and security features that Snap provides. The Opera Snap is supported on Debian, Fedora, Linux Mint, Manjaro, Elementary, openSUSE, Ubuntu, and more distributions," Canonical said in a press release.

"We are delighted to welcome Opera to the Snap Store and further expand the choice of applications available to the Linux community. It is popular applications, such as Opera, that have driven the impressive growth of new Snaps to the store and ever-increasing user installs over the last year," added Jamie Bennett, VP of engineering, IoT, and devices at Canonical.

To those who don't know, Snaps are containerized software packages, inspired by Docker containers, that are designed to offer isolation as well as fully self-contained packages that don't rely on system libraries and dependencies. As a result, developers can use the latest libraries and offer new features without being tied to the system. Snaps also help in treating Linux as a single platform instead of looking at each distro as a platform.

Snaps may help bring more mainstream apps to Linux.

Canonical Fixes Boot Failure Issues in Ubuntu

Canonical has been playing a cat-and-mouse game with patches and vulnerabilities. Canonical has released an update that fixes boot failures of machines running Ubuntu 18.04 LTS and 16.04 LTS.

Earlier this month, Canonical released security updates (USN-3695-1) for Ubuntu 18.04 LTS to fix six known vulnerabilities. According to the Ubuntu advisory, "Unfortunately, the fix for CVE-2018-1108 introduced a regression where insufficient early entropy prevented services from starting, leading in some situations to a failure to boot."

The latest update fixes the regressions. Canonical urges users to update their systems immediately. If you have installed any third-party kernel modules, you will have to recompile and reinstall them.

"Due to an unavoidable ABI change, the kernel updates have been given a new version number, which requires you to recompile and reinstall all third-party kernel modules you might have installed. Unless you manually uninstalled the standard kernel metapackages (e.g., *linux-generic*, *linux-generic-lts-RELEASE*, *linux-virtual*, *linux-powerpc*), a standard system upgrade will automatically perform this as well."

This is the third time Canonical has released fixes in the last 30 days. In June, Canonical released a patch for Ubuntu 14.04 LTS that lead to boot failure on some machines.

Source: *https://lists.ubuntu.com/archives/ubuntu-security-announce/2018-July/004503.html*

Weird Unofficial LibreOffice Version Shows Up in the Microsoft Store

A unofficial version of LibreOffice shows up in the Microsoft Store. The app was published by an obscure developer under the name ".net." There is no additional information about the developer. Clicking on the URL takes you to another app by the developer named "dress my doll."

How did this app make it into the store? Given the volume of apps submitted to Microsoft Store, App Store, and Google Play, it's virtually impossible for these vendors to vet each app manually. They all have an automated process.

Microsoft also has a certification process (*https://docs.microsoft.com/en-us/windows/uwp/publish/the-app-certification-process*): "When you finish creating your app's submission and click *Submit to the Store*, the submission enters the certification step. This process is usually completed within a few hours, though in some cases it may take up to three business days. After your submission passes certification, it can take up to 24 hours for customers to see the app's listing for a new submission, or for an updated submission with changes to packages. If your update only changes Store listing details, the publishing process will be completed in less than an hour. You'll be notified when your submission is published, and the app's status in the dashboard will be *In the Store*."

It's not clear if Microsoft also validates the authenticity of the app. It's not surprising that an app like LibreOffice would slip through the certification process and be available to users. Since LibreOffice is a fully open source project, anyone can compile it and redistribute the app, as long as they follow the terms of the license.

I reached out to The Document Foundation (TDF), the organization responsible for LibreOffice, and Italo Vignoli, one of the cofounders of TDF told me, "The Document Foundation has been made aware of an unofficial version of LibreOffice on the Windows [Microsoft] Store. We are investigating further, but we want to be clear: This is not an official version created by The Document Foundation, so the app's page is misleading. The only official source of the software (which can be downloaded for free, i.e., without any cost for the end user) is the LibreOffice website

MORE ONLINE

Linux Magazine
www.linux-magazine.com

ADMIN HPC
http://hpc.admin-magazine.com/

pdsh Parallel Shell • Jeff Layton
The pdsh parallel shell tool lets you run a command across multiple nodes in a cluster.

ADMIN Online
http://www.admin-magazine.com/

Flatpak, Snap, and AppImage • Valentin Höbel
The Flatpak, Snap, and AppImage package formats work across distributions, but each has its specific disadvantages.

Effective debugging of Docker containers
Martin Loschwitz
Bugs can live in Docker containers. Read on for tips on how to debug them.

Continuous integration with Docker and GitLab
Martin Loschwitz
GitLab provides the perfect environment for generating Docker containers that can help you operate critical infrastructure reliably and reproducibly.

ADMIN DevOps Focus
http://www.admin-magazine.com/DevOps

AWX: Web-Based Console Manager for Ansible • Chris Binnie
The upstream project of the Ansible Tower enterprise solution is now freely available as AWX. We look at Red Hat's new web-based console manager for Ansible deployments and discover its capabilities.

(*https://www.libreoffice.org/*). Also, the money from the Microsoft Store version is not collected by The Document Foundation."

My advice is to not download and install the app from Microsoft Store as we are unsure if there is any malicious code in it. Microsoft says it checks for malicious code before any app is published; it's better to be safe than sorry.

New Version of the Spectre Vulnerability Allows Attack from the Network

Monthly reports of new Spectre-related vulnerabilities are keeping security experts busy. Now a team of security researchers at the Graz University of Technology (Austria) has discovered another flaw, dubbed NetSpectre, that allows attacks over the network.

The crux of Spectre vulnerabilities is the way modern CPUs speculate on which workload will run next to improve performance. According to the team, "During speculative execution, the processor may perform operations the program usually would not perform. While the results of such operations are discarded if the speculative execution is aborted, microarchitectural side effects may remain."

© alphaspirit, 123RF.com

Attackers exploit these side effects to read memory contents. Previous versions of the Spectre attack have required some kind of local code executive to launch the attack, but the latest discovery changes that.

"NetSpectre marks a paradigm shift from local attacks to remote attacks, exposing a much wider range and larger number of devices to Spectre attacks. Spectre attacks now must also be considered on devices which do not run any potential attacker-controlled code at all," wrote the researchers.

The team informed Intel back in March, and Intel has patched the problem during previous patches released by the company. The best available defense is to keep your systems up to date and install all security patches.

SUSE Sold for $2.5 Billion

SUSE is like a seasoned football player who changes ownership after a few successful seasons. This time the Swedish group EQT is buying SUSE from British-owned Micro Focus. This is the fourth sale of SUSE since its inception in 1992, a year after Linus Torvalds announced the Linux kernel.

What's different this time is that SUSE is being acquired by an investment firm and not a tech company. SUSE CEO, Nils Brauckmann, sees this as a move towards independence, with the company charting its own course instead of being a business unit of another tech company. "By partnering with EQT, we will become a fully independent business," said Brauckmann. "Together with EQT, we will benefit both from further investment opportunities and having the continuity of a leadership team focused on securing long-term profitable growth combined with a sharp focus on customer and partner success."

SUSE is well aware of the fact that the open source community will be keeping a close eye on this development. In a Hangout chat, Richard Brown, openSUSE Board Chairman, and the face of openSUSE community, told me that he received a phone call from Brauckmann updating him with the news and also reassuring him that nothing will change when it comes to open source and community engagement.

"As a SUSE employee, I'm excited about my employer's new owners. As an openSUSE Contributor, I'm not only excited, but thrilled at the proactive steps SUSE has taken to reassure the community, which really shows just how well SUSE understands how to operate as part of the open source world," Brown said.

In case you are curious, EQT is an investment firm with approximately EUR50 Billion in raised capital across 27 funds. EQT has portfolio companies in Europe, Asia, and the US with total sales of more than EUR19 Billion and approximately 110,000 employees.

Linux Magazine **is your guide to the world of Linux. Look inside for advanced technical information you won't find anywhere else!**

Expand your Linux skills with:

- In-depth articles on trending topics, including Bitcoin, ransomware, cloud computing, and more!
- How-tos and tutorials on useful tools that will save you time and protect your data
- Troubleshooting and optimization tips
- Insightful news on crucial developments in the world of open source
- Cool projects for Raspberry Pi, Arduino, and other maker-board systems

If you want to go farther and do more with Linux, subscribe today and never miss another issue!

Subscribe now!

shop.linuxnewmedia.com/subs

GET IT NOW!
FAST DELIVERY WITH OUR PDF EDITION

Zack's Kernel News

Chronicler Zack Brown reports on the latest news, views, dilemmas, and developments within the Linux kernel community. *By Zack Brown*

New GNSS GPS Subsystem

There was recently a scuffle over adding a new subsystem that ended with an odd resolution. Johan Hovold posted some code to add a Global Navigation Satellite System (GNSS) subsystem to the Linux kernel to support GPS devices. One of the motivations for this was the wide array of input/output systems used by these devices. Some relied on UART for communications over a serial port; others used USB ports. There were a variety of other interfaces, too. Johan wanted to create an abstraction layer, so user code could interact with GPS devices regardless of their particular hardware interface requirements. His idea was to create a new `/dev/gnss0` file in user space, which could be used to query and control any GPS device attached to the running system.

Author

The Linux kernel mailing list comprises the core of Linux development activities. Traffic volumes are immense, often reaching 10,000 messages in a week, and keeping up to date with the entire scope of development is a virtually impossible task for one person. One of the few brave souls to take on this task is **Zack Brown**.

To get things started, Johan had also implemented drivers for the SiRFstar and u-blox GNSS chips.

Pavel Machek could not go along with this. He pointed out that there was not, in fact, any GNSS-specific code in Johan's patches and that the code was simply a serial-device power management subsystem that couldn't handle a variety of GPS devices. He said, "This will never handle devices like Nokia N900, where GPS is connected over Netlink," an already-existing socket interface within the Linux kernel.

Johan replied that the whole point of the abstraction layer was to let user space detect GPS devices without having to come up with its own hacky code to probe for them. That was what made it GNSS-specific. The latest version of his patch, he said, also exported a GNSS receiver type and also implemented the GNSS-specific drivers for SiRFstar and u-blox. Finally, he pointed out that the code was not serial device specific, because it worked on other interfaces too.

Pavel still disagreed. He said the exact same features Johan had implemented could be applied to AT modems (i.e., modems using the Hayes AT command set). Pavel pointed out that AT commands also went across a variety of different interfaces.

Pavel said there were plenty of GPS devices, such as the Nokia N900, that wouldn't work with Johan's code. A proper GNSS subsystem would need to support all GPS devices. He said, "I believe we really want to use your code for AT commands, too. And we really should keep GNSS/GPS names for future layer that actually provides single interface for userland."

Johan defended his patch, saying, "It's a matter of finding the right abstraction level. A userspace location service will now have easy access to the class of devices it cares about, without having to go through a list of other random devices which happen to use a similar underlying interface."

He added that if part of his code turned out to be reusable for a yet-deeper subsystem in the future, he'd be fine with that. But for the moment, he said, his code met the needs of the situation.

Johan also added that some vendors used their own priority binary-only protocols for GPS devices, and there it wouldn't be feasible to reverse engineer and support them all. He said that for a lot of those devices, it would be fine to let them remain a userspace problem. There was no need to put support for all of them into the kernel itself. Only the ones that were amenable.

Pavel reiterated his complaint that Johan's code was not a real GNSS subsystem. He said Johan should pick a name that more accurately described what his code did, so that a real GNSS subsystem wouldn't come along and find its proper name already taken.

But Johan replied, "It's about grouping related devices together, devices which share some common functionality. In this case, providing location data from some satellite system. I really don't understand how you can find having a class type named `gnss` for this to be controversial in any way. […] I find naming a subsystem for GNSS receivers `gnss` to be very reasonable."

He pointed out that the patch solved a real-world need, and he remarked that if Pavel found a better GNSS subsystem in the future, it could always be added into the kernel at that time.

Johan also looked at the specs for the Nokia N900 that Pavel had mentioned and said:

The N900 service you link to above, parses phone data, does some floating point calculations, and generates NMEA sentences, which it feeds to a pseudo terminal whose slave side is opened by `gpsd`.

That NMEA data could just as easily be fed through a different kernel subsystem, namely `gnss` *instead of* `tty`*, where it could be accessed through a common interface (for now, a raw* `gnss` *interface, with some associated metadata). (And from what I can tell,* `ugnss` *would also allow you to get rid of some hacks related to finding out when the GNSS is opened and needs to be powered on.)*

So the `ugnss` *interface looks like it will work for N900 just as it will for other phones.*

Pavel disagreed with Johan's idea of grouping devices together by functionality. He said, "We normally group devices by interface, not by functionality." He added that if the wrong name was chosen now, it would be impossible to fix later.

Johan replied:

I started off with separating the `gnss` *device itself from the raw interface (cf. hid) to allow for something like that, but the more I looked into this, the more it seems I was just over-engineering for something that would never be realized.*

Take a look at some of the papers on the `gpsd` *site about GNSS protocols and the problem of finding a common representation for all the various devices out there.* `gpsd` *itself has already gone through three revisions of its internal representation over the past decades. This does not seem like an exercise we want to repeat in the kernel with its rules about backwards compatibility, etc.*

So at least for the time being, I'm convinced that a raw `gnss` *interface is the right one.*

At this point, Greg Kroah-Hartman undercut the entire debate with one sweeping gesture. He accepted the patch, saying, "This all looks great. Thanks for doing this work and adding a new subsystem for something that has been asked for for many years. All now merged in my tree, nice job!"

Pavel replied that the debate wasn't finished yet. He reiterated his basic points, saying "there's nothing GNSS specific in those patches. It does not know about the format of the data passed around. (Best you can claim that somehow data flow characteristics are unique to GNSS.) And this takes namespace needed for real GNSS subsystem. Please don't do it."

But Johan just replied, "This is the real GNSS subsystem. Get over it." And Greg said the debate looked done to him. He said, "there was only a single set of patches, with no other working patches submitted from anyone else. If this turns out to be a 'bad' api, then we can deal with it then, but for now let's try this out."

And that was that.

It's unusual for one big-time kernel person (Greg) to overrule another big-time kernel person (Pavel) so abruptly. At the same time, Johan's code did address a current need, and no alternatives rose up to present themselves. One aspect of Linux development philosophy seems to be accepting a less-than-perfect solution, partly because it is a solution, and partly because it forces the naysayers to put up or shut up. Linus Torvalds tried this with BitKeeper years ago. While it didn't result in the naysayers producing a viable alternative, it was itself a good temporary fix and eventually did lead to the creation of the Git revision control system and the changing of the world for the better.

New LoRaWAN Subsystem

Jian-Hong Pan wanted to know if there was support among kernel developers for a new LoRaWAN subsystem. He explained, "A Low-Power Wide-Area Network (LPWAN) is a type of wireless telecommunication wide area network designed to allow long range communications at a low bit rate among things (connected objects), such as sensors operated on a battery. It can be used widely in IoT area. LoRaWAN, which is one kind of implementation of LPWAN, is a medium access control (MAC) layer protocol for managing communication between LPWAN gateways and end-node devices."

He asked, "Could or should we add the definitions into corresponding kernel header files now, if LoRaWAN will be accepted as a subsystem in Linux?" And he posted a link to his Git repository that held his work so far: *https://github.com/starnight/LoRa/tree/lorawan-ndo/LoRaWAN*.

Jiří Pírko pointed out that a repository wouldn't be enough – Jian-Hong would need to send a patch against the kernel tree itself.

And Marcel Holtmann remarked, "when you submit your LoRaWAN subsystem to NetDev for review, include a patch that adds these new address family definitions. Just pick the next one available. There will be no pre-allocation of numbers until your work has been accepted upstream. Meaning, that the number might change if other address families get merged before yours. So you have to keep updating. `glibc`

will eventually follow the number assigned by the kernel."

Andreas Färber also replied, saying he'd been working on a similar project for the past year. He gave a link to his proof-of-concept code: *https://github.com/afaerber/lora-modules*.

Andreas asked if Jian-Hong thought their two projects were independent or in conflict with each other.

Jian-Hong replied, "Wow! Great! I get new friends:)"

He said it looked like their projects had the same idea and proceeded to examine possible further areas of overlap and/or conflict, but the discussion ended there, probably because Andreas and Jian-Hong took it to private email.

It's not uncommon for two developers to suddenly discover that they've been working on the same project. It seems like a basic part of the scratch-an-itch philosophy is that more than one person might feel the same itch at the same time. Sometimes a developer might work in private for a long time, reluctant to reveal their project until they felt it could be defended. Sometimes two projects might represent such an opposing worldview that they are truly mutually exclusive: two different schedulers, for example, or two different out-of-memory killers, or two different load balancers. But two different implementations of the same known protocol are much more likely to mesh well and end up enhancing each other rather than forming a new point of conflict.

My guess is that Andreas and Jian-Hong will merge their projects, form a team, and get their code into the kernel twice as fast as they would have before.

Tracking Compiler Dependencies at Config Time

Masahiro Yamada recently posted some documentation for the new elements of the Kconfig macro language requested by Linus Torvalds in February. The idea was that the kernel build process had been accruing a vast number of hacky tests for various compiler features, covering kernel options that depended on whichever compiler the user had installed. There was no way, with the current setup, to reveal those details during the configuration phase.

Linus wanted the Kconfig macro language to include features to specify dependencies on specific compiler versions or particular compiler features, so that all of those dependencies could be resolved at config time rather than compile time. This would clean up the makefiles and make everything a lot more sane. It would also make it easier for kernel feature developers to add features without needing to know everything about compiler versions.

Masahiro documented the resulting language additions. It included a bunch of definitions for variables and dependencies, similar to those offered by `make`. In fact, `make` was specifically used as the model for the Kconfig enhancements.

Kees Cook liked the docs, although he preferred using a markup language on the documentation instead of just plain text, but he approved the patch. Randy Dunlap also had a few minor suggestions and approved the patch.

There was no particular discussion – Linus' favorite features tend to be implemented quickly and go into the tree with very few bumps.

Un-inlining for Debugging

Changbin Du from Intel posted a patch for kernel developers only – it would give developers the option of preventing GCC from auto-inlining code. In the C language, inlining a function tells the compiler not to actually call the function when the user invokes it. Instead, the compiler copies the function's code directly to the place that called it. The drawback is that the compiled binary is bulked up with copies of that particular function, but the benefit is that the code runs faster because it doesn't have to jump all the way over to the function for each invocation anymore.

But GCC can also use its own judgment to inline functions that the developer never specified. It's a great feature, which allows GCC to produce faster output in general. However, as a by-product, it makes certain Linux kernel debugging tools less effective, because something like the kernel function tracer will only trace functions that have not been inlined.

Changbin's patch prevented GCC from using its own judgment to inline functions. This meant that a whole lot more functions would be analyzed by the kernel function tracer.

The result was striking. Immediately, the function tracer started finding bugs all over the kernel, in areas it had never been able to test before.

Steven Rostedt was highly impressed by the patch's ability to uncover bugs throughout the entire kernel. Johan Hovold was also very impressed but did notice one warning produced by Changbin's patch that was a false positive. He asked if there was some way to trick the test code into just letting this one case slip through, and Steven suggested modifying the kernel code slightly to stop the warning.

Viresh Kumar took a look at the false positive and remarked, "I am not sure what would [be] the best way to get around this incorrect warning." The specific issue had to do with the way the code allocated memory in one particular spot, but didn't clear out the RAM buffer before making use of it. It also wasn't clear why the false warning showed up for only two of the four occurrences in the affected file.

Johan suggested that fixing the false positive might not be the best idea and that it was "probably best to leave things as they are, and let the GCC folks find a way to handle such false positives." Any other fix in the kernel sources, he said, would be "contrived." Viresh agreed that they might as well just leave the false positive as it was, since it did no harm.

Meanwhile Steven did actually come up with a patch to alleviate the false positive, but Johan felt it was overly complicated and remarked, "should we really be working around GCC this way? If the implementation of this new warning isn't smart enough yet, should it not just be disabled instead?"

The conversation ended there. The interesting part of all of that for me is the attention given to a benign and minor circumstance, just because it didn't fit properly. Changbin's code worked and uncovered bugs all over the place, which presumably all got patched. But a couple of false positives – not kernel bugs at all – inspired a discussion of the best way to allocate memory, possible ways to trick GCC into doing the expected thing, and ultimately the decision to allow GCC to continue to produce the false positive so that the GCC people might find and fix the compiler's own misbehavior. ■■■

OSCAMP

OPEN SOURCE CAMP

on Puppet

MEET THE PEOPLE
BEHIND THE PROJECT!

REGISTER NOW !

See you on NOV 8 2018 in Nuremberg, Germany

opensourcecamp.de

Tweaks for protecting your privacy when surfing with the Firefox browser

Spy Patrol

The Firefox browser is not so private under its default settings, but several add-ons and configuration settings will help you keep the spies in the dark. *By Erik Bärwaldt*

Product advertising already existed in ancient Greece, but it really got rolling in the 19th century with the rise of newspapers, magazines, and other print media. Now in the Internet age, advertising is spreading with an unprecedented intensity, and corporations are trying to track consumer habits and preferences as accurately as possible to assist with their advertising campaigns.

The Mozilla Foundation has strong roots in the open source movement, but through the years, it has derived a big share of its revenue from its affiliation with search engine companies that depend on tracking and analytics. As a result, the default settings for Mozilla's Firefox browser are not particularly private, but if you want to keep the spies away, Firefox offers add-ons and advanced configuration settings that will help you privatize your browser experience.

Mozilla Under Suspicion

Due to its wide distribution, Firefox has numerous plugins that put a stop to spying. Nevertheless, cautious users will want to check the Firefox browser itself and, if necessary, control it manually, because the Mozilla project has also been under suspicion several times.

In July 2017, it was revealed that Mozilla used Google Analytics [1] to spy on users calling `about:addons` in Firefox. Since anti-tracking tools such as Ghostery do not scan locally accessed pages, this contact to Google Analytics from Firefox remained unnoticed for a long time. Mozilla admitted the tracking, but explained that no data would be passed on to third parties and that there were contracts between Google and the Mozilla Foundation.

In the heated discussion about this privacy violation, Mozilla refused to remove the tracker. The developers of the Tor browser, which is based on Firefox, were also surprised by this development, and they have now disabled tracking [2].

Just a few months later, in October 2017, the Foundation was again caught out, this time by the Cliqz add-on, which was auto-

matically added to some Firefox systems without the user's knowledge [3]. The software makes suggestions to the user when entering search terms in the address line, and the manufacturer evaluates the data entered on its servers. Cliqz is a startup that belongs to the Hubert Burda Media group, which is closely linked to commercial data collector emetriq GmbH. Cliqz acquired the US anti-tracking service Ghostery in February 2017.

Disabling the Cliqz add-on does not completely remove the software from Firefox: All recent versions of the browser offer various settings that obviously serve the purpose of using Cliqz services when surfing the web. These are settings that affect the Test Pilot add-on, which developers use to test new experimental features in Firefox. Cliqz is presumably involved in the evaluation of the results.

Plugins

Armed with just a couple of extensions, you can easily block many attempts to spy on your privacy. The most important privacy plugin for Firefox is uBlock Origin, which additionally contains an anti-tracking engine that blocks web bugs,

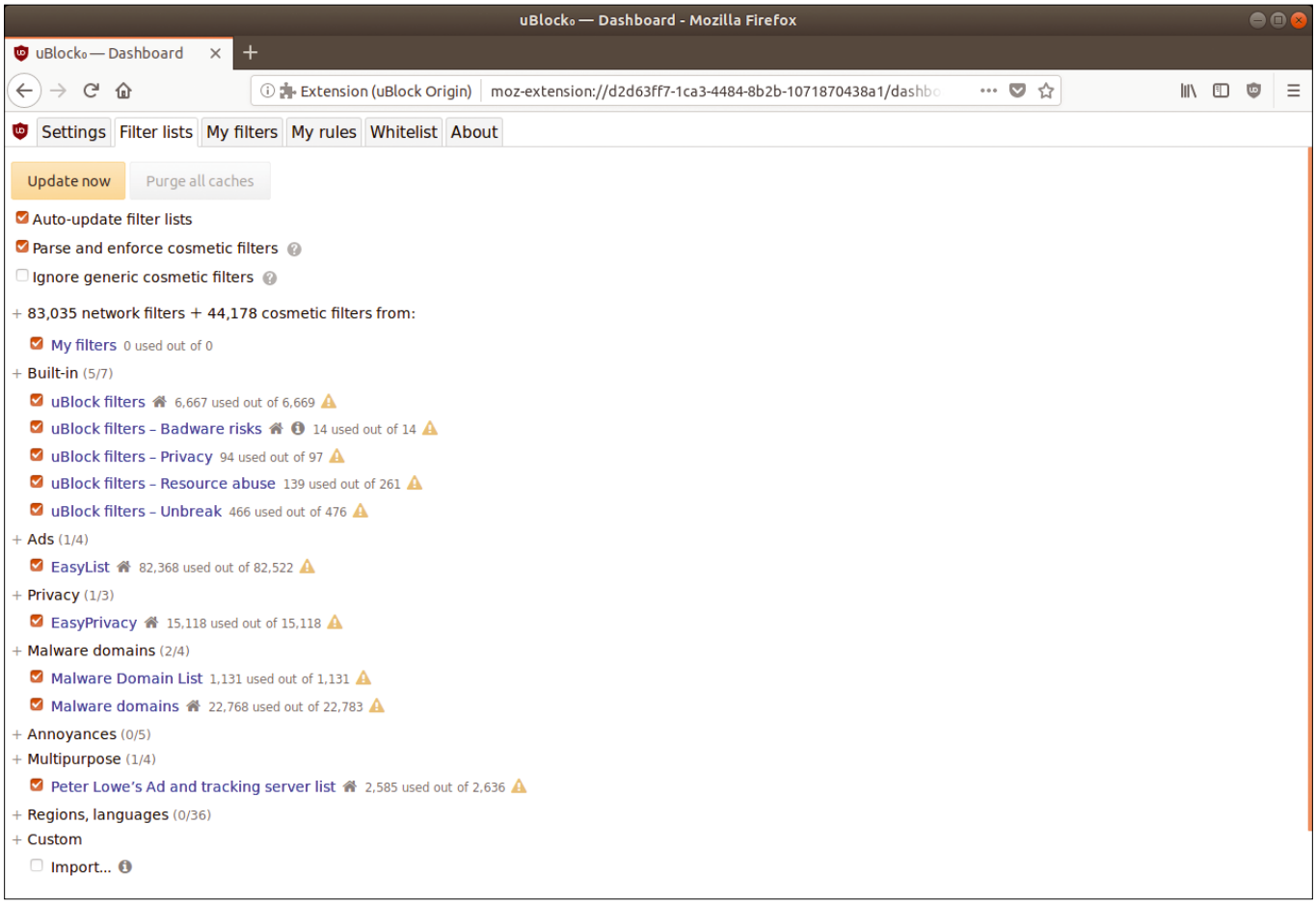


Figure 1: uBlock Origin helps you eliminate many of the spy technologies used on the Internet.

Figure 2: A single new rule helps you prevent retroactive loading of JavaScript from third-party sites.

JavaScript

JavaScript has been one of the core technologies on the Internet for many years. The JavaScript language was developed by Netscape in the mid-1990s was originally intended primarily to add flexibility to HTML content.

Over time, JavaScript has become a serious security risk when used on the Internet – and a formidable tool for commercial data collectors. Many website operators integrate external JavaScript code into the HTML of their pages in order to analyze user behavior and optimize their web presence. The high penetration of such services enables providers to track user behavior across different pages based on specific technical attributes.

If pages deliver advertising via externally integrated JavaScript, as offered by Google services such as DoubleClick, there is the risk of manipulated scripts causing malware to reach the system. Attackers can use modified libraries to steal data or reload code from other domains. So far, only the Subresource Integrity standard [4] offers protection against attacks of this kind, but as of now, hardly anyone has implemented it.

In Firefox, targeted espionage can be limited through some manual work using JavaScript and cookies. It does not matter where the companies gunning for your data reside. However, it is not possible to eliminate all trackers in all cases: Some trackers act through a combination of other spying methods, and a complete deactivation of all possible tracking technologies can block essential functions or interfere with how the pages display.

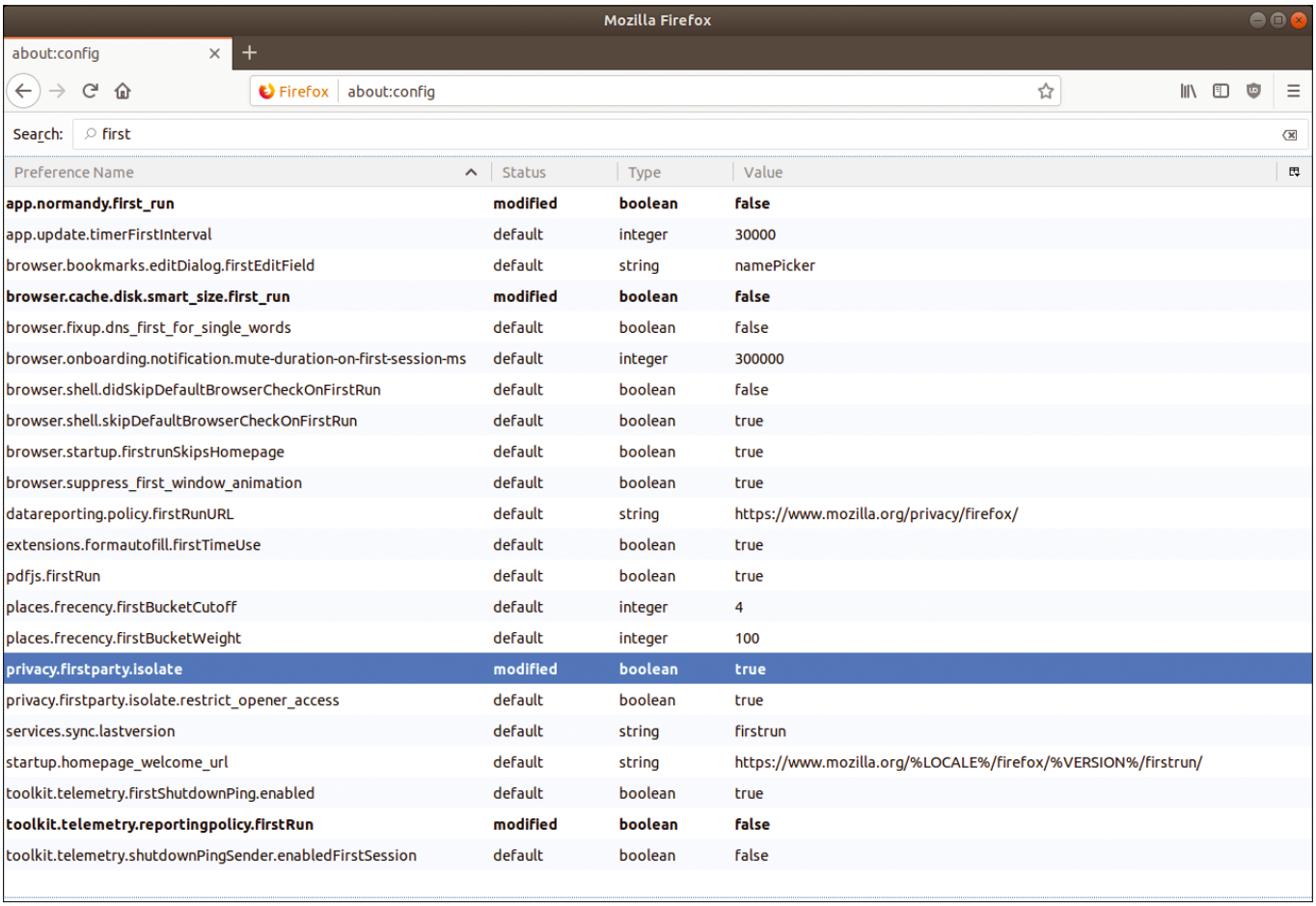
annoying advertising banners, and social sharing buttons. The plugin also saves resources and lets users adjust the filter lists (Figure 1).

uBlock Origin maintains extensive and frequently updated lists that reduce the risk of malware entering the system through manipulated advertising. You can also add your own filters with just a few mouse clicks. For example, you can eliminate unwanted ads in forums that do not reference the preset lists.

One strongly recommended uBlock Origin setting is to restrict loading of JavaScript code to ensure that it only comes from the originally visited page. (See the box entitled "JavaScript.") Open the *My Rules* tab in the plugin's dashboard and enter a line reading `* * 3p-script block` on the right of the *Temporary Rules* window. After saving, transfer this new rule to the *Permanent Rules* window on the left by clicking on the arrow to enable it permanently (Figure 2).

General blocking of all JavaScript libraries using uBlock Origin can cause problems when displaying some web pages. A small plugin named YesScript2 helps you switch the JavaScript filter on and off as necessary: If you install the YesScript2 plugin, an icon appears in the browser toolbar. When you visit a website for which you would like to disable JavaScript for the first time, click on the icon. The plugin will now blacklist the URL and disable all JavaScript elements associated with it.

Another useful add-on that stops content delivery networks (CDN) from loading content on the system is Decentraleyes.



Preference Name	Status	Type	Value
app.normandy.first_run	**modified**	**boolean**	**false**
app.update.timerFirstInterval	default	integer	30000
browser.bookmarks.editDialog.firstEditField	default	string	namePicker
browser.cache.disk.smart_size.first_run	**modified**	**boolean**	**false**
browser.fixup.dns_first_for_single_words	default	boolean	false
browser.onboarding.notification.mute-duration-on-first-session-ms	default	integer	300000
browser.shell.didSkipDefaultBrowserCheckOnFirstRun	default	boolean	false
browser.shell.skipDefaultBrowserCheckOnFirstRun	default	boolean	true
browser.startup.firstrunSkipsHomepage	default	boolean	true
browser.suppress_first_window_animation	default	boolean	true
datareporting.policy.firstRunURL	default	string	https://www.mozilla.org/privacy/firefox/
extensions.formautofill.firstTimeUse	default	boolean	true
pdfjs.firstRun	default	boolean	true
places.frecency.firstBucketCutoff	default	integer	4
places.frecency.firstBucketWeight	default	integer	100
privacy.firstparty.isolate	**modified**	**boolean**	**true**
privacy.firstparty.isolate.restrict_opener_access	default	boolean	true
services.sync.lastversion	default	string	firstrun
startup.homepage_welcome_url	default	string	https://www.mozilla.org/%LOCALE%/firefox/%VERSION%/firstrun/
toolkit.telemetry.firstShutdownPing.enabled	default	boolean	true
toolkit.telemetry.reportingpolicy.firstRun	**modified**	**boolean**	**false**
toolkit.telemetry.shutdownPingSender.enabledFirstSession	default	boolean	false

Figure 3: The First Party Isolation add-on helps you isolate cookies from websites in containers, which makes cross-website tracking more difficult.

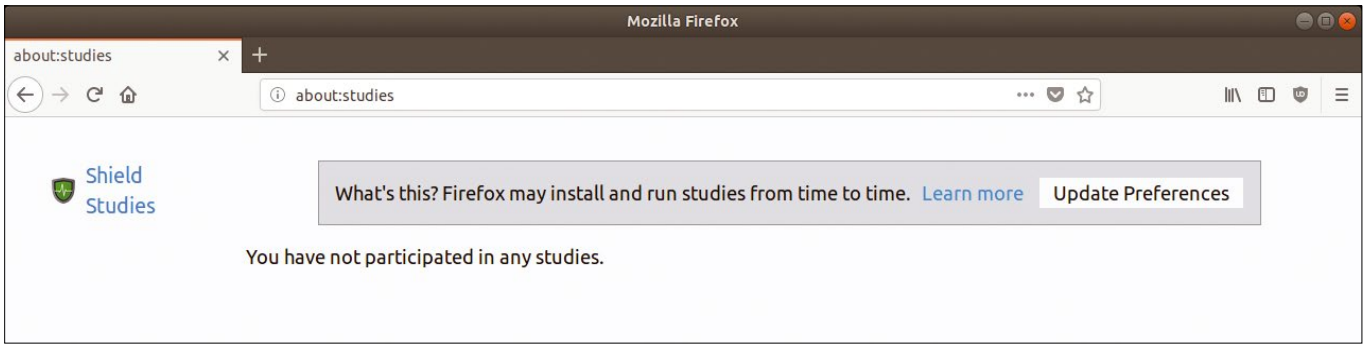


Figure 4: If prefer not to be a guinea pig, you can disable participation in the Firefox Shield Studies.

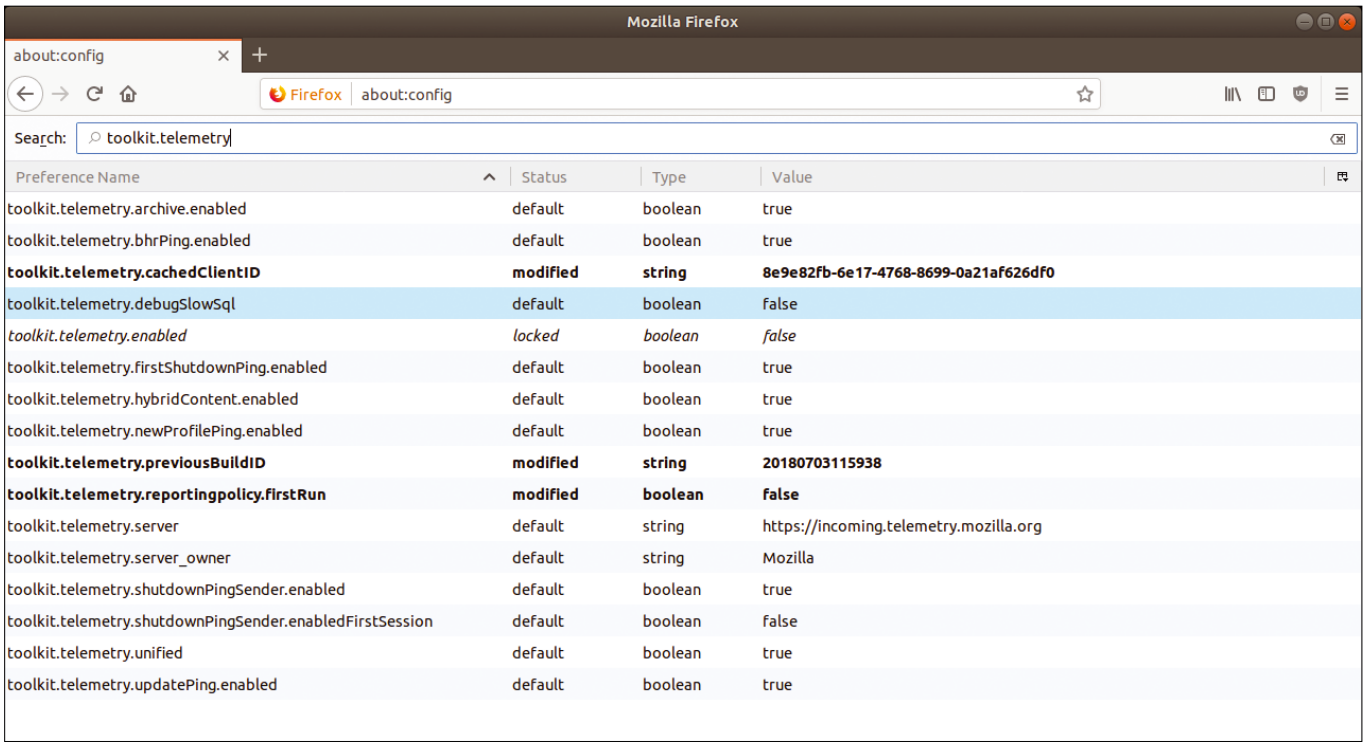


Figure 5: Telemetry data is sometimes used to spy on your behavior while you use the Internet.

CDNs, which are often used to integrate JavaScript libraries into websites, transmit data such as the IP address, screen resolution, browser type, color depth, and operating system version to the server. Decentraleyes intercepts the queries and intervenes to obfuscate the data.

Decentraleyes integrates numerous libraries from Google, Microsoft, Cloudflare, Yandex, Baidu, and others. After downloading from the Mozilla Add-ons page and installing in Firefox, the plugin is ready to use. If the software is installed correctly, you will find a green icon with an eye symbol in the browser toolbar. Since Decentraleyes performs a similar function to uBlock Origin with an individually activated JavaScript blocker, it is not necessary to use the two tools simultaneously.

Cookies

First Party Isolation is a useful plugin that prevents the random storage and reading of cookies, flash cookies, and HPKP supercookies.

First Party Isolation, which was originally developed by the Tor project and is now available at the Mozilla site, uses a container to isolate all data stored locally by a web page, First Party Isolation thus prevents software from reading cookies with a unique ID across several pages. This makes it difficult to identify and track a user on the Internet. The First Party Isolation plugin complements blockers such as uBlock Origin and is suitable for parallel operation.

However, the plugin only works with Firefox browser 58 and later. In older versions, you can achieve the same effect by setting `privacy.firstparty.isolate` to `true` in the configuration (`about:configin` the URL line) (Figure 3).

Table 1: Telemetry Parameters

`toolkit.telemetry.archive.enabled`
`toolkit.telemetry.enabled`
`toolkit.telemetry.unified`
`toolkit.telemetry.bhrPing.enabled`
`toolkit.telemetry.firstShutdownPing.enabled`
`toolkit.telemetry.hybridContent.enabled`
`toolkit.telemetry.newProfilePing.enabled`
`toolkit.telemetry.shutdownPingSender.enabled`
`toolkit.telemetry.updatePing.enabled`

Canvas Fingerprinting

Canvas fingerprinting is becoming increasingly popular as a technique for spying on Internet users. Hidden text on the website makes it possible to uniquely identify individual computers based on differences in the operating system, browser, installed fonts, and graphics adapter.

The Firefox CanvasBlocker add-on prevents canvas fingerprinting. After installation, CanvasBlocker works without any further configuration and blocks fingerprinting by various methods. Using the *Settings* button in the browser's *Advanced* section, you can view the configuration, along with the explanations, and make changes as needed.

Another clue that is often used to track users is the referrer. All common browsers transfer the URL of the original website when switching from one domain to the next. Capturing this URL makes it easy to follow the user's path. Firefox prevents this data from being transferred if you install the Smart Referer add-on.

Studies

Mozilla has introduced what are known as Shield Studies in current Firefox versions; this feature gives the user access to experimental functions installed in the browser, and Mozilla then collects data on the use of the function. Since data collection is not transparent, and Mozilla can import the add-ons into the program without you being prompted, it makes sense to avoid participating in the Shield Studies if you are concerned about privacy.

First call the `about:studies` command in your browser's address bar. A list of the studies activated in your browser appears (Figure 4). If you find any entries, remove them by clicking on the *Remove* button to the right of the add-on.

Then disable participation in the Shield Studies program by typing `about:preferences#privacy` in the address bar or by clicking on the settings in the *Privacy & Security* group in the Options menu. In the dialog, uncheck *Allow Firefox to send technical and interaction data to Mozilla*.

Telemetry

Some parameters can only be configured through Firefox's internal configuration dialog. You therefore need to familiarize yourself with the many options the configuration dialog offers for customizing your browser. To reach this configuration space, enter `about:config` in the address bar.

Firefox provides several telemetry entries – which are pretty much in vogue for many applications and are supposedly designed to improve the product through hidden data collection (see Table 1). Since some of these entries send telemetry pings even if you have explicitly switched off the data transfer to Mozilla, manual intervention is unavoidable.

In the configuration window, enter `toolkit.telemetry`, which gives you an impressive list of entries (Figure 5). In this table, set the entries from the *Telemetry Parameters* table to a value of `false` by double-clicking on the entry. By the way, Mozilla makes changes to the telemetry entries in almost every new Firefox release, so not all entries will be available in every variant of the browser.

Remove the URLs with the telemetry server addresses from the `toolkit.telemetry.infoURL` and `toolkit.telemetry.server` options. If you still find an entry for `toolkit.telemetry.rejected` in the list, set it to a value of `true`.

It is also a good idea to adapt the browser cache to prevent the cached data from being created. Search for all entries with the name `browser.cache` and change the value of the `browser.cache.disk.enable` entry to false. You will also want to set the `browser.cache.offline.enable` and `browser.cache.offline.insecure.enable` options to `false`.

Promotional Film

HTML5 has established a new standard that allows the unsolicited delivery of annoying advertising films and supports spying on the user. Many news portals and online daily newspapers now use this type of harassment, for example, to preface unwanted videos with editorially edited sequences.

Since all common browsers play these videos automatically by default, the user is suddenly confronted with multimedia advertising without taking any action to receive it. The uBlock Origin ad blocker reliably filters out these HTML5 videos but does not prevent automated playback of such content.

To decide for yourself what you want to see and what not, set the `media.autoplay.enabled` entry to false in the configuration dialog of Firefox and its derivatives. This setting ends the forced exposure to commercial broadcasts.

Derivatives

Over the years, many Firefox derivatives have appeared with different goals. In addition to the well-known Tor browser, Pale Moon is another prominent Firefox derivative [5]. You can free these browsers from overly aggressive espionage attempts by manually editing the configuration and installing some special add-ons.

However, not all add-ons offered by Mozilla work with Firefox derivatives, and not all entries will be identical to the equivalent entries in Firefox. For example, Pale Moon offers considerably fewer telemetry entries; however, Pale Moon has enjoyed steadily growing popularity in recent years and is therefore attracting more of its own developers, who are now at work on giving Pale Moon its own collection of extensions.

Conclusions

Although the Mozilla project vociferously defends the user's right to privacy, Firefox sometimes ignores privacy concerns in an effort to extract as much data as possible from users. However, add-ons and configuration settings are available to help you lock down Firefox and minimize privacy concerns. ■■■

Info

[1] Google Analytics Is Used to Track Users: *https://github.com/mozilla/addons-frontend/issues/2785*

[2] Firefox Tracks Users with Google Analytics in the Add-on Setting: *https://news.ycombinator.com/item?id=14753546*

[3] Testing Cliqz in Firefox: *https://news.ycombinator.com/item?id=15421708*

[4] Subresource Integrity standard: *https://infosec.mozilla.org/guidelines/web_security#subresource-integrity*

[5] Pale Moon: *http://www.palemoon.org/*

LIBRECON

powered by CEBIT®

THE REFERENCE EVENT FOR OPEN TECHNOLOGIES IN THE SOUTH OF EUROPE

21+22 Nov · 2018
Bilbao (Spain)

LATEST TRENDS IN OPEN SOURCE APPLIED TO:

INDUSTRY

The so-called fourth industrial revolution is written in open source.

PUBLIC AUTHORITIES

The most transparent administrations are based on open technologies.

FINANCE

The global stock exchanges have begun to adapt their technology to the open source.

JOIN UP to the unique event powered by CEBIT in the South of Europe

ORGANIZED BY:

[esle] ASOLIF

POWERED BY:

CEBIT® Deutsche Messe

FURTHER INFORMATION:

www.librecon.io

info@librecon.io
twitter.com/librecon
facebook.com/librecon
es.linkedin.com/in/librecon

Small tricks can have a big impact on your privacy

Quiet Time

You don't have to dig deep into your toolbox to protect your privacy: With a few simple tricks, you can disable the webcam and microphone and permanently delete data from your hard disk. *By Christoph Langner*

Whether you do your business with Microsoft, Apple, Google, or Facebook, large IT companies are eager to collect any information that you happen to toss their way. It is not for charity that corporations operate their own webmailers, search engines, network storage, and online communities: The data from these services can deliver highly-targeted advertising worth billions of dollars.

But it isn't just the big high-profile companies that are testing the limits of user privacy. For example, a smartphone app by a Spanish pay TV provider secretly enabled the GPS function and the microphone of the device and transmitted data (in a poorly anonymized form) to the company's servers. The TV station wanted to identify football bars that broadcast the game without paying royalties, and they enlisted thousands of unwitting football fans as involuntary undercover spooks [1].

Today's devices and Internet services are quite complicated, and unless you wrote the software yourself, you can never be totally sure exactly what it is doing.

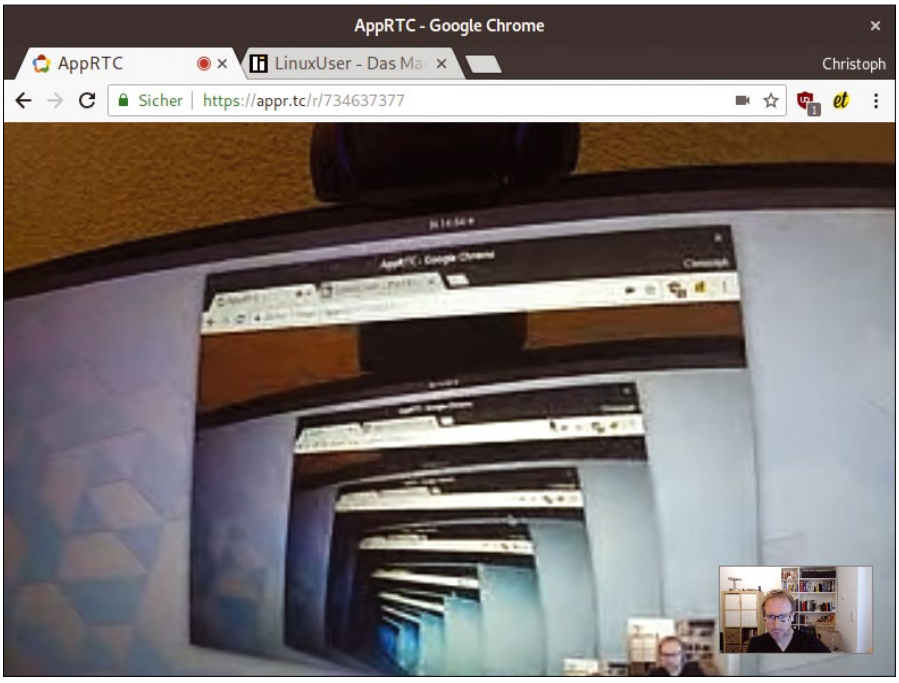


Figure 1: Thanks to WebRTC, Skype is no longer necessary: HTML can access a computer's webcam, which has led many users to protect their privacy by covering the lens.

This article introduces you to some small hacks that you can use to secure your online privacy in just a few easy steps without having to do anything drastic like encrypting you hard drive (which, by the way, only protects you if your computer is not running).

Disabling the Webcam and Microphone

A computer's webcam and microphone are often abused for attacks and privacy violations. Thanks to modern web technology such as HTML5 with WebRTC [2], a browser is all it takes to transfer the image and sound from your living room to the web. For example, video chats with AppRTC [3] can be handled directly in the browser (Figure 1); a web page could also tap the webcam for other purposes.

Although a browser requires the user to confirm that the website is allowed to enable the webcam and microphone, errors (on the part of developers, as well as users) occur from time and time again. And with a locally installed application – installed voluntarily or by a trojan – you might not even be prompted to confirm. Apart from a small light in the bezel, there is usually nothing to indicate that the device is recording. With some devices, the program can even turn off the webcam LED.

With a classic desktop PC, you can usually simply unplug the camera and microphone to stop the possibility of video spying.

Listing 1: Disabling uvcvideo

```
01 $ sudo modprobe -r uvcvideo
02 modprobe: FATAL: Module uvcvideo is in use.
03 $ sudo rmmod -f uvcvideo
04 $ sudo modprobe uvcvideo
```

This solution is not available for portable systems with integrated input devices. Users with laptops and smartphones therefore often apply stickers to the webcam and the internal microphones. Even Facebook founder Marc Zuckerberg demonstrated this practice [4] (but probably by mistake).

Alternatively, you can disable the webcam in the system settings – so that programs can no longer call it. On Linux, you have to disable the `uvcvideo` kernel module, which is normally loaded automatically at boot time. In principle, you can do this manually (Listing 1, line 1), but very often, some program will have claimed the module (line 2), forcing you to go for the heavy artillery (line 3). Applications such as Cheese will then no longer find a webcam on the system (Figure 2). If necessary, you can reload the module later (line 4). You will need administrative privileges on the system for all of these commands.

To prevent the Linux system from loading the kernel module responsible for the webcam, add it to the `/etc/modprobe` configuration file with the `blacklist` option in the `/etc/modprobe.d/blacklist.conf` configuration file (Listing 2). If

Listing 2: Blacklisting

```
# /etc/modprobe.d/blacklist.conf
# Webcam Disabled
blacklist uvcvideo
```

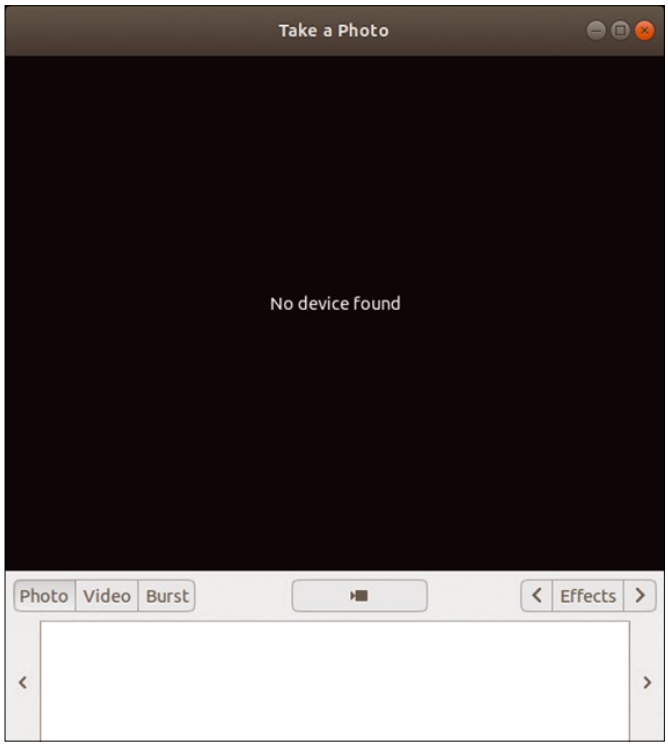


Figure 2: If the kernel driver for the webcam is missing, no image from the camera is displayed. You can save yourself the workaround of an ugly sticker on the lens.

Listing 3: Disable the Mic?

```
$ cat /proc/asound/modules
 0 snd_hda_intel
 1 snd_usb_audio
 2 snd_usb_audio
 3 snd_usb_audio
```

the file does not yet exist on your computer, simply create it. Theoretically, you are free to choose the name, such as `disable-webcam.conf`; the only important thing is the file extension `.conf`. After a restart, the webcam should not work, which you can test with Cheese or Skype. If necessary, load the kernel module manually, as shown in the last line of Listing 1.

Things are a little different with a microphone built into the device. Theoretically, as with a webcam, you need to disable the necessary kernel module, which you can determine quite easily using `cat/proc/asound/modules` (Listing 3). However, switching off `snd_hda_intel` not only takes down the microphone, but also the entire internal sound card. In this case, therefore, you need to compromise between sound or perfect privacy. If necessary, you could still connect a USB headset (even with an integrated microphone) – these devices use the `snd_usb_audio` kernel module.

Alternatively, you should also scan the BIOS or UEFI for an option to disable the integrated devices. Working at the BIOS or UEFI level gives you even more security: Theoretically, a blacklist created through the operating system can be reversed by software running with administrative privileges.

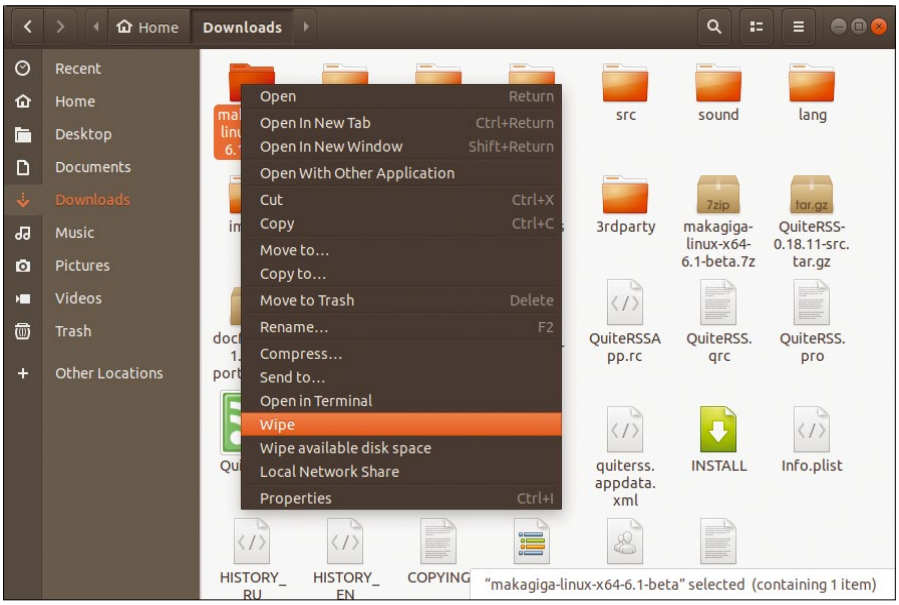


Figure 3: Nautilus Wipe is an extension that securely deletes files in Gnome Files.

Secure Deletion on SSDs

For the user, flash memory devices such as SSDs or USB sticks are used very much like classic hard disks. Under the hood, however, they work completely differently, since they depend on memory modules rather than mechanical read/write heads and rotating disks. This has certain consequences when deleting data. Unlike a hard disk, an SSD or other flash memory usually does not allow you to instruct the controller to erase a specific area of the mass memory: It tries to distribute all write operations as evenly as possible over all blocks of the device [10]. Only expensive SSDs designed for particularly critical tasks have the necessary deletion algorithms.

However, modern SSDs now offer a special function to reset the data carrier, including all reserve blocks, to the as-delivered state. Other SSDs automatically encrypt the stored data. For deletion, it is then sufficient to simply dispose of the key instead of getting rid of each individual block. This speeds up the process and extends the life of the drive. Some experts argue there is no reliable way to securely delete individual files without resetting the entire data carrier [11]. You will therefore either want to store critical data in encrypted containers on flash memory or encrypt the entire data carrier.

Securely Deleting Files

The old adage "gone is gone" is true in a buffet line, but not for computers. If you delete a file using the file manager, it does not exactly disappear. In most cases, it falls into the recycle bin, from which it can be quickly restored. But even if you empty the trash, the supposedly deleted data can still be reconstructed.

How much effort it takes to truly delete a file depends on how much data was written onto the medium and which filesystem is used. On FAT partitions, only the reference to the location of the file is actually removed during the so-called "deletion." The data, therefore, will be lost to the operating system, but it will still exist on the data carrier.

If you want to share hard disks, memory cards, or USB sticks with third parties, you need to delete the private data stored on them securely. Linux supports two console tools, `shred` (often installed on the system by default via the *coreutils* package) and `Wipe` – both of which securely delete files, directories, or entire partitions. In practice, however, you will rarely want to launch a terminal to wipe a file off your hard disk.

Alternatively, file managers can be equipped with similar functions. For example, the Gnome Files manager (formerly

Figure 4: As a rule, it is sufficient to overwrite the data twice.

Nautilus) has Nautilus Wipe [5], an extension that lets you securely delete files with a single mouse click (Figure 3). Nautilus Wipe is often used with a privacy- and security-conscious Live distribution such as Tails [6].

After installing the package (Ubuntu calls the package *nautilus-wipe*, and Arch has the same extension with the same name in the AUR), restart the file manager by running the `nautilus -q` command . Two new options are then available in the context menus of files and folders. The first (*Secure Delete*) overwrites the objects selected in the file manager with random data. The application lets you choose the number of deletion passes (Figure 4). Two passes are typically sufficient; further repetitions do not improve security.

After that you should also use the second option, *Secure deletion of available disk space*, which overwrites the space marked as free on the partition and ensures that backup copies and shadow files of the previously edited documents are securely overwritten and thus permanently deleted. See the "Secure Deletion on SSDs" box for more information on deleting data stored on flash memory devices.

Canonical Phone Home

In Ubuntu 18.04, Canonical introduced *Ubuntu Welcome*, a feature that sends a whole bundle of hardware and metadata to the manufacturer the first time the user logs onto the system [7]. This data includes information about the Ubuntu version, the

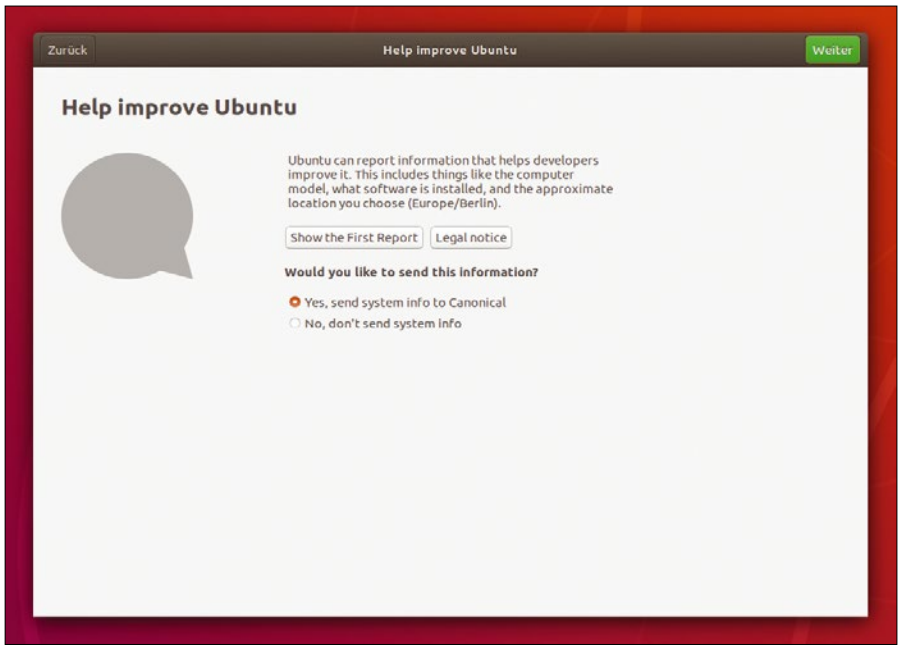


Figure 5: Since Ubuntu 18.04, Canonical has tried to collect telemetry data from users. To find out what is transmitted, click on *Show the First Report*.

Listing 4: Removing Amazon Icon

```
$ sudo apt remove ubuntu-web-launchers
$ sudo apt remove whoopsie
```

computer's hardware equipment (CPU, GPU, RAM, screens), the location (based on the locale settings chosen during the install), and a number of other settings. You can view this data by clicking on the *Show the First Report* button. By switching to *No, don't send system info*, you can prevent the system from sending any data to Canonical (Figure 5).

By switching to the Gnome desktop, Canonical has not only abandoned its own Unity desktop, but also an additional source of revenue that has caused the company much trouble in the past. In Ubuntu 12.10, developers integrated a shopping function that displayed goods and media from the Amazon catalog in the Unity dashboard to match the input. A click on one of the hits took the user to the Amazon portal, and after a purchase, commission was paid to Ubuntu's vendor. In addition, the developers initially implemented the feature so that both Canonical and Amazon were aware of all input in the Dash.

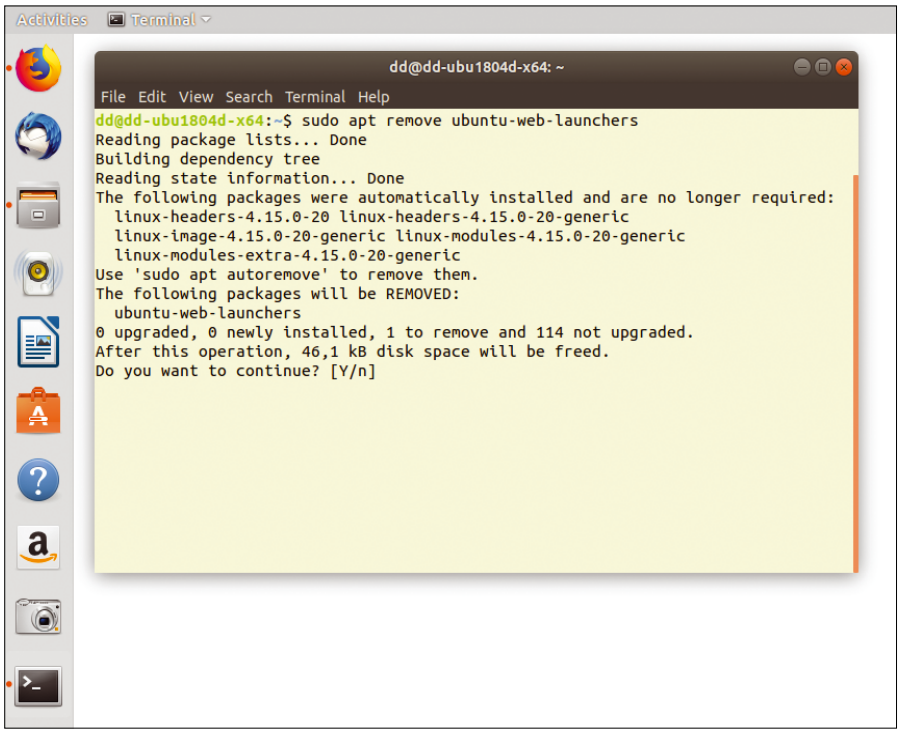


Figure 6: You can completely delete Amazon affiliate links and the often verbose bug reporting from the Ubuntu system.

In Ubuntu 18.04, only an Amazon starter icon reminds users of this past feature (see Amazon icon on the left in Figure 6). A click on the icon loads the `/usr/share/ubuntu-web-launchers/amazon-launcher` script, which determines the location of the user via the public IP address and opens the country-specific Amazon page in the browser. It attaches an affiliate tag to the link, so that a share of the sales is assigned to Canonical if you make a purchase. Private data is not routed via the web; similar links are also used by numerous website operators and bloggers on the Internet (but most of them are clearly marked as ads).

You can simply delete the icon from the sidebar using the context menu and the *Remove from Favorites* option. The Amazon entry will still appear in the application menu. But if you uninstall the *ubuntu-web-launchers package* with the package manager (Listing 4, first line), the Amazon icons disappear completely from the system, also for all user accounts created on the system.

Another feature that many users disable due to privacy concerns is the automatic submission of crash reports to Canonical. As a rule, such reports do not contain any personal data, but you can't really be sure [8]. Therefore, check whether automatic *Problem reporting* is enabled below *Settings* | *Privacy*. If in doubt, remove the *whoopsie* package to throw the background service off the system (Listing 4, last line).

Conclusions

Every user has different priorities when it comes to protecting their privacy. Some people are happy with a browser that does not store cookies and does not create a history; others will not even use a smartphone or will only access the Internet via Tor. See the Privacytools.io website for a very good overview of Internet monitoring and privacy concerns [9]. The site focuses on open source applications and services that are committed to protecting the privacy of users, including recommendations for webmail providers, VPN service providers, search engines, email programs, instant messengers, and audio and video messaging clients. The Privacytools.io website also explains some reasons why privacy-conscious users should avoid Windows 10. ■■■

Info

[1] "Spanish Football League Defends Phone Spying": *https://www.bbc.co.uk/news/technology-44453382*

[2] WebRTC: *https://webrtc.org*

[3] AppRTC video chat client: *https://github.com/webrtc/apprtc*

[4] "Marc Zuckerberg Tapes over His Webcam: Should You?": *https://www.theguardian.com/technology/2016/jun/22/mark-zuckerberg-tape-webcam-microphone-facebook*

[5] Wipe Tools: *http://wipetools.tuxfamily.org*

[6] Tails: *https://tails.boum.org*

[7] "More diagnostics data from desktop": *https://lists.ubuntu.com/archives/ubuntu-devel/2018-February/040139.html*

[8] Ubuntu's Data Privacy statement: *https://www.ubuntu.com/legal/dataprivacy*

[9] Privacytools.io *https://www.privacytools.io/*

[10] Secure Deletion on SSDs: *https://en.wikipedia.org/wiki/Solid-state_drive#Data_recovery_and_secure_deletion*

[11] "Reliably Erasing Data from Flash-Based Solid State Drives": *https://www.usenix.org/legacy/events/fast11/tech/full_papers/Wei.pdf*

Register Today

Register before October 15 and save.

sc18.supercomputing.org

Discover a broad program of HPC-focused technical presentations, papers, workshops, informative tutorials, timely research posters, and more.

Explore exhibits showcasing the latest HPC technologies from the world's leading vendors, universities, and research organizations.

Enjoy several days in Dallas, one of the country's top tech cities, and home to great amenities, food, and nightlife. Bring the family!

November 11–16, 2018

The International Conference for High Performance Computing, Networking, Storage, and Analysis

SC18

Dallas, TX | hpc inspires.

Sponsored by: IEEE computer society | TCHPC | ACM | sighpc

Encrypt cloud data easily and securely with Cryptomator

Bottle It

Cloud storage services help with data synchronization across multiple computers, but they do not usually provide encryption. Cryptomator adds encryption to the cloud storage environment. *By Christoph Langner*

The IT world is subject to fashions, trends, and tendencies, such as the current hype surrounding blockchain technology. For example, by changing its name to Long Blockchain Corp., the US beverage manufacturer Long Island Iced Tea Corp. briefly increased its share price 289% [1].

One of the biggest trends in recent years is cloud storage. What used to be a pig in a poke suddenly became "the cloud" overnight. Everything has to be stored in the ominous cloud: applications, services, data. Cloud storage is practical for the user at first glance. You don't have to worry about the hardware, the provider takes care of security and backups, and you have immediate access from any device.

At second glance, however, many cloud users have concerns: Who can access the uploaded data? What data does the provider evaluate for advertising purposes? Do security agencies have access? In the end, anyone giving away personal data must always expect third parties to gain insight into their own digital life.

The problem could be solved if cloud storage services would let users protect the uploaded data with a personal key. However, very few storage providers offer this function. The list does not include the most popular services, such as Google Drive or Dropbox. As a user, you have to take care of your own privacy.

Applying conventional encryption techniques to the cloud storage process is typically inefficient and requires extra steps. For example, opening containers encrypted with the LUKS Linux standard on an Android smartphone involves a great deal of effort. Users who do not want to worry about such details can turn to Cryptomator [2].

Cryptomator calls itself "free client-side encryption for your cloud files." The open source program uses a 256-bit AES key and a MAC master key to encrypt data. When generating the keys, Cryptomator uses Scrypt technology, which makes brute-force attacks more difficult. (Scrypt is an approach to generating keys that uses a random value and a password to reduce the possibility of a successful dictionary attack.) A deliberately simple interface makes it easy to create and integrate the encrypted containers that Cryptomator refers to as "vaults."

Installation

In addition to the Cryptomator encryption tool, you need a cloud storage client and an account with the service. Which provider you choose is up to you – in theory, services that you integrate into the system via SSH or WebDAV are also suitable. It makes sense to install and set up these services before Cryptomator, as you will later be creating a vault in the cloud memory.

You can download the currently supported version of Cryptomator from the project homepage (see also the "Cryptomator Beta" box). The developers make a request for a donation before the download, but you can dismiss the request with a click. The program is available as a DEB package for 32- or 64-bit Debian/Ubuntu systems, or there are RPM packages for openSUSE and Fedora. In addition, the project maintains a PPA package source for Ubuntu users, which you can use to automatically install the program and keep it up to date (Listing 1). On the Arch system used for this test, the program is located in the Arch User Repository.

During the install, the package manager automatically adds a Java runtime engine to the system. The Java basis of the application makes it easier for developers to port

Cryptomator Beta

You can pick up the current beta version of Cryptomator from the project's GitHub repository [7]. The easiest way to set it up is to use the program's AppImage: All you have to do is download the AppImage file and make it executable. Then start the beta by double-clicking in the file manager. Since the AppImage comes with all necessary dependencies, you do not need to install a Java engine or other libraries.

Listing 1: Installing on Ubuntu

```
$ sudo add-apt-repository ppa:sebastian-stenzel/cryptomator
$ sudo apt update
$ sudo apt install cryptomator
```

the software to other operating systems. Cryptomator is therefore also available for Mac OS X and Microsoft Windows.

Creating a Vault

The Cryptomator application window is limited to a few widgets at first start. As your first step, create a vault by clicking on the plus icon bottom left and selecting the *Create vault* option (Figure 1). In the following dialog, you name the new vault and define a location for it. This should be in the path synchronized by the sync client, for example in `~/Dropbox/` for dropboxes. Cryptomator automatically creates a subdirectory with the name of the vault.

Then select the vault in the list and assign a password. A scale from very weak to very strong indicates whether the password you chose is a good choice. Click on *Create vault* to complete the configuration of the vault. To work with the vault, you now need to integrate it into the system. Select the vault entry from the page list and enter the previously assigned password (Figure 2).

Pressing the *More options* button gives you the possibility to change the name of the virtual drive. Two buttons, *Save password* and *Auto-Unlock on Start (Experimental)*, are grayed in Linux: Although they are among the new features introduced in Cryptomator 1.3, they have not yet been implemented under Linux (not even in the first beta of Cryptomator 1.4) [3].

Also via FUSE

After unlocking the vault using the *Unlock vault* button, the file manager opens with the data encrypted in the container.

Figure 1: The Cryptomator interface contains just a few elements. Use the plus icon to create new vaults or integrate existing crypto containers.

Cryptomator 1.3 exclusively uses the WebDAV protocol to communicate with the service running in the background, which handles the encryption. The URL in the file manager therefore follows the pattern `dav://localhost:42427/ID/Name`. Nautilus shows you the address when you press Ctrl + L to display the address bar (Figure 3).

Cryptomator 1.4 sees the developers taking a new path with support for Filesystem in Userspace (FUSE). (FUSE is a kernel module that shifts the filesystem drivers from kernel mode to user mode, which allows users without admin rights to mount filesystems.) Instead of a network protocol, the opened vault is directly integrated into the data structure. Use the gear icon to switch drive integration from *WebDAV* to *Fuse* in the application settings. Cryptomator then mounts the opened vault in `~/.Cryptomator/Name`, or, if so desired, you can choose another directory.

In the vault's "raw data," you can only see the master key of the application (and a backup), and the directories `m/` and `d/` are visible. Cryptomator uses the first directory to save metadata and the second for the actual data. The encryption algorithm used by Cryptomator divides the vault into several files: This approach prevents conclusions

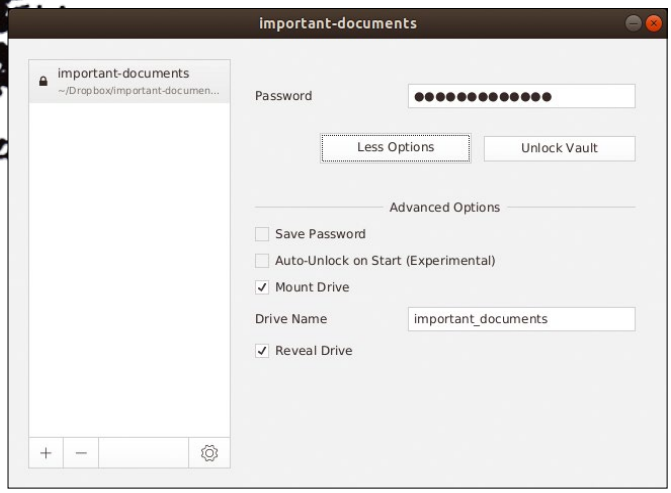


Figure 2: Entering the previously assigned password.

about the original directory structure, the unencrypted file names, and original file sizes. For the cloud storage sync client, however, it makes no difference whether it backs up data in plain text or encrypted data.

You can still see the vault via the provider's web front end, but you cannot unlock it or view or change the stored data (Figure 4). However, you do not have to do without the encrypted data on the road. The Cryptomator project has apps for Android and Apple devices on Google Play [4] and iTunes [5]. In contrast to the desktop programs for Linux, Mac OS X, and Windows, the apps cost EUR4.99, and the source code is not open [6].

Share with Key Only

One of the biggest advantages of cloud storage is the ability to easily share data with friends, co-workers, or customers. It is usually sufficient to share a file or directory via the file manager's context menu or the service's web front end and then send a link via email or instant message. With Cryptomator, sharing the data requires an additional step.

As usual, you load the data you wish to share into the encrypted vault. Then enable the sharing function and send the link to the recipient. In addition, you communicate the secret Cryptomator key, preferably using an encrypted email or a traditional one-to-one conversation. The recipient, who also needs Cryptomator, then mounts the existing vault directly using the *Open vault* function (below the plus icon).

You can only share the whole vault – no access controls exist for sharing individual files. If you want to send data to different partners, create a separate vault for each of them.

Conclusions

Cryptomator fills a gap that has been difficult to close for many years: It offers an easy-to-use, yet very secure approach to encrypting important data without you having to change your working methods or sacrifice flexibility. You can still access all data encrypted in the Cryptomator vault at any time and from any device. The only way to access the crypto containers is via the web front end.

Note, however, that an encrypted container does not replace a completely encrypted system. For example, if you create a LibreOffice document and move it to a vault, the intermediate versions and shadow copies saved by the office package will leave numerous traces of the document on the filesystem. Even though a vault does not necessarily have to reside in cloud storage, Cryptomator is not unjustified in referring to itself as an encryption tool for the cloud. ■■■

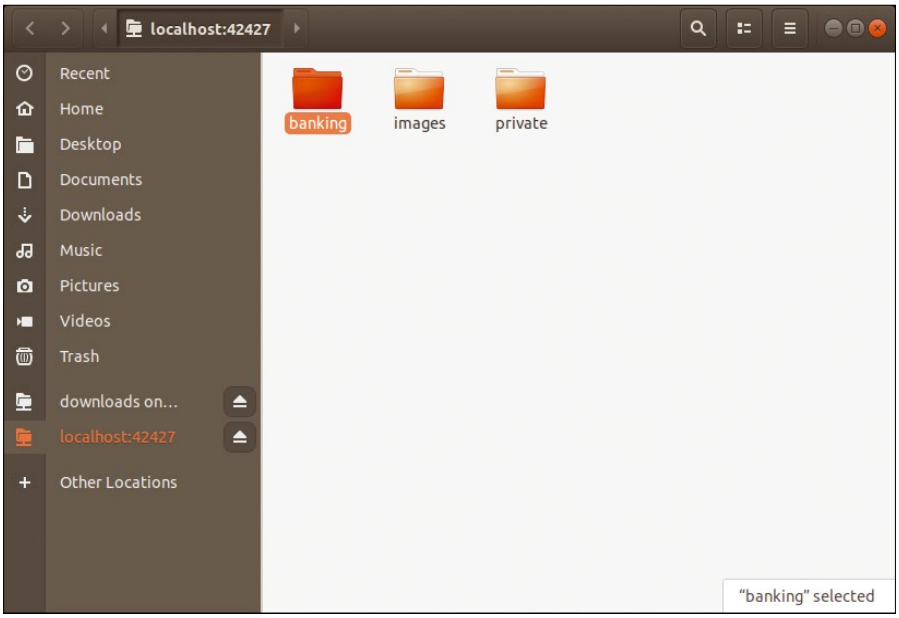


Figure 3: After unlocking the vault, you do not need to change your habits: The encrypted data is shown on the system like any normal directory.

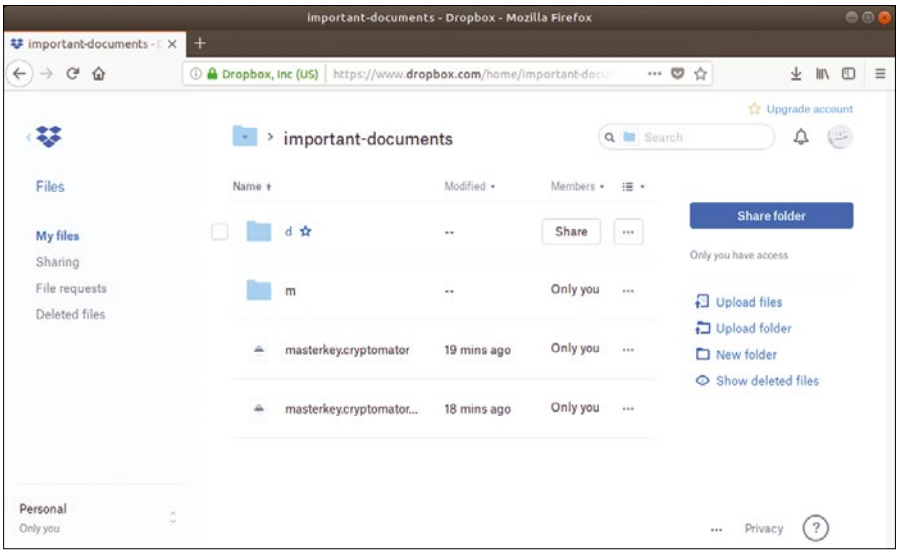


Figure 4: The cloud storage provider – and thus also the web front end of the service (here Dropbox) – cannot do anything with the encrypted data.

Info

[1] "Long Island Ice Tea Sours after Changing Its Name to Long Blockchain Corp.": *https://www.bloomberg.com/news/articles/2017-12-21/crypto-craze-sees-long-island-iced-tea-rename-as-long-blockchain*

[2] Cryptomator: *https://cryptomator.org*

[3] "Implement save password working on Linux": *https://github.com/cryptomator/cryptomator/issues/422*

[4] Cryptomator for Android: *https://play.google.com/store/apps/details?id=org.cryptomator*

[5] Cryptomator for iOS: *https://itunes.apple.com/app/cryptomator/id953086535?mt=8*

[6] Cryptomator for Android, no open source: *https://github.com/cryptomator/cryptomator-android/issues/1*

[7] GitHub repository: *https://github.com/cryptomator/cryptomator/tags*

THE OFFICIAL CONFERENCE OF THE APACHE SOFTWARE FOUNDATION

Learn about the latest Open Source innovations in Big Data, Cloud, Finance, Geospace, IoT, Machine Learning, Search, Servers, and more in a collaborative, vendor-neutral environment

100+ presentations on dozens of popular Apache projects, including Avro, Beam, CouchDB, Drill, Fineract, Groovy, Hadoop, HAWQ, HBase, Hive, HTTP Server, Kafka, Karaf, Lucene, MyNewt, NetBeans, OpenNLP, ORC, Parquet, Phoenix, Ranger, RocketMQ, Solr, Spark, Struts, Unomi, YARN, and Zeppelin. PLUS innovations in the Apache Incubator, BarCamp and more.

APACHECON North America

September 24-27, 2018
Montréal, Canada

KEYNOTE SPEAKERS

CLIFF SCHMIDT

MYRLE KRANTZ

BRIDGET KROMHOUT

EUAN MCLEOD

REGISTER TODAY!
http://apachecon.com/

We review six RSS readers

Needle in a News Stack

RSS feed readers bring order and clarity to the jungle of new news. *By Erik Bärwaldt*

In the face of ubiquitous ad trackers, a lack of transparency, and the abundance of spam in Facebook and Twitter news timelines, *Wired* magazine recently praised the virtues of Rich Site Summary (RSS) feeds [1] and even predicted that they will make a comeback.

RSS is an XML-based format used by blogs, news sites, and other web content providers to publish news posts in a machine-readable form. A client application, called a news aggregator or *RSS reader*, lets the user subscribe to various RSS sources and assemble the incoming stories into a single customized news stream.

RSS is still alive and well, although it has lost some attention recently with the arrival of modern-day social media platforms. If you want to define your own diverse news sources without depending on Facebook or another social media engine, the time is right for exploring the rich array of content choices available through RSS.

A well-designed RSS reader can manage thousands of messages and offer sophisticated search and filter functions. Several powerful RSS clients are available for Linux. In this article, I examine Akregator [2], Canto [3], FeedReader [4], Liferea [5], QuiteRSS [6], and Makagiga [7]. (See the "Not Considered" box for a summary of some other classic RSS readers for the Linux space.)

Not Considered

Linux has been home to several other RSS readers through the years, and some are no longer in development. PenguinTV [8], for instance, which specialized in multimedia content, was developed for Gnome 2.x. RSSOwl [9] is still quite well known today, although it was last updated in late 2013. BlogBridge [10], BottomFeeder [11], AmphetaDesk [12], and the terminal application Snownews have not received comprehensive maintenance for years.

Akregator

Akregator [2] has been the standard application for many generations for displaying RSS feeds on the KDE desktop. As part of the Kontact suite, the software replays Atom and RSS feeds, either in a purely text-based approach or through the KHTML engine. It optionally displays content in an external browser. Thanks to a modern GUI with tabs, Akregator also allows several news sources to be opened simultaneously. The program copes with the common, but sometimes incompatible, RSS formats of versions 0.9, 1.0, and 2.0 [13]. The software user interface is largely self-explanatory and requires little training (Figure 1).

Akregator lets you generate archives in folders that you create yourself to group important message sources. To do this, right-click at top left in the main window on the *All Feeds* root folder in the tree view with the message sources and then select *New Folder* in the context menu. Once you have entered a name and pressed *OK* to confirm, the new folder appears on the left side of the tree view. You also use this approach to create hierarchical structures by adding subfolders to newly created folders. The topical focus of the feeds is highly granular.

Lead Image Photo by Jenelle Ball on Unsplash

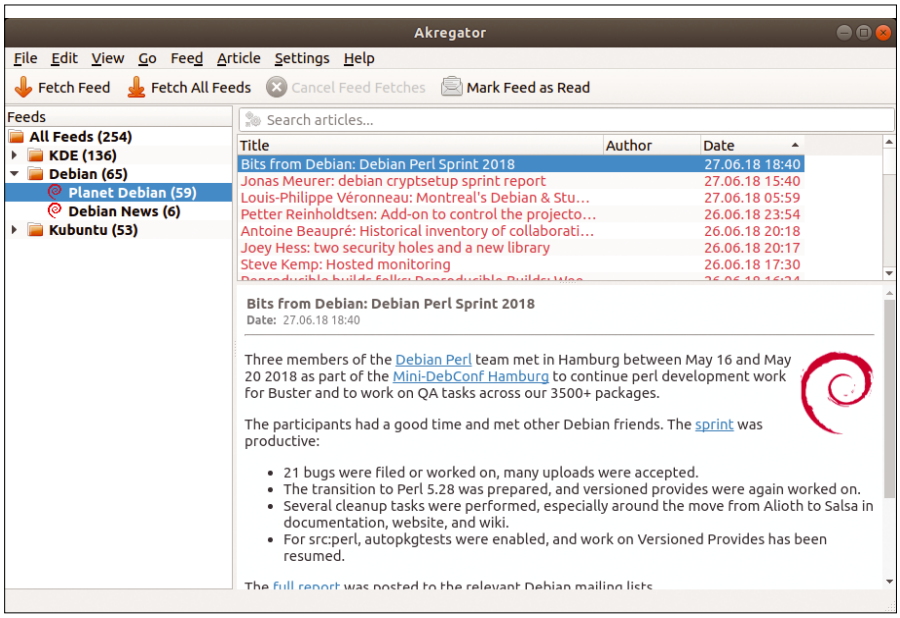


Figure 1: The Akregator program window is unspectacular.

Depending on how regularly the feed providers update and the number of feeds to which you have subscribed, thousands of messages can quickly accumulate and clutter the overview. Akregator therefore offers a delete option under *Settings* | *Set up Akregator* | *Archive*, which limits the capacity of the individual folders to a size specified by the user. You globally determine how many messages each archive stores and how long the messages are kept.

You can have different archiving settings for the individual RSS feeds. To do this, right-click on the desired feed and select the *Edit News Sources* option. In the settings window that opens, select the *Archive* tab and define the desired value in the corresponding dialog (Figure 2).

Akregator does not archive single messages separately but lets the user mark them as important with a context menu option. The software then uses an envelope with an exclamation mark to visually highlight these messages when the corresponding feed is called.

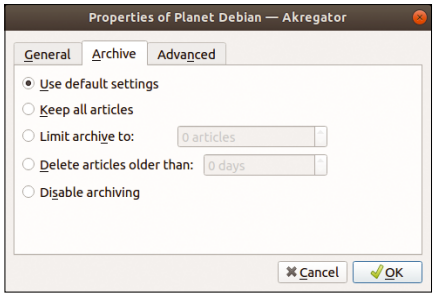


Figure 2: Users can remove obsolete messages in Akregator by using appropriate defaults.

If so desired, you can bookmark the message URL permanently in an external web browser. To do this, select the *Copy Link Address* option in the message context menu to copy the URL to the clipboard. Then insert it into the web browser's Bookmarks menu. Alternatively, messages can be displayed in an external browser using the *Open in External Browser* option in the context menu. You call this option in the Article menu.

Akregator also has its own HTML rendering engine that displays the selected post in a separate tab with graphics and images. This built-in browser can be accessed via *Articles* | *Open in tab*.

Akregator offers two search functions with which users can search individual articles using keywords or sort messages on the basis of terms. You enter the corresponding search terms in an input line directly above the article list on the right side of the program window. While you are typing, Akregator scans the news items in the currently opened source for articles containing the search term. The message list therefore changes permanently while you are entering the search term.

When you press Enter, the software opens the first message in which the search term appears in a new tab. Users can enable more tabs by right-clicking the desired messages and then selecting the *Open in new tab* option from the context menu.

To search for a term in an open message, you need to use a somewhat hidden text search tool accessed by pressing Ctrl + F. Much like other KDE programs or Firefox, it draws a horizontal input and control bar at the bottom of the screen, in which you can enter a search term. By clicking *Next* and *Back*, the search tool jumps to the individual color-highlighted matches in the text. The *Settings* button also lets you perform a case-sensitive search.

Canto

Canto [3] is a feed reader written in Python without a graphical interface. The program is available from the software repositories of the larger Linux distributions, but the source code is also maintained on the project website.

Once you have successfully installed it, you can call the software with the `canto` command in a terminal. If, on the other hand, you go for the newest version that does not yet exist in most repositories, the command is `canto-curses`. Canto subscribes to RSS and Atom feeds, but it can also import OPML files [14] from other readers or export them to this standard format. Canto also supports feed subscriptions with authentication.

Canto's developers completely changed the user interface, except for navigating the feeds, when upgrading from version 0.7.x to the new 0.9.x. Users of older versions can no longer easily operate the new versions.

To work with Canto in the versions from the distribution repositories, you need to store a configuration file named `conf.py` in the `~/.canto/` directory. The easiest way to do this is to copy the existing `conf.py` sample file, for example, using

```
cp conf.py.example conf.py
```

then open the new configuration file and add one or more feed addresses (Figure 3). You can also add display filters here to hide messages that you have read. In the configuration file, you can also define third-party programs to play multimedia content embedded in feeds. Canto also imports OPML lists – if available – with feed addresses from other RSS readers. The `canto -i <filename>` command helps here.

Figure 3: Users can add Canto functions with just a few lines.

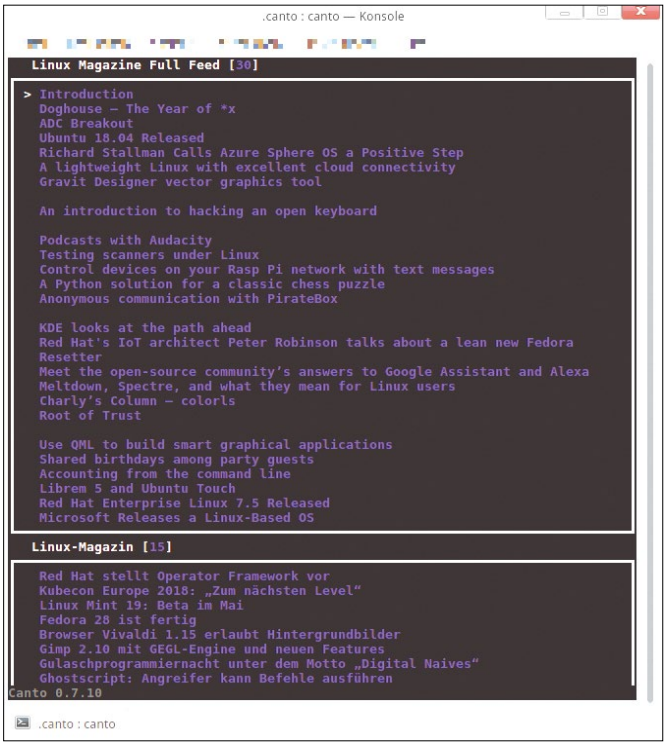


Figure 4: All messages appear below each other in Canto.

If you then call the software without further parameters, the latest headlines from the listed sources appear in the terminal. Canto has a special feature in this respect: The reader lists all messages below the respective feed, even if there are several subscribed sources. This means the user does not have to switch between feeds (Figure 4).

Canto displays the unformatted individual messages as a terminal-only reader. To display graphical elements, you need a web browser. The browser can be freely defined and entered in the configuration file. For example, if you want to use Firefox as an external browser, add the line:

```
link_handler("firefox \"%u\"")
```

When you press *G*, Canto sends the current message to Firefox.

The software also supports text-based browsers such as ELinks. If you want to customize the browser, add new feeds, and delete obsolete ones, you don't have to edit the configuration file in the terminal manually every time. It can be modified with command-line parameters. For a detailed list of parameters, refer to the software documentation in the man page [15].

The newsreader is operated with keyboard shortcuts. You then navigate between the feeds with the Up Arrow and Down Arrow keys. To open a message in the internal browser, press the Spacebar, which opens a window area with the message in a white font, while the rest of the program window still displays the individual headlines in a blue font.

If you want to update the feeds when you launch the program, add the `-u` parameter to the `start` command. Using the Left Arrow and Right Arrow keys and Shift + R and *R*, you can mark a message as read or unread.

The new versions of the 0.9.x branch let users edit the configuration directly from within the program, which means that previous command parameters for adding or deleting feeds, for example, are no longer required when calling the software. Instead, you start the application by typing `canto-curses`.

Commands are configured as in Vim by entering a colon in the program followed by the command. The command appears at the bottom of the window in the form of a command line. For example, `add: <feed address>` adds a new feed to the list.

You can delete a feed by moving to one of the feed's headlines and then typing `:del`. Canto will then tell you that it has deleted the feed. A detailed overview of the new commands and parameters is provided in the documentation [16].

Canto has a simple search function. It uses search terms to find messages; the search is limited to the headlines. If you press *F* in the message view, you can enter a search term at the bottom of the screen. Canto now displays all headings containing the search term in white.

FeedReader

The GNU GPLv3 program FeedReader [4] is a young project that only a few Linux distributions have in their portfolio thus far. It makes sense to install from the project's website, which offers numerous precompiled packages and their matching instructions. A Flatpak installation is also possible if the respective Linux distro supports this new installation option.

The software uses the Gtk+ toolkit and follows the Gnome desktop's conventions when it comes to the user interface. Operating the program turns out to be unusual, as does the interface: FeedReader has already implemented a large number of message feeds that the user can preselect. The *Local RSS* option lets you add your own content. The program window shows the individual feeds in subfolders on the left, the last message headers of the active feed in the middle, and the message on the right – correctly formatted (Figure 5).

FeedReader displays news without advertising, if possible, but cannot open links automatically. The software only lets you copy links to the clipboard to call them up in a browser with a right-click. Images in the news are enlarged after clicking on them; they can also be copied to local storage.

The most important controls are integrated into the titlebar in the usual way of Gnome applications. FeedReader also offers an input field for text searches, which then searches and filters the message list for the terms you enter: Articles that contain the search term appear in the message list. If you then open one of the arti-

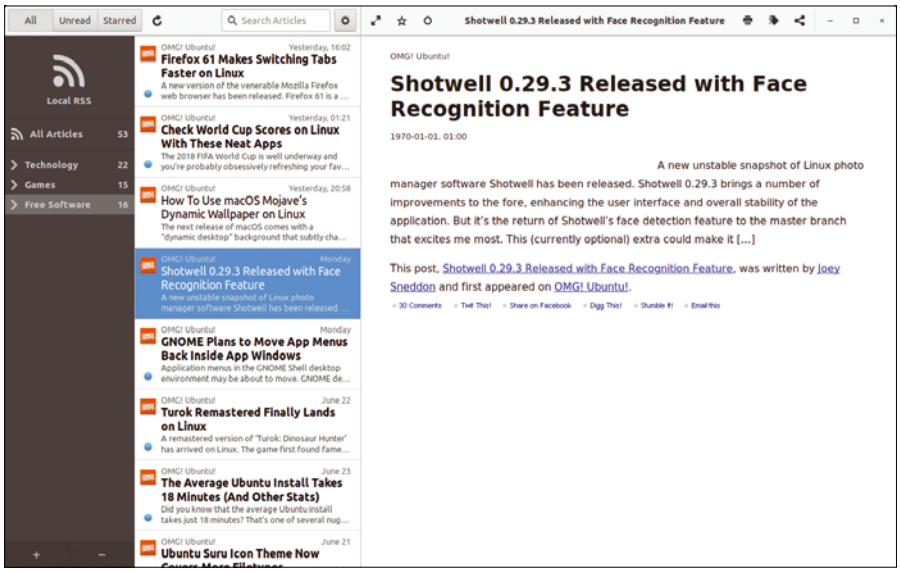


Figure 5: The FeedReader interface looks fresher than that of other programs.

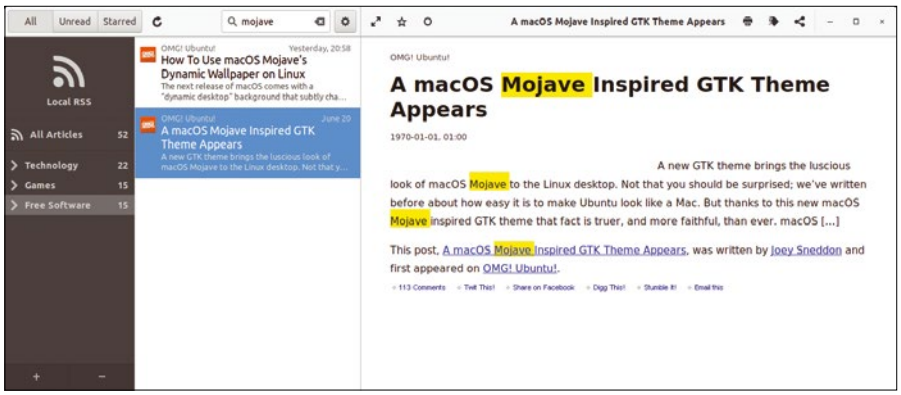


Figure 6: FeedReader also has a sophisticated search function.

cles, FeedReader also highlights in yellow the matching search term in the article.

By clicking *Tag article*, you can also add a tag to individual messages that then appears as a new category on the tree view on the left side. The reader then groups all the articles that are tagged with the respective tag into this category, giving the user quick access to topically sorted articles (Figure 6).

FeedReader has an archiving function, which you can set up from the gear symbol located top center in the titlebar and the *Settings* option. In the *Database* area, you can define deletion intervals from a selection menu, the longest interval being six months. You can also select the *Never* option to keep all messages indefinitely.

In the *Sync* area of the same Settings dialog, you can define the maximum number of posts to keep. To do this, enter an integer value in the corresponding input field. FeedReader creates the archives program-wide; they cannot be configured specifically for a group.

Liferea

Liferea [5] is a conventionally designed RSS reader with a four-pane window. At the top, a horizontal menu and button-bar list the most important functions. On the left, the individual subscribed feeds appear vertically in a tree view. The right pane lists the latest news from the active feed at the top of the screen. Below it, you will find the message itself. The program reads feeds in common RSS formats, as well as those in Atom format, and it displays podcasts (Figure 7).

The application, based on the Gtk+ component library, has an internal WebKit-based browser to display content, including multimedia objects such as video sequences. If you don't like the fact that the internal browser doesn't have an effective ad blocker, you can use an external browser, which you define individually in the Settings dialog of the software. Right-clicking on a message opens the context menu, and you can decide to open it in an internal or external browser. Because the program lets users open several tabs simultaneously, you can have a text and browser view at the same time in the individual tabs.

Liferea has a sophisticated search function that you enable by selecting *Search | Browse All Feeds* from the menu or by pressing the *Browse All Feeds* button in the buttonbar. In an intuitive search dialog, you can specify the actual search term, as well as any inclusion and exclusion criteria using advanced settings. Liferea not only browses the headlines, but also, for example, the subscription titles.

Additionally, the Read status can be defined as a rule; you can even combine

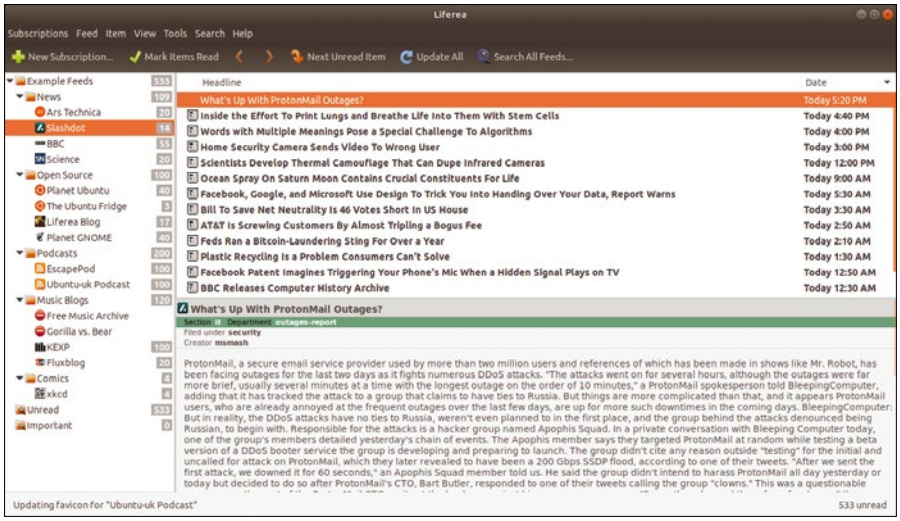


Figure 7: Liferea displays content in a conventional layout.

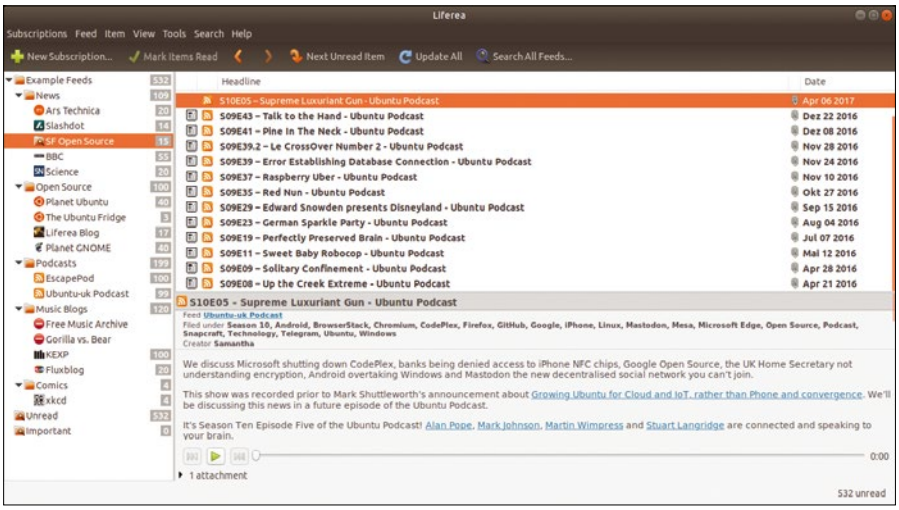


Figure 8: Liferea works with search folders to collect topic-specific messages.

several freely defined rules. Liferea also lets you mark particularly important messages with a tag that can also be used as a search criterion.

Much like the Gnome Evolution mail client, Liferea uses search folders. All matches for a search run are moved into this directory; the extended search dialog can be opened from the context menu with a right-click. You create a new search folder with *Subscriptions | New search folder*. The folder can then be inserted into the tree view at any position in the hierarchy.

The next dialog determines the search criteria. Right-clicking on the newly created folder and selecting the *Create new* option lets you search all subscriptions for the terms. Any messages found then appear in the list view sorted by feeds (Figure 8).

Group folders archive important messages in Liferea. However, the software does not update these folders automatically. You can create a group folder with *Subscriptions | New group folder*. It is then added at the current position in the folder structure tree, but it can be moved to another position later by dragging and dropping. You can then add messages to the collective folder by right-clicking on the message and selecting the *Copy to group folder* option.

If you have created several group folders, you can enable the desired folder in a fly-out menu. Group folders generally appear in the left tree view in an ochre yellow tone instead of pastel blue and can therefore be distinguished from search folders at a glance.

QuiteRSS

Russian feed reader QuiteRSS [6], which has been developed continuously since 2012, is still available in a few software repositories, but it is easiest to obtain from the project page. QuiteRSS has a conventional user interface with a buttonbar and three window areas, which clearly display all necessary information (Figure 9).

The reader can handle RSS, RDF, and Atom feeds, and it imports and exports OPML data. The software displays multiple messages in tabs. Users can also look forward to a built-in browser based on WebKit, which quickly displays the desired content. QuiteRSS also comes with an ad blocker that uses generally accessible filter lists and can be supplemented with your own filters.

QuiteRSS can also play back content in an external browser. The reader uses the standard system browser, although you can enable a different browser if required. Thanks to the click to Flash option, users can play Flash content – which is blocked by default for reasons of speed and resource savings – at the click of a mouse.

QuiteRSS offers a very efficient search feature even for large volumes of news, and it is not limited to categories. The search feature is located in the buttonbar arranged horizontally at the top of the program window. To the left of the search term input field, a magnifying glass icon takes you to a drop-down menu. You can use this to restrict the search to parts of a message, such as its author, title, or category.

In the program window's message area, the results are displayed during input. If you view the messages with the internal browser, the finds are highlighted in yellow.

To keep track of multiple feed subscriptions in the vertical list view on the main window's left side, QuiteRSS also offers the option to group feeds in arbitrarily defined folders. You can use the buttonbar in the program window for this.

QuiteRSS lets you categorize the individual news. Some default categories are available for this in the program win-

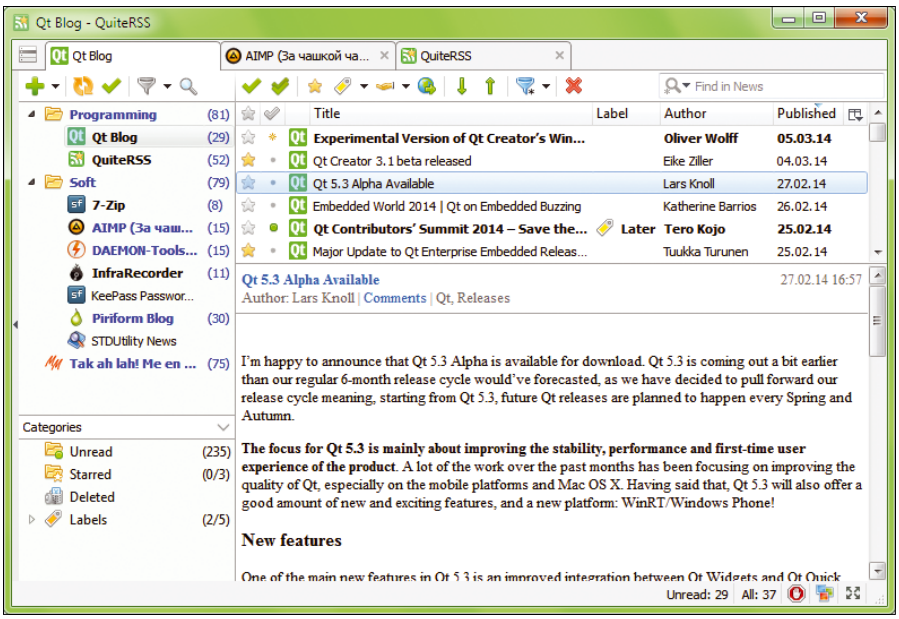


Figure 9: The still fairly young QuiteRSS reader also uses a conventional design.

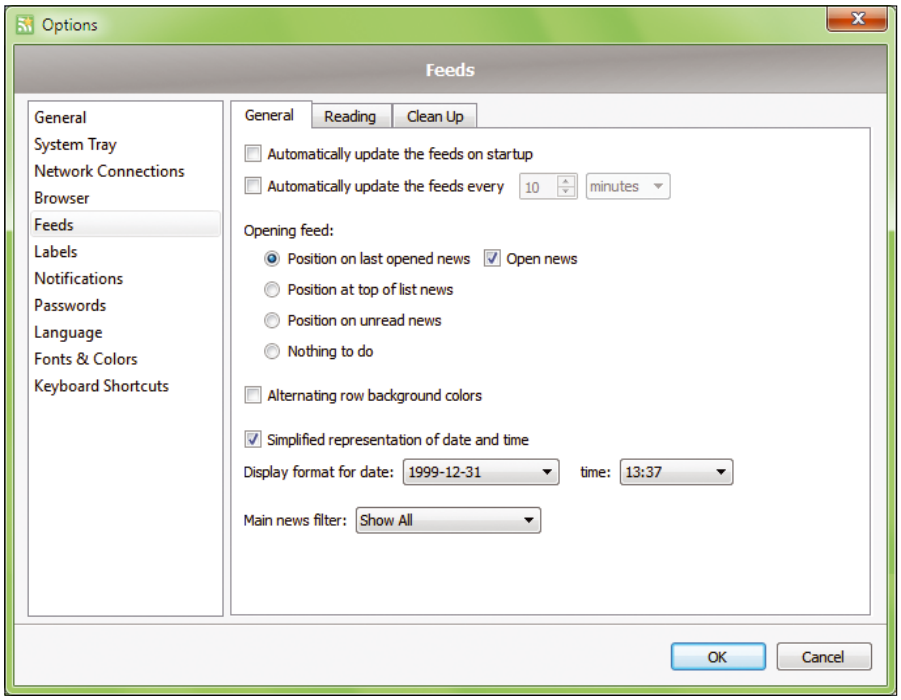


Figure 10: QuiteRSS offers very detailed settings.

dow's lower-left corner, and you can add more if needed. To create and manage categories, click on the small menu icon at top left in the main window next to the first tab. Once a message has been assigned to an individual category, QuiteRSS displays it in the news window below the headline. Categories can also be defined as search criteria.

Additionally, you can specify how long you want to keep messages with the *Tools* | *Options* menu in the *Feeds* | *Clean Up* dialog. The *Feeds* dialog lets you configure automatic update for news feeds (Figure 10).

One of QuiteRSS's unique selling points is the ability to start a data backup, which can be found in the main menu below *File* | *Create Backup*. The software asks for a backup folder and then saves the program's configuration in one file and your collection of subscribed feeds in another.

Because it stores the feeds in a SQLite database structure, you need to make sure you have enough free space in the destination folder, especially if you have many subscriptions.

Makagiga

Makagiga is not a dedicated RSS reader, but rather a complete work environment with a powerful feed reader on board [7]. In addition to the popular RSS formats, Makagiga also supports the Atom format. You do not have to install the reader as a plugin in the Java software; it is available in the application suite right from the start.

Once you have installed Makagiga, which also supports the new Java 9 Runtime Environment as a portable version for 64-bit systems, you will find an *RSS feed* entry in the task view on the left of the program window after the first start. Right-clicking on this opens a context menu in which you can subscribe to your feeds using *New* | *Add RSS Feed*.

The associated dialog takes some getting used to: You need to enter a name separately in addition to the feed address. You then click on the *Preview* button at bottom right in the dialog window; the software then displays a preview of the feed headlines after a short wait while it loads. Next, click on *Create*, and create the subscription on the main window's left side. Clicking on one of the subscriptions shows the current headlines below it.

Clicking on one of the headlines on the right side of the program window opens a conventional feed reader view with the headlines in the upper window area and the messages below (Figure 11).

Makagiga's RSS module also offers powerful search and sort functions: To search for a term within a message, open a search line at the bottom of the program window from the *Edit* | *Search* menu; alternatively, you can use Ctrl + F. Then, enter the search term in the search line and check the box to make the search case sensitive.

The first result is now shown in the message in a pastel yellow tone. Using the arrows in the search line, which let you navigation to the right (forward) and left (backward), you can jump to previous or subsequent results.

However, this search function does not let you browse headlines. To do so, you need the small search field below the tree view with the folder and RSS hierarchies. If you enter a search term, Makagiga clears the tree view and displays only the headlines in which it appears. Makagiga also color-highlights the term.

Using the *Sort by* tab, which is located below the field for entering the search term, you can also use the context menu to specify the criteria by which to sort the list. The options include the message date, the name, or, in the case of read

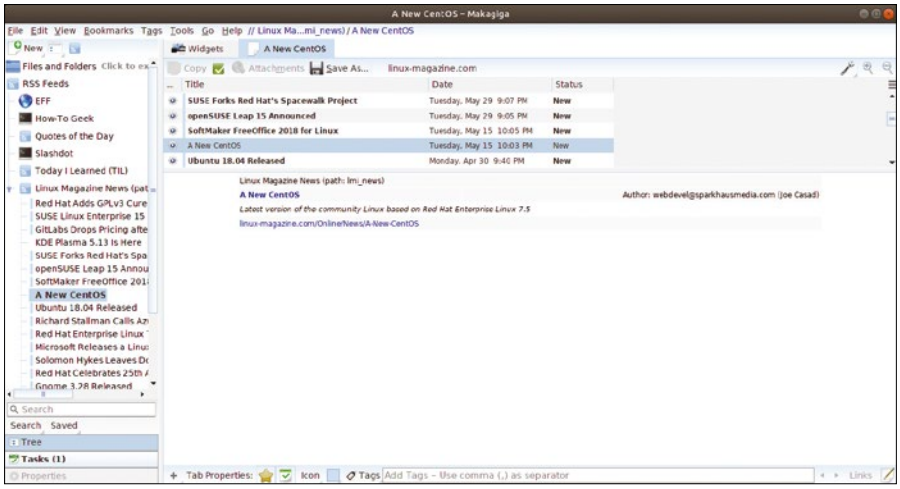


Figure 11: The interface of Makagiga's feed reader offers no surprises.

messages, a rating that you assigned (Figure 12).

Another unique feature of Makagiga's RSS reader is the ability to save individual messages as HTML or TXT files. To do this, press the *Save As* button above the headline view while accessing the message. In the following dialog, you then define the format and the storage location and save the file.

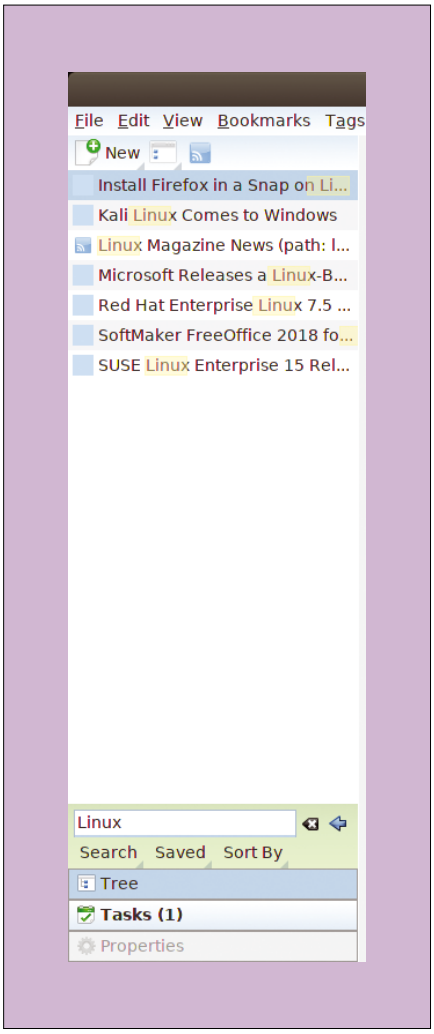


Figure 12: Makagiga offers useful sorting functions.

Conclusions

The feed readers tested here all reliably fulfill their intended task but focus on different target groups. Whereas Liferea, QuiteRSS, and Akregator offer conventional looks and cover the usual functional spectrum without massive resource requirements, FeedReader goes for visual appeal and impresses with its clarity.

As a text-based application, Canto is suitable for systems without an X server and is also extremely efficient with resources. Makagiga's feed reader, on the other hand, cannot deny its origin as a module of a complete workstation environment and therefore also offers additional functions, such as special sorting options or the ability to save individual messages as HTML files.

The differences between the individual readers are also obvious when searching for specific terms: Although all readers allow a headline search function, not all are able to search for tags or in the messages themselves.

Moreover, not all readers archive their messages. Although Canto lacks an option to store sorted content, all other programs offer at least a rudimentary archiving option.

However, the overall picture shows that the spectrum of RSS applications (Table 1) is broad enough to find a suitable application even 20 years after the advent of the first RSS feed readers. ■■■

Info

[1] RSS comeback: *https://www.wired.com/story/rss-readers-feedly-inoreader-old-reader/*

[2] Akregator: *https://www.kde.org/applications/internet/akregator/*

[3] Canto: *https://codezen.org/canto-ng/*

[4] FeedReader: *https://jangernert.github.io/FeedReader/*

[5] Liferea: *https://lzone.de/liferea/*

[6] QuiteRSS: *https://quiterss.org*

[7] Makagiga: *https://makagiga.sourceforge.io/download.html*

[8] PenguinTV: *http://penguintv.sourceforge.net*

[9] RSSOwl: *http://www.rssowl.org*

[10] BlogBridge: *https://github.com/pitosalas/blogbridge*

[11] BottomFeeder: *http://www.jarober.com/bottomfeeder/aboutbf.html*

[12] AmphetaDesk: *http://www.disobey.com/amphetadesk/*

[13] RSS versions: *https://en.wikipedia.org/wiki/RSS_%28Web-Feed%29*

[14] OPML format: *https://en.wikipedia.org/wiki/Outline_Processor_Markup_Language*

[15] 0.7.x branch man page: *http://manpages.ubuntu.com/manpages/trusty/man1/canto.1.html*

[16] 0.9.x branch documentation: *https://codezen.org/canto-ng/manual/*

Table 1: RSS Reader Overview

	Akregator	Canto	FeedReader	Liferea	QuiteRSS	Makagiga
License	GPLv2	GPLv2	GPLv3	GPLv2	GPLv3	AGPLv2
Cross-Platform	No	No	No	Limited	Yes	Yes
Graphical	Yes	No	Yes	Yes	Yes	Yes
Internal Browser	Yes	Limited	Yes	Yes	Yes	Yes
External Browser Possible	Yes	Yes	Yes	Yes	Yes	Yes
Ad Blocker	Yes	No	Yes	No	Yes	No
Extensible Filter Lists	No	No	No	No	Yes	No
Storing Messages	No	No	No	No	No	Yes (HTML)
Backup Function	No	No	No	No	Yes	No
Multimedia Content	No	Limited	No	No	Yes	No
Search folders	No	No	No	Yes	No	No
Formats	RSS/RDF/Atom	RSS/RDF/Atom	RSS/RDF/Atom	RSS/RDF/Atom	RSS/RDF/Atom	RSS/RDF/Atom
OPML Import and Export	Yes	Yes	Yes	Yes	Yes	Yes
Podcast Playback	No	No	No	Yes	No	No

OSMC

MEET THE MONITORING EXPERTS

PROGRAM ONLINE

Register now

osmc.de

Freeing Your Music Player with Rockbox

Musical Freedom

Turn your music player into open hardware with Rockbox's free firmware. *By Bruce Byfield*

If you have a large music collection or care about sound quality, just any mobile computing device won't do. However, while most music players support free-licensed formats like.flac, and a few even support.ogg, you won't find any that are sold with free-licensed firmware. That's where Rockbox [1]comes in. Rockbox is a project that develops free firmware, as well as an installer for adding the firmware to your music player. The process is the equivalent of rooting your phone, giving you complete control over a device. In effect, it creates a piece of open hardware where none existed before. Rockbox's small team of developers has been at work since 2001. Today, the project fully supports several dozen different models of music players, including Apple, Samsung, and SanDisk. Another dozen models are partially supported, and several others are in early development. However, if a music player is not listed as being supported on the project site's home page, check the forums. Sometimes, a model may be a repackaging of an earlier device. For example, while the SanDisk Clip Sport is not listed as supported, it turns out to be merely a rebranded SanDisk Clip Zip. As with any effort to replace firmware, the possibility exists that you could brick a device by installing Rockbox on it. The chance is slim, since Rockbox generally works by bypassing rather replacing a device's bootloader, and the installer includes an uninstaller that (at least in my limited experience) works perfectly well. All the same, disabling the music player remains a possibility. To avoid complete disasters, make sure that you have copies of all the files on a music player before you attempt to install Rockbox, and remember that you install at your own risk.

Figure 1: The Rockbox utility simplifies installing the free firmware.

Downloading and Installing Rockbox

To install Rockbox, download the stable version for your music player model to your home directory. Uncompress the archived file, and the Rockbox utility is ready to use. You can also install manually using your music player's online manual [2], but, for most cases, the Rockbox utility is easier and quicker to use (Figure 1).

However, the utility does have a few quirks. To start with, auto-detection does not work unless the music player is connected to your computer after the utility is opened. If the player is still not detected, you need to add its mount point and model manually. Once you have configured the installation, at a minimum you must install the bootloader and firmware to the device. However, before installation starts, you have the option to install extra fonts, themes, and games. Some of these extras require additional files, which should be copied to the same directory as the Rockbox utility. Press the Install button and the Prerequisite dialog opens, warning that you also need to download the firmware for your device – a detail I would have preferred to know earlier, so I wouldn't have to interrupt the installation to find

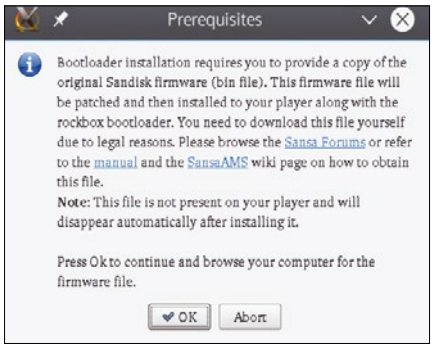


Figure 2: Installation is interrupted to tell you that you need to download the firmware to install.

the firmware. The dialog does not tell you where to place the firmware; instead, you have enter the path for it.

Overall, the Rockbox utility can best be described as basic. It does not do things it could easily do, such as download the extras selected or the required firmware. However, it is not difficult to use so much as annoying in little ways.

The Interface

Music players – especially ones that fit in your palm with room to spare do not have elaborate interfaces. The Rockbox interface is no exception and is generally an acceptable substitute for the one that shipped with my SanDisk Clip Sport, although with a few omissions and some sloppy menu structures. Like the original, the Rockbox interface is divided into seven sections: *Radio*, *Music*, *Books*, *Sports*, *Folder*, *Card*, and *Settings*, each with its own subdirectories. For example, *Books* is divided into *Audiobooks* and *Podcasts*, while *Sports* includes both a timer and stopwatch. The main purpose of the *Card* section is apparently to have a top-level menu item for external microSD cards, since a separate entry for the external card is available under *Folder*. All these top-level items can be turned off under *Settings* | *Customize*, which can eliminate endless scrolling through options you have no intention of using. Under *Music*, users can opt for *Shuffle*, to play songs in what Rockbox claims is a truly random order, unlike most music players, or to create playlists. However, unlike some interfaces I have seen, in the case of the SanDisk Clip Sport, Rockbox does not allow you to rank songs or to select by genre or any other meta fields associated with a song file. Instead, files can only be selected by artist, album, or songs, all of which are displayed in only fourteen characters. Neither can album art be displayed. How songs play is controlled generally under *Settings* | *System* Settings and specifically in the menu for each song. In both places, *Equalizer* and *Replay Gain* are available for tweaking songs. Under *Settings* | *Music Options*, you can also toggle *Shuffle* and *Repeat*. My subjective impression is that the Rockbox firmware loads and refreshes faster than the original on my SanDisk Clip Sport. However, on the negative side, where the original does not start to distort sound until the seventh of eighth sound levels, with Rockbox, the sound starts to distort at the sixth level, reducing the options to something that makes the Rockbox firmware less than ideal on a crowded bus or busy street. Another limitation is that files on a microSD card are not displayed on the Rockbox interface. Instead, you have to go through the *Card* section.

Figure 3: The Rockbox Utility also backs up installation files and uninstalls.

This arrangement is generally acceptable, unless you want to play songs in track order, in which case file names need to be prefaced by the track number. None of these shortcomings are crippling. However, for some, at least a few of the limitations might be unacceptable. But then, I suppose that the main point is to have free firmware, not a full-featured, logical interface.

Backup and Uninstalling

The Rockbox utility can also be used to update the firmware on a device, or to back up the files in the installation directory. You can also uninstall the firmware in one of two ways: by uninstalling the Rockbox bootloader, which is used to bypass the original firmware, or by a complete removal of all aspects of the firmware, including the extras (Figure 3). Unfortunately, as happened in my experiments, removing the bootloader may not always be an option for some devices. Fortunately, uninstalling all of the Rockbox firmware is a smoother operation, although, as the dialog warns, you do have to remove the bootloader from the music player manually; it will be the only `.bin` file in the player's top-level interface. If something goes wrong, you can always reinstall the Rockbox firmware so that the player remains usable.

At the end of my experiments, I remain intrigued by Rockbox, although mildly disappointed. As a free software user, I am no longer accustomed to situations in which using free software means awkwardness and the loss of some features, even if the inconvenience is mild. I could use Rockbox in its present state, but, practically speaking, I am not that eager to do so. Still, I will keep an eye on Rockbox and see how its firmware improves over the next few years. I see that the project is starting to define what is necessary to port the FiiO X1 [3], my music player when I'm on the road. When that development produces a stable release, I will be curious to see how well it is implemented. ■■■

Info

[1] Rockbox: *https://www.rockbox.org/*

[2] online manual: *https://www.rockbox.org/manual.shtml*

[3] port the FiiO X1: *http://forums.rockbox.org/index.php?topic=51047.30*

shop.linuxnewmedia.com

zstd

File Compression for Modern Computing

In an effort to meet modern computing needs, zstd offers a greater degree of compression at a faster compression rate, with unique options to enhance performance. *By Bruce Byfield*

Many standard Linux tools have been around so long that second-generation tools are being developed to meet modern needs. For instance, Neovim is an update of the Vim text editor, and `apt` is a rearrangement of the basic tools for `apt-get`, the Debian package manager. Similarly, Zstandard (`zstd`) [1] is a revision of compression tools like `tar` and `gzip`, except with higher degrees of compression at a faster rate. Additionally, `zstd` includes several unique tools for enhanced performance, such as advanced compression features, compression levels and strategies, and dictionaries.

`zstd` was written by Facebook employee Yann Collet and released in August 2016. Briefly, it is a lossless compression algorithm based loosely on the earlier LZ77 algorithm [2]. The command's syntax is deliberately similar to that of `gzip`, down to variations on the basic command that are the equivalent of popular options. For example, `zstdmt` is the same as `zstd -T0` (use the same number of threads as detected cores), whereas `unzstd` is the same as `zstd -d` (decompress), and `zstdcat` is the same as `zstd -dcf` (decompress, force write to standard output, and overwrite without prompt).

The Basics

Getting started with `zstd` is as simple as typing:

```
zstd FILE
```

Multiple files can be specified using a space-separated list. Unless you add `--rm` as an option, the original file is not deleted. A progress bar is displayed as a single file is compressed; unless `-q` is added to the command, an error produces a short help page. Unless otherwise specified, level 3 compression is

Author

Bruce Byfield is a computer journalist and a freelance writer and editor specializing in free and open source software. In addition to his writing projects, he also teaches live and e-learning courses. In his spare time, Bruce writes about Northwest coast art. You can read more of his work at *http://brucebyfield.wordpress.com*


```
bb@nanday:~/projects/creative/ilvarness/Map$ zstd test.png
test.png              : 95.48%   (42937356 => 40996162 bytes, test.png.zst)
bb@nanday:~/projects/creative/ilvarness/Map$
```

Figure 1: The basic `zstd` command.

used along with four threads (see below), and a data-integrity check is done on the original file before compression. The result is a file with the same name as the original file, but ending in `.zst` (Figure 1).

To decompress, type:

```
zstd -d FILE
```

and a decompressed file is created without the `.zst` extension. If you specify more than one `.zst` file to uncompress, all the files are decompressed into a single file. Another option is to run `--test` (`-t`) to check the integrity of compressed files without creating or deleting any files.

In any operation, you can specify file size as needed in kilobytes (using KiB, Ki, K, or KB) or in megabytes (using MiB, Mi, M, or MB).

Basic Options

Most of `zstd`'s basic behaviors can be modified by options. To start, `zstd` has both verbose (`-v`) and quiet (`-q`) modes for running the command. You can also use `-o FILE` to specify any file name you want for an output file, placing the option after the original file's name, instead of directly after the basic command with the rest of the options. Additionally, if you are aware that a compressed file of the same name as the output file already exists, you can add `--force` (`-f`) to overwrite any file of the same name without confirming the operation first.

Several options help speed up commands. You can save time by turning off the integrity check during compression with `--no-check`. The increased speed, of course, comes with the possibility that the compressed file might not be usable. A somewhat safer option to increase the speed is to enable `--sparse`, which reduces the number of zeroes in the output file, which can add a couple more percentage points of compression when dealing with a text file. For a graphics file, however, `--sparse` saves so little that it hardly seems worth using unless you are determined to save every bit of hard drive space possible.

As a recently created utility, `zstd` can also be compiled to use multiple CPU cores to make compression faster. By default, only one core is used, but you can adjust the number with the option `-T=NUMBER` (`--threads=NUMBER`).

If the value is `0`, then `zstd` will detect the number of cores and try to use all of them. Should the online help appear, you will know that the `zstd` version you are using was compiled without threading.

Compression Levels, Strategies, and Advanced Options

`zstd` approaches compression in two different ways. The more conventional tactic is to specify a specific compression level using `--compress` (`-z` or `-#LEVEL`). The default level `3` can be overridden with any number from `1` to `19`, with `1` being the quickest and least compressed, and `19` the slowest and most compressed. To give a sense of the choices involved, a compression setting of `1` reduced the size of a 42MB `.png` file by five percent in about six seconds, whereas a setting of `19` compresses the same file by just under eight percent in about 20 seconds. With a plain text file of 4,600 bytes, level 1 compression produces an archive file 55 percent smaller, wereas level 19 compression creates a file that is 59 percent smaller, both requiring only a few seconds. This difference between graphic and text files is typical.

You also have the option of adding the `--ultra` option to enable the high, more memory intensive compression of levels 20-22. However, when used by themselves, the advanced compression levels are no more efficient than level 19 compression. To get the most from the ultra-compression levels, you need to experiment with the advanced options.

The advanced options for compression are defined in the option:

```
--zstd=OPTION=SETTING,OPTION=SETTING
```

The easiest to use is `strategy=` (`strat=`). This option can be completed with a number from `0` to `7`, in which `0` is the fastest and `7` the most compressed. Each strategy contains a number of methods and searches the file being compressed for an opportunity to use them. This

```
bb@nanday:~$ zstd --train /home/bb/fonts/*
sorting 968 files of total size 0 MB ...
finding patterns ...
!  warning : selected content significantly smaller than requested (1661 < 112640)
!  consider increasing the number of samples (total size : 0 MB)
statistics ...
Save dictionary of size 1850 into file dictionary
```

Figure 2: A dictionary of related files can be specified to help `zstd` compress other files more efficiently. `zstd` outputs warnings, but leaves users to select the relevant files.

search greatly increases both the time and the memory required to compress the file. However, the use of 7 can double the compression for a file.

Other advanced options for compression can override any of the options used in `zstd`'s compression algorithm. For instance, `hashLog=BITS` sets the maximum number of bits for a hash table, making compression faster. Unfortunately, the man page lists the algorithm options with only a brief explanation of what they do, so most users will have to experiment blindly or else find other sources of information to understand what is being adjusted. Any algorithm option not specifically altered will use its default settings.

Compression Dictionaries

A dictionary is a file that stores the compression settings for small files. A dictionary is assembled from a group of typical small files that contain similar information, preferably over 100 files. For greatest efficiency, their combined size should be about one hundred times the size of the dictionary produced from them. If the files used are fewer or smaller in size than recommended, `zstd` will display a warning but still allow the dictionary to be created (Figure 2).

To create a dictionary, use the command:

```
zstd --train FILES
```

The dictionary will be saved with the default name *dictionary*, and a default size of 112,640KB. To give the dictionary its own name, add the name to the `train` option; for example, a dictionary called *quick* would be named using the option `--train-quick`. You can also force the dictionary to use the most compressed files by specifying the number of files to use after the name; for example,

```
--train-quick=k=NUMBER OF FILES
```

A specific size can be added with the option `--maxdict=SIZE`, and a specific ID with `--dicID=NUMBER`, which makes communication with the dictionary faster than relying on the name. To use a dictionary, add the option `-D FILE` to the command. Nothing in the output will indicate that the dictionary is in use.

In general, the smaller the file, the greater the improvement in compression. According to the man page, a dictionary can only increase the compression of a 64KB file by 10 percent, compared with a 500 percent improvement for a file of less than 1KB.

Benchmarking

To use `zstd` to its full potential requires experimentation. To use the advanced compression options, you probably will need to research the compression algorithm. However, with the methods listed here, `zstd` is sure to be efficient.

But how efficient? More particularly, how does `zstd` compare with other compression tools? `zstd` provides its own answer with a small selection of benchmarking options. To start, you can use the option `-bLEVEL` to set the compression level to test. Alternately, you can use `-bLEVEL` to indicate the start of a range of compression levels and use `-eLEVEL` to indicate the end of the range (Figure 3). You can also change the default of three seconds for the length of the testing with `-iSECONDS`. Of course, you can also make notes as you gain experience with `zstd`.

`zstd` has been released recently enough that, in many ways, it is still an expert's tool. However, although the documentation can be spotty for the advanced features, there is still enough to make `zstd` an alternative tool for any level of user, especially those who want a compression tool designed for modern computing. ■■■

Info

[1] zstd: *https://en.wikipedia.org/wiki/Zstandard*

[2] LZ77 algorithm: *https://en.wikipedia.org/wiki/LZ77_and_LZ78*

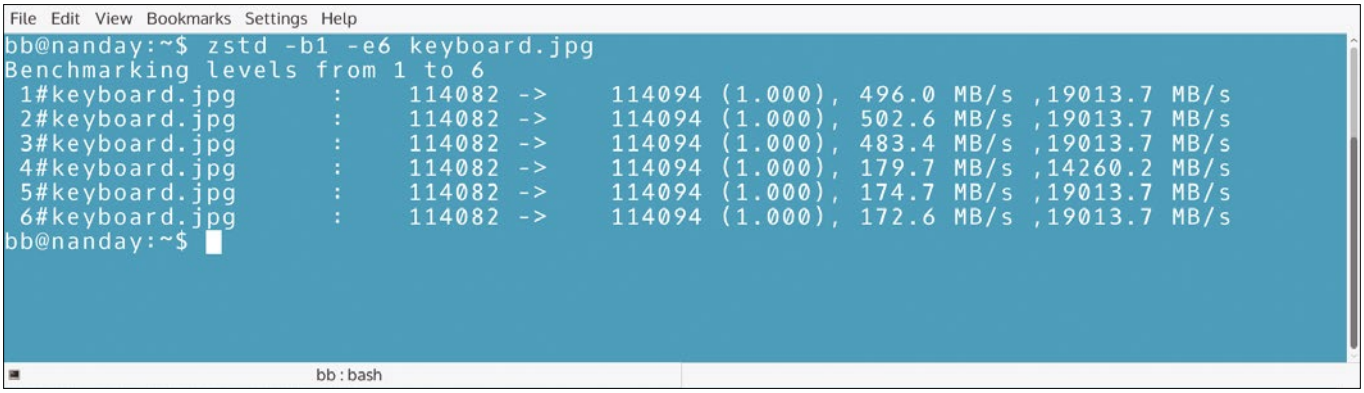
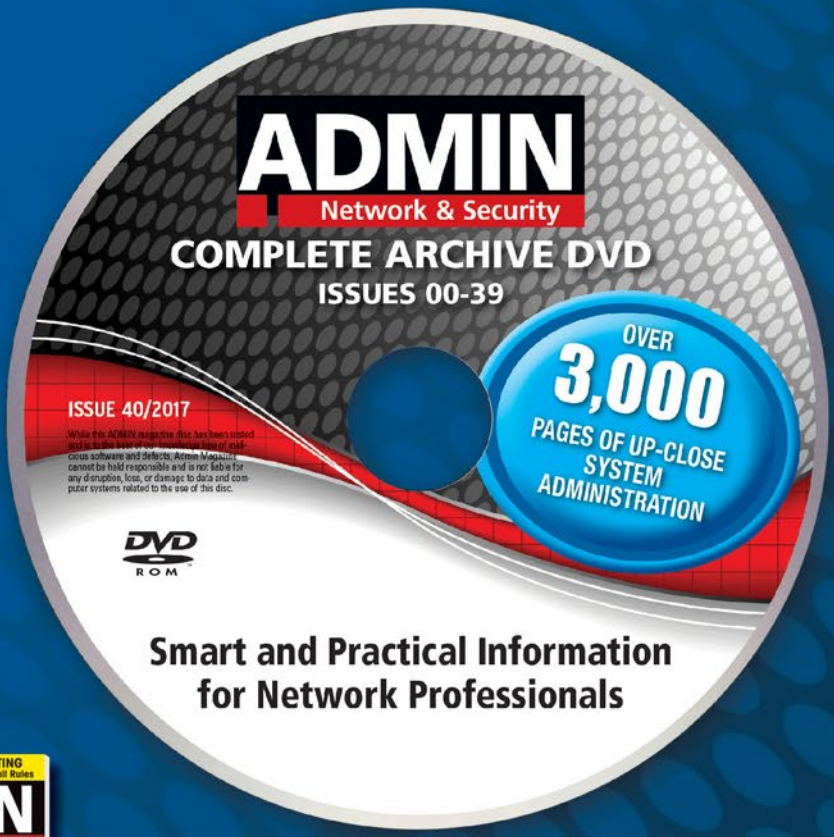


Figure 3: A benchmark of compression rate and compression speed for the same file at different levels of compression.

7 Years of ADMIN on One DVD



This searchable DVD gives you 40 issues of ADMIN, the #1 source for:

- systems administration
- security
- monitoring
- databases
- and more!

Clear off your bookshelf and complete your ADMIN library with this powerful DVD!

ORDER NOW!

shop.linuxnewmedia.com

The sys admin's daily grind: Reverse SSH tunnel and autossh

Tunnel Break

This month, Charly Kühnast draws attention to a widely unknown weather phenomenon: The instability of rarely used tunnels leading to a Raspberry Pi. Read on for greater insights.

By Charly Kühnast

Recording environmental data is one of my hobbies; I already reported about my magnificent dust sensor [1]. I also own a small weather station: the popular WH1080, which is sold under names like Fine Offset, Nextrend, Froggit, or TFA Nexus. It includes various external sensors, as well as an indoor base station.

I continually extract the measurement data via the USB port of the base system – any Linux machine would be capable of it, but my old Raspberry Pi 2B is the perfect choice. I use RRDtool to write the data to a round-robin database and conjure up colorful graphs for a web server (Figure 1.)

I had to come up with something to publish the data on the web. My ISP uses Dual-Stack Lite [2] and doesn't even offer static IPs for an extra charge – and, I have an aversion to dynamic DNS services.

A reverse SSH tunnel should fix it. The Raspberry Pi opens a connection from the inside through the NAT to a server outside, which I rented for a little money. (Performance doesn't matter; it's all about having a static IP address.) This SSH connection creates a direct tunnel between port 80 of the web server and port 80 of the Raspberry Pi. This was fine as a proof of concept; later, I converted it to HTTPS with a certificate from Let's Encrypt.

The syntax for tunneling is simple. On the Raspberry Pi, you enter:

```
ssh -R <Webserver>:80:localhost:
  80<User>@<Webserver>
```

The tunnel is set up; I access the web server and see *Connection refused*. So what went wrong? After five minutes with a search engine that I really don't trust, I found this out: On the server, I have to add a `GatewayPorts clientspecified` line to `/etc/ssh/sshd_config`. Without this, the port is only bound to localhost.

Not Enough Action in the Tunnel

Unfortunately SSH disconnects after a period of inaction; because my weather server is not quite as popular as the one run by the Met Office, this happens quite soon. I can set the timeouts, but sooner or later the tunnel always breaks down, and it does not reopen. I succeeded in eliminating this annoyance with `autossh` [3]. The tool monitors the connection and restarts crashed tunnels. The syntax for doing this is:

```
autossh -M 9999 -N -R <Webserver>:80:
  localhost:80 <User>@<Webserver>
```

The `-M` switch is the monitoring connection. I chose the port number arbitrarily; it and the next one (i.e., 10000) both need to be free.

Caution: The machine (here, the Raspberry Pi) must not have access to the internal LAN at the same time, because if an attacker took over, they would be handed my entire home network on a silver platter. In my environment, the Raspberry Pi is connected to special DMZ ports on the firewall, which isolates it from the LAN. So let's see: 21°C – time to go out into the garden and see if the watering Raspberry Pi is doing its job properly. ■■■

Info

[1] "Charly's Column: Particulate Matter Measurement with the Raspberry Pi" by Charly Kühnast, *Linux Pro Magazine*, issue 213, August 2018, p. 54, *http://www.linuxpromagazine.com/Issues/2018/213/Breathe-deeply/(language)/eng-US*

[2] Dual-Stack Lite: *https://en.wikipedia.org/wiki/IPv6_transition_mechanism#Dual-Stack_Lite_(DS-Lite)*

[3] autossh: *https://linux.die.net/man/1/autossh*

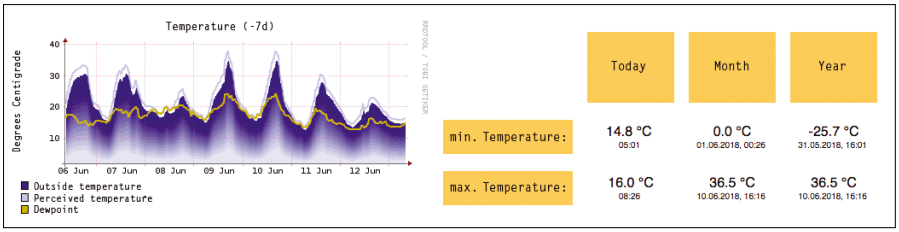


Figure 1: Charly's Raspberry Pi server records the weather station data with RRDtool.

Author

Charly Kühnast manages Unix systems in the data center in the Lower Rhine region of Germany. His responsibilities include ensuring the security and availability of firewalls and the DMZ.

What?!

I can get my issues SOONER?

Available anywhere, anytime!

Sign up for a digital subscription and learn the latest techniques for better network security, system management, troubleshooting, performance tuning, virtualization, and cloud computing on Windows, Linux, and popular varieties of Unix.

shop.linuxnewmedia.com/us/magazines/admin-magazine/digital-subscription.html

Implementing fast queries for local files in Go

Inventory Browser

To find files quickly in the deeply nested subdirectories of his home directory, Mike whips up a Go program to index file metadata in an SQLite database. *By Mike Schilli*

Those were the days when you could simply enter a line of code or a variable name into Google's code search page (Figure 1), and the data crawler would reveal in next to no time the open source repositories in which it appeared. Unfortunately, Google discontinued the service several years ago, probably because it failed to generate sufficient profit.

But not all is lost, because the GitHub Codesearch [1] project, with its indexer built in Go, at least lets you browse locally available repositories, index them, and then search for code snippets in a flash. Its author, Russ Cox, then an intern at Google, explained later how the search works [2].

How about using a similar method to create an index of files below a start directory to perform quick queries such as: "Which files have recently been modified?" "Which are the biggest wasters of space?" Or "Which file names match the following pattern?"

Unix filesystems store metadata in inodes, which reside in flattened structures on disk that cause database-style queries to run at a snail's pace. To take a look at a file's metadata, run the `stat` command on it and take a look at the file size and timestamps, such as the time of the last modification (Figure 2).

Newer filesystems like ZFS or Btrfs take a more database-like approach in the way they organize the files they contain but do not go far enough to be able to support meaningful queries from userspace.

Author

Mike Schilli works as a software engineer in the San Francisco Bay area, California. Each month in his column, which has been running since 1997, he researches practical applications of various programming languages. If you email him at *mschilli@perlmeister.com* he will gladly answer any questions.

Fast Forward Instead of Pause

For example, if you want to find all files over 100MB on the disk, you can do this with a `find` call like:

```
find / -type f -size +100M
```

If you are running the search on a traditional hard disk, take a coffee break. Even on a fast SSD, you need to prepare yourself for long search times in the minute range. The reason for this is that the data is scattered in a query-unfriendly way across the sectors of the disk.

To speed this up, I want an indexer to collect the metadata at regular intervals (e.g., during quiet periods in the early morning hours) and store them in an SQLite database that I can run a query tool against. The `updatedb` utility on Linux does similar things so that the user can search for file names at lightning speed with `locate`.

Because the "Programming Snapshot" series never shies away from new challenges, and this particular edition was being written during a vacation in Hawaii with me in a good mood and plenty of time on my hands, I used the Go programming language for the implementation – just as in the Codesearch project. An SQLite database is used as the index; it stores all data in a single file but packs enough punch to support optimized and reasonably fast queries, even for medium-sized data volumes.

To do this, Listing 1 [3] creates an SQLite database named `files.db` that inserts a table entry with the fields `path` (full file path), `size` (file size), and `modified` (timestamp of the file's last modification) for each file found in the directory tree.

To install the required SQLite driver, the `go get` tool included in every Go installation retrieves the driver's source

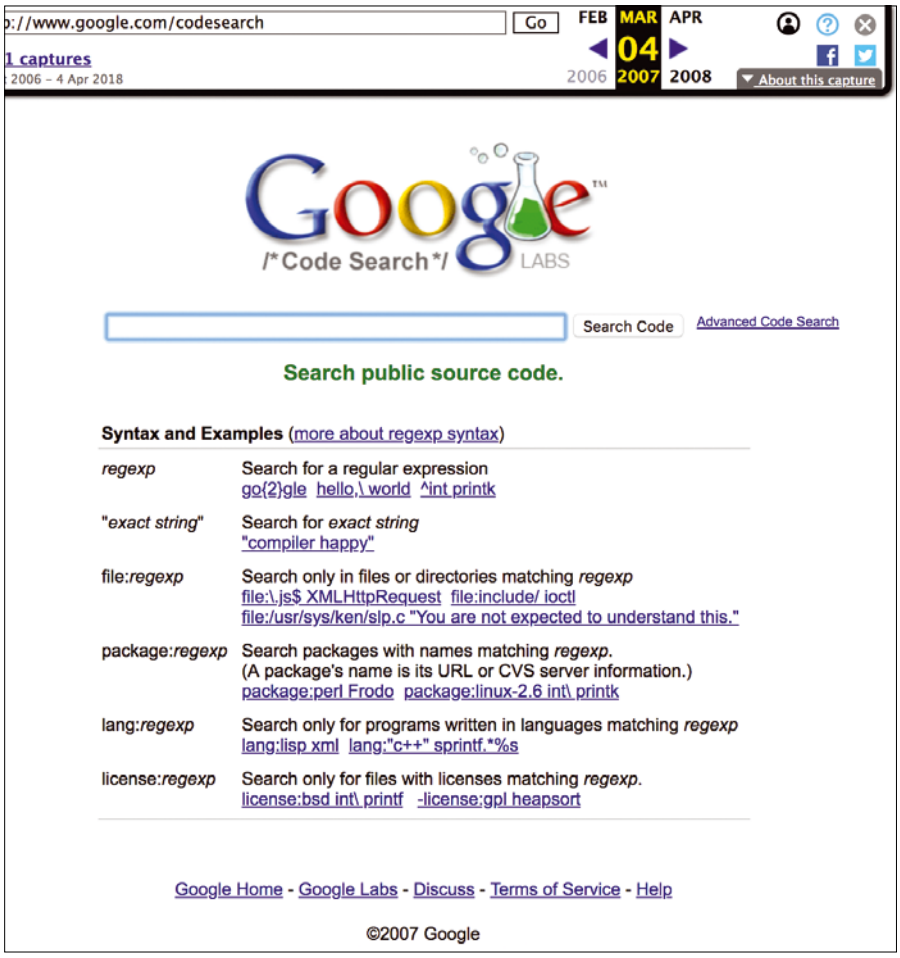


Figure 1: This is what Google Code Search looked like in 2007.

```
mybox$ stat home.html
  File: 'index.html'
  Size: 42796           Blocks: 88         IO Block: 262144 regular file
Device: 811h/2065d      Inode: 4500412696  Links: 1
Access: (0664/-rw-rw-r--)  Uid: (20352/mschilli)   Gid: (33211/pg125139)
Access: 2016-05-25 20:33:55.700977753 -0700
Modify: 2018-05-05 13:01:29.754614919 -0700
Change: 2018-05-05 13:01:29.754614919 -0700
 Birth: -
```

Figure 2: Inode metadata of a file, here determined by `stat`, can be used to build an index.

code from GitHub, resolves any dependencies, and compiles and installs the result locally:

```
go get github.com/mattn/go-sqlite3
```

Scripts like Listing 1 can then load the driver in their `import` sections under the full name starting with *github.com*. I had to employ one minor quirk to prevent the rather adamant Go compiler from exploding, though: It apparently felt that loading the driver package imported code that does not appear to be referenced anywhere in Listing 1. Go does not tolerate unnecessary dependencies, but in this case, the driver is actually required, and it is indeed used under the covers in the `database/sql` libraries. To calm down the Go compiler, line 5 imports the driver under the name _ (underscore), which turns off the is-it-being-used check.

The start of each Go program is a `package main` with a `function main()`, as in line 8 of Listing 1. The `Open()` method there uses the `database/sql` package, Go's standard database interface (and the SQLite driver under the hood), to open a connection to the `files.db` flat file database later.

If the file already exists or another error occurs, the return value `err` contains a value not equal to `nil`, and line 11 sounds the alarm with `panic()` and abruptly terminates the program flow.

Old School Error Checking

The SQL command `CREATE` in line 14 creates a new database table. Here, too, line 16 checks whether an error was returned and terminates with a message if necessary. Go still believes in old-school-style checking of individual return values instead of allowing the programmer to throw exceptions that bubble up for processing at a higher level.

Also, take a closer look at the different assignments in lines 9 and 14, which first use `:=` and then `=`. The first is Go's assignment with a declaration. If a variable has not yet been declared, the short form declaration with `:=` does the trick.

Watch Out!

If the declaration list of variables with a `:=` assignment contains only variables that have already been declared, though, Go stubbornly refuses to compile the

Listing 1: create.go

```
01 package main
02
03 import (
04   "database/sql"
05   _ "github.com/mattn/go-sqlite3"
06 )
07
08 func main() {
09   db, err := sql.Open("sqlite3", "./files.db")
10   if err != nil {
11     panic(err)
12   }
13
14   _, err = db.Exec("CREATE TABLE files " +
15       "(path text, modified int, size int)")
16   if err != nil {
17     panic(err)
18   }
19 }
```

Listing 2: index.go

```
01 package main
02
03 import (
04   "database/sql"
05   _ "github.com/mattn/go-sqlite3"
06   "os"
07   "path/filepath"
08 )
09
10 type Walker struct {
11   Db *sql.DB
12 }
13
14 func main() {
15   if len(os.Args) != 2 {
16     panic("usage: " + os.Args[0] +
17           " start_dir")
18   }
19   root := os.Args[1]
20
21   db, err :=
22       sql.Open("sqlite3", "./files.db")
23
24   w := &Walker{
25     Db: db,
26   }
27
28   err = filepath.Walk(root, w.Visit)
29   checkErr(err)
30
31   db.Close()
32 }
33
34 func (w *Walker) Visit(path string,
35           f os.FileInfo, err error) error {
36   stmt, err := w.Db.Prepare(
37       "INSERT INTO files VALUES(?,?,?)")
38   checkErr(err)
39
40   _, err = stmt.Exec(
41       path, f.ModTime().Unix(), f.Size())
42   checkErr(err)
43
44   return nil
45 }
46
47 func checkErr(err error) {
48   if err != nil {
49     panic(err)
50   }
51 }
```

code at all. In this case, the assignment must be made with an `=`, as in line 14.

Similarly unique to Go are functions with several return values. Often an error variable is contained in the list of values. If it is set to `nil`, everything is fine. If a return value is of no interest, such as the first return value of `db.Exec()` in line 14, Go developers type an underscore in place of an unnecessary variable. Picking up only one return value from a function that returns two would result in a compiler error.

The code is easily compiled by entering:

```
go build create.go
```

A fraction of a second later a `create` executable is available; it already contains all the dependencies and libraries, so it can be easily copied to another machine using the same operating system without a pile of dependencies having to be resolved. The resulting binary is about 4.5MB, which is not exactly lightweight either.

Closure as a Bridge

Listing 2 shows the indexer, which uses the `Walk()` method from the standard *path/filepath* package to navigate through a file hierarchy, starting with the start directory specified by the user on the command line. Arguments passed to the program are found in the `os.Args` array, as in C, with the program name as the first element and all of the call parameters in the following ones.

Browsing a file tree isn't rocket science, but Go uses the `Visit()` callback function to communicate with the traversing function in line 28. The problem here is that no database handle exists within the scope of this callback starting on line 34, which it needs to make the necessary changes to the database. The solution to this dilemma is to turn the `Visit()` function into a closure.

To do this, `Visit()` in line 34 defines a so-called receiver between the `func` keyword and the function name, thus telling Go to connect the `Walker` data structure (line 10), which contains a database handle, with the `Visit()` function. This allows `Visit()` to access the handle via the `w` variable used for defining the receiver. With the handle, it inserts new records into the database.

The actual work of setting up a database query is done by the `Prepare()` method, which prepares an SQL command and returns a statement handle. Line 40 then fires the `Exec` method at the latter and passes the parameters to be stored to the SQL command: the path to the file, its last modification timestamp, and its size.

To avoid the need for the program to check after each function call whether the error variable `err` has a value of `nil`, and thus everything is OK, line 47 defines a function named `checkErr()`, which does this and aborts the program with `panic`, if something unforeseen happens.

Finders, Keepers

After the indexer finished its work, the database table files on my computer had more than a million entries, as shown in Figure 3. The reason for the high number of files was probably numerous cloned Git repositories and Snapshot articles from more than 20 years. With this

```
$ sqlite3 files.db
SQLite version 3.19.3 2017-06-27 16:48:08
Enter ".help" for usage hints.
sqlite> .schema files
CREATE TABLE files (path text, modified int, size int);
sqlite> select count(*) from files;
1162647
sqlite> ^D
$
```

Figure 3: After the indexer has finished, there are more than one million file entries in the flat file SQLite database.

Listing 3: latest.go

```
01 package main
02
03 import (
04   "database/sql"
05   "fmt"
06   _ "github.com/mattn/go-sqlite3"
07 )
08
09 func main() {
10   db, err :=
11     sql.Open("sqlite3", "./files.db")
12   checkErr(err)
13
14   rows, err := db.Query("SELECT path, " +
15     "modified FROM files " +
16     "ORDER BY modified DESC LIMIT 10")
17   checkErr(err)
18
19   var path string
20   var mtime string
21
22   for rows.Next() {
23     err = rows.Scan(&path, &mtime)
24     checkErr(err)
25     fmt.Printf("%s %s\n", path, mtime)
26   }
27 }
28
29 func checkErr(err error) {
30   if err != nil {
31     panic(err)
32   }
33 }
```

data in the `files.db` SQLite database, a SQLite client can now quickly fire off queries and determine which files in my home directory have recently changed, for example.

To do this, Listing 3 connects to the SQLite database and issues a `SELECT` command that queries all rows in the table, sorts them in descending order of the timestamp in the `modified` column, and then outputs the first 10 matches.

The `rows.Next()` call in line 22 works its way step-by-step through the matches, and `rows.Scan()` retrieves the first two column values of each match and assigns them to the `path` and `mtime` variables passed in as pointers; both of these were previously declared as strings. Go supports pointers, but it does not leave memory management up to the user and does not blow up in smoke like C if an address is wrong because of a bug; instead, it quits with helpful error messages.

Which files in my home directory take up the most space? Listing 4 finds this out quickly by sorting all entries in descending order (`ORDER BY size DESC`) using the `SELECT` query from line 25 and `LIMIT`ing the output to a maximum number of matches. The user defines this number with the `--max-files` parameter at the command line, and Go provides a convenient interface for parsing the parameters of a command with the `flag` package.

It first expects the declaration of the variable that will hold the value passed in

Listing 4: max-size.go

```
01 package main
02
03 import (
04   "database/sql"
05   "fmt"
06   "flag"
07   "os"
08   "strconv"
09   _ "github.com/mattn/go-sqlite3"
10 )
11
12 func main() {
13   db, err :=
14     sql.Open("sqlite3", "./files.db")
15   checkErr(err)
16
17   max_files := flag.Int("max-files", 10,
18     "max number of files")
19
20   flag.Parse()
21   if len(flag.Args()) != 0 {
22     panic("usage: " + os.Args[0])
23   }
24
25   rows, err := db.Query("SELECT path," +
26     "size FROM files " +
27     "ORDER BY size DESC LIMIT " +
28     strconv.Itoa(*max_files))
29   checkErr(err)
30
31   var path string
32   var size string
33
34   for rows.Next() {
35     err = rows.Scan(&path, &size)
36     checkErr(err)
37     fmt.Printf("%s %s\n", path, size)
38   }
39 }
40
41 func checkErr(err error) {
42   if err != nil {
43     panic(err)
44   }
45 }
```

from the command line (`max_files` in line 17). The call to the `flag.Int()` method specifies that only integers can be used as values. Then `flag.Parse()` (line 20) analyzes the existing command-line parameters and – if the user has set `--max-files` – assigns this value to a variable that the `max_files` pointer references.

The `Itoa()` function from the *strconv* package converts the integer behind the dereferenced `*max_files` pointer back into a string, and line 28 injects it into the SQL command using a `LIMIT` clause. The advantage of this conversion type is that an integer actually ends up in the query and not a character string that could be abused for SQL injection attacks.

In comparison, Listing 5 shows that a database client in a scripting language like Python is easier to program. Since SQLite also features a Python driver, the same database created by Go earlier can be used by Listing 5 without further ado. It digs out all database entries whose file paths correspond to a predefined pattern. It expects a regular expression at the command line, stuffs it into an SQL query, and outputs the matches.

Listing 5: like.py

```
01 #!/usr/bin/env python3
02 import sys
03 import sqlite3
04
05 try:
06   _, pattern = sys.argv
07 except:
08   raise SystemExit(
09     "usage: " + sys.argv[0] + " pattern")
10
11 conn = sqlite3.connect('files.db')
12 c = conn.cursor()
13 like = "%" + pattern + "%"
14 for row in c.execute('SELECT path,size FROM files WHERE path LIKE ?', [like]):
15   print(row)
```

More Luxury, More Lines

Go's type checking and the fact that it does not run inside a bytecode interpreter, but as a compiled binary with more elegant memory management than a C or C++ program, has its price: It requires more detailed instructions and generally more lines of code. Go programs run faster than Python scripts, but, as is so often the case, the bottleneck in the use case at hand is not in processing instructions, but in communicating with external systems. In this case, database calls consume most of the compute time. Whether the program code itself runs 10 or 100 percent faster is largely irrelevant.

However, the compact binary format with embedded libraries and no dependency worries is a big advantage, and probably one of the reasons Go has become the first choice for all types of system programming tasks. ■■■

Info

[1] Google Code Search: *https://github.com/google/codesearch*

[2] Russ Cox, "Regular Expression Matching with a Trigram Index," 2012: *https://swtch.com/~rsc/regexp/regexp4.html*

[3] Listings for this article: *ftp://ftp.linux-magazine.com/pub/listings/linux-magazine.com/215/*

MakerSpace

The new PiXtend V2 board at a glance

Balanced Trio

The PiXtend board extends the Raspberry Pi with many useful interfaces and functions for new target groups. *By Martin Mohr*

The PiXtend board [1] opens numerous possibilities for the Raspberry Pi and provides additional flexibility, not only by the hardware, which meets industry standards, but also because of its support for professional software. Recently, the new PiXtend version 2.0 [2] was released (Figure 1).

Alternative

Manufacturer Qube Solutions provides the new board in three forms: as a plain vanilla extension board and as the ePLC Basic and the ePLC Pro. The last two alternatives include a Raspberry Pi 3 (RPi3) and a pre-installed operating system tailored to the respective application.

The ePLC Basic (~EUR240) is intended for easy installation in devices and for developing programs. The Pro version has everything you need for deployment as a professional-grade industrial programmable logic controller (PLC; a digitally programmable device for controlling or regulating a machine or system): a metal case suitable for mounting on a top-hat rail, which explains why the Pro version costs around EUR290. If you are unsure, start with the Basic version and retrofit the housing later. The PiXtend V2 S extension board is aimed at users who already own a Raspberry Pi: Neither the RPi3 nor an operating system are included, but it only costs around EUR165.

The PiXtend V2 is not intended to replace the previous version 1.3 board. Instead, it offers users who want to use the board in control systems a more cost-effective and space-saving alternative. A glance at the technical data of the PiXtend V2 will reveal whether it is suitable for the desired purpose.

Technology

The PiXtend V2 hardware is designed for practical use; the board is well thought out, like its predecessor, and components are almost exclusively surface mount, so that it is a good third smaller than its predecessor and accordingly costs a little less than version 1.3. Table 1 provides an overview of the various inputs and outputs.

The digital outputs deserve special attention: Version 2 works in PNP mode (version 1.3 uses NPN mode). PNP mode is used in industrial controls (PLC standard), and the operating voltage is at the output on switching, whereas in NPN mode, it switches to ground. The advantage of PNP mode is that the systems react in a more fault-tolerant way, because no permanent operating voltage is present at the consumer ends, in contrast to NPN mode.

Another novelty is persistent memory, which can store status information in

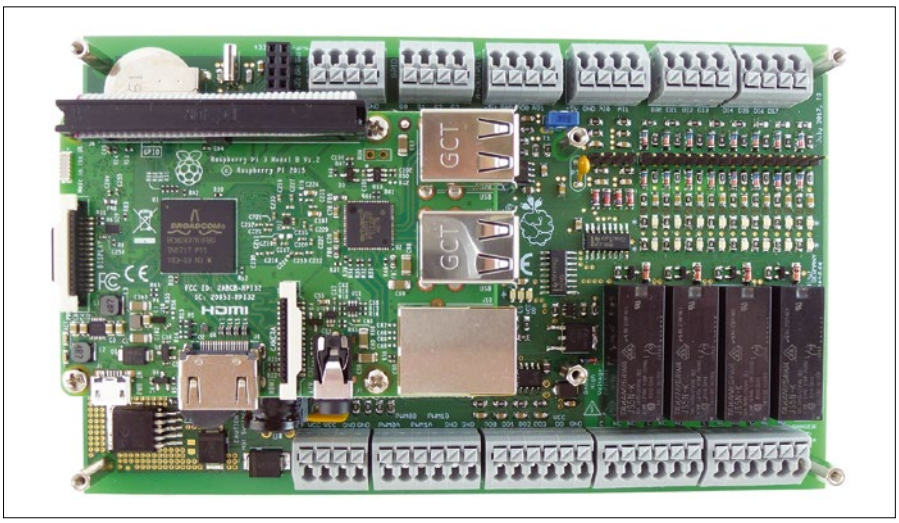


Figure 1: A PiXtend V2 with an assembled Raspberry Pi 3.

Author

Martin Mohr, born in the age of magnetic ring core memory, studied computer science after training as an electronics technician and developed mainly Java applications. The Raspberry Pi reawakened his love of electronics.

the controller permanently, even if you switch it off or the power fails (i.e., an unforeseen power failure means that there is not enough time to write the data to an SD card). If the operating voltage drops below 19V, the PiXtend V2 microcontroller saves the residual data in flash memory in just 5msec. This technology can otherwise only be found in very expensive industrial controls.

Supported Software

Codesys. As with the predecessor, version 2 of the PiXtend board has extensive software support: Codesys [3] is a hardware-independent programming system supported by many manufacturers in the field of professional industrial controls.

Codesys implements the IEC 61131-3 [4] standard and supports the creation of web visualizations that deliver data as HTML from a web server for display on all common browsers. The open Codesys system allows for an easy exchange of programs developed on different hardware.

Linux Tools. The functions of PiXtend V2 can be tested with Linux tools, and the board can be programed in C. SD card images for the PiXtend come with these tools pre-installed, which use Gordon Henderson's WiringPi library, also integrated into the images.

The `pxauto2s` tool provides a graphical interface for the Linux console, where you can read and set all values and settings of the board. It exchanges data with the hardware 100 times per second, which makes it very easy to test the basic functions of the board.

With the `pixtendtool2s` tool, you can change and read out the settings directly in the shell, allowing you to develop small scripts quickly that perform simple control tasks, for example, with FHEM [5], a home automation server that allows easy builds of smart home solutions. FHEM is open source software that controls components from almost all professional manufacturers. If required, you can install a large number of modules for various items of hardware on the server and connect them to each other in a central interface.

Python. The popular Python scripting language is often used for development in the IoT area. It has evolved over the past decades, with libraries for almost every scenario imaginable to make your work easier.

OpenPLC. This completely free software serves as the basis for PLCs. Because it is very easy to install and operate, it is often used in training and studies. The OpenPLC project supports programming language standards according to IEC 61131-3:

- IL (instruction list): text-based, comparable to Assembler.
- LD (ladder diagram): graphical, like an electric circuit diagram.
- FBD (function block diagram): graphical, similar to a logic circuit diagram.
- SFC (sequential function chart): graphical, a kind of state diagram.
- ST (structured text): text-based, high-level language based on Pascal.

FourZero. The FourZero [6] platform supports the development of IoT and automation applications. It abstracts dependencies on the deployed hardware, enables system-level tests, and prevents redundant developments. All these features significantly accelerate the development process and reduce project costs. FourZero uses a decentralized approach.

STEP. The Standard for the Exchange of Product (STEP) Model Data [7] – ISO 10303, the computer-aided design (CAD), manufacturing (CAM), and engineering (CAE) standard for describing product data and partly adopted in German DIN standards – is a text-based CAD format that virtually any design software supports. With the STEP model, it is possible to design the PiXtend board for a control cabinet or a system, making it easier for mechanical engineering companies to integrate the hardware into their systems. The STEP model also makes it very easy for 3D printer owners to create precisely tailored housings for the board.

Table 1: Technical Data

Interface	Details
Digital Inputs	8x (3.3/5/12/24V)
Digital Outputs	4x (max 30V, 0.A each)
PWM/Servo Outputs	4 (2x8, 16 bits each)
Relays	4x (max 230V, 6A)
Analog Voltage Inputs	2x (0-5V/0-10V)
Analog Voltage Outputs	2x (0-10V)
GPIOs	4x (5V)
Serial Interface	RS-232
Real-Time Clock (RTC)	RTC with battery buffer
Sensor Support	Up to 4 DHT11/DHT22/AM2302 sensors (temperature and humidity)
Transmitter (433MHz)	Slot; transmitter not included
Voltage Regulator	Onboard; input 12-24VDC (max 30V), output 5VDC/2.4A (powers PiXtend V2 S, Raspberry Pi, and connected USB devices)
Retain/Persistent Memory	32B flash EEPROM
Compatibility	Raspberry Pi B+, 2B, 3B
Certification	CE (EU consumer safety, health, and environmental conformity), RoHS (restriction of hazardous substances)

Conclusions

PiXtend V2 is geared to the needs of professional users, and version 1.3 is aimed at hobbyists or for educational purposes. Accordingly, Qube Solutions only offers version 2 as a complete tested device, whereas version 1.3 is available either as a kit or as an assembled device. However, the PiXtend V2 S extension board is very suitable for hobbyists and offers an unbeatable price-performance ratio. ■■■

Info

[1] PiXtend shop: *https://www.pixtend.de/shop/index.php?language=en*

[2] Information sheet: *https://www.pixtend.de/files/press/pixtend_v2_s_infoflyer_EN.pdf*

[3] Codesys: *http://codesys.com*

[4] IEC 61131: *https://en.wikipedia.org/wiki/IEC_61131*

[5] "Installing FHEM on the Raspberry Pi" by Jörg Hofmann, *Raspberry Pi Geek*, issue 18, 2016, pg. 32, *http://www.raspberry-pi-geek.com/Archive/2016/18/Installing-FHEM-on-the-Raspberry-Pi*

[6] FourZero: *http://www.automationofthings.com/fourzero-tm/*

[7] STEP: *https://en.wikipedia.org/wiki/ISO_10303*

MakerSpace

Dashboard for RaspPi-controlled toy sailboat with Node-RED

Sail Away

With Node-RED, you can create a web dashboard that instructs a Raspberry Pi to set the rudder position on a toy sailboat. *By Pete Metcalfe*

My daughters and I have built a number of toy boat projects with an assortment of Arduino, ESP8266 (WiFi), Bluetooth, and radio frequency interference (RFI) components, but the version we created for this article using a Raspberry Pi and Node-RED [1] offers one of the simplest solutions. The sailboat uses a basic catamaran design with a Raspberry Pi mounting inside a waterproof container. With Node-RED dashboards, you can control the sailboat's rudder from a smartphone. The complete Node-RED logic comprises only six nodes.

Building the Sailboat

Of the many different building materials from which you could choose, K'Nex construction pieces [2] are lighter than either Lego or Meccano, and they allow you to create reasonably large structures with a minimum number of pieces. If you do not have access to K'Nex pieces, popsicle sticks and cardboard would offer a good low-cost solution.

To build the sailboat we used:

- K'Nex building pieces
- Four plastic bottles
- Small plastic container with a lid
- String
- Duct tape
- Garbage bag
- Low-torque servo
- Raspberry Pi Zero W or 3
- Small USB phone charger

The base of the sailboat was a rectangular structure with 16 water-facing

Lead Image © shamain, 123RF.com

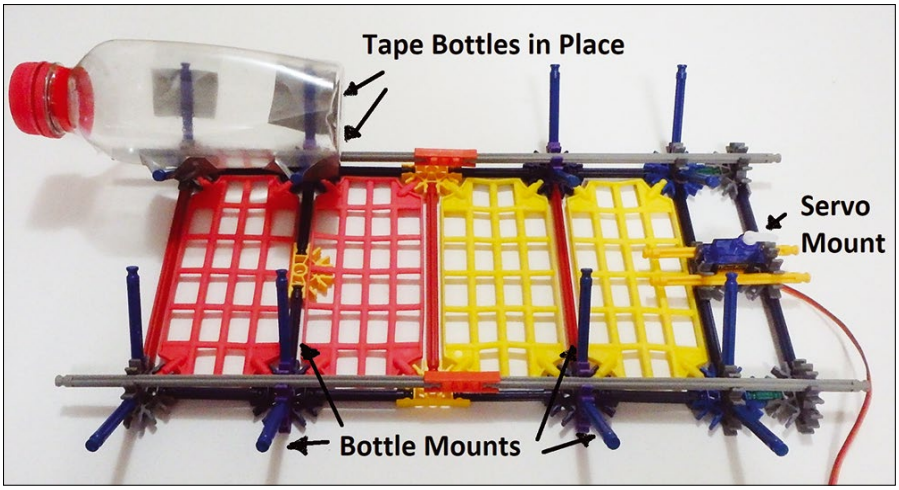


Figure 1: Sailboat base.

Author

You can investigate more neat projects by Pete Metcalfe and his daughters at *https://funprojects.blog*.

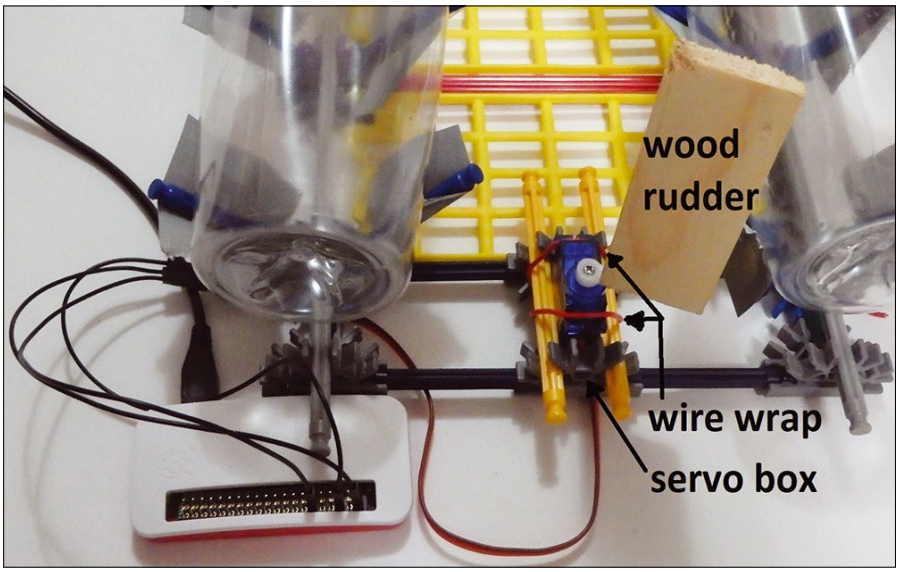


Figure 2: Servo and rudder mounting.

K'Nex pieces that allowed plastic bottles to be duct-taped in place (Figure 1). A few K'Nex pieces created a compartment for the servo, and wire secured the servo in place. A rudder was built by screwing a small piece of wood into the servo arm (Figure 2). A garbage bag was cut to the required size and taped to the mast to form a sail. The boom had a swivel connection to the mast and guide ropes connected to both the boom and mast (Figure 3).

Servo and Rudder

Only very low torque servos can be connected directly to Raspberry Pi GPIO pins (Figure 4). An example of a low-torque servo would be the TowerPro SG90 [3] ($4), which has a torque of 25 oz/in (1.80 kg/cm). If you have servos with greater torque, you will need to use either a custom Raspberry Pi servo hat (there are some good ones on the market) or a separate power and ground circuit for the servo.

The WiringPi `gpio` tool can be used to control the servo. This package is pre-installed on the Raspbian image, or it can be installed manually by entering:

```
sudo apt-get install -y wiringpi
```

Servos typically want a pulse frequency of 50Hz; however, the Raspberry Pi pulse width modulation (PWM) pins have a frequency of 19,200Hz, so some range definitions and scaling are required:

```
gpio -g mode 18 pwm
  #define pin 18 as the PWM pin
gpio pwm-ms  # use "mark space" mode
gpio pwmc 192  # set freq as 19200
gpio pwmr 2000  # use a range of 2000
```

The `gpio pwm` commands are not persistent after a reboot. A simple solution for this problem is to put the above commands in the Pi user login file `$HOME/.bash_login`.

After running the PWM setup commands, you need to do some manual testing to define your various rudder (servo) positions (Figure 5), such as Hard Left, Hard Right, Easy Left, Easy Right, and Straight. The `pwm` timing numbers will vary according to your requirements and the positioning of the servo arm. We used the commands

```
gpio -g pwm 18 200 #straight
gpio -g pwm 18 260 #hard left
gpio -g pwm 18 140 #hard right
gpio -g pwm 18 230 #easy left
gpio -g pwm 18 170 #easy right
```

for our sailboat.

Node-RED Logic and Dashboards

Node-RED is pre-installed on the Raspbian image, but it will need to be set to autostart on a Pi reboot with:

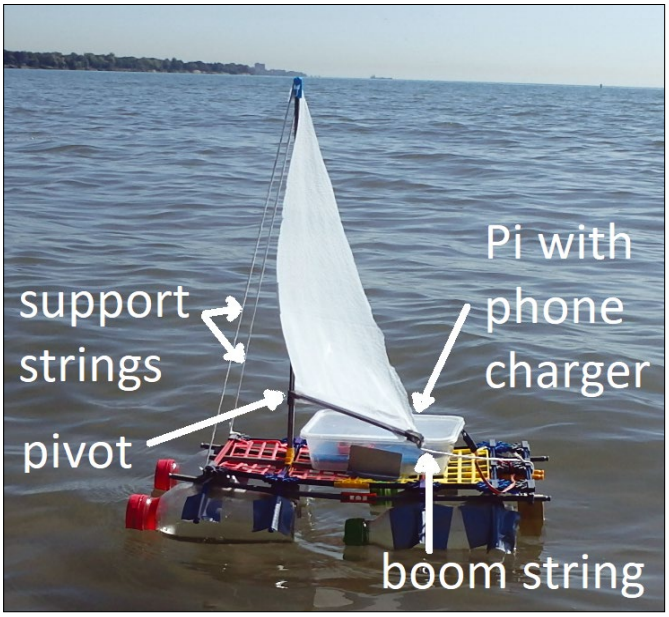


Figure 3: Sailboat top view.

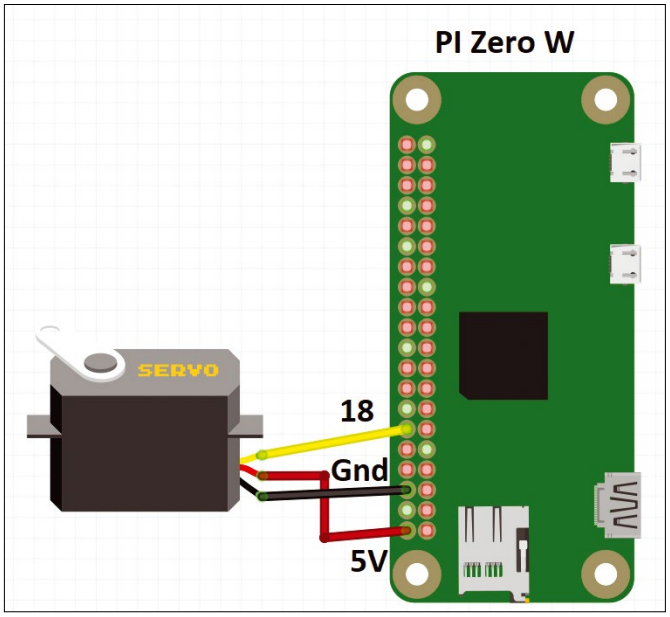


Figure 4: Raspberry Pi Zero W servo circuit.

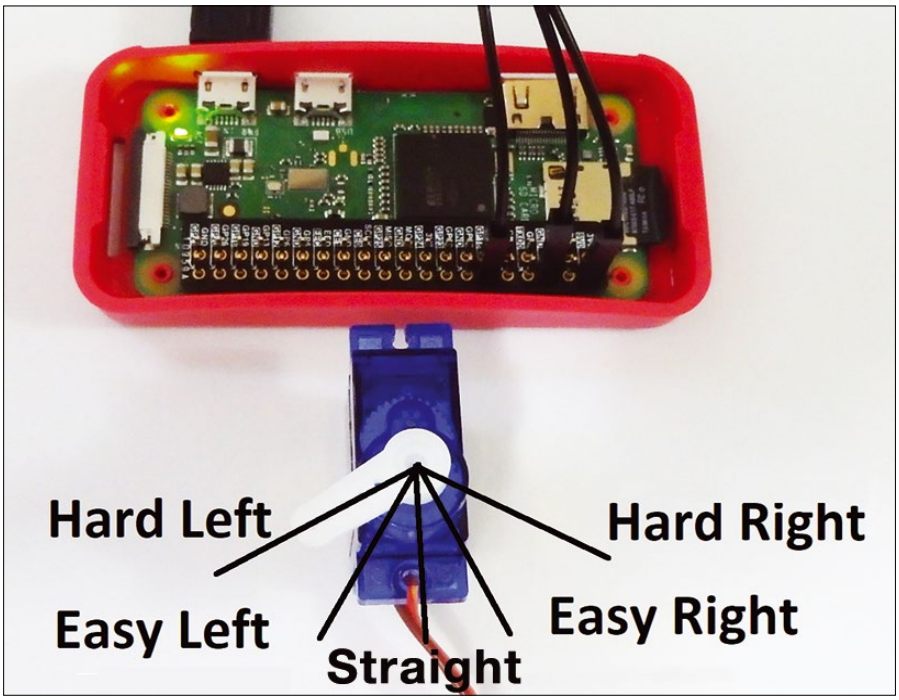


Figure 5: Servo/rudder settings.

```
sudo systemctl enable nodered.service
```

The Node-RED web interface for configuration is accessed at *http://localhost:1880* or *http://<pi_ip_address>:1880*. Selecting *View* | *Dashboard* brings up the options button on the far right (Figure 6), where you can define and change the web dashboard layout. To create logic, nodes are selected from the left node panel and dragged and dropped onto the center flow panel. Logic flow is then created by clicking and joining together the inputs and outputs on the nodes [4]. When a dashboard node is dropped onto the flow panel, it is added to the default web dashboard.

To call the `gpio -g pwm` command, use the *exec* node (Figure 7). The *button* dashboard node will pass the defined payload value (e.g., a *260* is passed when the *Hard Left* button is pushed). The button's payload value is appended to the `exec` command to make a complete servo position command.

Once you have completed your logic setup, press the *Deploy* button on the top right to make your configuration live and ready to test.

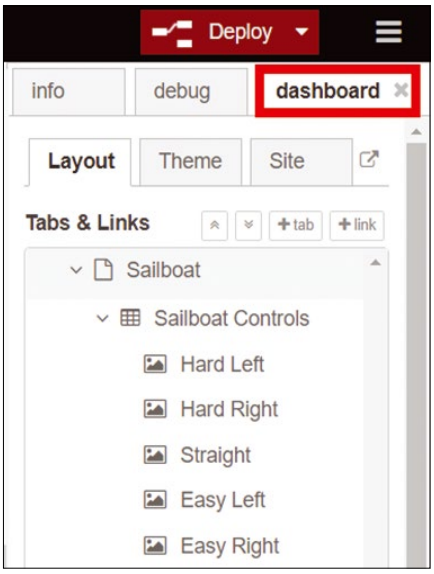


Figure 6: Node-RED dashboard layout.

Mobile Controller

The final step, enabling a smartphone or tablet to connect to the Raspberry Pi, can be accomplished either by making the Raspberry Pi a WiFi access point or by tethering the Pi to a cell phone. You can find some great guides online on how to set up a Raspberry Pi as an access point (e.g., the Raspberry Pi Foundation website [5]). For this project, we used the simple tethering method. Once the Pi is tethered to a phone, the Pi's IP address can be obtained from the *Wi-Fi hotspot users* list

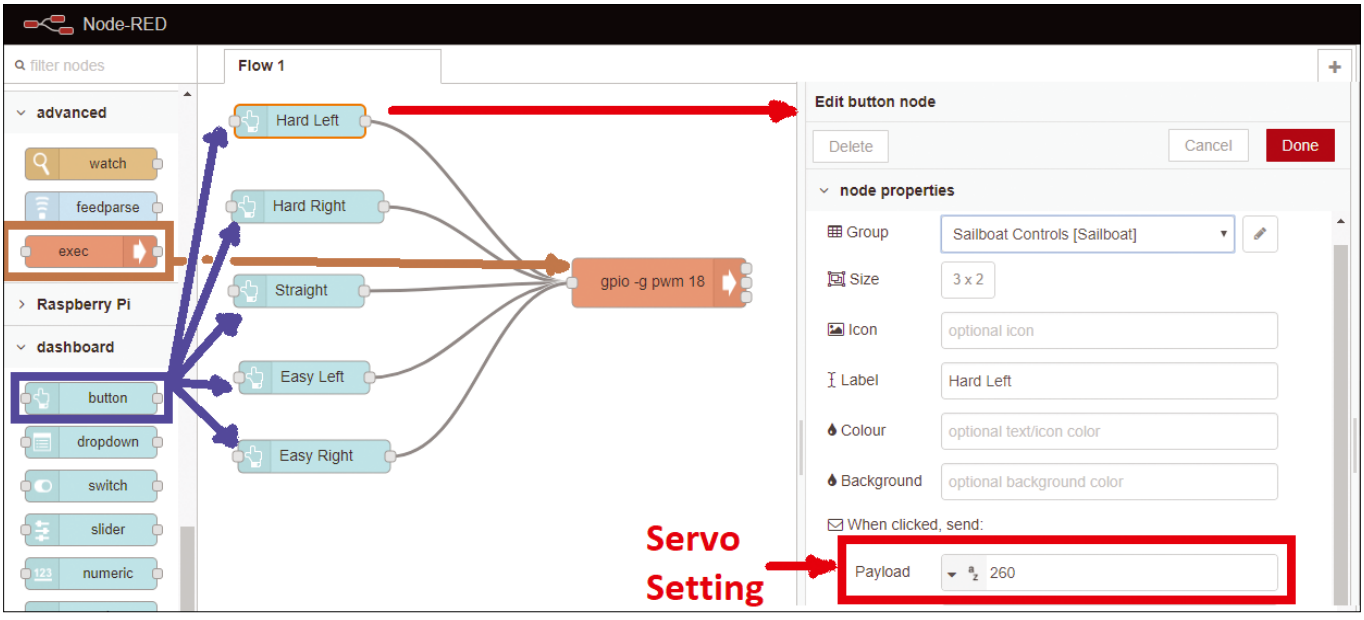


Figure 7: Node-RED logic.

on an Android device (Figure 8). The Node-RED dashboard is accessed on your phone with *http://<pi_ip_address>:1880/ui* (Figure 9). Assuming everything is connected correctly, you should be able to control the sailboat with your phone.

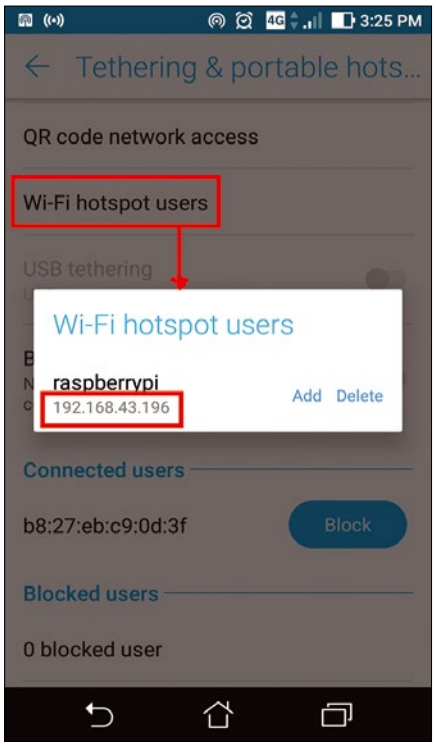


Figure 8: Pi hotspot address.

Figure 9: Node-RED mobile web dashboard.

Summary

Once you have mastered the basic Node-RED programming and sailboat construction, other projects are possible, such as motor boats, iceboats, and airboats [6]. ■■■

Info

[1] Node-RED: *https://nodered.org*

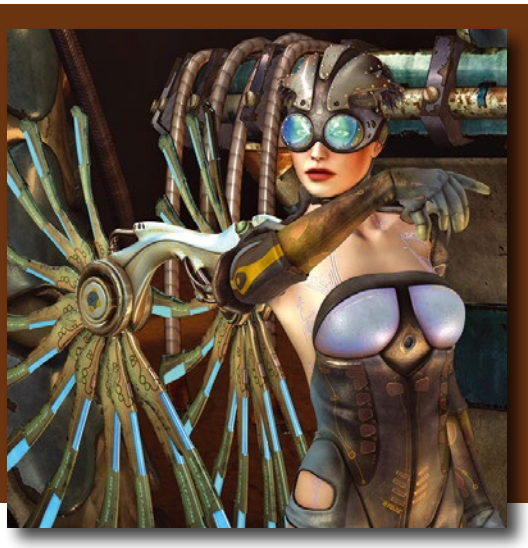
[2] K'Nex: *https://www.knex.com*

[3] TowerPro SG90: *http://www.towerpro.com.tw/product/sg90-7/*

[4] "Create amazing Pi apps without writing code" by Leah, Brooke, and Pete Metcalfe, *Raspberry Pi Geek*, issue 20, 2016, pg. 90, *http://www.raspberry-pi-geek.com/Archive/2016/20/Create-amazing-Pi-apps-without-writing-code*

[5] Raspberry Pi as an access point: *https://www.raspberrypi.org/documentation/configuration/wireless/access-point.md*

[6] "Use your smartphone to control an airboat" by Leah, Brooke, and Pete Metcalfe, *Raspberry Pi Geek*, issue 11, 2015, pg. 94, *http://www.raspberry-pi-geek.com/Archive/2015/11/Use-your-smartphone-to-control-an-airboat*



MakerSpace

Social Hardware's Prosthetic Hand Development Kit

Helping Hands

In true open hardware spirit, Social Hardware looks to produce a development kit for prosthetic hands to help rural amputees in India. *By Bruce Byfield*

India has more than half a million upper limb amputees. Most have no hope of a prosthetic, partly because of a world-wide shortage of prosthetic technicians, but mainly because prosthetics start at $30,000, far more than many amputees can afford. In this crisis, any innovative solutions are welcome, and Cameron Norris and Abhit Kumar of Social Hardware [1] are about to launch a crowdfunding campaign to provide one [2]. The campaign will be for the production of a Prosthetic Hand Development Kit, which will allow backers to build a hand for less than $500 – and, Norris hopes, will encourage buyers to donate their results to amputees in India (Figure 1).

Norris' background is in digital marketing. He adds, however, that "I've always gravitated towards design and technology." In pursuing those interests, Norris went to work for Wevolver, a London-based community site for sharing

Figure 1: The development kit's artificial hand.

open hardware projects [3]. As part of his work, he documented more than one hundred projects, as well as helping projects with community building and licensing issues, all of which helped him to understand the characteristics of successful open source projects.

Norris first became interested in prosthetics after reading a Reddit post about Ryan Cashman, an American who faced the amputation of a hand after a workplace accident [4]. Wevolver made Cashman a modified Exiii HACKberry hand [5] – only to learn that Cashman had opted for a partial hand amputation and was eventually outfitted with a prosthetic for weightlifting by a commercial company. However, in the process, Norris was put in touch with several dozen experts. Just as importantly, Norris concluded that "the method of crowdsourcing ideas and collaborating internationally to solve problems was definitely validated, and I felt optimistic for the future of open hardware."

Later in 2015, Norris took part in the Enable Makeathon [6], a 60-day hackathon in Bengaluru, India. There, he struck up a friendship with biomedical engineer Abhit Kumar, whom he had previously encountered on Reddit. During the hackathon, Norris says, "it became clear that access to prosthetic devices would have a massive impact on the quality of life for rural amputees from low-income communities, but existing solutions were either unhygienic, too expensive, or lacked the durability to function reliably in a rural environment."

Since then, Norris and Kumar have founded a company called Social Hardware and have participated in a startup incubation program. Norris has also served as an advisor on open source community development for Astroplant [7] – a project with the European Space Agency – and as an advisor for Disrupt Disability [8], a project developing open hardware wheelchairs.

Based on their experience, Norris and Kumar plan to use their Prosthetic Hand Development Kit to help to solve the problem with access to reliable prosthetics. According to Norris, the highest rate of amputation in India occurs in rural areas due to work industries. As cars and modern agricultural machinery are introduced, the injury rate continues to rise. The problem is further compounded by a rapidly aging population, who may not have the strength to use body-powered prosthetics.

The Development Kit

According to Norris, "the goals of the development kit are to raise awareness of India's aging, rural amputee population, encourage others to participate in the development of assistive technologies, and raise the money required to provide rural amputees with our prosthesis for free. We hope this will inspire more people to participate in humanitarian innovation by starting new initiatives in support of medically underserved communities elsewhere in the world."

The kit came about because of Norris's efforts to construct a hand for Cashman. Norris explains, "My own experience illustrated just how difficult it can be to source all of the required parts for a functioning device. Incomplete or overly complex documentation added to the confusion, forcing me to improvise during assembly, often with mixed results. With this in mind, the secondary goal of our kit is to ensure that open source hardware development is as pain-free and accessible as possible, so that developers can focus on learning and innovating, rather than scouring the web for discontinued electrical components they may never find."

The kit is aimed at students and hobbyists and is intended to orient them toward the resources they will need to construct a hand. Its contents will include links to online documentation and videos, as well as assembly guides, design files, open source software, motors, an Arduino-compatible ATmega328 control board, a power bank that can be used as an emergency power supply, sensors, and all the parts required to assemble a single device. Each finger will have three degrees of freedom – that is, each will be able to move in three directions.

The resulting hand will be made from antimicrobial silicone rubber, making it hygienic as well as resistant to ultraviolet light, dust, stains, and water. Since silicone rubber is heat resistant up to 300°C, the wearer of the hand will be able to use it around an open fire without the risk of melting a finger – which, Norris notes, "is possible with your typical 3D-printed hand due to the use of thermoplastics with a relatively low melting point. From the beginning, our aim has always been to develop a hand that is as durable and maintenance-free as possible, as lack of adequate durability is a common complaint among battery-powered prosthesis users internationally."

The kit will also include a free prosthetic design course designed with help from Autodesk on the subject of creating custom sockets for attaching the hand to the body. This feature is necessary because all amputees do not have the same amount of a limb remaining. However, Norris warns that the construction of a custom socket "should be for educational or research purposes only, unless a certified prosthetist is available to carry out the socket fitting." For this reason, the project is partnering with the Association of People with Disabilities to provide free on-site socket fitting and physiotherapy. In this way, the kit users will not need to attempt a task that is likely to be beyond their training and experience.

The finished unit will be compatible "with most standard M12 prosthesis controllers," [9] Norris says, "such as the widely used Ottobock 10V8 Movo [10], enabling patients to attach the device directly to their existing upper limb socket. Additionally, the hand can be controlled by virtually any sensor with a 3.5mm output jack, including commercial EMG sensors and open source alternatives, such as the low-cost IR sensor developed by Masahiro Yoshikawa from the Osaka Institute of Technology [11].

However, because keeping the kit's price low is a priority, the resulting hand will have only two motors and will only be able to control the index finger individually. These limitations mean that the hand will have only a limited number of possible grip patterns – although Norris does point out that the functionality will be roughly equivalent of an Exiii HACKberry. However, Norris hopes that kit users will extend the functionality of the hands (Figure 2).

Additionally, any changes made to the original design will have to be individually certified. However, Norris says, "We will provide support to any developers seeking to do this."

For the Future

The basic kit will undergo a large pilot study in rehabilitation centers in Gujarat, Karnataka, Rajasthan, Punjab, and West Bengal later this year. Based on the results of the pilot study, Norris hopes that the basic hand will be medically certified by early 2019. Future plans also include reaching out to international aid agencies and manufacturing the kits locally in India. As Norris points out, local manufacturing will decrease costs for the agency and allow quicker delivery while strengthening the local economies.

The problems the kit is designed to address are not simple, going far beyond the obvious medical ones. Currently, more than 70 percent of India's population is rural with limited access to prosthetics. Just as importantly, most of the prosthetics used in rural India are made in Europe and North America, costing far more than an average family's income. Nor are the available prosthetics designed for a rural lifestyle. In effect, many rural Indians have no access to suitable, much-needed prosthetics. The proposed kit and its implementation should go a long way to alleviating these problems. I can't help thinking that open source will play exactly the role for which it was designed. ■

Figure 2: The Social Hardware hand will be capable of several different grips.

Info

[1] Social Hardware: *https://www.socialhardware.in/*

[2] Crowdfunding campaign: *https://www.crowdsupply.com/social-hardware*

[3] Wevolver: *https://wevolver.com/discover*

[4] Ryan Cashman: *https://inside3dprinting.com/news/reddit-volunteers-team-to-build-3d-printed-weight-lifting-prosthesis/32548/*

[5] Exiii HACKBerry: *https://myhumankit.org/en/tutoriels/myoelectric-exiii-hand/*

[6] Enable Makeathon: *http://www.enablemakeathon.org/*

[7] AstroPlant: *https://www.astroplant.io/*

[8] Disrupt Disability: *https://www.disruptdisability.org/#intro1*

[9] Prosthesis controls: *https://en.wikipedia.org/wiki/Robotic_prosthesis_control*

[10] Ottobock 10V8 Movo: *https://professionals.ottobockus.com/Prosthetics/Upper-Limb-Prosthetics/Body-Powered-Systems/Movo-Wrist-Units/Wrist-Units/Wrist-Unit--Ratchet-Type-Rotation/p/10V8*

[11] Masahiro Yoshikawa: *https://www.researchgate.net/profile/Masahiro_Yoshikawa2*

Welcome to Linux Voice. This month we feature some tools that bring new powers to familiar technologies. You probably know you can compress the data on your hard drive. But what about RAM memory? Data compression is more efficient than it used to be, and at today's processor speeds, the performance penalty is quite small. Before you add more memory to your system, why not try data compression with RAM? It might not work for everything, but you could save some money or prolong the useful life of a memory-challenged system. Also inside, we show you the Pass password management tool. Sure you can use Git for managing code, but did you know you can use it for managing passwords? We also help you get started with Docker and describe some scripts and tricks for working with LibreOffice metadata.

Image © Olexandr Moroz, 123RF.com

LINUXVOICE

Doghouse – Linux History 67
Jon "maddog" Hall
"maddog" takes us on a brief tour of Linux history.

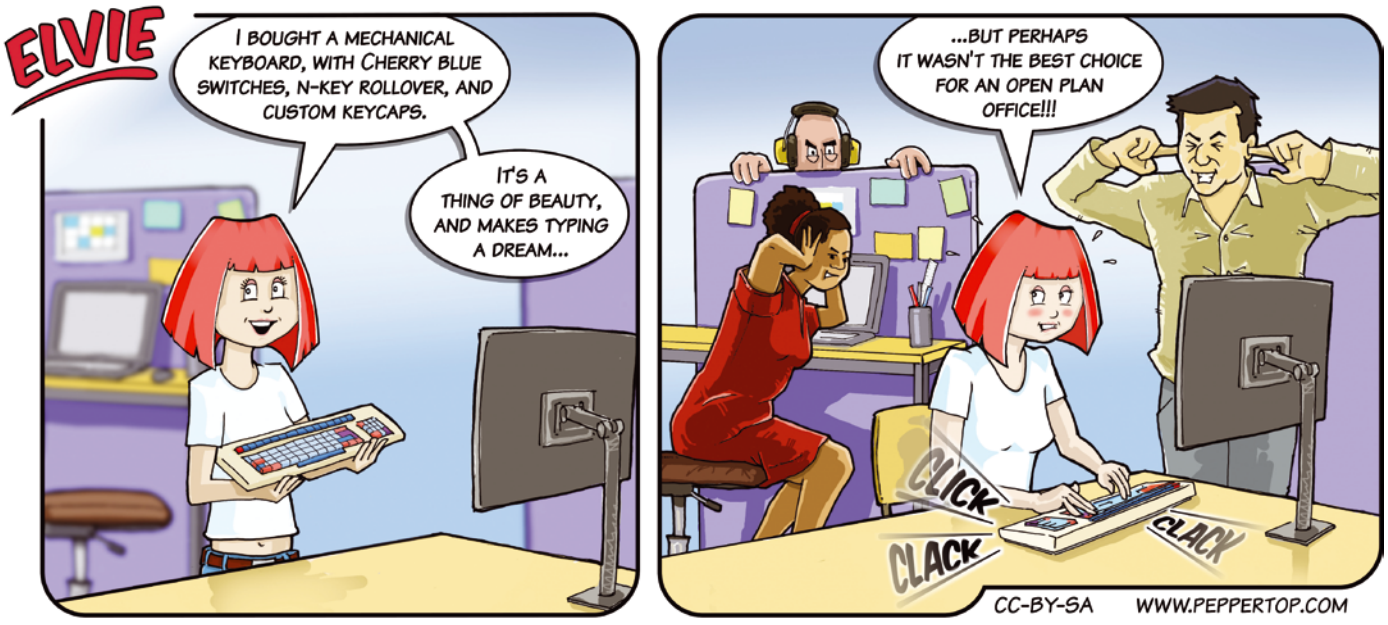
Memory Compression 68
Frank Hofmann and Mandy Neumeyer
Data compression costs virtually no computing power today. Why not save some space by putting data compression techniques to work on RAM and cache memory?

Pass 72
Andreas Bohle
This simple shell script helps you manage and synchronize passwords using Git.

FOSSPicks 76
Graham Morrison
This month Graham looks at Brackets, Browsh, Borderlands, Timekpr, Fractal, HyperRogue, and much more!

Tutorials – Metadata 82
Marco Fioretti
ODF files contain useful metadata that is very easy to read or modify.

Tutorials – Docker 101 90
Paul Brown
The Docker container management system isn't just for sys admins. Here's how to get started implementing your own Docker container environment.



Now Appearing on
APPLE NEWSSTAND

New age convenience...

Our inspired IT insights are only a tap away.

Look for us on Apple Newsstand and the iTunes store.

Download a FREE issue of each publication now!

MADDOG'S DOGHOUSE

Jon "maddog" Hall is an author, educator, computer scientist, and free software pioneer who has been a passionate advocate for Linux since 1994 when he first met Linus Torvalds and facilitated the port of Linux to a 64-bit system. He serves as president of Linux International®.

"maddog" takes us on a brief tour of Linux history.

BY JON "MADDOG" HALL

Life of Linux

I am writing this article in Montevideo, Uruguay, on the 68th anniversary of my birthday in 2018. It also happens to be the month that this magazine offers all 200+ issues on a DVD, dating back to October 2000, which brings back lots of memories for me.

These issues do not reach back to the beginning of the Linux kernel. No magazine does, because when Linus started the kernel project in 1991, it was "just for fun." Few (if any) people imagined that this fledgling project would significantly affect computing.

After all, in 1991, Microsoft ruled the desktop (with a relatively few "apples" thrown in) and proprietary systems (MVS, VMS, Unix, and others) ruled the data center. The idea that a university student would someday challenge these proprietary operating systems created by companies like IBM, Digital Equipment Corporation, Sun Microsystems, and others with a kernel written by a gang of "amateurs" was absurd. Microsoft was busy trying to advance their wunderkind WNT (Windows NT), and even the publisher O'Reilly seemed to have given up on Unix systems, producing more and more books about programming with Microsoft.

By technical standards, Linux was not the greatest of operating systems in the beginning. A single-CPU, 32-bit operating system with a filesystem that was relatively weak, the kernel took until 1994 to reach V1.0 status and be considered useful.

When symmetric multiprocessing (SMP) was started in version 2 of the kernel, it used the Big Kernel Lock (BKL) to manage the serialization of critical sections, whereas other operating systems like Digital Unix had fine-grain control of these sections.

Real time was also poor, as many of the kernel sections were not deterministic in the amount of time they would lock out interrupts.

A friend of mine who owned a firm that helped customers determine which commercial Unix systems would be useful to them called Linux a "toy" when I asked him about it in 1995.

Certainly another issue was a lack of applications. Hard-core Unix people could use GNU/Linux (the distributions started coming out in 1994) for their own work, but there were no "real" applications in the beginning.

However, a lot of Unix system administrators discovered that they could use GNU/Linux to repurpose older hardware, giving it new life as DNS servers, firewalls, thin clients, and other devices, saving them from having to buy additional hardware to do these tasks.

In 1994, another major event happened: the invention of the Beowulf-style supercomputer, what we call "HPC" today. Two men from NASA, Donald Becker and Dr. Thomas Sterling, conceived of and implemented this method of replacing far more expensive and complex supercomputers with many (often cast-off) PCs hooked together with "high-speed" networking.

This innovation created a flood of people who found they could now afford supercomputers made from "cheap PCs" to do a lot of the work for which Crays had been purchased.

It also created a nice market for the Linux/Alpha port, which started in 1994 and was finished nine months later; now, people could have Beowulf systems with 64-bit address spaces to process huge amounts of data. The Alpha port had another main feature in the history of Linux. It forced Linus Torvalds to make sure the kernel sources could support at least two architectures and, in fact, be set up to support *N* architectures.

Other applications started to show up, but probably one of the most significant was when the commercial databases started to arrive in 1998. Informix, Oracle, Sybase, and other closed-source databases joined the more "open" versions, and GNU/Linux could now be used in a database engine package. The database companies supporting their products on GNU/Linux made other companies sit up and take notice.

The next major step in the life of Linux came in the year 2000, with the advent of networked embedded systems.

Up until the 2000, most embedded systems were proprietary, closed-source systems written from scratch by companies for a specific market or platform. Memory was still very expensive, processors were still relatively slow, and, if the units were "networked," it was typically serial lines on a private network. No really sophisticated networking was needed.

In the late 1990s the need arose for these systems to join the Internet and speak TCP/IP. To fit a network stack onto these proprietary OS versions, and at the same time port all this code (some of which was in assembler language and very non-portable), would have been a lot of work.

Fortunately, there was already an operating system that worked on these chips, had the compilers needed, had the networking stacks, and was easily licensable and free of royalties. Almost overnight, Linux became the most-used operating system in new embedded designs.

Now, on the DVD in this magazine, you can read the rest of the story. ■■■

Deflated

Data compression costs virtually no computing power today. Why not save some space by putting data compression techniques to work on RAM and cache memory?

BY FRANK HOFMANN AND MANDY NEUMEYER

Random access memory (RAM) is used to store data for ongoing operations. RAM has an ultra-short access time, but it is comparatively expensive to buy. Moreover, you cannot extend RAM infinitely, because the board can only take a certain size and only offers a fixed number of slots for it. Additionally, memory modules for older devices are sometimes difficult to obtain.

Instead of adding more RAM to the computer, another approach is to make more data fit in the RAM you already have. Current computers are fast enough that on-the-fly data compression hardly makes a difference to the total execution time. Data compression started as a technology for archiving data on hard disks, but you can also use compression for data in RAM memory.

The three methods, zram, zswap, and zcache, provide three implementations that extend content compression to RAM. These features made it into the main kernel branch [1] in the 3.14 kernel, but haven't received much everyday use. Only Android 4.4 and Knoppix 7 or higher activate zram by default.

Inventory

Before using compression technologies, you should know how much memory your system actually has. You don't have to open the hood and physically check your computer – software tools can reliably detect and display the values. At the command line, `dmidecode` [2] does the trick; in a graphical environment you can use HardInfo and `lshw-gtk`.

`dmidecode` relies on the Desktop Management Interface (DMI) [3] to discover information about the built-in system components like the CPU, the motherboard, and the memory. To retrieve only the RAM information, call the program with the `--type memory` parameter. Listing 1 shows the output, which comes from a computer equipped with a single 8GB DDR3 RAM module that runs at 1,600MHz. The overview shows that the maximum capacity on the board is 16GB – another 8GB module would be possible.

Listing 1: Retrieving RAM Information

```
$ sudo dmidecode --type memory
# dmidecode 2.12
SMBIOS 2.7 present.

Handle 0x0007, DMI type 16, 23 bytes
Physical Memory Array
Location: System Board Or Motherboard
Use: System Memory
Error Correction Type: None
Maximum Capacity: 16 GB
Error Information Handle: Not Provided
Number Of Devices: 1

Handle 0x0008, DMI type 17, 34 bytes
Memory Device
Array Handle: 0x0007
Error Information Handle: Not Provided
Total Width: 64 bits
Data Width: 64 bits
Size: 8192 MB
Form Factor: SODIMM
Set: None
Locator: ChannelA-DIMM0
Bank Locator: BANK 0
Type: DDR3
Type Detail: Synchronous
Speed: 1600 MHz
Manufacturer: Samsung
Serial Number: 25252105
Asset Tag None
Part Number: M471B1G73DB0-YK0
Rank: Unknown
Configured Clock Speed: 1600 MHz
```

Figure 1 shows the output from Hardware Lister (`lshw-gtk`). The data comes from a virtual machine running Debian 9, which only has about 2GB of RAM.

Two programs, `free` and `htop`, help you find out how much RAM the running system is currently using. Figure 2 shows `free` in action. The call was made with the `-h` parameter (long form `--human`), which makes the output easier for people to read. In this case, a total of 2GB of RAM is available, of which the running processes use 356MB.

RAM is a valuable commodity. The more working memory you have, the more overhead providing the matching number of memory pages (memory units provided by the kernel for processes) for the individual applications is involved. Memory pages that have not been used for a long time are stored in swap space (see the "Swap" box), which is usually on (slower) mass storage. This is exactly where the three methods mentioned earlier enter.

The zram, zswap, and zcache methods help you use available memory cells more effectively. Zram (formerly known as compcache [4]) creates a compressed partition as a block device in RAM that can be used as a virtual swap space or as a RAM disk. Zswap [5] employs a compressed buffer that can be used as a write-back cache for physical swap space. This delays or prevents access to the mass storage, reducing the I/O load. Zcache serves as a back end for a specific virtual RAM type that can be used to buffer filesystem contents or swap data.

In Detail

Zram [8] lets you move compressed data directly in RAM. If no more memory pages are available, the kernel tries to keep the data on this block device. This procedure doubles or triples the corresponding storage capacity.

There is currently no package available for Debian, so you will need to set up such a virtual swap device manually (Listing 2). First load the `zram` kernel module (line 1). The `num_devices=2` statement ensures that two devices, `/dev/zram0` and `/dev/zram1`, are created. The first serves as an additional swap partition in the following example, the second simply as a compressed directory in RAM.

Line 2 shows how to find the next available zram device and assign it a size of 512MB. The `zramctl` command from the *util-linux* package is used here. Use the `mkswap` command to create the device as a swap partition (line 4); this assigns a unique identifier in the form of a UUID (line 6). Calling `swapon /dev/zram0` finally mounts the device as additional swap space (line 7).

The approach for integrating `/dev/zram1` as a compressed directory in memory is similar [9]. First use `zramctl` to determine the next device; then create a file system on it. The ext2 type is

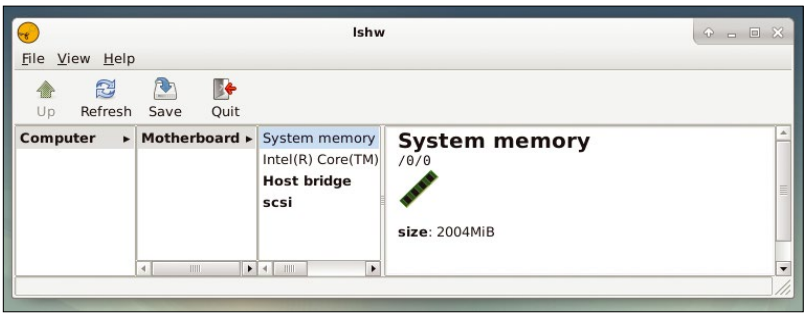


Figure 1: Hardware Lister provides information about a PC's components, including the specifics of the main memory.

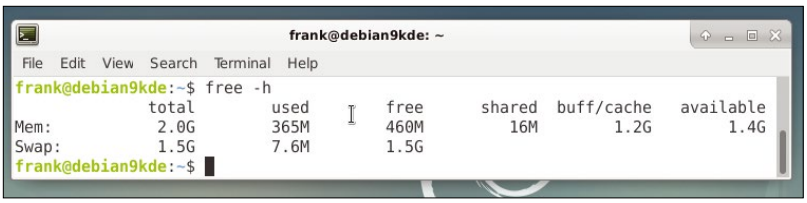


Figure 2: The `free` command-line tool helps you determine how much RAM a system has and how much of it is used by running processes.

Swap

The term "swap" stands for the external storage or outsourcing of the main memory's contents. Swap [6] only works in conjunction with a swap partition or a swap device [7]. As soon as the RAM is full, the system removes the least recently used memory pages. This is based on the assumption that the data will not be needed again so quickly. The data is stored in the swap space without compression. If these data are accessed, it takes much longer, because the system first has to reload the data from the swap partition instead of being able to access the computer's working memory directly. However, if the system works without swap and the memory fills up, the kernel terminates programs at its own discretion to free up RAM again. This usually results in unwanted data loss.

Listing 2: A Virtual Swap Device

```
01 $ sudo modprobe zram num_devices=2
02 $ sudo zramctl --find --size 512M
03 /dev/zram0
04 $ sudo mkswap /dev/zram0
05 Setting up swapspace version 1, size = 512 MiB (536866816 bytes)
06 no label, UUID=009222e1-d138-41aa-8656-64a2c3655004
07 $ sudo swapon /dev/zram0
08 $ sudo swapoff /dev/zram0
09 $ sudo zramctl --reset /dev/zram0
```

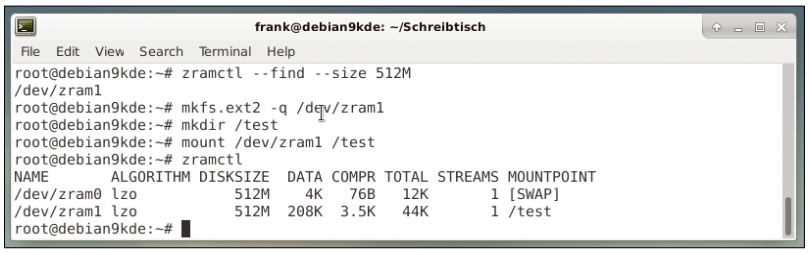


Figure 3: With just a few commands, you create a zram device, which you then format and mount in any directory.

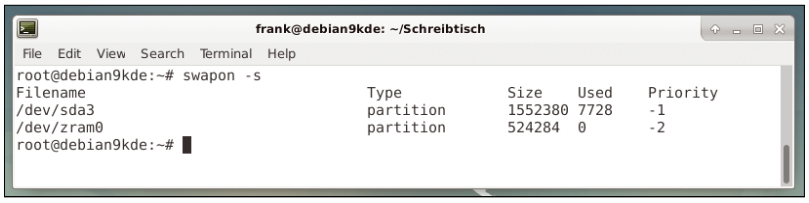


Figure 4: The `swapon -s` command shows you which swap partitions are mounted on the system.

fine here, since you do not need journaling RAM. Then, assign the new device to a mountpoint. Figure 3 shows the procedure and, in particular, the output from `zramctl`. The columns in the output include the device names, the compression algorithm used (*lzo*), the device's disk space, the data (uncompressed, compressed, and total), the number of compression streams, and the respective mountpoints.

Figure 4 shows the call to `swapon -s`. The `-s` switch (the short form of `--summary`) displays an overview of all currently mounted swap partitions – in this case, `/dev/sda3` and `/dev/zram0`. The output contains five columns: the name of the file or device, the type, the size, the bytes in use, and the priority. The higher the priority, the more important the partition is when swapping out.

To remove the `/dev/zram0` swap partition and clean it up in memory, use the commands from the last two lines of Listing 2.

Ubuntu users have a package named *zram-config*. After installing this with the package manager, set up the zram device on the fly. To do so, use `systemctrl` to start the `zram-config` service (first line of Listing 3).

The new `/dev/zram0` device is then immediately available as an additional swap partition, as shown by the `swapon -s` command (Figure 5). The software automatically allocates half of the physical memory to the swap device. Currently, no other value can be set when calling the service or as a configuration argument. If you want to remove the swap device in RAM, the call from the last line of Listing 3 is all it takes.

Swap Becomes zswap

Whereas you have additional swap space on a zram device to provide more space for outsourcing, zswap takes a different approach: This kernel function creates a compressed cache and is useful for all installations that already have a swap area.

Instead of moving memory pages to a corresponding device, zswap compresses the data first; then, it stores the data in a dynamically allocated memory pool. The advantage of this method is that the system thus postpones or even completely avoids writing back to the swap device. Writing and reading in the cache is far faster than from a swap device on a data carrier.

If you have a recent kernel, this function will be included and can be enabled as a module. Figure 6 shows this for Debian kernel v4.9.x. If necessary,

Listing 3: Installing zram-config

```
$ sudo systemctrl start zram-config
$ sudo systemctrl stop zram-config
```

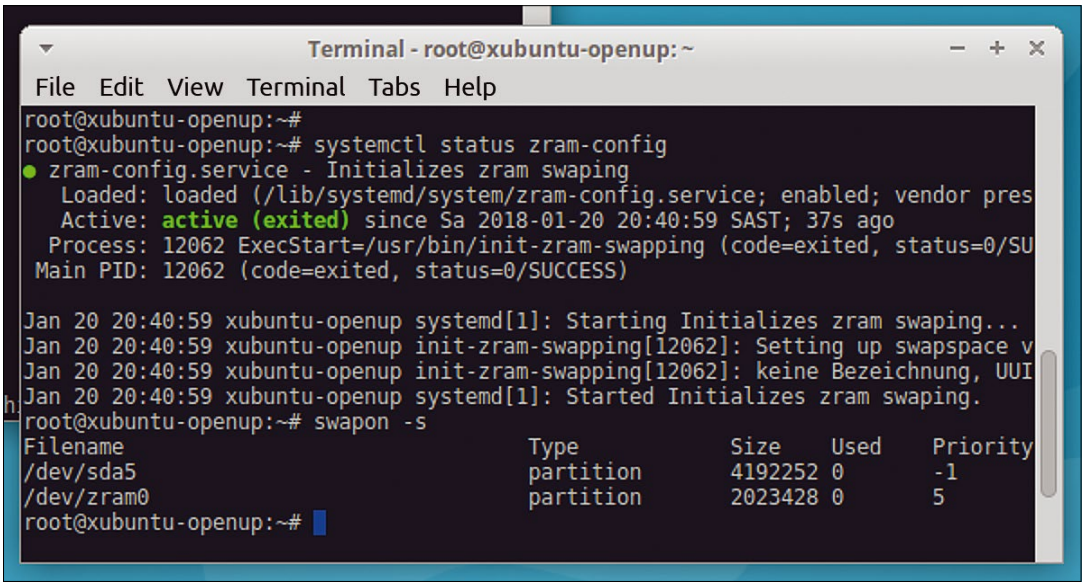


Figure 5: Under Ubuntu, the *zram-config* package automatically provides additional swap space.

just stipulate `zswap.enabled=1` in the GRUB configuration to enable the feature.

zcache

Finally, the third candidate, zcache [10], is a back end for a special type of virtual RAM known as transcendent memory. It is used for compressed intermediate file storage or for swapping out such files. Zswap is a simplified version of zcache. However, zcache is currently considered experimental. You activate this kernel feature by adding the `zcache` parameter to the GRUB configuration and rebooting the system.

Conclusions

All three methods are designed to avoid access to slower media. The three technologies achieve this with noticeable success; no impairments occurred in run-time tests. Whereas zram creates its own block device, zswap and zcache do without one and are already integrated into the Linux kernel as modules. On the basis of our observations, all three methods work with the Lempel-Ziv-Oberhumer (LZO) compression algorithm, but in principle, they also support alternative methods. A more effective use of memory can offer significant benefits, especially with smaller devices. ■■■

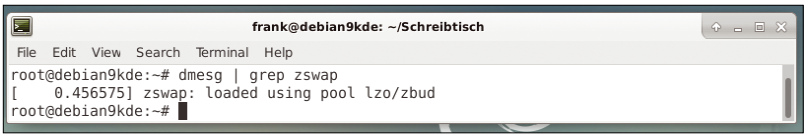


Figure 6: You can use zswap to compress data in the kernel first, instead of storing the data directly on a swap device.

Acknowledgments

The authors thank Gerold Rupprecht and Axel Beckert for their suggestions and appraisals in the run-up to the article.

The Authors

Frank Hofmann works from Berlin, Geneva, and Cape Town as a developer, trainer, and author. He is also the coauthor of the *Debian Package Management* book (*https://www.dpmb.org/index.en.html*). Mandy Neumeyer has lived in South Africa for nine years, works in tourism, and is currently building up additional income as a digital nomad.

Info

[1] zram: *https://en.wikipedia.org/wiki/Zram*

[2] "How to get hardware information with dmidecode command on Linux": *https://www.tecmint.com/how-to-get-hardware-information-with-dmidecode-command-on-linux/*

[3] DMI: *https://www.dmtf.org/standards/dmi*

[4] compcache: *https://code.google.com/archive/p/compcache/*

[5] zswap: *https://www.kernel.org/doc/Documentation/vm/zswap.txt*

[6] "Linux kernel memory management: Swap space": *https://linuxhint.com/linux-memory-management-swap-space/*

[7] "Optimizing Linux memory usage": *https://linuxhint.com/optimizing-linux-memory-usage/*

[8] zram: *https://wiki.gentoo.org/wiki/Zram*

[9] zramctl: *http://karelzak.blogspot.co.za/2014/08/zramctl.html*

[10] "How do I use/enable zcache?": *https://askubuntu.com/questions/300685/how-do-i-use-enable-zcache*

Secret Pass

Pass is a simple shell script that helps you manage and synchronize passwords using Git. BY ANDREAS BOHLE

If you are like many Linux users today, you have dozens of online accounts for different services and websites. If you care about security, you are really not supposed to give those accounts the same password; nor is it considered safe to write all your passwords down on paper and store it in your desk drawer. Password managers evolved as a means for letting users store their passwords safely. Password managers differ in form and functionality, but the basic idea is that the password manager stores your passwords in a safe, encrypted format, and you provide one password to gain access to the password store. Instead of having to memorize all your passwords, you just have to memorize one.

Pass [1] is a lean password manager that relies on some classic Linux tools to help you generate and manage your passwords.

What Is Pass?

Pass is basically just a shell script that bundles various tasks involved in managing passwords into functions that are easy to use. The Pass script integrates a number of other tools that make it possible to back up data using strong cryptography, to copy passwords directly to the clipboard, or to manage the entire database using a version control system.

Pass uses GnuPG to encrypt files, Git to manage the files, and `xclip` to copy them to the clipboard. When installing Pass, most package man-

Listing 1: The tree Tool

```
$ pacman -Si pass
Repositorium              : community
Name                      : pass
Version                   : 1.7.1-1
Description               : Stores, retrieves, generates, and synchronizes passwords securely
Architecture              : any
URL                       : https://www.passwordstore.org/
Licences                  : GPL2
Groups                    : None
Provides                  : passmenu
Depends on                : xclip  bash  gnupg  tree
Optional dependencies     : git: for Git support
                            dmenu: for passmenu
                            qrencode: for QR code support
Conflicts with            : passmenu
Replaces                  : passmenu
Download size             : 18.33 KB
Installation size         : 45.00 KB
Packer                    : Lukas Fleischer <lfleischer@archlinux.org>
Created on                : Fr 14 Apr 2017 10:31:38 CEST
Verified by               : MD5 sum  SHA-256 sum Signature
```

Table 1: Key Variables

PASSWORD_STORE_DIR	Folder for password files
PASSWORD_STORE_GENERATED_LENGTH	Standard length of passwords generated
PASSWORD_STORE_ENABLE_EXTENSIONS	Enables extensions
PASSWORD_STORE_EXTENSIONS_DIR	Folder for extensions

agers automatically update the existing GnuPG files. In addition to `xclip`, the program usually requires a tool for displaying directory hierarchies (`tree`) (Listing 1).

GnuPG lets you create a key suitable for signing or encrypting data, and Pass takes advantage of this function. It is still a somewhat complicated matter to create a key correctly and then keep it safe. However, various pages on the web can help you get started with the subject [2].

Initializing

If you call the tool with the `init` parameter, it also requires the ID of the GnuPG key that you will be using to encrypt the files. Use the `-p <directory>` option to specify that the key only applies to a subtree of the password collection. You can separate business passwords from private passwords in this way.

All the files you manage with Pass are located below `.password-store/` in your home directory. If you want to use a different directory, set the environment variable `PASSWORD_STORE_DIR` accordingly (see Table 1).

If you already use a password management program, the Pass website is well worth a visit: You will find a whole range of tools and scripts that help you export data from other programs, such as Revelation or KeePass. In our lab, exporting from Revelation worked without any problems. In each of the files, the additional entries were located in one line.

Pass only evaluates the first line of the file in which it writes the password for the respective access case. Further information is only used to note down usernames or URLs and other data.

If you are starting from scratch, call Pass with the `generate` parameter, a name for the access case, and a number for the desired password length:

```
$ pass generate test/access 23
```

In the example, the password would be 23 characters long, and the corresponding file would end up in the `access` file below the `test/` password directory subfolder. If you call Pass for this access case, the software prompts you for the password to access the GnuPG key. If you enter this correctly, Pass opens the file and outputs the content to your standard output.

The `-c` option copies the content directly to the clipboard, from where you can paste it again with Ctrl+V. The XSel program is used for this step. If you enter a number as the option parameter, the program tries to read the corresponding line and copy it to the clipboard. In this way, a username could also be read from the corresponding file.

Over time, a volume of access data will tend to accumulate in this way. To organize the data, simply create subfolders in the root folder (`.password-store`) and move the files with the data accordingly. If you call Pass without parameters in the terminal, it shows exactly this tree structure (Figure 1). The names of the files and directories correspond to the entries for the nodes. However, the tool does not include the `.gpg` extension, thus making it easier to read.

Properly Managed

If you use more than one computer, you might not always store the access data on the same computer. It is also easy to imagine circumstances in which a new account becomes necessary while you are traveling. The data you need is then on your laptop's hard disk, but you might want to continue working on your home PC later on.

A whole series of approaches are now available to help you keep a synchronous set of data across all systems; after all, Pass works with simple files. Tools like `rsync` work this way: In a single step, you copy either from the mobile computer to the stationary PC or vice versa. Synchronizing in the opposite direction always requires a second step, which could be done in

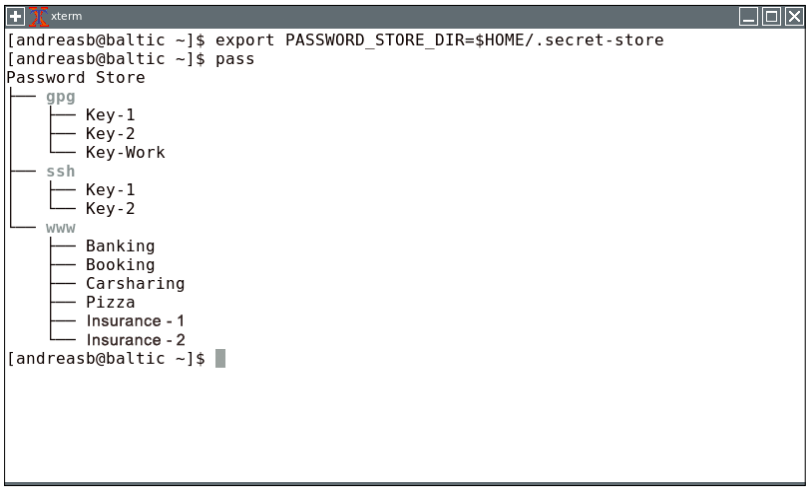


Figure 1: Once you have accumulated various access datasets, you can easily organize them in a folder structure that Pass displays as a tree when called without parameters.

TIP
Make sure you don't forget the `git` parameter; otherwise, Pass will attempt to reinitialize the password memory.

Listing 2: Creating a Repository

```
$ mkdir -p repos/password-store
$ git --bare init repos/password-store
```

Listing 3: Adding a Remote Repository

```
$ pass git init
$ pass git remote add origin pi@storage:repos/password-store
$ pass git push
```

one go with Unison [3], but it can be done even more elegantly.

At this point, the Git [4] version control system (VCS) enters the game. Git's real purpose is to manage source code and, often additionally, the documentation and other files that come with programs. However, the software is designed in such a way that it does not really matter what kind of data you manage with it.

Pass allows you to use Git to set up your own commands. You can synchronize files and, if desired, even store the files on a central host.

Central Collection Point

To store the access data with the use of Git on a central host on the local network, the VCS must be installed there. If you use a preconfigured network-attached storage (NAS), look for the software in the NAS's package manager. One alternative is the use of an energy-saving Raspberry Pi for this task. With a minimal system, for example based on Raspbian Lite, you have an ideal computer as a remote station.

The host should preferably support login via Secure Shell. A separate article in this issue explains how to set up such a login and then configure SSH so that it suits your daily work schedule as effectively as possible.

In the following example, I assume you are running a Raspberry Pi as a NAS on the local network with hostname `storage`. Now create the repository in the home directory of user *pi*. To do this, switch to the host (e.g., by SSH) and create a new repository first. The first command from Listing 2 creates a simple directory, and the second then initializes the repository.

Use the `--bare` option to tell Git that this is not a working directory, but one where you can add commits or retrieve changes. This setup has the practical effect that the files that Git needs for administration are located directly in the folder, instead of in a hidden `.git` directory below it.

To use the new central repository in the password manager, first make sure that the data in `.password-store/` is under Git's control. You do this by typing `pass git init` in a terminal. The VCS outputs the typical status messages.

The structure of the commands is basically always the same: You use Git's regular syntax but always prefix it with the program name `pass`. This allows you to take full advantage of Git's features without much more configuration (see the "Tip" box).

Once you have initialized the local repository, add a remote repository with which you can exchange data. You do this with the commands from Listing 3.

When adding an external repository, first assign a name (`origin` in the example) and append the appropriate URL to it. You can freely assign the name; `origin` has just established itself as a convention. Finally, use the last command from Listing 3 to synchronize the remote target with the dataset from the local repository.

If you have also placed the local files on other computers under Git's control, you just need to configure the central computer as a host to integrate the respective files, as well.

If you want to retrieve the data from the central computer, simply type `pass git pull`. You will then find the same encrypted files on the host in question as on the central computer. In other words, if you want to work with files from different computers, this system makes synchronization easy. However, it remains your responsibility to keep the GnuPG keys consistent across all hosts.

Conclusions

The Pass password manager lets you store as many different passwords as you like for different accounts, and you'll still only need one password to decrypt them all. Although the technology behind Pass is relatively simple – after all, it's just a shell script – the combination of mature components adds to the overall effect.

Git lets you synchronize different hosts via a shared repository: It should go without saying that a public repository is not suitable for storing passwords. ■■■

Info

[1] Pass: *https://www.passwordstore.org*

[2] GnuPG: *https://www.gnupg.org/gph/en/manual/book1.html*

[3] "Unison: Data transfer" by Erik Bärwaldt, *Ubuntu User*, issue 8, 2011, p. 64

[4] Git: *http//www.git-scm.org*

COMPLETE YOUR LIBRARY

Order a digital archive bundle and save at least **50% off** the digisub rate!

ORDER YOURS TODAY!
shop.linuxnewmedia.com

You get an **entire year of your favorite magazines** in PDF format that you can access at any time from any device!

FOSSPicks

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

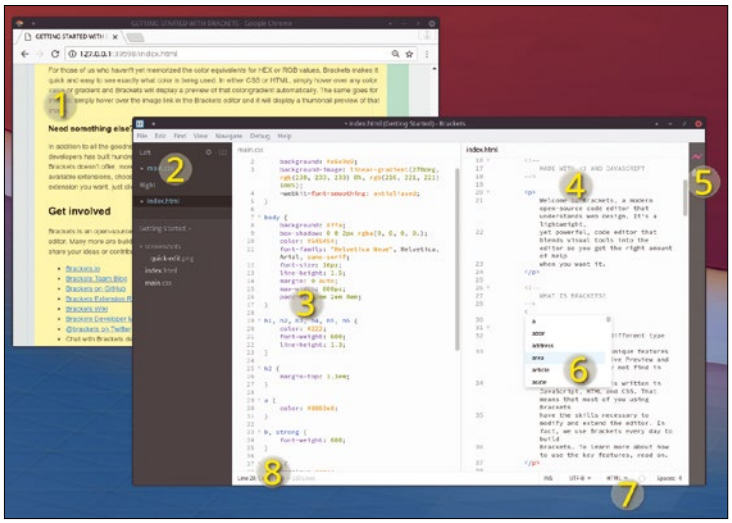
Graham tears himself away from updating Arch Linux to search for the best new free software. BY GRAHAM MORRISON

Web-focused text editor

Brackets

When potential Linux users are asked which proprietary software they'd most like to see on Linux, Adobe's suite of applications are always near the top of the list. This is because professionals rely on Photoshop, Illustrator, and InDesign to do their jobs, and it's difficult to find open source replacements while remaining ahead of the industry curve. With Linux now being used in so many professional production companies, many of us have been surprised that Adobe hasn't looked at creating Linux versions of its most popular applications. But there's still hope, and Brackets is a good reason to hope.

Brackets is an open source text editor targeted at web design. There's obviously a Linux version, but the part that makes Brackets unusual is that it's developed by Adobe, and it has been in development since 2014. This may lead you to think the project is some kind of failed trial by Adobe, but Brackets is far from being a failure. In fact, it's rather brilliant. The first thing you notice when you launch the application is that it looks nothing like a typical Adobe application. It actually looks good, and its user interface doesn't impinge on usability, with a large text pane holding the editor itself, complete with beautifully rendered text using whatever font you prefer. Start typing into an HTML document, and the autocomplete helps you start and finish elements, fixes indentation, and subtly highlights the tags from the text. This helpful functionality extends to colors, where you add the hex value for a specific hue and the editor will show you the color you've dialed in.



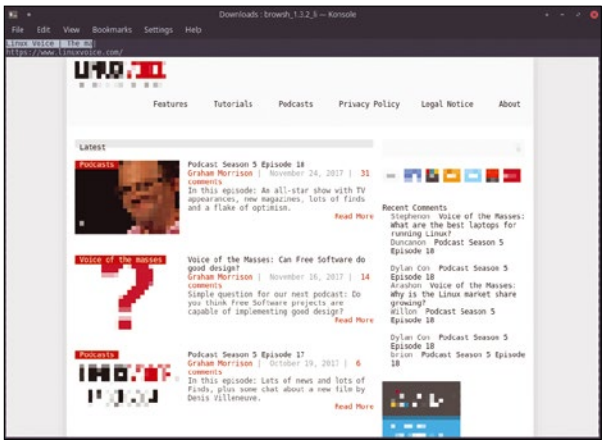
1 Live preview: Changes made to the raw text source of a site update the preview. **2 Multiple files:** Tab between open files or create both vertical and horizontal splits. **3 Code highlighting:** Text looks crisp and easy on the eye, and HTML colors are previewed in small swatches. **4 Distraction free:** A simple mode can be enabled to remove all the window furniture so you can focus on the code. **5 Extensions:** Augment your editing environment with dozens of easily installed plugins and extensions. **6 Code completion:** Use the drop-down menu to help with element memory. **7 Multiple languages:** Designed for the web, but you can use Brackets with many different languages. **8 Refactor and linting:** Change names easily and check and even fix common errors automatically.

But its smartest feature is also a new addition: the live preview. Selecting this will open a simple web browser window containing the rendered output of the HTML and CSS files you are editing. The clever part is that as you edit the source text files, the live preview updates instantly to reflect those changes. It feels like the developer modes you find in popular web browsers, where you can temporarily change how a page is rendered, but the difference here is that your changes are saved to the files used to build your eventual site. Thanks to its age and provenance, there are also dozens of add-ons that can be installed, allowing you to add themes, watch videos, make notes, and even turn the editor into a fully fledged IDE. While the emphasis is obviously on CSS and HTML, Brackets also supports a huge variety of formats and programming languages, from Bash to YAML, where you can take advantage of its excellent font rendering, refactoring, and split panes. It's a clean and effective editor. Although it's never going to replace something like Dreamweaver when it comes to designing a website without touching the source, it's perfectly suited to the modern role of web developer.

Project Website
http://brackets.io/

Terminal web browser

Browsh



Finally, a terminal-based web browser that is as easy to use and as compatible as a desktop browser.

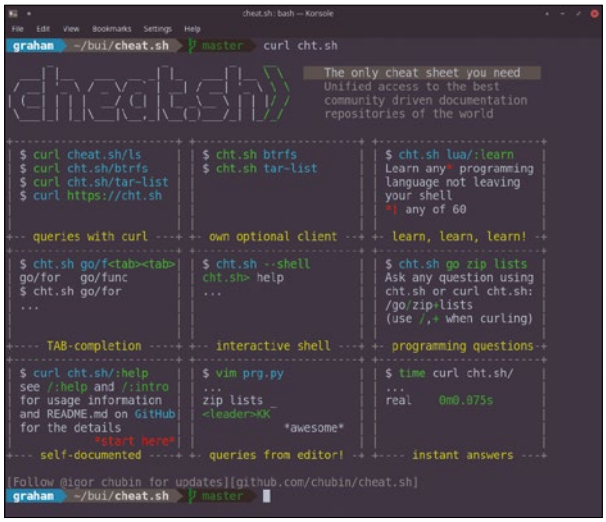
Despite the world wide web demanding more and more of our systems, many of us just want a simple browsing experience that neither detracts from the information we want nor diverts system resources. For this reason, a web browser running in a terminal is very attractive. Ideally, it would focus on the text and ignore both the images and wider site design, letting you read and download only the parts that matter. This would be brilliant if you're also on a low-bandwidth connection or connecting via SSH to a headless low-powered server such as a Raspberry Pi. But the absolute minimalism of console browsers like Links, Lynx, ELinks, and w3m is often too much for a modern site, both in the way their limited rendering makes a complex site difficult to navigate, and in their compatibility with modern web technologies like HTML5, CSS3, JavaScript, video, and even WebGL.

It's these problems that Browsh attempts to solve, albeit in an unconventional way. The unconventional way is that while it does run from the command line, it still requires you to have Firefox 57 or later installed. This is because Browsh uses Firefox to render the pages you request before rendering them as ASCII for use within your terminal. It may sound like a cheat, but it works perfectly and means that Browsh is compatible with every site that's compatible with Firefox – a huge advantage! The rendering is obviously blocky and pixelated, but it's also clear enough to be navigable. Vitally, the text is still raw text, which means that reading a page of content on the terminal is often clearer than reading the same content on a design-heavy site, and it's quicker and easier than using something like Reader.

Project Website
https://www.brow.sh

Online and offline help

cheat.sh



Use `cheat.sh` to get access to brilliant cheat sheets for your favorite commands and tools.

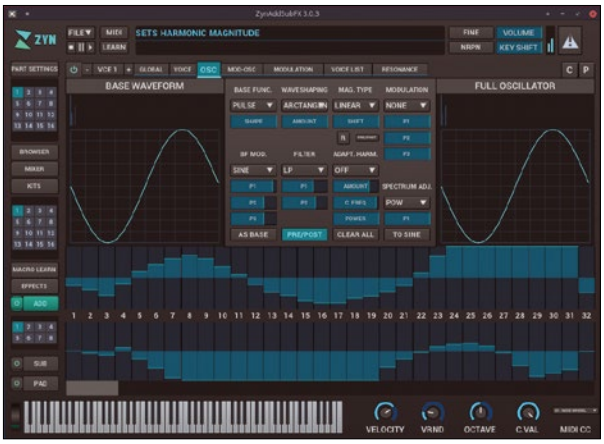
I love using the command line. It's where I spend most of my time, and I'll go out of my way to find a command-line solution, even when there's a potentially easier-to-use desktop application that does the same thing. But my memory is rubbish, and unless I use something every day, I soon forget commands and shortcuts for doing things in the utilities and languages I don't use that often. What I need is universal access to a cheat sheet system that enables me to quickly see the most commonly used command shortcuts and arguments for the tools I want to use. cheat.sh is that tool, developed to hit seven noble targets: (1) It's concise, only containing the details you want; (2) it's fast, delivering results on the command line almost instantly; (3) it's comprehensive, with access to plenty of tools and information; (4) it's universal, available everywhere; (5) it's unobtrusive when you're working; (6) it helps you learn; (7) it is inconspicuous.

cheat.sh is a GitHub repository that delivers on all of these promises by allowing you to grab prepared text documents that help you to work with the tools you commonly use. Type `curl https://cheat.sh/ssh`, for example, and you'll see quick examples of the SSH command complete with single-sentence descriptions. You can even list the cheat sheets that may be available for your favorite programming language – typing `curl cht.sh/cpp/:list`, for instance, lists 30 documents on C++, from arrays to logical and bitwise operators. If you don't like this remote `curl` approach, you can easily `curl` a local command-line client, `cht.sh`. You can now run the client locally to access the same information, and you can even integrate the search and results into editors like Vim and Emacs, which is perfect for people like me who can never even remember a simple `for` loop's syntax or even have the Vim commands from within Vim itself!

Project Website
https://github.com/chubin/cheat.sh

Subtractive synth

ZynAddSubFX 3.0



ZynAddSubFX has always been one of the best sounding Linux synths, and it's now one of the best looking.

The crazily named ZynAddSubFX has been around for a long time, but it has stood the test of time for one simple reason – it sounds sublime. This is often a hard trick to pull off with synthesizers, because even when you have all the correct elements in the mix – the analog sounding oscillators, the creamy filters, and the snappy envelopes – the sound can often be stale and cold regardless. ZynAddSubFX never had this problem. It has always sounded warm, rich, and fat, emulating the sound of old synthesizers, as well as generating more experimental output with its additive synth engine or soft chords with the pad synth. But it has also always had a big problem: the user interface. The ZynAddSubFX GUI reminded you of the early 2000s, because that's how old the software is – Motif-driven sliders and large gray buttons, totally unlike modern software synthesizers. And this is where Zyn-Fusion comes in.

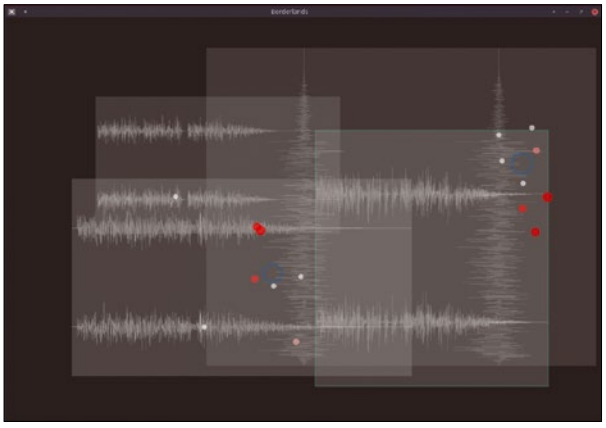
Zyn-Fusion is an all new user interface to ZynAddSubFX, designed and implemented by the project's current maintainer, Mark McCurry. It looks amazing and immediately elevates the synth into professional territory. The main difference is that almost everything can be controlled from a single window that still includes the keyboard but adds waveform output, part settings, and a tabbed list of controls for changing the oscillator, filter, and voice settings. It feels like the kind of synth for which you'd pay good money, and the project is in fact asking users to pay for this upgrade. However, the new interface is still open source; you can still build it yourself, although I'd recommend contributing to the project if you can afford to. Either way, the new design means ZynAddSubFX finally gets the look it deserves, while at the same time making its sounds far more intuitive to control and accessible.

Project Website

http://zynaddsubfx.sourceforge.net

Granular synthesizer

Borderlands



Manipulate audio waveforms and grain cloud generators to create evolving audio textures in Borderlands.

This is one of the most interesting sound generators I've ever come across. Borderlands is a granular synthesizer that generates sound from randomly selected "grains," or samples, in a waveform. It started life as a commercial iPad application, where its unique user interface was perfectly suited, and ended up as open source abandonware. Borderlands offers such an uncommon set of features and such a unique way of interacting with sound that it is worth the effort to get it installed and, hopefully, worth the effort of adopting the project if you have the skills. The project's dependencies may take some tracking down, but as long as you have *build-essential* installed, you should be able to build the source code with a simple `make` after that. With that done, source some interesting `.wav` files and place them in the correct folder. Running the application will also require a running and preconfigured JACK environment. With a bit of luck this can be as simple as installing JACK and running the daemon with the `pasuspender -- jackd` command.

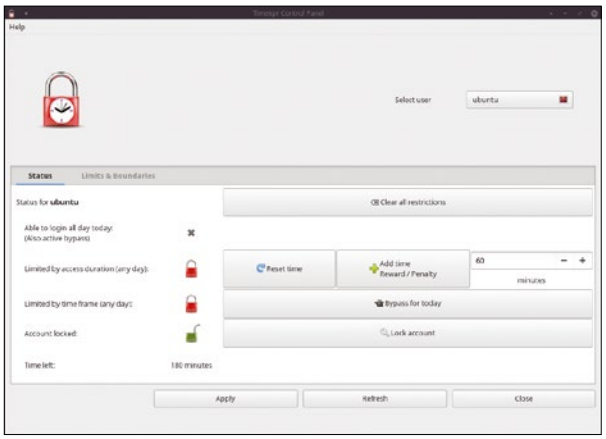
It's worth the effort. The waveforms of whatever audio files you drop into the correct folder will be displayed in a window. You can use the mouse to move, rotate, and scale around the background canvas. It works just like editing images in something like Gimp. It was this interaction that was so effective on the iPad. But the real magic happens when you press *G* to add the granular synthesis generators. These nodes can be moved and scaled across the waveforms, and the circular areas within them are used to generate particles that play the discrete audio element beneath their pixel positions. By moving these and the waveforms together, you constantly create new and evolving sounds that take their inspiration from your original audio, and it sounds amazing.

Project Website

https://ccrma.stanford.edu/~carlsonc/256a/Borderlands/

Internet limiter

Timekpr



Limit the amount of time user accounts on your system can access the Internet.

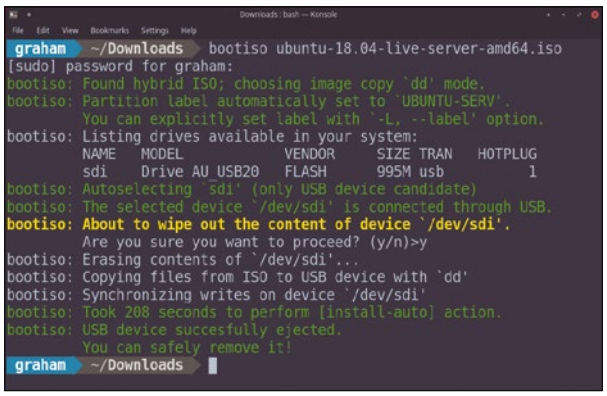
Even Apple has noticed that this thing it started, a compulsion to look at a screen every 10 seconds, may have its downsides, especially for children. Its latest devices have a rather neat feature that lets an adult see how long someone is using their device, as well as the kind of activities they're up to, making a distinction between "social networks" and "creativity," for instance. Many newer routers, too, will let you limit specific MAC addresses in the amount time spent online and which services are available and when. Linux is of course technically capable of implementing similar limits. Most of those routers are likely running Linux, but I'd ideally like to see an application that made setting such limits as easy to use as possible for the largest number of users, and that's what Timekpr does.

Timekpr has been around for a while, but recent releases are far more desktop agnostic than the Gnome-bound earlier versions. I had no issues running it on the latest KDE Plasma release. After installation, it will run automatically, and when you load up its control panel you'll be able to adjust time and access limits for each user on your system. Just select a user from the drop-down list and switch to the *Limits & Boundaries* tab. From there you can set both an access duration for each day of the week and a time frame when access is allowed, such as an hour in the evening for homework. From the status tab, you can see when the user is allowed access and easily grant them extra time or reduce their time limit to act as a reward or penalty. It all works perfectly, and while it can't yet enable or disable specific services, it's perfect for running on a family laptop or shared computer.

Project Website
https://launchpad.net/timekpr-revived

ISO writer

bootiso



Don't risk overwriting your root partition again when creating a USB stick with `dd`.

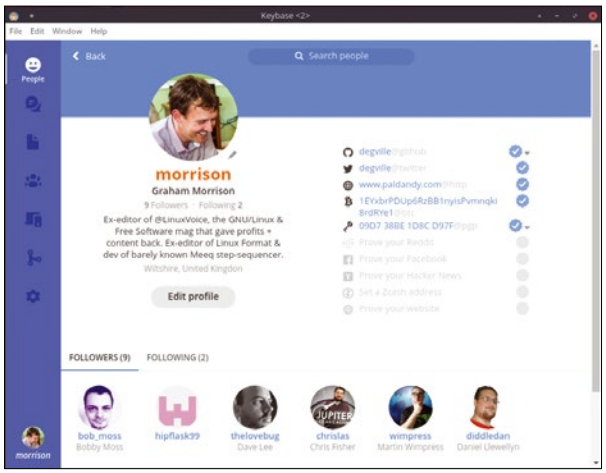
Many of us still resort to the humble `dd` command when writing a distro image to USB storage because it's simple and untainted by options or system interaction. But it does have some profound problems. Worst of all is that if you get a single character wrong when declaring the device node, such as `/etc/sda`, `dd` will start overwriting your internal storage without even asking politely whether you're certain, and you'll soon be diving for Ctrl+C. A great alternative is `bootiso`, which doesn't present the same risk, adds loads of new features, and still runs from the command line. At its simplest, you can run it against an ISO file with the `-p` argument, and it will tell you about whether your ISO is hybrid and capable of being written to USB storage and whether enough USB storage has been found.

In the background, it's also running lots of integrity checks on the ISO, making sure it will boot, and has the correct MIME type, as well as whether the potential destination is correct and not a single partition. This is useful in itself, and it will even let you know which device node your storage is hanging off in case you still want to risk `dd`. But `bootiso` is even better when you want to write the image, and that's because it still uses `dd`. Give it your ISO as the single argument, for example, and it will join up its detection routine with its encapsulation of `dd` to write the image automatically, carefully asking whether you're certain, after presenting the above details on what's going to happen. It will even erase the storage first. While there's no progress indicator – just like `dd` – by choosing the optimal block size in the background, the transfer is often quicker than trying your luck with `dd`.

Project Website
https://github.com/jsamr/bootiso

Encrypted comms

Keybase App



After a period of being "invitation only," anyone can now join Keybase and strengthen its web of trust.

After a period of being "invitation only," anyone can now join Keybase and strengthen its web of trust. Keybase.io is an online and proprietary service that differs from other services in important ways. Firstly, its primary objective is to help make secure encrypted communication between people easier to achieve; secondly, all the tools it uses to accomplish this goal are open source. It accomplishes the first part by being a proprietary service as it attempts to build a web of trust between its members. To do this, when you sign up with Keybase.io, you verify your identification against your own GnuPG key signature and a variety of online services that in theory only you can prove. These include Facebook, Twitter, GitHub, cryptocurrency addresses, and your own web domains.

With these proofs in place, other users can verify your identity with some certainty, which only becomes stronger as more choose to "follow" you, just as you might with friends in real life, or meet to share keys. This web of trust is going to be difficult to usurp, especially when taking the social element into account. It means that anyone can download your public key from Keybase and send you a message that only you can decrypt with more confidence than if they downloaded your public key from a random keyserver.

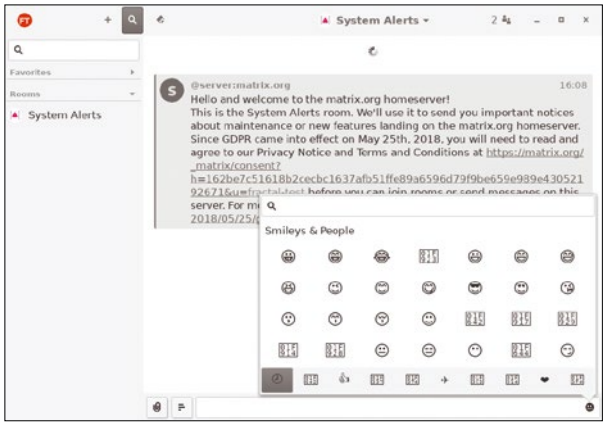
To help with all of this, and in an attempt to expand on its services to include trusted communication within teams, Keybase.io offers Linux users both a desktop application and a command-line interface. These tools allow you to update your keys from your local GnuPG installation, add new trusted devices (even from the command line, which features its own QR code generator), list followers, and send messages. The GUI chat is particularly powerful because it uses your contacts' respective GnuPG keys to ensure communication is end-to-end encrypted, a little like Telegram with more trust.

Project Website

https://keybase.io/

Matrix client

Fractal



Chat to anyone without a central server thanks to Matrix.org and this excellent Gnome client.

Matrix is a wonderful idea. It's a federated communication platform that provides a way to chat in groups or with one another without having to use one single server or portal. It works, and it's already very popular. Because it's open source and the APIs are well documented, you can find many different clients, from the web portal to Android. Fractal is a Gnome client for Matrix, hoping to bring the combined communication convenience to the world's most popular desktop. It can be installed from a Flatpak or with a manual build. It's built on Rust, which means building it yourself may be slightly different from what you're accustomed. It also helps if you have an account at *matrix.org* before running the application. With credentials added, the main window is much as you'd expect. Rooms and contacts are listed on the left, with the main chat window on the right. The chat window currently supports nearly all the media supported by Matrix (text, images, video, and audio) with the exception of notices and location. Decoration is minimal, in line with Gnome's aesthetic, but it feels much sharper and more modern than the *riot.im* web portal to the same chat services and its associated Android app.

You can add groups and contacts from the titlebar, and favorite rooms and people can be listed separately for easy access. The conversation is listed exactly as you'd expect, and there are buttons for easily uploading files or for adding emoji. It does everything you want with the exception of end-to-end encryption. This is planned for a future release but has no timeline, so it may be something to watch out for if security is important outside of the public channels you join; however, Fractal is still an excellent option for group chat.

Project Website

https://gitlab.gnome.org/World/fractal

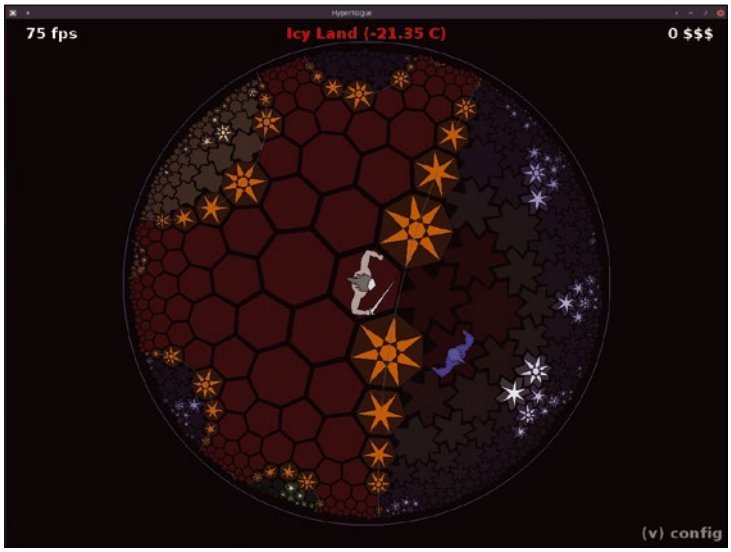
Roguelike

HyperRogue

There can't be many Linux users who don't know the classic Rogue or NetHack games. Both broke the ground for the concept of exploration as the central gaming component, where the dungeon you explore is entirely procedurally generated and different every time you play, just like the encounters you have and the locations of the objects littered throughout the levels to help you on your quest. But these games are both from a different era. If you've been around Linux long enough, you'll know they were two of the only games you could play on a 1990s early pre-X Window GNU/Linux installation. Their procedurally generated nature hid the requirement for memory and storage that handmade levels would require, as well as the human effort required in designing those levels. But somehow, the principles behind these games have survived the decades since their creation, passing through 3D acceleration, game consoles, and multicore CPUs. This means that if you're a newcomer to Linux, you're still likely to have heard of "Roguelike" even if you can't pin your first experience on a dirty amber screen in a 1990s computer lab.

The reason why this game style has far outlived the limitations that necessitated its design is the generations of balancing that now govern the algorithms that generate the levels and the difficulty. The levels may be procedural, but the algorithms that generate them now benefit from the influence of tens of thousands of player hours and many hundreds of developers. New games latch on to this addictive mechanic, adding their own elements to the tried and tested formula as they attempt to bring these games into the 21st century. HyperRogue is one of these games. There are several significant ways it differentiates itself whilst still remaining true to the original idea – one of which is that, although it's open source, it's also available as a commercial game on Steam, which is a great way of supporting the game if you enjoy it.

The first thing you're going to notice is its visual style. It describes itself as "non-Euclidean" Roguelike, and this means it doesn't feature normal geometry. Instead, exploration takes place on a hyperbolic plane – a grid of hexagons and heptagons that move in and out of view as they scale in from the display's circumference. On first glance, it looks like you're moving across the surface of a sphere, but the non-Euclidean scaling means it's more like looking through a concave lens, where you can see more detail along the edge. There are no parallel lines in this universe. While this does make navigation more challenging, its brain-defying difference makes you feel lost on another planet, which is exactly how a game like this should make you feel. The terrain will also change as you explore further from the starting point, with more than 60 different lands to explore. The general quest is to find 10 treasures from each land. Uniquely, you only have a single hit point, but so do your opponents, and combat is usually a single click as you fight off converging baddies. This makes combat only a peripheral challenge as you try to find enough treasure to open the next land, but it also makes the game fast and addictive to play.



The non-Euclidean geometry in HyperRogue means that there are no straight lines.

Each land has different characteristics, such as moving walls, as do all of the monsters, which you need to master to progress.

Project Website
http://roguetemple.com/z/hyper/

Metadata in ODF Files

It is no secret that the native file format of LibreOffice and OpenOffice, the Open-Document Format (ODF), is a truly open standard for word processing documents, spreadsheets, and presentations. What most people do not know is that ODF files contain lots of metadata that is very easy to read or modify. BY MARCO FIORETTI

Metadata means "data about data." The text messages you exchange using your phone, for example, are a form of data. The people with whom you exchange those messages, when, how often, from where, and so on are metadata about your messaging habits and connections.

Metadata is really important. I once heard French philosopher Bernard Stiegler observe that "the production of metadata has been the principal activity of those in power from the time of the proto-historical empires right up to today."

On a less philosophical and more practical level, lots of metadata is stored in your office documents, and you'll find many valid reasons for messing with the metadata in office files. This tutorial describes the most common of those reasons and offers a general approach to reading and writing metadata in ODF files – an approach that is quite easy and really extendable, because an ODF file is really just a standard ZIP archive of different kinds of plain text or image files.

Why Read and Write ODF Metadata?

Analyzing ODF metadata can help you work better and sometimes learn more about your organization than you thought possible. Editing the same metadata means controlling what everybody else knows about you. Together, these two procedures help to identify and fix many problems, from privacy and security to compliance and indexing. You may, among other things, automatically find, report, and "fix" (see below) ODF files that contain:

- Dangerous, obsolete, or redundant macros
- Information not compliant with your company policies
- Images containing location, author name, or other sensitive information

The raw metadata in ODF files can also be aggregated to create statistics, graphs, or report about whole collections of documents or to feed the same data into some external database. Numeric data that may be averaged goes from word counts to the number and overall duration of edits to each document. This, in turn, may facilitate both simple decisions ("which documents should

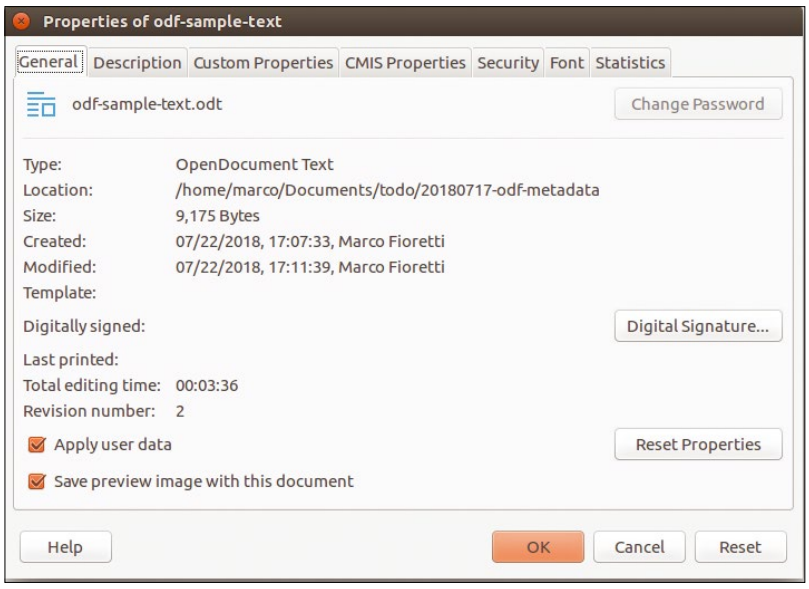


Figure 1: Almost all the metadata stored in the `meta.xml` component of any ODF file is accessible through the *File | Properties* tab in LibreOffice or OpenOffice. These are the general variables.

be updated first?") and more complex ones ("is our team working in the most efficient way?").

On the editing side, you may do the following, for example:

- Normalize and complete metadata (e.g., insert missing author names or titles, all with the same spelling, or change company or department names after a reorganization)
- Hide sensitive data (e.g., remove authors or comments inserted for internal use before sharing documents online, as an ODF, or even as a PDF)
- Add or update disclaimers for compliance with new regulations or company rules
- Add custom properties for better indexing
- Give files names that match the title of the document (or vice versa)
- Insert watermarks into pictures
- Remove metadata from inside pictures

Methodology and Scope

In this tutorial, I introduce a relatively simple way to read or write ODF metadata that works even on systems where LibreOffice or OpenOffice are not installed, including systems running Windows or Mac OS. All you need is support for shell scripts and a few other command-line utilities like `grep`, `sed`, `exiftool`, and ImageMagick: they are all included, or installable as binary packages, on almost every Linux distribution. Besides, this ODF metadata processing approach that you are going to learn can be useful in many other text-processing contexts.

When I say "introduce" or "approach," I mean that, while I provide working code, it is not a complete solution, but rather a collection of examples to use as inspiration and as building blocks for your own ODF metadata problems. One reason for this is that the mere printing of a script that could handle all possible cases with optimal performance would be longer than this whole article.

The other, more important reason is that almost nobody would need such a solution or "top" performance. ODF metadata hacks can save you many days of works, if not many weeks. They did for me. However, unless you really have to process thousands of files every day, you (like me) will only use these hacks in two ways:

- A few times a year, maybe in a different way every time
- Regularly, once per day or less, but as jobs that can run slowly in the background only on the files that have changed since the previous run

In scenarios like these, it is more efficient to put some code together quickly that just works, instead of optimizing it to death. What matters is knowing how to put that code together when the need suddenly arises.

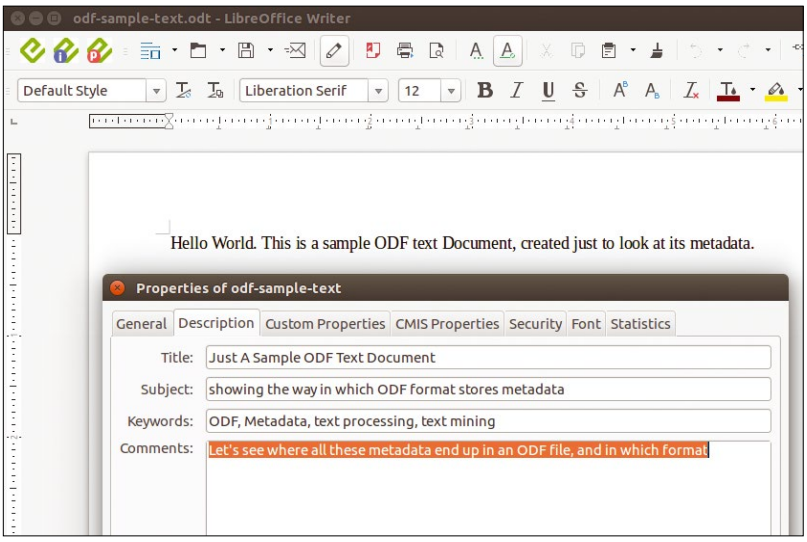


Figure 2: These are the descriptive metadata variables.

ODF Metadata

Mainly, there are two types of metadata in ODF files. The first consists of the data that you may read or set in the *LibreOffice File | Properties* tabs shown Figures 1 to 4. Some of those variables are present in every ODF file, others only in certain types, but they are all saved in a file called `metadata.xml` inside the ODF ZIP archive.

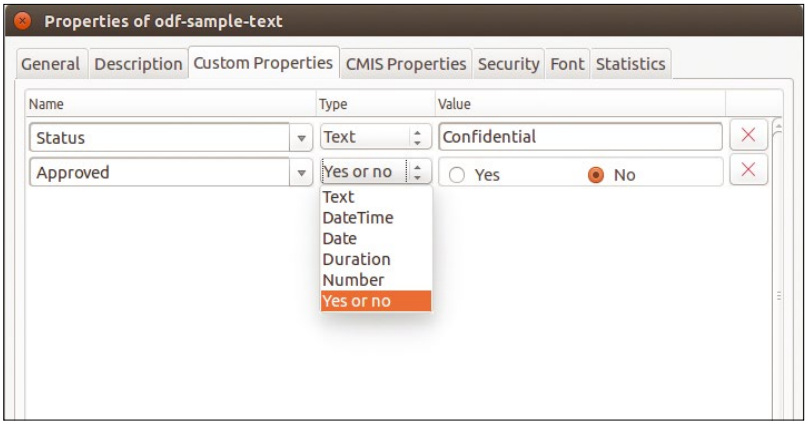


Figure 3: Users can also add custom metadata fields of several types, as they like.

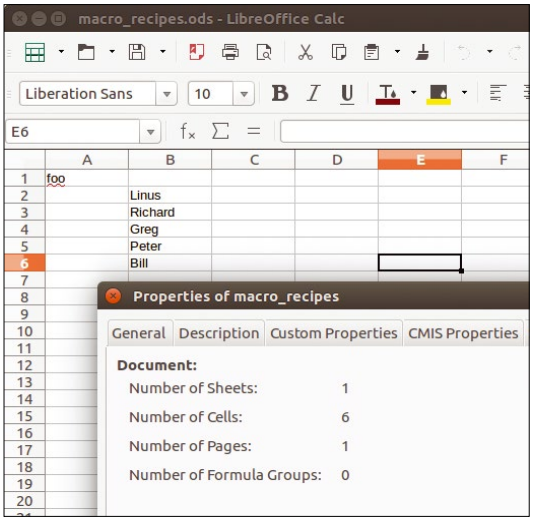


Figure 4: Some metadata, especially that in the ***Statistics*** category, is only defined for certain types of ODF files. *Number of Sheets* and *Number of Cells*, for example, only exist for spreadsheets.

In addition to this, so to speak, "official" metadata, there is what I would call "hidden" metadata – metadata in, or about, the "non textual" content of an ODF document, which is mainly macros and images. I will now show you how to read, and then write, both types of ODF metadata.

A Simple ODF Metadata Reader

Listing 1 shows a script, called `odfmetareader.sh` that follows the Unix philosophy of small tools that each do just one thing but can be connected in a pipeline. It just prints out, one per line, all the explicit and hidden metadata it finds in the single ODF file passed to it as an argument. Analysis of the output, or its insertion into some database or spreadsheet, is delegated to other tools. You can use this script inside a loop to work on as many files as you like, as shown later in the tutorial. Of course, you also can, and should, change the script to format its output to best suit your needs. Listing 1 shows how the code works.

The overall flow is very simple: The script makes a copy of the given file and unzips it in the temporary folder `/tmp/odfmetareader` (lines 3-8). The final command on line 55 removes that folder, but I recommend leaving it commented until you have figured out (by looking into that same folder) the internal structure of ODF files.

The central part of Listing 1 prints out the variables in the `meta.xml` files and two lists: one of macros and one of pictures, with all their own embedded metadata.

The `echo` commands containing the `## METADATA` string (e.g., lines 10 and 11) have the same purpose: They separate the several output sections (one hopes) making them more readable and easier to parse by other scripts.

Line 15 extracts all the metadata from the `meta.xml` file. It does seem like ancient Martian, but it is less obscure than it may seem at first sight. It is a concatenation of one long command in Perl and four invocations of the `grep` utility.

Listing 1: odfmetareader.sh

```
01 #! /bin/bash
02
03 rm -rf /tmp/odfmetareader
04 mkdir  /tmp/odfmetareader
05 cp $1  /tmp/odfmetareader/odf.zip
06 cd     /tmp/odfmetareader
07
08 unzip odf.zip >& /dev/null
09
10 echo "## METADATA DOC START     for document $1;"
11 echo "## METADATA ODF START     for document $1;"
12
13 # extract explicit ODF metadata
14
15 cat meta.xml | perl -e 'while (<>)
     {s/document-statistic//g; s/<(meta|dc):([^>]+)>/\n$2=/g;
     s/user-defined /user-defined-/g; s/<\/(meta|dc).*//g;
     s/ meta:value-type=/ value-type/g; s/ meta:/\n/g;
     s/\/=//g; s/<\/office:[^>]+>//g; print}
     print "\n"' | grep -v '<office:document' | grep -v
     '^<?xml version' | grep -v '^generator=' | grep '='
16
17 echo "## METADATA ODF END       for document $1;"
18 echo
19
20 # extract metadata about macros
21 if [ -d "Basic" ]
22 then
23   echo "## METADATA MACRO START   for document $1;"
24
25   MACRONUM=`find Basic -type f -name "*xml" | grep -v /
               script- | wc -l`
26
27   echo "macronumber=$MACRONUM"
28   for M in `find Basic -type f -name "*xml" | grep -v /
               script-`
29   do
30   echo macrofile:$M
31   grep 'sub ' $M
32   done
33   echo "## METADATA MACRO END for document $1;"
34   echo
35 fi
36
37 # extract metadata from images
38
39 if [ -d "Pictures" ]
40 then
41   for P in `find Pictures -type f`
42   do
43   N=`basename $P`
44   echo "## METADATA PICTURE START for document $1 /
           Picture $N;"
45   echo picturename: $N
46   exiftool $P | egrep '^(Artist|GPS)'
47   echo "## METADATA PICTURE END   for document $1 /
           Picture $N;"
48   done
49 fi
50 # final cleanup
51
52 echo
53 echo "## METADATA DOC END       for document $1;"
54 echo
55 #rm -rf /tmp/odfmetareader
56
57 exit
```

The Perl part is, basically, a series of regular expressions separated by semicolons that remove all the XML markup you don't need to see in the output. For example, this part

```
s/<\/(meta|dc).*//g;
```

replaces, with an empty string, every string that begins with `</meta` *or* `</dc`, *plus* all the characters that follow it until the end of the current line (that is what the `.*` part means). The four `grep` commands just remove header and footer lines in the XML file that don't contain any metadata. The best way to understand what line 15 actually does, and how to customize it for your needs, is to run the script on any ODF file and compare its output with the original content of the `meta.xml` file.

Native macros in ODF files are stored, if present, inside the `Basic` folder of the ZIP archive, and line 21 checks if this folder exists. If it does, the script finds all the macro files inside the folder and prints the value in the variable `MACRONUM` (lines 25-27). The loop in lines 28 to 25 finds and prints all the lines in the macro files that contain macro names.

The last loop of the script, in lines 39 to 49, checks if a `Pictures` folder exists. If the answer is yes, it scans all the pictures inside it (line 41), to print their names (lines 43-45) and then runs the `exiftool` command on them (line 46). `exiftool` is free software capable or reading and writing all the metadata stored inside today's digital photographs that use Exif and other similar standards.

When given a file name, as in line 46, `exiftool` just prints all the metadata in that file, one per line. The `egrep` command in line 46 discards all lines, except those that begin with either *Artist* or *GPS*, probably the most sensitive data.

Listing 2 shows a small excerpt, heavily edited for clarity, of the `odfmetareader.sh` output from the sample ODF document shown in Figure 5, which contains one macro and one photograph.

Publishing online ODF files (or office files in general, probably) without "cleaning" them first may mean letting everybody know where, and by whom, each photograph contained in the file was taken (as shown, starting in line 27). Sometimes this is OK; sometimes it is not.

The macro section (lines 21-25), as commented, lists number, location, and names of all the macros inside the document. The initial section (lines 1 to 19), is just a plain text version of the metadata shown in Figures 1 to 4. It is easy to imagine how many of the lines above, from editing cycles and duration to word count and keywords, may be filtered or fed to some other script to answer any kind of question.

As an example, the following lines show how you may discover which ODF files in a whole directory tree have *Linux Magazine* as the creator:

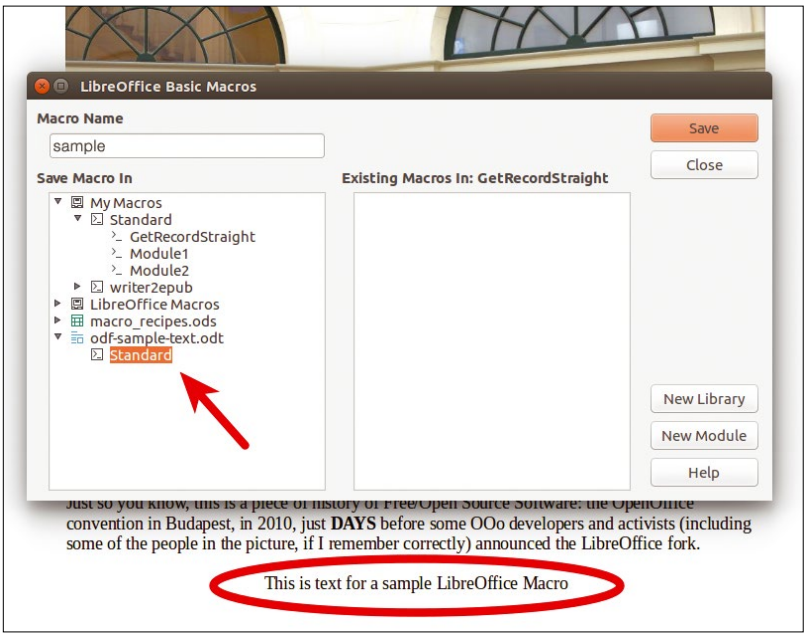


Figure 5: Basic macros in ODF documents can be organized in groups, which correspond to subfolders in the `Basic` folder of an ODF file. The macro in this figure will be saved in the file `Basic/Standard/sample.xml`.

Listing 2: odfmetareader Results

```
01 ## METADATA ODF START     for document odf-sample-text.odt;
02 initial-creator=Marco Fioretti
03 creation-date=2018-07-22T17
04 date=2018-07-22T18:07
05 creator=Marco Fioretti
06 editing-duration=PT33M32S
07 editing-cycles=9
08 description=Let's see where all these metadata end up...
09 keyword=ODF
10 keyword=Metadata
11 keyword=text processing
12 keyword=text mining
13 subject=showing the way in which ODF format stores metadata
14 title=Just A Sample ODF Text Document
15 image-count="1"
16 word-count="81"
17 character-count="468"
18 user-defined-meta:name="Approved" value-type"boolean"=false
19 user-defined-meta:name="Status"=Confidential
20
21 ## METADATA MACRO START   for document odf-sample-text.odt;
22 macronumber=1
23 macrofile:Basic/Standard/samplemodule.xml
24 sub Main
25 ## METADATA MACRO END     for document odf-sample-text.odt;
26
27 ## METADATA PICTURE START for document odf-sample-text.odt /
      Picture sample-picture.jpg;
28 picturename: sample-picture.jpg
29 Artist                          : Marco Fioretti
30 GPS Latitude                    : 47 deg 30' 20.53" N
31 GPS Longitude                   : 19 deg 2' 43.75" E
```

```
for F in `find . -type f | egrep '(odt|ods|odp)$`
  do
  FOUND=`odfmetareader $F | grep -i ^creator | ➥
    grep -i -c 'Linux Magazine'`
  if [ $FOUND gt 0 ]
    then # = "there was at least one line with ➥
      that string"
    echo found $F
  fi
done
```

Writing ODF Metadata

Extracting metadata from ODF files is great. Being able to erase or modify it is even better. You can learn how to do so by playing with the `odfmetawriter` script in Listing 3, which was written to order for didactical purposes. To begin, it only performs one operation per run for simplicity, always in the same way: Extract the file(s) that must be changed, process them, and then put them back in a copy of the zipped ODF file.

Listing 3: odfmetawriter.sh

```
01 #! /bin/bash
02
03 if [ ! -e "$1" ]
04 then
05   echo "script launched on non-existing file: $1;
         aborting"
06   exit
07 fi
08
09 STARTINGDIR=`pwd`
10
11 rm -rf /tmp/odfmetawriter
12 mkdir /tmp/odfmetawriter
13 cp $1 /tmp/odfmetawriter/odf.zip
14 cp $1 /tmp/odfmetawriter/new-$1
15 cd    /tmp/odfmetawriter
16
17 unzip odf.zip >& /dev/null
18 cp meta.xml meta.orig.xml
19
20 case "$2" in
21   creator|title|description)
22   echo "Changing $2 to: $3"
23   sed -i -- "s/<dc:$2>.*<\/dc:$2>/<dc:$2>$3<\/dc:$2>/"
       meta.xml
24   zip -f new-$1 meta.xml
25   ;;
26
27   addkeyword)
28   sed -i -- "s/<meta:keyword>/<meta:keyword>$3<\
       /meta:keyword><meta:keyword>/" meta.xml
29   zip -f new-$1 meta.xml
30   ;;
31
32   addcustom)
33   sed -i -- "s/<meta:user-defined/<meta:user-defined
       meta:name=\"$3\">$4<\/meta:user-defined>
       <meta:user-defined/" meta.xml
34   zip -f new-$1 meta.xml
35   ;;
36
37   renamefromtitle)
38   EXT="${1##*.}"
39   TITLE=`perl -e  'while (<>) {next unless
         m/.*<dc:title>(.*)<\/dc:title>/; $T = $1;} $T =~
         s/\W+/-/g; print $T' meta.xml`
40   mv -i new-$1 $STARTINGDIR/$TITLE.$EXT
41   exit
42   ;;
43
44   watermark)
45     if [ -d "Pictures" ]
46   then
47     for P in `find Pictures -type f`
48     do
49     convert $P  -font Arial -pointsize 60
         -draw "gravity center   fill yellow  text 1,11
         '$3' " temp-watermarked
50     mv temp-watermarked $P
51     zip -f new-$1 $P
52     done
53   else
54     echo "No Pictures in this ODF Document!"
55     exit
56   fi
57   ;;
58
59   removepicsdata)
60     if [ -d "Pictures" ]
61   then
62     for P in `find Pictures -type f`
63     do
64     exiftool -all= $P
65     zip -f new-$1 $P
66     done
67   else
68     echo "No Pictures in this ODF Document!"
69     exit
70   fi
71   ;;
72
73   *)
74   echo "unknown or unsupported option, please retry: $2;"
75   rm -rf /tmp/odfmetawriter
76   exit
77   ;;
78 esac
79
80 mv -i new-$1 $STARTINGDIR/
81
82 #rm -rf /tmp/odfmetawriter
83
84 exit
```

Then, to give you an idea of how you might alter both explicit and "hidden" ODF metadata, the script can do the following:

- Rewrite title, creator, or description
- Add an extra keyword
- Add a custom field
- Rename the file to match the document title
- Insert a textual watermark in all pictures
- Remove Exif data from pictures

The script must be launched always in the same way:

```
#> odfmetawriter <ODF-file-name> <operation> ➥
  <options>
```

The beginning and end are almost the same as `odfmetareader`: Create a temporary folder, work inside it, and remove it when done. Pay attention to line 14, though, which makes a copy of the file passed as an argument with the `new-` prefix: It is this file that will be "filled" with the new metadata and eventually (line 80) copied in the same directory where the script was launched.

The core of the script is the `case` statement (lines 20-78). It has seven branches: one for each of the operations listed above and a final one (lines 74-77) that exits with an error message in all other cases.

Lines 21 to 30 all do the same thing – that is, update or add a variable in the `meta.xml` file.

If the variable passed as a second argument (`$2`) is `creator`, `title`, or `description`, the first branch (lines 21-25) of the `case` statement finds the corresponding variable and, using the `sed` command, replaces its value with the string passed as the third argument.

The two other branches add keywords or custom fields (with a value equal to `$3`) when `$2` is equal to `addkeyword` or, respectively, `addcustom`. They work almost in the same way as the first one, with the only difference being that they prepend the XML markup defining the new variable to the other variables of the same kind.

In all cases, after the `meta.xml` file has been "updated," it is put back in the copy of the ODF file (lines 24 and 29).

The fourth supported operation does not change anything in the file. When the `$2` parameter is equal to `renamefromtitle`, the script:

- Takes note of the original file extension (`EXT`, line 38)
- Uses Perl to extract the title string from `meta.xml`, replace all of its non-alphanumeric characters with single dashes (line 39), and save the result in the `TITLE` variable
- Makes a copy of the original file, with the name `TITLE.EXT`, in the original directory

The last two operations supported by `odfmetawriter` are insertion of the textual watermark passed as the third parameter inside all the pictures (lines 44-57) and removal of all Exif metadata from the same pictures (lines 59-71).

The watermark is inserted with the ImageMagick's `convert` tool. The code in line 49 is copied almost verbatim from the relevant ImageMagick documentation [1]. Line 64, instead, tells `exiftool` to give all Exif variables in the current picture an empty value [2]. As before, the modified pictures (`$P`) are zipped back in the right place, in the copy of the original document (lines 51 and 65). Running the following commands, in sequence, on the sample ODF document shown in Figure 6

```
#> odfmetawriter odf-sample.odt title ➥
  'New title for Linux Magazine'
#> odfmetawriter odf-sample.odt description ➥
  'Here is an ODT file with its metadata ➥
      changed by a script'
#> odfmetawriter odf-sample.odt addkeyword ➥
  'ODF metadata processing'
#> odfmetawriter odf-sample.odt renamefromtitle
#> odfmetawriter New-title-for-➥
  Linux-Magazine.odt watermark ➥
  'Watermarked for Linux Magazine'
```

produces the results shown in Figure 7. (For simplicity, the renaming commands after each

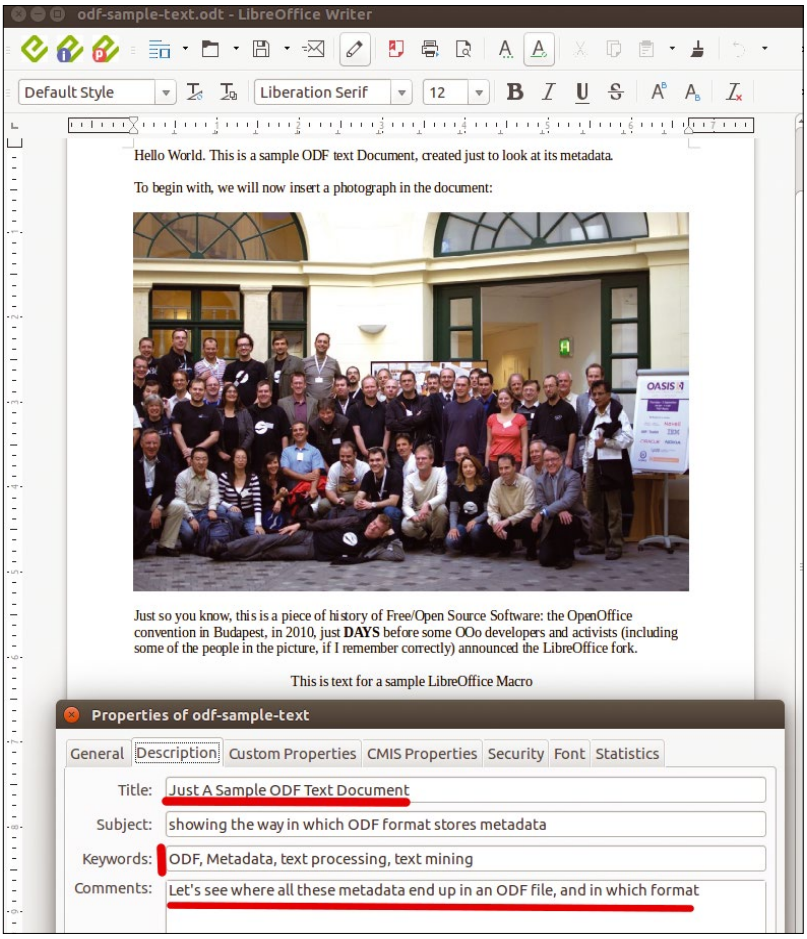


Figure 6: A sample ODF text file, with metadata and pictures inserted manually.

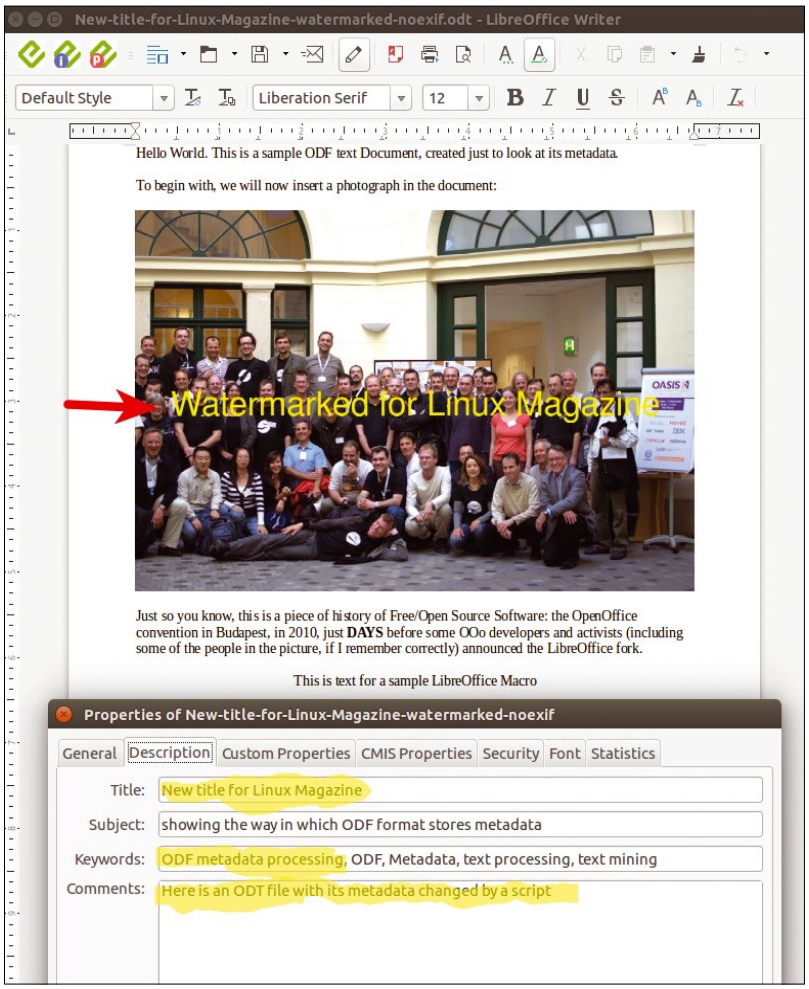


Figure 7: The same ODF text file, after the `odfmetawriter` script has automatically updated some metadata and watermarked the picture.

operation have been omitted.) As you can see for yourself, the metadata has the new values, and the picture is properly watermarked. Isn't ODF great to hack?

Code Limits

I already said this, but let me repeat it: The two scripts above do work, but they are not perfect or robust. As a minimum, they would need extra checks to refuse input files not in ODF format, or to handle properly non-alphabetic languages or strings with quotes inside them. In `odfmetawriter`, for example, `addcustom` will fail if there isn't already at least one custom field present. Also, `odfmetawriter` does not change the `initial-creator` of an ODF file. Another issue is dates: It is trivial to alter dates in the `meta.xml` file, but unless you do it right, you will end up with inconsistent documents (e.g., having ODF files with `last-modified` timestamps that are earlier than some of the revisions they contain). Finally, neither script is optimized for performance.

Still, look at the result in Figure 7: A quick and dirty mix of a few standard Linux commands and utilities is all you need to analyze or produce automatically any number of perfectly valid documents with just the metadata you want (or don't want). Is this cool, or what?

Final Thoughts and Warnings and a Request

In general, metadata hacking has issues that have nothing to do with the code or with ODF, as such. As Spider-Man's Uncle Ben would put it (and Voltaire did), "With great power comes great responsibility." Years ago, in a discussion over this same topic, someone commented "maybe we shouldn't teach our documents lying." Use the techniques you learned here responsibly. Be aware that digital signatures are the only way to guarantee that no part of an ODF file has been modified.

Last, but not least, even other parts of an ODF file contain stuff that maybe should count as metadata, even some people (including me, to some extent) may disagree: I'm talking of multiple revisions, but also of hidden paragraphs (or cells in spreadsheets), and of the content, author, and timestamps of embedded comments. All of this stuff may still be analyzed or "updated" with the same general approach presented here, thanks to the ODF format's openness and simplicity, but that is a different problem left as an exercise for the reader, with the suggestion that you use my ODF scripting examples [3] as a basis.

What's left? The request, of course: Please share how you use or modify these scripts for your own ODF metadata processing! ■■■

Info

[1] Watermarking: *www.imagemagick.org/Usage/annotating/#wmark_text*

[2] Removing Exif metadata: *www.linux-magazine.com/Online/Blogs/Productivity-Sauce/Remove-EXIF-Metadata-from-Photos-with-exiftool*

[3] ODF scripting: *http://freesoftware.zona-m.net/tag/odf-scripting*

The Author

Marco Fioretti (*http://mfioretti.com*) is a freelance author, trainer, and researcher based in Rome, Italy. He has been working with free/open source software since 1995 and on open digital standards since 2005. Marco also is a Board Member of the Free Knowledge Institute (*http://freeknowledge.eu*).

LINUX UPDATE

Need more Linux?

Our free Linux Update newsletter delivers insightful articles and tech tips to your mailbox twice a month. You'll discover:

- **Original articles on real-world Linux**
- **Linux news**
- **Tips on Bash scripting and other advanced techniques**
- **Discounts and special offers available only to newsletter subscribers**

www.linux-magazine.com/newsletter

Docker 101

You might think Docker is a tool reserved for gnarly sys admins, useful only to service companies that run complicated SaaS applications, but that is not true: Docker is useful for everybody. BY PAUL BROWN

Docker [1] manages and runs containers, a thing that acts like an operating system. It is similar to a virtual machine, but a container uses a lot of the underlying operating system (called the "host") to work. Instead of building a whole operating system with emulated hardware, its own kernel, and so on and so forth, a container uses everything it can from the underlying machine, and, if it is well-designed, implements only the bare essentials to run the application or service you want it to run.

Whereas virtual machines are designed to run everything a regular machine can run, containers are usually designed to run very specific jobs. That is why Docker is so popular for online platforms: You can have a blogging system in one container, a user forum in another, a store in another, and the database engine they all use in the background in another. Every container is perfectly isolated from the others. Docker allows you to link them up and pass information between them. If one goes down, the rest continue working; when the time comes to migrate to a new host, you just have to copy over the containers.

But there's more: Docker is building a library of images [2] that lets you enjoy whole services just by downloading and running them. These libraries are provided by the Docker company or shared by users and go from the very, very general, like a WordPress container [3], to the very, very niche, like a container that provides the framework to run a Minetest [4] server [5].

This means exactly what you think it means: Download the image, run it (with certain parameters), and your service is ready, madam – no dependency hunting, very little configuring, and not much more beyond hooking up the service to a database (running in another container) and setting your password as the service administrator.

Getting Started

To enjoy the marvels of Docker, first install it on your box. Most, if not all, of the main distributions have relatively modern versions of the Docker packages in their repositories. In Debian, Ubuntu, and other Debian-based distributions, look for a package called *docker.io*. In Fedora, openSUSE, Arch, Manjaro, Antergos, and others, it is simply *docker*. You will also find official and updated versions of the software for several systems at the Docker website [6].

Once Docker is downloaded and installed, check that the daemon is running:

```
systemctl status docker
```

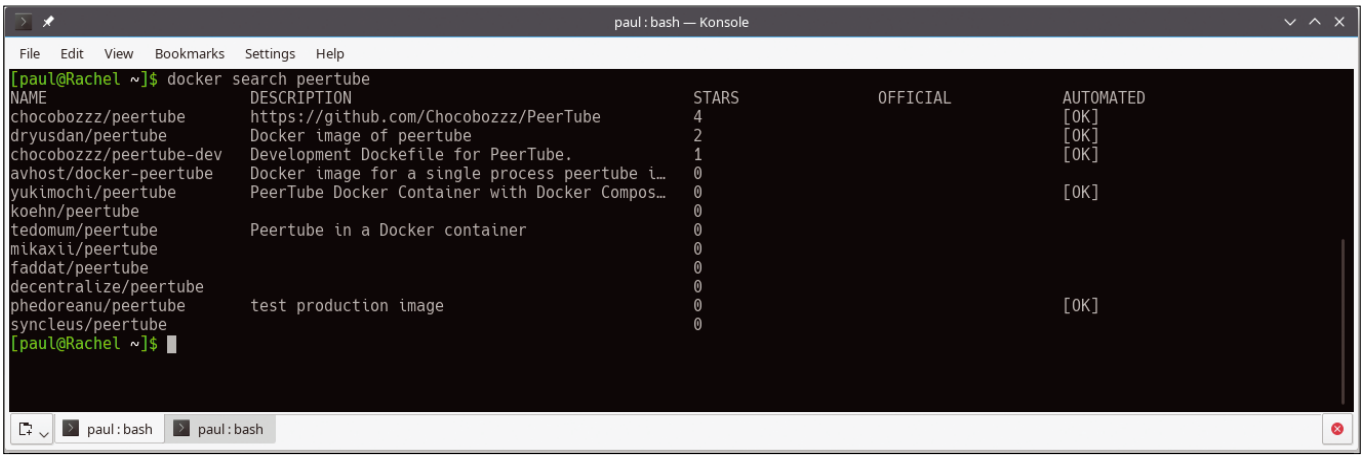


Figure 1: You can search Docker for images just as if you were using your software manager.

What is PeerTube?

PeerTube [7] is a video portal service akin to YouTube and Vimeo (Figure 2), but without any of the dumb restrictions of those closed and proprietary alternatives. It is called PeerTube because anyone can set up a server and join a federated network of PeerTube instances; any video a user uploads to one instance gets propagated to the other instances. All instances share the load of streaming the videos to visitors using P2P technology.

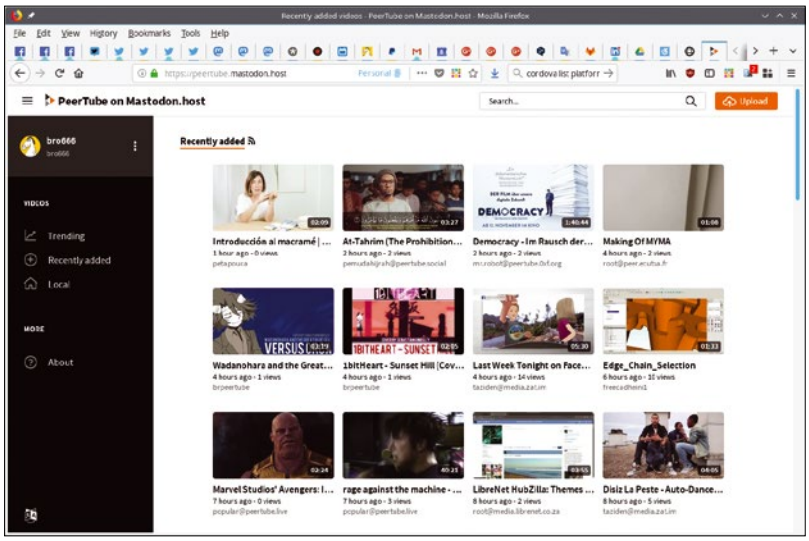


Figure 2: PeerTube is a much more democratic and freedom-respecting video platform than YouTube and Vimeo.

If it is not, start it and enable it so that it runs every time you boot your machine:

```
sudo systemctl start docker
sudo systemctl enable docker
```

Now Docker is running, it is time to get some images.

Imagine

An image is similar to an ISO image you would use to install a GNU/Linux operating system, except you don't need to burn it to a DVD or USB thumb drive.

You can use the `docker` utility to search for images like this:

```
docker search peertube
```

Docker will show you all the available images that contain the word "peertube" in the name or description (Figure 1). It will also tell you its rating given by users – more stars is better.

To install an image, you can `pull` it from a repository:

```
docker pull chocobozzz/peertube
```

This will download a PeerTube image (see the "What is PeerTube?" box) from Docker's repository and add it to your roster.

You can check that the image is now installed by running:

```
docker image list
```

Among other things, the list will give you a unique identifier (just in case you have two images with the same name) and will tell you how much space the image takes up on disk.

You could also just `run` the image, even before downloading it. The command

```
docker run chocobozzz/peertube
```

will have Docker look for the PeerTube image on your hard disk, and, if it can't find it, it will download it, drag in all the dependencies it needs (including other images, like an image for a PostgreSQL server), and run it (Figure 3, top).

When you run an image, Docker creates a container with the software running inside it. In many cases, it will show the software's output so you can check that everything is working correctly (Figure 3, bottom). In this case, the output tells you that your PeerTube instance is running on *localhost*. However, if you visit *http://localhost:80* with your browser, you probably won't see the PeerTube interface, because Docker sets up its own network for its containers.

To know which IP PeerTube is running on, first list your running docker containers like this:

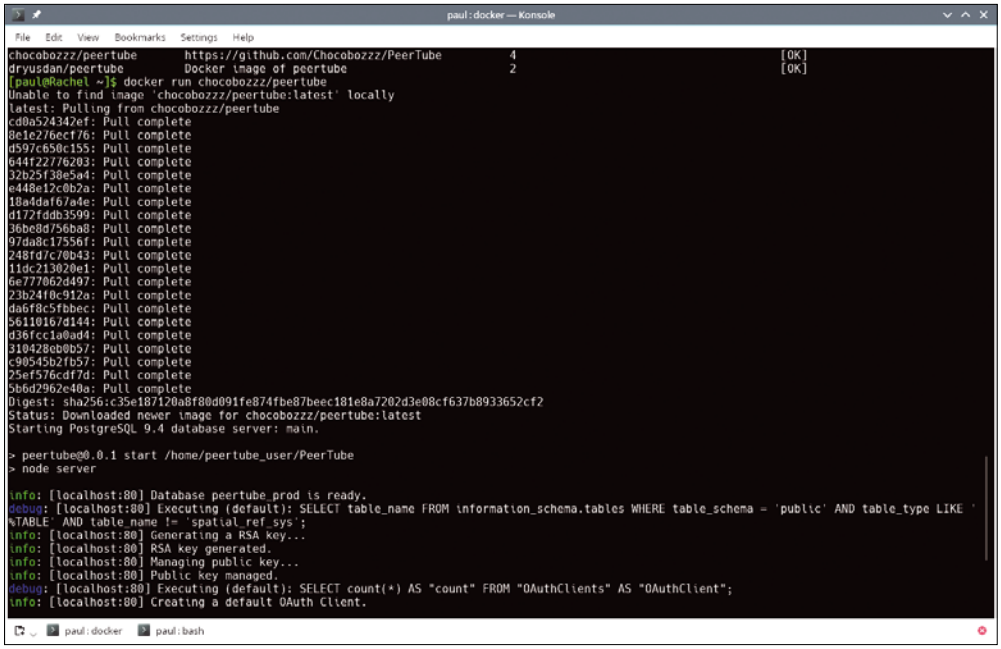


Figure 3: Running an image not already on your hard disk makes Docker download it and then run it.

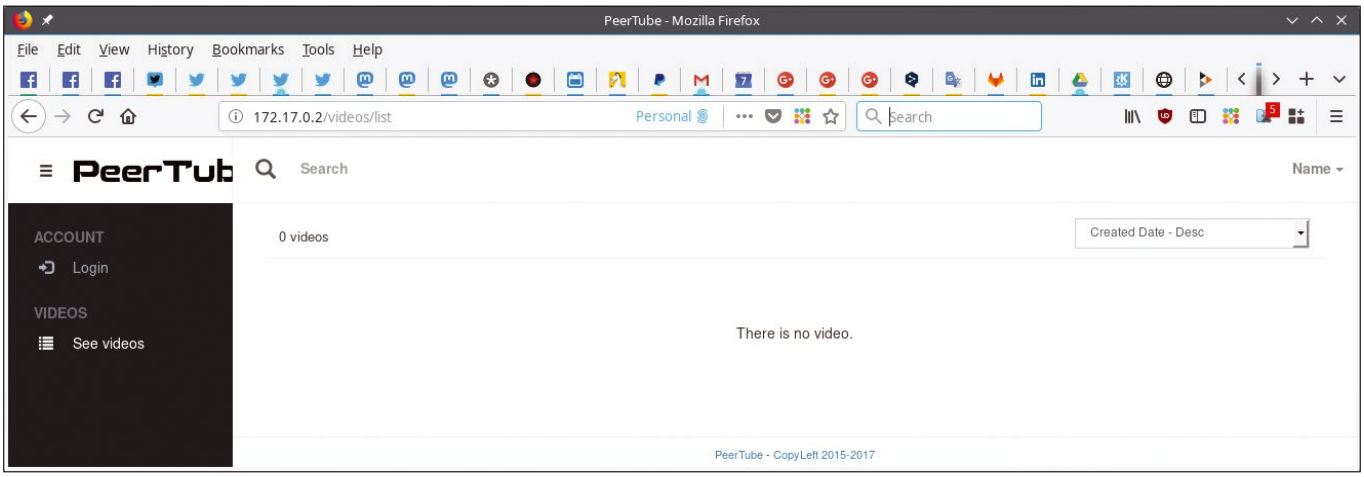


Figure 4: Setting up a PeerTube video platform requires virtually no work if you use a Docker image.

```
docker container list
```

This will give you a *container ID* (something like *8577b5867b93*) and a *name* that Docker makes up by mashing together random words (something like *hopeful_volhard*) for your container. You can use either to identify your container and get some details using:

```
docker container inspect <container_id_or_name>
```

Toward the end of the output, you will see a line that says *"IPAddress":* and, well, an IP address. If you haven't changed Docker's default configuration, it will be something like *172.17.0.2*. Point your browser at that, and … Voilà! PeerTube (Figure 4).

You can stop a container with `stop`:

```
docker stop <container_id_or_name>
```

And start it again with `start`:

```
docker start <container_id_or_name>
```

Using `run` will create a completely new Docker container from your original image. If you have made changes (like created or modified a file) within a Docker container of the same image, your changes will *not* be in the new container. The "Getting Rid of Stuff" box explains how you can cleanly remove both containers and images.

Getting Rid of Stuff

List the images you have installed and use the ID of the one you want to remove to delete it:

```
docker image rm <id_number>
```

You may get an error informing you that the image is in use or needed by a certain container. Note that, even if all your containers are stopped, they are not necessarily removed and are sitting there waiting to be restarted. You can see all you containers, even those that are not running, with:

```
docker container list --all
```

and then you can remove the offending container with:

```
docker container rm <container_id_or_name>
```

After that, you can go back and remove the image.

Inside the Container

Finishing off the configuration of PeerTube would require an article of its own (watch this space!), so I'll move on to a more generic image for experimentation. Grab yourself a Linux distro image, like Ubuntu,

```
docker pull ubuntu
```

and run it with:

```
docker run -i -t ubuntu bash
```

After a few seconds, Docker will dump you into a shell within the container. Unpacking that last command line, the `-i` option tells Docker that you want an interactive exchange with the container, which means that the commands you type into the host's stdin (usually your shell) will be pushed to the Docker image. The `-t` option tells docker to emulate a terminal over which you can send the commands. You will often see both options combined together as `-it`.

Next comes the ID or name of the image you want to interact with (`ubuntu` in this case). Finally, you pass the name of the command you want to run, in this case a Bash shell.

Find out what the name or ID of the container is (`docker container list`), and you can open a new shell in the running container using the `exec` command:

```
docker exec -it <container_id_or_name> bash
```

The instruction above logs you into the container, and you can install and remove software, edit files, start and stop services, and so on.

To stop the shell in the container, issue an `exit` as you would do to exit a regular shell. Once you

log out from all the shells, and as long as no other processes are executing, your Ubuntu container will stop. Docker containers are designed to run one process and one process only. Although you can run more, this is frowned upon by Docker purists and considered suboptimal. When that unique process ends, Docker is designed to close down the container.

However, if you want to keep a container running in the background (so you can have it run a non-interactive command sent to it from time to time), you can do this:

```
docker run -t -d <image_id_or_name>
```

As you saw above, `-t` tells Docker to create a faux terminal. The `-d` option stands for *detached* and tells Docker to run the container in the background.

To run a command non-interactively in a running container and have the output appear under the command, enter

```
docker exec <container_id_or_name> ls
```

which will show the default working directory's contents. You can also show the contents of a directory that is not the default by adding the path, as you would with a regular `ls` command:

```
docker exec <container_id_or_name> ➥
  ls </path/to/container/directory>
```

Talking of working directories, if you are not sure which is the container's current working directory, try this:

```
docker exec <container_id_or_name> pwd
```

Another thing you can do is share directories between the host and a container. For example:

```
docker run -it -v /home/<your_username>:➥
  /home/brian ubuntu bash
```

The `-v` option takes the path to the directory on the host (in this case, your own home directory) and maps it to the directory within the container. If either of these directories do not exist, Docker will try and create them for you.

Once you have shared your directory, from within the container, `ls` the `/home/brian` directory, and you will see the files from your own home directory. If you execute `touch /home/brian/from_docker.txt` from inside your container, you will see the file `from_docker.txt` pop up in your home directory on the outside.

This is very useful for when you want to use a Docker container to do some dirty work for you, like when you want to make an app for Android.

Docker for Developers

More precisely I should call this section "Docker for Fly-By Developers," because everybody likes a spot of coding, right?

One would presume developing for mobile devices would have become democratized by now. It is true that there are plenty of frameworks that let you use your favorite programming language to create apps for the likes of Android; however, setting up the SDK, NDK, libraries, dependencies, toolchains, and so on is a bit complicated …. No! Scratch that. More like: "Setting up the Android SDK, NDK, and so on is an absolute hell-hole of broken dependencies that will drive the hardiest among us to total and irredeemable insanity." There, much better.

What is the lazy and cavalier programmer to do? Use Docker, of course.

Turns out there are Docker images that provide all the hooks and doodads for Cordova (see the "Cordova" box). They also provide a correct and working installation of the Android tools you need to build and deploy your apps.

To get started, check what the Docker repository has to offer,

```
docker search cordova
```

and grab the image with the highest score. At the moment of writing, that was `beevelop/cordova`:

```
docker pull beevelop/cordova
```

Cordova provides a way of generating a skeleton app that you can run to test things, and later you can expand it to include your own features.

To build it, move to the directory to the location in which you want to store your app and run:

```
docker run --rm -i -v /$PWD:/workspace ➥
  -w /workspace --privileged beevelop/cordova ➥
  cordova create hello come.example.hello ➥
  HelloWorld
```

Don't panic: It isn't as hard as it looks. The first option, `--rm`, makes sure that Docker deletes the container after it has run the command you pass to it. Because you don't want the container hanging around after every step of the process, this is a good idea.

You already know what `-i` does: It gives you interactivity with the container. If there are any ques-

Cordova

Cordova [8] is a framework that allows you to create Android, iOS, and Windows apps using HTML, JavaScript, and CSS. Its aim is to democratize the creation of apps for mobile devices, especially Android. However, the complexity of installing the Android bits and pieces and making it work with a real phone dampens this lofty goal.

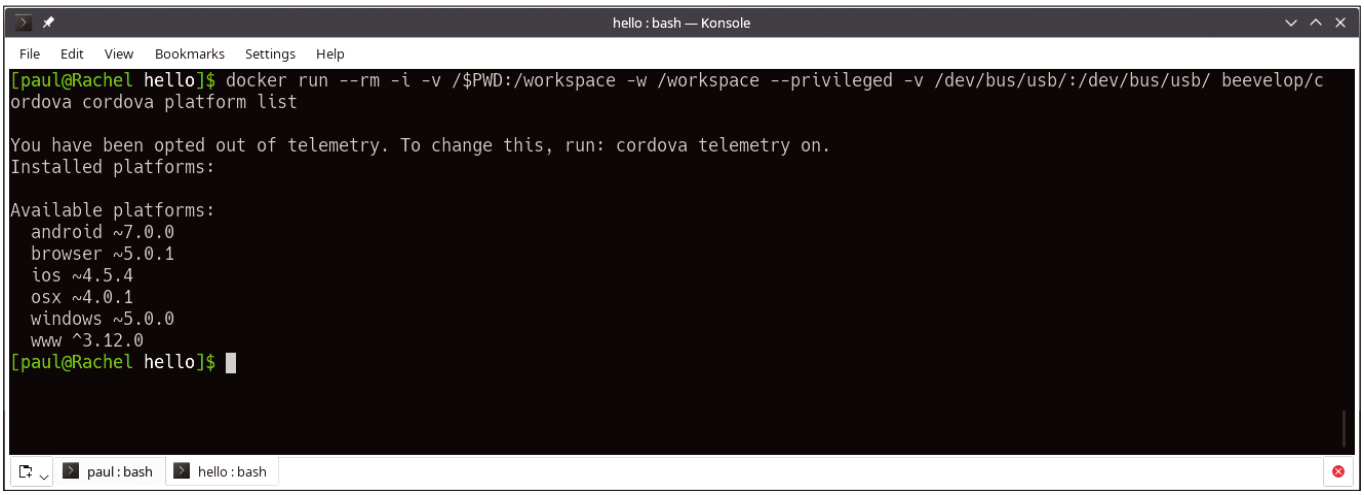


Figure 5: Cordova tells you for which platforms you can build your app.

tions to answer or fields to fill in, `-i` makes sure you will be able to do that.

The `-v` option is also familiar: It mounts the directory from the host entered before the colon into the container at the directory indicated after the colon. In this case, it will mount your current directory (`$PWD`) into a directory called `workspace/` within the container.

The option `-w` is new, but not difficult to grasp: It tells Docker which directory it should use as the working directory. In this case, all the files Cordova generates will go into the `workspace/` directory you created with the `-v` option.

The `--privileged` option gives superuser-like privileges to the container and allows Cordova to read from and write to the directories you mounted.

The `cordova create hello come.example.hello HelloWorld` chain is the Cordova command line to run. The `cordova create` part creates a new project, and `hello` is the directory where all the projects files will live. The `come.example.hello` part provides the basic building template for the `HelloWorld` project (you can call it something else, by the way) and is part of the standard Cordova package.

Prepare Your Phone

To make your phone ready for development, go to *Settings* | *About phone* and scroll down until you see the *Build number* section. Tap on that several times until your phone tells you that you have become a developer.

Connect your phone to your computer using a USB cable and move back to *Settings*. Now, you will see a new submenu called *Developer options*. Tap on that and scroll down until you see the *USB debugging* option. Activate it.

Your phone is ready.

After running the instruction, a `hello/` subdirectory will pop up in your current directory that contains a basic framework for an app. I will not go into each of the bits and pieces, because I will be looking at Cordova-based mobile applications in a future article, but you should now change into your `hello/` directory, because the rest of the instructions in this tutorial require that you execute them from within a Cordova-generated folder.

So far, Cordova has generated a platform-neutral application. Because Cordova allows you to build for more than one platform, the next step is to download all the stuff you need for Cordova to adapt the app to a specific platform.

To see what platforms are available, you can enter:

```
docker run --rm -i -v /$PWD:/workspace ⮒
  -w /workspace --privileged beevelop/cordova ⮒
  cordova platform list
```

The only new thing here is `cordova platform list`, which is self-explanatory (Figure 5).

To add a platform (e.g., Android), you use `cordova platform add`:

```
docker run --rm -i -v /$PWD:/workspace ⮒
  -w /workspace --privileged beevelop/cordova ⮒
  cordova platform add android
```

You can add as many platforms as you like.

The next step is building the code for the platform(s) you just added:

```
docker run --rm -i -v /$PWD:/workspace ⮒
  -w /workspace --privileged beevelop/cordova ⮒
  cordova build
```

After some rather profuse output, Cordova informs you it has built an Android Package Kit (APK) and placed it into the `platforms/android/app/build/outputs/apk/debug/` subdirectory.

You could now copy the APK to your device and install it by hand, or you can have Cordova do that for you and run the app as a test.

The first matter of business is to make sure Cordova can talk to your device. First, make sure your phone is ready and in development mode. Follow the instructions in the "Prepare Your Phone" box. Second, run the instruction:

```
docker run --rm -i --privileged ➥
  -v /dev/bus/usb:/dev/bus/usb ➥
  beevelop/cordova adb devices
```

In this command line, you are sharing your `/dev/bus/usb/` directory with your container, since that is where Cordova will find your phone. Cordova itself is using the Android Debug Bridge (`adb`) to try and locate your phone. The `devices` option shows a list of connected devices.

The first time around, your device may show up as unauthorized. This is normal. Go into *Settings* | *Developer options* and make sure you have enabled *USB debugging*. While you are there, again run

```
docker run --rm -i --privileged ➥
  -v /dev/bus/usb:/dev/bus/usb ➥
  beevelop/cordova adb devices
```

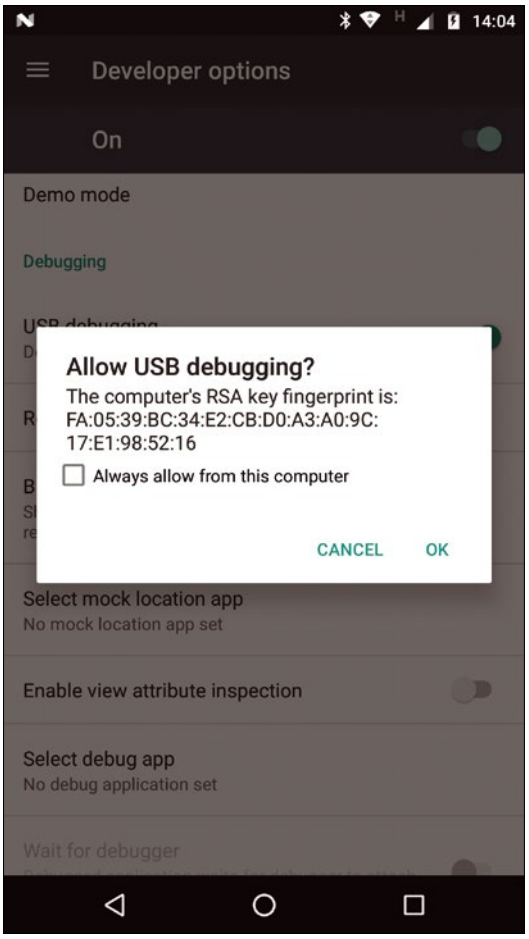


Figure 6: You need to authorize your computer before you can transfer your app to your phone.

and a dialog will pop up on your phone asking you to authorize your computer (Figure 6). Give your computer permission, and try listing your devices again. Your phone should now appear as available. Now you can push your app to your phone:

```
docker run --rm -i -v /$PWD:/workspace ➥
  -w /workspace --privileged ➥
  -v /dev/bus/usb/:/dev/bus/usb/ ➥
  beevelop/cordova cordova run android
```

Cordova will automatically install and run the app, so you can check that everything is okay (Figure 7).

Conclusion

Docker is being marketed as a solution for professional sys admins who manage dozens of services on busy server farms. True, Docker and all the other technologies built up around it are super-useful for those guys.

However, it is also useful for the rest of us: the casual home admin or amateur developer who wants to tinker or build stuff for personal use. That is why I will be incorporating Docker into my toolbox in future installments of this series.

In the meantime, have fun playing with Docker! ■■■

Figure 7: Cordova's Hello World app running on a phone.

Info

[1] Docker: *https://www.docker.com/*

[2] Docker's list of prepackaged containers: *https://hub.docker.com/*

[3] Official WordPress container: *https://hub.docker.com/_/wordpress/*

[4] Minetest: *https://www.minetest.net/*

[5] Docker container for Minetest: *https://hub.docker.com/r/linuxserver/minetest/*

[6] Updated Docker versions: *https://store.docker.com/search?type=edition&offering=community*

[7] PeerTube: *https://joinpeertube.org/en/*

[8] Cordova: *https://cordova.apache.org/*

FEATURED EVENTS

Users, developers, and vendors meet at Linux events around the world. We at *Linux Magazine* are proud to sponsor the Featured Events shown here.

For other events near you, check our extensive events calendar online at *http://linux-magazine.com/events.*

If you know of another Linux event you would like us to add to our calendar, please send a message with all the details to *events@linux-magazine.com*.

Drupal Europe

Date: September 10-14, 2018

Location: Darmstadt, Germany

Website: *https://www.drupaleurope.org/*

Drupal Europe is both a technology conference and a family reunion for the Drupal community. Eleven industry tracks focus on real life case studies and success stories alongside emerging new best practices. Drupal Europe is put on by a group of community volunteers in collaboration with the German Drupal Association and the Drupal Europe Foundation.

All Things Open 2018

Date: October 21-23

Location: Raleigh, North Carolina

Website: *https://allthingsopen.org/*

All Things Open is the largest "Open" technology event on the east coast of the United States. Join 3,500+ technologists and decision makers for three days of technical sessions from some of the most well-known experts in the world.

LISA18

Date: October 29-31

Location: Nashville, Tennessee

Website: *https://www.usenix.org/conference/lisa18*

LISA is the premier conference for operations professionals. This three-day event brings sys admins, systems engineers, IT operations professionals, SRE practitioners, developers, IT managers, and academic researchers together to share real-world knowledge about designing, building, securing, and maintaining the critical systems of our interconnected world.

Events

Atlassian Summit Europe	September 3–5, 2018	Barcelona, Spain	https://www.atlassian.com/company/events/summit-europe
Drupal Europe	September 10-14, 2018	Darmstadt, Germany	https://www.drupaleurope.org/
DevOpsDays Berlin	September 12–13, 2018	Berlin, Germany	https://www.devopsdays.org/events/2018-berlin/welcome/
The Linux Foundation Legal Summit	September 12–14, 2018	San Francisco, California	https://events.linuxfoundation.org/events/lf-member-legal-summit-2018/
Open Source Firmware Conf.	September 12–15, 2018	Erlangen, Germany	https://osfc.io/
Storage Developer Conf.	September 24–27, 2018	Santa Clara, California	https://www.snia.org/events/storage-developer
Open Networking Summit Europe	September 25–27, 2018	Amsterdam, Netherlands	https://events.linuxfoundation.org/events/open-networking-summit-europe-2018/
Open Source Backup Conference 2018	September 26–27, 2018	Cologne, Germany	https://upcoming.org/event/open-source-backup-conference-2018-0xlw0zogn5
All Systems Go	September 28-30, 2018	Berlin, Germany	https://all-systems-go.io/
OSDI '18	October 8-10, 2018	Carlsbad, California	https://www.usenix.org/conference/osdi18
LinuxDay Vorarlberg	October 13, 2018	Dornbirn, Austria	https://www.linuxday.at/
Open Source Automation Day	October 16, 2018	Munich, Germany	https://osad-munich.org/
All Things Open	October 21-23, 2018	Raleigh, North Carolina	https://allthingsopen.org/
Open Source Summit Europe	October 22-24, 2018	Edinburgh, UK	https://events.linuxfoundation.org/events/open-source-summit-europe-2018/

CALL FOR PAPERS

We are always looking for good articles on Linux and the tools of the Linux environment. Although we will consider any topic, the following themes are of special interest:

- System administration
- Useful tips and tools
- Security, both news and techniques
- Product reviews, especially from real-world experience
- Community news and projects

If you have an idea, send a proposal with an outline, an estimate of the length, a description of your background, and contact information to *edit@linux-magazine.com*.

The technical level of the article should be consistent with what you normally read in *Linux Magazine*. Remember that *Linux Magazine* is read in many countries, and your article may be translated into one of our sister publications. Therefore, it is best to avoid using slang and idioms that might not be understood by all readers.

Be careful when referring to dates or events in the future. Many weeks could pass between your manuscript submission and the final copy reaching the reader's hands. When submitting proposals or manuscripts, please use a subject line in your email message that helps us identify your message as an article proposal. Screenshots and other supporting materials are always welcome.

Additional information is available at:
http://www.linux-magazine.com/contact/write_for_us.

NOW PRINTED ON recycled paper from 100% post-consumer waste; no chlorine bleach is used in the production process.

Authors

Erik Bärwaldt	16, 32
Swapnil Bhartiya	8
Andreas Bohle	72
Paul Brown	90
Zack Brown	12
Bruce Byfield	40, 44, 62
Joe Casad	3
Mark Crutch	65
Marco Fioretti	82
Jon "maddog" Hall	67
Frank Hofmann	68
Charly Kühnast	48
Christoph Langner	22, 28
Vincent Mealing	65
Pete Metcalfe	58
Martin Mohr	56
Graham Morrison	76
Mandy Neumeyer	68
Mike Schilli	50

Contact Info

Editor in Chief
Joe Casad, jcasad@linux-magazine.com

Copy Editor
Amy Pettle

News Editor
Swapnil Bhartiya

Editor Emerita Nomadica
Rita L Sooby

Localization & Translation
Ian Travis

Layout
Dena Friesen, Lori White

Cover Design
Lori White

Cover Image
© Marina Andrienko, 123RF.com

Advertising
Brian Osborn, bosborn@linuxnewmedia.com
phone +49 89 3090 5128

Marketing Communications
Gwen Clark, gclark@linuxnewmedia.com
Linux New Media USA, LLC
2721 W 6th St, Ste D
Lawrence, KS 66049 USA

Publisher
Brian Osborn

Customer Service / Subscription
For USA and Canada:
Email: cs@linuxpromagazine.com
Phone: 1-866-247-2802
(Toll Free from the US and Canada)

For all other countries:
Email: subs@linux-magazine.com

www.linuxpromagazine.com – North America
www.linux-magazine.com – Worldwide

Printed in Nuremberg, Germany by hofmann infocom GmbH on recycled paper from 100% post-consumer waste; no chlorine bleach is used in the production process.

Distributed by Seymour Distribution Ltd, United Kingdom

LINUX PRO MAGAZINE (ISSN 1752-9050) is published monthly by Linux New Media USA, LLC, 2721 W 6th St, Ste D, Lawrence, KS, 66049, USA. Periodicals Postage paid at Lawrence, KS and additional mailing offices. Ride-Along Enclosed. POSTMASTER: Please send address changes to Linux Pro Magazine, 2721 W 6th St, Ste D, Lawrence, KS 66049, USA.

Published monthly in Europe as Linux Magazine (ISSN 1471-5678) by: Sparkhaus Media GmbH, Zieblandstr. 1, 80799 Munich, Germany.

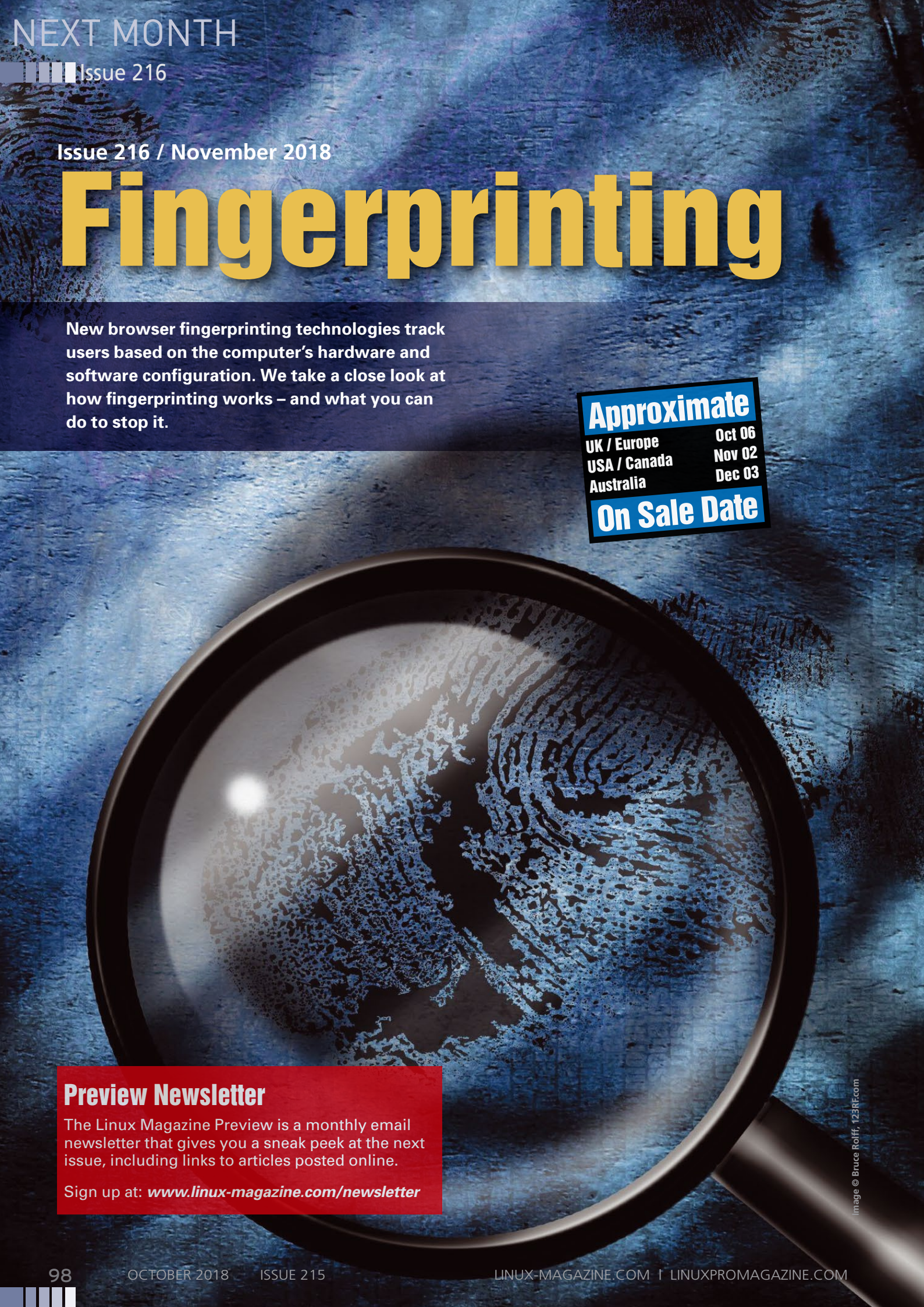
Issue 216 / November 2018

Fingerprinting

New browser fingerprinting technologies track users based on the computer's hardware and software configuration. We take a close look at how fingerprinting works – and what you can do to stop it.

Approximate On Sale Date

UK / Europe	Oct 06
USA / Canada	Nov 02
Australia	Dec 03



Preview Newsletter

The Linux Magazine Preview is a monthly email newsletter that gives you a sneak peek at the next issue, including links to articles posted online.

Sign up at: ***www.linux-magazine.com/newsletter***

Complete Your Open Source Library with Archive DVDs!

Each fully-searchable archive DVD includes past articles so you can find the content you need quickly.

Save hundreds off the print and digital copy rate with a convenient archive DVD!

Order Your DVD Now!

shop.linuxnewmedia.com

BUYING AND SELLING COMPUTING POWER SIMPLIFIED BY BLOCKCHAIN

750+ DAILY PARTICIPANTS NOW SHARING CLOUD COMPUTING RESOURCES USING SUBUTAI

VISIT https://subutai.io/

Cloud computing is expensive and dominated by a few major conglomerates with no interest in user privacy.

The Subutai platform leverages peer-to-peer cloud computing, energy-efficient cryptocurrency mining, and blockchain technology to provide you with a holistic solution to this problem. We're making it easy to utilize untapped computing resources. Focusing on flexibility, users are free to choose to utilize one, some, or all Subutai products:

Subutai™ PeerOS v7.0

Open Source container-based P2P Cloud and IoT software and firmware allows users to create virtual private clouds across idle computer resources.

Subutai™ Bazaar

"The Airbnb of cloud and IoT computing resources": global reputation-based marketplace for trading computing resources and applications. Buy/sell Bazaar products using GoodWill, the Subutai sidechain smart token.

Subutai™ Blockchain Router v2.0

Open hardware, advanced P2P Cloud router and IoT gateway. Eco-friendly, "green" broadband router serves as a plug-and-play cryptocurrency wallet and mining device. Drawing just 18-watts, FPGA-based miner is 1,083% more efficient than GPUs/ASICs.

Subutai™ Blueprints

Templates that reduce system administration overheads and simplify the process of application deployment. Deploy anything from Blockchain-in-a-Box to Wordpress to GitLab to NextCloud to a Minecraft server and more in a few easy clicks.

KHAN™

The Ethereum blockchain-based reserve currency and staking token of the Subutai Platform. Utility token used for provider and consumer reputation "staking" via Service Level Agreement escrows; part of dual token model; details at https://subutai.io/khan.html

JOIN THE HORDE TODAY!

https://bazaar.subutai.io/