



imgp: Batch editing for image files



FREE DVD

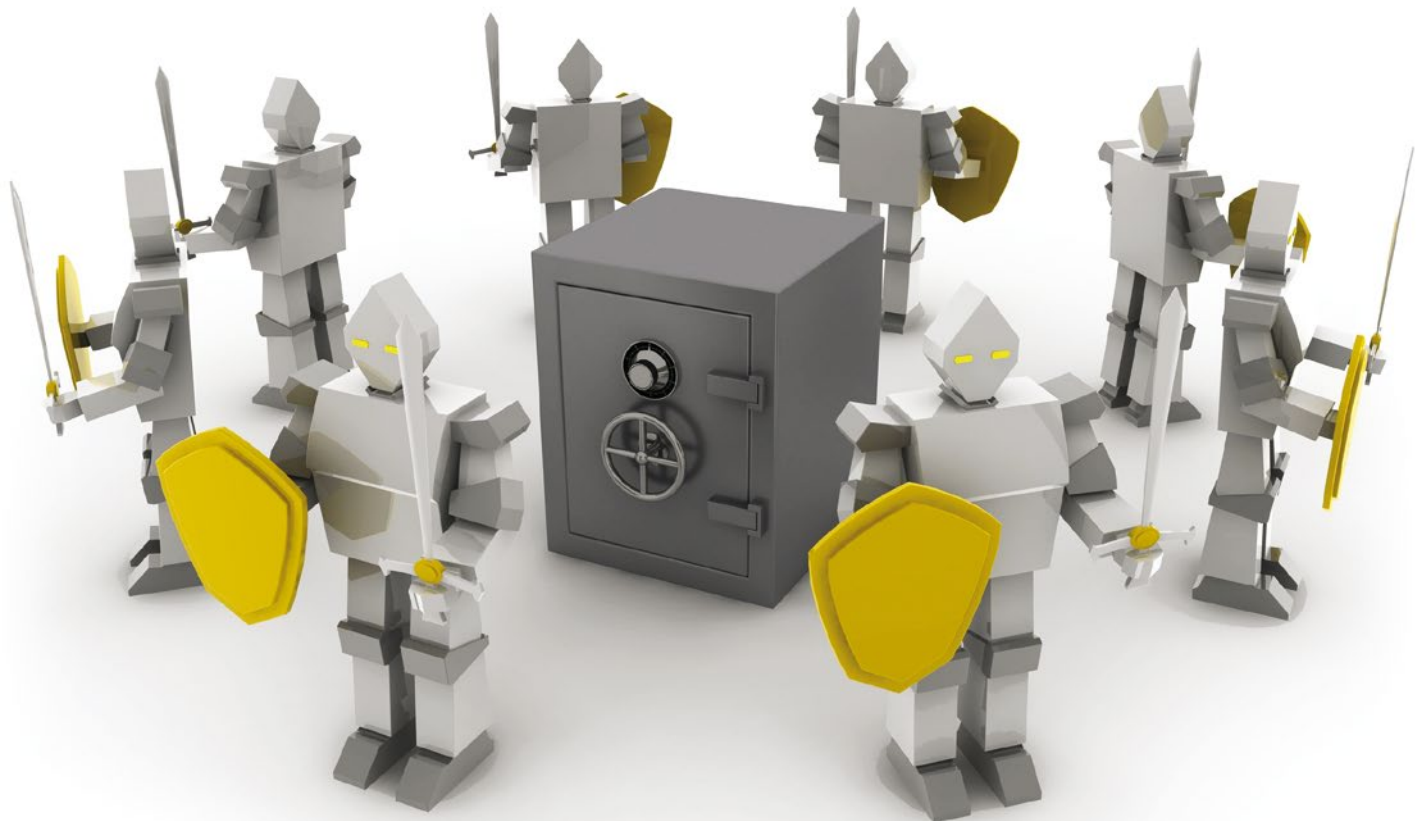
LINUX **PRO**

MAGAZINE

ISSUE 252 – NOVEMBER 2021

Locked Down!

New tools for tighter security



LINUXVOICE

Build Your Own Saltwater Battery

Pandoc: Convert Markdown text to a web page

ufw: Easy firewall configuration

KXStitch: Create cross-stitch patterns with this handy embroidery aid

10

FABULOUS FOSS TOOLS



LINUX NEW MEDIA
The Pulse of Open Source

Meet Me in St. Louis.

HPC is fueling breakthroughs across industries in science and beyond, from academia to commercial enterprises, by harnessing data to unlock insights faster and more accurately than ever before. HPC powers new capabilities in artificial intelligence and machine learning that, combined with modeling and simulation, accelerate discovery in solving our toughest challenges.

Join us in St. Louis to learn how HPC is empowering innovations that were never before possible to improve everyday life across the globe. Register early and save!

Program

14–19

NOVEMBER

Exhibits

15–18

NOVEMBER

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



Register Today!

sc21.supercomputing.org

Sponsored by:   |  

RULES FOR THE GARDEN

Dear Reader,

There is this abiding faith within the Linux world that walled gardens and single-vendor control over global markets will eventually collapse under the weight of inefficiency. If you look at the history of the last 20 years, however, you would see a very different story. Just when the menace of the Microsoft monopoly began to subside, a whole new generation of walled gardens grew up around the mobile phone industry and the pecunious package managers we know as *app stores*. Apple, for instance, controls every aspect of their app store, forcing developers to agree to terms where Apple manages all the money and restricts the vendor's access to sales data and other background information. Google, owner of Android, is a little more open than Apple, but that isn't saying much. Critics contend that both companies stifle competition through their heavy-handed control. Despite the excesses of the modern mobile app store, the case for any significant antitrust action within existing US law is not so clear. Apple, after all, has a relatively small share of the global cell phone market, even if they do exert monopoly-like control over the users who happen to fall into their walled garden. It is one of those cases where some direct and unambiguous legislation that leveled the playing field and created more competition would really help.

Last month, it seems, some real legislation did actually appear in the US Senate. The Open App Markets Act is an attempt to force competition into the mobile app marketplace. The bill would only apply to app stores with 50 million or more users, so it is clearly aimed at companies like Google and Apple, rather than at the many small-time vendors who are attempting to compete with the giants. If it passes, the bill would have several important effects. For instance, it would:

- Stop app stores from forcing developers into using the app store's own payment system
- Prohibit app store companies from punishing vendors who sell their apps for less elsewhere
- Stop app store companies from using non-public sales data to create and market their own apps that compete with other products in the store

Note that this bill will *not* do anything about the other big global problems that commentators like me like to complain about. It does not stop user tracking or the epidemic of misinformation raging on the uncurated web. The focus is much more narrow, but maybe that narrow focus will increase the likelihood of success. The primary beneficiaries would be independent and third-party app

developers, although the public as a whole would benefit if the measures lead to increased competition and a more open market.

As of now, the Open App Markets Act appears to have bipartisan support, but the process is just getting started, and don't expect Google and Apple to sit back and stay on the sidelines. Many promising ideas have met their fate in the US Senate, where politicians of both parties often find it safer to "take a stand" against things rather than taking a stand in favor. Lobbying efforts have already started to paint the bill as overreach or as pointless tinkering in an industry that is doing fine on its own. The lobbyists have many cards to play. The high tech industry, in general, is particularly adept at walking both sides of the street, appealing to conservative voters by sending the alarm that "government is interfering in business" and appealing to progressives with the warning that "government is intervening to restrict Internet freedom."

If the bill survives, however, it could be a promising first step at addressing a problem that has been percolating in the industry for years: the power of big companies to make themselves even bigger at the expense of everyone else.



Joe Casad,
Editor in Chief



ON THE COVER

34 Pandoc

Why break your head on HTML? Write the text in Markdown and generate a web page at the command line.

60 Saltwater Battery

Roll your own battery using saltwater, and then test it with a Raspberry Pi.

74 ufw

This easy-to-use interface takes the mystery out of firewalls.

78 KXStitch

Linux has a tool for everything – including the popular craft of cross-stitching.

NEWS

08 News

- System76 Updates the 15" Pangolin with Ryzen
- KDE Plasma 5.23 Promises Plenty of Subtle Improvements
- Gnome 41 Now in Beta
- Debian 11 "Bullseye" Released
- AlmaLinux Makes CentOS SIG Repositories Available

11 Kernel News

- Bug Hunting and Process Appreciation
- Refusing "Useful" Patches

COVER STORIES

14 Doas

The Sudo privilege management tool is big and complicated, with many advanced options. Doas is far simpler – which might just make it safer for desktop users.

18 OpenPGP Smartcard

Improve communication security with GnuPG and the OpenPGP smartcard.

24 Nitrokey Pro 2

The Nitrokey Pro 2 is a small device that covers a wide range of cryptographic functions.

26 SiriKali

SiriKali encrypts files and directories with just a few mouse clicks, without the inefficiency of fixed-size containers.

REVIEW

30 Distro Walk – Redcore Linux

With its point-and-click installation, Redcore aims "to be to Gentoo what Manjaro is to Arch Linux."

IN-DEPTH

34 Pandoc Website

Build a simple web page in Markdown and then convert it to HTML at the command line.

38 Command Line – lsd

One of several new revamps of ls, lsd offers color coding plus revised options relevant to the modern computer.

41 Charly's Column – Age

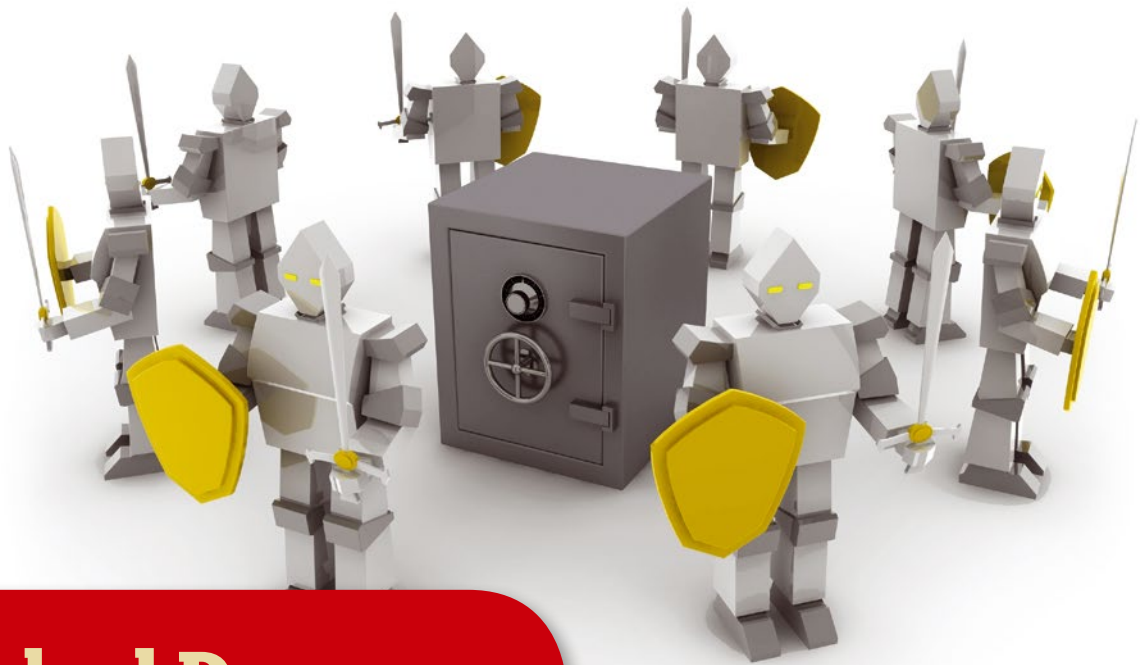
Charly makes life easier for himself by using the lean Age tool for command-line data encryption tasks.

42 imgp

In no time at all, imgp can change the resolution of images, as well as convert files from PNG to JPEG, remove metadata, and rotate images.

46 Epoptes

If your school's computer lab consists of Linux machines, Epoptes provides an interesting alternative to conventional management and monitoring programs.



Locked Down

The security landscape keeps changing, and experienced users know they need to keep their eyes open for tools and techniques that give an edge. This month we study smartcards, hard drive encryption, and a less-bloated alternative to Sudo.

IN-DEPTH

50 Programming Snapshot – Go GPS Data Retrieval

The hiking and cycling app komoot saves your traveled excursion routes. Mike Schilli shows you how to retrieve the data with Go.

58 Popsicle

If you need to flash the same image to multiple USB media, Popsicle saves time by letting you write in parallel.

MakerSpace

60 Saltwater Battery

Saltwater batteries offer a cheaper and greener approach to storing energy.

66 Ren'Py

Ren'Py helps you create Android, Linux, macOS, Windows, and HTML5 games and apps.

LINUXVOICE

71 Welcome

This month in Linux Voice.

73 Doghouse – Bug Reports

For a better bug report, maddog offers a refresher course on crafting a clear statement that will help get your problem fixed.

74 ufw

Canonical's ufw lets you configure your firewall without the hassle of the iptables tool, while reducing the risk of misconfiguration and simplifying maintenance.

78 KXStitch

Modern technology and new programs can revive old handicrafts. KXStitch helps design cross-stitch patterns and even automatically converts imported images.

82 FOSSPicks

This month Graham looks at MyGNUHealth, Sniptt, Pigiron, CudaText, KnobKraft Orm, D2X-XL, and more!

88 Tutorial – HedgeDoc

HedgeDoc lets you write documents collaboratively in Markdown and publish them online.



TWO TERRIFIC DISTROS
DOUBLE-SIDED DVD!

SEE PAGE 6 FOR DETAILS

Debian 11 and Redcore Linux 2101

Two Terrific Distros on a Double-Sided DVD!



Debian 11
64-bit

Compared to other distributions, Debian releases are rare. Debian 11, code-named Bullseye, is the first Debian release in two years. As a long-term support release, it next will be updated in 2026.

Debian has never tried to be a cutting-edge release, and Debian 11 is no exception. Its new features are conservative. For example, systemd now turns on journal logs by default, and security is enhanced by using yescrypt instead of SHA-512. One new package treats printers as a network device and extends driverless printing for both printers and scanning, while a new `open` command starts files from the command line with a specified app. And as usual, the new release has a new default desktop wallpaper. Otherwise, packages are updated from Debian 10 but are a release or two behind the latest available release.

So why install Debian 11? The answer is that Debian is second to none when it comes to stability and security, including speedy updates. That is why network administrators and security experts overwhelmingly favor Debian. If stability and security are your main concerns, then Debian is the distribution for you.



Redcore Linux 2101
64-bit

To understand the appeal of Redcore Linux, you need to understand the appeal of Gentoo Linux. Two decades ago, Gentoo Linux was a leading distribution. Compiling packages from source, it delivered optimized installations far faster than most distributions but required hours of compiling and consulting documentation. Gentoo continues to thrive today, but its lack of a typical installer means that its popularity dwindled as desktop Linux arose.

That's where Redcore comes in. A successor to Kogaion Linux, Redcore draws on Gentoo's stable and unstable repositories and provides an installer for a quicker way to install a pure Gentoo system. If the installation still takes about twice the time as an Ubuntu installation – perhaps half an hour – that is still a vast improvement over Gentoo itself. After installation, Redcore provides a rolling release to keep your system updated.

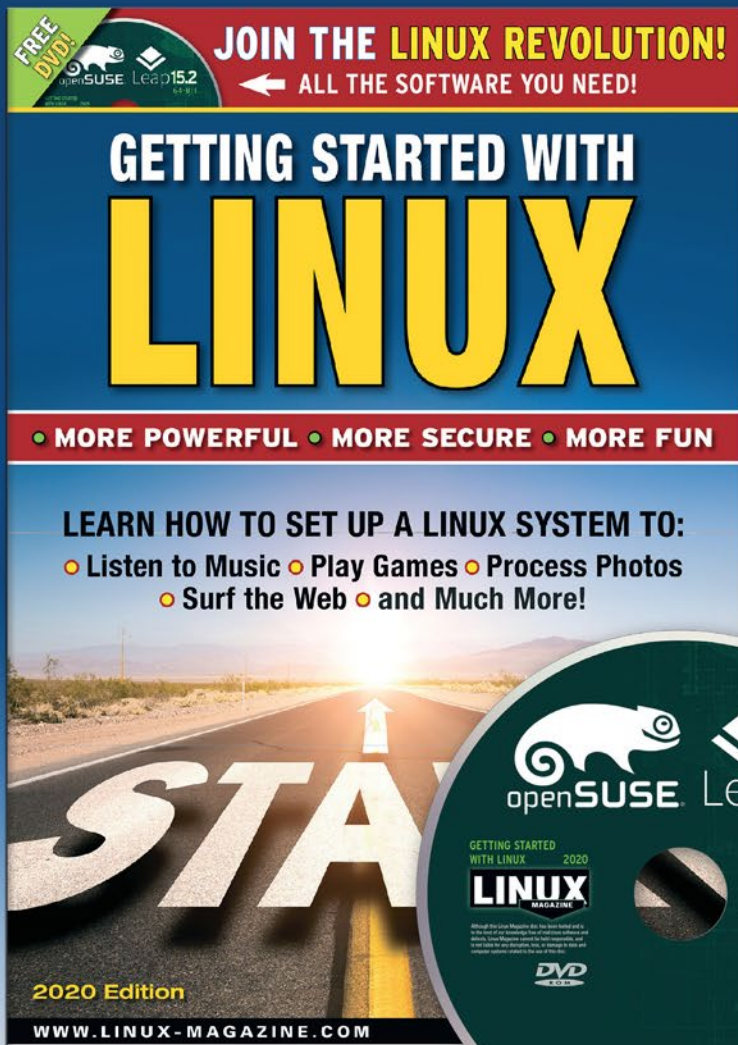
For those who have wondered what Gentoo was about but never had the time to investigate, Redcore provides a painless way to find out. Start Redcore after installation, and you will quickly understand why an optimized system is so appealing.

Defective discs will be replaced.

Please send an email to subs@linux-magazine.com.

Although this Linux Magazine disc has been tested and is to the best of our knowledge free of malicious software and defects, Linux Magazine cannot be held responsible and is not liable for any disruption, loss, or damage to data and computer systems related to the use of this disc.

Hit the ground running with Linux



Want your friends and colleagues to make the switch to Linux?

This single issue shows beginners how to:

- install Linux
- download and install free software for your Linux system
- play games
- create documents and spreadsheets
- process photos
- play music and videos
- and much more!



ORDER ONLINE: shop.linuxnewmedia.com/specials

NEWS

Updates on technologies, trends, and tools

THIS MONTH'S NEWS

- 08** • System76 Updates the 15" Pangolin with Ryzen
- KDE Plasma 5.23 Promises Plenty of Subtle Improvements
- 09** • Gnome 41 Now in Beta
- More Online
- 10** • Debian 11 "Bullseye" Released
- AlmaLinux Makes CentOS SIG Repositories Available

System76 Updates the 15" Pangolin with Ryzen

System76 (<https://system76.com/>) is not just one of the most popular makers of Linux-only hardware, they're also really good at pushing all of the boundaries and reinventing all of the wheels. This latest move might not push or reinvent, but it certainly gives consumers options for more mobile power.

But this isn't news, right? The Pangolin was already available with an AMD Ryzen processor. This is a situation where out with the old, in with the new applies because the company is upping the Ryzen ante by offering the 15" Pangolin with a Ryzen 7 5700U CPU.

Of course, along with this much-improved CPU comes a slightly steeper price. The previous Ryzen model starts at \$1,199 and the new version starts at \$1,398 (due to the \$199 CPU upgrade). Both iterations include the same base specs (minus the newer CPU). You can purchase the Pangolin with the older Ryzen 5 5500U (with 2.1 up to 4.0 GHz, 8MB cache, 6 cores, and 12



© lynea, Fotolia.com

threads), or you can opt to upgrade to the Ryzen 7 5700U (with 1.8 up to 4.3 GHz with 8MB cache, 8 cores, and 16 threads).

The base specs for both machines include 8GB of RAM and a 240 GB SSD, and both ship with Pop!_OS.

Find out more about the Pangolin laptop (<https://system76.com/laptops/pangolin>).

KDE Plasma 5.23 Promises Plenty of Subtle Improvements

KDE Plasma 5.23 is set for release on October 7, 2021, and it promises several (subtle) improvements across the board. This new release is powered by Qt 5.15

and KDE Frameworks version 5.86. Along with the improvements found in those software stacks, there's plenty of goodness to go around within KDE itself.

Many of the new features and improvements are all about refinement, so you probably won't see anything that's mind-blowing or drastic. For example, one of the highlighted new features is that you can specify an alternative accent color from that which is configured by the theme. Other subtle newness includes the ability to choose list or grid view for all Kickoff items (so it no longer only applies to Favorites), a new QML-based Overview effect has been introduced that is similar to Gnome's Workspace view, the Dolphin file manager can be configured to display hidden files/folders first, and selected items can be deleted from the clipboard pop-up.

There have also been some updates to Wayland support, including that copying text from notifications now works as expected, and middle-click paste has finally returned (huzzah!).

Much of the improvement for KDE Plasma 5.23 comes under the hood and should bring a significant bump in performance.

For those interested in testing the new version, download the Unstable Edition of KDE Neon (<https://neon.kde.org/download>). Just remember, do not use this release for production environments.



© DeVlce, Fotolia.com

Gnome 41 Now in Beta

Gnome 40 brought to life some major changes to the desktop environment (such as the horizontal workflow). Although topping that release would be a monumental task, the developers do have some tricks up their sleeves for the soon-to-be-released Gnome 41. And now that the desktop environment is in beta, you can experience those tweaks for yourself.

The big-ticket items for this upcoming release include a new multitasking panel in the Settings window. This new Settings tab allows you to configure Hot Corners,

Active Screen Edges, Workspace options, and app-switching preferences.

Another new option found in the Settings tool is called Cellular, which allows users to configure mobile connections and modems. This option will only appear when Gnome recognizes the necessary hardware for that purpose.

The Gnome Software tool is also getting a few tweaks, most of which are merely cosmetic. You'll find the Power Profiles tool (introduced in Gnome 40) has evolved to make it easier to quickly switch between profiles. Those profiles have also been better integrated into the system. For example, the low power profile is now automatically activated



Photo by Sarah Brink on Unsplash

when a laptop battery reaches a low level.

A new remote desktop app, called Connections, has been introduced, which makes it possible to easily connect and switch between multiple remote desktop sessions.

Finally, there have been numerous boosts to the performance of the desktop, so expect it to run even better than Gnome 40 (which should be impressive).

MORE ONLINE

Linux Magazine

www.linux-magazine.com

ADMIN HPC

<http://www.admin-magazine.com/HPC/>

Darshan I/O Analysis for Deep Learning Frameworks

• Jeff Layton

The Darshan userspace tool is often used for I/O profiling of HPC applications.

Prolog and Epilog Scripts

• Jeff Layton

Virtually all HPC systems use a resource manager, also called a job scheduler.

ADMIN Online

<http://www.admin-magazine.com/>

Best Practices for Secure Script Programming

• Tam Hanna

The lax syntax verification of shell scripts and a lack of attention to detail in programming can create impressively dangerous security vulnerabilities.

Tinkerbell Life-Cycle Management

• Martin Loschwitz

Tinkerbell specializes in bare metal deployment and life-cycle management, allowing intervention in every phase of the setup.

Fast and Scalable ownCloud Infinite Scale

• Martin Loschwitz and Markus Feilner

A complete rebuild of ownCloud delivers the speedy and scalable ownCloud Infinite Scale file platform.

Find out more about what's been improved and added in the official announcement (<https://mail.gnome.org/archives/devel-announce-list/2021-August/msg00002.html>).

Debian 11 "Bullseye" Released

Debian is often called the "Mother of distributions" because it is used by so many Linux variants as a base. Ubuntu is based on Debian, which is in turn used by many developers to create other distributions, which helps to verify Debian as the mother of so many distributions. Although you might think, given Debian 10 was released in 2019, that Debian 11 would come with a massive amount of new features, don't get too excited. Although there is a good number of new features, the bulk of Debian 11 is updates to already-included packages.

In total, there are 11,294 new packages and 42,821 updated packages. Those are some pretty staggering numbers, which clearly indicate the developers have been working hard to bring this new release to life.

One of the biggest additions to Debian is support for exFAT filesystems. With this new addition, users no longer have to employ exFAT-FUSE. All of this has been rolled into the `exfatprogs` package.

Another exciting addition is that most modern printers can now use driverless printing and scanning, without the need for third-party software. This is accomplished with `ipp-usb`, which uses the vendor-neutral IPP-over-USB protocol supported by most newer printers.

Desktop environment updates include Gnome 3.38, KDE Plasma 5.20, LXDE 11, LXQt 0.16, MATE 1.24, and Xfce 4.16.

To find out more about what's new with Debian 11 "Bullseye," check out the official release notes (<https://www.debian.org/News/2021/20210814>).



© Maurizio Ascione, Fotolia.com

AlmaLinux Makes CentOS SIG Repositories Available

SIGs (Special Interest Groups) are groups within the CentOS community focusing on a specific set of issues. For example, there's the Storage SIG, Virtualization SIG, Infrastructure SIG, Core SIG, Automotive SIG, and the Hyperscale SIG. Each SIG releases its own repositories which contain packages related to a chosen focus.

Once upon a time, users would have to locate a specific repository and then download/install the relevant `centos-release-*` packages just to enable that particular repository. AlmaLinux decided to simplify that process by releasing upstream CentOS Special Interest Groups' release packages.

Even at this early release stage, you can already enable SIGs for Cloud, Config Management, Messaging, Network Functions Virtualization, Ops Tools, Storage, and Virtualization. To find out how to enable these SIGs, head over to the AlmaLinux CentOS SIGs Repository page.

As to why AlmaLinux went this route, they've said, "By encouraging developers/contributors interested in the various SIG development to work upstream and submit contributions there (and submit bugs), we can ensure this work remains centralized and continues to be part of the long-term stability, success, health, and vitality of the community."

For SIG developers, AlmaLinux has this to say, "Instead of building against CentOS 8 and CentOS Stream 8, SIGs should build against RHEL 8 and CentOS Stream 8! This would ensure that the SIG's output is consumable by the entire RHEL family of distributions, including rebuilds like AlmaLinux."



**Get the latest news
in your inbox every
two weeks**

**Subscribe FREE
to Linux Update
bit.ly/Linux-Update**

Zack's Kernel News



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

By Zack Brown

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**.

Bug Hunting and Process Appreciation

There was a fun little bug hunt recently, when Mikael Pettersson reported that Linux 5.10.56 wouldn't boot on his system. This was particularly interesting because the 5.10 kernel is the "stable" series, meaning it receives long-term support from Greg Kroah-Hartman.

The stable vs. development dichotomy is one that has gradually emerged over many years. Originally, there was no such distinction, and Linus Torvalds simply released new patches and new versions as developers submitted them. The kernel was generally stable, but there was not yet any concept of a series of releases that actually aimed for stability, either in terms of bugs or features.

Eventually, Linus began to alternate between stable and development cycles. During the stable cycle, he would accept almost exclusively bug fixes and no new features. During the development cycle, it was a free-for-all, with everyone's favorite project hurtling headlong into the kernel.

But that approach was reviled, especially by developers who had to vacuum-style breathe during the stabilization periods. Eventually Linus hit upon a better approach, which is the current Release Candidate (RC) cycle. With this technique, he puts out a series of RC kernel versions over the short term. Each of these have increasing stability goals, until the final RC becomes the official kernel release.

With the RC technique, developers have frequent opportunities to hurl their ongoing work in Linus's direction, with relatively short periods of stabilization in between.

The result is more stable than simply doing ongoing hard-core development, but it's not as stable as a lot of users, especially corporate users, need. So, Greg decided to pick certain official releases and do his own stabilization cycle (with massive community assistance).

The 5.10 series, as one of Greg's picks for the stable cycle, is therefore relied on by, well, everyone. A 5.10 kernel that fails to boot is a 5.10 kernel that will receive a lot of TLC quickly.

So when Mikael reported that 5.10.56 wouldn't boot on his system, it was always going to be a thing. In this case, he knew that he did not have the problem before the 5.10.47 release, and he noticed it on the 5.10.56 release. So, one of the patches that went into the Git tree between those two releases had to be the culprit.

It's worth mentioning that the git tool is Linus's other successful attempt to take over the world. And with git, Mikael was able to bisect the revision log in order to find exactly which patch introduced the problem. In fact, he had a slightly tougher time than just simply doing that, but, yes, he did that.

Bisecting is simply a binary search, and everyone uses it in kernel development and other software projects. Since Mikael had one version that was known to work and one version that was known to be broken, he could choose the patch directly in the middle of those two, and test it. If it was also broken, he could choose the patch directly in the middle between this test version and the known working version. By jumping from midpoint to midpoint in this way, each test cuts in half the total number of patches that might be causing the problem. For ex-

ample, if Mikael had 4,000 patches to sift through, it would take him only 12 tests to find the bad one using the bisection technique.

In fact, Mikael made a mistake with his bisection – at first he thought he had bisected to one particular patch that came after Linux version 5.10.52.

Finn Thain responded to this bug report, pointing out that the very same patch also went into Linus Torvalds's main-line Git tree, which would make the problem even more widespread. Mikael tested the main-line tree and did not see the problem. He said, "I suspect the commit has some dependency that hasn't been backported to 5.10 stable."

Finn continued the hunt, trying to isolate the potential bug in the main-line tree, on the assumption that even if Mikael couldn't reproduce the problem locally, the bad code from the stable branch was still also in the main-line tree and needed to be rooted out.

But at this point, while helping Finn track the problem into the main-line tree, Mikael realized he had made a mistake with his initial bisection. He ran it again, this time marking all the known good versions of the kernel, and the bisection landed on a different patch. Reverting that patch fixed the problem.

Michael Rapoport looked at the specific patch Mikael was talking about and confirmed that, yes, there did seem to be something wrong in that code. He posted a two-line patch and asked Mikael to test it out. Mikael did and confirmed that this fixed his boot problem.

For me, the most interesting parts of kernel development are not new features or new hardware support but, instead, the development process itself, as it has evolved and continues to evolve over time: The emergence of the long-term stable tree, the patterns of stabilization in the development tree, the invention of Git and the strange twists and turns that preceded it.

In this particular case, I find the relatively quick and straightforward process of identifying a bug and fixing it interesting. If you look at other software projects, bug hunting is not necessarily this quick and straightforward. The reason it is in this case is because of additional policies and practices put in place by Linus. The requirement is that the kernel should at the very least be com-

patible and bootable after every patch. Not all software projects have that requirement. But because of this requirement, something like Mikael's bisection process could be largely automated, the problem patch quickly identified, and the fix found.

These kernel development aspects are not just little details. Linus receives such a massive truckload of patches each month that when his work first started to be trackable via revision control a lot of long-time developers were shocked at the sheer quantity of it all. It simply hadn't occurred to anyone that a single person could be the focus of so many contributions and still be able to handle it all. It's the development of these essentially social aspects of kernel development that make such things possible.

Refusing "Useful" Patches

There is a certain type of patch that would improve things for certain users but that nevertheless has a near-zero chance of ever getting into the kernel. These are patches that help users recover from things they shouldn't have been doing in the first place. And one reason those patches don't tend to go into the kernel is that it is really a bottomless pit.

The motivations, however, are pure. For example, Florian Weimer recently said, "We have a persistent issue with people using cp (or similar tools) to replace system libraries. Since the file is truncated first, all relocations and global data are replaced by file contents, result[ing] in difficult-to-diagnose crashes. It would be nice if we had a way to prevent this mistake."

This was part of a discussion (and patch series) dealing with file permissions and related control over who could write to a filesystem and when.

There started to be some discussion about how to block users from doing this particular thing. And, at a certain point, Linus Torvalds brought the hammer down, saying:

"This is definitely a 'if you overwrite a system library while it's being used, you get to keep both pieces' situation.

"The kernel ETXTBUSY thing is purely a courtesy feature, and as people have noticed it only really works for the main executable because of various reasons.

It's not something user space should even rely on, it's more of a 'ok, you're doing something incredibly stupid, and we'll help you avoid shooting yourself in the foot when we notice'.

"Any distro should make sure their upgrade tools don't just truncate/write to random libraries executables.

"And if they do, it's really not a kernel issue."

The whole discussion took place amidst a patch submission from David Hildenbrand that addressed file permissions and related user controls, which is why the issue came up. But Linus added, "This patch series basically takes this very historical error return, and simplifies and clarifies the implementation, and in the process might change some very subtle corner case (unmapping the original executable entirely?). I hope (and think) it wouldn't matter exactly because this is a 'courtesy error' rather than anything that a sane setup would `_depend_` on."

Still, Florian's concerns are eternally compelling. And Eric W. Biederman replied to Linus, "I am trying to come up with advice on how userspace implementations can implement their tools to use other mechanisms that solve the overwriting shared libraries and executables problem that are not broken by design."

Eric went on to say, "today the best advice I can give to userspace is to mark their executables and shared libraries as read-only and immutable. Otherwise a change to the executable file can change what is mapped into memory."

And while Eric agreed that this was fundamentally not a kernel issue, he did add, "What is a kernel issue is giving people good advice on how to use kernel features to solve real world problems. I have seen the write to a mapped executable/shared lib problem, and Florian has seen it. So while rare the problem is real and a pain to debug."

He added, "As I am learning with my two year old, it helps to give a constructive suggestion of alternative behavior instead of just saying no. Florian reported that there remains a problem in userspace. So I am coming up with a constructive suggestion. My apologies for going off into the weeds for a moment."

In the course of the discussion, Linus also had some interesting comments about GNU Hurd, the operating system that was supposed to be the crown jewel of the GNU system before Linux came along and stole its thunder.

Eric had said at one point, “Given that MAP_PRIVATE for shared libraries is our strategy for handling writes to shared libraries, perhaps we just need to use MAP_POPULATE or a new related flag (perhaps MAP_PRIVATE_NOW) that just makes certain that everything mapped from the executable is guaranteed to be visible from the time of the mmap, and any changes from the filesystem side after that are guaranteed to cause a copy on write.”

And Florian had remarked, “I think this is called MAP_COPY:” and gave a link to some GNU Hurd documentation [1]. He added, “If we could get that functionality, we would certainly use it in the glibc dynamic loader. And it’s not just dynamic loaders that would benefit.”

But now Linus really put on his stomping boots, saying:

“Please don’t even consider the crazy notions that GNU Hurd did.

“It’s a fundamental design mistake. The Hurd VM was horrendous, and MAP_COPY was a prime example of the kinds of horrors it had.

“I’m not sure how much of the mis-designs were due to Hurd, and how much of it due to Mach 3. But please don’t point to Hurd VM documentation except possibly to warn people. We want people to _forget_ those mistakes, not repeat them.”

Returning to the main topic, Linus went on:

“I’ll just repeat: stop arguing about this case. If somebody writes to a busy library, THAT IS A FUNDAMENTAL BUG, and nobody sane should care at

all about it apart from the ‘you get what you deserve’.

“What’s next? Do you think glibc should also map every byte in the user address space so that user programs don’t get SIGSEGV when they have wild pointers?”

“Again – that’s a user BUG and trying to ‘work around’ a wild pointer is a worse fix than the problem it tries to fix.

*“The exact same thing is true for shared library (or executable) mappings. Trying to work around people writing to them is *worse* than the bug of doing so.*

“Stop this completely inane discussion already.”

Andy Lutomirski, never one to simply sit down and shut up, replied, “How about we attack this in the opposite direction: remove the deny write mechanism entirely. In my life, I’ve encountered -ETXTBUSY intermittently, and it invariably means that I somehow failed to finish killing a program fast enough for whatever random rebuild I’m doing to succeed. It’s at best erratic – it only applies for static binaries, and it has never once saved me from a problem I care about. If the program I’m recompiling crashes, I don’t care – it’s probably already part way through dying from an unrelated fatal signal. What actually happens is that I see -ETXTBUSY, think ‘wait, this isn’t Windows, why are there file sharing rules,’ then think ‘wait, Linux has *one* half baked file sharing rule,’ and go on with my life. Seriously, can we deprecate and remove the whole thing?”

To which Linus, always happy to continue a discussion after asking for it to end, replied, “I think that would be ok, except I can see somebody relying on it. It’s broken, it’s stupid, but we’ve done that ETXTBUSY for a _loong_ time. But you are right that we have removed parts

of it over time (no more MAP_DENYWRITE, no more uselib()) so that what we have today is a fairly weak form of what we used to do. And nobody really complained when we weakened it, so maybe removing it entirely might be acceptable.”

He added, “I guess we could just try it and see.... Worst comes to worst, we’ll have to put it back, but at least we’d know what crazy thing still wants it.”

Meanwhile Al Viro remarked, “I’m fairly sure that there used to be suckers that did replacement of binary that way (try to write, count on exclusion with execve while it’s being written to) instead of using rename. Install scripts of weird crap and stuff like that. [...] and before anyone goes off – I certainly agree that using that behaviour is not a good idea and had never been one. All I’m saying is that there at least used to be very random (and rarely exercised) bits of userland relying upon that behaviour.”

And David, whose patch inspired the whole conversation, replied, “Removing it completely is certainly more controversial than limiting it to the main executable. [...] I’d vote for keeping it in, and if we decide to rip it out completely, do it [as] a separate, more careful step.”

The discussion continued for a bit, with nothing being really decided. But it’s funny that rather than enhance a user protection against doing something dumb, Linus would rather remove the protection in its entirety, on the grounds that it’s not the kernel’s job to stop users from doing dumb things. ■■■

Info

[1] GNU Hurd documentation:
<https://www.gnu.org/software/hurd/glibc/mmap.html>

Doas authenticates as a simpler version of Sudo

Little Brother

The Sudo privilege management tool is big and complicated, with many advanced options that only an expert would need. Doas is far simpler – which might just make it safer for desktop users.

By Ferdinand Thommes

In Unix and Linux, best practices call for a strict segregation in the assignment of rights between daily work and administrative tasks. Administrative chores were once reserved for the superuser account. If you are logged in as the superuser, which is normally named *root*, you are allowed to do everything up to deleting the entire system.

Admins at New York University developed Sudo back in 1980 to prevent students from getting unneeded privileges. The name *Sudo* stands for “Superuser Do.” It lets you give privileges to a user who is a member of the *sudo* group for limited time or for a specific task. To do this, prepend *sudo* to a command whose execution requires these privileges. Then enter your user password to authorize it.

Sudo became more fashionable in Linux after it was adopted by Ubuntu, and it is now a standard feature of most distributions. Sudo sounds simple on the surface, but it is actually highly evolved software with many advanced features most desktop users never need. Rights assignments in Sudo can be regulated by role-based access controls [1] and by mandatory access controls [2] or configured via LDAP and the Network Information Service (NIS) directory service.

A very complex configuration file can result from all these options. All this complexity can quickly overtake newcomers, provoking errors which can compromise security.

Too Powerful for Home

The special powers of Sudo might make sense for a large and busy enterprise server, but many users believe Sudo is too big (with more than 412,000 lines of code) and too complicated for a single desktop user at home.

Because of its powerful and arcane feature set, Sudo is also a magnet for hackers. In the past, Sudo has been the subject of several security-critical bugs, most recently in January 2021 [3]. These bugs remained undiscovered for up to 10 years because of the complexity of the Sudo code.

Because the vast majority of users only need a minimal fraction of Sudo’s powers, for many, it makes sense to use an alternative. Doas has emerged recently as a compact counterpoint to Sudo. At 4,000 lines, Doas contains only one percent of the code in the bloated Sudo and thus represents a far smaller attack vector. The reduced functionality means Doas might not

be appropriate for complex enterprise environments, but it is all you need for many desktop systems.

Even on a system with several users, the Doas configuration file typically consists of a single line for each user, written in plain language. This simplicity reduces the likelihood of configuration errors, but Doas also lets you create more complex authorization systems if necessary.

Doas and OpenDoas

Doas development started about six years ago at OpenBSD, when Ted Unangst set out to implement 95 percent of Sudo’s functionality with a far smaller codebase. Doas is now the standard in some BSD distributions, and you can install it via the repositories of many Linux distributions using the *open-doas* package [4]. On Linux, Doas almost always takes the form of the OpenDoas fork. Versions of OpenDoas also run on macOS [5].

Version 6.8.1 of the actively developed OpenDoas is currently available. If your distribution offers an older version of OpenDoas or doesn’t have it at all, as is the case in Debian 10, you can get the source code from Github [6] and build it yourself using the three familiar Linux commands (Listing 1). If the last command fails, check if the path and filename match. You’ll also have to take care of the PAM configuration to secure Doas [7]. A suitable template for PAM is `/etc/pam.d/sudo`.

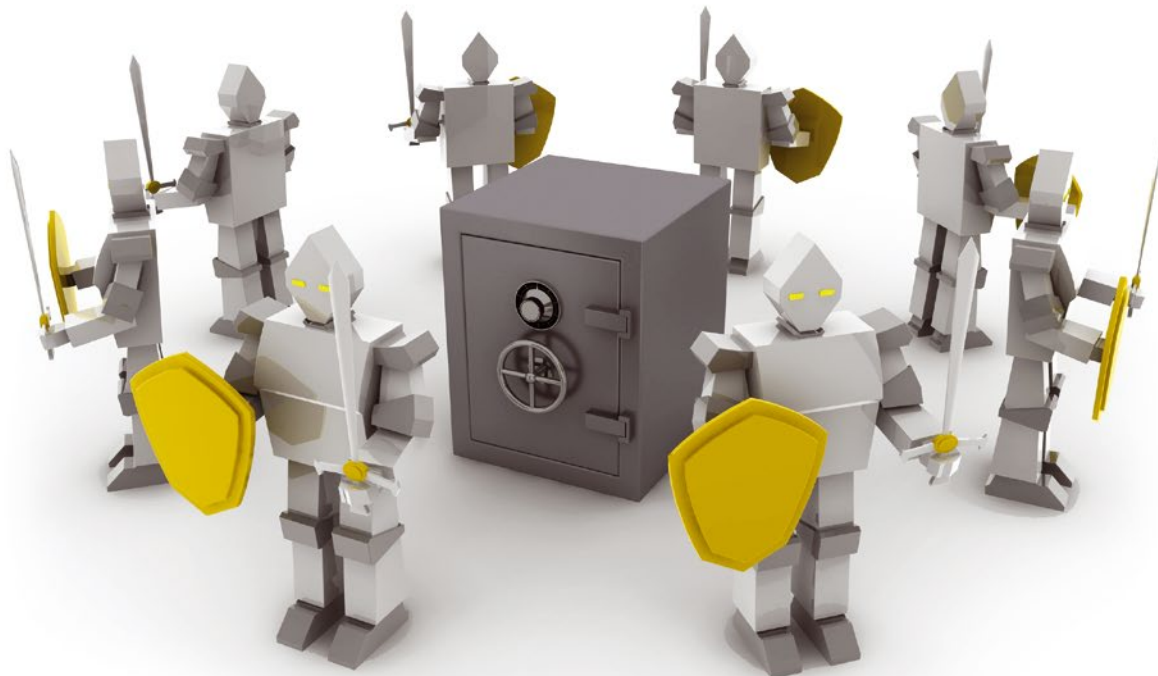
Listing 1: Build OpenDoas from the Source Code

```
# Preparation:
sudo apt install build-essential checkinstall git byacc

# Deploy
git clone git://github.com/Duncaen/OpenDoas --depth 1
cd OpenDoas

# build
./configure --with-timestamp --mandir=/usr/share/man
make
checkinstall -D --install=no --maintainer=user
--pkgname=openoas

# Install
sudo apt install /home/$USER/OpenDoas/openoas_*.deb
```



Listing 2: Entering Users and Groups

```
01 # Enter your own user account
02 echo "permit :$USER" | sudo tee /etc/doas.conf
03 # Define admin group as allowed group
04 echo "permit :admin" | sudo tee /etc/doas.conf
05 # Deny command for current user
06 echo "deny :$USER apt update && apt dist-upgrade" | sudo tee /etc/doas.conf
07 # Use without password for current user
08 echo "permit nopass :$USER" | sudo tee /etc/doas.conf
09 # Preserve environment variables
10 echo "permit keepenv :$USER" | sudo tee /etc/doas.conf
```

Once you have OpenDoas installed on your system, the best way to test it is to simply prepend `doas` instead of `sudo` to a command. If you see an error message, the problem is probably due to your distribution – some maintainers ship OpenDoas without a configuration file.

The config file is quickly created, along with the entry required for the current user, with a single command (Listing 2, Line 2) (Figure 1). On multi-user systems, you would authorize all members of the `admin` group with the command from Listing 2, Line 4.

If you want to configure deployment for another user, either add the user to a group or replace `$USER` with the appropriate account name in the commands in Listing 2.

The `deny` keyword in `/etc/doas.conf` denies users certain privileges (Listing 2, Line 6). On single-user systems, it might be a good idea to use the software entirely without a password (Listing 2, Line 8).

Most of the time it is not advisable to start GUI applications as `root`, but apps such as `Gparted` or `Krusader-Root` need these permissions already at startup. To allow these graphical applications with Doas, the `keepenv` parameter comes into play (Listing 2, Line 10).

```
GNU nano 5.4 /etc/doas.conf
permit keepenv nopass :ft
```

Figure 1: Usually a single line in the configuration is enough if there is only one user on the system.

Persistence and Autocompletion

OpenDoas, unlike `Sudo`, does not remember the password. `Sudo` stores the password for 15 minutes, but Doas theoretically requires you to enter it again with every command.

If you want to emulate the persistence of `Sudo` (15 minutes) with OpenDoas on Debian and derivatives, you need to compile the application yourself with the `--with-timestamp` option (Figure 2).

Another difference between Doas and `Sudo` is that Doas does not enable auto-completion in most distributions. The lack of an auto-completion feature means that, if you prefix the command with `doas`, the shell does not complete commands when you press the tab

```
ft@blue:~/OpenDoas$ ./configure --with-timestamp
Setting UID_MAX          65535.
Setting GID_MAX          65535.
Checking for explicit_bzero ... yes.
Checking for strlcat ... no.
Checking for strlcpy ... no.
Checking for errc ... no.
Checking for verrc ... no.
Checking for setprogname ... no.
Checking for readpassphrase ... no.
Checking for strtonum ... no.
Checking for reallocarray ... yes.
Checking for execvpe ... yes.
Checking for setresuid ... yes.
Checking for setresgid ... yes.
Checking for setreuid ... yes.
Checking for setregid ... yes.
Checking for closefrom ... no.
Checking for sysconf ... yes.
Checking for dirfd ... yes.
Checking for fcntl_h ... yes.
Checking for F_CLOSEM ... no.
Checking for dirent_h ... yes.
Checking for sys_ndir_h ... no.
Checking for sys_dir_h ... yes.
Checking for ndir_h ... no.
Checking for login_cap_h ... no.
Checking for __attribute__ ... yes.
Checking for pam_appl_h ... yes.
Using auth method        pam.
Using persist method     timestamp.
ft@blue:~/OpenDoas$
```

Figure 2: The `persist` option is only available if you build Doas yourself from source code. To do this, add the `--with-timestamp` parameter to the `./configure` command.

```

GNU nano 5.4 /usr/share/bash-completion/completions/doas
# bash completion for doas(8)                                -*- shell-script -*-
PATH=$PATH:/sbin:/usr/sbin:/usr/local/sbin ←
doas()
{
    local cur prev words cword split
    _init_completion -s || return

    local i mode=normal
    [[ $1 == *sudoedit ]] && mode=edit

    [[ $mode == normal ]] &&
        for ((i = 1; i <= cword; i++)); do
            if [[ ${words[i]} != -* ]]; then
                local PATH=$PATH:/sbin:/usr/sbin:/usr/local/sbin
                local root_command=${words[i]}
                _command_offset $i
                return
            fi
            if [[ ${words[i]} == -@(!(-*)e*)-edit ]]; then
                mode=edit
                break
            fi
        [[ ${words[i]} == \

```

Figure 3: Add the path variable as the second line of the completion file created previously as a workaround. If you then replace the occurrences of `sudo` with `doas`, Bash completion will also work after a restart.

key. This is annoying if you are used to autocompletion. A bug report from the author has not yet been processed, but you will find a workaround in the box entitled “Bash completion.”

Add an Alias

The options described thus far are sufficient for the needs of most desktop users. The problem of muscle memory causing you to automatically type `sudo` instead of `doas` is countered with an alias in the `.bashrc` file; just add the line `alias`

Bash Completion

First, create an empty configuration file for Doas (Listing 3). Copy the contents of `/usr/share/bash-completion/completions/sudo` to it, and replace all occurrences of `sudo` with `doas` (Figure 3). In the first line, add the following:

```
PATH=$PATH:/sbin:/usr/sbin:/usr/local/sbin
```

After rebooting the computer, Bash completion should work.

Listing 3: Creating a Configuration File

```
sudo touch /usr/share/bash-completion/completions/doas
```

`sudo="doas"` to point your Sudo commands towards Doas.

Man pages exist for both the `doas` command and the `doas.conf` file. The man page for `doas.conf` contains examples of advanced configurations. The developer has written on advanced concepts in his blog [8].

Conclusions

Doas is virtually unknown in Linux, but this lean Sudo alternative is worthy of attention. Doas is much simpler than the overly complex Sudo, and the plain-language configuration is unlikely to pose a problem even for newcomers. Only the absence of Bash completion in the default configuration and the need to re-enter the password for each command taint the otherwise good impression. However, both these

problems have easy workarounds.

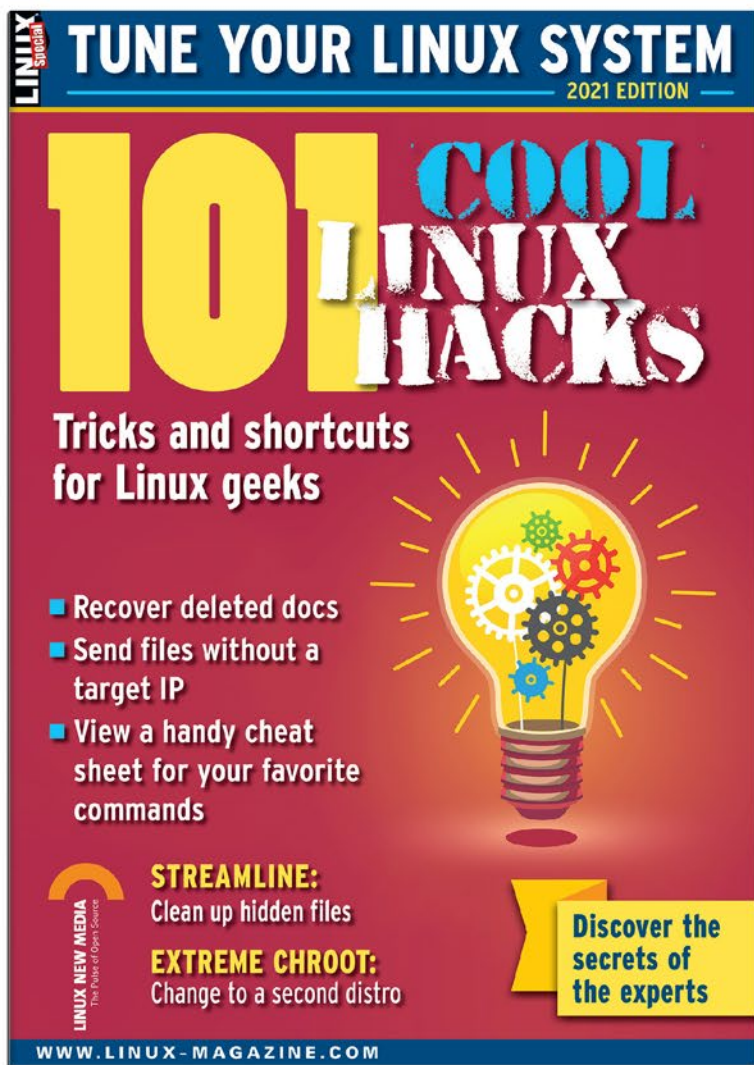
I have used Doas for about a year, and I have no cause for complaint. I especially like the simple configuration and the ability to work without a password on standalone systems. ■■■

Info

- [1] Role-based access control: https://en.wikipedia.org/wiki/Role-based_access_control
- [2] Mandatory access control: https://en.wikipedia.org/wiki/Mandatory_Access_Control
- [3] "10-year-old Sudo security bug lets Linux users gain root-level access" by Catalin Cimpanu, ZDNet: <https://www.zdnet.com/article/10-years-old-sudo-bug-lets-linux-users-gain-root-level-access/>
- [4] OpenDoas versions: <https://repology.org/project/opendoas/versions>
- [5] Doas: <https://github.com/slicer69/doas>
- [6] OpenDoas at Github: <https://github.com/Duncaen/OpenDoas>
- [7] PAM: <https://tldp.org/HOWTO/User-Authentication-HOWTO/x115.html>
- [8] Doas Mastery: <https://flak.tedunangst.com/post/doas-mastery>

SHOP THE SHOP
shop.linuxnewmedia.com

GET PRODUCTIVE WITH 101 LINUX HACKS



Improve your Linux skills with this cool collection of inspirational tricks and shortcuts for Linux geeks.

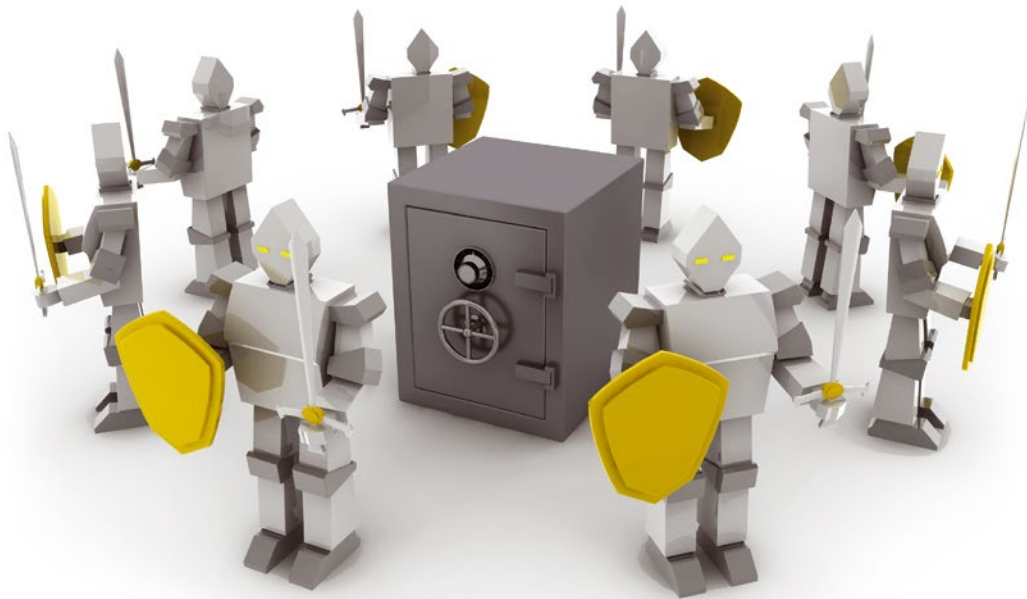
- Undelete lost files
- Cure the caps lock disease
- Run C one-liners in the shell
- Disable your webcam and mic
- And more!

ORDER ONLINE:
shop.linuxnewmedia.com/specials

Card Games

Improve communication security with GnuPG and the OpenPGP smartcard.

By Daniel Tibi



Gnu Privacy Guard [1] (sometimes called GnuPG or just GPG) has long served the open source community as a tool for encrypting email and other documents. GnuPG, which is based on the OpenPGP standard, uses the familiar asymmetric key exchange approach, with a public key to encrypt the message and a private key to decrypt it. The public key is shared with other users, and the private key is kept secret by the message receiver. As long as no one discovers the private key, only the message receiver will be able to read the message.

GnuPG is thus a powerful tool for ensuring confidential communication – as long as the private key stays private. But protecting a private key is not as easy as it sounds. If you store the private key on your computer, the key is only as safe as your computer is. Your system could fall victim to malware, cyber attack, or a nosy system administrator, and if so, the private key would be at risk. On the other hand, if you are a mobile worker and you have the need to use the private key from multiple locations, lugging it around on a thumb drive poses a whole different set of security issues.

One solution that is gaining momentum is to carry the private key on a smartcard. The ISO/IEC 7816-4 standard defines

a method for encoding cryptographic keys on a smartcard. The OpenPGP smartcard is an implementation of ISO/IEC 7816-4 for GnuPG and other OpenPGP-compatible encryption systems.

Smartcard and Card Reader

If you want to get started with the OpenPGP smartcard, the first step is to purchase the necessary hardware and software. You can obtain the smartcard (Figure 1), which is currently version 3.4, from several online outlets [2] for around EUR20. To work with the smartcard, you also need a card reader. Basically, any model that can handle contact smartcards is suitable. Some card readers have an integrated keyboard, which means you can enter the PIN of the card directly on the reader. If you enter the PIN directly on the reader, you avoid the possibility of it being sniffed when you type it at the keyboard.

The next step is to install the required software. Communication with a card reader relies on the CCID protocol [3]. The CCID protocol (short for “Chip Card Interface Device”) facilitates communication between a USB chip card reader and a PC via a uniform interface.

Download the current version of the CCID software, which was 1.4.33 at the time of this article, and unpack the archive in



Figure 1: Make GnuPG even more secure with a GnuPG smartcard.

the shell with the `tar xfvj ccid-1.4.33.tar.bz2` command. Then change to the directory where you unpacked the files and execute the commands from Listing 1.

Separate packages are also available for different distributions. On Ubuntu, you can alternatively use the command from Listing 2, Line 1 for the install. For some card readers, it happens that you need an additional driver. Some other card readers do not need any additional software beyond the standard driver and are ready to use immediately after plugging in [4].

The PC/SC standard provides the interface to the smart card reader. In the Linux world, PC/SC Lite [5] is a good tool for the interface. Download the program from the project page. Unpack the current version 1.9 in the shell by typing `tar xfvj pcsc-lite-1.9.0.tar.bz2`. Then change to the directory where you unpacked the files and execute the commands from Listing 1. Depending on the distribution, you can alternatively install the program via the package manager (Listing 2).

The PCSC Tools [6] let you test whether the card reader is installed correctly and whether it detects the smartcard correctly. You can download PCSC Tools from the project website. Unpack the archive in the shell by typing `tar xfvj pcsc-tools-1.5.7.tar.bz2`. In this case, too, you need to build the program from the archive you unpacked at the command line with the usual three steps (Listing 1). Alternatively, you can install the program from the package sources; on Ubuntu, Listing 2 Line 3 handles the installation.

Once you have connected a card reader to the system and installed the required software, run the `pcsc_scan` command in the shell to test if the device works correctly. Insert your smartcard

Current GPG

Make sure you install the latest version of GnuPG, but in any case, avoid everything from the deprecated 1.x branch. On older distributions, you may only have this outdated version in the package sources. You can check the installed version in the shell with the `gpg --version` command.

Version 1.x and version 2.x can be installed side by side. In this article, the `gpg` command is used when working with GnuPG. If you are not sure whether an outdated version is still installed in parallel, use the `gpg2` command instead. This command will ensure that you are always working with version 2.

into the card reader. If the device displays it as OpenPGP Card V3 (as shown in Figure 2), the card and reader are ready for use.

Before you start, make sure GnuPG [7] is in place. The current version is 2.3.1 (see the box entitled “Current GPG”). After downloading the current version, unpack the archive via the shell with the `tar xfvj gnupg-2.3.1.tar.bz2` command. Then change to the directory where you unzipped the files and run the three build steps (Listing 1). On Ubuntu, you could alternatively use the command from Listing 2, Line 4. You can also install GnuPG through your distro’s package manager.

If you prefer to use GnuPG via a graphical user interface, the GNU Pri-

vacy Assistant

is a good choice. You can install the program via your distribution’s package manager. On

Ubuntu, use the command from Listing 2, Line 5.

After the installation, you will

Listing 1: Compiling

```
./configure
make
sudo make install
```

Listing 2: Installing on Ubuntu

```
01 sudo apt install libccid
02 sudo apt install pcscd
03 sudo apt install pcsc-tools
04 sudo apt install gnupg2
05 sudo apt install gpa
```

```
daniel@tardis: ~
daniel@tardis:~$ pcsc_scan
Using reader plug'n play mechanism
Scanning present readers...
0: REINER SCT cyberJack RFID standard (0396704272) 00 00

Mon May 24 11:29:24 2021
Reader 0: REINER SCT cyberJack RFID standard (0396704272) 00 00
Event number: 1
Card state: Card inserted,
ATR: 3B DA 18 FF 81 B1 FE 75 1F 03 00 31 F5 73 C0 01 60 00 90 00 1C

ATR: 3B DA 18 FF 81 B1 FE 75 1F 03 00 31 F5 73 C0 01 60 00 90 00 1C
+ TS = 3B --> Direct Convention
+ TA = DA, Y(1): 1101, K: 10 (historical bytes)
TA(1) = 18 --> Fi=372, Di=12, 31 cycles/ETU
129032 bits/s at 4 MHz, fMax for Fi = 5 MHz => 161290 bits/s
TC(1) = FF --> Extra guard time: 255 (special value)
TD(1) = 81 --> Y(i+1) = 1000, Protocol T = 1
-----
TD(2) = B1 --> Y(i+1) = 1011, Protocol T = 1
-----
TA(3) = FE --> IFSC: 254
TB(3) = 75 --> Block Waiting Integer: 7 - Character Waiting Integer: 5
TD(3) = 1F --> Y(i+1) = 0001, Protocol T = 15 - Global interface bytes following
-----
+ TA(4) = 03 --> Clock stop: not supported - Class accepted by the card: (3G) A 5V B 3V
+ Historical bytes: 00 31 F5 73 C0 01 60 00 90 00
Category indicator byte: 00 (compact TLV data object)
Tag: 3, len: 1 (card service data byte)
Card service data byte: F5
- Application selection: by full DF name
- Application selection: by partial DF name
- BER-TLV data objects available in EF.DIR
- BER-TLV data objects available in EF.ATR
- EF.DIR and EF.ATR access services: by GET DATA command
- Card without MF
Tag: 7, len: 3 (card capabilities)
Selection methods: C0
- DF selection by full DF name
- DF selection by partial DF name
Data coding byte: 01
- Behaviour of write functions: one-time write
- Value 'FF' for the first byte of BER-TLV tag fields: invalid
- Data unit in quartets: 2
Command chaining, length fields and logical channels: 60
- Extended Lc and Le fields
- RFU (should not happen)
- Logical channel number assignment: No logical channel
- Maximum number of logical channels: 1
Mandatory status indicator (3 last bytes)
LCS (life card cycle): 00 (No information given)
SM: 9000 (Normal processing.)
+ TCK = 1C (correct checksum)

Possibly identified card (using /usr/share/pcsc/smartcard_list.txt):
3B DA 18 FF 81 B1 FE 75 1F 03 00 31 F5 73 C0 01 60 00 90 00 1C
OpenPGP Card V3
```

Figure 2: The `pcsc_scan` command detects a card reader of type “REINER SCT CyberJack RFID Standard” with an OpenPGP card inserted.

find a launcher in the application menus. Alternatively, you can call up the tool with the `gpa` command in a terminal. Then set up your smartcard via the *Window | Card Management* menu.

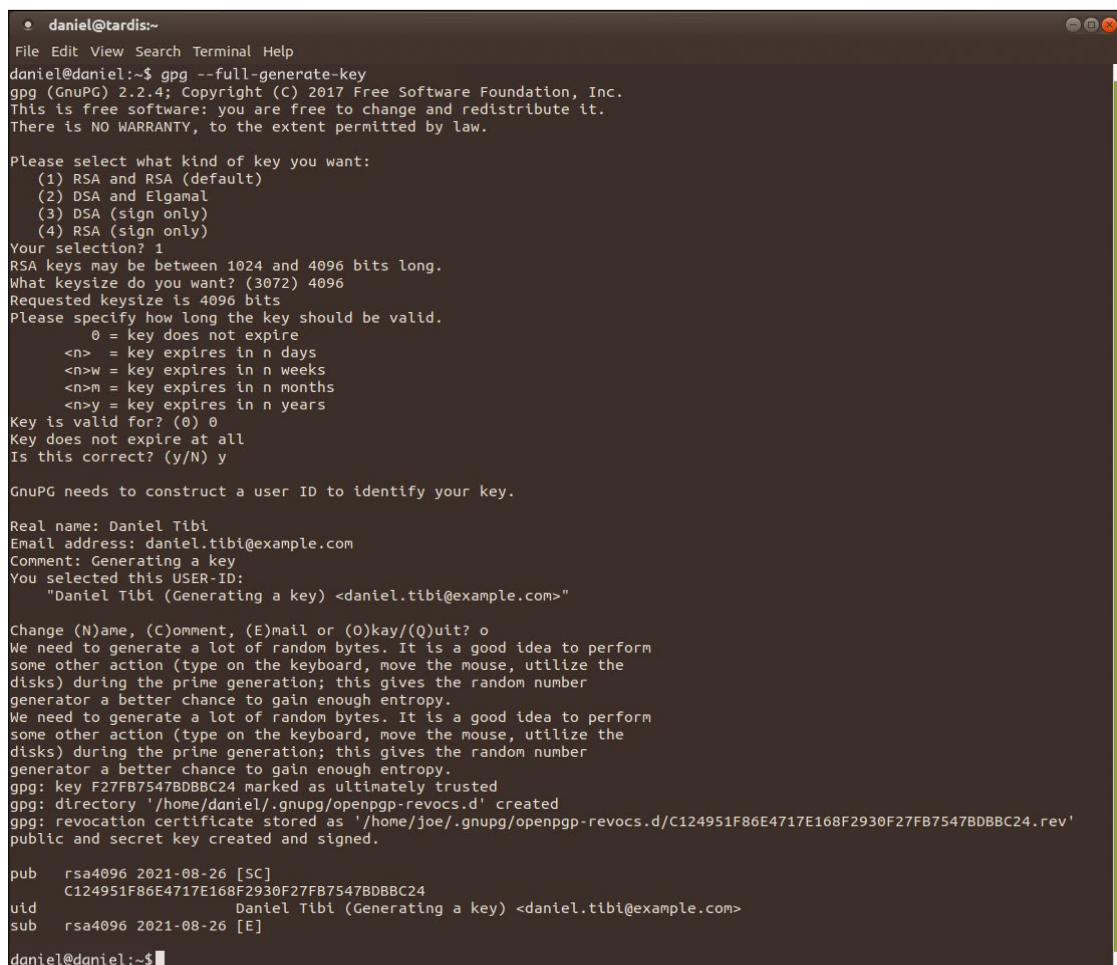
Set Up OpenPGP Smartcard

Once you have obtained the necessary hardware and installed the required software, the next step is to set up your new



```
daniel@tardis:~$ gpg --card-status
Reader .....: REINER SCT cyberJack RFID standard (0396704272) 00 00
Application ID ...: D276001240103040005000A85F000
Application type ..: OpenPGP
Version .....: 3.4
Manufacturer .....: ZeitControl
Serial number ....: 0000A85F
Name of cardholder: [not set]
Language prefs ...: de
Salutation .....:
URL of public key : [not set]
Login data .....: [not set]
Signature PIN ....: zwingend
Key attributes ...: rsa2048 rsa2048
Max. PIN lengths ..: 64 64 64
PIN retry counter ..: 3 0 3
Signature counter ..: 0
KDF setting .....: off
Signature key ....: [none]
Encryption key....: [none]
Authentication key: [none]
General key info...: [none]
daniel@tardis:~$
```

Figure 3: The `[none]` in the last four lines of the output of `gpg --card-status` shows that the card does not yet contain any keys.



```
daniel@tardis:~$ gpg --full-generate-key
gpg (GnuPG) 2.2.4; Copyright (C) 2017 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Please select what kind of key you want:
  (1) RSA and RSA (default)
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (3072) 4096
Requested keysize is 4096 bits
Please specify how long the key should be valid.
  0 = key does not expire
  <n> = key expires in n days
  <nw> = key expires in n weeks
  <nm> = key expires in n months
  <ny> = key expires in n years
Key is valid for? (0) 0
Key does not expire at all
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

Real name: Daniel Tibi
Email address: daniel.tibi@example.com
Comment: Generating a key
You selected this USER-ID:
  "Daniel Tibi (Generating a key) <daniel.tibi@example.com>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: key F27FB75478DBBC24 marked as ultimately trusted
gpg: directory '/home/daniel/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/joe/.gnupg/openpgp-revocs.d/C124951F86E4717E168F2930F27FB75478DBBC24.rev'
public and secret key created and signed.

pub   rsa4096 2021-08-26 [SC]
      C124951F86E4717E168F2930F27FB75478DBBC24
uid           Daniel Tibi (Generating a key) <daniel.tibi@example.com>
sub   rsa4096 2021-08-26 [E]

daniel@daniel:~$
```

Figure 4: You can create a new key either directly on the OpenPGP smart card or locally on your computer, as shown in the figure, and then move it to the card.

OpenPGP smartcard. Insert the card into the card reader and execute the `gpg --card-status` command. The output should look like Figure 3. The empty card now needs to be populated.

You can edit the data stored on the card with `gpg --card-edit`. The output from the command first shows you the current contents of the card. After that, you will see the `gpg/> card>` prompt, which is waiting for commands to edit the OpenPGP smart card. Type `admin` to start editing the card. Then type `help` to get an overview of the possible commands.

The first thing to do is to change the PIN and the admin PIN of the card by typing `passwd` at the GPG prompt. If you choose `1 – change PIN` in the selection menu, the software will first ask for the old PIN for the card. The preset default PIN is 123456. You can then type a new PIN. If you have chosen a card reader with an integrated keyboard, input the PIN via the keys on the reader.

If you enter the PIN incorrectly three times in a row when using the card, the smartcard is blocked. In this case, you can unlock the card with the `unlock` command in the *Admin* menu. You will need the Admin PIN to unblock the PIN; you can set up the Admin PIN using the same approach as the normal PIN.

To do so, select `3 – change Admin PIN` from the PIN menu. The preset default Admin PIN is 12345678. Entering the Admin PIN incorrectly three times in a row when using the card will permanently lock the card. Once you have successfully changed both PINs, exit the PIN menu again via the menu item `Q – quit`.

You can individualize your OpenPGP smartcard by saving your name with the `name` command and your salutation with the `salutation` command at the GPG prompt. The `lang` command lets you set a different language.

On the Card

After setting up the OpenPGP smartcard, it is time to create a new key. You can generate the key directly on the OpenPGP smartcard with the `generate` command. The first thing you need to decide is whether you want to store a backup copy of the new key pair outside the card. This is usually a good idea because other-

wise the key pair will be irretrievably lost if the card stops working. Confirm the prompt by pressing *Y*. After that, the software will ask for the PIN you just set.

Next, decide how long the new key is valid. If you never want the key to expire, type *0*. Then enter your name and e-mail address and set a password for the key. To compute the key, the system uses random data that it generates based on the activity on the computer. It is a good idea to move the mouse and press various keys while the key is being generated (Figure 4). In the test, the system needed only a few seconds to create the key. Your new key is then ready for use.

On the PC

If you generate the key directly on the smartcard, you are limited to a key length of 2048 bits. If you want to create a more secure key with a length of up to 4096 bits, the card reader must be able to handle the Extended APDU format, which is not the case with all devices. If your card reader does not support this feature, you can create your GnuPG key pair with up to 4096 bits on your PC and then move the private key to your card.

On your PC, you can create a new key with the `gpg --full-generate-key` command. First, choose what kind of key you want. The OpenPGP smartcard only handles RSA, so only the default *(1) RSA and RSA* option is eligible. After that, you decide on a key length between 1024 and 4096 bits. Finally, you will be asked for the key's expiration time, your name, your email address, and a password for the key. The new key is then ready.

Now you need to move the private key you just created to the smartcard. The public key remains on your PC. You can also move a private key you created separately to the smartcard.

Moving the private key to the card will delete it from your computer, so it is a good idea to make a backup copy. Use the command from Listing 3, Line 1 to create a backup. Modify the email address accordingly. The command stores a copy of your private key in the `myseckey.asc` file on your desktop.

Move the file with the private key to a safe place, such as a USB stick. To be prepared for any eventuality, make a backup copy of the public key right away using the command from Listing 3, Line 2 – again using your own email address.

On the desktop, you will find the `mypubkey.asc` file with your public key, which you can save on a USB stick. But leave it on the desktop for the time being because you will need it to configure the email program.

Copying a Private Key

Moving the private key to the OpenPGP smartcard is more complicated than it sounds because the card is not happy with just one key and instead expects three subkeys: one for signing, one for encryption and decryption, and one for authentication. The key you just created only handles signing and encryption/decryption, so you need to add a subkey for authentication. Use the command from Listing 3, Line 3, to add an authentication key – again using your own email address.

Listing 3: Exporting a Secret Key

```
01 gpg -a --export-secret-key user@example.com >> ~/Desktop/myseckey.asc
02 gpg -a --export user@example.com >> ~/desk/mypubkey.asc
03 gpg --expert --edit-key user@example.com
```

A list of your keys appears. At first, you will see only keys for signing (usage: *S*) and for encryption and decryption (usage: *E*). You can add the missing subkey for authentication (Use: *A*) at the GPG prompt with the `addkey` command. The system will ask you for the type of key you want to create. From the drop-down menu, now choose *(8) RSA (usage can be set by yourself)*, then *(A) Toggle authentication usability*, and finally *(Q) Quit*.

The program now creates the new subkey. You'll need to enter the length of the key and the expiration date. `4096` is a good choice for a long and secure key. For the expiration date, enter `0` if you don't want the key to expire. Then move the private master key to the OpenPGP smartcard using the `keytocard` command at the GPG prompt. When asked which key you want to move to the card, choose *(1) Signature key*.

Once you have moved the master key, the next step is to move the encryption, decryption, and authentication subkeys to the smartcard. At the GPG prompt, type `key 1` to select the subkey for encryption and decryption. The output that follows will mark the selected key with an asterisk (look for something like `ssb*rsa4096/key_ID`). At the end of the line, you'll see the entry `Use: E`, meaning use for encryption and decryption. The `keytocard` command copies the selected key. As the storage location, specify *(2) encryption key*. You can select the key again later with `key 1`.

Now repeat this process for the authentication key. You can select the key with `key 2`. Again, you will see an asterisk to the right of the key in the key list; this time, it should be followed by `use: A`. You can move the selected key to the card with `keytocard`. As the storage location, use *(3) Authentication key*.

You have now successfully moved the keys to the card. Don't forget to save your changes to the smartcard with `Save`. Then enter `gpg --card-status` in the shell; you should see a `Signature key`, an `Encryption key`, and an `Authentication key` on the card.

Thunderbird Configuration

The OpenPGP smartcard is now ready to use. The steps for using the smartcard vary depending on the application. I'll describe how to use it with the Thunderbird email client [8].

Version 78 and newer of Thunderbird no longer access GnuPG via the Enigmail plugin to manage PGP keys but, instead, manage the keys internally. This means that Thunderbird currently no longer has a GUI for working with the OpenPGP smartcard as in previous versions, and you'll need to do configuration work first.

The first thing to do is to install the Enigmail [9] plugin. Select the *Add-ons* menu item in Thunderbird. You will now see a list of installed extensions. Search for *Enigmail* in *Find more add-ons*. You can install the add-on via the *Add to Thunderbird* button.

Then configure Thunderbird so that the program does not use its internal key management but uses GnuPG instead. Go to the *Preferences | General* menu and click the *Config Editor* button at the bottom. Look for the `mail.openpgp.allow_external_gnupg` setting and set the value to `true`.

Finally, import your private and public GnuPG keys into Thunderbird. Since the private key is on the OpenPGP smartcard and the corresponding public key is stored locally, this takes two steps. You make all the necessary settings in the *Ac-*

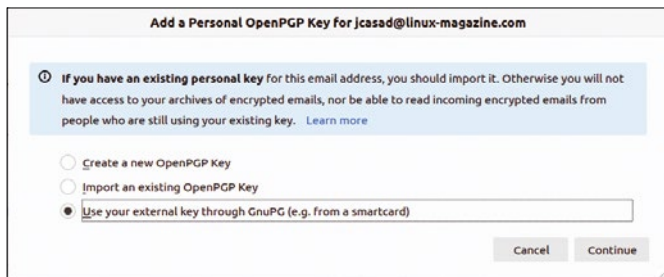


Figure 5: You'll need to tell Thunderbird you're using an external GnuPG private key on a smartcard.

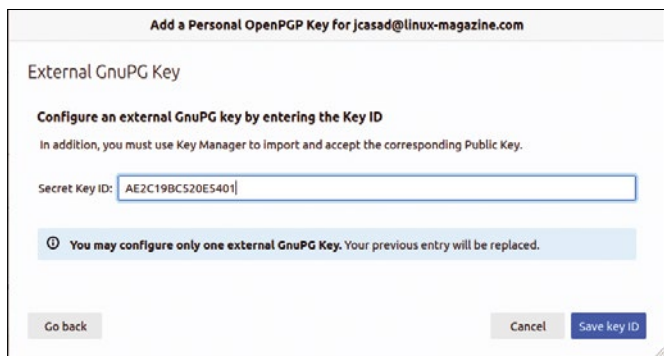


Figure 6: Thunderbird asks for the ID of the private key.

count Settings | End-to-end encryption menu. Now to set up your private key in Thunderbird, click *Add Key* and choose the option *Use your external key through GnuPG* (for example, from a smartcard in the dialog (Figure 5).

Thunderbird will ask you for the ID of your private key. To discover the ID, insert your smartcard into the card reader and run the `gpg --card-status` command. The output should look similar to the output in Figure 4. In the lower third of the output, you will see a line that gives the encryption method and key length followed by the ID (for example, `sec> rsa2048/AE2C19BC520E5401`). This line in this example tells you that the key is an RSA private key with a length of 2048 bits. The value after the slash is the ID of the key. You need to copy the value `AE2C19BC520E5401` into the dialog box in Thunderbird and confirm by pressing *Save Key ID* (Figure 6). Now Thunderbird shows you that it will use an external GnuPG key.

You can add your public key by clicking the *Manage OpenPGP Key* button. In the dialog that follows, select the *File | Import public key from file* menu item. Your public key is probably still on the desktop under the name `mypubkey.asc`. You can use this same procedure to set up the public keys of your email communication partners.

Encrypted Email

Once you complete the somewhat complex configuration steps, sending encrypted email is far easier. Click the *Write* button in the main window in Thunderbird as usual. To encrypt your email, select *Security | Require Encryption* from the menu.

You can also sign your email, either in addition to encryption or without encryption. To sign a message, opt for *Security | Digitally Sign this Message*. To send your email, place your OpenPGP smart-

card in the card reader and click *Send*. Thunderbird will now ask you for the PIN for your card and the password for your key. Provide both, and the email is on its way to the recipient.

Encrypting Files

The most common application for GnuPG is probably encrypting and signing emails, but GnuPG also lets you protect files against prying eyes, both asymmetrically and symmetrically. That is, files that you want to pass on to another person, as well as files that you want to encrypt in storage on your own PC.

If you want to pass a file to another person, use the command from Listing 4, Line 1. This command tells GnuPG to asymmetrically encrypt the file named at the end of the command so that only you and the recipient with the email address `user@example.com` can decrypt the file using their own GnuPG key.

If you want to encrypt the file symmetrically instead, i.e., not pass it on, use the command from Listing 4, Line 2. In this case, GnuPG will ask you for a password to encrypt the file. In both cases, you unlock the encrypted `file.gpg` file with the command from Listing 4, Line 3.

Conclusions

GnuPG boosts the security of email traffic and encrypts important files. You can use GnuPG even more securely with an OpenPGP smartcard. The one-off setup is a little more complicated, but once everything is configured, the encryption process is quite effortless and will be easy to build into your routine.

If you need a digital password safe and a generator for one-time passwords in addition to the functions described in this article, it is worth taking a look at the Nitrokey Pro 2 (p. 24 of this issue), which also comes with an integrated OpenPGP card in addition to these functions. However, keep in mind that these additional functions mean the Nitrokey is considerably more expensive. ■■■

Info

- [1] Gnu Privacy Guard: <https://gnupg.org/>
- [2] Open PGP SmartCard V3.4 at Cryptoshop: <https://en.cryptoshop.com/products/smartcards/cryptographic-smart-cards/open-gpg-smartcard-v2.html>
- [3] CCID: <https://ccid.apdu.fr>
- [4] Overview of card readers supported or not supported by the CCID standard driver: <https://ccid.apdu.fr/ccid/section.html>
- [5] PC/SC Lite: <https://pcsclite.apdu.fr>
- [6] PCSC Tools: <http://ludovic.rousseau.free.fr/software/pcsc-tools/>
- [7] GnuPG: <https://www.gnupg.org/download/index.html>
- [8] Thunderbird: <https://www.thunderbird.net>
- [9] Enigmail: <https://addons.thunderbird.net/en-US/thunderbird/addon/enigmail/>

Listing 4: Passing a File

```
01 gpg --output file.gpg --encrypt --recipient user@example.com file_name
02 gpg --output file.gpg --symmetric file_name
03 gpg --output file_name --decrypt file.gpg
```

Turn your ideas into reality!

This is not your ordinary computer magazine! MakerSpace is a new special issue that focuses on technology that you can use to build your own stuff.

If you're interested in electronics but haven't had the time or the skills (yet), studying these maker projects might be the final kick to get you started.

This special issue will help you dive into:

- Raspberry Pi
- Arduino
- Retro Gaming
- FPGA
- and much more!

BONUS
Complete Raspberry Pi Geek Archive DVD
free with the print edition!



ORDER ONLINE:
shop.linuxnewmedia.com/specials

Securely encrypt passwords with Nitrokey Pro 2

Locked

The Nitrokey Pro 2 is a small device that covers a wide range of cryptographic functions. *By Daniel Tibi*

The small and inconspicuous Nitrokey Pro 2 is a digital door opener: You can use the Nitrokey's password safe to securely lock up your access credentials, and you can generate one-time passwords for more secure logins to online services. An integrated OpenPGP card lets you encrypt and sign emails. (See the article on the OpenPGP smartcard starting on p. 18 in this issue.)

You can purchase the Nitrokey Pro 2 for around EUR50 via the manufacturer's online shop [1] (Figure 1). The online shop is also where you will find the Nitrokey Storage 2, which provides the same functions as the Nitrokey Pro 2 but also includes encrypted storage capacity ranging from 16 to 64GB. Depending on how much storage you need, the Nitrokey Storage 2 costs somewhere between EUR109 and EUR199.

Configuration

To set up the Nitrokey, you also need the Nitrokey App [2], which is available for various operating systems. For Linux, the manufacturer offers packages for various distributions on its website, as well as the source code, which you can compile yourself.



Figure 1: Use the Nitrokey Pro 2 like an OpenPGP smartcard. You can also generate one-time passwords and store access credentials. If you opt for the Nitrokey Storage 2, you will have access to encrypted storage of up to 64GB. © Nitrokey GmbH

Once you have purchased the Nitrokey and installed the app on your computer, plug the stick into the computer and start the software with the `nitrokey-app` command in the shell or by clicking on the icon in the application menus.

Access to the Nitrokey is protected by a PIN. The PIN keeps your data safe, even if you lose the stick. To change the settings, you first need to enter the Admin PIN (see the "Start PIN" box). Before you start working, the first thing to do is to set your own PIN and Admin PIN. Select *Menu | Configure | Change User PIN* and *Change Admin PIN* in the Nitrokey App (Figure 2).

You can now use the password safe to store important access credentials. Unlock the safe in the app via *Menu | Unlock Password Safe* and enter the PIN. Then click on the *Password Safe* tab, where you can store up to 16 passwords and credentials. Select a slot on the list, assign a name, and enter the login information and password.

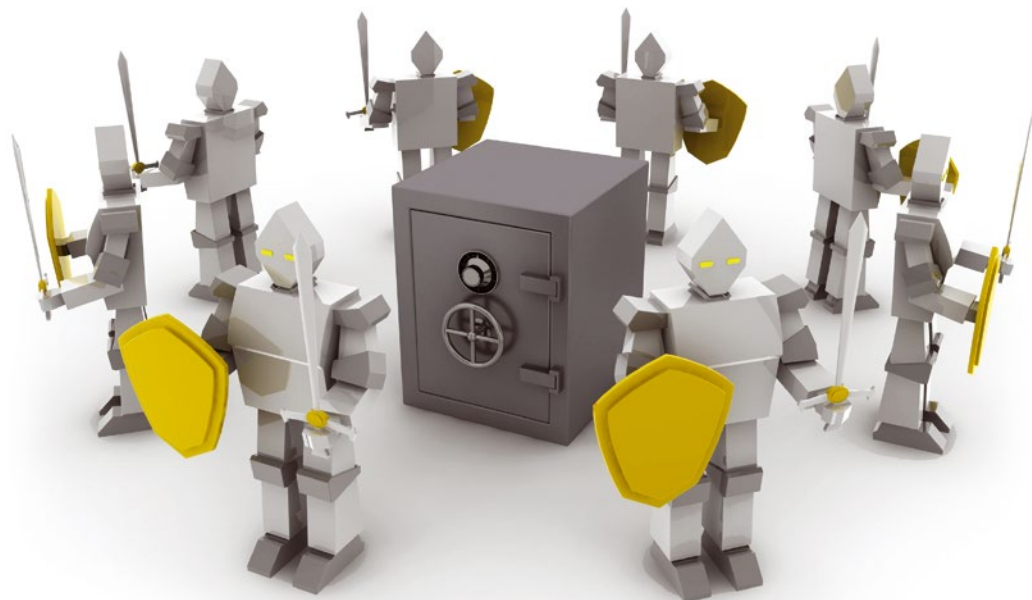
If you are just logging in to an online service, the app will help you choose a new password after clicking *Generate random password*. The storage space on the Nitrokey is limited, so you will see the maximum number of characters to the right of each field. Once all the data is entered, don't forget to press *Save*.

Unlocking

Once you have captured the passwords, you can use them anytime you need them. Provided that the Nitrokey is plugged in and the password safe is unlocked, you will find a list of passwords stored on your Nitrokey in *Menu | Passwords*. After you click on the desired entry, the program copies the appropriate password to the clipboard of the desktop environment. You can then paste it onto the login screen.

Note that this is a weak point: The password is sent in plain text to the clipboard, where it would theoretically be possible for an attacker to intercept it. Caution is therefore advisable when working on a computer that you do not own.

To prevent your password from staying in the clipboard indefinitely, use the *Settings* tab in the app to set the time at which the password is deleted from the clipboard. The default is 60 seconds, but 30 seconds is usually long enough. After that, the password disappears from the clipboard. This feature can be an issue if you use a clipboard manager. In



the test, the copied passwords remained in the clipboard manager's history.

One-Time Passwords

To improve login security, online services often use one-time passwords that are sent to the user by text. For many online services, you can simply generate a one-time password using the Nitrokey App so that you do not have to rely on the provider's app for each user account. Look for instructions at the Nitrokey website [3].

The basic principle is the same for all services: enable two-factor authentication for the service and enter the secret key, which will actually be used to generate one-time passwords via the provider's own app, in the Nitrokey App.

For example, log in to your Google account via <https://myaccount.google.com>. Then click *Security* on the left and, under *Sign in to Google*, opt for *Confirm in two steps*. When you get there, first set up your smartphone. After that, the system will show you different ways to use your smartphone for two-factor authentication. By default, Google sends you one-time passwords as text messages.

Start PIN

The Nitrokey's start PIN is always 123456, and the startup Admin PIN is always 12345678. You will want to change the PIN immediately before using the Nitrokey for the first time. To change the PIN, select *Menu | Configure* in the Nitrokey App.

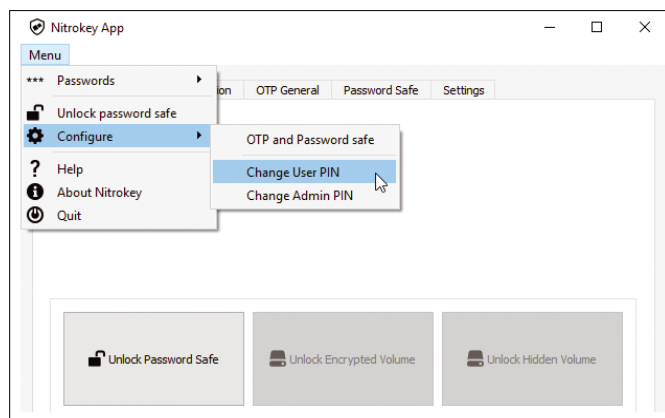


Figure 2: Set up a PIN to keep your passwords safe.

Select *Authenticator App* from the list of options and click *Setup*. You don't really want to use the Authenticator App, but that's the only way Google will hand over the private key you're after. Now a barcode appears on the page, which you would scan with the Authenticator App if you were using it. But don't do that; instead click on *You can't scan it*.

Google will then show you the private key. Switch to the Nitrokey App and call up the *Disposable passwords entries* tab. Now, assign a name for the entry, in this case, *Google*. Enter the private key in the *Secret* field and click *Save*. This step completes the setup in the Nitrokey App. Switch back to the Google account because there the configuration goes a little further.

In the dialog from which you just copied the private key, click *Next*. Google will ask you for a six-digit code, which will be shown to you by the Authenticator App. You can now directly test whether everything is set up correctly in the Nitrokey App.

Launch the Nitrokey App and click *Menu | Passwords | Google*. The Nitrokey App will then generate a one-time password and copy it to the clipboard. From there, paste it into the dialog box in your Google account. This completes the setup of your Google account, and from now on, you can use the Nitrokey App to generate one-time passwords to log in.

Conclusions

The Nitrokey Pro 2 is a useful helper. The matching app is easy to install and use. You can put your credentials in a digital safe and have a tool at hand to generate one-time passwords for more secure logins to online services.

Because an OpenPGP card is integrated, you won't even need a card reader if you choose the Nitrokey. The Nitrokey Storage 2 model also comes with encrypted mass storage, but you will have to pay a little extra. ■■■

Info

- [1] Nitrokey Pro 2 at Nitrokey Shop: <https://shop.nitrokey.com/shop/product/nk-pro-2-nitrokey-pro-2-3>
- [2] Nitrokey App: <https://www.nitrokey.com/download>
- [3] Two-factor authentication with one-time passwords (OTP): <https://docs.nitrokey.com/pro/otp.html>

Hardened Choice

SiriKali encrypts files and directories with just a few mouse clicks, without the inefficiency of fixed-size containers. *By Erik Bärwaldt*

Many Linux users are wary of encrypting their data – primarily because most of the available tools don't offer a graphical user interface, and also – but less often – because of a perceived lack of flexibility in handling encrypted data files. But

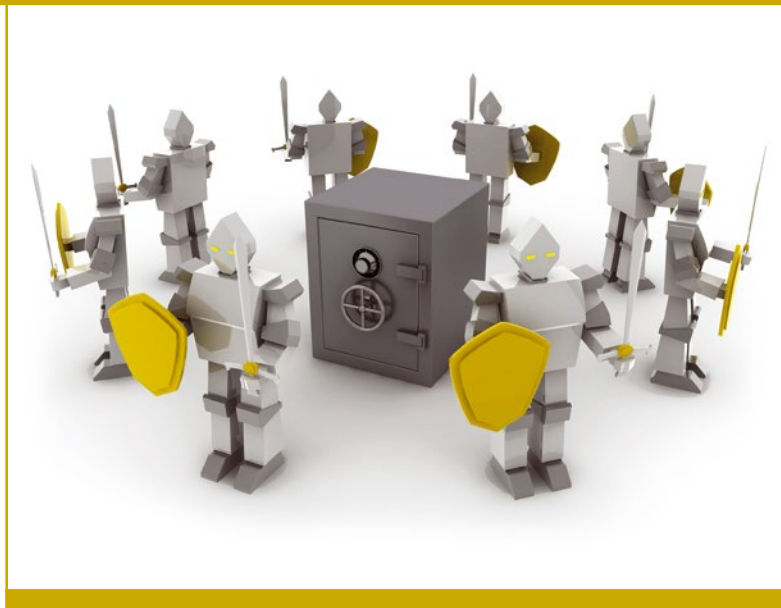
in the modern era, when high-capacity USB media are easy to find, encryption is becoming more important. Tools such as TrueCrypt or its successor VeraCrypt have a graphical user interface but create containers of a fixed size, without the flexibility to deal with growing volumes. If you store only a few files, you are wasting storage space with a big container. On the other hand, if you create a container that is too small, running out of space will mean a time-consuming overhaul.

SiriKali [1] is an encryption tool that avoids fixed sizes for containers. By encrypting at the directory and file level, you only use as much storage space as the data actually takes up. SiriKali relies on various encryption back ends, with support for fscrypt, SecureFS, eCryptFS, CryFS, EncFS, gocryptfs, and SSHFS. (If you're considering EncFS, keep in mind that security vulnerabilities were discovered in an audit in 2014.)

In SiriKali, you deploy encrypted filesystems in user space – with the help of the FUSE kernel module, which means that you can work with the tools without needing admin privileges. SiriKali recognizes the back ends installed in the system and lets you use them without having to enter any parameters.

Installation

SiriKali is based on the Qt libraries and is included in the repositories of several popular distributions. You can use your



default package manager for the install. If you don't have a back end in place, you'll need to drag in one of the back ends to handle the actual data encryption work.

You could also integrate your own repository for the application with a standard package management system.

The developers provide detailed documentation on the project's GitHub [2]. The setup creates a launcher in the desktop menu.

Operation

When you launch SiriKali for the first time, it hides itself away in the system tray on the desktop, featuring a blue icon with a stylized padlock by default. The icon gives you direct access to the tool's most important options without having to take a detour via the menus.

After the first launch, a window opens up with a large vacant space for displaying the files. At the bottom is a buttonbar, but there is no extensive menubar. The software lets you change the locale under *Menu | Settings | Select Language*; the current version 1.4.8 supports English, French, and Russian.

First, use *Create Volume* to create a volume, to which you will later back up the data you wish to encrypt. In the selection box, first choose the target filesystem. Only the options installed in the operating system are available; all other options are not active (Figure 1).

After selecting the filesystem, the software opens a small window where you can enter the required data. In the *Volume Name* box, type the name of the volume; below that you need

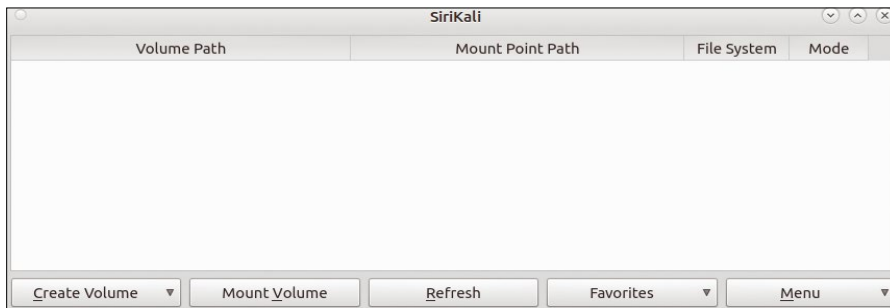


Figure 1: SiriKali's main window appears sparse at first glance, but beginners will have no trouble learning the intuitive interface.

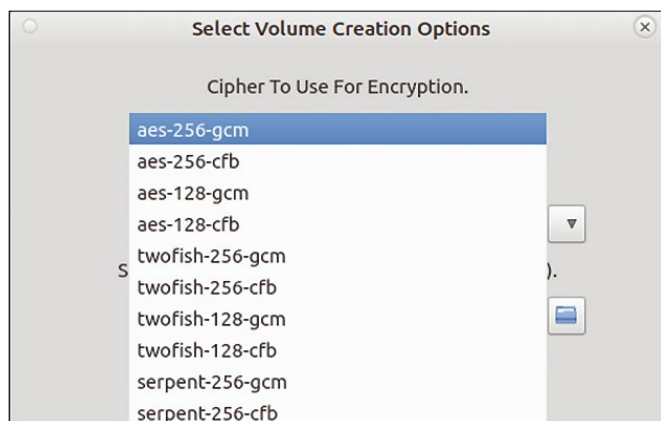


Figure 2: SiriKali lets you select the algorithm for some encryption methods.

to define the path for the digital container. By default, the tool uses your home directory. A click on the folder icon to the right lets you enter a different directory if necessary.

Below the box with the path, you need to choose the authentication method. Clicking on the small triangle on the right opens a pop-up menu; by default, the program uses password input. But keys, which you can specify manually, or existing key files are more elegant. Alternatively, you can include

Gnome and KDE wallets in SiriKali for authentication. The selection menu also supports the use of a Yubikey token.

Depending on the chosen method, enter the desired password in the next field. In the *Options* field, you can also specify whether SiriKali will also encrypt the file names.

Depending on the selection, the program prompts you in another dialog for the algorithm you will use to encrypt the data. You will find a list at the top of the window that shows the available algorithms (Figure 2).

After completing the settings, close the window by pressing *OK*, and then create the volume in the *Create* window. SiriKali mounts the volume like a conventional drive. You can create multiple drives with different back ends for encryption, and SiriKali lists them in a table in the main window, along with the back end and path (Figure 3).

Shop the Shop

shop.linuxnewmedia.com

Missed an issue?

You're in luck.

Most back issues are still available. Order now before they're gone!

shop.linuxnewmedia.com

GET IT NOW!
SAVE TIME ON DELIVERY WITH OUR ALTERNATIVE PDF EDITIONS



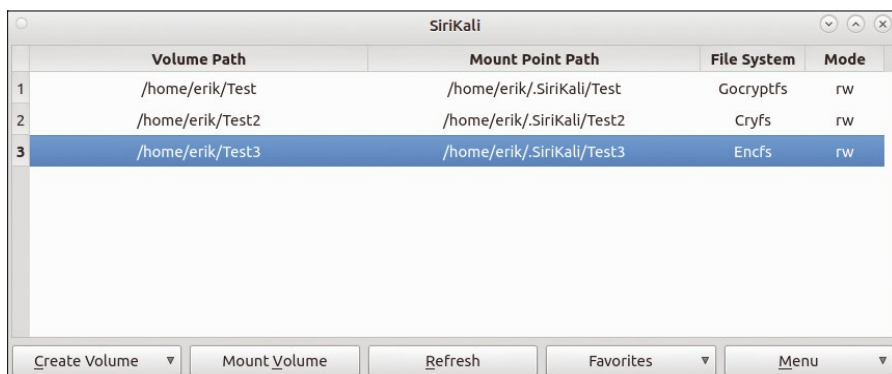


Figure 3: SiriKali supports simultaneous work with multiple containers.

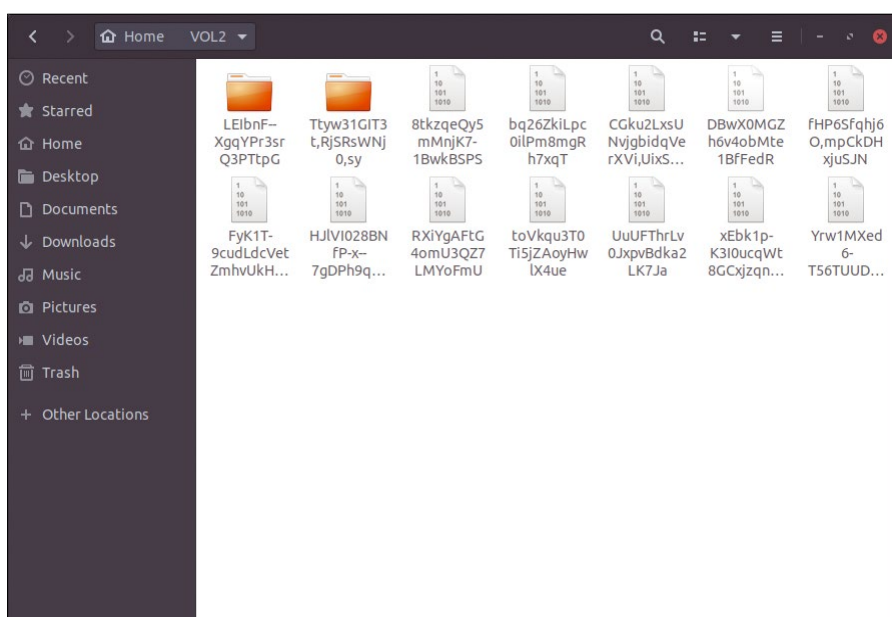


Figure 4: If so desired SiriKali will encrypt the file names and prevent any conclusions about the content.

Precautions

To move files or directories to one of the encrypted volumes, click on the entry in the main window and select *Open Folder* from the context menu. The software opens the volume in the file manager, where it behaves like a normal, unencrypted directory. You can drag and drop the data into the encrypted volume from another open instance of the file manager in the usual way.

The data is encrypted in real time. After finishing the transfers, it makes sense to unmount the encrypted drive by clicking again on the entry in the main SiriKali window and selecting *Unmount* from the context menu. The entry for the drive will now disappear from the table.

Open Up!

When opening a new session, SiriKali starts with an empty list. To mount one of the encrypted drives, click the *Mount Volume* button in the main window. The application will now open a file manager view where you select the desired drive.

Depending on the authentication method, a modal dialog appears where you can enter the password to open the volume. Then the file manager launches again with the decrypted contents of the drive, and the matching entry appears in the list in the main SiriKali window.

You can work with the contents in an unlocked state as you would with any conventional, unencrypted directory. If you call the volume directly from the file manager without the detour via SiriKali, you will see numerous files, all of them encrypted. If you also encrypt the file names, the displayed names will not give you any clues as to the files' contents (Figure 4).

Select *Menu | Unmount all* in the program window to unmount all the existing drives with a single click when you are done with your work. If you want to terminate the session at the same time, select *Unmount All And Quit*.

In the System Tray

SiriKali creates a launcher in the system tray of your working environment when you open a session. When you exit the software using the *Close* button in the titlebar, only the window disappears; the software continues to run in the background.

Pressing the blue button in the system tray opens a menu where you can open or close the program window by clicking on *Show/Hide*. In this way, if you tempo-

rarily stop working with encrypted volumes, you do not need to close the application completely every time.

Conclusions

SiriKali makes encrypting files and directories far easier for users without root privileges. Integration with various desktop environments lets you handle encrypted data just as easily as working with conventional drives. The program encrypts the contents in near real time, so you won't experience any significant latency.

SiriKali is a very useful choice for users who want to encrypt files and directories quickly and efficiently, without having to delve into the depths of a more complex application. ■■■

Info

- [1] SiriKali: <https://mhogomchungu.github.io/sirikali/>
- [2] Documentation: https://software.opensuse.org//download.html?project=home%3Aobs_mhogomchungu&package=sirikali

CLEAR OFF YOUR BOOKSHELF WITH DIGITAL ARCHIVES

Complete your collection of *Linux Magazine* and *ADMIN Network & Security* with our Digital Archive Bundles.

You get a full year of issues in PDF format to access at any time from any device.



Lead Image © enki, i23RF.com

<https://bit.ly/archive-bundle>

2020
Archives
Available
Now!



Bringing Gentoo Linux to the masses

A FRIENDLY DERIVATIVE

With its point-and-click installation, Redcore aims “to be to Gentoo what Manjaro is to Arch Linux.”

By Bruce Byfield

Redcore Linux [1] (Figure 1) is a Gentoo [2] derivative designed to give users some of the optimization of Gentoo with an easier install. It is a successor to Kogaion Linux, which in later releases was also based on Gentoo. However, after five years of development, the RogentOS Development Group, Kogaion’s owner, discontinued it in November 2016. In response, Ghiunhan Mamut, a member of RogentOS, created Redcore Linux. With Redcore featured in this month’s download DVD, Mamut agreed to answer our questions.

Mamut discovered free software around the turn of the millennium. Unusually, his high school ran Mandrake Linux, one of the earliest user-friendly distributions, rather than Windows. Later, when he bought his first home computer, he thought it natural to install Linux himself. “Back then,” he recalls, “the free aspect did not matter much. But as time passed, I started learning about the system, and I was exposed to the free software movement. Naturally, I started tinkering with it – not programming but trying different

configurations. I broke the system on purpose more times than I can remember, then tried to fix it. I soon found myself helping others fix and configure their systems.” Mamut went on to help found the first Linux User Group in his city and made his first contribution: a patch for the chat client Pidgin. “It felt good,” Mamut says, and he went on to create an installer for an online TV solution and to compile various packages as he hopped between distros.

Then, 10 years ago, Mamut discovered Gentoo Linux. “It was like nothing I ever used before,” Mamut recalls. “It took a very long time to install – and I mean a very, very long time, about two weeks, since computers weren’t that powerful back then. However, at the end of those weeks, I had a system which I built from scratch. I kept tuning it to my liking to eliminate stuff I don’t use, and I guess I really liked the result, because after that I stopped distro-hopping.” Mamut made

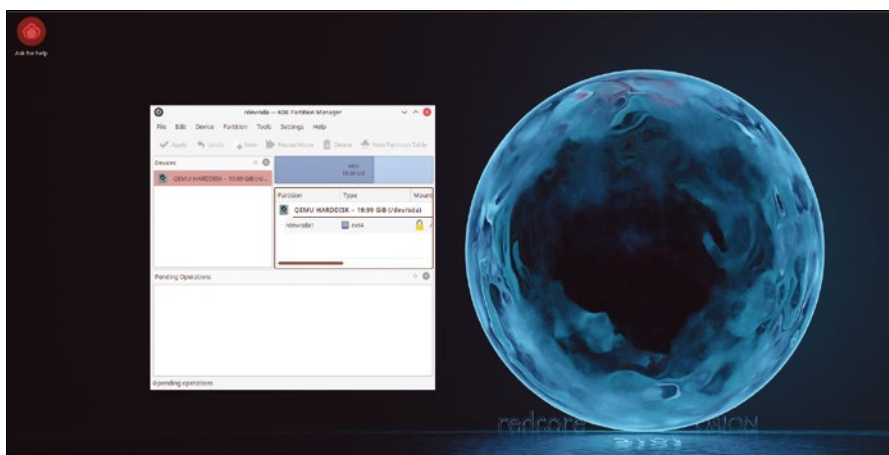


Figure 1: Redcore defaults to KDE, but other desktops are also available.

Photo by Count Chris on Unsplash

an exception for Kogaion Linux, attracted by its efforts to make Gentoo easier, but “when I tried it, it felt very rough and unpolished,” he says. “I decided to change that and I got involved in Kogaion’s development. Gradually, we rebased the distribution on Gentoo Linux exclusively and it became a lot better, easy to install and quick to run. However, after just two successful releases, real life commitments got in the way and the team behind it disbanded. Yet I wasn’t ready to give up, so I forked the codebase and Redcore Linux was born.”

Redcore Today

Today, Redcore continues to rely on Gentoo as its upstream source. According to Mamut, 2129 out of 2228 packages in Redcore come unmodified from Gentoo’s stable and testing branches. The rest of the packages are Redcore specific and include different kernels, the Sisyphus package manager (Figure 2), artwork, configurations files, and modified Gentoo packages. Because of the dependency on Gentoo, Redcore syncs with Gentoo every three days. Three days may seem a lot for a rolling release, but Mamut notes, “that’s a big improvement, since Redcore Linux used to sync changes on a weekly basis and sometimes longer, lagging behind. We aim at bringing that number down to once per day, but we’ll see how life commitments get in the way.”

Because Redcore is based mostly on Gentoo’s testing, packages are further organized into three repositories: master (stable), next (testing), and edge (development). New packages from Gentoo are synced into edge and, if necessary, recompiled against newer libraries. Generally, the process goes smoothly, but occasionally a bug has to be patched, either from fixes in Gentoo or by Redcore. “Sometimes,” Mamut adds, “we have to file [a] bug upstream with our proposed fix so every Linux distribution can benefit.”

Redcore has tried in the past to gather information. One release was downloaded 2,000 times in eight hours from the distro’s server alone. However, the distro has not gathered statistics consistently. “We only know for certain that like 10 people use it,” Mamut jokes, “and that only because they join our IRC channel and chat with us.” However, Redcore developers do not view their target audience as beginners, although they are aware of at least one newcomer to Linux who successfully installed their work. Rather, the goal is “to be to Gentoo what Manjaro is to Arch Linux. It targets people with Linux experience but with no Gentoo Linux experience,” Mamut explains. “Gentoo Linux is fantastic, but its installation can take a very long time and some users may find it intimidating. It really isn’t, but a point-and-click installation

is more appealing to some, even to long-time Gentoo Linux users. We’ve been made aware that some people are installing Redcore Linux, then removing Redcore specific bits and converting their systems back to Gentoo Linux. It’s just quicker for them this way, and that is perfectly fine with us, since it fits with our design goals.”

How Redcore Works

“I’ve read many posts claiming Redcore Linux doesn’t support optimization the way Gentoo Linux does,” Mamut says. “But that couldn’t be further from the truth. One thing Redcore Linux does provide is some defaults, yet those are not carved in stone. One can change every aspect of the distribution, like USE flags, CFLAGS, keywords, and so on the same way as in Gentoo. Heck, one can even recompile the whole distribution to take advantage of a certain CPU instruction set or install packages from Gentoo itself.”

Mamut goes on to explain, “Redcore is not as optimized by default as a from-scratch install of Gentoo. It is rather generic so that it can be installed on as many configurations as possible, but it does nothing to prevent such an optimization – it just has to be done after the installation. Of course, if somebody is willing to do such an optimization, we would suggest installing Gentoo itself. Redcore is for those who want a starting point, with some defaults you can change and fine tune to your liking.”

Coming Soon

Redcore has a workable distribution, but it is not standing still. It is currently working on an ARM64 port, optimized for the Raspberry Pi 4. At the time of writing, core userspace programs and utilities can be compiled, but the kernel and the coordination of subsystems remains to be managed. “We hope to have something by the end of the year.” Meanwhile, on AMD54 architectures, Redcore continues in the best tradition of derivatives, taking the rough edges off its parent distribution. ■■■

Info

- [1] Redcore Linux: <https://redcorelinux.org/>
- [2] Gentoo Linux: <https://www.gentoo.org/>

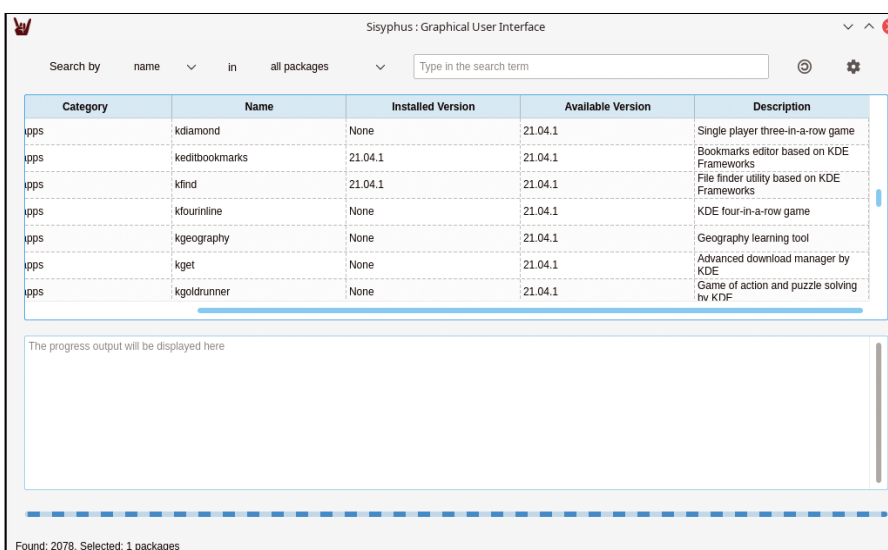


Figure 2: Since Redcore is a rolling distribution, the Sisyphus package manager displays installed and available releases prominently. The name is a joke: In Greek mythology, Sisyphus was eternally condemned to roll a stone to the top of a hill only to have it roll back down – a fitting symbol of a rolling release.

A modern library interior with a curved staircase and bookshelves. The scene is brightly lit, with a blue overlay at the top and bottom. The text is centered on the blue overlay.

Hone your skills with special editions!

Get to know Shell, LibreOffice, Linux, and more from our Special Edition library.

The *Linux Magazine* team has created a series of single volumes that give you a deep-dive into the topics you want.

Available in print or digital format

Check out the full library!
shop.linuxnewmedia.com

FREE DVD!
 JOIN THE **LINUX REVOLUTION!**
 ← ALL THE SOFTWARE YOU NEED!

GETTING STARTED WITH LINUX

• MORE POWERFUL • MORE SECURE • MORE FUN

LEARN HOW TO SET UP A LINUX SYSTEM TO:

- Listen to Music
- Play Games
- Process Photos
- Surf the Web
- and Much More!

2020 Edition
LINUX Special
 WWW.LINUX-MAGAZINE.COM

LINUX Special **301 BEST BASH COMMANDS**

LINUX SHELL HANDBOOK

2021 Edition **LINUX Special**

SUPERCHARGE YOUR LINUX SKILLS

Power at Your Fingertips

- Pipe and redirect output
- Monitor processes
- Create custom scripts

Keep this guide as a permanent reference!

LINUX NEW MEDIA
 The World of Open Source
 WWW.LINUX-MAGAZINE.COM

FREE DVD!
 LibreOffice Full Version

Become a LibreOffice Expert!

2020 Edition

LibreOffice

Dive deep into the world's greatest free office suite

Write Your Own LO Macros
 Save time and automate common tasks

Digital Signatures
 Lock down your private documents

Edit and Save MS Office Files

Create Professional:

- Text Documents
- Spreadsheets
- Presentations
- Databases

Replace MS Office and Google Docs!

LibreOffice*
 Includes full versions for Windows, macOS, and Linux

LINUX Special
 WWW.LINUX-MAGAZINE.COM

LINUX Special **TUNE YOUR LINUX SYSTEM**

2021 EDITION

101 COOL LINUX HACKS

Tricks and shortcuts for Linux geeks

- Recover deleted docs
- Send files without a target IP
- View a handy cheat sheet for your favorite commands

STREAMLINE:
 Clean up hidden files

EXTREME CHROOT:
 Change to a second distro

Discover the secrets of the experts

LINUX Special
 WWW.LINUX-MAGAZINE.COM

Building a website in Markdown with Pandoc

Fast Web

Build a simple web page in Markdown; then convert it to HTML at the command line. *By Mats Axelsson*

Creating a website is a lot of trouble – especially if you just have a few files you wish to publish online. You could type in all the HTML codes by hand, or you could employ a graphic design tool that looks simple but still might be more effort than you want to spend.

Another easier option is to use Pandoc [1]. Pandoc is a universal document converter. You can give Pandoc a text file in any of several markup formats, and it will convert the document to any of several output formats. One common scenario is to format a text file using the simple and expressive Markdown markup language and then use Pandoc to convert the file to an HTML page. Pandoc is a command-line tool, so it allows you to convert a file to HTML in a single command. A collection of command-line options lets you add extra features to the web page, such as a footer bar or a rudimentary navigation menu. With the right libraries, Pandoc can even read programming languages.

You wouldn't want to use Pandoc for a complex site with interactive features and a backend database, but if you are just looking for a quick-and-dirty tool for publishing text to the web, Pandoc

is a very good option. For instance, some organizations use Pandoc in situations where there is a need to maintain documentation that is accessible from the command line but still easily convertible to HTML.

Markdown Magic

The scenario begins with a text file in Markdown format. This article is not intended as an introduction to Markdown, but if you're looking for a primer, you'll find several cheat sheets and tutorials

online [2]. Listing 1 shows a sample Markdown file. As you can see, the format is largely self-explanatory.

You will quickly notice Markdown features such as headings (with # for a top heading and ## for a second-level sub-head). Double tilde makes the text strikethrough. You can also make tables with a simple combination of pipes | and dashes -.

To convert the Markdown text in Listing 1 to an HTML page, run the standard Pandoc command:

```
defaults
html-template.tpl
index.html.md
nwx.sh
site
├── index.html
├── style.css
└── SiteGen.sh
src
├── 10_Ideas_page.md
├── about.md
├── HireMe.md
├── index.md
├── practice.md
└── testimonials.md
style
├── style.css
templates
├── css.default
└── html.default
```

Figure 1: This sample tree is a suggestion, but make sure you know where your original text is and where you put your result!

Lead Image © Martin Blech, Fotolia.com

Listing 1: Sample index.md

```

01 # Escape Mundane Life
02
03 On this website, you will never be ~~BORED~~!
04
05 ## Fun Jokes
06
07 Well, come up with something yourself!
08
09 ## Pure science
10
11 The best way to get to the truth about werewolfs!
12
13 | Moon phase      | Phang size | Measurement time | Result |
14 | ---            | ---        | ---              | ---   |
15 | New Moon       | 0.1        | 00:14            | Skinny |
16 | Waxing Crescent | 0.2        | 00:21            | Skinny |
17 | First Quarter  | 0.6        | 00:09            | Normal |
18 | Waxing Gibbous | 2.3        | 00:32            | Strong |
19 | Full           | 9.3        | 00:19            | Bulky  |
20 | Waning Gibbous | 6.1        | 00:59            | Dirty  |
21 | Third Quarter  | 0.8        | 02:01            | Flappy |
22 | Waning Crescent | 0.2        | 00:01            | Skinny |
23 |                |            |                  |        |
24
25
26 ---
27
28 ```
29 {
30     "Human name": "Ben";
31     "Wolf Name": "Slasher";
32     "age": "32"
33 }
34 ```
35
36 > Conclusion
37 > Ben is a *Werewolf* who should build his human body! [^1]
38
39 ## A link to the city.
40
41 - Some Pictures!
42 1. A bustling City!
43
44     ![Big City](https://www.abc.se/~m9779/images/Storstan.bmp)
45
46 2. Chilling in the bay!
47
48     ![Some boats](images/boats.bmp)
49 ---
50
51 [^1]: Ben is fictive name.
52 <C>

```

```
$ pandoc -t html index.md -o index.html
```

Pandoc outputs to standard output unless you specify an output file with the `-o` parameter.

Structure

Your Pandoc-generated website will be simple and relatively sparse, but you will still want to be methodical about keeping it organized. Pandoc only generates one page at a time, so to reduce clutter and confusion, set up a basic directory structure for your source files – something like in Figure 1.

In the `src/` directory, put your `index.md`, `about.md`, and other Markdown files you will convert to HTML for your site.

Options

The `pandoc` command has many additional options. By default, Pandoc creates document fragments. If you would like to output a standalone HTML document (with `<head>` and `<body>` sections), use the `-s` or `--standalone` option. The `--standalone` flag supports some other options for where to locate text:

```
- -H --include-in-header=
- -B --include-before-body=
- -A --include-after-body=
```

A CSS file is required when generating an ePub but is optional for simple, standalone pages.

If you want to maintain your site using many files, you can list all the files as input and output straight to one file.

```
pandoc *.md -o index.html
```

This command will process all the `.md` files in the order they would have been listed by `ls`. In most situations, you will want more control over where the content falls within the output file, so processing the files in `ls` sort order has some complications – you might want to consider more sophisticated techniques.

Styling with CSS

Styling should occur in a separate file, outside of the page. Any file you point to with the `-H` option will be copied in, and that can include a CSS file as part of the final directory structure.

Usually, you will start by using an existing style. In this example, you can copy the Tufte CSS file into one of your sub-directories and add the link to a header file:

```
<link rel="stylesheet" href="Tufte/tufte.css"/>
```

Footer and Header

If you want your website to look tidy and professional, you might want the site name and URL to appear in a header or footer bar to let people know where they are regardless of which page they are on.

As mentioned previously, the options `-B` (before the body) and `-A` (after the body) let you create headers and footers. You could use these options to create a navigation bar at the top or a consistent footer for all pages. You can repeat these options as many times as you like, though the final result will suffer if you overdo things.

To create the header text you want, save the text in a file and specify the file name with the `pandoc` command using the `-B` option (Listing 2).

Listing 2: Specifying Headers and Footers

```
$ pandoc -t html\
  -H head.html \
  -s -o index.html \
  -A footer.html \
  -B toptext.html \
  --metadata title="Kick Mundane Out"
index.md \
```

Note that Listing 2 also brings in the CSS style sheet using the `-H` parameter. In this case, the `head.html` file only contains a link to the stylesheet.

Suppose I wish to add a copyright notice at the bottom. You can also imagine other things that might appear in a footer, such as social links and other legal information. The footer id tag is common in CSS stylesheets (Listing 3).

A simple nav bar in the header could show the other pages on the site in a list (Listing 4).

Templates

When you convert to a new format, Pandoc will use a default template. Alternatively, you can create your own template. The easiest way to create a template is to write an existing template to your directory and edit it:

```
pandoc -D html > MyTemplate.tpl
```

The simplest sites do not need templates. However, templates provide a means for incorporating variables and

Listing 3: footer.html Footer

```
footer.html
<C>
<div id="footer">
  &copy; 2021 Mats Tage Axelsson
</div>
<C>
```

Listing 4: toptext.html Header

```
<C>
<div id="nav-bar">
  <ul>
    <li>Home</li>
    <li>projects</li>
    <li>About</li>
  </ul>
</div>
<C>
```

simple conditionals that add power and convenience to your web presence.

A template is a file in the final format that you seek – in this case, HTML files.

The big difference between a template and an ordinary file is that templates have variables embedded. In short, variables are enclosed in dollar signs, and you assign values to those variables using your defaults file or command-line options.

For instance, you could use a metadata variable as a placeholder for a title. Then in the defaults file or as part of the `pandoc` command, you could specify a title for the page:

```
--metadata title="
"Escape Mundane Life"
```

If the title shows up in the default template, it comes with conditions. Pandoc checks if `title` is defined elsewhere, and if not, it puts it in (Listing 5).

As you can see, Listing 5 defines both the title and subtitle.

Some ready-made templates are available to help you add common features to your website. For instance, the `pandoc-bootstrap` [3] template includes

Listing 5: Conditional Default Title

```
01 $if(title)$
02 <header id="title-block-header">
03 <h1 class="title">${title}</h1>
04 $if(subtitle)$
05 <p class="subtitle">${subtitle}</p>
06 $endif$
```

Listing 6: Setting Variables with -V

```
pandoc index.md\
  -o index.html\
  -s\
  -t html5+smart\
  -V date="2021-08-15"\
  -M subtitle="More Vitality"\
  --template ../pandoc-bootstrap-adaptive-template/template.html\
  -B toptext.html\
  -A footer.html\
  -c ../pandoc-bootstrap-adaptive-template/template.css\
  --metadata title="Escape Mundane Life"
```

several elements that add value to your site. Bootstrap sets up CSS and connects to some JavaScript to give you a navigation bar and maybe a table of contents. You can also add an author name and a date.

In the code, you define where you want the variables to go:

```
$for(author-meta)$
  <meta name="author"
content="${author-meta}" />
$endfor$
$if(date-meta)$
  <meta name="date"
content="${date-meta}" />
$endif$
```

There are two ways of setting the values; one method is to add a YAML block [4] at the top of your document, in this case, `index.md`.

The three dashes indicate the start and end of a block:

```
---
subtitle: "More creativity"
author: "Mats Tage Axelsson"
date: "2021-08-12"
---
```

The other way to set the variables is with the `-V` parameter. For instance, note that the command in Listing 6 sets the date using the `-V` variable.

As you can see in Figure 2, the data is added to your web page.

Slides

Pandoc has built-in support for slide-shows and supports several different slideshow standards. Regardless of which output format you wish to use, the format of your source document will be in Markdown.

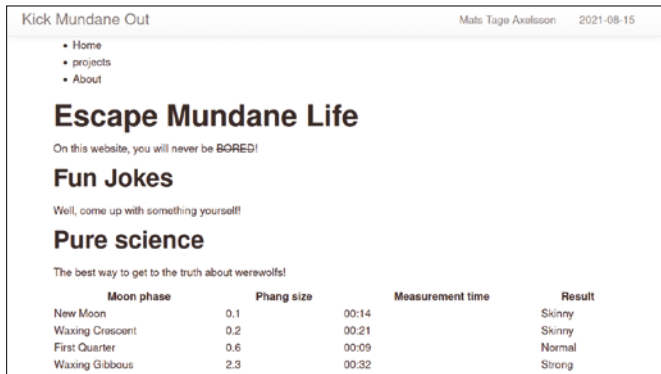


Figure 2: After you have added the metadata in the command, the date shows up in the right hand top corner, thanks to CSS.

Begin each slide with a horizontal rule. A slide-level header also starts a new slide. If you want to have notes available while you are present, just add them using the `:::` notes tag, closing it with `:::`.

Adding notes is really simple; styling them is a little trickier – CSS to the rescue. If you are using `reveal.js`, you set all variables as you would without Pandoc. Otherwise, you can set styling using a directory under your Pandoc `$DATA-DIR$` – usually `~/pandoc`; standards are available in the system directory.

The command is the same as usual – just choose the output format you like.

```
pandoc -t slidy -i slideshow.md
-o site/slideshow.html -s
```

Listing 7: Multiple Output Files

```
#!/usr/bin/env bash
for page in $(ls pages); do
  pandoc --from markdown_github+smart+yaml_metadata_block+auto_identifiers
    "pages/$page" \
    -o "public/$(basename $page .md).html" \
    --template templates/page.html \
    -V navigation="$(cat navigation.html)" \
    -V footer="$(cat footer.html)"
done
```

You should be able to pick up more information from the web sites that support the slideshow standards.

Converting to HTML

One command with many input files will create one output file. A more conventional approach is

to create one page for each of the parts of your site. This approach also enables blogging-like functions, if you do it right. The simple script for creating multiple output files is shown in Listing 7.

To use the script in Listing 7, you have to create the tree structure, including a `pages/` directory that has all the pages for your site. On top of the `pages/` directory, you have the templates and files that the command references (`templates/page.html`, `navigation.html`, and `footer.html`). The complete code is available online [5].

To take full advantage of these templates, and make your own template, look through the code, and set any necessary variables.

MkPage

A good front end can make things easier. One such front end is MkPage [6].

The MkPage Project uses Pandoc and its template language and adds a few tools, allowing you to define the entire site in a small project directory. All the tools follow *nix standards. To run MkPage, you set values on the command line that matches the variables in your template file.

You can use all tools in the project separately. Available tools include the main MkPage tool, as well as a tool called BlogIt and others. As you might guess, the BlogIt tool lets you write all your blog posts in Markdown.

Conclusion

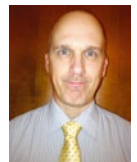
Using Pandoc to create a static website in Markdown will save you from using big, complicated build systems on the web or writing your own HTML. This option is not suitable for developing advanced websites. The emphasis is on creating small, functional sites that present you or your offerings in the best possible light without a lot of complications. ■■■

Info

- [1] Pandoc: <https://pandoc.org/>
- [2] Markdown tutorial: <https://www.markdowntutorial.com/>
- [3] Bootstrap template: <https://github.com/diversen/pandoc-bootstrap-adaptive-template/>
- [4] YAML: <https://yaml.org/>
- [5] Pandoc template: <https://github.com/jillesvangurp/www.jillesvangurp.com/blob/devtoarticle/templates/page.html>.
- [6] MkPage Project: <https://caltechlibrary.github.io/mkpage/>

Author

Mats Tage Axelsson has a set of modes that he still has not found the keyboard shortcut to change.





This is your command line on lsd

Colorful lsd

A revamp of ls, lsd offers color coding plus revised options relevant to the modern computer.

By Bruce Byfield

If you work at the command line, you probably use `ls` [1] frequently to list directories and their contents. However, it is one of the oldest commands, and many of its options are no longer relevant to modern computer use. In fact, most people can get by with

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 (<http://brucebyfield.wordpress.com>). He is also co-founder of Prentice Pieces, a blog about writing and fantasy at <https://prenticepieces.com/>.

only a small sub-set of its available options. One of several new revamps, `lsd` (LSDeluxe) [2] (Figure 1) attempts to modernize and simplify `ls` by assuming a default color code and – apparently – by reducing the number of options and adding options of its own. However, since this is 0.2.21 release, it is hard to be sure exactly what the general release will look like.

The `lsd` release page includes Linux packages for Debian and Gentoo. An alternative is to build from source with the Rust Cargo package manager or install a Snap package. Should you install from source, you will also need to install Nerd Fonts [3] in order to use fancy icons to identify file types.

On installation – or perhaps later, after you have tried `lsd` and have a better sense of what it can do – you may also want to add a configuration file (Figure 2), copying and modifying the sample provided on the project page. This configuration file should be placed in `~/.config/lsd/config.yaml`. Table 1 lists the main fields available and their settings. Default configurations can be overridden from the command line but can reduce the length of entered commands.

Note that in the current release the set colors cannot be altered, although that may change in later releases. Figure 3 shows the default colors currently in use. Colors are only the most obvious ways

```
bb@nanday:~$ lsd
├── Applications
├── Arduino
├── art-wallpaper.png
├── ATmega32u4
├── backup
├── blog-banner1.png
├── bookmarks-logons.html
├── bookmarks.html
├── Bruce Byfield.html
├── bruce-three-quarters-large.png
├── Byfield-Alteration-Agreement-October-7-2015.doc
├── Calibre Library
├── inigo.jpg
├── keyboard.jpg
├── Mail
├── mast.jpg
├── mast.png
├── mature-student-award-card.odt
├── mature-student-speech.ogg
├── music
├── Notebooks
├── notes-on-hiring-an-editor.txt
├── 00oPy-1.11
├── outline-chess-on-wednesdays.odt
├── Pictures
├── Webcam
├── work
└── Zotero
```

Figure 1: One of the leading updates to `ls`, `lsd` attempts to modernize and simplify the command.

Lead Image © bluedarkat, 123RF.com

that directories and files can be coded, the others being `--icons` and `--icon-themes` (Figure 4) and `--classify` (`-F`), which uses special characters (`*`, `/`, `=`, `>`, `@`, and `|`) to indicate file types. Probably, most users will want to use only one of these choices.

Options

The current `ls` command has 58 options. By contrast, the current `lsd` release has 25. More might be added before the general release, but `lsd` is already a functional match for `ls` and broadly backwardly compatible with it, with much of the same structural logic. However, some of the `ls` options are less relevant than they once were. For instance, today only a few applications like Bluefish use `~` to indicate a backup file, so the `ls` option `--ignore-backups` (`-B`) is unlikely to be missed. Similarly, the decline of Emacs's popularity means that the `--dired` (`-D`) option to generate output for Emacs's Dired mode seems no longer necessary or at least a low priority to implement. Nor does `lsd` require the `-C` option for displaying in columns – because it does so by default and other options such as `tree` or single-line displays are folded into the same command. The overall result is that `lsd` is easier to learn than `ls`, especially when more than basic options are used.

So what options are left? First, the options that can be placed in a configuration file (Figure 2). They take the same possible values as in the configuration file and override the file as well. Another set of

options are those found in `ls`. In particular, `--all` (`-a`) displays dot files, and `--no-symlink` does not display symbolic links, while `--directly-only` (`-d`) lists only directories. The same sorting options are also

```
# == Classic ==
# This is a shorthand to override some of the options to be backwards compatible
# with 'ls'. It affects the "color"->"when", "sorting"->"dir-grouping", "date"
# and "icons"->"when" options.
# Possible values: false, true
classic: false

# == Blocks ==
# This specifies the columns and their order when using the long and the tree
# layout.
# Possible values: permission, user, group, size, size_value, date, name, inode
blocks:
- permission
- user
- group
```

Figure 2: Use a configuration file to customize `lsd`.

User/Group	Permissions	File Types	Last time Modified	File Size
User	Read	Directory	within the last hour	Small File
Group	Write	Executable File	within the last day	Medium File
	Execute	Non-Executable File	older	Large File
	Execute with Stickybit	Broken Symlink		Non File
	No Access	Pipe/Symlink/Blockdevice /Socket/Special		
		CharDevice		

Figure 3: `lsd` uses a complex color code for files.

Table 1: Selected Configuration Settings

Field	Description	Values	Default
<code>classic</code>	Sets backward compatibility with <code>ls</code>	<code>false, true</code>	<code>false</code>
<code>blocks</code>	Specifies the columns and their order	<code>permission, user, group, size, size_value, date, name, inode</code>	<code>permission, user, group, size, date</code>
<code>color</code>	When to colorize	<code>never, auto, always</code>	<code>auto</code> (never when <code>classic</code> is set to <code>true</code>)
<code>date</code>	Date format for date column	<code>date, relative, '+date_format'</code> (a strftime formatted value, i.e., <code>'%d %b %y %X'</code>)	<code>date</code>
<code>dereference</code>	Whether to dereference symbolic links	<code>false, true</code>	<code>false</code>
<code>display</code>	What items to display	<code>all, almost-all, directory-only</code>	<code>all</code>
<code>icons:when</code>	When to use icons	<code>always, auto, never</code>	<code>auto</code> (never when set to <code>classic</code>)
<code>icons:theme</code>	Which icon theme to use	<code>fancy, unicode</code>	<code>fancy</code> (unicode if Nerd Fonts not installed)
<code>icons:separator</code>	How the icon is separated from the file name	<code>' '</code>	Single space
<code>layout</code>	Which layout to use	<code>grid, tree, oneline</code>	<code>grid</code>
<code>recursion:enabled</code>	Whether to enable recursion	<code>false, true</code>	<code>false</code>
<code>recursion:depth</code>	The recursion depth	Any positive number	<code>3</code>
<code>size</code>	The format of the size column	<code>default, short, bytes</code>	<code>default</code>
<code>sorting:column</code>	Specify what to sort by	<code>extension, name, time, size, version</code>	<code>name</code>
<code>sorting:reverse</code>	Whether to reverse the sorting	<code>false, true</code>	<code>false</code>
<code>sorting:dir-grouping</code>	Whether to group directories and where	<code>first, last, none</code>	<code>none</code>

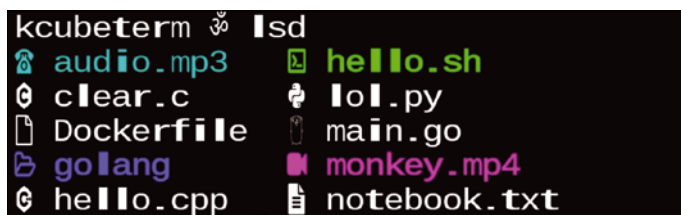


Figure 4: A close-up of the fancy icons that lsd can use.

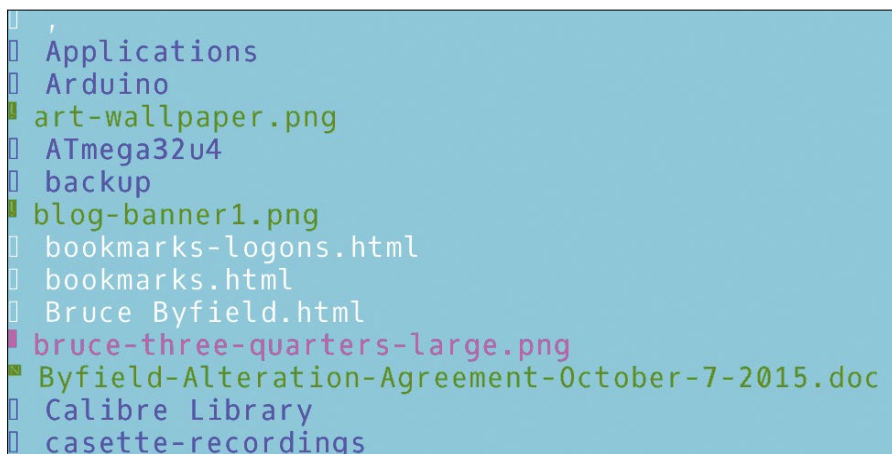


Figure 5: An alternative to lsd's column display is to display file names, one per line.

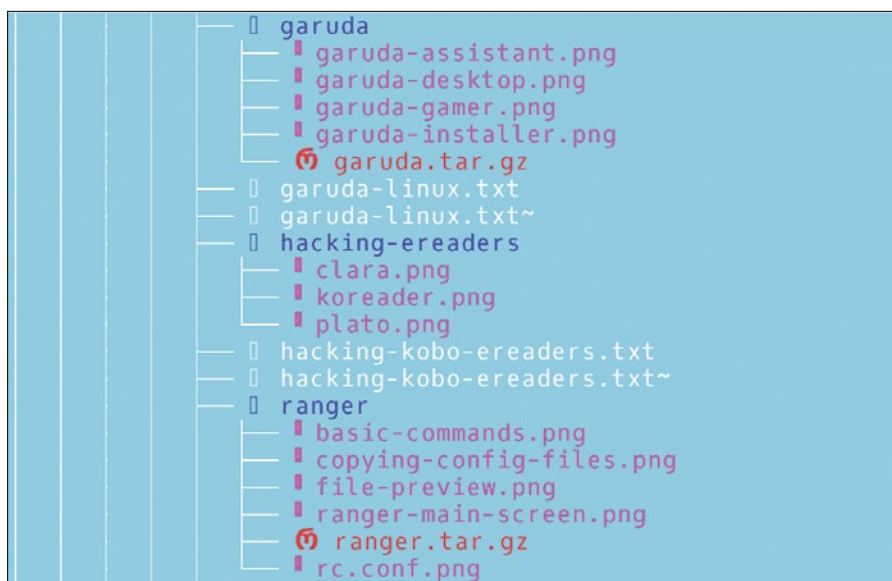


Figure 6: A tree structure is also an option in lsd.

used: `--recursive (-r)`, `--reverse (-r)`, `--sizesort (-s)`, and `--timesort (-t)`, all of which are self-explanatory. Other options include `--total-size`, which displays the total size of directories, and `--oneline (-1)` and `--tree`, which offer alternative displays (Figure 5 and 6). And if you have no wish for color or icon coding, you can use `--classic` to suppress them. If you are proficient with `ls`, many of these options will be familiar.

Naturally, which options you require depends on your current tasks, but you are likely to develop preferences in layout. When you do, consider creating an alias for `lsd` so that you have no need to break the habit of typing `ls`. To give the simplest alias, in your `~/.bashrc` file or its equivalent, add the line:

```
alias ls='lsd'
```

You may want to set other aliases for `lsd`, each with its own set of options.

Other ls Updates

In addition to `lsd`, at least five more revisions of `ls` are competing for the same niche: `exa`, `colorls`, `ls++`, `ls-go`, and `k`. Many of these are released under an MIT license, and most focus on a color display. However, `lsd` is an ideal place to begin exploring the new alternatives. Backwards compatibility makes it an obvious replacement for `ls`. While various forms of file coding may be too complex for some users, `lsd` offers several choices. However, if `lsd` is not for you, you have others to fall back on. Obviously, the time has come to revamp one of Linux's most basic commands. ■■■

Info

- [1] `ls`: <https://en.wikipedia.org/wiki/Ls>
- [2] `lsd`: <https://github.com/Peltoche/lzd#faq>
- [3] Nerd Fonts: <https://github.com/ryanoasis/nerd-fonts/blob/master/readme.md>

The sys admin's daily grind: Age

Master of the Keys

Charly makes life easier for himself by using the lean Age tool for command-line data encryption tasks. *Charly Kühnast*

Age (think “aghe” in “spaghetti”) promises an uncomplicated approach to encrypting and decrypting files. Written in Go, both the source code and precompiled binaries for various platforms can be found on GitHub [1]. I tried Age on a Raspberry Pi, but the tool also runs on Linux on a PC, macOS, and Windows. The author says the current release from mid-June 2021 is “maybe actually the last v1.0.0 release candidate.”

Age supports two ways to encrypt and decrypt files. With the first option, you can create a key file (Age calls this an

identity file) based on SSH keys with whose contents you then encrypt the data. In contrast to SSH, this is a symmetrical procedure, so the same file is also used for decryption.

As a second option, Age lets you use a passphrase to encrypt the data. This has the advantage that you can think up a new passphrase for each encryption process. A person you trust and give a passphrase to can use it to decrypt your data – but only the data you encrypted with precisely that one passphrase.

Let's look at both methods. By way of an example, I copied the file `/etc/passwd`

to my home directory and will now encrypt it using both methods in turn. For the key file method, I need to generate the key file first – I can do this with `age-keygen` (Figure 1).

The next step is to encrypt the `passwd` file with the key I just generated (Listing 1, line 2). This creates the encrypted `passwd.age` file. To decrypt it, use the command from the line 3 of Listing 1.

Now I'll do the same thing but with the passphrase method. Again, I encrypt the `passwd` file and give it the name `passwd.age`. After the corresponding call shown in line 1 of Listing 2, Age asks me for the desired passphrase. If I don't enter one, the tool generates one itself. Decryption works in the same way, but you can leave out the `--passphrase` parameter here (line 4).

The entire procedure's security relies, of course, on careful handling of the key or the passphrase. ■■■

```
pi@carpi:~$ age-keygen -o age-key.text
Public key: age14s7swvqklmvnqxp3xf0mu0xn8uum8wak
nrk5gpn438nspd4d5qdqgez8k4
pi@carpi:~$ ls -l age-key.text
-rw----- 1 pi pi 189 Jul  8 13:57 age-key.text
```

Figure 1: `age-keygen` handles the task of generating a key file.

Listing 1: Key Method

```
01 $ age-keygen -o age-key.txt
02 $ age --encrypt -i age-key.text -o passwd.age ./passwd
03 $ age --decrypt -i age-key.text -o passwd passwd.age
```

Listing 2: Passphrase Method

```
01 $ age --passphrase -o passwd.age passwd
02 Enter passphrase (leave empty to autogenerate a secure one):
03 Using the autogenerated passphrase "release-response-step-brand-wrap-ankle-pair
  -unusual-sword-train".
04 $ age --decrypt -o passwd passwd.age
05 Enter passphrase:
```

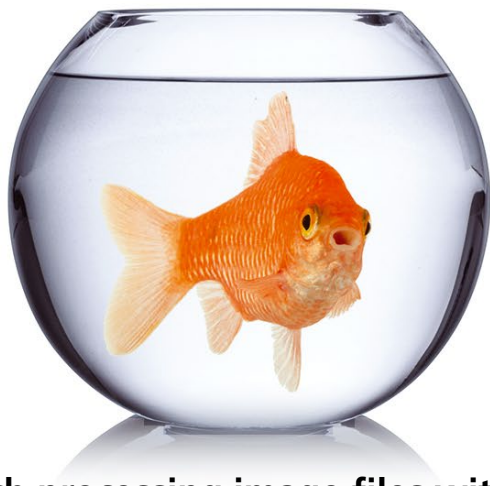
Info

[1] Age: <https://github.com/FiloSottile/age>

Author

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





Batch processing image files with **imgp**

Rapid Resizer

In no time at all, **imgp** can change the resolution of images, as well as convert files from PNG to JPEG, remove metadata, and rotate images. *By Ferdinand Thommes*

Suppose you have hundreds or even thousands of photos that you want to present on a web page. If the images use the original camera resolution, they could severely slow down or even paralyze a website populated with this data. Optimizing the images can save bandwidth on the network and space on the storage medium.

With a large number of images, a batch processing tool such as **imgp** [1] comes in handy. The command-line tool, formerly known as **imgd** and written in Python, provides functions for resizing images in JPEG and PNG formats, as well as converting from PNG to JPEG, rotating the images, and removing metadata.

Even with thousands of files, **imgp** does its job at lightning speed. The developer, Arun Prakash Jana, achieves this through multicore processing, an adaptive algorithm, and the Python Pillow library [2].

On processors that support single instruction, multiple data (SIMD) [3], you can use Pillow-SIMD [4] as an alternative to further speed up processing. In testing on an external hard disk with a

USB 2 connection, Jana claims to have converted 8,823 images of different resolutions (with a total size of 4.5GB) to a resolution of 1366x1000 pixels in about eight minutes, with the resulting total size dropping to 897MB.

Easy Access

To install **imgp** on a current distribution, you can simply use the package manager in most cases. The current version 2.8 is available on Arch Linux, CentOS, Debian Testing and Unstable, Devuan, Fedora, openSUSE Leap 15.3, Ubuntu 21.04, and others.

If your distribution ships with an older version of **imgp**, install the packages from GitHub for DEB and RPM-based distributions or build the package from the source code [5]. Python 3.5 or newer is required. The **nnn** [6] terminal file manager, also developed by Jana, offers batch conversion using **imgp** as a script. (Jana is also the developer of another useful tool, **googler** [7]).

Using **imgp**'s options, you can resize image files by specifying a percentage or a resolution. You can also rotate the image clockwise by a freely selectable angle, optimize images to save space,

convert PNG to JPEG, delete Exif metadata, and more.

Options

To test out various **imgp**'s capabilities, we tried out various **imgp** command-line options on 65 vacation photos taken with a smartphone. The image sizes ranged from around 2 to 5MB, with a total size of 215.7MB. The tests were run on a PC with a Ryzen 7 3700X CPU with 32GB RAM.

If you enter **imgp** at the command line without specifying any other parameters, the help page (Figure 1) loads with a list of all the options for manipulating the size or orientation of an image. As usual, **imgp** assumes the current directory as the source unless you specify a different source. However, you can also process multiple sources in a single command.

Amazingly Fast

As the simplest test case, we entered the following command

```
imgp -x 1024x768
```

in the directory containing the 65 image files. This command converts all the

```
ft@blue:~$ imgp
usage: imgp [-h] [-x res] [-o deg] [-a] [-c] [-e] [-f] [-H] [-l] [-k] [-m] [-M res] [-n] [-N] [-O] [-P] [-q N] [-r]
          [-s byte] [-w] [-d]
          [PATH ...]

Resize, rotate JPEG and PNG images.

positional arguments:
  PATH                source file or dir [default: current dir]

optional arguments:
  -h, --help          show this help message and exit
  -x res, --res res   output resolution in HxV or percentage
  -o deg, --rotate deg rotate clockwise by angle (in degrees)
  -a, --adapt         adapt to resolution by orientation [default: off]
  -c, --convert       convert PNG to JPG format [default: off]
  -e, --eraseexif    erase exif metadata [default: off]
  -f, --force         force to exact specified resolution [default: off]
  -H, --hidden       include hidden (dot) files [default: off]
  -l, --includeimgp  re-process _IMGp files. * RISKY: refer to docs
  -k, --keep         skip (honors -c or --pr) images matching specified
                    H or V or --res=100 [default: off]
  -m, --mute         operate silently [default: informative]
  -M res, --minres res min resolution in HxV or percentage of --res to resize
  -n, --enlarge     enlarge smaller images [default: off]
  -N, --nearest     use nearest neighbour interpolation for PNG [default: antialias]
  -O, --optimize    optimize the output images [default: off]
  -P, --progressive save JPEG images as progressive [default: off]
  -q N, --quality N quality factor (N=1-95, JPEG only) [default: 75]
  -r, --recurse     process non-symbolic dirs recursively [default: off]
  -s byte, --size byte minimum size to process an image [default: 1024]
  -w, --overwrite  overwrite source images [default: off]
  -d, --debug      enable debug logs [default: off]

Version 2.8
Copyright © 2016-2020 Arun Prakash Jana <engineerarun@gmail.com>
License: GPLv3
Webpage: https://github.com/jarun/imgp
ft@blue:~$
```

Figure 1: To access the help page, enter `imgp` without any other parameters. The options listed on the help page include all the common functions for image manipulation.

images (with various resolutions) to 1024x768 pixels. The entire

process took 1.29 seconds in the test and reduced the folder size from 215.1MB to 8.4MB (Figure 2). Listing 1 shows the command structure for processing multiple folders and files simultaneously.

Listing 1: Processing Multiple Sources

```
imgp -x 1024x768 ~/images/image1 ~/photos/image2 ~/pictures/
```

Listing 2: Changing the Quality

```
01 imgp -q 60 -x 1024x768 <image>
02 imgp -x 1024x768 -s 51200
```

If you are in a directory that contains images in both JPEG or PNG formats, the program will process them without you having to specify the directory explicitly. The reduced images will have a new extension (`_IMGp.jpeg` or `_IMGp.png`) and end up in the same directory as the source material.

If you use the `-wx` option instead of `-x`, `imgp` will delete the source files, and the converted images will keep the original file names without the additional `_IMGp`. If you use the `-n` or `--enlarge` option, only PNG files with resolutions smaller than

Pixels or Percentages

Instead of specifying the resolution in pixels, you also can specify a percentage of the original image. The command then changes to

```
imgp -x 30
```

which reduces all images to 30 percent of the original resolution. This time, the process took 1.31 seconds and reduced the size of the directory to 12.3MB.

Instead of a fixed resolution or percentage, you can also specify the desired width (800, in this case) as follows:

```
imgp -x 800x0
```

and `imgp` will keep the original image's aspect ratio. Note: On `imgp`'s GitHub page, the aspect ratio information is shown as horizontal to vertical (abbreviated to `h x v`, which is different from the standard `h x w` for height to width).

```
Ostsee: bash — Konsole
./IMG_20200928_045907.jpg
3648x2736 -> 1024x768
4051196 bytes -> 236730 bytes

./IMG_20201002_114132.jpg
3648x2736 -> 1024x768
5880053 bytes -> 270685 bytes

./IMG_20201002_141456.jpg
2736x3648 -> 576x768
1672901 bytes -> 49906 bytes

./IMG_20201003_151500.jpg
3648x2736 -> 1024x768
3902875 bytes -> 187030 bytes

./Alte Allee auf Rügen.jpg
4496x3000 -> 1024x683
2841654 bytes -> 172640 bytes

./IMG_20201002_141415.jpg
3648x2736 -> 1024x768
2085250 bytes -> 83574 bytes

./IMG_20200928_045837.jpg
3648x2736 -> 1024x768
4492895 bytes -> 250275 bytes

./IMG_20200808_081246.jpg
3648x2736 -> 1024x768
4414185 bytes -> 281003 bytes

./IMG_20200928_192419.jpg
5632x4224 -> 1024x768
4190208 bytes -> 114549 bytes

65 processed in 1.2963 seconds.
ft@blue:~/imgp-test/Ostsee$
```

Figure 2: In the simplest case, you just specify the desired resolution. The software converts the images in a very short time.

the target resolution will be processed. The `-f` or `--force` option forces precisely the specified target resolution (Figure 3).

Quality Control

Unless you use the `-i` or `--includeimgp` option, `imgp` ignores previously converted files. You can suppress the output for each manipulated image in the terminal by specifying `-m` or `--mute`. You can further reduce image size with `-o` or `--optimize`, but it will result in loss of quality and a longer processing time.

For JPEG files, you use `-q` or `--quality` and assign a value from 1 to 95 to the pa-



Figure 3: If you want to experiment, you can force any aspect ratio with the `-f` or `--force` option.

```

png-jpg : bash — Konsole
ft@blue:~/imgp-test/png-jpg$ imgp -c
Source omitted. Processing current directory...

./grub.png
705x403 -> 705x403
31525 bytes -> 21397 bytes

./Screenshot_20210611_220020.png
798x568 -> 798x568
81375 bytes -> 51322 bytes

./cute1.png
705x383 -> 705x383
112143 bytes -> 25881 bytes

./viv4-trans.png
1430x589 -> 1430x589
443469 bytes -> 117237 bytes

./cutefish.png
1918x1004 -> 1918x1004
665933 bytes -> 90260 bytes

5 processed in 0.0751 seconds.
ft@blue:~/imgp-test/png-jpg$

```

Figure 4: To convert PNG files to JPEG, use the `-c` or `--convert` option.



Figure 5: You can remove the Exif metadata, which the camera stores in the image.

parameter to specify the desired quality. Line 1 of Listing 2 defines a quality of 60 (75 is the default). You can also stipulate that you only want to process files above a certain size. Line 2 of Listing 2 only processes files that are larger than 50KB (51200 bytes).

Another trick that `imgp` has up its sleeve is clockwise rotation. If an image has the wrong orientation, you can fix this by rotating it through 90 degrees with the command:

```
imgp -o 90 ~/image.jpg
```

The tool can additionally convert from PNG to JPEG if you add the `-c` or `--convert` option (Figure 4).

The `-e` or `--eraseexif` option removes the metadata from the processed images (Figure 5). To see how much space could

be saved by removing the metadata, we used the following command on a directory consisting of 8.4MB:

```
imgp -ex 1024x768
```

The result was a directory size of 6.4MB.

Last but not least, to solve problems, the `-d` or `--debug` switch lets you create a logfile.

Conclusions

There are many conversion tools similar to `imgp` (both with and without a GUI), including ImageMagick, GraphicsMagick, and Converseen. What sets `imgp` apart is the speed at which the tool does its work. Using `imgp` as a preliminary step, you can bulk process large collections with thousands of images in a very short time and then move on to focus on advanced editing. ■■■

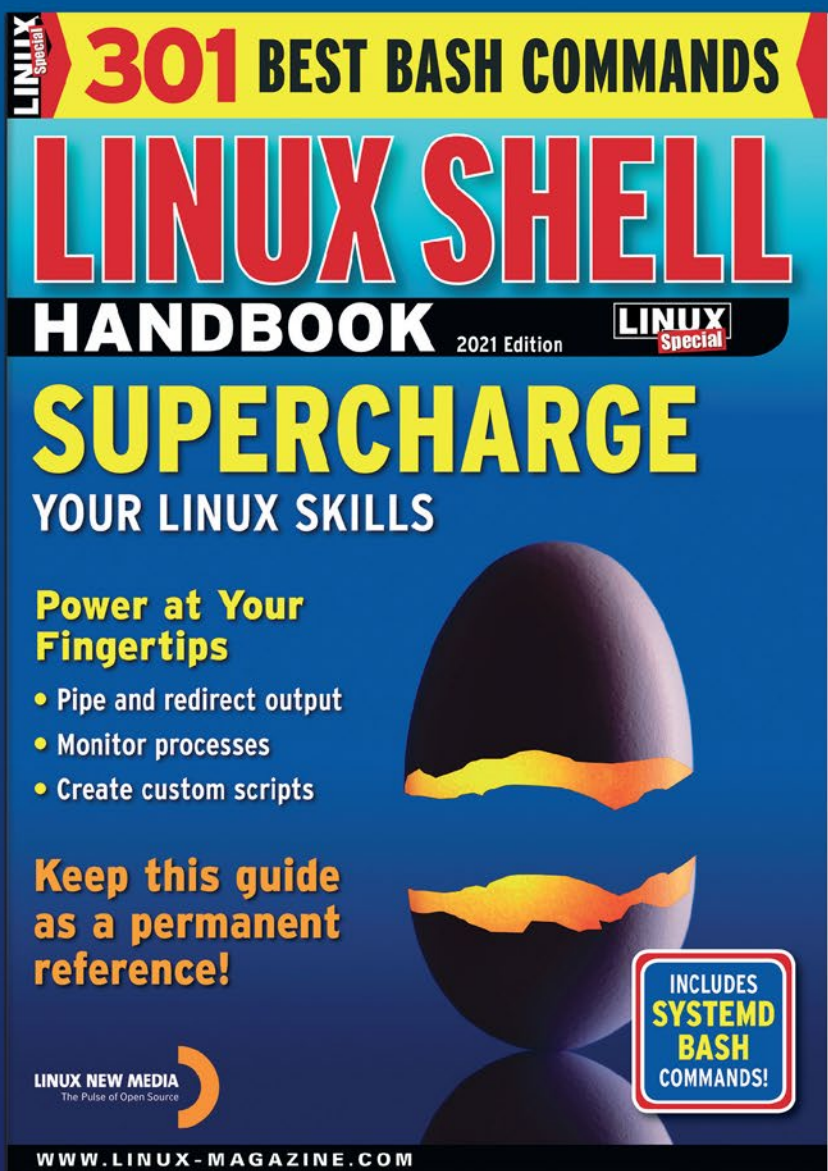
Info

- [1] `imgp`: <https://github.com/jarun/imgp>
- [2] Pillow: <https://pillow.readthedocs.io/en/stable/>
- [3] SIMD: <https://en.wikipedia.org/wiki/SIMD>
- [4] Pillow-SIMD: <https://github.com/uploadcare/pillow-simd>
- [5] Binary packages: <https://github.com/jarun/imgp/releases/tag/v2.8>
- [6] `nnn`: <https://github.com/jarun/nnn>
- [7] `googler`: <https://github.com/jarun/googler>

THINK LIKE THE EXPERTS

Linux Shell Handbook 2021 Edition

This new edition is packed with the most important utilities for configuring and troubleshooting systems.

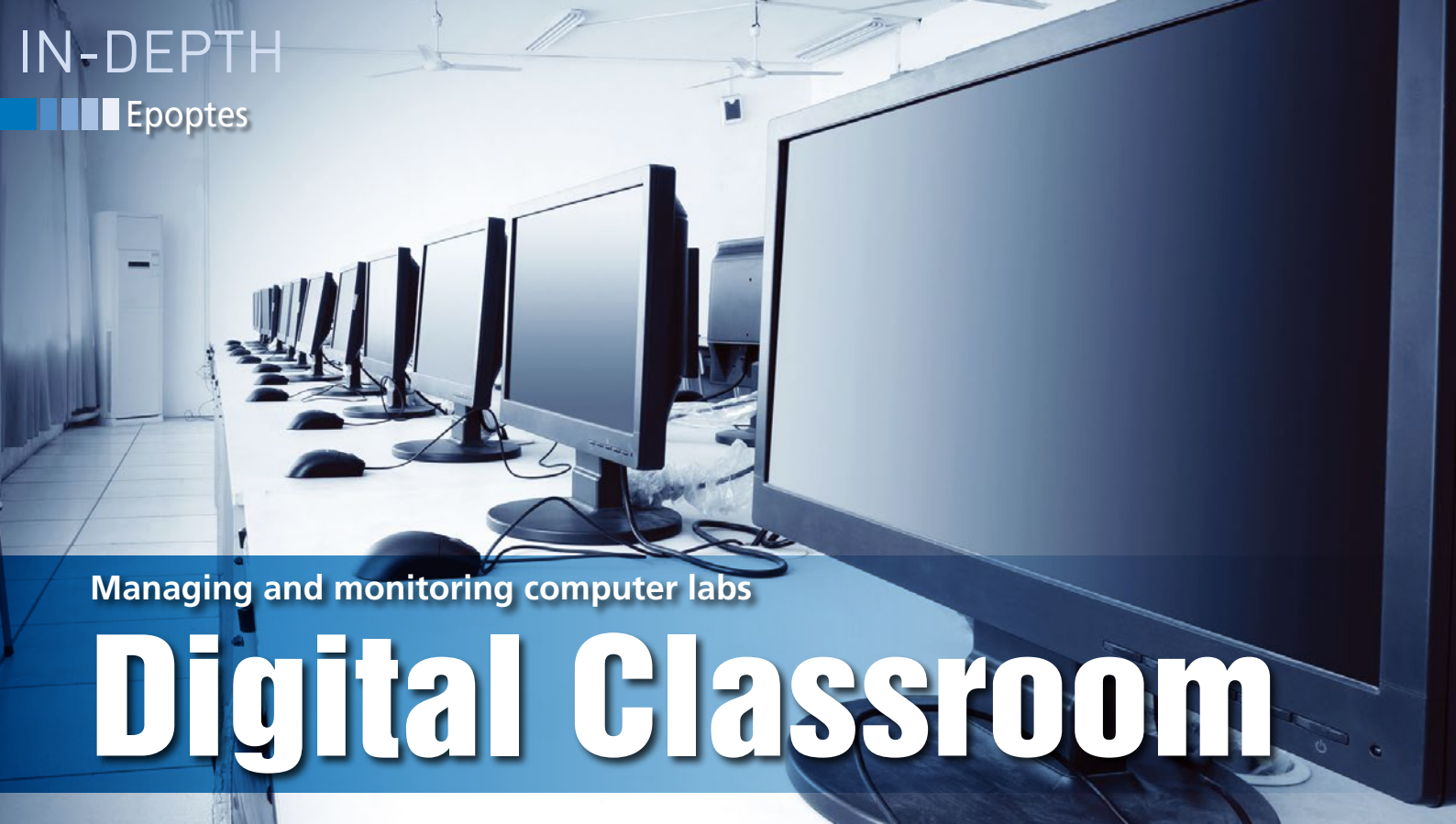


Here's a look at some of what you'll find inside:

- Customizing Bash
- Regular Expressions
- Systemd
- Bash Scripting
- Networking Tools
- And much more!

ORDER ONLINE:

shop.linuxnewmedia.com/specials



Managing and monitoring computer labs

Digital Classroom

If your school's computer lab consists of Linux machines, Epoptes provides an interesting alternative to conventional management and monitoring programs. *By Erik Bärwaldt*

For most students, lessons in the computer lab are a welcome change from the traditional classroom setting. However, especially for younger elementary students, learning on a computer can also be a distraction, making it hard for teachers to keep a student's attention on the current lesson.

Digital classroom management software can help solve this issue by letting the teacher control the students' computers to ensure that the screen content does not constantly distract students. Linux offers several open source management solutions for school computer labs, including the well-known Veyon [1].

In this article, I explore another option, Epoptes (Greek for overseer) [2]. Epoptes offers an equally sophisticated client-server solution for managing and monitoring Linux-based computer labs, with the added advantage of being easy to set up and maintain.

Functions

Epoptes maintains separate packages for the server (the teacher's computer) and the clients (the students' computers), with precompiled binaries available in the respective repositories for Debian, Ubuntu, Fedora, and openSUSE. Epoptes can also be integrated into derivative

distributions, such as Linux Mint. For Arch Linux, the software can be found in the Arch User Repository (AUR).

The Epoptes server is used to control heterogenous client environments; the students' computers do not have to run on the same distribution, but they must be Linux environments (it does not support other operating systems such as Windows or macOS). Epoptes opens a connection between the server and the clients via encrypted reverse VNC channels, with certificates serving as the authentication mechanism.

Epoptes combines classic control tools, allowing the client systems to be turned on and off, as well as locked individually. Epoptes supports monitoring and controlling each student computer, as well as broadcasting the teacher's screen (the server) to the client displays. In addition, the server can also remotely execute commands on the students' systems and send messages to them.

Installation

Epoptes is a 64-bit application that requires corresponding versions of the supported Linux distributions. It can be conveniently set up from the respective repositories using the distribution's own graphical front ends. You install the server package (*epoptes*) on the teacher's computer

and then the client package (*epoptes-client*) on each student's computer.

To run the server, you still need to create an *epoptes* group on the teacher's computer. Using the commands from Listing 1, you then join the group and log in. The next step is to start the Epoptes server with the desktop menu; this takes you to a clear-cut administration window (Figure 1).

For the clients and the server to connect, you next need to tell the clients the location of the server system using the `/etc/default/epoptes-client` file. Open this file in any editor and specify the DNS name of the server machine in the `SERVER=` line. Alternatively, if you run the server machine with a static IP, you can also enter the server's address and hostname in the `/etc/hosts` file on the students' machines.

To ensure secure communication between the clients and the server, load the server's certificate onto the client computers in a final configuration step. To do this, call

```
sudo epoptes-client -c
```

Listing 1: Logging in to the Server

```
$ sudo usermod aG epoptes $USER
$ newgrp -- epoptes
```

Lead Image © Gui Yongnian, 123RF.com

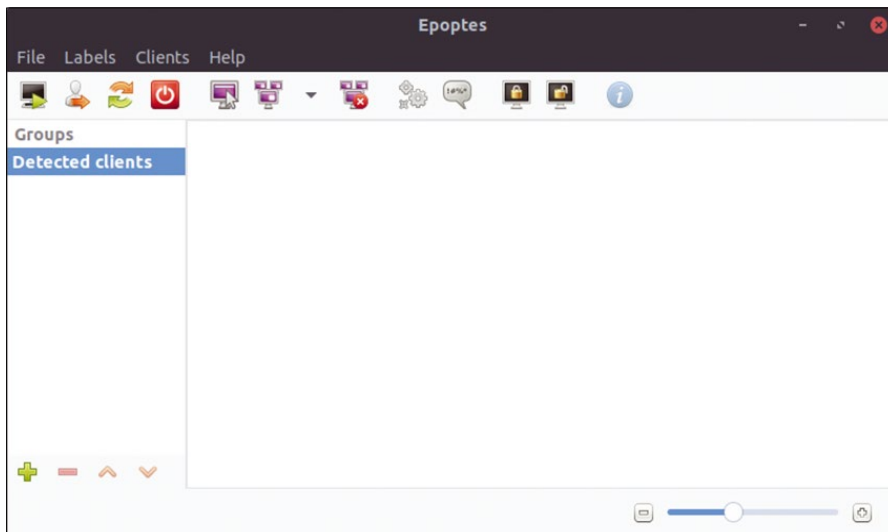


Figure 1: The administration window on the teacher's computer is self-explanatory.

on each client and then reboot.

Getting Started

Upon start-up on the teacher's computer, Epoptes automatically connects to the switched-on student computers. After a moment, a thumbnail preview of each student desktop appears in the right pane of the program window (Figure 2). Following the client hostnames, Epoptes displays the name of the respective logged-in user.

To control a student computer remotely, simply double-click on the corresponding thumbnail. TigerVNC [3], which is implemented in Epoptes, opens a large window with the selected computer desktop. From the teacher's computer, you can now control the selected client's computer using keyboard and mouse input.

Core Tools

The core tools for managing and monitoring client computers can be found in the buttonbar and menubar on the server computer. The first five buttons in the buttonbar are for basic control of the student computers: After a separate configuration, you can boot selected clients, log out users on the selected client, restart selected clients, shutdown selected clients, and take control of a selected client, respectively.

The sixth button lets you broadcast the teacher's screen to all active clients, and the seventh button stops broadcasting. The eighth button lets you run a command on the students' computers,

while the ninth button opens a dialog with an input field for free text, which is broadcast to all active clients when you press *Send*. The message appears on the client computer in a new window and stays there until the student clicks the *Close* button.

The 10th and 11th buttons lock and unlock the selected clients. The final button on the far right displays some information about the selected client system, such as the hostname and IP address, as well as information about the installed CPU and memory size.

Managing Client Computers

The menubar offers some additional functions for managing and monitoring computers in the lab. From the Labels

menu on the teacher's computer, you can modify the client labels found in the teacher's program window. Hostnames and usernames can be shown and hidden here.

Clients | Broadcasts lets you directly influence the connected client systems via the server computer. For example, *Monitor user* pulls up the screen of the currently active student computer into a VNC window on your screen but without you controlling the client. *Assist user*, on the other hand, opens a TigerVNC window on the server that displays the active client's screen, but now the teacher's computer can take control of the student computer.

The *Broadcast screen* and *Broadcast screen (windowed)* options show the screen of the teacher's computer in a full-screen view or in a window on the selected client. You can also select multiple student computers here. In full-screen mode, the client computer's desktop disappears completely and is replaced by the teacher's screen. If you select the windowed option, the teacher's screen appears in a window, but the student's panel bar remains visible. When broadcasting screens, the teacher's screen should not have a higher screen resolution than the clients. Otherwise the teacher's screen content will appear cut off on the client computers and require scrolling to see the entire broadcast screen. The *Stop broadcasts* option lets you stop the transmission of the teacher's screen to the clients.

In the *Clients | Execute* menu, you will find different options for running com-

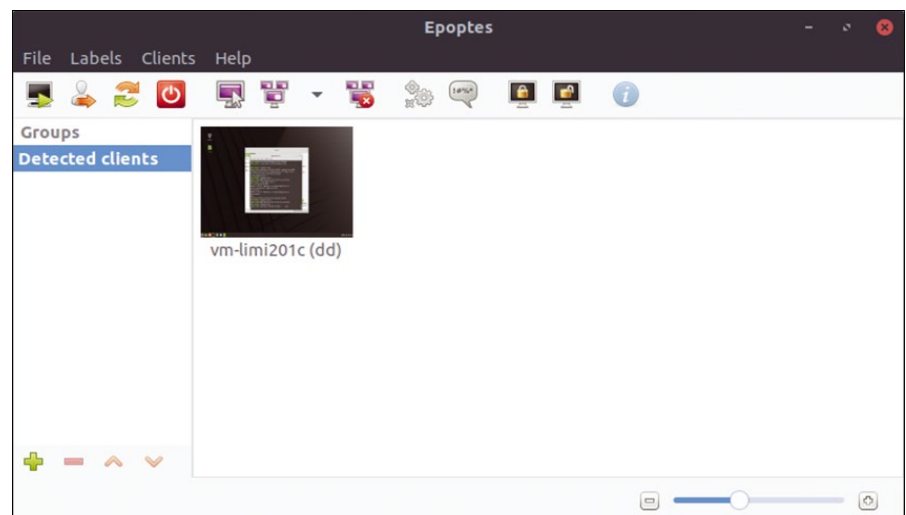


Figure 2: The Epoptes server automatically finds active clients on the intranet.

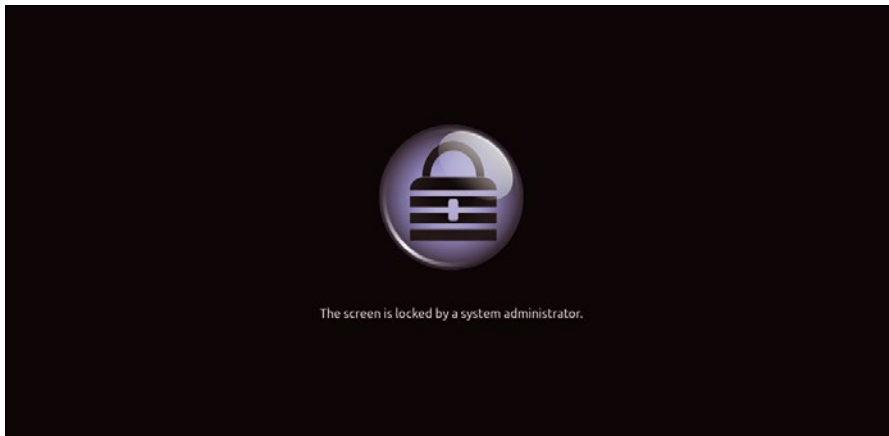


Figure 3: If required, individual clients can also be locked from the Epoptes server computer.

mands on the selected student computers. *Execute command* opens an input line where you can enter a program file name or a URL. Epoptes then starts the corresponding program file on the client or calls the specified URL in the client's web browser.

This option does not work for console commands. Instead, you should select *Open command line* for programs that need to run in the terminal. You can use this option to open a terminal with user or root privileges on the teacher's computer or, alternatively, a terminal with root privileges on the students' computers. In the opened terminal, you then invoke the appropriate commands. Terminal commands executed on the teacher's computer are used, for example, to demonstrate the input of certain commands with parameters and their output when transmitting the teacher's screen to the student computers.

Execute | Send message allows the server computer to send more extensive text messages to the selected clients.

Clients | Restrictions lets the teacher turn audio on or off on the selected student computers. In addition, the teacher can lock the selected clients here if required (this can also be done from the

Computer	Upload bandwidth	Download bandwidth
vm-limi201c	651.9 Mbps	260.2 Mbps

Overall bandwidth statistics
 Average server download: 260.2 Mbps Average server upload: 651.9 Mbps
 Total server download: 260.2 Mbps Total server upload: 651.9 Mbps

Figure 4: Epoptes also displays the bandwidth used from the server to the individual clients if desired.

buttonbar) to prevent students from creating keyboard and mouse input. Once locked, a closed padlock icon then appears on the client screen with a corresponding note (Figure 3). The same menu item unlocks the client computers again.

You can also create groups to organize clients by clicking on the green plus sign at the bottom of the left pane in the main program window. Then use the *Add to group* option from the Client menu to assign clients to the individual groups, which allows you to send group-specific messages and apply actions to individual groups.

Benchmark

Clients | Network Benchmark lets you specify how long you want the implemented network benchmark to test the available bandwidth. When the test is complete, Epoptes displays the result in

a separate window, giving the overall result as well as the upload and download bandwidth for each individual machine (Figure 4).

A Closer Look

Sometimes the size of the client thumbnails displayed on a server computer makes it difficult for a teacher to see what's happening on a student's screen. A slider at the bottom right of the program window (Figure 5) lets you continuously zoom in and out on the right pane of the program window containing the client thumbnails, while leaving the buttonbar, menubar, and Groups pane on the left untouched.

Conclusions

Epoptes offers a comprehensive approach to controlling and managing a Linux-based computer lab. Teachers don't need to read long manuals to handle the configuration and administration. Epoptes can be set up on the server and client computers in just a few steps and then put into operation immediately because using Epoptes is largely intuitive. The range of functions covers all requirements that may arise in the school computer lab setting, without any unnecessary bells and whistles in the user interface. As a result, Epoptes offers a serious alternative to expensive commercial packages. ■■■

Info

- [1] Veyon: <https://veyon.io/en/>
- [2] Epoptes: <https://epoptes.org>
- [3] TigerVNC: <https://tigervnc.org>

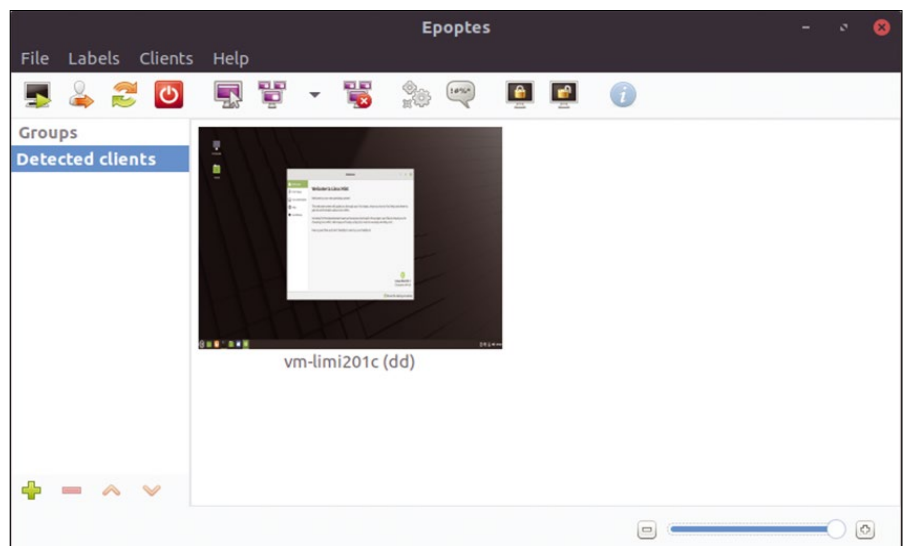


Figure 5: The client thumbnails can be continuously zoomed in or out.

REAL SOLUTIONS *for* REAL NETWORKS

ADMIN is your source for technical solutions to real-world problems.

Improve your admin skills with practical articles on:

- Security
- Cloud computing
- DevOps
- HPC
- Storage and more!

GET IT FAST
with a digital subscription!

6 issues per year!

ORDER NOW

shop.linuxnewmedia.com

Go retrieves GPS data from the komoot app

Plan Your Hike

The hiking and cycling app komoot saves your traveled excursion routes. Mike Schilli shows you how to retrieve the data with Go. *By Mike Schilli*

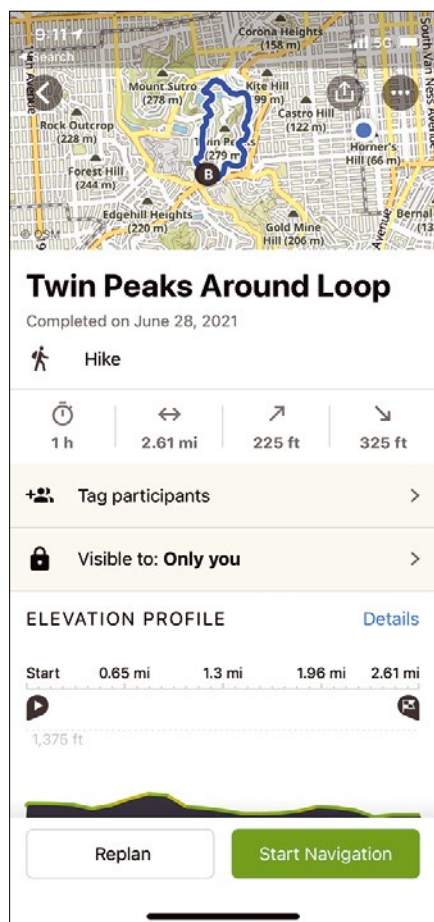
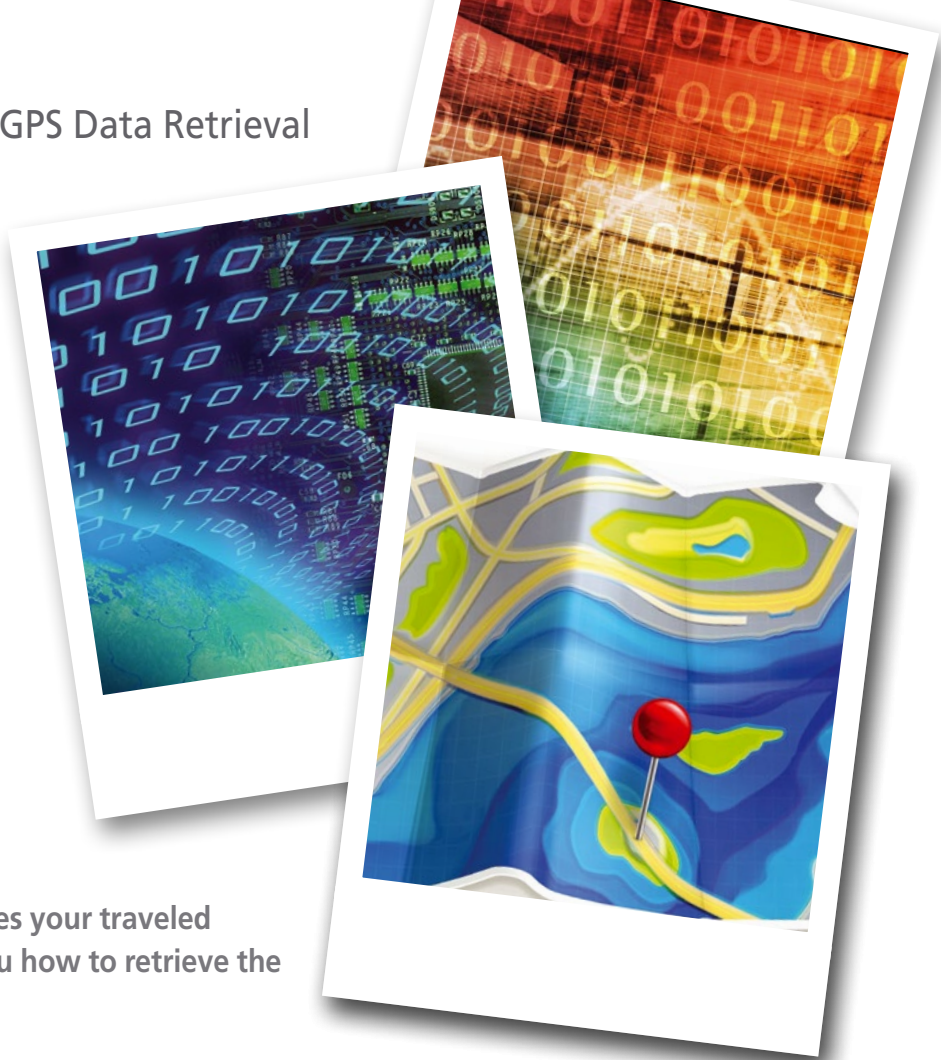


Figure 1: Practical: The app shows the trail and helps navigate it.

As we all know, the past year and a half were completely lost to the pandemic. Because of various lockdowns, there wasn't much left to do in your leisure time in terms of outdoor activity. Playing soccer was prohibited and jogging with a face covering too strenuous. This prompted my wife and me to begin exploring areas of our adopted city, San Francisco. We hiked hidden pathways, previously unknown to us, every evening on hour-long neighborhood walks. To our amazement, we discovered that even 25 years of living in a city is not enough to explore every last corner. We found countless little hidden

stairways, unpaved paths, and sights completely unknown to major travel guides.

Memorizing all the turnoffs for these newly invented, winding city hiking trails is almost impossible, but fortunately a mobile phone can step in as a brain extension here. Hiking apps plan your tours, record your progress during the walk, display traveled paths on an online map, and allow for sharing completed tours with friends (Figure 1). One of the best-known hiking (and biking) apps is the commercial komoot, which is based on OpenStreetMap data and remains free of charge as long as

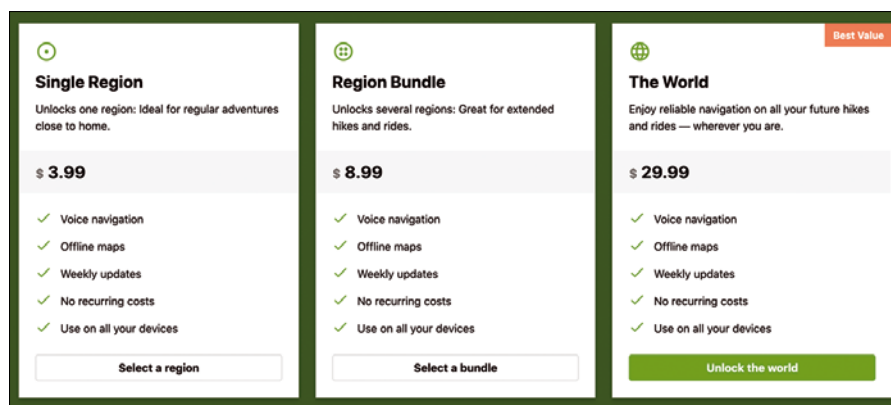


Figure 2: Cost of using komoot.

Lead image © Natalia Natykach, 123RF.com

the user gets invited by another (even new) user and limits themselves to one local hiking area.

Staying Safe

This cheap subscription should suffice for basic needs, but I ended up buying the World Pack. The one-time payment of US \$30 is not exactly a pittance (Figure 2), but I thought I'd support the komoot whippersnappers, helping them to keep the lights on in the datacenter. But then it dawned on me: What happens if komoot goes out of business at some point? What happens to my painstakingly created trails in that case? Fortunately, komoot lets you export the GPS data from the routes you hiked, and you can restore the tours from these data in an emergency. However, seeing that I have dozens of saved trails (Figure 3), manual downloads would be too labor-intensive, not to mention the discipline required to back up new trails on a regular basis – after all, you never know when an app will fail or be abandoned.

For this reason, a program that logs into komoot once a week via a cron job and stores tours that have not yet been saved locally in a backup directory would be just the thing. Komoot does offer an API for script-driven retrieval of user data but not for ordinary people like me.

When I asked about getting an API key, the support team there referred me to a

B2B department that only deals with business partners. But with a bit of tweaking, a web scraper can also scrape the GPS data from the web page. That's exactly what the Go program presented in this article does, based on some reverse-engineering work published on GitHub [1].

Cron Mirror

The scraper negotiates the login process on the website, queries all the tours stored under the account, downloads their JSON data, and converts them into the GPX format supported by all GPS trackers [2]. Now, if komoot were to lose the saved routes for some

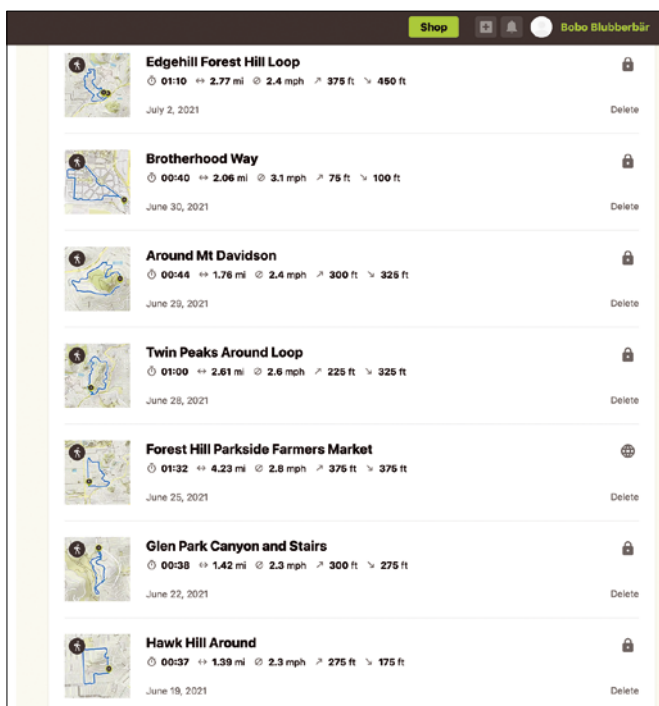


Figure 3: Archived city tours in the komoot app.

Listing 1: kbak.go

```
01 package main
02
03 import (
04     "fmt"
05     "io/ioutil"
06     "log"
07     "os"
08 )
09
10 const saveDir = "tours"
11
12 func main() {
13     kc := NewkColl()
14
15     _ = os.Mkdir(saveDir, 0755)
16
17     err := kc.kLogin()
18     if err != nil {
19         log.Fatalf("Login returned %v", err)
20         return
21     }
22
23     jdata, err := kc.kTours()
24
25     if err != nil {
26         log.Fatalf("Fetching tour ids returned %v", err)
27         return
28     }
29
30     ids := tourIDs(jdata)
31
32     for _, id := range ids {
33         gpxPath := fmt.Sprintf("%s/%s.gpx", saveDir, id)
34         if _, err := os.Stat(gpxPath); err == nil {
35             fmt.Printf("%s already saved\n", gpxPath)
36             continue
37         }
38
39         jdata, err = kc.kTour(id)
40
41         if err != nil {
42             log.Fatalf("Fetching tour %s returned %v", id, err)
43             return
44         }
45
46         gpx := toGpx(jdata)
47
48         fmt.Printf("Saving %s\n", gpxPath)
49         err := ioutil.WriteFile(gpxPath, gpx, 0644)
50         if err != nil {
51             panic(err)
52         }
53     }
54 }
```

Listing 2: creds.go

```

01 package main
02
03 import (
04     "gopkg.in/yaml.v2"
05     "io/ioutil"
06     "os"
07 )
08
09 var credsPath = "creds.yaml"
10
11 func readCreds() map[string]string {
12     creds := map[string]string{}
13
14     f, err := os.Open(credsPath)
15     if err != nil {
16         panic(err)
17     }
18     defer f.Close()
19
20     bytes, err := ioutil.ReadAll(f)
21     if err != nil {
22         panic(err)
23     }
24
25     err = yaml.Unmarshal(bytes, &creds)
26     if err != nil {
27         panic(err)
28     }
29
30     return creds
31 }

```

reason, the tour collection could be restored from the backup because the GPX data represents the hikes as a set of GPS coordinates with timestamps, independently of a particular app.

Listing 1 [3] shows the main Go program. A cron job calls the program once a week, and Go downloads the .gpx files of all tours that exist in a komoot account and saves them in a subdirectory named tours/. To do this, the Go program logs into the komoot web page with a username and password in the `kc.kLogin()` function, uses `kc.kTours()` to query a list of all tours saved under that account, and downloads only the .gpx files of the tours it doesn't already have in a local directory.

The call to `toGpx()` in line 46 converts the JSON data from komoot to the platform-independent GPX tracker format, while the code starting in line 49 saves newly converted data to a .gpx file in the

tours/ subdirectory. The procedure is gentle on the komoot servers and should not bother anyone there, and it helps the user to maintain ownership of routes they created themselves.

Caution, Password

It wouldn't be good style to hard code the username and password into the program, so Listing 2 offloads this to the `creds.yaml` file. You don't want to check the file into a GitHub repo; just keep it local. The web scraper later reads this YAML file before fetching your tour data and uses the komoot email, the associated password, and the numeric user ID stored there in RAM only while the program is running. Sample values are shown in Listing 3. For active use of the program, replace them with the values of your komoot account.

To parse the YAML in `creds.yaml`, Listing 2 retrieves the `gopkg.in/yaml.v2` package from GitHub, which – in the exported `Unmarshal()` function in line 25 – unravels the YAML data structure from Listing 3 and converts it into an internal Go hashmap. The data structure returned as `creds` contains the email address serving as the username for the komoot account used in the `email` key; `password` is unsurprisingly the password, and the `client_id` is the numeric ID of the user account with which komoot accesses the user's data. For an active account, the browser displays the numeric user ID in the URL field after you log in (Figure 4), from where it can be easily copied into the YAML file.

Scraping the Web

The web scraper is running as a compiled Go binary called from the command line. As a browser replacement, it uses the Go *Colly* package, which has been featured in our Snapshot programming series previously [4]. The functions in Listing 4 log in to the komoot account (`kLogin()`, line 23), retrieve a list of tours stored there (`kTours()`, line 48), and extract the GPS data from individual tours (`kTour()`, line 70).

Go does not offer classic object orientation, but with a data structure like `kColl` in line 11, a constructor like `NewkColl()` in line 16, and receivers on the left side of the function names used as methods, it has something very similar, to all extents and purposes. The functions share the data structure, which the caller initializes once at the beginning with the constructor. The constructor in the code at hand stores an instance of the Colly scraper and the `creds` hashtable with the previously obtained user credentials.

The Colly open source scraper library jumps to the `OnRequest()` callbacks before it executes the requested HTTP request with the `Visit()` or `Post()` functions. In Listing 4, `Print()` shows the user which URL is currently being processed and, in some cases, sets special HTTP headers so that the

Listing 3: creds.yaml.sample

```

01 email: "foo@bar.com"
02 password: "hunter123"
03 client_id: "2014254181621"

```

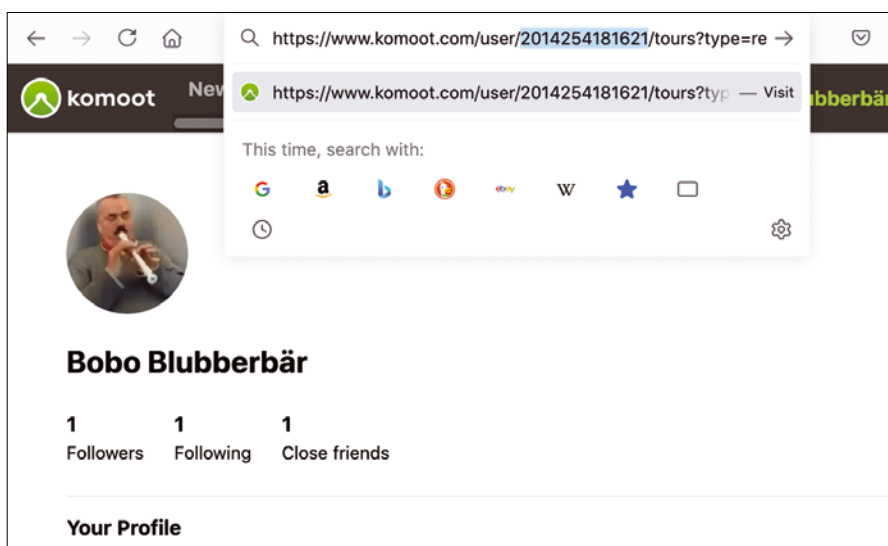


Figure 4: The browser displays the komoot account's numeric user ID.

komoot servers won't return HTML code but easier-to-analyze JSON data.

Tasty Cookies

All three functions share a scraper instance that preserves the cookies set at the beginning of the komoot session, which starts when logging in because the server would not hand out the tour data to simply every Tom, Dick, and Harry. One thing to look out for,

though: The Colly scraper does not replace the callbacks in the `OnRequest()` calls when you set them again later but stacks them up so that, in the code at hand, the third function would not output the URL being accessed once but three times. This is remedied by clones created with `Clone()`, which keep the cookies but reset the callbacks. Figure 5 shows how the program compiles with the listings that will be explained in the

remaining sections of this article. It also illustrates the program's typical output as it finds tours on the server but only downloads them if they are not already available locally.

Cats and Dogs

Komoot's web server delivers both the tour list and the details of individual tours in JSON format because of the headers set in Listing 4. JSON and Go

Listing 4: kfetch.go

```

01 package main
02
03 import (
04     "fmt"
05     "github.com/gocolly/colly/v2"
06 )
07
08 var loginURL = "https://account.komoot.com/v1/signin"
09 var signinURL = "https://account.komoot.com/actions/
    transfer?type=signin"
10
11 type kColl struct {
12     *colly.Collector
13     creds map[string]string
14 }
15
16 func NewkColl() kColl {
17     return kColl{
18         c:    colly.NewCollector(),
19         creds: readCreds(),
20     }
21 }
22
23 func (kc kColl) kLogin() error {
24     c := kc.c.Clone()
25     c.OnRequest(func(req *colly.Request) {
26         fmt.Println("Visiting", req.URL)
27     })
28
29     payload := map[string]string{
30         "email":    kc.creds["email"],
31         "password": kc.creds["password"],
32         "reason":   "null",
33     }
34
35     err := c.Post(loginURL, payload)
36     if err != nil {
37         return err
38     }
39
40     err = c.Visit(signinURL)
41     if err != nil {
42         return err
43     }
44
45     return nil
46 }
47
48 func (kc kColl) kTours() ([]byte, error) {
49     c := kc.c.Clone()
50     toursURL := fmt.Sprintf(
51         "https://www.komoot.com/user/%s/tours",
52         kc.creds["client_id"])
53
54     jdata := []byte{}
55     var err error
56
57     c.OnRequest(func(req *colly.Request) {
58         fmt.Println("Visiting", req.URL)
59         req.Headers.Set("onlyprops", "true")
60     })
61
62     c.OnResponse(func(resp *colly.Response) {
63         jdata = resp.Body
64     })
65
66     c.Visit(toursURL)
67     return jdata, err
68 }
69
70 func (kc kColl) kTour(tourID string) ([]byte, error) {
71     c := kc.c.Clone()
72     tourURL := fmt.Sprintf(
73         "https://www.komoot.com/tour/%s", tourID)
74
75     jdata := []byte{}
76     var err error
77
78     c.OnRequest(func(req *colly.Request) {
79         fmt.Println("Visiting", req.URL)
80         req.Headers.Set("onlyprops", "true")
81     })
82
83     c.OnResponse(func(resp *colly.Response)
84     {
85         jdata = resp.Body
86     })
87
88     c.Visit(tourURL)
89     return jdata, err
90 }

```

```

$ go mod init kbak
$ go build kbak.go kfetch.go tours.go gpx.go creds.go
$ ./kbak
Visiting https://account.komoot.com/v1/signin
Visiting https://account.komoot.com/actions/transfer?type=signin
Visiting https://www.komoot.com/user/2094084181541/tours
Visiting https://www.komoot.com/tour/410473454
Saving tours/410473454.gpx
tours/408002397.gpx already saved
tours/406334507.gpx already saved
tours/405638419.gpx already saved
...
$ ls -l tours
total 2136
-rw-r--r--  1 mschilli  staff   42518 Jul  4 12:36 405638419.gpx
-rw-r--r--  1 mschilli  staff   49867 Jul  4 13:07 406334507.gpx
-rw-r--r--  1 mschilli  staff   66934 Jul  4 14:00 408002397.gpx
-rw-r--r--  1 mschilli  staff   85755 Jul  4 20:02 410473454.gpx
...

```

Figure 5: A typical call to `kbak` retrieves new tours from komoot.

are as compatible as cats and dogs, however, because JSON offers dynamic types with few type checks, while Go insists on precisely defined data structures. To convert deeply nested JSON text into internal Go data structures, programmers need to really coax the language. If you wanted to import JSON into a scripting language such as Python and convert it to GPX later on, you could do so effortlessly with just a dozen lines of code. Go, on the other hand, as you can see from Listing 5 and Listing 6, calls for some pretty exhausting requirements.

Since the komoot data is nested a whopping nine levels deep, the officially prescribed approach for the conversion would be a bit of a pain. It would mean defining the complete data structure with all its levels using struct declarations in Go. If you are wary of this much typing, you can simply define a one-dimensional map with an empty `interface{}` as a placeholder instead, like in line 8 of Listing 5, and make a type assertion to a `hashmap` each time when descending into the depths of the sub-hashmaps (line 38). Go then looks at the

value, concludes that it might be a map, and lets you dig deeper.

Cowboys

This handy, if cowboyish, drilling method is wrapped up in the `drill()` function starting in line 36 of Listing 5. It takes an array of keys, descends into the sub-hashmaps, and outputs the data found at the end of the keychain. Listing 5 collects the numeric IDs of all the tours the user has in their account. These can be tours planned on a virtual drawing board – which the user has added to the map with komoot’s idiosyncratic browser interface – as well as completed hikes that the user has already taken and recorded with the app or even another GPS tracker.

From JSON to GPX

Using the IDs, the main program, supported by the `kTour()` function in Listing 4, can then fetch the data of individual tours from komoot. Finally, Listing 6 converts the fetched JSON data into GPX format, which popular trackers from Garmin and others understand, and which komoot also accepts when you are uploading new tours. This is the restore part of the backup solution. The result of the conversion is shown in Figure 6 in the XML typical of GPX, while Figure 7 shows

Listing 5: tours.go

```

01 package main
02
03 import (
04     "encoding/json"
05 )
06
07 func tourIDs(jdata []byte) []string {
08     var data map[string]interface{}
09
10     err := json.Unmarshal(jdata, &data)
11     if err != nil {
12         panic(err)
13     }
14
15     data = drill(data,
16         []string{"kmtx", "session",
17             "_embedded", "profile",
18             "_embedded", "tours",
19             "_embedded"})
20
21     items :=
22         data["items"].([]interface{})
23
24     ids := []string{}
25
26     for _, item := range items {
27         table :=
28             item.(map[string]interface{})
29         id := table["id"].(string)
30         ids = append(ids, id)
31     }
32
33     return ids
34 }
35
36 func drill(part map[string]interface{}, keys []string)
37     map[string]interface{} {
38     for _, key := range keys {
39         part = part[key].(map[string]interface{})
40     }
41
42     return part
43 }

```

```

<gpx>
  <trk>
    <trkseg>
      <trkpt lat="37.701504" lon="-122.442850">
        <ele>197.4</ele>
        <time>2021-06-29T23:59:19Z</time>
      </trkpt>
    </trkseg>
    <trkseg>
      <trkpt lat="37.701345" lon="-122.442770">
        <ele>197.5</ele>
        <time>2021-06-29T23:59:36Z</time>
      </trkpt>
    </trkseg>
    <trkseg>
      <trkpt lat="37.700509" lon="-122.441501">
        <ele>197.4</ele>
        <time>2021-06-30T00:01:47Z</time>
      </trkpt>
    </trkseg>
    <trkseg>
      <trkpt lat="37.700472" lon="-122.441449">
        <ele>197.5</ele>
        <time>2021-06-30T00:01:53Z</time>
      </trkpt>
    </trkseg>
  </trk>
</gpx>
"405632786.gpx" 1888 lines --0%--

```

Figure 6: The downloaded JSON data converted to GPX format.

that komoot accepts the downloaded data (after conversion to GPX) as a new tour, without any complaints.

However, converting Go data to XML in a prescriptive manner, much like the JSON conversion before it, requires a huge number of type declarations linking its sub-elements in the various layers. Ultimately, the Go code would have to declare the entire GPX syntax. The structure could then be populated with the parsed

data and turned into GPX-compliant XML using the *encoding/xml* package. To make things simpler – if also cowboyish – Listing 6 simply writes the XML as a parameterized text string instead.

Lines 13 through 20 of Listing 6 again use the `drill()` function defined in Listing 5, which works its way into the sub-hashmaps of the unpacked JSON data. Listing 6 saves the geo-coordinates of the tour in line 19 in the `items` array slice after a type assertion confirming that it is an array of unknown content (`[]interface{}`).

Timing

The timestamps of the recorded waypoints are a special case because the JSON format in komoot lists the start time of a tour once, only at the beginning, in RFC 3339 format. Komoot then gives the individual times of the track points each in thousandths of a second relative to the start time. Unfortunately, the GPX format lists the respective absolute time for each waypoint, meaning that Listing 6 has to do some math.

Shop the Shop  shop.linuxnewmedia.com

Discover the past and invest in a new year of IT solutions at Linux New Media's online store.

Want to subscribe?

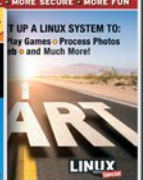
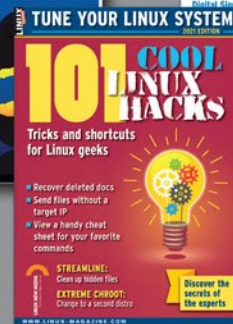
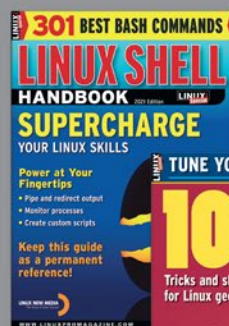
Searching for that back issue you really wish you'd picked up at the newsstand?

 shop.linuxnewmedia.com

DIGITAL & PRINT
SUBSCRIPTIONS



SPECIAL EDITIONS



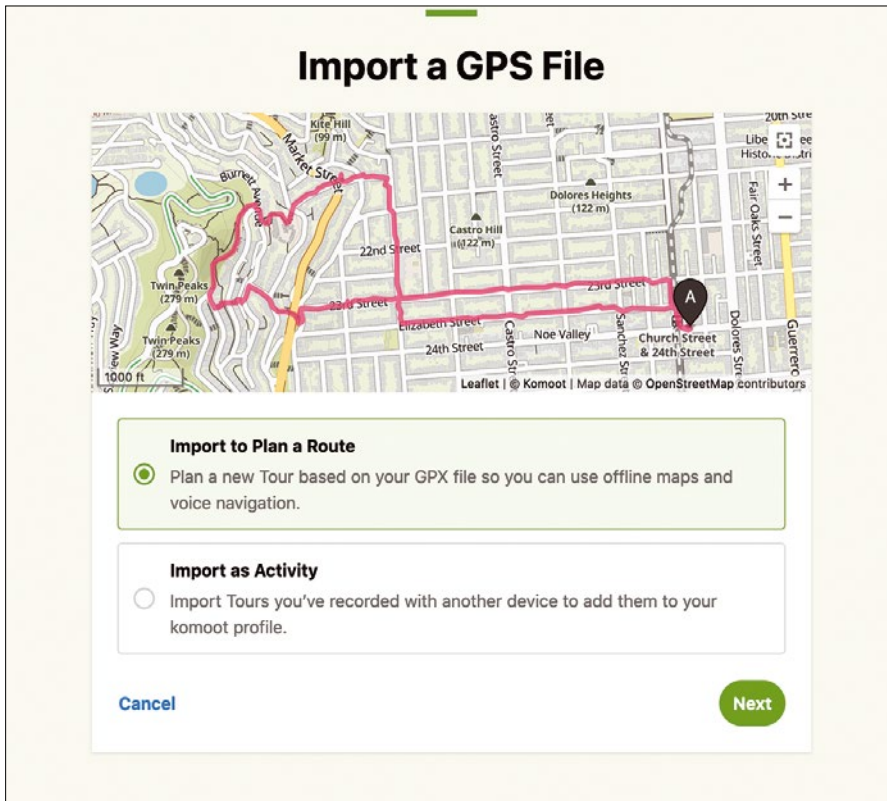


Figure 7: Komoot accepts the reimported GPX file without complaints.

Line 21 reads the time found in the tour start field and converts it to Go's internal time format. While the for loop then rattles through the excavated track points starting in line 27, line 29 divides the time

delta found there in milliseconds by 1,000 and calls `Add()` to add the resulting seconds value to the start time in line 30. The result is the timestamp for the respective waypoint, which line 37 converts back

Listing 6: gpx.go

```

01 package main
02
03 import (
04     "encoding/json"
05     "fmt"
06     "time"
07 )
08
09 func toGpx(jdata []byte) []byte {
10     var data map[string]interface{}
11
12     json.Unmarshal([]byte(jdata),
13         &data)
14     tour := drill(data, []string{
15         "page", "_embedded", "tour"})
16     start := tour["date"].(string)
17
18     coord := drill(tour, []string{
19         "_embedded", "coordinates"})
20     items :=
21         coord["items"].([]interface{})
22     ts, err := time.Parse
23         (time.RFC3339, start)
24
25     if err != nil {
26         panic(err)
27     }
28     xml := "<gpx><trk>"
29     for _, item := range items {
30         pt := item.(map[string]
31             interface{})
32         secs := pt["t"].(float64) /
33             1000.0
34         t := ts.Add(time.Duration(secs)
35             * time.Second)
36         xml += fmt.Sprintf(`<trkseg>
37             <trkpt lat="%f" lon="%f">
38                 <ele>%.1f</ele>
39                 <time>%s</time>
40             </trkpt></trkseg>`, pt["lat"],
41                 pt["lng"], pt["alt"],
42                 t.Format(time.RFC3339))
43     }
44     xml += "</trk></gpx>\n"
45     return []byte(xml)
46 }

```

into RFC 3339 format and injects into the XML as a string.

Added to this are the entries for `lat` (latitude) and `lng` (longitude, but `lon` in XML), which are present as generic `interface{}s` but are converted to a `float64` format by the `%f` placeholder of the `Sprintf()` function via type assertion. The same applies to the `alt` (altitude) above sea level, which goes into the `ele` (elevation) field of the GPX format. What the main program gets back from `toGpx()` is a byte array containing the XML data, which it writes to the backup file on disc and that completes the backup. If you upload the generated `.gpx` file again by way of a test, you will see with growing enthusiasm that komoot recognizes it as a new tour just as if you had walked it: a perfect backup solution.

It must be said that the disadvantage of officially unmaintained web scrapers such as this one is that even the smallest layout changes to the provider's website can break the program and would require small adaptations. You have to live with that. But maybe komoot will eventually take pity and decide to grant hobbyists access to the API and register an OAuth2 client ID. That would be much cleaner. ■■■

Info

- [1] "Get Komoot tour data without API": <https://python.plainenglish.io/get-komoot-tour-data-without-api-143df64e51fa>
- [2] GPX format: https://en.wikipedia.org/wiki/GPS_Exchange_Format
- [3] Listings for this article: <ftp://ftp.linux-magazine.com/pub/listings/linux-magazine.com/252/>
- [4] "Programming Snapshot – Colly" by Mike Schilli, *Linux Magazine*, issue 223, June 2019, [https://www.linux-magazine.com/Issues/2019/223/Data-Scraper/\(language\)/eng-US](https://www.linux-magazine.com/Issues/2019/223/Data-Scraper/(language)/eng-US)

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.



Public Money

Public Code



Modernising Public Infrastructure with Free Software



Free Software Foundation Europe

Learn More: <https://publiccode.eu/>



Simultaneously flashing multiple USB devices

Mass Distribution

If you need to flash the same image to multiple USB media, Popsicle saves time by letting you write in parallel. *By Erik Bärwaldt*

Sometimes you need to copy the same ISO image to several USB storage devices. Since most available tools only let you write to one media at a time, this process can be a time-consuming task. With Popsicle [1], you can flash an image to multiple media simultaneously.

Installation

Originally written for the Ubuntu derivative Pop!_OS, you can find Popsicle in the repositories for Arch Linux and its derivatives. For other distributions, Popsicle's GitHub page [1] provides the source code and detailed installation in-

structions. In addition, Popsicle is available as a Flatpak [2].

Operation

After installation, you will find a launcher in your desktop's Start menu. Popsicle does not require any configuration and supports intuitive use. Upon start-up, a small dialog opens prompting you to select the desired file for flashing using the *Choose an Image* button (Figure 1). After selecting your image, you have the option to validate the image with a checksum in the *Hash*: input field using SHA-256 or MD5.

Clicking on the *Next* button (upper right corner) takes you to the *Select Drives* dialog. Here you will find a list of all USB storage devices (including drive names) found on your computer. Popsicle reads the model names from the volumes on the media and displays them, making it easier to clearly identify the individual USB drives. You can select the desired drives by using the checkboxes to the left of the drive names (Figure 2). If you want to transfer the image to all USB drives connected to the computer, check the *Select all* option. Then, click

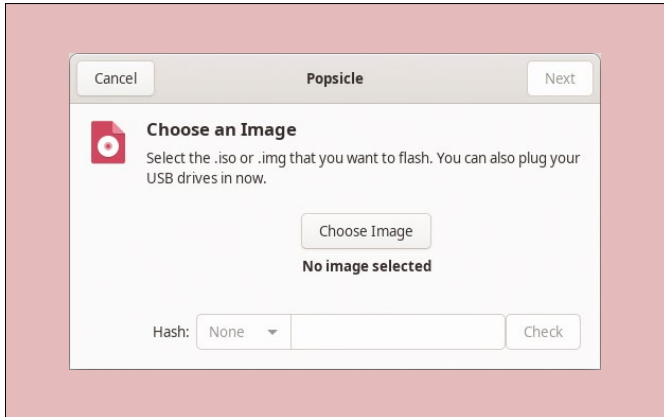


Figure 1: As the first step, Popsicle prompts you to select an image for flashing.

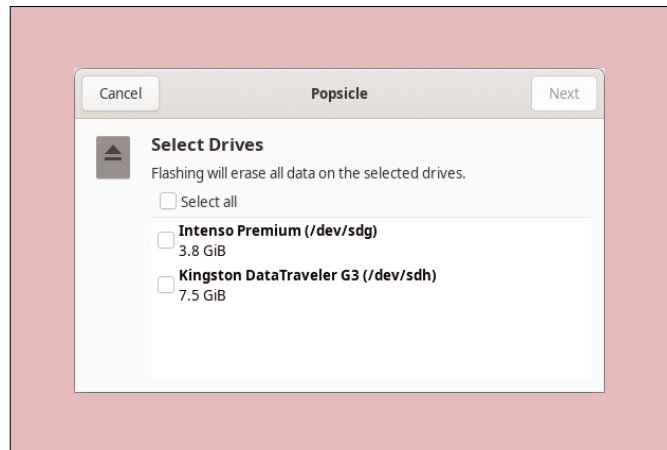


Figure 2: You can select the desired USB drives for flashing from the list or choose *Select all* to write to all drives connected to your computer.

Photo by Lindsay Moe on Unsplash

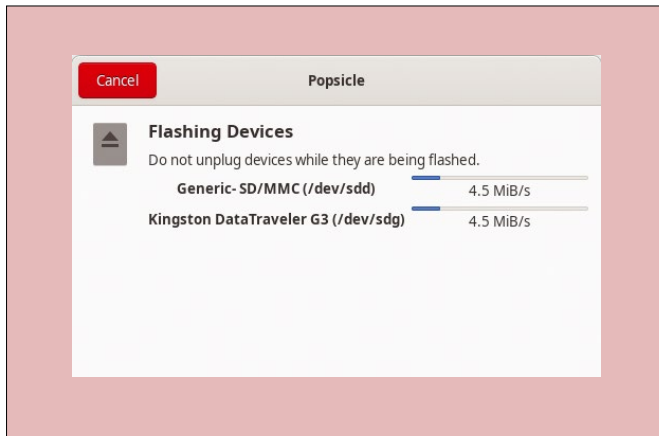


Figure 3: Popsicle shows a separate progress bar for each drive while writing.

on *Next* again, and Popsicle will start writing the data simultaneously to all selected drives, showing a progress bar for each drive (Figure 3).

After completing a run, Popsicle notifies you about the number of successfully written storage devices. If you want to write to more USB drives, select the media with the program window still open and click *Flash Again* (top left corner). Popsicle will then guide you through the appropriate dialogs.

independently determines the supported write speeds and communicates with each drive at the appropriate transfer rate. As a result, significant deviations between the individual media will occur in practice.

Popsicle can also write to SD cards. If you use a reader for different formats on your computer, Popsicle integrates these formats as devices with the name *Generic-SD/MMC* if required, which, for

You can skip the image selection dialog this time; just select the desired media and write the data again.

Versatile

Popsicle lets you use USB drives by different manufacturers and with different capacities when flashing the images, making it a versatile tool. Popsicle in-

example, allows teachers to create identical SD cards for students' Raspberry Pis.

Conclusions

Popsicle puts an end to tedious rounds of copying when you want to write images to multiple USB media. By writing to any number of storage devices simultaneously – even devices connected via hubs – Popsicle achieves the optimal speed for each device.

USB drives from different manufacturers and with different capacities do not pose any problems for Popsicle. Popsicle also lets you write image files to an arbitrary number of microSD and SD cards in parallel. With support for mixed operation with USB sticks, Popsicle makes life far easier for users who frequently have to write images to a large number of removable media. ■■■

Info

- [1] Popsicle: <https://github.com/pop-os/popsicle>
- [2] Installation via Flathub: <https://flathub.org/apps/details/com.system76.Popsicle>

What?!

I can get my issues SOONER?



Available anywhere, anytime!

Sign up for a digital subscription and enjoy the latest articles on trending topics, reviews, cool projects and more...

Subscribe to the PDF edition: shop.linuxnewmedia.com/digisub

Now available on ZINIO: bit.ly/Linux-Pro-ZINIO





MakerSpace

Build a saltwater battery Clean Power

Rechargeable lithium batteries are expensive, and manufacturing them damages the environment. Saltwater batteries offer a cheaper and greener approach to storing energy. *By Martin Mohr*

Author

Martin Mohr experienced the complete development of modern computer technology live. After his studies, he mainly developed Java applications. The Raspberry Pi rekindled his old love of electronics.

The supply of lithium is finite, mining lithium pollutes the environment, and the batteries are expensive. None of this applies to saltwater batteries. Common household utensils are all you need to build one for yourself.

Energy turnaround and environmental benefits are some of today's central topics. However, it is not just a matter of making energy production more green

but of looking at all aspects of the energy industry. It makes no sense to store clean solar power in lithium-ion batteries if producing them causes massive environmental damage. Recycling these batteries is also problematic. For stationary use, the saltwater battery offers a truly ecological alternative.

Everyone is shaped by their knowledge and experiences. When I first heard about storing energy in saltwater batteries, my first thought was: That can't possibly work. What about the electrochemical voltage series and the anode and cathode made of the same material? No, that's really impossible. A simple experimental setup quickly shed light on the matter. Spoiler alert: It worked!

Test Setup

The rather simple experimental setup comprises three jam jars containing pencil leads wrapped in handkerchiefs. The electrolyte in the jars is a solution of Glauber's salt [1], often used as a home remedy.

If you don't have any Glauber's salt in the house, you can find some online [2]. Glauber's salt is absolutely harmless compared with lithium. The only thing it sometimes causes, when injected, is diarrhea, which is why it serves as a natural laxative in the materia medica.

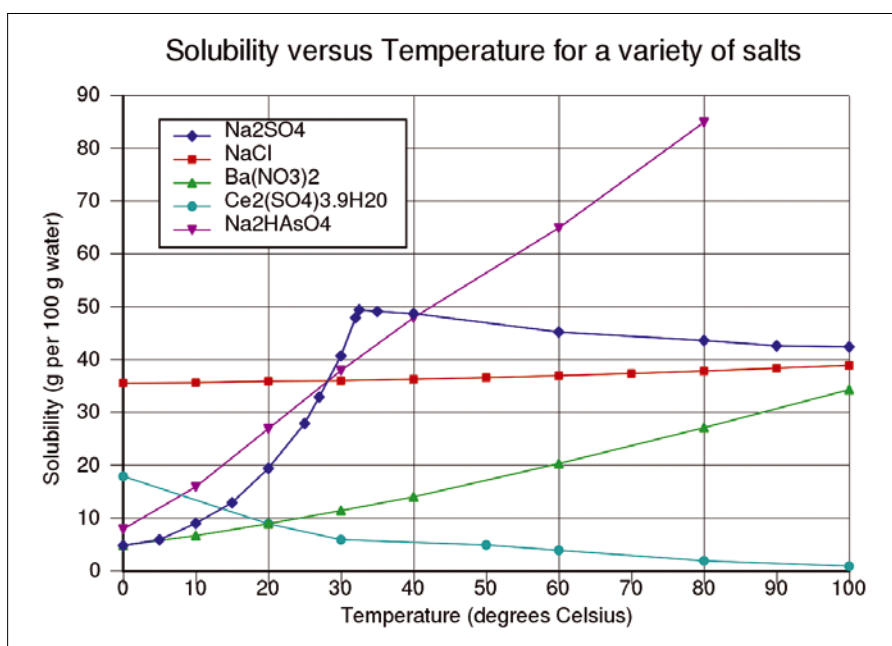


Figure 1: The amount of salt that can be dissolved in water. The blue line is Glauber's salt. (GRAPH: Wikimedia, TheKiteGuy, CC0 1.0 [4])

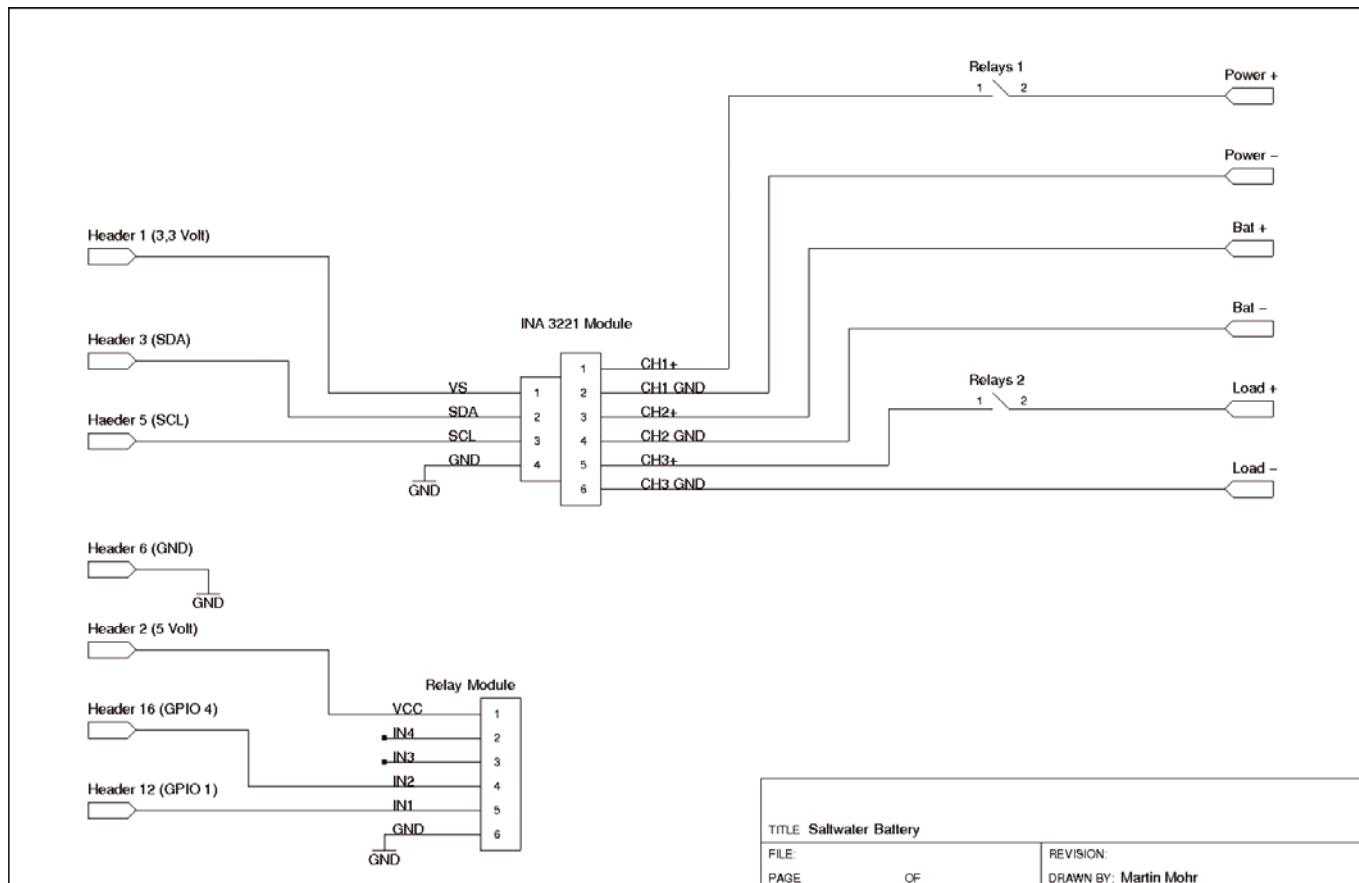


Figure 2: Circuit diagram for the test setup for measuring the charging and discharging cycles.

To activate many ions to transport the charge, you need to dissolve as much salt as possible in the water. As Figure 1 shows, about 20g of Glauber's salt can be dissolved in 100ml of water at a room temperature of 20°C [3]. The jam jars each happen to hold about 100ml of water, so I dissolved 20g of Glauber's salt in each.

The job of charging and discharging the battery is handled by a program. A relay module [5] switches the connections to the power supply unit and consumer. To measure the current, I used an INA3221 module [6]. In this experiment, I only need one of the three measuring inputs, but the module was already available in my tinkering box. The circuit diagram for the test setup is shown in Figure 2, and the complete setup is shown in Figure 3.

Setup

For the project, I used the latest generation Raspberry Pi OS Lite. You can use RPI Imager [7] to transfer the system to the memory card.

The INA3221 module is controlled over the I2C interface, which you need to enable by typing:

```
sudo raspi-config
```

In the interface, navigate to *5 Interfacing Options | P5 I2C* and select *YES*. Next, install Java and some additional tools on the Raspberry Pi:

```
$ sudo apt update
```

```
$ sudo apt upgrade
```

```
$ sudo apt install \
  default-jdk i2c-tools wiringpi
```

The experimental setup is now ready to test. To begin, you scan the I2C bus to find the devices (Listing 1). As expected, the INA3221 module resides on address 0x40.

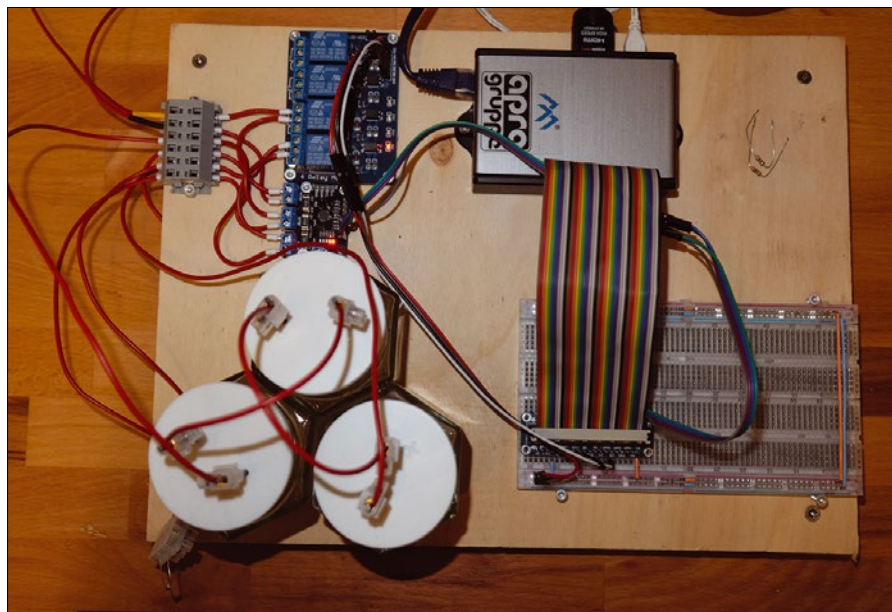


Figure 3: The test setup for the saltwater battery.

Listing 1: Scanning the I2C Bus

```
$ i2cdetect -y 1
    0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
10:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
20:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
30:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
40: 40  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
50:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
60:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
70:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
```

Use the commands

```
$ gpio mode 1 out
$ gpio mode 4 out
$ gpio write 1 0
$ gpio write 1 1
$ gpio write 4 0
$ gpio write 4 1
```

to make sure the relay is connected correctly. The first two lines set the GPIOs to output mode. The next two lines activate relay 1 for loading and switch it off again. The final two lines do the same for relay 2. The switching logic of the relay module is inverted, but this need not cause any confusion.

Measuring Program

To calculate the charge in this do-it-yourself battery, you need a small program that measures the current and time during charging and discharging. If the discharge current reaches 0mA, the battery is completely discharged.

When charging, things are a bit trickier, because you don't know the end-of-charge voltage of the battery, so you simply charge for a certain time at 12V. In principle, you should charge with a voltage at which no gases escape from the battery. If your set-up starts to bubble, you need to reduce the charging voltage.

The program for the test setup is coded in Java. The `i2c-detect` and `gpio` tools control the GPIO and read out the measured values; the Java program starts these tools as sub-processes (Listing 2).

The `switchBytes()` method (line 9) plays a special role: It swaps the high and low bytes and interprets the most significant bit as a sign. If the number is negative, the program forms the two's complement and sets the sign. These hacks are needed because the INA3221

module delivers the measured values in a machine language format that Java does not understand.

The measurement program is in the `go()` method (line 40). The first for loop (lines 49-53) measures the current once per second when the battery is charging, giving the amount of energy stored in the battery. The second for loop (lines 58-64) discharges the battery with a load resistor. When discharging, the program outputs the current voltage of the battery with each measurement. Here, too, it determines the amount of energy extracted.

At the end of the cycle, the script outputs the measured values. To find out a little about the characteristics of the battery, I discharged it with different load resistances (Table 1).

Not surprisingly, the battery built from household objects does not have particularly good efficiency. Much of the charge energy is lost, but it also clearly shows that the battery does store energy. Moreover, it

Table 1: Discharge Results

Load Resistance (ohms)	Stored (mA)	Discharged (mA)
10	13,207	-395
30	13,798	-440
100	14,358	-462
100	14,370	-483
100	14,374	-507

Listing 2: Controlling and Measuring

```
01 import java.util.Scanner;
02 public class SaltBattery {
03     public void setup() throws Exception {
04         new ProcessBuilder( "/usr/bin/gpio","mode","1","out" ).start();
05         new ProcessBuilder( "/usr/bin/gpio","mode","4","out" ).start();
06         loadOff();
07         unLoadOff();
08     }
09     public int switchBytes(int input){
10         input =((input & 0x00ff) << 0x08)|((input >> 0x08)&0x0ff);
11         int sign=1;
12         if (0<(input&0x8000)){
13             sign=-1;
14             input=((~input)&0xffff)+1;
15         }
16         return sign*input;
17     }
18     public void loadOn() throws Exception {
19         new ProcessBuilder( "/usr/bin/gpio","write","1","0" ).start();
20     }
21     public void loadOff() throws Exception {
22         new ProcessBuilder( "/usr/bin/gpio","write","1","1" ).start();
23     }
24     public void unLoadOn() throws Exception {
25         new ProcessBuilder( "/usr/bin/gpio","write","4","0" ).start();
26     }
27     public void unLoadOff() throws Exception {
28         new ProcessBuilder( "/usr/bin/gpio","write","4","1" ).start();
29     }
30     public double measureCurrent() throws Exception {
```

Listing 2: Controlling and Measuring (continued)

```

Process p = new ProcessBuilder("i2cget", "-y", "1", "0x40", "0x03", "w").start();
Scanner s = new Scanner(p.getInputStream());

return switchBytes(Integer.decode(s.next())/20;
}

public double measureVoltage() throws Exception {
    Process p = new ProcessBuilder("i2cget", "-y", "1", "0x40", "0x04", "w").start();
    Scanner s = new Scanner(p.getInputStream());
    return switchBytes(Integer.decode(s.next())/1000;
}

public void go() {
    double sumCurrentUnload=0;
    double sumCurrentLoad=0;
    double current=0;
    int measurementSteps=600; // Sec
    try {
        setup();
        System.out.println("Start battery charge");
        loadOn();
        for (int i=1; i<measurementSteps ; i++) {
            current = measureCurrent();
            sumCurrentLoad +=current;
            Thread.sleep(1000);
        }
        loadOff();
        System.out.println("Start battery discharge");
        Thread.sleep(2000);
        unLoadOn();
        for (int i=1; i<measurementSteps ; i++) {
            current = measureCurrent();
            sumCurrentUnload += current;
            System.out.println("Voltage:"+measureVoltage());
            if(current==0.0) break;
            Thread.sleep(1000);
        }
        unLoadOff();
    } catch (Exception e) {
        System.out.println(e);
    }
    System.out.println("Charge:"+sumCurrentLoad+" mAs");
    System.out.println("Discharge:"+sumCurrentUnload+" mAs");
} // go

public static void main(String[] args) {
    SaltBattery b=new SaltBattery();
    b.go();
} // main
} // class

```

looks like the capacity increases with the number of charging cycles; noticeably, the battery delivers slightly more power with less discharge.

This small test series shows that a saltwater battery works in principle. In the next section, I take a look at a commercially available saltwater battery.

Commercial Battery

Some manufacturers specialize in the production of these environmentally friendly energy storage systems. BlueSky Energy [8], for example, has a product – the Greenrock system – for both private and industrial use in its portfolio.

The storage systems are intended for stationary use only and are ideal for storing the energy generated by solar systems for personal consumption. As Figure 4 shows, a saltwater battery can certainly compete with the other electric storage systems on the market. In only two categories does it perform worse than a lithium-ion battery: energy density and C rating, which indicates how quickly the energy can be retrieved.

The energy density is not as good as that of lithium-ion batteries but is still as good as that of a lead-acid battery. Because of the lower energy density, the batteries are larger and heavier than lithium-ion batteries with the same capacity, but this is not really a problem when used indoors in a stationary environment.

With respect to the C rating, the saltwater battery turns out to be more of a marathon runner: It delivers a constant amount of energy over a long period of time, which is the application scenario in domestic use; however, it is not suitable for delivering high currents for short periods of time.

Saltwater batteries have several advantages over lithium-ion batteries. Apart from environmental friendliness and low fire hazard, the availability of resources plays a major role.

Discussions about electric vehicles have revealed that lithium mining causes considerable environmental damage. Recycling it is certainly a good idea, but it requires large amounts of energy and is expensive. Saltwater batteries fare far better in this respect; they cause virtually no environmental damage and are easy to recycle. The components of the saltwater battery exist on

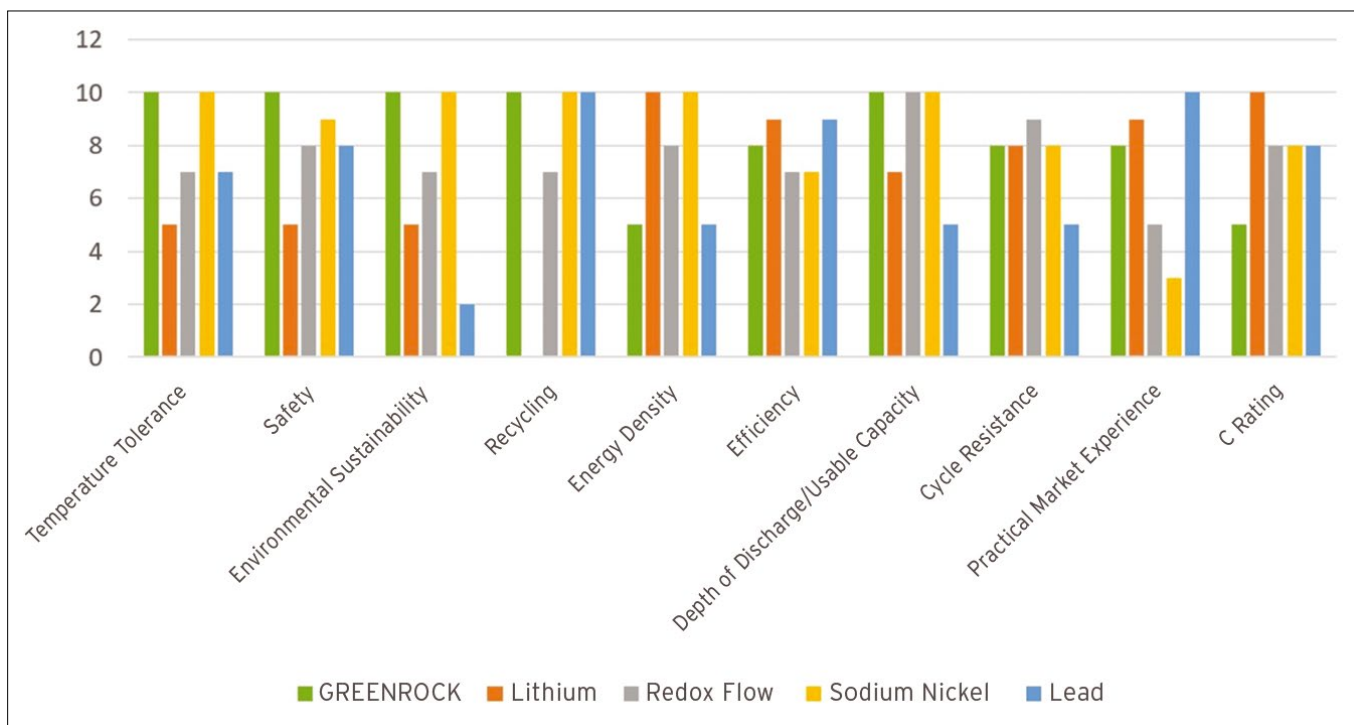


Figure 4: When comparing electricity storage systems, saltwater batteries perform quite well.

Earth in large quantities, whereas lithium does not.

A really serious issue that speaks against the use of lithium-ion batteries in buildings is their combustibility. You can find a number of YouTube videos that show the kind of fireworks that burning batteries of this type can cause. A saltwater battery, on the other hand, remains unfazed, even after prolonged exposure to heat [9].

Conclusions

The simple test setup in this article demonstrates that an energy storage device can be built with the simplest household objects. The Raspberry Pi helped keep the charging and discharging cycles the same and measured the charge. Although this scenario was merely an experimental setup that cannot be used in production, it works, and it can be modified for further experiments. One interesting question would

be, for example, what level of self-discharge occurs.

Saltwater batteries have long been a mature and usable technology. They are well suited as energy storage for photovoltaic systems and, in contrast to lithium-ion

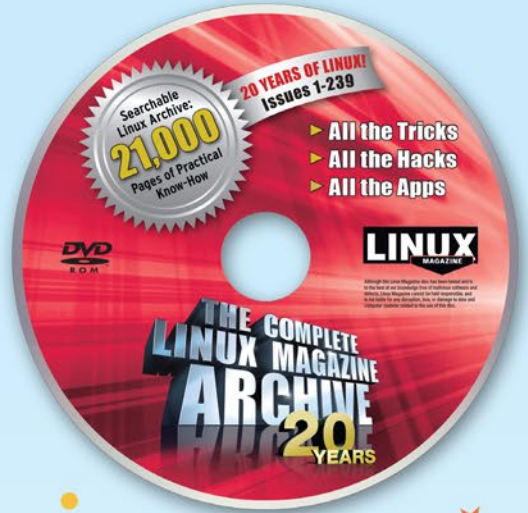
batteries, they are unproblematic. If you are planning to install a photovoltaic system soon, check whether you really want to have lithium in your house or whether you prefer to rely on a safe and environmentally friendly battery technology. ■■■

Info

- [1] Glauber's salt: https://en.wikipedia.org/wiki/Sodium_sulfate#History
- [2] Salt source: https://www.amazon.com/Laboratory-Grade-Sodium-Sulfate-Powder-Anhydrous/dp/B07JDH9TNF/ref=sr_1_4
- [3] Diagram on saline solubility: <https://commons.wikimedia.org/w/index.php?curid=51036796>
- [4] CC0 1.0 Universal (CC0 1.0) Public Domain Dedication: <https://creativecommons.org/publicdomain/zero/1.0/deed.en>
- [5] Relay module: https://www.amazon.com/Gikfun-Channel-Optocoupler-Arduino-Raspberry/dp/B00Q9YC0LS/ref=sr_1_51
- [6] INA3221: https://www.amazon.com/Lopbinte-INA3221-Triple-Channel-Current-Voltage/dp/B092HT8VP8/ref=sr_1_4
- [7] Raspberry Pi Imager: <https://www.raspberrypi.org/software/>
- [8] BlueSky Energy: <https://www.bluesky-energy.eu/en/home-2/>
- [9] Incombustible saltwater battery: <https://www.youtube.com/watch?v=HXmZW8Wnvko>

20 YEARS

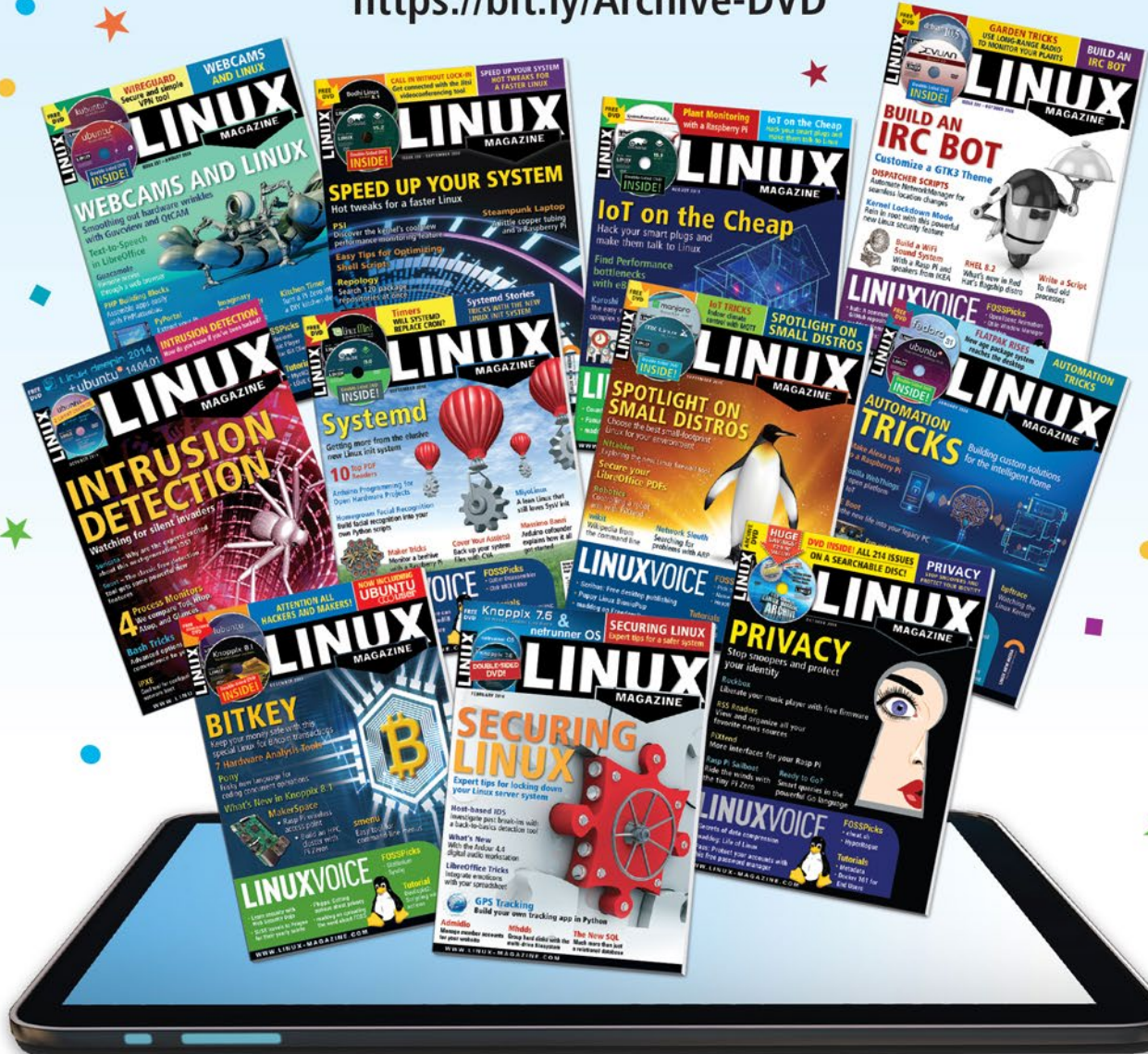
LINUX MAGAZINE

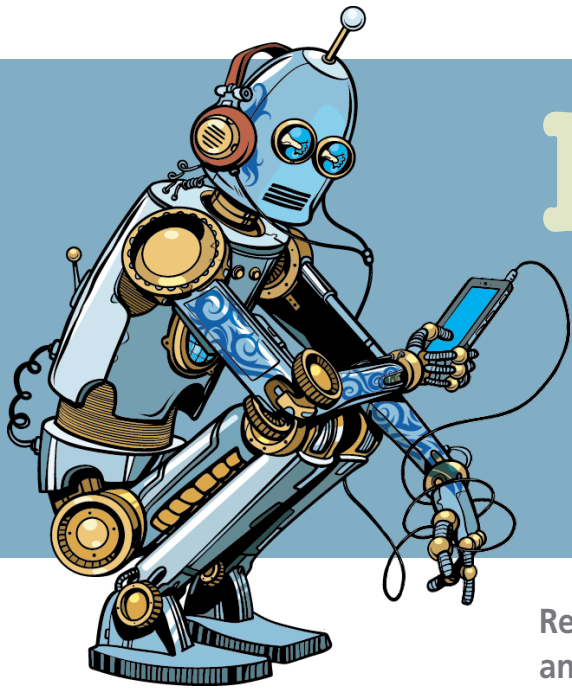


LINUX MAGAZINE 20 YEAR ARCHIVE DVD

ORDER NOW!

<https://bit.ly/Archive-DVD>





MakerSpace

Cross-platform game and app development for new programmers

Easy Does It

Ren'Py helps you create Android, Linux, macOS, Windows, and HTML5 games and apps. *By Pete Metcalfe*

Although you can find some excellent cross-platform development tools, many of these tools come with a steep learning curve for new programmers. Ren'Py [1] is a visual novel engine that has been around for more than 10 years, and it is one of the easiest packages to learn for game and app development.

The nice thing about Ren'Py is that you don't need any previous programming experience. Ren'Py uses a simple "screen language" that allows you to add backgrounds, images, character dialogs, and menus. For more complex requirements, Ren'Py supports inline Python and Python blocks.

The Ren'Py software development kit (SDK) is supported on Linux, macOS, and Windows, with final applications that can be built to run on Android, iOS, Linux, macOS, and Windows and in browsers with HTML5 (Figure 1).

In this article, I introduce Ren'Py with three examples. The first example will be the start of a visual novel. The second example will be a tourist guide with graphic menus, and the final example uses Python to create a dashboard screen that shows dynamic values and bars.

Getting Started

If you want to play with Ren'Py on a Raspberry Pi or an Ubuntu/Debian system, you can load a lightweight installation with:

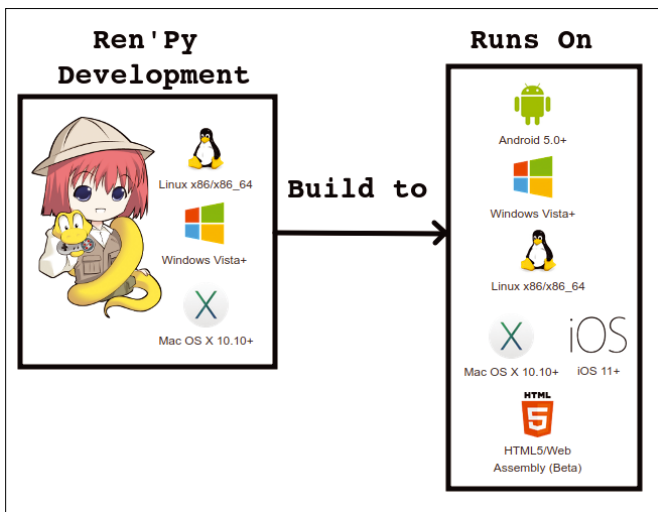


Figure 1: Ren'Py development and build options.

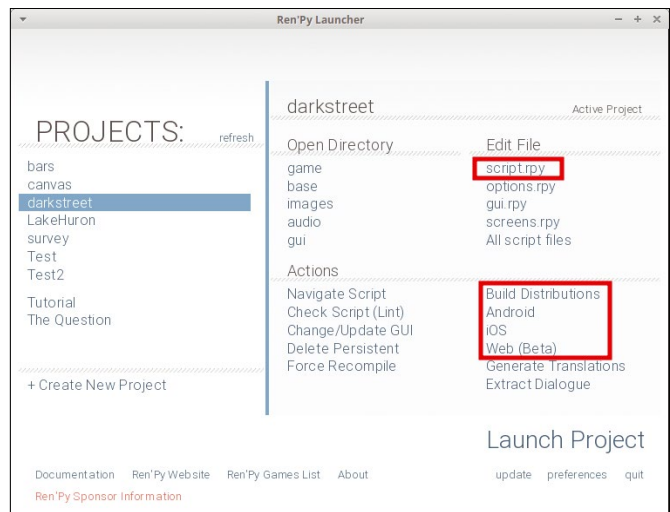


Figure 2: Ren'Py interface.

```
sudo apt-get install renpy
```

The apt-get version of Ren'Py, however, does not have any tutorials or extra build features, so it is recommended that you go to the Ren'Py site for the complete installation directions [2].

The Ren'Py user interface creates new projects with all of the required files and supports all of the different build options (Figure 2). The script.rpy text file contains all the logic for an application.

Visual Novel

A visual novel is sort of like a comic book that can have multiple paths and story lines that users select as they work their way through the novel. The first step is to define some characters and background images. Creating character drawings from scratch can be a lot of work; luckily, you can find some open source solutions that can help out.

Character Creator [3] is an excellent free website you can use to generate male or female head, torso, and full-body images. It also supports a variety of facial expressions (Figure 3). The character image files and all background images are stored in the project's game/images directory.

The next step is to use a standard text editor and add screen language code to the script.rpy file. Figure 4 and Listing 1 show the code required to display a background with a character and some dialog. Lines 3 and 4 define two characters cop and me. These definitions are used to output or show dialog text. Ren'Py uses labels to jump between code segments. The application begins at the start label (line 10).

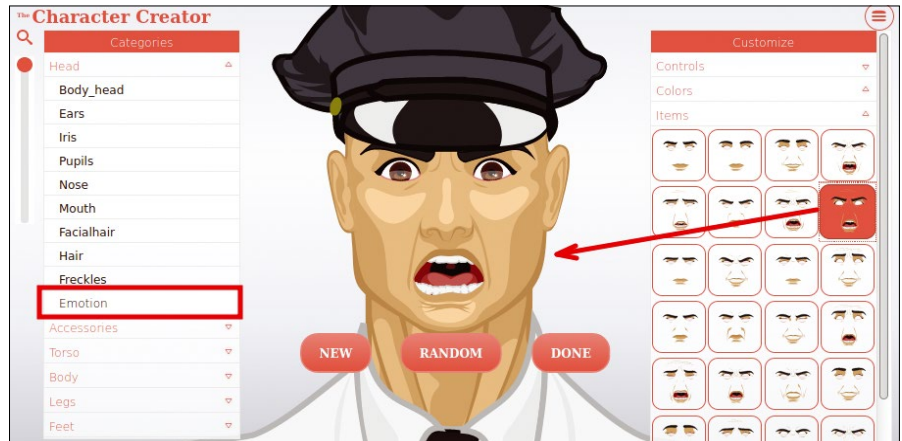


Figure 3: Character Creator generates character emotions.



Figure 4: Start of a visual novel.

Images in the game/images directory can be displayed simply with:

```
show image_name at position
```

In this example, cop_head.png is displayed at the center of the screen (line 15) by:

```
show cop_head at truecenter
```

Images can also be resized, rotated, moved, or adjusted. In line 6, the background (darkstreet.jpg) is sized to fill the screen. Dialog is shown by referencing the character and then the dialog text (lines 19-20).

Figure 5 shows the next phase of the story. The policeman is hidden with the hide statement (line 22) and an image of

Listing 1: Visual Novel

```
01 # darkstreet - script.rpy
02
03 define cop = Character("Cop")
04 define me = Character("Me")
05
06 image darkstreet = im.Scale("darkstreet.jpg", config.
    screen_width, config.screen_height)
07
08 # The game starts here.
09
10 label start:
11
12     # Show a background.
13     show darkstreet
14
15     show cop_head at truecenter
16
17     # These display lines of dialogue.
18
19     cop "Hey Kid! Stop right there!\n
20         Why do you have blood on your hands?"
21
22     hide cop_head
23     show me_crying_head at truecenter
24
25     me "It wasn't me...I don't know what happened."
26
27
28     return
```

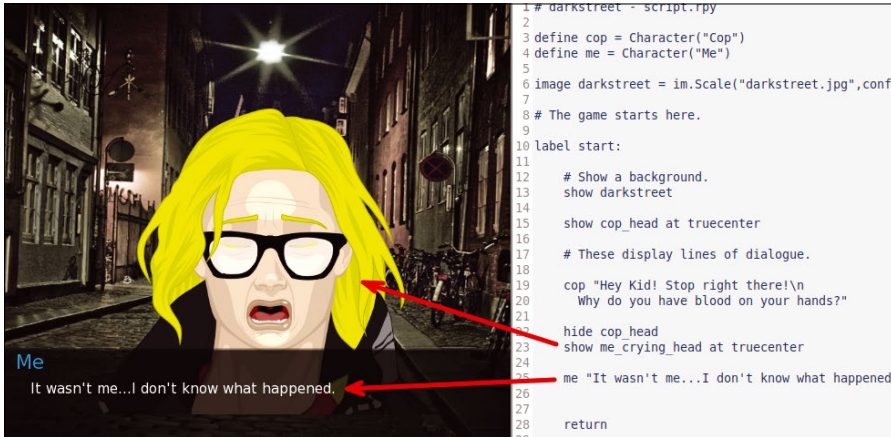


Figure 5: Second character in the visual novel.

a crying girl (me_crying_head) is shown with some dialog (line 25). The use of hide and show statements allow you to present different characters and backgrounds.

At the end of the story, the application finishes with a return statement (line 28).

Tourist Guide

Most apps and games require menus that allow multiple branches or outcomes. A Ren'Py menu is created simply with a menu: statement. Each menu item is defined with button text and a jump statement, which is like an old-school

Basic GOTO statement. Each jump has a label to which the code links.

Figure 6 shows the start of a tourist guide for the Bruce Peninsula [4]. A menu is called at the start of the program. Each menu item jumps to a specific section of the code with a label. Within a subsection, (e.g., the beach), a new background and text are shown. After the user sees this information, a jump start statement puts the user back to the main menu.

For smaller applications (Figure 7), you can put the submenus, display logic, and Python code directly in the menu: logic.

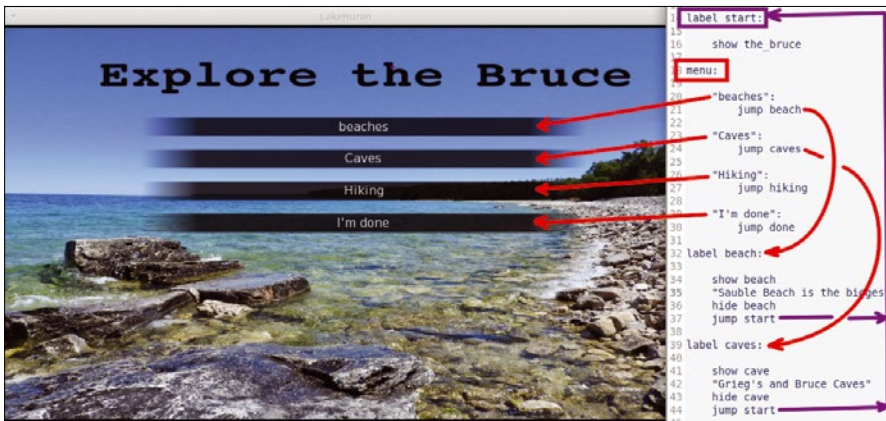


Figure 6: Tourist guide with menus.



Figure 7: Submenus and logic within menu options.

Dynamic Screens

Ren'Py supports custom screen layouts that can have labels, text, buttons, and bar charts. In a visual novel, a custom screen could be a small box at the top of the application that shows items like inventory, money, or percent complete.

For the next example, I use Python code with Ren'Py to find some PC hardware readings and present the information on a large custom screen.

The Linux sensors [5] command can be used to show current hardware information:

```

$ sensors
dell_smm-virtual-0
Adapter: Virtual device
Processor Fan: 2721 RPM
CPU: +46.0°C
Ambient: +39.0°C
...
    
```

By adding some Bash/Awk code, the CPU and ambient temperature values can be extracted:

```

# use show Bash/Awk code to get temps
$ sensors | grep CPU | z
awk '{ printf "%d\n", $2}'
46
$ sensors | grep Ambient | z
awk '{ printf "%d\n", $2}'
39
    
```

Python calls Bash scripts with the subprocess.check_output method, which is part of the subprocess library (Listing 2).

Listing 3 shows the code required for the CPU statistics application (Figure 8). The screen cpu_data(): statement creates a user interface (line 3) that can contain both Python blocks and display elements.

The Python block (lines 5-9), start with a python: statement, and the succeeding lines are indented per the Py-

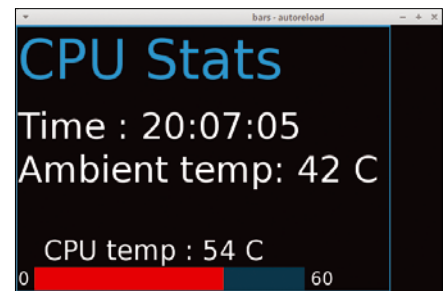


Figure 8: CPU stats screen.

thon standard. CPU and ambient temperature variables (lines 8-9) are created with the `subprocess.check_output()` call and the earlier Bash sensors statements.

For this screen, a `frame:` (line 11) is used to group the elements together. Vertical boxes (`vbox:`) and horizontal boxes (`hbox:`) are used for further groupings. Python variables are inserted into text output strings with square brackets (lines 16-18). A horizontal bar (line 23) shows the ambient temperature (`atemp`) with a range of 0-60.

The application begins at `label start:` (line 32) by showing the `cpu_data` screen (line 35). An inline Python

statement (line 40) pauses the execution for two seconds within a `while` loop. The screen logic is refreshed after each `renpy.pause()` iteration.

Building Applications

The Ren'Py IDE has a *Launch Project* button that allows testing in the native operating system environment. The IDE's *Build* option allows you to create a variety of different packages (Figure 9).

The HTML5 build is still in beta; it appears to work well for standard visual novel apps, but I found that it had issues with

some Python library calls. The HTML5 build creates a separate directory structure for the Ren'Py application that needs to be mapped into your web server configuration.

If you are looking to do some simple testing, a Python standalone web server (Figure 10) can be run from the project's web directory:

```
# Run a python standalone server ↗
on port 8042
python3 -m http.server 8042
```

Listing 2: subprocess.check_output

```
$ python
Python 2.7.15+ (default, Oct 7 2019, 17:39:04)
[GCC 7.4.0] on linux2
>>> import subprocess
>>> subprocess.check_output("sensors | grep CPU | awk '{
printf \"%d\\\" , $2}'" , shell=True)
'46'
>>>
>>> subprocess.check_output("sensors | grep Ambient | awk '{
printf \"%d\\\" , $2}'" , shell=True)
'39'
```

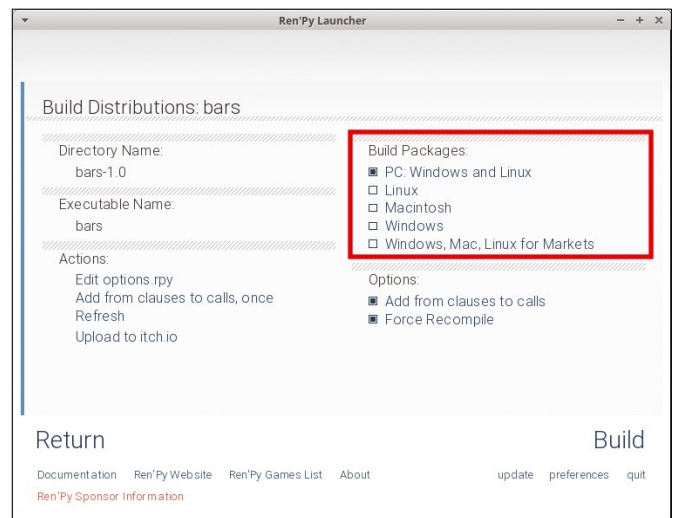


Figure 9: Ren'Py build options.

Listing 3: Dynamic CPU Stats Screen

```
01 # script.rpy - create a Ren'Py screen that shows CPU sensor
    information
02
03 screen cpu_data():
04
05     python: # Use Python to get sensor variables
06         now = datetime.now()
07         nowtime = now.strftime("%H:%M:%S")
08         ctemp = subprocess.check_output("sensors | grep CPU
    | awk '{ printf \"%d\\\" , $2}'" , shell=True)
09         atemp = subprocess.check_output("sensors | grep
    Ambient | awk '{ printf \"%d\\\" , $2}'" ,
    shell=True)
10
11     frame: # create a frame with text, values and a bar
12         has vbox
13         label "CPU Stats" text_size 120
14
15         vbox:
16             text "Time : [nowtime]" size 80
17             text "Ambient temp: [atemp] C \n" size 80
18             text " CPU temp : [ctemp] C " size 60
19         hbox:
20             vbox:
21                 text "0 " size 40
22             vbox:
23                 bar value atemp range 60 xalign 50 yalign
    50 xmaximum 600 ymaximum 50 left_bar
    "#FF0000"
24             vbox:
25                 text " 60 " size 40
26
27     init python:
28         # Define Libraries and any system variables
29         import subprocess
30         from datetime import datetime
31
32     label start:
33
34         # Start with Weather screen
35         show screen cpu_data()
36
37         define cycle = "True"
38         # Cycle every 2 seconds
39         while cycle == "True" :
40             $ renpy.pause(2)
41
42         return
```

Final Thoughts

If you're looking to create some simple cross-platform visual presentation apps, Ren'Py is a good fit, it is super easy to

learn, and doesn't require strong programming skills. I found the first-time call-up for Android was longer than expected, but I was impressed with the final result.

Ren'Py supports simple screens that can have dynamic bars and text; however, if you're looking to incorporate gauges or line charts. Ren'Py probably isn't the best fit. ■■■

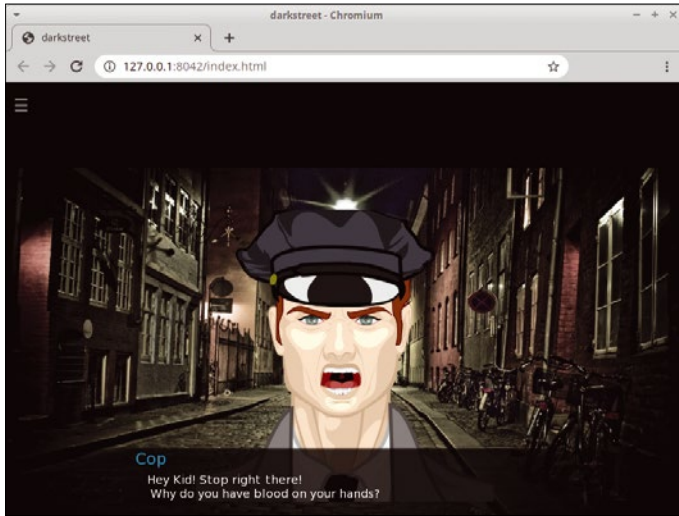


Figure 10: Ren'Py web application.

Info

- [1] Ren'Py: <https://www.renpy.org>
- [2] Download Ren'Py: <https://www.renpy.org/latest.html>
- [3] Character Creator: <https://charactercreator.org>
- [4] Bruce Peninsula, Ontario: <https://visitbrucepeninsula.ca>
- [5] Im_sensors (sensors) documentation: https://wiki.archlinux.org/title/Im_sensors

Author

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



IT Highlights at a Glance





Too busy to wade through press releases and chatty tech news sites? Let us deliver the most relevant news, technical articles, and tool tips – straight to your Inbox.

[Linux Update](#) • [ADMIN Update](#) • [ADMIN HPC](#)

Keep your finger on the pulse of the IT industry.

ADMIN and HPC: bit.ly/HPC-ADMIN-Update

Linux Update: bit.ly/Linux-Update

The IT industry has been obsessed with security for years, so you'd think all our systems would be really, really secure by now. Seriously, though, doesn't it sometimes seem that things are getting worse? In the great security arms race, the attackers are innovating just as quickly as the defenders. What can you do to stay safe? Install updates, don't click on strange links, and tune in to your firewall.

One reason some users tend to avoid configuring firewalls is that they are *just too complicated*. The Uncomplicated Firewall (ufw) was developed as an attempt to demystify firewall configuration. Ufw, which was created by Ubuntu but is supported in many other GTK-based Linux systems, is a simplified interface for configuring netfilter, the firewall system built deep into Linux. We'll help you get started with ufw in this month's Linux Voice. Also inside, we introduce you to HedgeDoc, a versatile Markdown editor that lets you convert your documents to HTML and other formats.



Image © Olexandr Moroz, 123RF.com

LINUXVOICE ▶

Doghouse – Bug Reports **73**

Jon "maddog" Hall
For a better bug report, maddog offers a refresher course on crafting a clear statement that will help get your problem fixed.

ufw **74**

Tim Schürmann
Canonical's ufw lets you configure your firewall without the hassle of the iptables tool, while reducing the risk of misconfiguration and simplifying maintenance.

KXStitch **78**

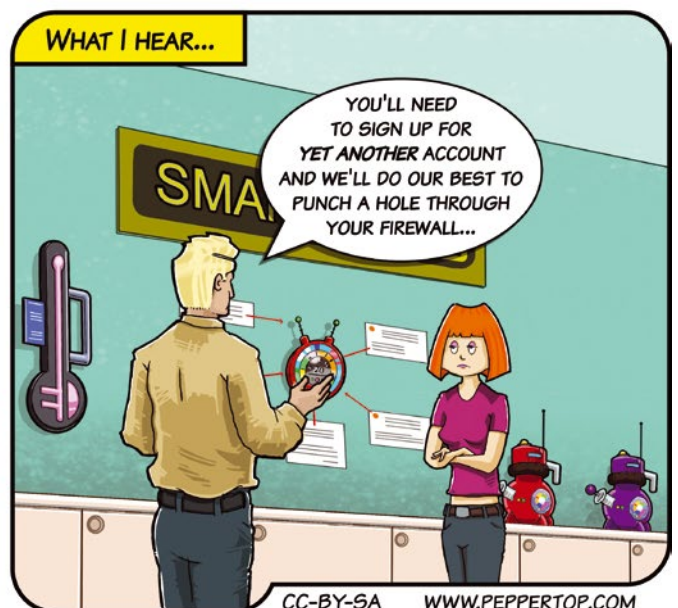
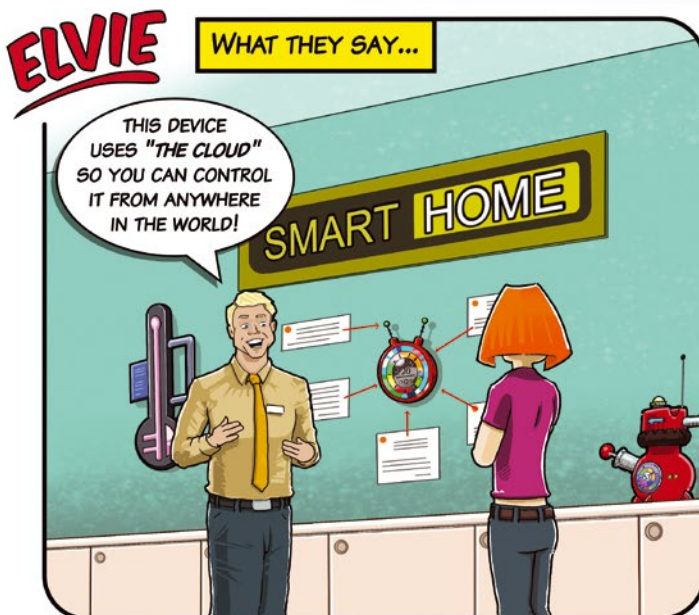
Sirko Kemter
Modern technology and new programs can revive old handicrafts. KXStitch helps design cross-stitch patterns and even automatically converts imported images.

FOSSPicks **82**

Graham Morrison
This month Graham looks at MyGNUHealth, Snippt, Pigion, CudaText, KnobKraft Orm, D2X-XL, and more!

Tutorial – HedgeDoc **88**

Marco Fioretti
HedgeDoc lets you write documents collaboratively in Markdown and publish them online.



CC-BY-SA WWW.PEPPERTOP.COM



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



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®.



For a better bug report, maddog offers a refresher course on crafting a clear statement that will help get your problem fixed. BY JON “MADDOG” HALL

Better Bug Reports for Better Help

A blog I read from time to time had a short article complaining about “bug reports” and bemoaning the fact that users telling the programmers “it broke” was somehow not good enough. I have written articles about writing bug reports in magazines and books over the years, but apparently a “refresher” is needed every once in a while. Plus, containers, virtual machines, and more modern processors make things more complicated.

Specify the Problem

First, try to determine whether the problem is a true problem rather than you just do not know what to do when you want to do it. If it is the second issue, then search in whatever documentation exists. Many large applications (word processing systems, projects of every kind) have fairly good documentation in their online Help menu.

After looking through the documentation, you might search the Internet to see if someone else is having the same problem. A few well-worded queries in your favorite browser may find other people with the same problem, and you may find a way of working around the issue. If it still seems that people are just not understanding the documentation, then that is still a problem and should be reported. After all, it is causing people frustration and wasting time, so it is still a bug, but now you have more insight into the issue.

You now have done about all you can do before assuming the problem really is the application code, or perhaps the system. Next, you need to describe the problem to someone, somewhere, who can fix it.

Provide Information

Start gathering information about your problem. What software version and release number are you using? You can usually find this in the Help documentation. Do not only supply the application information, but also give information for the operating system on which you are running the application.

Include the distribution name (Ubuntu, Mint, Red Hat, etc.), version number, and release (or revision) number of the operating system. You also want to get the version number and release number of the kernel, which will probably be different. You can easily find the kernel information by typing in `uname -a` into a terminal emulator, for example:

```
$ uname -a:
Linux shaman 5.4.0-80-generic #90~18.04.1-Ubuntu SMP Tue
Jul 13 19:40:02 UTC 2021 x86_64 x86_64 x86_64 GNU/Linux
```

To find the information you need for the kernel, look at the output following the name of your machine, in this example “shaman”.

What architecture is your system? Is it Intel, AMD, ARM, or RISC-V? How many cores does your processor have, or specifically, what is your processor’s model number?

How much main memory do you have?

Are you running the operating system in a virtual machine (VM)? What is the VM’s version and what are the settings?

Are you running the application in a container? Give as much information about the container as possible.

Try to make your environment as simple as possible. See if the problem remains when you run it directly on a distribution and the distribution directly on the hardware of the machine.

To make this a lot easier, many systems have a command that tells you all the necessary information with one click. All you have to do is cut and paste or direct the information into a file.

If your application uses a browser, state which browser name and version you are using.

Report the Facts

Now you should make a *clear* statement of what is wrong. Be as precise as possible. Do not say “it broke.” What “broke”? What did the program do or not do? Take a screen shot, if possible. Supply an example of input and output.

If the system crashed, do not just say “it crashed.” It may have been the web browser, the X server, or the operating system. Be clear: “The X-server seems to have crashed. The screen went blank for a couple of seconds, but it came back quickly.” Or perhaps, “The screen went blank and the operating system rebooted.”

If the system is “hung,” describe what else happens. Is it only one window that is not responsive or all the windows? Does anything move or react?

Make a Demo

If you are a programmer, write a small program that exercises the same problem, which you can provide to the operating system or library developer. If your small program does not create the same fault as your larger program, then perhaps something else in your program is causing the problem.

If you do experience the same problem, also supply the name and version number of the compiler you used to compile your program and the options that you used when you compiled it.

Supplying as much information as you can on your first trouble report may inspire the programmer to fix your problem instead of someone else’s problem. ■■■

Simplify your firewall setup

Fire Protection

Canonical's ufw lets you configure your firewall without the hassle of the iptables tool, while reducing the risk of misconfiguration and simplifying maintenance.

BY TIM SCHÜRMAN

The netfilter firewall included in the Linux kernel can be comprehensively controlled with the iptables tool. However, iptables' complexity not only drives some users crazy, it also increases the risk of unintentionally tearing holes in the firewall with incorrect rules or typos.

Canonical offers a remedy with the Uncomplicated Firewall (ufw) [1]. The command-line program accepts clearly structured rules, which it translates into the appropriate iptables calls in the background. This approach also allows you the advantage of supplementing your setup with more complex rules in iptables, if needed.

Installation

Originally developed by Canonical for Ubuntu, ufw has been part of the distribution since Ubuntu 8.04. Alternately, you can install it with the `ufw` package. You can also now find ufw on other distributions.

If your distribution's repositories do not contain ufw, you can pick up the source code online [2]. To get started, ufw requires Python v3.4 or later, iptables 1.4 or later, `gettext`, and `make`. After unpacking the source code archive, just call

```
python3 ./setup.py install
```

with root privileges for a global installation. To start the firewall at boot time, integrate the command

```
/lib/ufw/ufw-init start
```

into the respective start scripts. An example unit for systemd is available in the source code archive in `doc/systemd.example`.

If you also want to regulate IPv6 traffic, open the configuration file located in `/etc/default/ufw` and make sure it contains a line stating `IPv6=yes`. In this article, all examples use IPv4 addresses, but the commands will also work with IPv6.

Listing 1: Connections

```
$ sudo ufw default deny incoming
$ sudo ufw default allow outgoing
```

Blockade

Before getting started, check whether the firewall is running with:

```
sudo ufw status
```

If a **status: inactive** message appears, launch ufw by typing

```
sudo ufw enable
```

This command also ensures that the firewall starts up automatically at boot time. If necessary, you can disable it again at any time with:

```
sudo ufw disable
```

By default, ufw blocks all incoming requests and allows all outgoing messages from the machine to pass. This prevents attackers in particular from reaching any service on the corresponding system. At the same time, the behavior gives you a safety net that catches everything; unless another rule says otherwise, ufw applies the default rules. For example, if you do not define a rule for SSH access, ufw automatically blocks access from outside based on the default rules.

You can change the default behavior with the two commands shown in Listing 1. The first line takes care of all incoming connections, while the second line is for outgoing connections; `deny` prohibits access, while `allow` permits it. Consequently, the two commands ensure the default behavior. If you were to replace `allow` with `deny` in the second line of Listing 1, ufw would automatically prohibit all network traffic.

Like all other commands, the two commands in Listing 1 are intuitive. Even without knowledge of the individual parameters, you can decipher what the command does. Ufw uses its own syntax, which is based on the OpenBSD PF firewall's syntax. If you have previously worked with other tools like iptables, you will need to learn ufw's syntax.

Regulators

Ufw lets you drill holes in the firewall in a targeted way. To do this, specify the appropriate service after the following command:

```
sudo ufw allow
```

For instance, the command shown in Figure 1

```
sudo ufw allow ssh
```

allows SSH connections from the outside. Repeat this step for all other services you want to allow. For example,

```
sudo ufw allow http
```

allows access to an HTTP browser via port 80. All supported names and services can be found in the `/etc/services` file.

In addition, ufw also understands the names of some applications. For example,

```
sudo ufw allow 'CUPS'
```

sets up custom rules for the CUPS printing system. To determine which application names a system currently supports, use:

```
sudo ufw app list
```

On Ubuntu, the range of available applications depends largely on the installed services. For example, if the web server Nginx is not available on your system, ufw does not support it either.

If an application name contains spaces, such as Nginx Full, you will need to quote it in the ufw call, as shown in the CUPS example. Otherwise, the shell interprets the words in the name as individual parameters. It is a good idea to get into the habit of always enclosing application names in quotes.

An application's rules are defined by an application profile (Figure 2). All existing profiles are grouped in the directory `/etc/ufw/applications.d/`. The files in this directory can be used as a basis for your own application profiles; its structure is self-explanatory.

When creating a firewall rule, you can also specify the port directly. For SSH, for example, you would specify the port as follows:

```
sudo ufw allow 22
```

Ufw then automatically sets up matching rules for the TCP and UDP protocols. To allow only a specific protocol, append it to the port number with a forward slash (`22/tcp`). Complete port ranges can also be stored. For example,

```
sudo ufw allow 8080:8082/tcp
```

opens ports 8080, 8081, and 8082 for incoming TCP connections.

```
dd@ubuntu:~$ sudo ufw allow ssh
Rule added
Rule added (v6)
dd@ubuntu:~$ sudo ufw status
Status: active

To          Action    From
--          -
22/tcp      ALLOW     Anywhere
22/tcp (v6) ALLOW     Anywhere (v6)

dd@ubuntu:~$
```

Figure 1: The command `ufw allow` automatically adds rules for IPv4 and IPv6 connections, as evidenced by the `ufw status` that follows.

Fine Tuning

Ufw stores all the rules and enables them automatically after a system reboot. The IPv4 rules are stored in the `/etc/ufw/user.rules` file, and the IPv6 counterparts in `/etc/ufw/user6.rules`. After creating new rules, you should reload these files for safety purposes by typing:

```
sudo ufw reload
```

Based on the default rules, ufw allows all outgoing connections. To specifically deny a service access to the network, use `deny` instead of `allow`. In addition, use `out` at the end of the line to indicate that the rule applies to outgoing connections. For example, to prohibit outgoing traffic on port 22, use:

```
sudo ufw deny out ssh
```

After issuing this command, the system can no longer contact another host via SSH. Rules for incoming connections are tagged in the same way with `in`. In all the previous examples where this keyword is missing, ufw automatically assumes that the rule applies to incoming connections.

In addition to `allow` and `deny`, `reject` signifies that the firewall does not simply ignore access attempts but also notifies the sender of the attempts. Also, `comment` lets you attach a note to all rules (Listing 2, first line). Each rule always applies

```
dd@ubuntu:~$ sudo ufw app list
Available applications:
  CUPS
dd@ubuntu:~$ sudo ufw allow 'CUPS'
Rule added
Rule added (v6)
dd@ubuntu:~$ sudo ufw status
Status: active

To          Action    From
--          -
22/tcp      ALLOW     Anywhere
CUPS        ALLOW     Anywhere
CUPS (v6)   ALLOW     Anywhere (v6)

dd@ubuntu:~$
```

Figure 2: Only CUPS and an SSH server are installed here, which means that ufw only maintains application profiles for the two services.

Listing 2: Comments and Interfaces

```
$ sudo ufw reject out ssh comment 'no ssh access allowed'
$ sudo ufw allow in on enp0s3 ssh
```

Listing 3: Unblocking

```
$ sudo ufw allow from 192.168.1.101 to any port 22
```

to all network interfaces. To restrict a rule to one interface, specify its name after `in` or `out` (Listing 2, second line).

Bouncer

Access via SSH should only be allowed for defined hosts. To do this, first deny SSH access globally with

```
sudo ufw deny ssh
```

Since this is also the default setting, you can alternatively remove the rule

```
sudo ufw delete allow ssh
```

which deletes the `allow ssh` rule. If you can't remember the rules, call

```
sudo ufw status verbose
```

In addition, each rule is internally given a sequential number, which can be displayed with:

```
sudo ufw status numbered
```

You can use these numbers to delete specific rules. For example, to remove the rule assigned the number 2, use:

```
sudo ufw delete 2
```

Now that access via SSH is generally blocked, the command shown in Listing 3 exclusively allows SSH access for the computer with the IP address `192.168.1.101`. If you omit to `any port 22`, the IP address is allowed to access all services. Similarly, you can use `deny` to block specific requests from an IP address.

Numerous requests within a short time indicate an attack and can also overload the affected service. If so desired, `ufw` can detect this kind of access attempt and then block it specifically. Currently, however, this useful function only works with IPv4 connections. For example, the firewall monitors the SSH service with the command

```
sudo ufw limit ssh
```

and blocks access if there are too many requests in a short time.

Chatterbox

If you get tangled up in too many rules, use the following command to start over:

```
sudo ufw reset
```

When creating new rules, you can use various reports for help. Use

```
sudo ufw show listening
```

to return all services that are currently listening on any port. This helps you find applications that you didn't know were running or that shouldn't be running at all (Figure 3).

If you are familiar with `iptables`, you can take an in-depth look into the firewall's current configuration with:

```
sudo ufw show raw
```

`Ufw` stores detailed information about its work in a log, which you can enable with

```
sudo ufw logging on
```

and then view in `/var/log/ufw.log`.

Detours

Since version 0.34, `ufw` now also supports routing. This means that the firewall can wave through incoming packets and, for example, forward all requests arriving on network interface `enp0s3` to the interface `enp0s8` (shown in Listing 4).

For IP forwarding to work, the corresponding function must be enabled in the `sysctl.conf` configuration file. On Ubuntu, you use the `/etc/`



Figure 3: Besides CUPS, the Avahi daemon and NetworkManager also are listening on the network interfaces.

ufw/sysctl.conf file for this purpose; enter the lines from Listing 5 or – if they already exist – enable them by removing the preceding hashtags (#). If you made some changes, re-start ufw by typing

```
sudo ufw disable
```

followed by

```
sudo ufw enable
```

This more or less brings us to the end of ufw’s feature set. In particular, ufw does not yet support masquerading, where the firewall changes, among other things, the source and destination ports in the packets that pass through the firewall. But, as mentioned earlier, more complex rules can be added using iptables. The corresponding configuration is stored either in the /etc/ufw/before.rules file or in /etc/ufw/after.rules. These rules are applied by the firewall before or after the rules

Listing 4: Forwarding Requests

```
$ sudo ufw route allow in on enp0s3 out on enp0s8
```

that you defined with the ufw command-line program.

Listing 5: Enable Forwarding

```
net/ipv4/ip_forward=1
net/ipv6/conf/default/forwarding=1
net/ipv6/conf/all/forwarding=1
```

Gufw

It is even easier to configure the firewall with Gufw [3], the ufw’s graphical user interface. However, since it is not officially part of the ufw project, you usually have to install it in a second step. On Ubuntu, you can install Gufw with:

```
sudo apt install gufw
```

After starting Gufw, click the button next to Status to fire up the firewall. Then, in *Inbound* and *Outbound*, set the respective default rules. The *Report* tab (Figure 4), an extremely practical feature, displays the running services more clearly than the matching ufw show listening command. Clicking the plus icon also automatically creates a matching firewall rule.

All existing rules can be found in the *Rules* tab. Use the gear icon to edit the currently selected rule and the plus icon to add another rule. Under *Preconfigured*, you can select an application profile; Gufw sorts the applications into categories. CUPS, for example, can be found below *Network* in the *Print* subcategory. If you don’t want to use application profiles, switch to the *Simple* tab. Even more granular settings are allowed by the *Advanced* tab (Figure 5).

Conclusions

With the comparatively simple ufw, a firewall can be configured far faster than with the more complex iptables. The simple ufw rules also reduce the risk of misconfiguration and simplify maintenance. Nevertheless, ufw provides all the critical functions required to harden popular services. If you reach ufw’s limits, you can add further rules with iptables. However, ufw and iptables’ different syntax does prove to be a hindrance here. The bottom line, however, is that ufw makes setting up a firewall far easier. ■■■

Info

- [1] ufw: <https://launchpad.net/ufw>
- [2] ufw source code: <https://code.launchpad.net/ufw>
- [3] Gufw: <http://gufw.org>

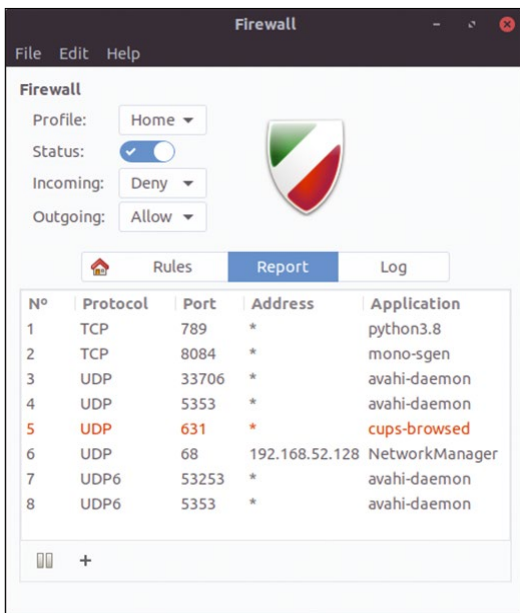


Figure 4: The Report tab in the Gufw user interface shows the services running on the system.

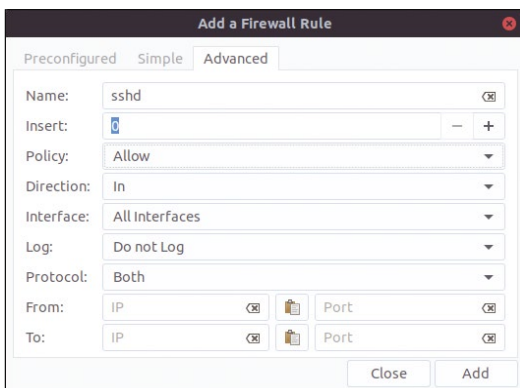


Figure 5: This rule blocks access to port 22 via TCP.

Create cross-stitch templates with KXStitch

Cross-Stitched

KXStitch lets you design cross-stitch patterns with the option to automatically convert imported images.

BY SIRKO KEMTER

Cross-stitch is a very old needlework technique. Formerly a pastime of the nobility or higher-ranking ladies, cross-stitch is now considered a popular and easy-to-learn hobby among children and adults alike.

In cross-stitching, you embroider a small X-shape on a coarsely woven fabric with a good countable structure. This is done by first applying a forward angled ground stitch followed by a backward angled cover stitch over the ground stitch, resulting in an X. However, there are also quarter, half, and three-quarter stitches – in cross-stitch jargon, they are known as broken stitches. In addition, there is the backstitch and the knot stitch, also known as a French knot.

There are numerous templates and motifs for cross-stitch available in stores and on the Internet but not always free of charge. Alternatively, you

can design your own patterns with the help of your computer. KXStitch [1] offers an open source solution for creating your own cross-stitch patterns for Linux and the Raspberry Pi OS.

Start Creating

KXStitch converts images or texts using the fonts installed on your computer into stitch patterns. Patterns can be saved in a library and reused. Once a pattern is created, KXStitch prints the pattern or exports it as a PDF file. In addition, the patterns only print in black and white, not in color, so the colors must be displayed in encoded form.

Most popular distributions include KXStitch in their package sources, so the installation is easy. If you don't find what you are looking for in the repos, KDE provides an AppStream package on the KDE Apps pages. There are also installable packages for various distributions in the openSUSE Build Service. You can find the KXStitch source code on GitHub [2].

After starting KXStitch, you need to first adjust the settings in *File | File Properties* (Figure 1), where you control the palette, the pattern size, and the cloth density (*Cloth Count*), in addition to meta information, such as author, title, and copyright, as well as any instructions.

For units of measurement, KXStitch only supports stitches, inches, and centimeters, which makes it difficult to share files with some graphics programs. You may want to change the default setting from *Stitches* to *Centimeters*. For the color palettes in the *Floss Scheme* section, the program supports three different floss schemes: *DMC*, *Madeira*, and *Anchor*.

Next, you must adjust the palette. Otherwise, many of the functions are disabled, making things a bit confusing. To do this, call *Palette | Palette Manager*. In the *Palette Manager* dialog, transfer the desired colors from the previously selected color scheme to the project's palette (Figure 2) by selecting the desired color in the right-hand section and clicking the middle button with the two arrows pointing to the left. In the dialog that pops

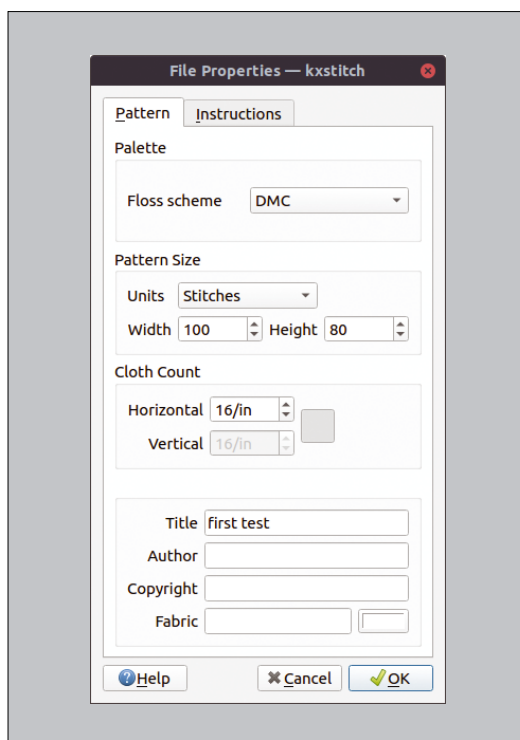


Figure 1: KXStitch settings: Cloth Count indicates the number of stitches per centimeter of fabric.

up when you're done, press *OK* to confirm your selection. Otherwise, KXStitch will not create the palette.

After that, KXStitch is ready for use. By default, the program is in a drawing mode. Similar to a paint program, you work with the drawing area, creating stitches instead of pixels. This becomes clear as soon as you zoom in to the drawing with *Ctrl++* or via *View | Magnify*. Similarly, you can zoom out by pressing *Ctrl+-*. However, KXStitch does not show the current scale. There is also a limit to the magnification.

The Tools menu offers some of the options already mentioned, such as displaying text (Figure 3). KXStitch opens a dialog for inserting text where you can enter the text, the font, and the size. Click *OK* to transfer the text to the canvas. The red frame around the text indicates that you can currently move the text freely to the canvas. But be careful: This will work once only. After that, KXStitch anchors the text object firmly on the canvas. Drawing rectangles, ellipses, or polygons with and without fill works in a similar way.

Managing the Library

The *Library Manager* menu plays an important role, especially if you are more interested in traditional pattern-oriented designs. To do this, however, you must first create and populate the library, which means drawing and saving many smaller patterns. In the library, these patterns can then be broken down into categories and subcategories.

To populate the library, first create a new category in the *Libraries* pane by right-clicking in the

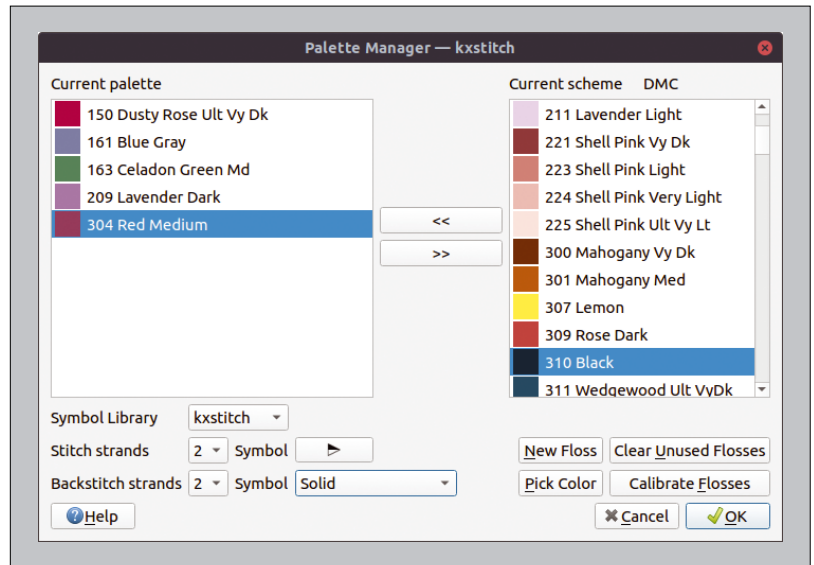


Figure 2: Before you can get started, you need to add colors to palettes. Later on, KXStitch will use the symbols in the printout to identify the different threads.

pane and selecting the *New Category* option. Then draw the desired object on the canvas, select it using the *Tools | Select* option, and copy it to the clipboard using *Ctrl+C*. Then use *Paste* in the context menu to transfer the object to the library after right-clicking in the left pane. *Ctrl+V* does not work here.

You can simply drag and drop the patterns from the library back onto the canvas. Alternatively, you can use the *Alphabet* function to search the existing libraries for the corresponding pattern that has been assigned to the keyboard shortcut, such as *Shift+L* for the Linux pattern shown in Figure 4. This lets you quickly

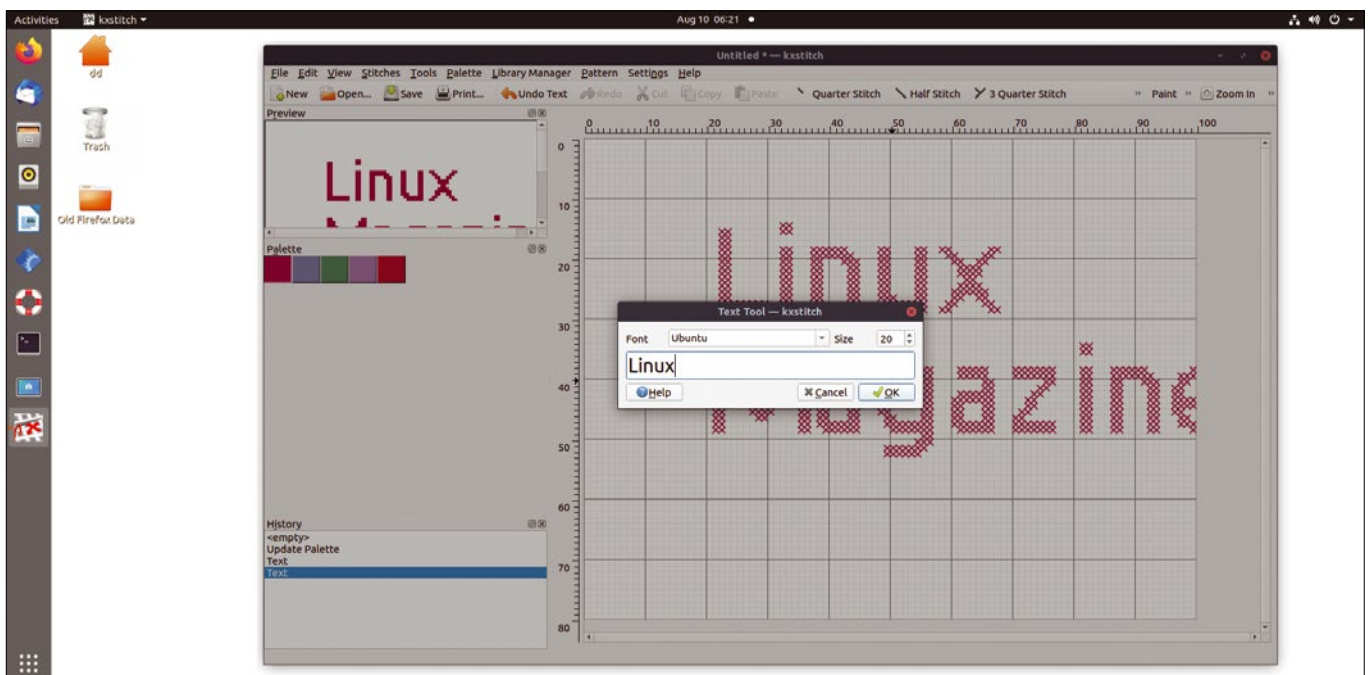


Figure 3: The opened Text Tool with the Palette pane (left) and the transferred text shown on the canvas (right) in the background. The Palette pane displays the icons for the threads.

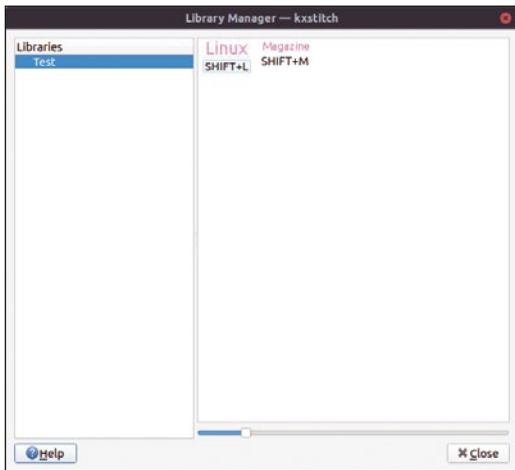


Figure 4: The pattern library makes it easier to work with repeating patterns. Use the keyboard shortcut specified below the pattern to insert the element on the canvas.

insert your own letters or even patterns onto the canvas.

If you want to design a pattern using images, you need to import graphics. KXStitch offers two options. If you import directly via *File | Import Image*, KXStitch automatically converts the image into a cross-stitch pattern (Figure 5). In our lab, this worked reliably with pixel graphics in JPG or PNG format, but vector graphics in SVG format reproducibly crashed

the program. You may need to convert your image to a PNG graphic with an image editing program beforehand.

If you are importing an image as a background using the *File | Add Background Image...* option, KXStitch only displays the image in the background and does not calculate a cross-stitch pattern. You can trace the imported graphic yourself as a pattern by hand, much like tracing an image with tracing paper. In *View | Display Background Image*, you can turn the display of the background image on and off at any time.

Converting Images

One of the central problems with cross-stitch is printing out the pattern images. Very few hobby stitchers have a printer capable of producing hard copy the size of a tablecloth. Even if they did, such a huge printout would be difficult to handle. It's a matter of breaking the image down into manageable pieces. Since the whole thing is usually printed in black and white rather than in color, you will also need a glossary of the characters used for the colors and stitches.

KXStitch impresses here. The program handles this task at the push of a button. To view the breakdown and for an overview, you can click on the button with the three dots in the upper left corner in the Print Setup dialog, which you access by selecting *File | Print ...* KXStitch then automatically breaks down the contents of the canvas into pieces that fit the set paper size and adds a glossary of the colors used. At the same time, the application calculates the required length for each thread, which makes purchasing the material easier (Figure 6).

On the Rasp Pi

KXStitch is also available in the package sources for the Raspberry Pi OS. Combined with the Rasp Pi, KXStitch makes an ideal craft project for children. A child can simply draw a design on paper, then scan or photograph it, and import it into KXStitch. KXStitch then converts the design into an

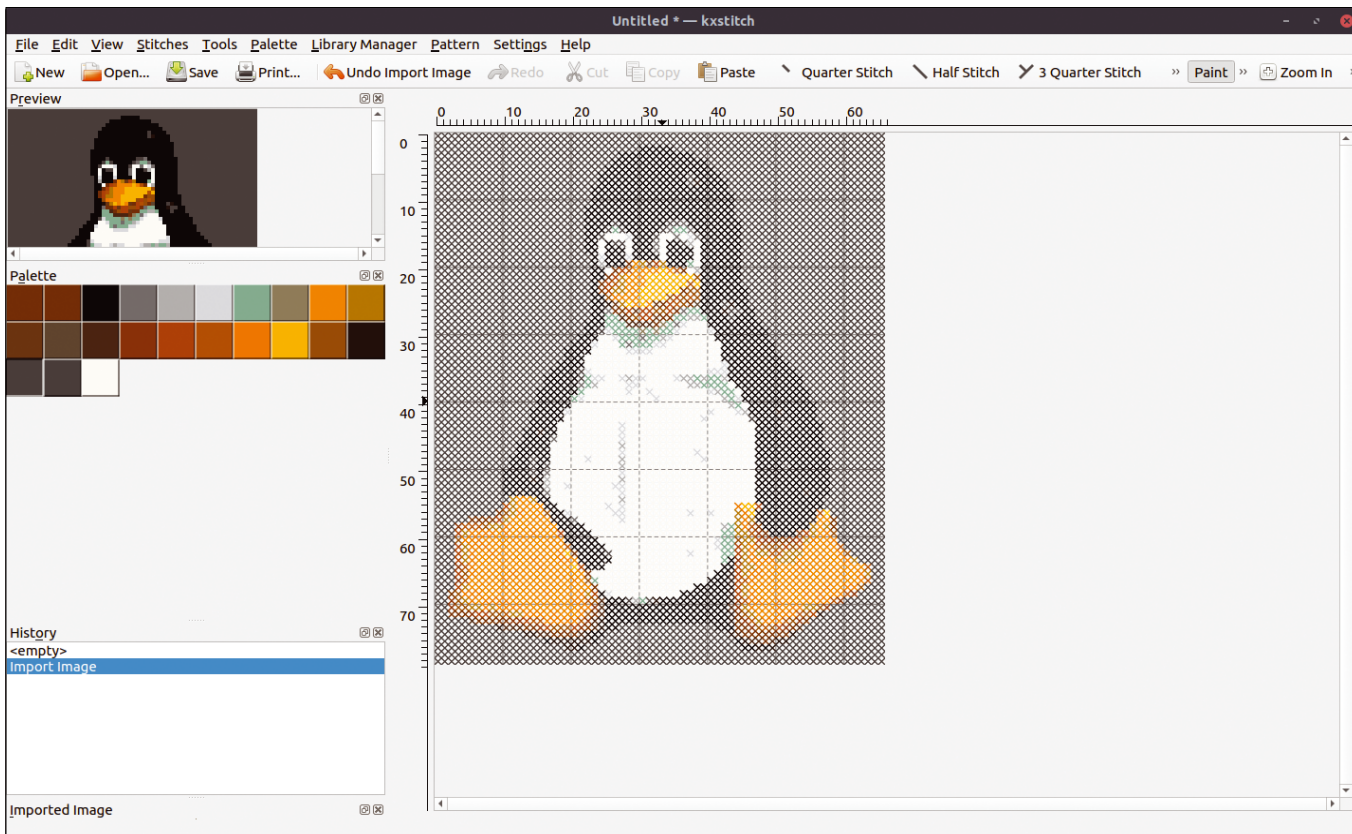


Figure 5: KXStitch either automatically converts imported images into a pattern or places the image on the background of the canvas for tracing.

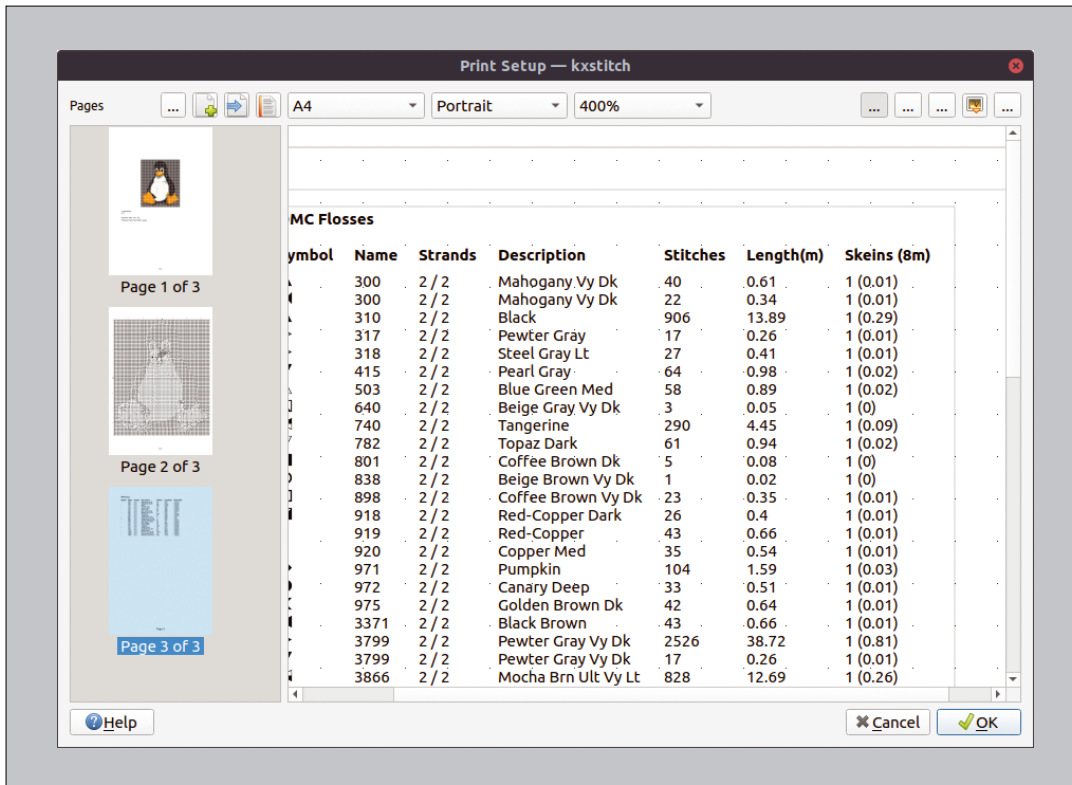


Figure 6: The Print Setup dialog distributes the project across several pages. It creates an overview of the threads used and calculates the required thread length for each color.

embroidery pattern at the touch of a button, and children can then manually transfer the design to the fabric in the final step.

Conclusions

KXStitch lets you use interesting images and patterns to create cross-stitching patterns. However, KXStitch's complicated handling does tend to spoil the fun slightly.

The latest version of KXStitch is already more than two years old. One can only hope that KXStitch will continue to be actively promoted. For

this, it needs both active developers and a certain number of users who work with the program and push its development forward with bug reports and requests. Unfortunately, there seems to be a lack of both at the moment – too bad. ■■■

Info

- [1] KXStitch:
<https://apps.kde.org/de/kxstitch/>
- [2] Source code:
<https://github.com/KDE/kxstitch>



FOSSPicks

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



Graham recently bought a Bela Pepper – a DIY, open source, and Pure Data-based synth platform – in yet another example of procrastinating rather than making any music. **BY GRAHAM MORRISON**

Health diary and tracking

MyGNUHealth

In many ways, the modern obsession with bioinformatics and personal health monitoring started with the open source movement. Self-hacking and obsessive self-monitoring were popular subjects at technical conferences and in local Linux User Groups 15 years ago. Our open source ecosystem was perfect for developing ad hoc algorithms and software that held personal and private data about your health. But proprietary smartwatches and other common exercise sensors changed this with their penchant for zero public API access, data gathering, and linked services. However, open source health monitoring that puts privacy first is still a thing; it just needs to find its way onto our devices and desktops.

Which is exactly what the KDE Plasma project has done with MyGNUHealth (they really should have thought of a better name).

The name MyGNUHealth is a consequence of it being an application designed to integrate with an upstream organization, the GNU Health Federation, which is a broader initiative designed to operate as a hospital information system and manage electronic health records. Its goals are also modular and very much include those of the more modest MyGNUHealth. To this end, MyGNUHealth is both a desktop and a mobile application, with Plasma seamlessly scaling between the two. Upon first launch, you're asked for your name, date of birth, gender, and height – the kind of information

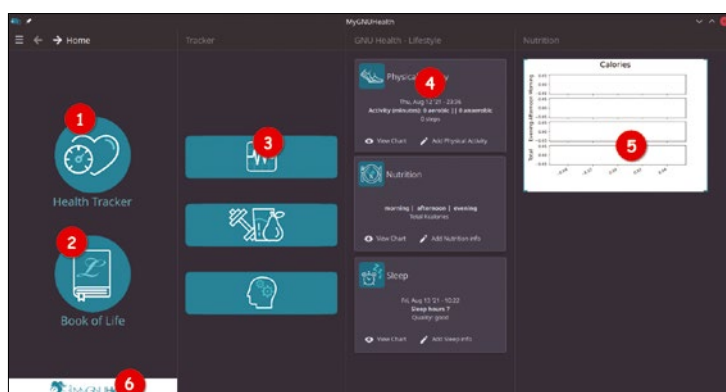
that you wouldn't ordinarily give away freely. Luckily, you can inspect the code if desired. MyGNUHealth also asks for an encryption password, further safeguarding your data from exploitation.

With this brief configuration out of the way, the application offers two main functions: *Health Tracker* and *Book of Life*. Both are connected. *Health Tracker* is what you might expect from its name and will be familiar to anyone with a Fitbit, Garmin, or Apple Watch. It presents several different graphical elements which are used to track a variety of personal metrics, from blood pressure measurements and physical activity details to perceived mood and energy, among many others. There's currently no way to automate their input from linked hardware, but hopefully that will come. The clever part is that much of this is done simply through wonderful iconography and artwork. To enter a mood, for instance, a simple slider is used to select a level alongside a person who animates to show their level of happiness. Similarly, a battery icon is used to correlate energy levels. Some, such as adding a physical activity, do require data entry such as aerobic and anaerobic duration, which can only be gained through your own calculations or

hardware, but the design is so inviting you actively want to add more details.

Whatever you do and whatever data you enter will be added as a log entry to the second main application function, the *Book of Life*. This acts as a global activity tracker and enables you to easily see your levels of activity, nutrition intake, and any other metric you happen to track. It also allows you to add diary entries for medical information, including many types of health or medical screening. Adding entries for social interaction allows you to track external factors such as stress, unemployment, and addiction to better understand their effects on your overall welfare. Unlike the proprietary systems we alluded to earlier, MyGNUHealth offers no advice or recommendations itself, but it's invaluable for personal introspection and health tracking, as well as for logging essential information for medical professionals who can take advantage of the GNU Health Foundation's standards to import your data to, hopefully, better understand your health. It's hugely ambitious, but what else would you expect from an application that might just save your life?

Project Website
<https://github.com/KDE/mygnuhealth>



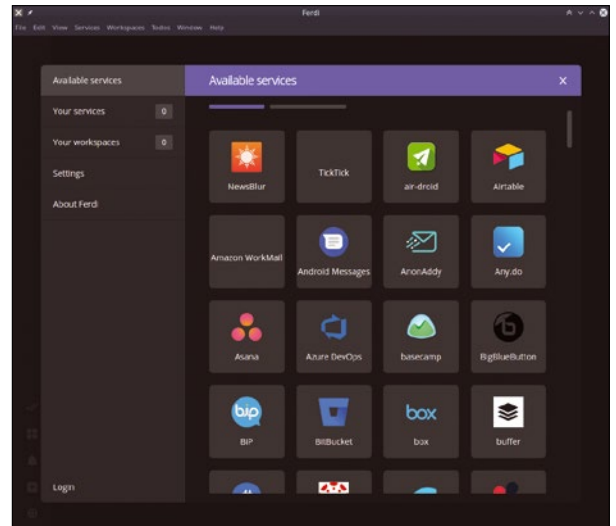
- 1. Health Tracker:** The main functions are split into tracking and diary entries.
- 2. Book of Life:** This is a social diary facility that aggregates tracking events and lets you add your own data.
- 3. Data tracking:** Different categories of tracking are used to collect data about your health and lifestyle.
- 4. Metrics:** Data needs to be added manually, such as for exercise, sleep, or meals.
- 5. Charts:** See changes over time based on the data points you log.
- 6. Federal health accounts:** It's also possible to link your logging to other sources.

Messenger browser

Ferdi

Many of our day-to-day desktop applications, such as email and chat clients, text editing, music playback, and more have switched to become web applications. This puts a resource strain on our web browsers but also a cognitive strain on our brains as we struggle between using these applications and more traditional web searches and page views in a browser with dozens of tabs open. One possible solution to this is to offload these “Software as a Service” types of pages to a separate application, one that might even be able to aggregate shared functionality to make things even simpler. This is what Ferdi hopes to do, with an emphasis on applications that help us to communicate.

Ferdi does this via its own online service, but it’s also possible to use the application without creating an account. Either way, when Ferdi first launches, you’re presented with a list of web apps and services it supports. This list is huge, with more than 100 services supported, from Android Messenger to Zeplin – almost anything with a web client. If something isn’t in the list, there’s a good chance you can add it via a custom URL, and in our experience, it will work. Selecting or adding a service will step you through account authentication, after which the app will appear within a global toolbar on the left. The apps themselves operate exactly like their browser counterparts in a traditional web browser because they are. Ferdi isolates these web applications to help you to keep your focus and avoid too much context switching. It can also help limit or manage the time you spend checking Slack or Discord, for example, because they’re



Ferdi consolidates many of your communication web apps into a single window.

confined to a single, different application, and it’s handy having all unread notifications and services aggregated to a single place or icon. It would be great if this principle could be taken further, with time limits and access intervals when certain applications become available, but maybe those are possibilities for a future release.

Project Website

<https://github.com/getferdi/ferdi>

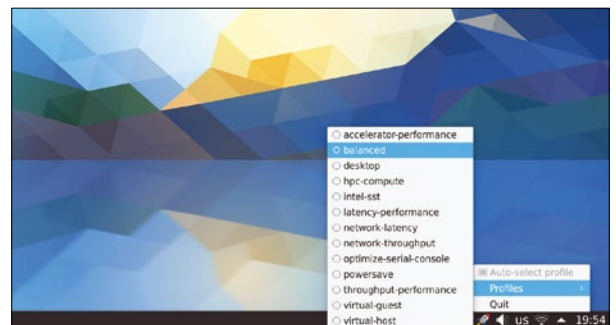
Performance profiles

Tuned Switcher

The Tuned service is a daemon that can control, or tune, certain performance-affecting hardware characteristics on your computer. This can help you squeeze the most from your machine when you need it for a specific application or simply for battery life. While it is possible to start by creating your own configuration for the tasks you need to perform, the service comes pre-packaged with profiles for certain tasks. These are split into two categories: power-saving modes and performance-boosting modes. Performance-boosting modes are useful when running virtualization software or when you want the best possible network performance. Power saving will help prolong battery

life or keep your hardware from overheating. You can define and switch between profiles using configuration files, but there is a better way – an application called Tuned Switcher.

Tuned Switcher is a simple tool that can itself operate in two modes. The first, and default, is as an applet in your desktop system tray. When running from here, a right-click will open a menu containing a list of whatever Tuned profiles are installed on your system, including those for performance, virtualization, and network access. Selecting one will change the performance characteristics of your hardware and operating system immediately, and a notification pane will appear to inform you which mode has been selected. There’s



Change the performance characteristics of your hardware on the fly with Tuned Switcher.

also an automatic function that will ensure an appropriate profile is always selected to match whatever job the system detects you’re trying to accomplish. Alongside the applet, the second operating mode is a widget. This is more like a traditional desktop application but doesn’t offer the same number of functions as the applet. For that reason, if a system can use either, the widget is selected by default. It’s a simple but effective system that’s going to be of most use to people traveling with a laptop, because it lets you put your hardware into a high-power performance mode when your environment might otherwise automatically preconfigure for battery life.

Project Website

<https://github.com/EasyCoding/tuned-switcher>

Data sharing

Sniptt

We've looked at many different tools that can help you manage an ad hoc file transfer across a network or across the Internet. Magic-Wormhole remains one of our favorites, because it requires nothing of the recipient other than the software and some way of telling them the human-readable encryption words used to negotiate both the network protocol and decrypt the transfer. This part is usually accomplished via a secure secondary channel, such as using the Signal messenger client. But this leads to another common use for Signal: sending messages to yourself to act as a cross-device copy and paste buffer for secret wormhole keywords and even passwords and other sets of private

information. This isn't a bad idea if you also delete that data after it's used, but in our experience, the end result is more commonly a personal chat history full of your most important secrets.

Sniptt is a command-line tool (and web service). It helps with password and file sharing in the same way Magic-Wormhole can help with file sharing and helps with secret storage in the same way as your private Signal group. But it's also more ambitious because it's a database that uses OpenPGP and requires no configuration other than going through the process of creating a new local account. This process will ask for an email account, which is used for verification and the password to use to encrypt your content. With that

```

Welcome to Sniptt.

Sharing secrets with password managers is slow and expensive.
Instead, share and read secrets without leaving your terminal or IDE.

What email address should we use to verify your identity? · graham@paldandy.com
Sending email verification request
Please enter the verification code sent to graham@paldandy.com
Registering device
What should we name your account? · valuable_logger
What master password would you like to use to encrypt your content? (12 characters or more)
Please confirm master password to continue
Would you like us to store your master password securely in Keychain? (Y/n) · false
Configuring account

[] Configuration written to /home/graham/.config/sniptt/default.json.
Let's try adding a new snip:

$ snip add
  
```

All data in Sniptt is end-to-end encrypted and never leaves your local storage.

done, you can start adding simple key and value pairs, such as usernames and passwords, with the hidden part entered via a prompt. You can even add files. All of this can be retrieved with a simple `get` command. The clever sharing part comes from creating a vault. A vault is a container for a set of secrets, but it also lets you add accounts for other people who can also access the credentials stored in the vault. Even more impressively, you can generate a one-time shared URL for a secret, which you can send to someone to access. It's like a powered-up version of Magic-Wormhole and means you can finally delete that secret Signal group.

Project Website

<https://github.com/sniptt-official/snip>

MIDI router

Pigiron

Pigiron is a command-line utility that brings unity to the world of MIDI and Open Source Control (OSC). MIDI is the protocol used to send and receive musical notes and data, usually between your computer and a synthesizer. OSC was designed to be MIDI's successor, but in many ways it was over-engineered. Instead of the specific commands provided by MIDI, for example, such as `note on`, `note off`, and `velocity`, OSC is entirely dynamic. Applications and hardware create their own commands and publish these as a kind of API. This complexity makes it unintuitive compared to MIDI and might explain why MIDI is still so common. But OSC has become widely adopted when flexibility and power are required

for cross-device communication, performance, and automation. All it really needs is a way to interact with old-school MIDI devices. Enter Pigiron.

After launching Pigiron, you find yourself within an interactive, command-driven environment. Typing `help` will list the various commands that are available, and each command has its own associated help document. In general, commands beginning with `q` (for query) will return data, such as `q-midi-inputs` to list which MIDI inputs are available on the system. But the main command type is an operator. An operator can be a MIDI input or output, a MIDI player, a monitor, a transposer for changing MIDI notes, or a delay, plus several more, and they process data that passes between them. These commands can be used either within the client itself or across OSC, allowing you to dynamically create MIDI and OSC chains of

```

Configuration directory is '/home/graham/.config/pigiron'
Configuration file: /home/graham/.config/pigiron/config.toml
Logging to: /home/graham/.config/pigiron/log
Version 0.1.0 Beta

OSC Listening: 127.0.0.1:8020 /pig

/pig: help topics
----- OK
/pig/help
  [ 0 ]

Help topics:
ChannelFilter
Delay
Distributor
MIDIInput
MIDIOutput
MIDIPlayer
Monitor
  
```

Alongside entering commands into the interpreter or sending them over OSC, Pigiron can also run executable batch files.

data and transformations. This can really help in a live situation or one where you want to use another application, such as Pure Data, to control a MIDI stream outside of its direct control. It sounds complicated, but it's not really when you start using it. We can't wait to see additional procedural operators added to the mix.

Project Website

<https://github.com/plewto/Pigiron>

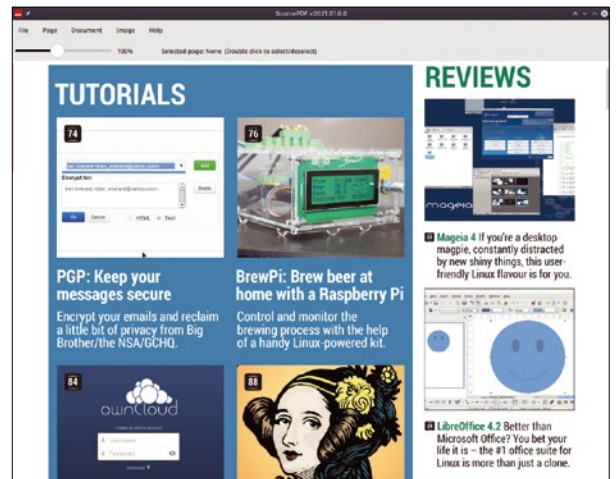
PDF editor

SourcePDF

Until recently, it was an inconvenience that you couldn't properly edit PDF documents on Linux. But it was also a seldom-required function that could usually be side-stepped by using LibreOffice or a web form. Recent events have changed this with everyone from local schools to doctors' offices wanting a PDF edited, or signed, or added to and reshuffled, all of which can be a challenge on Linux. Our PDF tools have improved – and KDE Plasma's Okular is a good example of a PDF viewer that can now do things such as annotations and filling out forms in a meaningful way – but they're still a long way from the capabilities of Adobe Acrobat. While there still isn't a tool that can do everything we need

from a PDF editor, SourcePDF is something new that can fill an important editing gap.

SourcePDF can obviously load and view PDF documents, but its main "downloading and installing" benefit is that it can also merge them. Multiple PDF documents can be loaded at a time and reordered before being saved as a new single file. You can also choose to add an image or a new PDF before or after a specific page in the main document and then change the page ordering before again saving the entire view as a new PDF. Saving new documents also allows you to change the PDF compression level so that you can make larger documents more readily transferable via email or to archive. All of this is accomplished via SourcePDF's only dependency,



While SourcePDF can't help you edit the contents of a PDF, it can help you concatenate them and insert and remove pages.

the venerable Ghostscript. This is a platform and set of tools that are capable of high-quality output and PDD processing. Ordinarily, Ghostscript can be tricky to learn and master, especially when it comes to casual PDF use. SourcePDF has been able to hide this complexity, creating a decent viewer with some unique page and image editing and adding features that other PDF editors can't easily rival.

Project Website

<https://yeahlowflicker.com/sourcepdf>

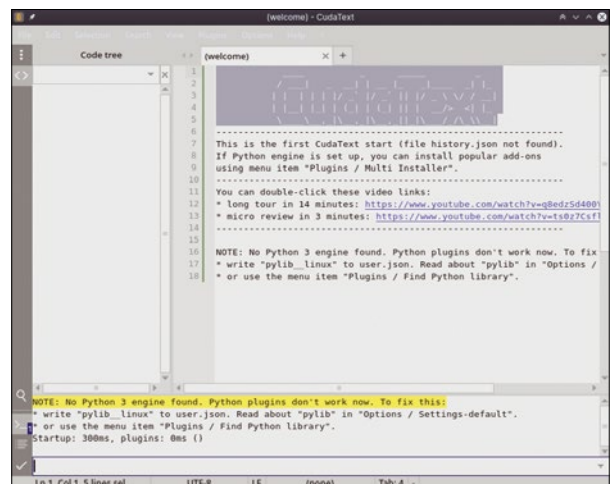
Text editor

CudaText

We've made it through a couple of months without giving in to the temptation of reviewing yet another text editor. But CudaText is worth the break because it genuinely offers some unique features that other text editors don't. It's also worth pointing out that this isn't a text editor designed to help you with NVidia's GPU processing platform, CUDA, as its name might imply. CudaText is instead an editor written with the Lazarus IDE in the Pascal programming language, a language that might be familiar to anyone who studied computer science in the '80s or '90s (like me!). This might also help to explain its best feature, because CudaText is truly cross-platform. And we don't just mean Linux, Windows, and macOS. We mean

FreeBSD, OpenBSD, NetBSD, DragonFly BSD, Solaris, and even Haiku. CudaText runs on all of these, sometimes making it the platform's best GUI-driven text editor. We only wish there was an additional Amiga version.

This is important because CudaText is crammed full of modern features. There's a tabbed UI (optionally powered by Qt on Linux), regular expression support, syntax highlighting, code folding, multiple splits, and a JSON configuration file. There's even code completion and picture preview if you're editing HTML or CSS files. But the real power comes from a plugin system that can extend the regular functionality with extra options to help with your specific tasks. There's a snippet library for code clips, spellchecker, code linter, diff viewer, and merge manager, plus



CudaText brings new meaning to "cross platform," but we do wish there was an Amiga version.

tools for managing an entire project tree and individual sessions. After adding a few plugins, CudaText can start to feel more like an IDE than a text editor, and that's what makes it so powerful. It's a good, functional editor without any further modification, but if your needs grow, the editor can also grow with your project. And it works on almost any system you're likely to have encountered over the past 20 years.

Project Website

<https://cudatext.github.io/index.html>

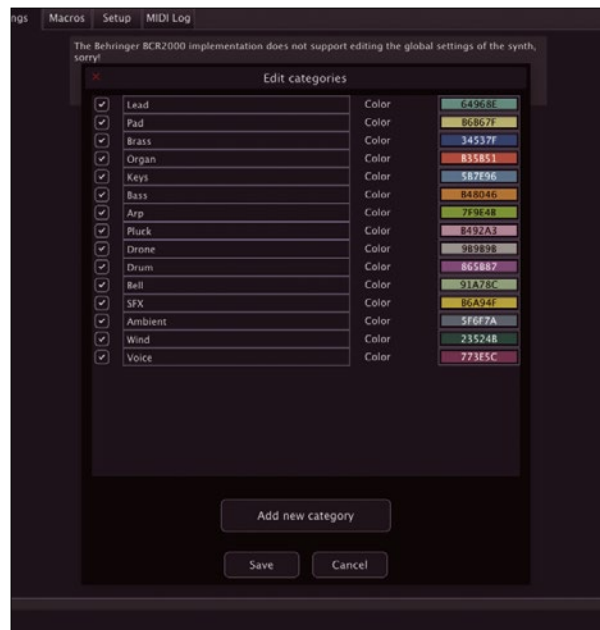
MIDI librarian

KnobKraft Orm

Over the past 40 years, the MIDI specification has enabled physical controllers, hardware synthesizers, software synthesizers, audio applications, and numerous other hardware categories to talk to one another. This is an incredible achievement for a protocol born before TCP/IP. Even today, MIDI has never been more popular. Which is why it's particularly bemusing that there isn't a decent open source settings librarian for MIDI devices. Librarians are essential if you want to save a set of MIDI values for a device and also for computer-driven archival and patch management. MIDI devices are usually limited in their storage capacity and user-interface design, which means you often need to save your data to a computer. The easiest way to generate this data is usually via a system-exclusive data dump from your hardware, which is a transfer of the raw internal device data over MIDI as opposed to data common to each device, such as note or volume data. These dumps can usually

be sent back to the device to re-activate the configuration, but you need a librarian if you want to save these dumps individually and to manage, preview, and categorize them. Without a librarian, this saved data simply exists as a single file without any further context or structure.

One possible solution is the brilliant Ctrlr platform, which offers many different kinds of editors and librarians as well as plugin compatibility and scripting. But it's also a victim of its own success: Data management and librarian functions are often left wanting, and the editors themselves vary hugely in their quality. What's needed is a simple, high-quality MIDI librarian that focuses on a finite number of devices and supports those well. This is exactly what the oddly named KnobKraft Orm does. KnobKraft Orm can natively talk to and save data from many devices, including various Access Virus, Kawai, Roland and Sequential synths, and Behringer's RD-8 drum machine and its BCR2000 control unit. All of



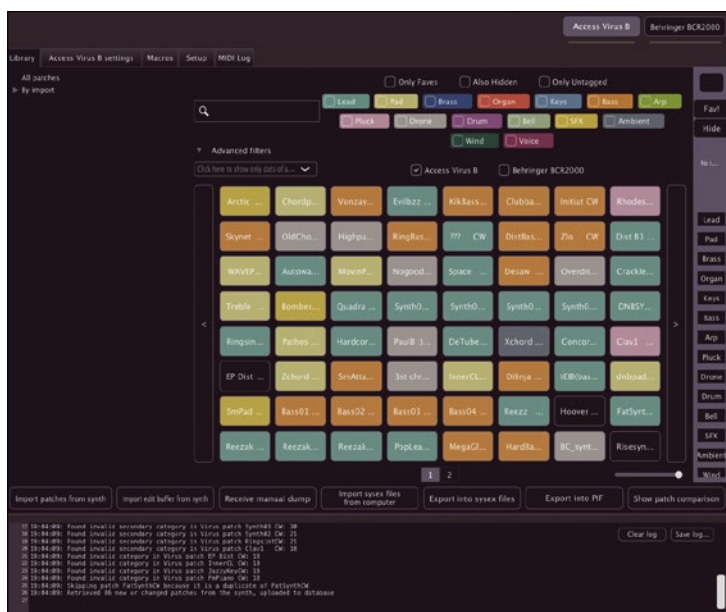
Even if your device isn't supported, if you know the difference between a bit and a byte, it's relatively straightforward to create your own.

these can have their internal data captured, saved, and re-sent, and the user interface works the same for each. Data dumps, or patches, are shown in a grid, with color used to differentiate between different patch categories. These can then be filtered quickly, excluded, or searched for, much as you would with a tag cloud. This is something you cannot accomplish on the hardware, and it makes it trivially easy to manage collections containing thousands of data dumps. Selecting one will usually send it to the device for immediate preview, avoiding the need to save the data on the external device. While you can't edit parameters within those patches, KnobKraft Orm does free you from the tyranny of using a device's limited storage and patch management.

KnobKraft Orm's best feature is the ability to add your own support for your favorite hardware, even when it's not a native part of the application. This is through what KnobKraft

Orm calls an adaptation, and there are already many. An adaptation is written in Python and exists as a script outside the (C++ written and compiled) main application. This is a much more powerful solution than using a static syntax such as JSON to handle the system-exclusive data, because Python lets you easily process the data. You still need access to the device's MIDI specification, but thanks to the excellent programming guide, turning the specification into an adaption is actually fun if you already have good knowledge of the MIDI device. An adaption then becomes just as integrated as the native device support, theoretically letting you manage every device in your studio – something that hasn't been possible since development on the venerable Sound Diver was abandoned in 2005.

Project Website
<https://github.com/christofmuc/KnobKraft-orm>



Regardless of the platform or license, KnobKraft Orm is the closest modern MIDI librarian to the wonderful and woefully deprecated Sound Diver.

Multi-user space station

Among sus

There can't be many people who haven't heard of the game Among Us, especially if you've been around young adults over the past 12 months. It's an ingenious 2D, top-down game set on a space station. As you might expect from a game with a young following, it is also fundamentally online multiplayer. At the start of each round, each player appears on the station in the cafeteria. One player is designated the imposter, and all the other players are regular crewmates, none of whom know who the imposter is. Crewmates need to perform tasks while the imposter attempts to "kill" them, but if anyone gets a whiff of who the imposter might be, they can summon a discussion, and everyone votes to try and tease the identity of the imposter from the

ship's roster. The non-imposter crewmates win if they correctly determine the imposter's identity.

Among sus is an open source reimagining of Among Us built purely as an interactive text game run through a server for multiplayer access. With the server running, anyone can connect with a simple Telnet or Netcat (nc) command, and each new connection becomes a new player. The first player can initiate the game, pulling everyone into the text description of the cafeteria. Players then use commands such as `go reactor`, `examine room`, or `check tasks` to interact with the game. Just like in the original, one player is dubbed the imposter. Players can "kill" another player when they're in the same location, but when a body is detected, the discussion

```

- Grakes
[Grakes] has joined the lobby
[Paul] has joined the lobby
[Robert] has joined the lobby
[Paul]
[Robert]
[Game is starting]
-----
You are a crewmate, complete your tasks before everyone is killed.

You are in a spaceship, one of the crew of 4 people.
The tasks have been handed out and the daily routine is starting up, but there are rumors one of your fellow crewmates isn't a crewmate at all!
! Help
! commands: help, examine room, go (room), murder crewmate, if
! commands in this room: press emergency button
! examine room
You are standing in the middle of the cafeteria, in the center there's an emergency button
You can move to: lobby, admin, weapons
You also see Paul in the room with you
You also see Robert in the room with you
! west
! avoid destruction
! go west
! INLAND MOVEMENT
-----
nc 127.0.0.1 1234
Welcome player 1!
Enter your name:
> Paul
[Paul] has joined the lobby
[Robert] has joined the lobby
-----
[Game is starting]
-----
You are a crewmate, complete your tasks before everyone is killed.

You are in a spaceship, one of the crew of 4 people.
The tasks have been handed out and the daily routine is starting up, but there are rumors one of your fellow crewmates isn't a crewmate at all!
-----
nc 127.0.0.1 1234
Welcome player 2!
Enter your name:
> Robert
-----
[Game is starting]
-----
You are a crewmate, complete your tasks before everyone is killed.

You are in a spaceship, one of the crew of 4 people.
The tasks have been handed out and the daily routine is starting up, but there are rumors one of your fellow crewmates isn't a crewmate at all!
-----
nc 127.0.0.1 1234
Welcome player 3!
Enter your name:
> Grakes
-----
[Game is starting]
-----
You are a crewmate, complete your tasks before everyone is killed.

You are in a spaceship, one of the crew of 4 people.
The tasks have been handed out and the daily routine is starting up, but there are rumors one of your fellow crewmates isn't a crewmate at all!
-----
nc 127.0.0.1 1234
Welcome player 4!
Enter your name:
> Paul
-----
[Game is starting]
-----
You are the imposter, kill everyone without getting noticed.

You are in a spaceship, one of the crew of 4 people.
The tasks have been handed out and the daily routine is starting up, but there are rumors one of your fellow crewmates isn't a crewmate at all!
-----

```

Among sus is a text-based version of Among Us with a single concession to graphics. Type `map` to see an ASCII visualization of the ship.

stage is triggered. This is where players can raise their suspicions and ultimately vote for who they believe to be the imposter. As with the original game, whoever they decide on as the imposter will be ejected from the ship and loses the game. The crew can only win if they find the imposter and complete all their tasks. Similarly, the imposter can only win by removing all their crewmates from the game. This text reinterpretation may have started as a joke, but it's actually strangely compelling and addictive!

Project Website

<https://git.sr.ht/~martijnbraam/among-sus>

Descent player

D2X-XL

There are many modern compatibility conversions of old classic games. They often require the original data files but have a re-engineered game engine written for modern compilers and graphics engines. Many are brilliant, but most of the interest in playing them comes from nostalgia and revisiting a misspent youth. Very few can live up to modern playability standards, but Descent and its immediate sequel, Descent 2, are possible exceptions. Originally released in 1995 and 1996 for the PC and original PlayStation, both games were the first to realize the potential in giving the player "six degrees of freedom." Doom, which had revolutionized the first-person shooter, offered four or five degrees of

freedom: movement left and right (1), movement forwards and backwards (2), movement up and down (3), and rotation (4). You could jump and look up or down, which was a fifth degree of freedom, though limited by gravity pulling the player back to earth.

The Descent games solved this limitation by making the player a spaceship in zero-gravity tubes. This meant you could freely point in any direction and rotate around the forward axis, creating a sixth degree of freedom. Add this to the freshly invented dual-axis PlayStation controller for analog control of the principle axes, and you had a game with an unprecedented level of immersion and challenge. The levels had you flying through tubes in underground



There are two popular Descent clones, but we've found D2X-XL builds and works best on Linux.

bases and space stations, often using many axes at once to stalk your prey, save captives, and blow up the base. Despite more modern sequels, including Overload from some of the original Descent developers, the Descent experience is still difficult to beat. Which is why in this case the D2X-XL compatibility conversion, along with modern graphics and mod support, is more than your average nostalgia trip. It ups the graphics ante, adds better controller support, and even insane multiplayer support.

Project Website

<https://github.com/arbruijn/d2x-xl/>

Write, share, and publish documents with HedgeDoc Markdown Magic

HedgeDoc lets you write documents collaboratively in Markdown and publish them online. BY MARCO FIORETTI

Markdown syntax [1] provides an easy way to create rich text documents out of plain text files. The magic of Markdown (see the “Markdown” box) is more than mere paragraphs, titles, and subtitles – you can also work with high-end features such as slideshows and embedded videos.

Several available tools help users create, read, and edit Markdown files, but one of my favorites is HedgeDoc [2], a web-based Markdown editor and online publisher. HedgeDoc, the open source version of the online editing service HackMD [3], lets you create Markdown documents and publish them online from any browser. In addition, HedgeDoc lets you work collaboratively, making it useful for schools or organizations that need to create shared documents in a reusable format. HedgeDoc is also a privacy-friendly alternative to Google Docs or other commercial document services.

Markdown

Markdown files are plain text files, but they allow much more structure than an ordinary plain text file. I say “structure” instead of “formatting”; Markdown’s real power lies in not caring about the actual formatting.

With Markdown, you add typographic detail to your text in the most generic way. Markdown makes it easy to write or generate text that is easily converted to any other document format. With just one click or command, a Markdown file can become a web page, a PDF file, a database record, a note in a mind-mapping application, and much more.

The Markdown cheat sheet in Figure 1 shows how Markdown works. For instance, a line starting with one or more hash characters denotes a chapter or subchapter header, text inside two asterisks is bold, square brackets followed by parentheses enclose a hyperlink, and so on.

Workspace

HedgeDoc is a website, usable from desktop, tablet, and mobile devices, that looks and acts like a text editor or simple word processor. While HedgeDoc is reminiscent of Google Docs and Etherpad, HedgeDoc has the ability to show both the actual Markdown source code and the HTML web page preview (Figure 2).

HedgeDoc also lets you choose the privacy settings for each Markdown document, which HedgeDoc calls a “note.” Privacy settings range from *Freely* (everyone can read, write, and eventually publish the note) to fully *Private* (Figure 3). If you set a note’s permission to *Editable*, all you need to do to invite others to work on the note is to send them the note’s URL. On the other hand, if you want the note to be read-only, you can share it by pressing the *Publish* button and using the URL it generates.

HedgeDoc’s user interface (Figure 4) is as simple as it is functional. The toolbar located above the text input offers all the common formatting

Example	Syntax
Header	# Header
• Unordered List	- Unordered List
1. Ordered List	1. Ordered List
<input type="checkbox"/> Checklist	- [] Checklist
Blockquote	> Blockquote
Bold	**Bold**
<i>Italicize</i>	*Italicize*
Strikethrough	~~Strikethrough~~
19 th	19 th
H ₂ O	H-2~0
<u>Underlined text</u>	==Underlined text==
Highlighted text	==Highlighted text==
Link	[link text](https:// "title")
Image	![image alt](https:// "title")
Code	`Code`
<pre>1 var i = 0;</pre>	<pre>'''javascript var i = 0; '''</pre>

Figure 1: The Markdown syntax supports most of the elements required by complex documents.

options, from bold and italic to lists, links, images, and tables. While these tools are great for beginners, expert Markdown users will rarely use them because typing the Markdown codes is faster. However, the tools do come in handy for all users when it comes to making tables and formatting images.

Above the toolbar, you'll find the workspace mode buttons. There are two modes for all device types: View and Edit. Clicking on the *View* button (the eye icon) displays only the formatted HTML text, and the *Edit* button (the pencil icon) displays only Markdown text. On a desktop or tablet (sorry, mobile devices), you have a third option: the *Both* button (the split-screen icon) shows you both the HTML and Markdown side-by-side. Rounding out the display modes are the half-moon icon, which lets you toggle between night (the default) and day mode for the menus and the HTML pane, and the question mark icon for help and documentation.

On the right side of the top bar, you'll find *New* and *Publish* buttons, a drop-down menu with some essential functions, and the *Online* button, which tells you how many users are working on the current document (more on this later).

Drop-down menu options of interest include *Revision*, *Clipboard*, download options, and *Slide Mode*, which I'll cover in the "Slide Show" section. Selecting *Revision* shows a note's existing revisions, with all changes highlighted, plus the option to download any revision. *Clipboard* lets you copy text taken from another source (a web page or document) and then paste it into your note in Markdown format.

You also have the option to download a note on your computer, saving it as Markdown, HTML, or Raw HTML. I strongly recommend saving each note's Markdown source code on your computer, unless you choose to use a script, which I'll address at the end of this article.

In my opinion, two things that could be improved in HedgeDoc's user interface are located in the toolbar. Resizing the browser window below a

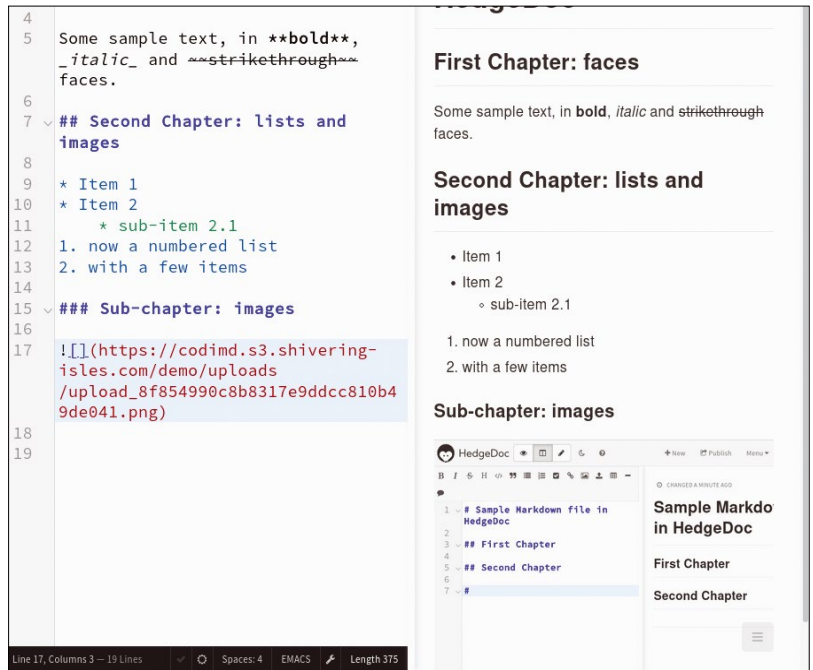


Figure 2: HedgeDoc let's you see both the Markdown source (on the left) and rendered HTML (on the right).

certain width makes some or all elements of the top buttonbar disappear. Also, the button that lets you switch between night and day mode for the Edit view is located at the very bottom of the window, instead of at the top for the other two views.

TOC, Two Ways

HedgeDoc offers two table of contents (TOC) options. You can embed a clickable TOC anywhere in your Markdown note using the [TOC] syntax (Figure 5, item 1). Alternatively, you can use HedgeDoc's autogenerated TOC. Pressing the hamburger icon in the bottom right corner of the HTML pane (Figure 5, item 2) opens a collapsible outline. Unlike the embedded TOC, the autogenerated TOC is a part of the HedgeDoc interface, which lets you navigate anywhere in the document down to three levels of headers without adding the [TOC] syntax to your Markdown code.

Choice of Editors

HedgeDoc can emulate three different editors: Sublime (the default), Emacs, and Vim. To switch between editor modes, click on the editor button (EMACS in Figure 5) in the toolbar at the bottom of the Markdown editor pane, and select your editor

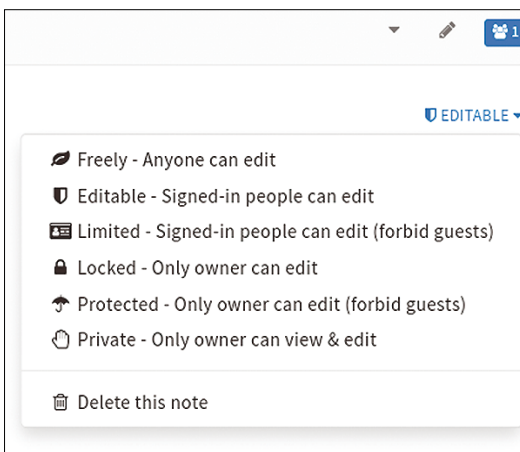


Figure 3: As a collaborative editing tool, HedgeDoc provides several levels of access to each note.

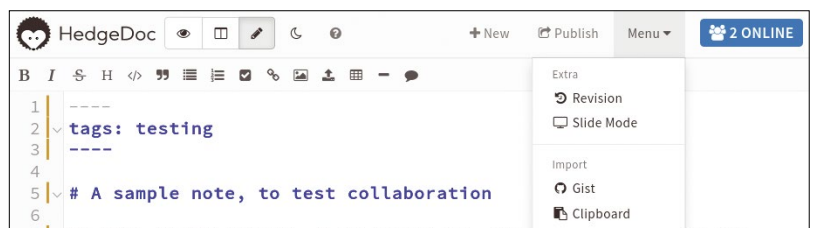


Figure 4: The HedgeDoc interface: Everything you need to edit and publish text, and nothing more.

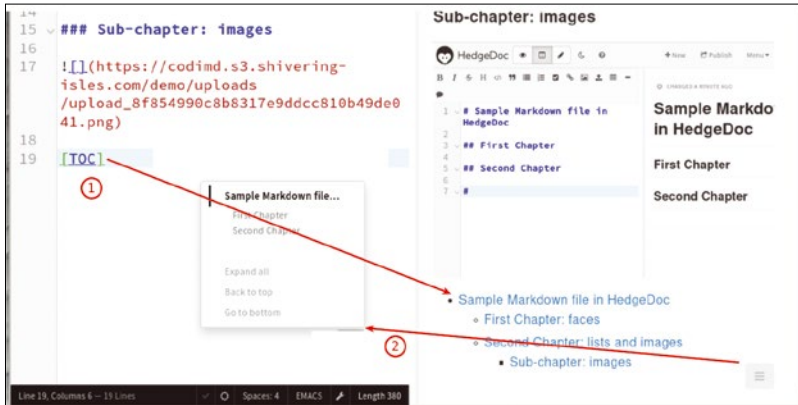


Figure 5: You can embed a TOC as HTML code anywhere in a HedgeDoc note (1) or use HedgeDoc’s autogenerated TOC (2).

of choice from the drop-down menu that opens. Besides having its own keymap, each editing mode can collapse chapters or subchapters (when you click on the arrow symbol to the left of the header). The editors also support auto-completion for several types of markup. For example, if you type #, you will get the pop-up window with auto-completion hints as shown in Figure 6.

Special Formatting

In addition to regular text, HedgeDoc lets you format source code, emojis, and images using Markdown syntax. To format software code, you use three backticks (```) at the beginning and end of a code block. The code will be highlighted according to the language used. For emojis, you type : at the beginning of the line. An ! at the beginning of a line tells HedgeDoc to insert the code to embed an image in that position.

When embedding an image, its source location affects how HedgeDoc embeds the image. If the image is already online (on the same or another server), you just insert the image’s URL in the Markdown code that HedgeDoc auto-inserts after the exclamation mark. If the image is on your computer, click on the *Upload Image* button in the toolbar and upload the image to the HedgeDoc server (or to an alternative lo-

cation as described in the “Self-Hosting” section). Alternatively, you may drag and drop an image from your file manager directly to where you want it to appear in the note.

At time of writing, HedgeDoc 1.8.2 still lets you embed YouTube videos, as follows:

```
{%youtube ID_NUMBER_OF_YOUTUBE_VIDEO %}
```

However, this specific markup is officially deprecated. It should soon be replaced by embedding “plain links” (the meaning of which is currently unclear).

Metadata

Similar to most Markdown-capable applications, HedgeDoc supports adding metadata to files to set browser behavior. For the metadata to be recognized, it must be placed in a special section, called “frontmatter,” at the beginning of each file.

You use YAML to format the frontmatter. The syntax to mark the start of the frontmatter is three dashes, and three dashes mark the end of the frontmatter. The YAML metadata goes inside, with one key/value pair on each line, separated by a colon, as follows:

```
---
title: real title
tags: tutorial, open source, writing
description: Another useful tutorial
robots: nofollow
---
```

HedgeDoc will use the `title` value as the HTML note’s title, overriding the first level 1 heading (which would be used if there were no metadata). These tags can be used by any system (including search engines) that catalogs the notes by topic. The values of the `description` and `robots` keys will be added in the HTML note’s header, to make search engines properly index the note and the links it contains. However, you can set `robots` to `noindex` if you don’t want the page indexed. Set `robots` to `nofollow` if you do not want the pages linked

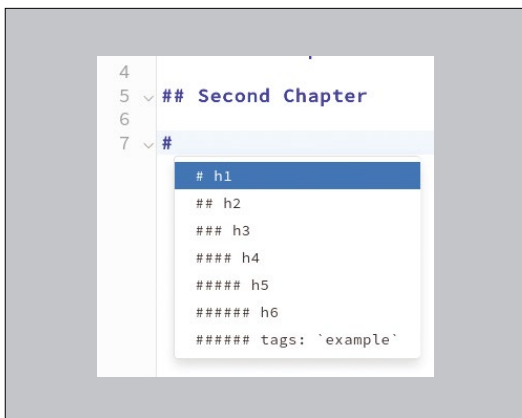


Figure 6: Auto-completion makes writing Markdown even faster.

Listing 1: PlantUML Sequence Diagram Format

```
01 @startuml
02 Alice -> Bob : Hello, is the store open?
03 Bob -> Alice: Yes, it is. What do you want to buy?
04 Alice -> Bob : I want to buy some jeans.
05 Alice <- Bob : Please tell me the color.
06 Alice -> Bob : Black
07 Alice <- Bob : Now please tell me the size.
08 Alice -> Bob : 42
09 @enduml
```

from the current page to be indexed. In addition to these metadata tags, HedgeDoc recognizes many other tags. For more information, consult the documentation on HedgeDoc's website [2].

Diagrams and Mathematical Expressions

In addition to text, source code, and images, HedgeDoc can also format diagrams and charts, as well as complicated mathematical expressions. In my PlantUML tutorial [4], for example, I showed how to create a sequence diagram out of a text file as shown in Listing 1.

With HedgeDoc, you can obtain the same results by inserting sequence diagram text, but you use ````sequence` at the beginning instead of `@startuml` and ````` at the end instead of `@enduml`, as shown in Listing 2.

The `sequence` keyword following the backticks at the beginning of Listing 2 is what actually creates the diagram shown in Figure 7. If you left `sequence` out, HedgeDoc would treat the diagram text as source code. To learn about all the diagram types that HedgeDoc supports, see the many examples available online [5].

HedgeDoc also supports mathematical expressions by using the MathJax library [6]. MathJax reads plain text descriptions in MathML, LaTeX, or AsciiMath formats and renders them as complex mathematical expressions. Depending on the situation, the expression is rendered using a combination of HTML code with special CSS stylesheets, web fonts, MathML code, or scalable vector graphics (SVG).

Because MathJax support is built into HedgeDoc, you can easily enter a mathematical expression in any of the source formats and immediately see how it looks. Figure 8 demonstrates the rendered expression from the LaTeX expressions shown in Listing 3.

The value HedgeDoc offers in rendering

mathematical expressions can't be overestimated.

Math students and teachers worldwide have been using LaTeX formulas for years, with thousands of expressions scattered inside countless LaTeX files.

HedgeDoc offers an easy way to write and publish any paper based on these formulas.

Slide Shows

At time of writing, slide show support in HedgeDoc is in beta. However, as demonstrated by a screenshot from the official HedgeDoc slide show demo [7] (Figure 9), it is already possible to

Listing 2: HedgeDoc Sequence Diagram Format

```
```sequence
Alice->Bob: Hello, is the store open?
Bob->Alice: Yes, it is. What do you want to buy?
Alice->Bob: I want to buy some jeans.
Bob->Alice: Please tell me the color.
Alice->Bob: Black
Bob->Alice: Now please tell me the size.
Alice->Bob: 42
```
```

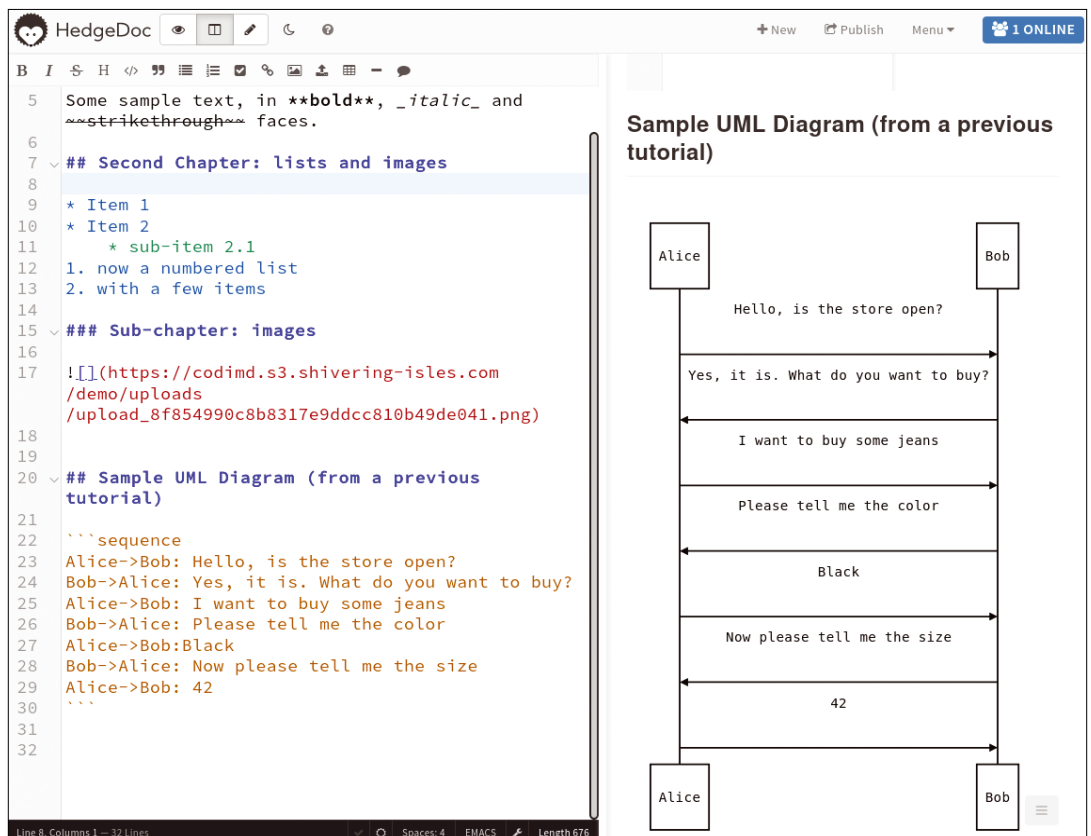


Figure 7: The Markdown syntax in the left pane generates the sequence diagram on the right.

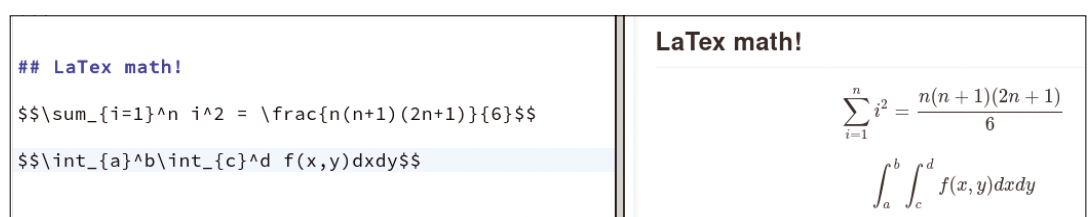


Figure 8: The Markdown syntax (left) results in a beautiful formula (right).

Listing 3: LaTeX Expressions

```


$$\sum_{i=1}^n i^2 = \frac{n(n+1)(2n+1)}{6}$$


$$\int_a^b \int_c^d f(x,y) dx dy$$


```

turn a HedgeDoc note into a simple slide show viewable in any browser. To create a slide show, you must declare

the note type as `slide` in the frontmatter. You can specify how the slides should look in the frontmatter using the following options:

```

slideOptions:
  transition: fade
  theme: white

```

For more details on slideshows, please see the HedgeDoc documentation and demo.

Collaborative Editing

After logging in (see the “Self-Hosting” section for information on how to create an account), HedgeDoc greets you with the default homepage (Figure 10). From here, you can create a new note. You can also search for an existing note using tags (from the note’s frontmatter) or generic key-

words (from the note’s text), with the results ordered by time or title.

If you are working collaboratively with other users, perhaps simultaneously, on the same note, you will see something similar to the two browser windows in Figure 11. In the left window, after logging in as a registered user (*mfioretti*), I created a sample note as a test. Then I opened the same note in another window but this time as an anonymous guest. Note that HedgeDoc assigns guests an arbitrary name (e.g., *Pena* in Figure 11).

The large blue button in the top right corner of each window lets you know who is currently working on a note. Clicking this button lists all users currently working on the active note, with each registered assigned a

color (light brown for *mfioretti*). As shown in Figure 11, the paragraphs I wrote working in the left window as *mfioretti* are marked with a vertical light brown bar to the right of the line number. Both guests and registered users see changes made by each other in real time.

Figure 11 also highlights a problem with HedgeDoc’s interface. Based on this screenshot, it looks like only registered users have the *Publish* button. This is not true. If a window (on the right Figure 11) is below a certain width, some elements of the top toolbar disappear. With a wider screen, I would have seen two identical toolbars; even the anonymous guest should have the *Publish* button because I had set the note’s permission to *Freely*, which gives all users permission to do whatever they want with the note. In this example, the only difference between a registered user and an anonymous guest is that only the registered user, who created the note, can change the permissions.

Self-Hosting

Instead of using the web-based app, you can self-host HedgeDoc. At time of writing, the current stable 1.x version is available in several formats, including a Cloudron app [8]. However, the simplest way to host a standalone HedgeDoc instance appears to be the official Docker images [9] (although they are only available for the AMD64 architecture). Almost all the screenshots in this tutorial come from HedgeDoc v1.8.2 running in a Docker container on an Ubuntu 21.04 system.

The recommended way to install a Docker container consists of downloading the sample configuration file (in YAML format) from the HedgeDoc Docker page [10], customizing it according to your needs, and then launching Docker with it. As an example of what you can do with this configuration file see the excerpt shown in Listing 4.

To summarize the instructions in Listing 4: Because HedgeDoc needs a PostgreSQL database, you need to download an additional container for PostgreSQL. You then start both containers, making HedgeDoc listen on TCP port 3000. A complete configuration file can contain much more than what is shown in Listing 4.

The HedgeDoc Docker configuration page [10] describes all the variables you can set in the configuration file. I recommend first testing HedgeDoc without changing anything in the configuration file. Once you understand how it works, you can customize the variables.

To download the container and start it as described in the configuration file, run the `docker-compose` command:

```

#> sudo docker-compose up -f Z
/absolute/path/to/docker-compose.yml

```



Figure 9: You can also turn Markdown text into an online slide show.

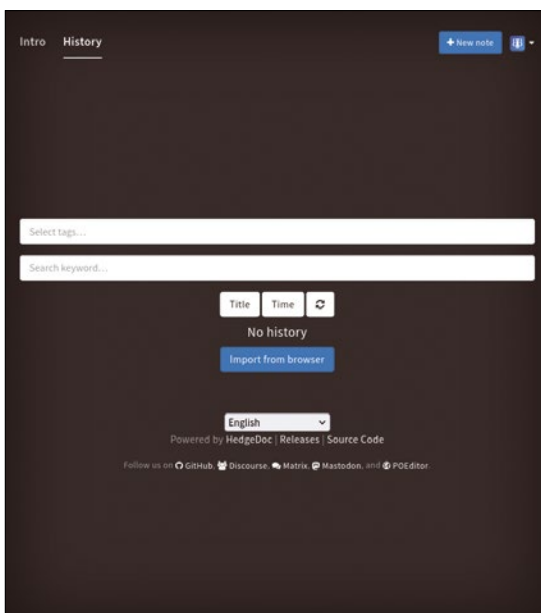


Figure 10: The HedgeDoc’s default homepage after login.

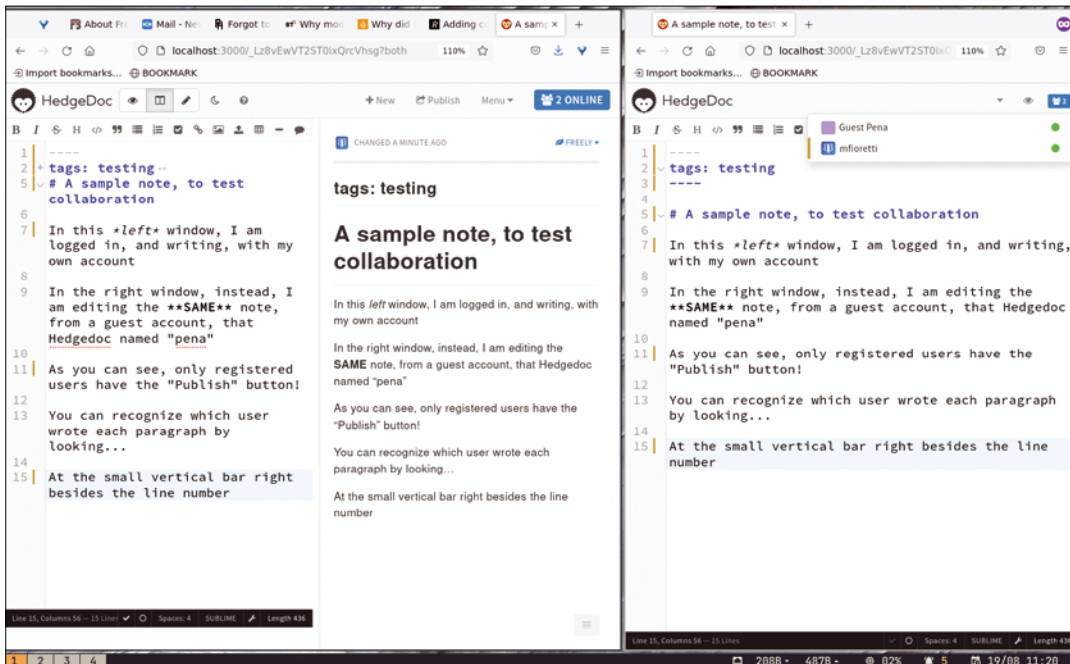


Figure 11: A HedgeDoc note edited by two users simultaneously: one logged in and one an anonymous guest.

Unless you see an error message, this should be enough to let you load the homepage of your very own HedgeDoc instance by pointing your browser to `https://localhost:3000` (if you installed it on your own computer) or replacing the `localhost` with the URL of the server where you started the container.

After setting up your instance, you need to create user accounts for you and your team if you want to keep private at least some of the notes that you and your team create. There are two ways to do this. One option is to have each user self-register from the HedgeDoc homepage and receive a password via email. For this option to work, you have to set both self-registration and email delivery in the container configuration file, as described in the HedgeDoc documentation.

The other method, which I personally prefer, involves logging in to the container that runs HedgeDoc and then adding as many users as needed with the command-line utility, `manage_users`, which is bundled with HedgeDoc. To create my personal account, after launching the container, I found the container's identification code with the `ps` option of the `docker` command (Listing 5).

Then I used that code (`572918fbff01`) to log in to the container, inside a Bash shell, as shown in Listing 6.

Then I ran `manage_users` to create a HedgeDoc account with my email address as the username and `testing` as the password. You can also use

Listing 4: `docker-compose.yml` (Excerpt)

```
version: '3'
services:
  database:
    image: postgres:9.6-alpine
    environment:
      - POSTGRES_USER=hedgedoc
      - POSTGRES_PASSWORD=password
      - POSTGRES_DB=hedgedoc
    ...
  app:
    # Make sure to use the latest release from https://hedgedoc.org/
    latest-release
    image: quay.io/hedgedoc/hedgedoc:1.8.2
    environment:
      - CMD_DB_URL=postgres://hedgedoc:password@database:5432/hedgedoc
      - CMD_DOMAIN=localhost
    ...
  ports:
    - "3000:3000"
```

Listing 5: Finding the Container Identification Code

```
#> sudo docker ps
[sudo] password for marco:
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS        NAMES
572918fbff01   quay.io/hedgedoc/hedgedoc:1.8.2    "/usr/local/bin/dock...
```

Listing 6: Logging in to the Container

```
#> sudo docker exec -it 572918fbff01 /bin/bash
root@572918fbff01:/hedgedoc#
#> root@572918fbff01:/hedgedoc# ./bin/manage_users --pass testing --add
      mfioretti@nexaima.net
Using password from commandline...
Created user with email mfioretti@nexaima.net
root@572918fbff01:/hedgedoc#
```

`manage_users` to delete users or reset their passwords.

Another thing to do the first time you log in to the container is to look at the internal configuration file, `files/config.json`. In this file, you can tell HedgeDoc to, among other things, do the following:

- Accept user logins via LDAP, Facebook, Twitter, or GitHub
- Store notes in Dropbox instead of locally
- Store images inside Imgur, Minio, or S3 accounts.

The HedgeDoc main configuration page [11] describes all the values for each of these options. Keep in mind that these options can all be overwritten with environmental variables, which normally have a `CMD_` prefix (e.g., `CMD_DB_URL` or `CMD_DO_MAIN` in Listing 4).

Scripting HedgeDoc

Besides managing users, you can also manage HedgeDoc notes using the command line or shell scripts. You can use `hedgedoc-cli` [12] to publish or save an existing note as a PDF file, to create new notes from existing local files, or to quickly import all the pads in an Etherpad account into HedgeDoc [13]. For instance, these two commands:

```
#> hedgedoc import Z
      /path/to/markdown/file.md abcdef
#> hedgedoc publish abcdef
      exampleid
```

create a note that would be editable at `https://HEDGEDOC_URL/abcdef` and readable at `https://HEDGEDOC_URL/s/exampleid`.

Conclusion

If you ask me, Markdown is so easy, powerful, and flexible that it is a shame more people do not al-

ready use it as their preferred text formatting language. HedgeDoc makes Markdown even easier to use, and possibly also to teach to others, so give it a try! ■■■

Info

- [1] Markdown: www.markdownguide.org
- [2] HedgeDoc: <https://hedgedoc.org/>
- [3] HackMD: <https://hackmd.io>
- [4] “Drawing Diagrams with PlantUML” by Marco Fioretti, *Linux Magazine*, issue 235, June 2020, <https://www.linux-magazine.com/Issues/2020/235/PlantUML-Diagrams>
- [5] Diagrams examples: <https://bramp.github.io/js-sequence-diagrams/>
- [6] MathJax: <https://www.mathjax.org/>
- [7] Slideshow demo: <https://demo.hedgedoc.org/p/slide-example>
- [8] HedgeDoc Cloudron app: <https://docs.cloudron.io/apps/hedgedoc/>
- [9] HedgeDoc Docker images: <https://quay.io/repository/hedgedoc/hedgedoc>
- [10] HedgeDoc Docker configuration: <https://docs.hedgedoc.org/setup/docker/>
- [11] HedgeDoc general configuration: <https://docs.hedgedoc.org/configuration/>
- [12] `hedgedoc-cli`: <https://github.com/hedgedoc/cli>
- [13] How to import Etherpad pads into HedgeDoc: <https://docs.hedgedoc.org/guides/migrate-etherpad/>

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>) and blogs about digital rights at <https://stop.zona-m.net>.



LINUX NEWSSTAND

Order online:
<https://bit.ly/Linux-Newsstand>

Linux Magazine is your guide to the world of Linux. Monthly issues are packed with advanced technical articles and tutorials you won't find anywhere else. Explore our full catalog of back issues for specific topics or to complete your collection.



#251/October 2021

Linux From Scratch

Building an operating system is not like compiling a desktop app. You'll need to create a complete development environment – and if you follow the steps carefully, you'll emerge with a deeper understanding of Linux.

On the DVD: Linux Mint 20.2 Cinnamon Edition and Garuda Linux KDE Dr460nized Edition



#250/September 2021

Inside the Kernel

The only real way to celebrate the 30th anniversary of Linux is to write about Linux itself – not the agglomeration of software we know as a Linux distro, but the real Linux – the beating heart in the center of it all: the Linux kernel.

On the DVD: AlmaLinux Minimal 8.4 and SystemRescueCd 8.03



#249/August 2021

Turn Your Android into a Linux PC

UserLAnd lets you run Linux applications on your Android phone – all without replacing Android OS.

On the DVD: openSUSE Leap 15.3 and Kubuntu 21.04 Desktop



#248/July 2021

Brain Tools

Sometimes you want the computer to think for you, and sometimes you want the computer to make you think. This month we present a selection of free Linux tools for learning and thinking.

On the DVD: Ubuntu 21.04 and Fedora 34 Workstation



#247/June 2021

Post-Quantum Encryption

Quantum computers are still at the experimental stage, but mathematicians have already discovered some quantum-based algorithms that will demolish the best of our current encryption methods. What better time to look for quantum encryption alternatives?

On the DVD: Knoppix 9.1 and ZORIN OS 15.3 Core



#246/May 2021

Faster Startup

Weary of waiting for a login window? Your driver-drenched Linux distro was configured for all systems, not for your system. This month we show you how to optimize your system for faster startup.

On the DVD: Manjaro KDE Plasma 20.2.1 and Clonezilla Live 2.7.1

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 <https://www.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.

NOTICE

Be sure to check the event website before booking any travel, as many events are being canceled or converted to virtual events due to the effects of COVID-19.

SeaGL

Date: November 5-6, 2021

Location: Virtual Event

Website: <https://seagl.org/>

The Seattle GNU/Linux conference (SeaGL) is a grassroots technical conference dedicated to spreading awareness and knowledge about the GNU/Linux community and free/libre/open source software and hardware. SeaGL strives to be welcoming, enjoyable, and informative for professional technologists, newcomers, enthusiasts, and all other users of free software.

SC21

Date: November 14-19, 2021

Location: St. Louis, Missouri and Virtual

Website: <https://sc21.supercomputing.org/>

Expand your knowledge, enrich your experiences, and network with others in the HPC community. Learn from world-leading scientists, engineers, and technologists, and experience a rich program of cutting-edge research, live demos, and talks that focus on emerging technologies in artificial intelligence, exascale computing, container software, and more.

Events

| | | | |
|--|----------------|-----------------------------|---|
| Cloud Native eBF Day North America | October 11 | Los Angeles, CA | https://events.linuxfoundation.org/ |
| O3DECon | October 11-12 | Los Angeles, CA | https://events.linuxfoundation.org/ |
| Open Networking & Edge Summit | October 11-12 | Los Angeles, CA and Virtual | https://events.linuxfoundation.org/ |
| Kubernetes on EDGE DAY | October 11-12 | Los Angeles, CA and Virtual | https://events.linuxfoundation.org/ |
| Cloud Native Security Conference North America | October 12 | Los Angeles, CA and Virtual | https://events.linuxfoundation.org/ |
| All Things Open | October 18-19 | Raleigh, NC and Virtual | https://www.allthingsopen.org/ |
| IEEE Quantum Week 2021 | October 18-22 | Virtual Event | https://qce.quantum.ieee.org/ |
| OpenPOWER Summit 2021 | November 4 | Virtual Event | https://cfp.openpower.foundation/summit2021/ |
| SeaGL (Seattle GNU/Linux Conference) | November 5-6 | Virtual Event | https://seagl.org/ |
| Open Source Strategy Forum | November 9-10 | New York, NY | https://events.linuxfoundation.org/ |
| SC21 | November 14-19 | St. Louis, MO | https://sc21.supercomputing.org/ |
| Linux Storage Filesystem & MM Summit | December 6-8 | Palm Springs, CA | https://events.linuxfoundation.org/ |
| DeveloperWeek Global: Enterprise | December 7-8 | Virtual Event | https://www.developerweek.com/global/conference/enterprise/ |
| Cloud Expo Europe | December 8-9 | Frankfurt, Germany | https://www.cloudexpo-europe.de/en |
| Big Data World Frankfurt | December 8-9 | Frankfurt, Germany | https://www.bigdataworldfrankfurt.de/ |
| Data Centre World 2021 | December 8-9 | Frankfurt, Germany | https://www.datacentreworld.de/data-centre-world-2020 |

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.

Contact Info

Editor in Chief

Joe Casad, jcasad@linux-magazine.com

Copy Editors

Amy Pettle, Aubrey Vaughn

News Editor

Jack Wallen

Editor Emerita Nomadica

Rita L Sooby

Managing Editor

Lori White

Localization & Translation

Ian Travis

Layout

Dena Friesen, Lori White

Cover Design

Lori White

Cover Image

© tarokichi, 123RF.com

Advertising

Brian Osborn, bosborn@linuxnewmedia.com
phone +49 8093 7679420

Marketing Communications

Gwen Clark, gclark@linuxnewmedia.com
Linux New Media USA, LLC
4840 Bob Billings Parkway, Ste 104
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

While every care has been taken in the content of the magazine, the publishers cannot be held responsible for the accuracy of the information contained within it or any consequences arising from the use of it. The use of the disc provided with the magazine or any material provided on it is at your own risk.

Copyright and Trademarks © 2021 Linux New Media USA, LLC.

No material may be reproduced in any form whatsoever in whole or in part without the written permission of the publishers. It is assumed that all correspondence sent, for example, letters, email, faxes, photographs, articles, drawings, are supplied for publication or license to third parties on a non-exclusive worldwide basis by Linux New Media USA, LLC, unless otherwise stated in writing.

Linux is a trademark of Linus Torvalds.

All brand or product names are trademarks of their respective owners. Contact us if we haven't credited your copyright; we will always correct any oversight.

Printed in Nuremberg, Germany by hofmann infocom GmbH.

Distributed by Seymour Distribution Ltd, United Kingdom

LINUX PRO MAGAZINE (ISSN 1752-9050) is published monthly by Linux New Media USA, LLC, 4840 Bob Billings Parkway, Ste 104, 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, 4840 Bob Billings Parkway, Ste 104, Lawrence, KS 66049, USA.

Published monthly in Europe as Linux Magazine (ISSN 1471-5678) by: Sparkhaus Media GmbH, Bialasstr. 1a, 85625 Glonn, Germany.

Authors

| | |
|--------------------|------------|
| Mats Tage Axelsson | 34 |
| Erik Bärwaldt | 26, 46, 58 |
| Zack Brown | 11 |
| Bruce Byfield | 6, 30, 38 |
| Joe Casad | 3 |
| Mark Crutch | 71 |
| Marco Fioretti | 88 |
| Jon "maddog" Hall | 73 |
| Sirko Kemter | 78 |
| Charly Kühnast | 41 |
| Vincent Mealing | 71 |
| Pete Metcalfe | 66 |
| Martin Mohr | 60 |
| Graham Morrison | 82 |
| Mike Schilli | 50 |
| Tim Schürmann | 74 |
| Ferdinand Thommes | 14, 42 |
| Daniel Tibi | 18, 24 |
| Jack Wallen | 8 |

Issue 253 / December 2021

OpenBSD

BSD often flies under the radar, but many devoted users believe it is the best free operating system in the world. Next month we take a look at the safe and popular OpenBSD variant as an alternative to Linux.

Approximate

UK / Europe Nov 06

USA / Canada Dec 03

Australia Jan 03

On Sale Date

Please note: On sale dates are approximate and may be delayed because of logistical issues.



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: <https://bit.ly/Linux-Update>

Image © Isselee Eric Philippe, 123RF.com



Custom Tailored Suit

Select essential parts for your mobile computing needs



Processor
Intel or AMD CPU



Graphics Card
Integrated or dedicated



System Memory
Upgrade any time



Data Storage
Fast, large and switchable



Network
Wireless, Cellular
or Gigabit LAN



100%
Linux

5

Year
Warranty



Lifetime
Support



Built in
Germany



German
Privacy



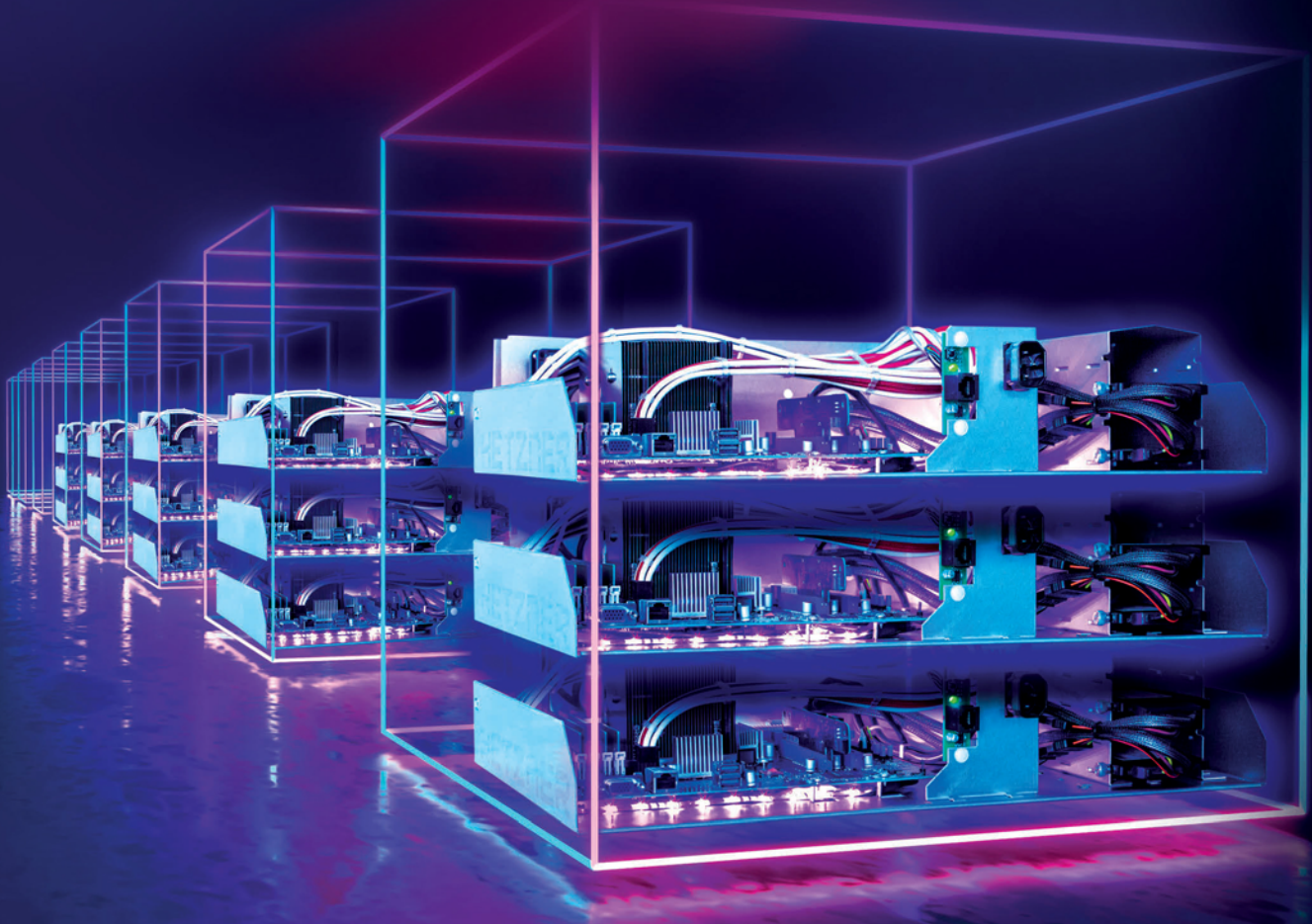
Local
Support

TUXEDO
COMPUTERS

tuxedocomputers.com

HETZNER

DEDICATED ROOT SERVER DESIGNED FOR PROFESSIONALS



Dedicated Root Server AX41-NVMe

- ✓ AMD Ryzen™ 5 3600
Simultaneous Multithreading
- ✓ 64 GB DDR4 RAM
- ✓ 2 x 512 GB NVMe SSD
- ✓ 100 GB Backup Space
- ✓ Traffic unlimited
- ✓ Location Finland and Germany
- ✓ No minimum contract
- ✓ Setup Fee £34



monthly from **£30**

Dedicated Root Server AX51-NVMe

- ✓ AMD Ryzen™ 7 3700X
Simultaneous Multithreading
- ✓ 64 GB DDR4 ECC RAM
- ✓ 2 x 1 TB NVMe SSD
- ✓ 100 GB Backup Space
- ✓ Traffic unlimited
- ✓ Location Finland and Germany
- ✓ No minimum contract
- ✓ Setup Fee £51



monthly from **£47**

All prices exclude VAT and are subject to the terms and conditions of Hetzner Online GmbH. Prices are subject to change. All rights reserved by the respective manufacturers.

www.hetzner.com