



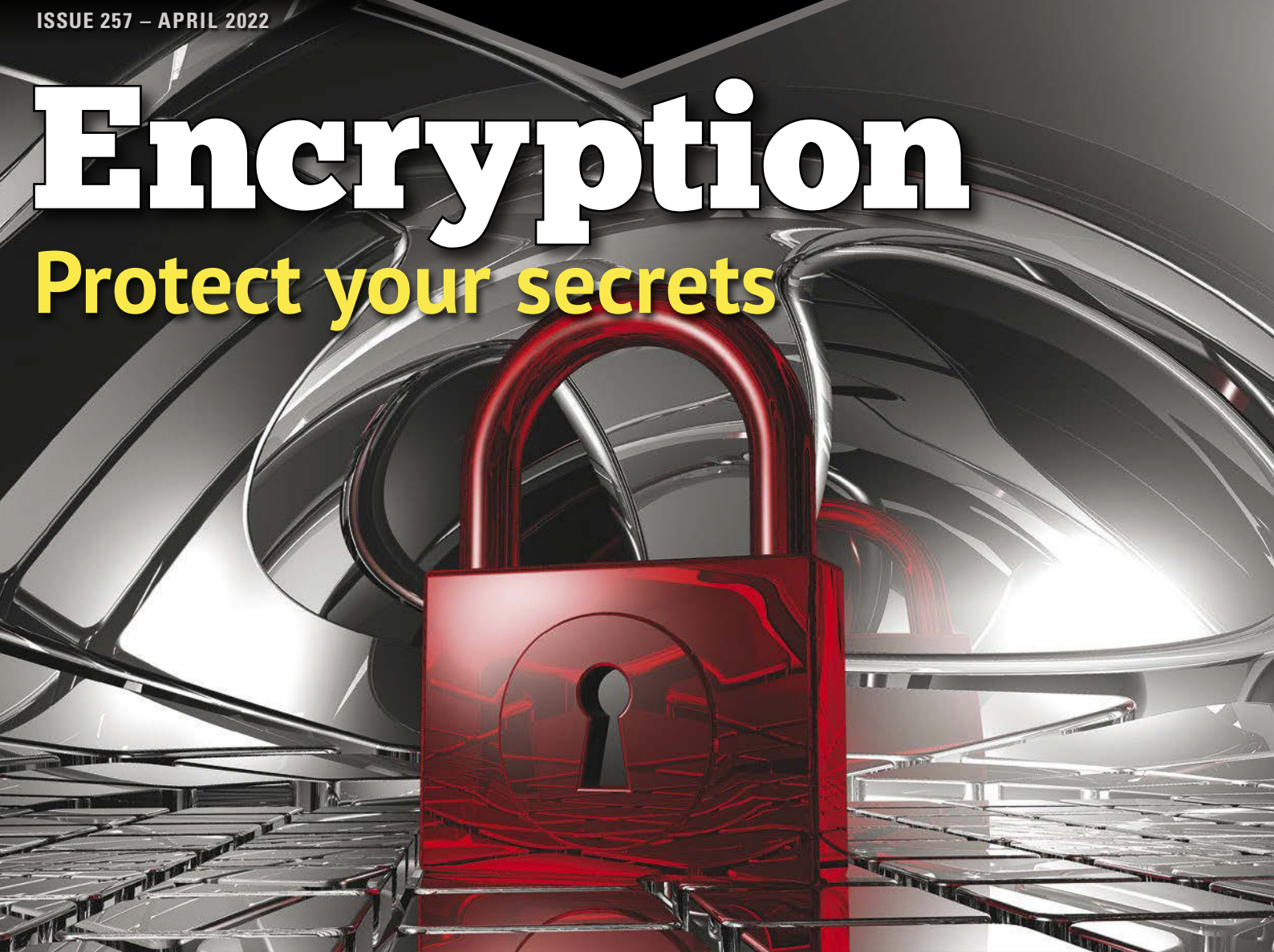
LINUX **PRO**

MAGAZINE

ISSUE 257 – APRIL 2022

Encryption

Protect your secrets



LINUXVOICE

Installing NVIDIA Drivers

Matplotlib: Draw better graphs with this powerful Python library

Reaper: Professional audio workstation for Linux

LibreOffice vs OpenOffice

10 FINE FOSS TOOLS



LINUX NEW MEDIA
The Pulse of Open Source



PEARL Edition

TUXEDO InfinityBook S 14



Intel Core i5-1135G7
Intel Iris Xe Graphics



1,1 kg light
16,8 mm thin



Metallic rosé special color
Trendy & exclusive



73 Wh battery
and Low Power display



100%
Linux

5

Year
Warranty



Lifetime
Support



Built in
Germany



German
Privacy



Local
Support

TUXEDO 18th
COMPUTERS ANNIVERSARY

tuxedocomputers.com

BEHIND THE NEAREST CLOUD

Dear Reader,

We in the Linux community are steeped in the conventional wisdom that Linux is much more secure than Windows. It is, of course, and it always has been, but then, that isn't saying much. The old versions of Windows that were around 20 years ago, when Linux was first starting to pick up steam, were really ridiculously insecure. Meanwhile, Microsoft kept bragging about how great Windows was and how you'd better be using it or you'd be left in the dust. The combination of Microsoft's engineering buffoonery and malicious marketing (they called Linux a cancer) gave the Linux community an attitude that remains to this day.

That attitude gave Linux developers an edge over the years when it came to competition and innovation, but the edge of attitude can be sharp and precipitous. Most Linux users are aware of the need to take standard security precautions, but there is also a tendency for denial among some of the Linux faithful about whether all that security advice really even applies to them. Ransomware? Privilege escalation? Must have been a Windows problem....

Well this isn't 2002 anymore. In many ways, Linux has already won the battle of the Internet. More Internet servers run Linux than any other system – even the Microsoft Azure cloud runs Linux servers. The pathetically low hanging fruit offered by systems like Windows 95 doesn't even exist anymore, which means that cybercriminals have to work harder to find a way inside. And once they start working harder, yes, they have found ways to get inside Linux.

The fact that Linux servers are so ubiquitous means that it is almost impossible for criminals to avoid them. In fact, it is getting to the point where cybercriminals *need* to attack Linux if they are going to continue to flourish. (OK, maybe "flourish" is too kind of a term, but you get my point.)

VMware's Threat Analysis Unit released a report recently about malware in Linux-based multi-cloud environments. The report [1], which is free for download if you enter your name and email address, focuses on ransomware and cryptomining attacks against cloud-based Linux instances. According to the report, "Threat actors know that current malware countermeasures are mostly focused on addressing Windows-based threats, leaving many public and private cloud deployments vulnerable to Linux-based attacks."

Info

[1] Threat Report: Exposing Malware in Linux-Based Multi-Cloud Environments: <https://www.vmware.com/resources/security/exposing-malware-in-multi-cloud.html>

As you might expect, out-of-date and poorly managed Linux systems are considered the most vulnerable, but too often it seems that we don't really know which systems are "poorly managed" until it is too late. The report states that the most common threats "take advantage of weak authentication, vulnerabilities, and misconfigurations in container-based infrastructures to infiltrate the environment with remote access tools (RATs)."

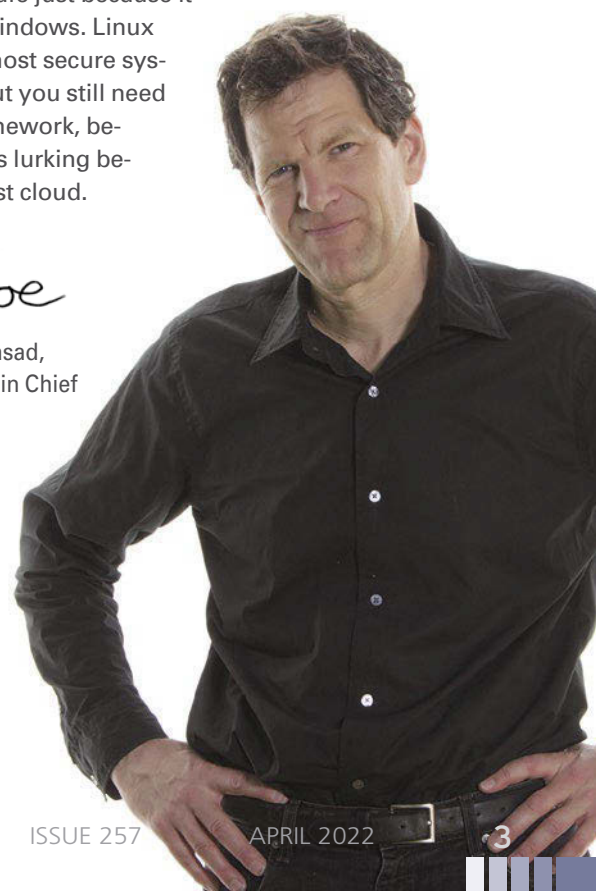
Cobalt Strike is a commercial penetration testing tool that is also used by cybercriminals. The report outlines how attackers have implemented a Linux version of the Beacon implant used with Cobalt Strike in order to port the attack to Linux-based systems. Similar conversions are happening elsewhere as criminals adapt to a Linux-focused Internet.

The usual advice applies here as well as everywhere: Use strong passwords and keep your system up to date to prevent attackers from getting a foothold. But these time-honored measures aren't enough to protect your Linux cloud environment. The VMware team recommends "...complete visibility into all workloads with detailed system context that makes it easier to understand and prioritize mitigation efforts." They also say you should use an Endpoint Detection and Response (EDR) system that will monitor, collect data, and send automated alerts if anything is awry.

The important thing is, pay attention, and don't assume your Linux is safe just because it is safer than Windows. Linux might be the most secure system around, but you still need to do your homework, because danger is lurking behind the nearest cloud.

Joe

Joe Casad,
Editor in Chief



ON THE COVER

38 **Reaper**

We test drive the Linux version of this leading digital audio workstation.

58 **LibreOffice vs. OpenOffice**

Clones or distant cousins?

68 **Rasp Pi Light Show**

Get the party started with DMX lighting technology.

76 **NVIDIA Drivers in Linux**

Help for the desperate.

88 **Matplotlib**

Get the answer in a glance with this Python visualization library.

NEWS

08 **News**

- Zorin OS 16 Educational Spin Now Available
- System76 Releases AMD-Powered Kudu Laptop
- Ubuntu Budgie Sets Its Sights on Gamers
- Linux Mint Edge Is Ready for the Newest Hardware
- Linux Kernel 5.17 Code Merge Window Is Closed
- Another Serious Flaw Found in All Major Linux Distributions

12 **Kernel News**

- The “Filesystem” System
- Maintaining GitHub Kernel Forks
- Going In or Going Out?

COVER STORIES

16 **Disk Encryption**

Encrypted volumes have long since ceased to be an exception or luxury. Corporate policies and compliance rules often demand encryption for critical data. This article looks at tools for disk encryption on Linux.

22 **The State of Email**

Email encryption is not that difficult – and it is more important now than ever before. We take a look at some important tools and trends in email encryption.

28 **Cryptography and Provable Security**

A concept called provable security brings the rigor of mathematics to the art of cryptography.

REVIEW

34 **Distro Walk – deepin**

Deepin offers a visually stunning OS with a few unique quirks.

IN-DEPTH

38 **Reaper**

Linux users looking for a professional digital audio workstation can now take advantage of Reaper’s large feature set.

44 **Command Line – watch and fswatch**

Two monitoring tools, watch and fswatch, let you gather system information from the command line.

46 **Charly’s Column – A FIGlet Farewell**

After two decades as a sys admin columnist, Charly bids a fitting farewell with a login banner created with FIGlet.

47 **Checkmk**

We’ll show you how to use the Checkmk open source monitoring tool to monitor your home router.

52 **Programming Snapshot – Rotating Photos with Go**

Cell phones often store photos upside down or sideways for efficiency reasons and record the fact in the Exif metadata. However, not all apps can handle this. Mike Schilli turns to Go to make the process foolproof.



Encryption

This month, we survey the state of encryption in Linux. We look beyond the basics to explore some of the tools and technologies that underpin the system of secrecy – and we show you what you need to know to ensure your privacy is airtight.

IN-DEPTH

58 LibreOffice vs. OpenOffice

Although LibreOffice and OpenOffice have a shared past, LibreOffice outstrips OpenOffice in contributors, code commits, and features.

MakerSpace

62 DIY Boombox

A HiFiBerry HAT, the matching system, and a Raspberry Pi combine with two old speakers to create a contemporary boombox.

68 Rasp Pi Light Show

The DMX protocol for controlling lighting technology was created for theater and events, but you can also use it for home automation. We show you how to control DMX devices with the Revolution Pi module.

LINUXVOICE

73 Welcome

This month in Linux Voice.

75 Doghouse – IoT Need For Openness

Closed IoT devices can use unexpected bandwidth “reporting home,” pointing to a need for free devices to allow the user more control over their household gadgets and WiFi use.

76 NVIDIA Drivers in Linux

A terminal-based solution helps ease the frustration of installing NVIDIA drivers.

80 FOSSPicks

This month Graham looks at Blender 3.0, woob, Fokus, bat, GNU poke, Bladecoder Adventure Engine, Crispy Doom, and more!

88 Tutorial – Matplotlib

Build illuminating graphs into your Python programs.



Linux Mint 20.3 Cinnamon and deepin 20.4

Two Terrific Distro on a Double-Sided DVD!



Linux Mint 20.3 Cinnamon
64-bit

Codenamed “Una,” Linux Mint 20.3 Cinnamon is the latest long term support (LTS) release for Linux Mint, with support until 2025. It features Cinnamon, Mint’s popular desktop environment. Like most standard Mint releases, Una is based on Ubuntu, which in turn is based on Debian. The Debian edition of the same release is available as Linux Mint Debian Edition (LMDE) 5, codenamed “Elsie.”

Cinnamon has a reputation for cautious innovation, with a few new features in every release, and Linux Mint 20.3 is no exception. Many tweaks have been made to the look and feel, with makeovers of existing themes and several new wallpapers, as well as the enlargement of titlebars and their buttons, with rounded corners adding a softer look. For those who prefer the old look, the Mint-Y-Legacy theme is available. In addition, a number of apps have also added search fields, including Hypnotix IPTV and the Sticky Notes applets. New to Cinnamon is the Thingy e-reader and a calendar and scheduler applet. By contrast, previous users of Mint might notice the disabling of the Snap Store for universal packages, which was done on the grounds that, while snap packages are open source, the code for the Snap Store is not.

Like previous Mint releases, Una is a distribution for all levels of users.



deepin 20.4
64-bit

Deepin is one of the leading distributions from China. Based on Debian’s Stable repository, deepin is prized for its Deepin Desktop Environment (DDE), which is built with Qt. DDE contains over two dozen unique applications ranging from standard items such as an editor, font installer, and package store to more unusual items like a remote assistant, repair guide, and presentation assistant. DDE is also frequently praised for its easy-to-follow installer and for its clean and beautiful design.

If you have used deepin previously, you will notice numerous small improvements in the 20.4 release, starting with a new default wallpaper. Also new in the release is an undo function (Ctrl+Z) for the trash icon and the display of an application window’s titlebar when hovering the mouse over the dock. The Control Center now supports biometric authentication, hibernation time settings, and more detailed settings for custom docks. Numerous bug fixes have also been made. As in previous releases, deepin 20.4 includes numerous tweaks not found in other desktops that extend functionality in small but useful ways.

Deepin has been accused several times of passing user information to the Chinese government, but none of these allegations have ever been proven. You will, however, have to check a box that says you agree to the deepin EULA and privacy policy. (See the article on deepin on p. 34 of this issue.)

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**

Global standard. Globally affordable.

Linux Professional Institute's mission is to promote the use of open source by supporting the people who work with it. That's why we have lowered the price of the world's most popular open source certification in more than 140 countries. Read what drives us at lpi.org/why.

Discover the new Linux Professional Institute exam pricing for your country at lpi.org/exam-pricing-2022.
More information and all LPI certifications on lpi.org



NEWS

Updates on technologies, trends, and tools

THIS MONTH'S NEWS

08

- Zorin OS 16 Educational Spin Now Available
- System76 Releases AMD-Powered Kudu Laptop

09

- Ubuntu Budgie Sets Its Sights on Gamers
- Linux Mint Edge Is Ready for the Newest Hardware
- More Online

10

- Linux Kernel 5.17 Code Merge Window Is Closed
- Another Serious Flaw Found in All Major Linux Distributions

Zorin OS 16 Educational Spin Now Available

Zorin OS 16 Education is a Linux distribution aimed at preschools as well as primary and secondary schools. Building on their already fantastic desktop experience, the developers have added a few tools to help educational institutions prepare students with the skills they need to navigate the waters of an ever-growing and complicated world.

One of the key pieces of software added is Kolibri, which aims to bridge the educational divide by providing access to an enormous library of educational content. Even better, an Internet connection isn't required to use the tool. With Kolibri, school administrators can either create their own curriculum or download materials from educational leaders such as Khan Academy, Open Stax, MIT, TED-Ed, and Sikana. Once downloaded, the content can be shared, peer-to-peer, over a local network.

Other educational apps include Minder (mind mapping), OpenBoard (whiteboard), Minuet (music knowledge), KTouch (typing), KTurle (educational programming environment), KBrunch (practice exercises with fractions), KWordQuiz (flashcards), and Step (physics experiment simulation). Beyond educational apps, Zorin OS Education includes the likes of Firefox, LibreOffice, and Gimp.

Download a copy of Zorin OS 16 Education at <https://zorin.com/os/education/>,

and read about what's new in the general Zorin OS 16 release at <https://blog.zorin.com/2021/08/17/2021-08-17-zorin-os-16-is-released/>.



Photo by Ben Mullins on Unsplash

System76 Releases AMD-Powered Kudu Laptop

System76 has finally brought back the popular Kudu laptop workstation powerhouse. This time around the laptop is equipped with a 3rd-gen AMD (Zen 3) Ryzen 9 5900HX H-class processor with 8 cores and 16 threads. The CPU runs at 3.3 GHz but can be boosted up to 4.6 GHz.

The Kudu is paired with NVidia RTX 3060 graphics and up to 64GB of Dual Channel DDR4 memory.

According to Ben Shpurker, Product Manager at System76, "This combination makes it the perfect machine for creating on the go."

The Kudu laptop has a 15-inch FHD matte display and a thin bezel, but for creators of all kinds, the laptop can manage up to three external displays with one HDMI port, one Mini DisplayPort, and one DisplayPort over USB-C.

Looking at ports, you'll find one USB 3.2 Gen 1 Type-A port, one USB 3.2 Gen 2 Type-A port, one USB 3.2 Gen 2 Type-C port with DisplayPort 1.4, one USB 2.0

Type-A port, one Mini DisplayPort 1.4 port, and one HDMI port with HDCP, as well as a headphone/microphone combo jack and a dedicated microphone jack.

Other features include a multitouch click pad, multi-color backlit US QWERTY keyboard, 2.5Gb Ethernet, WiFi 6, Bluetooth 5, 1.0M 720p HD webcam, 48.96 Wh Li-Ion battery, and 2 x M.2 SSD (PCIe NVMe only) for up to 4TB total storage.

The Kudu is now available to the general public and has a base price of \$1,799.

■ Ubuntu Budgie Sets Its Sights on Gamers

Ubuntu Budgie is already a well-designed Linux desktop distribution with a pleasant UI that makes interacting with Linux incredibly simple. And with the upcoming release of 22.04, the developers are adding a new layer of goodness to the platform.

First and foremost, the new release will include tools to vastly improve the gaming experience. Gamers will find tools such as MangoHUD (a Vulkan and OpenGL overlay for monitoring FPS, temperatures, CPU/GPU load, and more), CoreCtrl (allows you to control computer hardware with application profiles), Polychromatic and Open-RGB (RGP lighting management).

Next up comes the easy installation of apps such as Steam, Lutris, RetroArch, Discord, and OBS Studio. The developers are also mulling over adding GreenWithEnvy, the GTK system utility that is designed to provide information, control fans, and overclock NVIDIA video cards and GPUs.

Finally, the Budgie Welcome App will suggest the installation of gaming-specific libraries, based on any applicable hardware it detects.

All of the gaming features are currently in alpha but hopefully will be ready for the masses once the final release of Ubuntu Budgie 22.04 is made available this April. For those interested in testing the early release, you can download an ISO of the daily build for now (<https://cdimage.ubuntu.com/ubuntu-budgie/daily-live/current/>).



■ Linux Mint Edge Is Ready for the Newest Hardware

Linux Mint 20.3 is now widely available and ships with kernel 5.4. For anyone that uses the latest-gen hardware, that older kernel could be problematic. So, for those Mint users who do have hardware unsupported by the 5.4 kernel, there's now an option.

Linux Mint 20.3 Edge is a version of the distribution that ships with kernel 5.13.0-25, which means you'll find more new hardware supported. By employing this new kernel, Edge adds support for Apple M1 (initial support), preliminary Intel Alder Lake S graphics, AMD GPU Freesync/Adaptive-Sync HDMI, AMD Alderbaran accelerator, generic USB display, Loongson 2K1000, preparations for Intel discrete graphics, and Intel DG1 Platform Monitoring Technology.

As far as the user-facing features for Linux Mint Edge, they'll be the same as you'd find with the default release. You'll find the improvements for the Hypnotix IPTV player (including a new channel search function), a new document manager tool (Thingy), a search feature added to Sticky Notes, improvements to the default theme, and plenty of Cinnamon updates.

It's important to note, however, that Edge doesn't guarantee every piece of next-gen hardware will function properly. But if you are using a newer piece of technology, you'll have more luck with Edge than the standard Mint release.

Download an ISO of Linux Mint Edge at <https://www.linuxmint.com/edition.php?id=296>, and read the official release notes for Linux Mint 20.3 at https://www.linuxmint.com/rel_una_cinnamon.php.

MORE ONLINE

Linux Magazine

www.linux-magazine.com

ADMIN HPC

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

Mounting Compressed Archives as a User

• Jeff Layton

Why splat a compressed archive in your storage when you can just mount it like a storage device?

ADMIN Online

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

Secure SSH Connections the Right Way

• Simon Böhm

Admins use SSH, the proverbial Swiss Army knife of system management, for many of their daily tasks, but no matter how powerful the tool might be, it typically does not offer adequate protection. We look at ways to tighten SSH security.

Getting Started with Prometheus

• Stefano Chittaro

Prometheus is a centralized time series database with metrics, scraping, and alerting logic built in. We help you get started monitoring with Prometheus.

Hunt Down Vulnerabilities with the Metasploit Pen-Testing Tool

• Markus Feilner

The veteran Metasploit is by no means obsolete and is still used as a typical workflow to find and analyze security vulnerabilities in Windows 10 and Linux systems.

Linux Kernel 5.17 Code Merge Window Is Closed

Now that the holidays are in the rearview mirror, Linus Torvalds' family-related travels are over and he can go back to doing what he does best ... the Linux kernel. As expected, there are plenty of bug fixes, code cleanups, and new features. Although there aren't any absolute game-changers coming, there are some interesting fixes and additions.

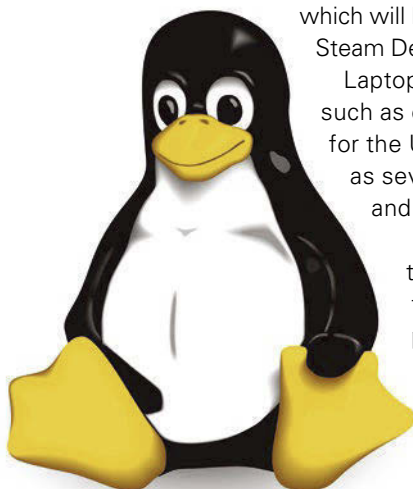
ARM64 will receive the addition of a Kernel Concurrency Sanitizer (KSCAN), which is a race condition detector. Also introduced is the initial work for the Scalable Matrix Extensions that will provide better and faster support for Matrix operations on ARM64 architecture. ARM is also gaining support for Snapdragon 8 (Gen 1) and X65 platforms.

AMD is bringing k10temp-based CPU temperature monitoring for its AMD Zen 19h line of CPUs. AMD also sees the inclusion of the AMD P-State driver, which will lead to better power efficiency on hardware such as the Steam Deck.

Laptops and tablets will see a good number of improvements, such as custom fan curve support for ASUS ROG laptops, support for the Universal Stylus Initiative and NVidia Tegra Tablets, as well as several performance improvements and bug fixes for sleep and sound issues on AMD laptops.

Other improvements and fixes include: initial support for Raptor Lake S graphics; Intel Alder Lap P graphics is now stable in the mainline kernel; Intel's Gen Icelake Graphics receives support for Variable Refresh Rate/Adaptive-Sync; EXT4 now uses the new Linux Mount API; performance increases for F2FS, Btrfs, and XFS; FS-Cache and CacheFiles modules have been rewritten; and a floppy disk hang bug has been fixed.

You can now download Linux kernel 5.17-rc2 for testing purposes (<https://www.kernel.org/>).



CC-BY-SA-4.0

Another Serious Flaw Found in All Major Linux Distributions

CVE-2021-4034 has been identified (<https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-4034>). This new vulnerability, named PwnKit, was tracked to an initial commit for pkexec from over 12 years ago. Because of the age of the flaw, every Linux distribution that depends on Polkit is affected.

The pkexec negotiates the interaction between privileged and unprivileged processes and allows authorized users to execute commands as other users. Researchers at Qualys discovered the pkexec command (<https://blog.qualys.com/vulnerabilities-threat-research/2022/01/25/pwnkit-local-privilege-escalation-vulnerability-discovered-in-polkits-pkexec-cve-2021-4034>) could be used by local attackers to increase privileges to root in Ubuntu, Debian, Fedora, and CentOS (and warn that it's most likely exploitable in other distributions as well).

It's important to understand that with this vulnerability an attacker can gain full root privileges on your system using just the default polkit configuration.

Of course, two of the major distributions, Ubuntu and Red Hat, have released patches for the vulnerability. Those patches are available for Ubuntu 14.04, 16.04 ESM, 18.04, 20.04, and 21.04, and Red Hat for Workstation and Enterprise products.

For those who use a distribution that has yet to patch this problem, a quick fix is to strip pkexec of the setuid bit with the command:

```
sudo chmod 0755 /usr/bin/pkexec
```

If you use one of the listed Ubuntu or Red Hat releases, make sure to update your systems immediately.



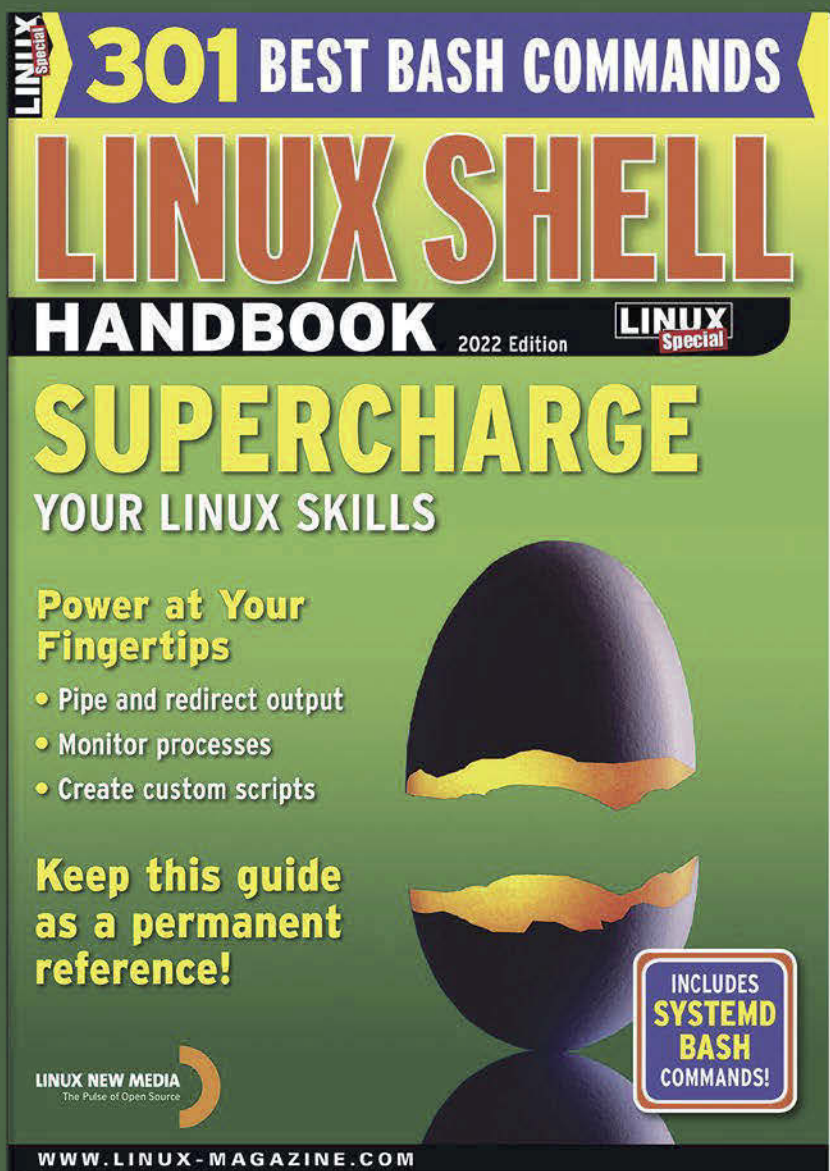
**Get the latest news
in your inbox every
two weeks**

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

THINK LIKE THE EXPERTS

Linux Shell Handbook 2022 Edition

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



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

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

ORDER ONLINE:

shop.linuxnewmedia.com/specials

Zack's Kernel News



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

By Zack Brown

The "Filesystem" System

Dov Murik from IBM posted some confidential computing (coco) patches. This is gussied-up marketing speak for "sandbox" (i.e., an isolated set of processes that implement security protections for whatever is going on inside). But, I'm not here to talk about coco. The interesting thing was that Dov's patches used SecurityFS to provide the interface between the user and the secure area, which turned out to be controversial. SecurityFS came along about 15 years ago in the 2.6 timeframe in response to a frightening proliferation of homespun filesystems. The idea was for security modules to use the same SecurityFS application programming interface (API) and just do whatever insanity was boiling their brains in the back end. SecurityFS simply provided a consistent gateway so everyone's various concepts could be straightforwardly navigated by everyone else.

Greg Kroah-Hartman replied to Dov:

"Why are you using securityfs for this?"

"securityfs is for LSMs [Linux Security Modules] to use. If you want your own filesystem to play around with stuff like this, great, write your own, it's only 200 lines or less these days. We used to do it all the time until people realized they should just use sysfs for driver stuff."

"But this isn't a driver, so sure, add your own virtual filesystem, mount it somewhere and away you go, no messing around with securityfs, right?"

But James Bottomley (also from IBM) took exception to Greg's statement. He replied, "we use it for non LSM security purposes as well, like for the TPM BIOS log and for IMA. What makes you think we should start restricting securityfs to LSMs only? That's not been the policy up to now." He added, "I really think things with a security purpose should use securityfs so people know where to look for them."

Greg replied that using SecurityFS for LSMs "was the original intent of the filesystem when it was created, but I guess it's really up to the LSM maintainers

now what they want it for." But he suggested, "Why not just make a cocofs if those people want a filesystem interface? It's 200 lines or so these days, if not less, and that way you only mount what you actually need for the system. Why force this into securityfs if it doesn't have to be?"

James pointed out that coco was only the first of potentially many uses of the particular feature they were implementing – transferring secret data into a virtual machine without the virtual machine itself being able to read it. And he said, "It's not being forced. Secrets transfer is a security function in the same way the bios log is."

Greg was very dubious and was surprised to hear that the BIOS log used SecurityFS. James confirmed, "Yes. It's under /sys/kernel/security/tpm0/. All the ima policy control and its log is under /sys/kernel/security/ima/ that's why I think declaring securityfs as being for anything security related is already our de facto (if not de jure) policy."

And James also added, "I know Al [Viro] likes this business of loads of separate filesystems, but personally I'm not in favour. For every one you do, you not only have to document it all, you also have to find a preferred mount point that the distributions can agree on and also have them agree to enable the mount for, which often takes months of negotiation. Having fewer filesystems grouped by common purpose which have agreed mount points that distros actually mount seems a far easier approach to enablement."

Greg did not find these arguments convincing. For one thing, he said, regardless of whether you make a new filesystem or use an existing one, you still need to document the data you store in it.

As for Linux distributions negotiating on mount point – or deciding to adopt the given feature in the first place – Greg felt this was normal. He said, "Enabling it does take time, which is good because if they do not think it should be present because they do not want to use it, then

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**.

it will not be, which means either they do not need your new feature, or you have not made it useful enough. So again, not an issue.”

Greg’s fundamental issue was security. The wider the array of SecurityFS users, the more likely each of those users would be to add more features to SecurityFS, exposing various powers to user space, and generally presenting a more tempting surface for attackers to look for security holes.

The discussion ended inconclusively. There is probably not much at stake either way – if coco uses SecurityFS it will simply be doing the same as other things that use SecurityFS. And if it makes its own filesystem, it will likewise be doing the same as other things. What’s interesting about a discussion like this is that it’s possible to witness development policies emerging and transforming.

Various developers have different interpretations about how the parts of the kernel should be used for further development. These differences don’t generally interfere with their ability to get patches in, so they can happily continue working, each believing different things about how to code for the kernel.

Then when something like IBM’s coco patches come along, suddenly everyone’s different ideas become relevant in a practical way. Sometimes a giant clash follows and sometimes not. In the current case, we can see this question: Should projects all create their own filesystems, or should similar users group themselves together to minimize filesystem proliferation? Eventually the question will probably have to be resolved. But for now, there are just these faint rumblings.

Maintaining GitHub Kernel Forks

Konstantin Komarov from Paragon submitted an NTFS patch – or rather, he requested that Linus Torvalds pull the patch from Paragon’s GitHub tree. Linus replied, saying:

“For github accounts (or really, anything but kernel.org where I can just trust the account management), I really want the pull request to be a signed tag, not just a plain branch.

“In a perfect world, it would be a PGP signature that I can trace directly to you

through the chain of trust, but I’ve never actually required that.

“So while I prefer to see a full chain of trust, I realize that isn’t always easy to set up, and so at least I want to see an ‘identity’ that stays constant so that I can see that pulls come from the same consistent source that controls that key.”


A few minutes later, he replied to his own email, saying he would just pull the tree this time to avoid delay, but he asked Paragon to set up PGP and start using signed tags moving forward.

On a related note, Linus added:

“I notice that you have a github merge commit in there.

*“That’s another of those things that I *really* don’t want to see – github creates absolutely useless garbage merges, and you should never ever use the github interfaces to merge anything.*

“This is the complete commit message of that merge:

```
Merge branch 'torvalds:master' 
into master
```

“Yeah, that’s not an acceptable message. Not to mention that it has a bogus ‘github.com’ committer etc.

“github is a perfectly fine hosting site, and it does a number of other things well too, but merges is not one of those things.

*“Linux kernel merges need to be done *properly*. That means proper commit messages with information about what is being merged and *why* you merge something. But it also means proper authorship and committer information etc. All of which github entirely screws up.*

“We had this same issue with the ksmbd pull request, and my response is the same: the initial pull often has a few oddities and I’ll accept them now, but for continued development you need to do things properly. That means doing merges from the command line, not using the entirely broken github web interface.”

There was no discussion or controversy following Linus’s posts. The interesting thing is that the signature issue is not hypothetical. Linus had to endure some legal hassles a few years back over licensing issues surrounding various patches. Consequently, patch signing became the standard way to trace all patches back to an origin point for licensing verification purposes.

Going In or Going Out?

An ongoing question in operating systems development is which is better: a microkernel or a monolithic kernel? The debate is a little odd since there is no clear definition of a monolithic kernel, although everyone seems to agree Linux is one.

The microkernel camp, in the open source world, is the GNU project with its Hurd microkernel. The idea of a microkernel is that user space should do as much work as possible and that only the bare minimum functionality should be implemented in the kernel itself. By doing it this way, the microkernel people argue that it's possible to produce a highly secure system because the kernel has fewer attack vectors. The smaller scope will mean less code churn over time and therefore fewer bugs that might be exploited by attackers.

The debate is odd because it frames the question in terms of efficiency and safety versus total chaotic insanity. In fact (and also the subject of this column), pretty much everyone wants their operating system kernel to be as small as possible for exactly those same reasons. Perhaps the debate really ought to be over exactly what belongs in user space and what doesn't.

For example, in the Linux Kernel Mailing List recently, Shijie Huang posted some non-uniform memory access (NUMA) patches. NUMA is a multiprocessing hardware architecture like symmetric multiprocessing (SMP). But whereas SMP hides the differences between the memory and CPUs that might be in use at any given time, NUMA can have some hardware that is faster than others. For consumer products, this means less expensive manufacturing. The question for operating system developers is how to support the allocation process so that faster hardware is used when needed and slower hardware is used when speed isn't as important.

Shijie's patch addressed the need for the OS to be aware of which hardware, and therefore which speeds, would be used for a given task. For example, a given file loaded into memory would have only one cached version of itself, in spite of the fact that there were multiple hardware speeds that might be relevant to using that cached version.

Shijie wanted users to be able to specify the NUMA needs of any given file, so the kernel would use the faster or slower hardware whenever it needed to access that file. In other words, he asked, "is it possible to implement the per-node page cache for programs/libraries?"

This, however, was easier said than done. As you might imagine, controlling which hardware would operate on a given file takes place in a fairly deep part of the kernel. As Matthew Wilcox put it in his response:

"At this point, we have no way to support text replication within a process. So what you're suggesting (if implemented) would work for processes which limit themselves to a single node. That is, if you have a system with CPUs 0-3 on node 0 and CPUs 4-7 on node 1, a process which only works on node 0 or only works on node 1 will get text on the appropriate node."

"If there's a process which runs on both nodes 0 and 1, there's no support for per-node PGDs. So it will get a mix of pages from nodes 0 and 1, and that doesn't necessarily seem like a big win."

Al Viro also replied to Shijie, saying, "What do you mean, per-node page cache? Multiple pages for the same area of file? That'd be bloody awful on coherency..."

Coherence is a crucial part of operating system design – it means that when the kernel writes to a piece of memory, subsequent attempts to read that same piece of memory should show the written value rather than show an older value that was there before the write. Loss of memory coherence in an operating system is sort of the equivalent of suddenly discovering the building you're in is engulfed in flames.

Shijie agreed that coherence was important, but he said that if the data was restricted to being read-only, coherence wouldn't be a problem because the values would never change. He posted a patch to demonstrate what he was talking about.

Linus Torvalds came in at this point, saying, "You absolutely don't want to actually duplicate it in the cache."

There was some implementation discussion between Shijie and various others, but at one point Linus came in again, a bit harder, saying:

"You can't have per-node i_mapping pointers without huge coherence issues."

"If you don't care about coherence, that's fine – but that has to be a user-space decision (ie 'I will just replicate this file')."

"You can't just have the kernel decide 'I'll map this set of pages on this node, and that other [set] of pages on that other node', in case there's MAP_SHARED things going on."

"Anyway, I think very fundamentally this is one of those things where 99.9% of all people don't care, and DO NOT WANT the complexity."

"And the 0.1% that _does_ care really could and should do this in user space, because they know they care."

"Asking the kernel to do complex things in critical core functions for something that is very very rare and irrelevant to most people, and that can and should just be done in user space for the people who care is the wrong approach."

"Because the question here really should be 'is this truly important, and does this need kernel help because user space simply cannot do it itself'."

"And the answer is a fairly simple 'no.'"

The discussion ended at that point, and Shijie went back to the drawing board.

For me, this is an example of a feature that could have gone into the monolithic Linux kernel. It would have performed a useful function for a certain bunch of users. It would have handled use cases that exist in the real world. It would have made the kernel as a whole slower, but it would be exactly the sort of thing the microkernel people would expect a monolithic kernel to include.

It seems very clear that proponents of both microkernels and monolithic kernels believe that if something can be done in user space, then it shouldn't be put into the kernel. And the point of difference becomes what exactly do we use to draw the line? At this point, it becomes largely a matter of personal opinion. Linus would probably say that it depends on a lot of factors. Can the feature run much faster in the kernel than in user space? Will the kernel code look sane and be maintainable? And so on. And probably the GNU Hurd people would have different opinions, as would the Redox people, the Genode people, the HelenOS people, and the contributors to various other microkernel projects. ■■■



2021
Archives
Available
Now!

CLEAR OFF YOUR BOOKSHELF WITH DIGITAL ARCHIVES

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

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

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

Encrypt your disks on Linux

Unreadable

Encrypted volumes have long since ceased to be an exception or luxury. Corporate policies and compliance rules often demand encryption for critical data. This article looks at tools for disk encryption on Linux.

By Martin Loschwitz

It's no coincidence that portable computers have pushed desktop PCs into the background over the past 10 years. Today, users only need desktop systems for computationally intensive work such as video rendering or games.

For everything else, even mid-range laptops are now perfectly adequate. But laptops also have one disadvantage: They are far easier to steal than a standalone PC. An appropriate insurance policy can cushion the cost of replacing the device in case of theft. However, it is not so easy to compensate for the loss of data.

Corporations and users can only protect themselves effectively against this kind of horror scenario by completely encrypting the data carriers in the device, from USB sticks to external hard drives. How can a Linux user best secure disk data by means of encryption? This article describes some leading encryption methods and tools for Linux.

Cryptsetup with LUKS

Just about everyone who has ever dealt with encryption on Linux will have come across the abbreviation LUKS [1], which stands for Linux Unified Key Setup. The LUKS standard describes what disk encryption should look like on Linux (Figure 1). LUKS is based on the Cryptsetup tool, which in turn uses the Dmccrypt kernel module of the Linux kernel to manage encrypted volumes.

At first glance, this sounds considerably more chaotic than it actually is – at least if you keep in mind how the Linux kernel has handled block devices and their drivers for decades. The block device layer of the kernel resorts to a trick, allowing different drivers to be connected in series in order to combine their functions.

Dmccrypt forms part of the block device layer. If the administrator instructs the device mapper (which includes LVM, for example) to prepend the Dmccrypt driver to a block device before accessing it, all Dmccrypt functions are available for the block device. In fact, Dmccrypt also implements its own basic encryption. However, these measures are not nearly

enough to meet today's requirements in the eyes of the kernel developers. Accordingly, they created the LUKS format, which standardizes all the functions needed for encryption and defines them as part of a header in the partition table. This also means that the definition of encrypted drives on Linux is independent of the distribution and vendor.

Integrated into the System

Today, Cryptsetup with LUKS support is included with all distributions. Most manufacturers have also integrated the tool directly into their setup routines. You can start encrypting when installing the individual directories, such as /home, or you can encrypt the entire non-removable disk.

Once you encrypt the disk, the operating system can no longer boot without the password. If you remove the data medium from the device and try to read it, you will only see a mess of data. It is unanimously understood that this way of using encrypted drives under Linux is by far the most secure approach today. It also uses hardware acceleration. If you don't have an Atom or another low-cost processor in the device, the CPU will probably come with built-in hardware support for various encryption algorithms.

Devices connected to the system via USB can also be encrypted with Cryptsetup and LUKS – just like a built-in NVMe drive. However, there are differences in the setup between the individual distributions, and different desktop environments also offer their own tools to operate Cryptsetup and LUKS. If you want to avoid an excursion to the command line, you will need to familiarize yourself with your system's defaults.

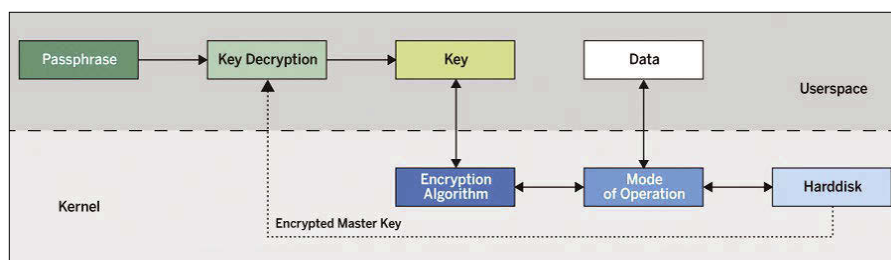


Figure 1: LUKS, the standard format for encrypted disks on Linux, works across distributions. © Suse



Of course, a combination of Cryptsetup and LUKS has one major disadvantage: It offers virtually no interoperability with other operating systems. You'll need a way to deal with a kind of chicken-and-egg problem. When LUKS and Cryptsetup were just gathering speed, there were already solutions that worked equally well on all the major operating systems. These alternative solutions are not as deeply integrated into the system, but they work across operating system boundaries, and not just on Linux.

Remembering Truecrypt

An article about disk encryption under Linux would be incomplete without mentioning Veracrypt [2] and its famous predecessor. Veracrypt emerged from Truecrypt in 2013. At that time, Truecrypt was considered by many observers in the Linux world to be the only valid alternative to the combination of Cryptsetup and LUKS on Linux.

Truecrypt development came to an end as a result of an audit in 2014, and it was the Truecrypt developers themselves who warned people not to use their own software. Shortly thereafter, the developers provided a new and final version of Truecrypt, which was massively limited in terms of its functionality. According to the developers, this final release was only intended to convert existing setups into Bitlocker setups with Microsoft's standard encryption.

The end of Truecrypt caused wild speculation in the community, which even considered the involvement of intelligence agencies. This speculation did not subside when additional audits completed retrospectively found no significant problems in the way Truecrypt worked. The actual reason for Truecrypt's end will probably never be clarified.

Truecrypt's Heir

Veracrypt, which had spun off from Truecrypt a year earlier as an independent project, became Truecrypt's de facto successor. All in all, Veracrypt is a highly functional data carrier encrypter from today's point of view.

One huge advantage that Veracrypt inherited from its ancestor is that it works on Windows, Linux, and macOS (Figure 2). It uses the existing infrastructure to the extent possible on the

various operating systems. On Linux, for example, Veracrypt directly relies on Dmccrypt. However, it also ensures that the containers and data carriers created remain mutually compatible across operating system boundaries.

An admin with the data medium and the corresponding key can mount existing devices with Veracrypt encryption on Windows, Linux, and macOS in the same way. Veracrypt is therefore particularly suitable for users who frequently switch from Linux to Windows and back, for example, because they use different systems on their home and work computers.

Plausible Deniability

Another decidedly handy feature in Veracrypt is the ability, inherited from its predecessor, to nest encrypted drives and thus achieve what security researchers call *plausible deniability*. Plausible deniability offers a means for concealing data without it looking like you are concealing data. The idea is to completely hide the existence of encrypted media. The attacker doesn't even know that data is being withheld and, according to the theory, doesn't even think of asking.

Veracrypt supports plausible deniability by creating multiple nested containers on a drive. The first container and its signature on the disk are easily recognizable from the outside. However, because this container contains only

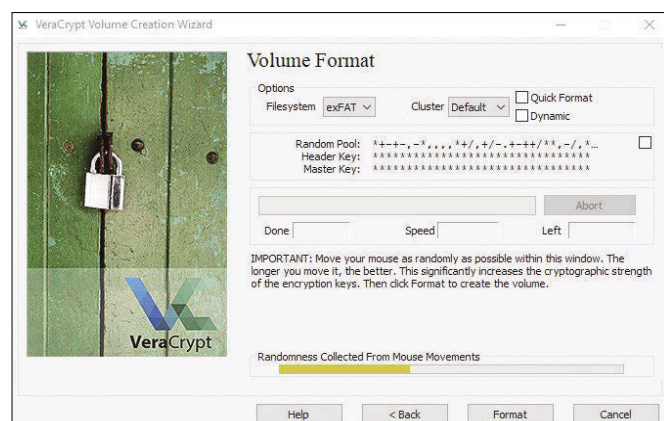


Figure 2: Encrypted drives created in Veracrypt can be used on Windows, MacOS, or Linux systems.

ostensibly relevant data when handled correctly, the owner of the drive can hand over the key (Figure 3). The second volume hides its header in the header of the first volume. You need to know that it exists to be able to open it, because from the outside, it looks like random data. And its existence can be easily explained by the existence of the first encrypted volume.

On Windows, Veracrypt now even goes one step further and can keep a hidden operating system in an encrypted drive. From the outside, this drive looks like a normal data medium, but under the hood, it hides a complete system partition.

Easy to Use

If you are looking for a quick solution for encrypted data media, Veracrypt is the right choice. After the install, which you handle with the package manager on Linux and with the installer on Windows and macOS, a wizard opens in the form of a graphical interface. The wizard guides you through the entire creation process. Veracrypt offers the option of creating encrypted containers on the file system or encrypting entire block devices.

No matter which approach you choose, any system administrator worth their salt should have no difficulty understanding and responding to Veracrypt's questions. If you have never dealt with disk encryption previously, you might not find Veracrypt particularly user-friendly. However, if you have been in

the saddle for a while longer, you will get along well with the program on all supported platforms.

GocryptFS

Another state-of-art tool for encrypted disks on Linux is GocryptFS [3]. GocryptFS also owes its existence to an earlier system that lost its shine for the FOSS community: EncFS.

EncFS came under fire when something happened to it during an audit that should never happen to an encryption solution. The testers detected a vulnerability in the source code that, in the worst case, allowed the key used for encryption to be computed. This danger existed with EncFS especially when attackers got their hands on several versions of the encrypted container. The actual key could then be found out by simple matching and an exclusion procedure. Basically, since this incident, EncFS has been under fire from everyone concerned with security and encryption (Figure 4).

However, Austrian Jakob Unterwurzacher found the idea behind EncFS so convincing that he set about creating a new implementation. In doing so, he worked with the Go programming language, which not only explains the program's name, but also translates to considerable speed advantages in practice compared to the previously used C + +. The use of FUSE remains unchanged, so GocryptFS (Figure 5) can be used on any Linux system with an activated FUSE environment.

Because FUSE is also available for Windows, GocryptFS also

works on Windows systems. With its own wrapper, however, the solution goes by the name of CppcryptFS, and, as the name suggests, the Windows version of GocryptFS is written in C + + instead of Go. MacOS users are less fortunate: macFUSE does exist, but it is not available under a free license. The FOSS community therefore tends to avoid the tool. In the meantime, however, an alternative has been devised that takes GocryptFS to macOS, too, although some special functions have been dropped.

If you don't want to use GocryptFS on the command line, you can turn to SiriKali, which offers a graphical user interface for GocryptFS and its predecessor EncFS.

What GocryptFS Offers

As described, GocryptFS always creates complete, encrypted containers in a file or on a disk. Basically, GocryptFS is designed as an overlay file system. In



Figure 3: By hiding volumes, Veracrypt makes it possible to conceal the very existence of sensitive data, helping to achieve plausible deniability.



Figure 4: If you install EncFS on Ubuntu, you will see a clear warning. It is better to use GocryptFS, which has fixed EncFS's security problems.


```

dd@ubuntu:~$ mkdir DATA
dd@ubuntu:~$ mkdir ENCRYPTED
dd@ubuntu:~$ mkdir $HOME/.config/gocryptfs
dd@ubuntu:~$ gocryptfs -init -config $HOME/.config/gocryptfs/LUTESTFORWARD.conf $HOME/ENCRYPTED
Using config file at custom location /home/dd/.config/gocryptfs/LUTESTFORWARD.conf
Choose a password for protecting your files.
Password:
Repeat:

Your master key is:

d26bb2c9-5bc933bc-c4f1908f-c4e03eef-
57ec8a84-099d5b1a-8046ca8-58414275

If the gocryptfs.conf file becomes corrupted or you ever forget your password,
there is only one hope for recovery: The master key. Print it to a piece of
paper and store it in a drawer. This message is only printed once.
The gocryptfs filesystem has been created successfully.
You can now mount it using: gocryptfs ENCRYPTED MOUNTPOINT
dd@ubuntu:~$

```

Figure 5: GocryptFS can create encrypted containers with a virtual mount point in a very short time; you can mount or unmount the containers as needed.

normal operating mode, you enable a GocryptFS container, write something to it, disable the container again, and then the encrypted content appears as such on the underlying file system. Creating a GocryptFS overlay is quick and easy with the help of the documentation.

GocryptFS offers a second option, known as reverse mode, for storage targets that are implicitly insecure. If you want to

store your backups in the cloud, for example, you can activate reverse mode in GocryptFS. Reverse mode exposes the encrypted data directly so that you can copy it to another location.

GocryptFS is about as simple as Veracrypt, however, GocryptFS offers a clear speed advantage. At least subjectively, there are considerable differences when encrypting, and these differences are despite the fact that Veracrypt and GocryptFS support acceleration for the various crypto algorithms in the same way. If you want to manage encrypted data, you will definitely want to take a look at GocryptFS.

Universal GUI: Zulucrypt

The final candidate in this encryptors showcase is a program that does not actually do the encryption work itself. Zulucrypt [4], together with Zulu-mount, which is included with it, is more concerned with organizing the proliferation of encryption methods available under Linux.

Zulucrypt (Figure 6) promises to unify classic crypt setups with LUKS, Veracrypt, and GocryptFS in a single interface. First

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

and foremost, it is a GUI that integrates and calls the other tools in the background.

Zulucrypt is extremely user-friendly. The tool is included with most of the popular Linux distributions and can otherwise easily be installed via the provider's website. Immediately after that, the main window of Zulucrypt reveals how powerful the tool actually is.

In Zulucrypt, you can either create or open LUKS devices with Cryptsetup, create Veracrypt volumes or use existing ones, and mount file systems via FUSE with GocryptFS. Unmounting works in the same way. Zulucrypt relieves the user of the need to deal with the details of the CLI or GUI applications of the individual solutions. Nevertheless, it does not hurt to at least be familiar with the basic features of the individual programs, in case something goes wrong in Zulucrypt that the program itself cannot fix. In everyday operations, however, this kind of incident is likely to be an exception.

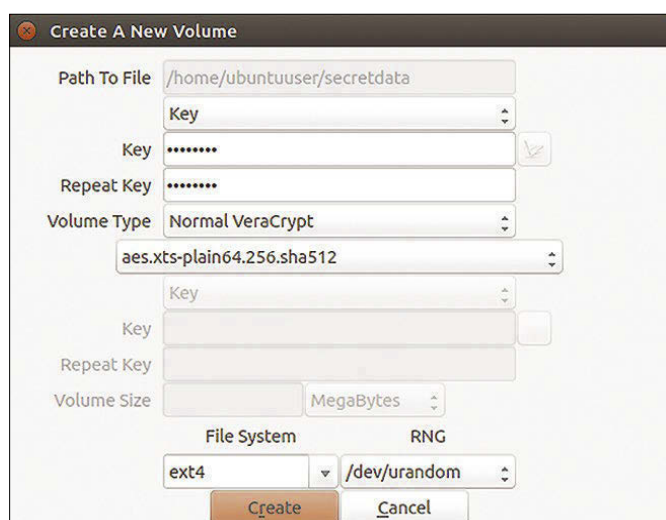


Figure 6: Zulucrypt is versatile when it comes to handling the various encryption solutions that the admin has to deal with under Linux.

All in all, Zulucrypt is a powerful little helper that saves admins a part of the learning curve and efficiently reconciles different encryptors under Linux. Zulucrypt is especially beneficial for people who use, or have to use, more than one encryption solution.

Conclusions

Encryption on Linux, whether on servers or desktops, is no longer a problem. Linux comes with the required built-in tools in the form of Cryptsetup and LUKS. Distributions integrate these built-in tools so well that you can set up a system with an encrypted root partition and encrypted home directories without any worries.

For mobile devices like laptops, these encryption tools put Linux on a par with other products like Windows (Bitlocker) or macOS (File Vault). The same applies to the encryption of portable media like USB sticks or external USB hard drives. Here, solutions are available in the form of Veracrypt, GocryptFS, and again Cryptsetup/LUKS, which can be set up quickly and work in an absolutely reliable way. If you travel between OS worlds from time to time, you will even find a solution in the form of Veracrypt that you can use on all common operating systems.

One consideration is that an existing system without encryption cannot easily be switched to encryption. Against the background of fast external drives with USB-C and Thunderbolt interfaces, it seems to make more sense to migrate all data from the device, re-install with encryption, and copy the data back. ■■■

Info

- [1] LUKS: <https://gitlab.com/cryptsetup/cryptsetup>
- [2] Veracrypt: <https://www.veracrypt.fr/en/Home.html>
- [3] GocryptFS: <https://github.com/rfjakob/gocryptfs>
- [4] Zulucrypt: <http://mhogomchungu.github.io/zuluCrypt/>

Author

Freelance journalist **Martin Gerhard Loschwitz** focuses primarily on topics such as OpenStack, Kubernetes, and Ceph.



FOSSLIFE

Open for All

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

FOSSlife.org

Secret Letters

Email encryption is not that difficult – and it is more important now than ever before. We take a look at some important tools and trends in email encryption.

By Peer Heinlein

From the point of view of a mail server, and therefore also its administrator, emails are simply ASCII text files that are transferred from server to server. They end up on the file-system both in transit and at the destination as a text file that is freely readable for the administrator. Reading mail is no problem for administrators, even if the individual message is quickly lost due to the huge numbers. But if you know what you are looking for, you can read whatever you want. Cyber criminals often hack email inboxes and systematically analyze the messages they find.

Email has existed for a generation, and the problem of email security is nearly as ancient. This article offers some thoughts on the state of email encryption today, including what works and some complications you might want to know about.

SSL/TLS and Beyond

The first step in securing email is securing the communication channel. Most well-known and responsible providers now provide secure connections to and from their mail servers by means of SSL/TLS encryption, so that the data crosses the Internet in an encrypted format.

It would be great to say *all* email providers offer secure communication with SSL/TLS, but that is not the case. Gaps remain, often with very old, unmaintained mail servers in settings where a commercial firewall or spam filter does not easily support SSL/TLS. Old Exchange servers that are directly accessible on the Internet for mail reception often stand out due to their lack of encryption capabilities. The good news is that whenever Linux-based mail servers such as Postfix are used, SSL/TLS is usually offered as a matter of course.

The sender neither can see whether, or assume that, a message is encrypted using SSL/TLS. Mailbox.org was the first provider to introduce a procedure that shows the user whether and how the target system can be reached via SSL/TLS as soon



as the email is written. Other providers have now copied and implemented this procedure.

Simply enabling SSL/TLS is not enough today. Because encryption is optional with the Simple Mail Transport Protocol (SMTP), a man in the middle could prevent the SSL connection and force an unencrypted plaintext connection. Alternatively, an attacker could swap the SSL certificates of the mail servers for their own certificates. But there is a remedy for this, too. The DNS-based Authentication of Named Entities (DANE) [1] standard, which has been available for several years, defines how mail servers can publish the fingerprints of the SSL certificates they use by means of matching TLSA type DNS records. This step effectively prevents manipulation or suppression of the certificates by third parties.

But DANE has only been used to a limited extent. Data-protection-savvy mail providers such as Mailbox.org, Mail.de, and Posteo were early DANE adopters, and several other providers have followed suit, but many email providers still find DANE very difficult to use today. The reason is that publishing the certificate fingerprints via DANE only works if they themselves secure the DNS zone against manipulation by means of DNSSEC. However, DNSSEC presents some infrastructures with greater challenges, so many IT managers are wary of taking the step.

Postmasters who have a DNSSEC-secured domain can publish the checksums of their certificates in just a few simple steps. Special websites [2] help to generate the required

TLSA-type DNS record. But be careful: Every time you change the SSL certificate in the future, you also have to remember to update your own DANE record. Mail servers with a state-of-art configuration like Postfix will stop outgoing mail if the certificate and fingerprint published via DANE do not match or if a target does not offer SSL/TLS even though a DANE/TLSA record has been published. The `main.cf` file of the sending mail server needs a `smtp_tls_security_level = dane` line, even if the server does not offer DANE for incoming messages.

Securing the Contents

SSL is in place. Is everything fine? Not at all, because SSL only secures the TCP/IP connection between two specific mail servers. No one knows whether the message might be transmitted over unencrypted connections further down the line. In addition, the email itself ends up in plaintext on the servers, both in the recipient's inbox and in the sender's Sent folder.

In an electronic mailbox that has operated for years, perhaps even decades, many highly personal things could exist that you would probably prefer to keep out of the public eye. Every user needs to ask what impact it would have if their entire email inbox became public, because that can happen all too quickly. A poorly chosen IMAP password can be easily hacked; it may have even been revealed in an inattentive moment to a phishing site or intercepted by a local virus infection.

The only way to ensure that your email will remain private is to encrypt the messages using PGP or S/MIME. Both methods are based on the same principle: Each user has an asymmetric key pair. It consists of a public key, which is used for encryption and a digital signature, and a private key, which is used for decryption and verification of a digital signature. A password chosen by the user protects the private key against unauthorized access. This password must not be forgotten under any circumstances. If it is lost, the key file is unusable, and all data encrypted with it can no longer be accessed.

Pretty Good Privacy

Many Linux users today are accustomed to working with PGP. You store your own public key, together with those of other recipients in a local key database, also known as a *keyring*. Since several keys can be generated for a mail address over time, each key has its own ID. In addition, the authenticity of a key can be verified using a mathematically unique fingerprint. This fingerprint can be published on business cards or in your web imprint or verified by phone, which allows communication partners to ensure that they really have the right key.

Generating a public/private key pair is quite simple nowadays and usually occurs automatically the first time you use the PGP software tool. The public key can then be transmitted individually or automatically to communication partners. Mail programs such as Thunderbird offer to attach the public key to an

outgoing email at the push of a button. You can also upload the public key to public PGP key servers, where interested parties will find it automatically.

Once a key has been put into circulation, it can no longer be deleted, but it can be declared invalid in the event of key theft or password loss. To declare a key invalid, you should generate a revoke certificate ahead of time (i.e., immediately after generating the key) and store it in a secure place in your own filesystem. If your own password is lost, you no longer have the option of creating a revoke certificate. In addition, keys should not be issued for an indefinite period of time but assigned an expiration date a few years in the future.

All of these options are usually found in the mail program's key manager. Press the right mouse button to call up the properties and potential actions for your own key (Figure 1).

One problem remains: It is initially impossible to verify whether a key actually originates from the owner of the mail address. Attackers can easily generate their own PGP keys for any mail address and put them into circulation. The public PGP keys of other users are therefore initially considered untrustworthy in your own keyring. However, the sender and recipient can have the specific PGP fingerprint displayed there and verify it via another communication medium – by making a short phone call if necessary.

So much for the theory. In practice, many users today use the keys they receive without verification. In the settings of most popular mail programs, you can define the trust level from which a key can be used for sending. This is certainly not ideal and could convey a deceptive sense of security. On the other hand, using any kind of encryption is better than having none at all, so in the interest of the matter at hand, just take a pragmatic approach if in doubt. A received key is insecure if it has been manipulated by a third party at the time of receipt and the compromised key is trusted permanently. But if you assume that the key exchange (which may have occurred long ago) was secure at the time, then the key can successfully provide protection against future unwanted readers and the dreaded man in the middle.

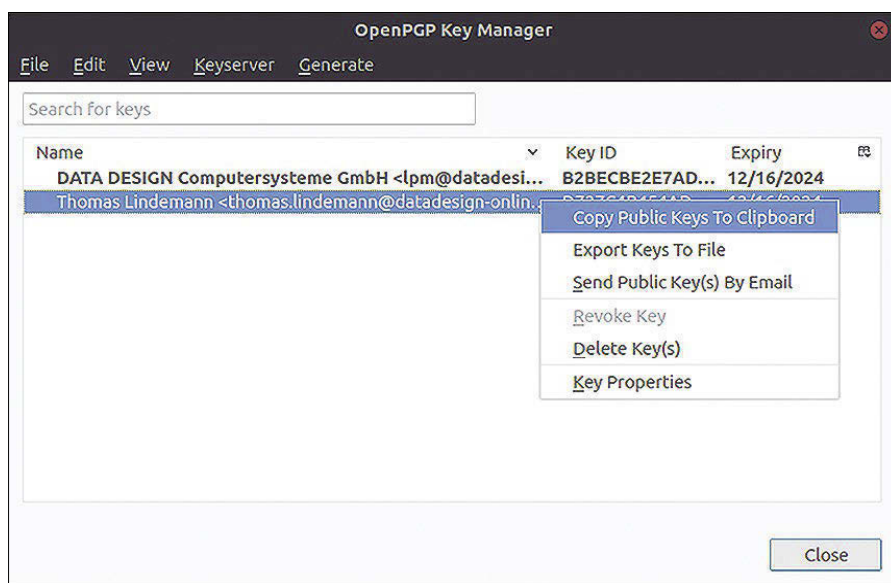


Figure 1: Your mail program's key manager should offer several options for managing keys.

Since manual checks are not a good way to handle large numbers of communication partners, PGP implements the concept of a web of trust. After careful personal verification of their identity, users sign the keys of other users (Figure 2), thereby increasing their trustworthiness. The signatures then become part of the user's public key (Figure 3). If, at some point, keys have a sufficiently good reputation due to the many signatures of known trustworthy communication partners, you can do away with your own manual verification. Even today, PGP keysigning parties are held at conferences. You have to bring your own PGP keys and, by default, your ID card or passport as proof of identity, because careful verification is mandatory.

But the concept of mutual PGP signing is complex, requires initiative, and requires a great deal of interaction – one of the main reasons why PGP has never gained widespread acceptance. Modern approaches therefore envisage publishing your own PGP key via special key servers run by your provider, from which communication partners can obtain it automatically. Although this approach does not rule out manipulation by the provider's administrators, it does ensure that keys can no longer be generated and circulated arbitrarily by third parties. Using methods such as WKS/WKD [3], domain owners can therefore specify whose PGP key servers need to be queried for the keys for their own domain. However, only a few providers systematically offer PGP key servers and have set up corre-

sponding WKD records for their mail domains. If they do, nothing stands in the way of an automatic key exchange. PGP implementations such as GnuPG can then automatically detect the correct key servers and download key material.

S/MIME: A Central Approach

S/MIME [4], the second known mail encryption method, basically encrypts and signs in a way comparable to PGP but takes a more centralized approach to key generation and management. S/MIME certificates are issued, monitored, and declared invalid if necessary by a central certification authority. S/MIME is therefore suitable for structured use in a very controlled environment, for example, in public authorities or large companies. It is not surprising that classic business mail clients such as Outlook have always supported S/MIME natively.

On the other hand, S/MIME is less practical for private use, because there is initially no sensible centrally organized certification authority. Many years ago, there was an attempt by savings banks to issue S/MIME certificates to the masses via the S-Trust project, but this initiative did not catch on. The most practicable approach to date appears to be the Volksverschlüsselung (People Encryption) project [5], which is run by the Fraunhofer Institute [6], which is worthy of recommendation but has not achieved widespread acceptance.

Unencrypted Metadata

If an email is encrypted using PGP or S/MIME, the encryption affects the entire body of the email, including the message text and any file attachments. The metadata of the mail header (i.e., information such as sender, recipient,

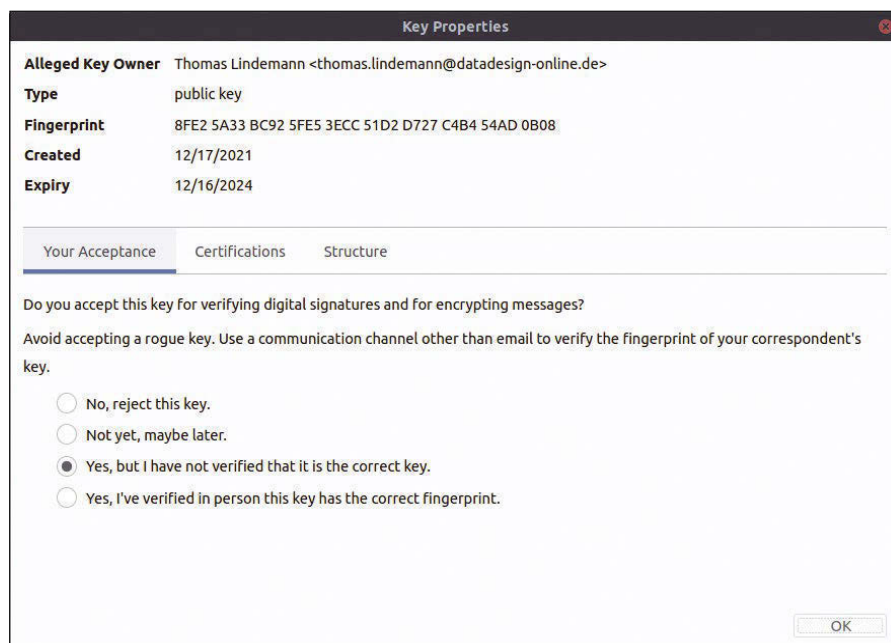


Figure 2: Signing a third-party key increases its trustworthiness and requires appropriate care.

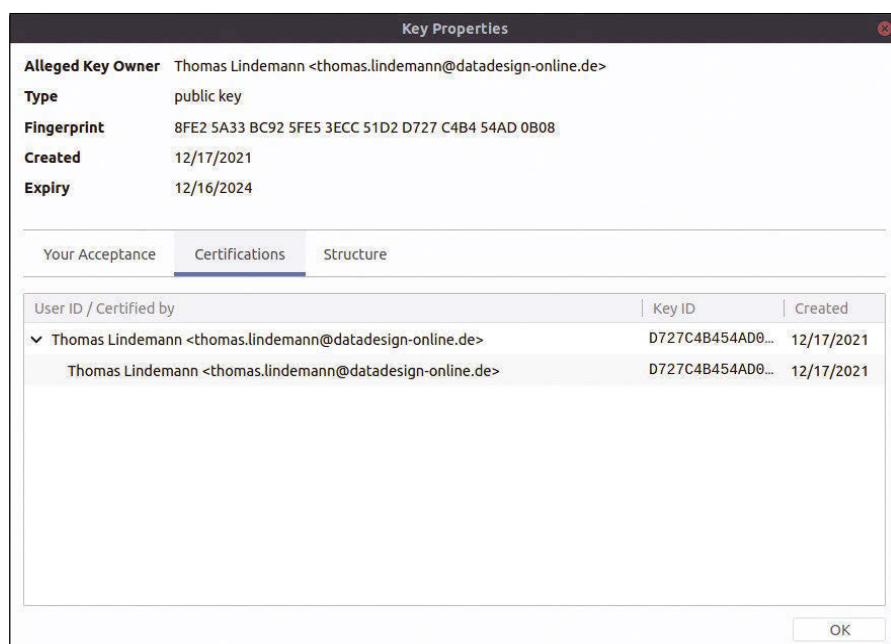


Figure 3: The properties of a key show its fingerprint and the signatures of other users.

message ID, and date) remains readable. The subject of an email is also classically part of the non-encrypted header data of an email. If desired, modern PGP implementations can also hide the subject in the encrypted body of the message so that it only becomes readable after decryption.

Encrypting in the Mail Client

If a mail program like Thunderbird encounters an encrypted message, it first identifies the key IDs used to encrypt the message and then checks whether there is a secret key for decrypting it in the internal keyring. If so, the program prompts the user to unlock their own key by entering a password. The mail can then be read normally and without any problems and, if required, displayed in the inbox together with the subject.

It makes sense for the mail program to keep the decrypted key in memory for a longer period of time, so that it can be used for hours without entering a password. The user can thus conveniently and transparently read and edit countless encrypted messages without even noticing that the messages are currently being decrypted in the background. It is only after a longer period of inactivity or after restarting the mail program that the user has to enter the key password again. This approach ensures maximum usability but theoretically also harbors the risk that an attacker could read the unsecured key from RAM.

To send an email, on the other hand, the generally known and password-free public key of the recipient is used if it exists. If a message is sent to several recipients, the emails can be encrypted simultaneously with the keys of all recipients and thus opened by all of them without any problems. The only minor disadvantage is that experts could identify hidden BCC recipients on the basis of the key they now use in the email. By the way, if the sender's own address is included in the distribution list, the copy sent back to the sender is also encrypted by default.

Encrypting with Webmail

Some webmail systems are behind the times in providing comprehensive encryption. The Mailvelope browser plugin [7], which is available for Chrome, Edge, and Firefox, adds secure OpenPGP communication to webmail. It runs locally in the user's web browser and detects when the provider's webmailer contains a PGP-encrypted email. It then decodes the contained email, exchanges the contents of the web page for the unencrypted message, and displays the message.

Mailvelope can also send encrypted email. Before a message written in plaintext is sent on its way, Mailvelope encrypts it locally and only then transmits it to the provider's webmail system. The process seems good at first glance, because

decoding occurs locally on the user's computer. However, security experts have complained about the implementation of Mailvelope as a browser plugin: it leads to the sensitive PGP key material being stored in the browser's plugin area, which cannot be 100 percent protected. In addition, JavaScript is not considered suitable for implementing secure cryptography.

Implementations such as the Guard system of the Open-Xchange groupware solution [8] take a somewhat different approach. These solutions store the key securely on the provider's server, and a password entered by the user protects it against unauthorized access. The server takes care of encryption and decryption, removing the need for a browser plugin. This means that users can access their own mailboxes from other computers at any time, even when if they are on the road.

Conclusion

Cyber snoopers are more sophisticated than ever, which means there has never been a better time to get familiar with email encryption. However, as this article has shown, you can't just install SSL/TLS or PGP and expect a safety guarantee. It pays to consider the details and look closely at what you need to ensure your messages remain private.

Whether trusting your email provider offers you more security, or whether you are better off keeping your own key on your private PC, is a matter for every user to determine. But either way, in view of the recent gamut of virus and ransomware attacks, it pays to be cautious. ■■■

Info

- [1] DANE: <https://datatracker.ietf.org/doc/html/rfc6698>
- [2] TLSA generator: <https://ssl-tools.net/tlsa-generator>
- [3] WKS/WKD: <https://wiki.gnupg.org/WKD>
- [4] S/MIME: <https://en.wikipedia.org/wiki/S/MIME>
- [5] Volksverschlüsselung: <https://volksverschlueselung.de> [In German]
- [6] Fraunhofer Institute: <https://www.fraunhofer.de/en.html>
- [7] Mailvelope: <https://mailvelope.com/>
- [8] Open-Xchange: <https://www.open-xchange.com/>

Author

Peer Heinlein is responsible for running business-critical Linux infrastructures with his company Heinlein Support. Specializing in mail services since 1992, he wrote the Postfix book and is responsible for the mail servers of many ISPs, data centers, and enterprises. His own mail provider, Mailbox.org, specializes in data security and privacy.



Hone your skills with special editions!

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

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

Available in print or digital format

Check out the full library!

shop.linuxnewmedia.com

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

GETTING STARTED WITH

LINUX

• MORE POWERFUL • MORE SECURE • MORE

LEARN HOW TO SET UP A LINUX SYSTEM

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



LINUX NEW MEDIA

WWW.LINUX-M

LINUX **301 BEST BASH COMMANDS**

LINUX SHELL

HANDBOOK 2022 Edition

SUPERCHARGE

YOUR LINUX SKILLS

Power at Your Fingertips

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



TUNE YOUR LINUX SYSTEM

2021 EDITION

101 COOL LINUX HACKS

Tricks and shortcuts for Linux geeks

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



STREAMLINE: Clean up hidden files

EXTREME CHROOT: Change to a second distro

Discover the secrets of the experts

WWW.LINUX-MAGAZINE.COM

FREE DVD! LibreOffice Full Version

Dive deep into the world's greatest free office suite

2022 Edition

LibreOffice Expert

Edit and Save MS Office Files

Write Your Own LO Macros

Save time and automate

ARCHIVE DVD! **RASPBERRY PI GEEK** THE COMPLETE ARCHIVE 2,000 pages of maker projects and more! FREE DVD \$39.90 VALUE!

MakerSpace

HANDS-ON PROJECTS FOR MAKERS

Dive Into

- Raspberry Pi
- Arduino
- Retro gaming

Discover FPGA

Learn to program hardware for retro computing



Airtight!

A concept called provable security brings the rigor of mathematics to the art of cryptography.

By Dominique Schröder

Cryptography was originally a mystical art form with the goal of concealing information from unauthorized persons. For example, a simple method for encrypting messages was developed by Gaius Julius Caesar around the year 100 BC. The basic idea of the method was to shift each letter of a message by a fixed amount in the alphabet. This position then served as a secret key. For example, if Caesar were to encrypt *Veni, vidi, vici* with a secret key of 3, it would result in a ciphertext of *Yhql, ylgf, ylfj*.

However, the method was soon seen through, then improved, cracked again, and so on. In this way, over time, a kind of cat-and-mouse game developed between the designers of encryption methods and the attackers who cracked the methods in order to access the secret information. This dragged on until the discovery of provable security in the 1980s.

Provable security is a field that seeks to assess the security of systems through mathematical proofs. People often think of cryptography as a sea of digits and huge numbers, which it often is; however, for a mathematician, the goal is to sweep away the details and define the system symbolically in a way that lends itself to logical proof. The development of provable security has brought rigor to the ancient practice of cryptography, which was once considered more art than science.

This article will introduce you to the concept of provable security and will put the principles to work with an example analysis of email communication. I'll also describe some typical errors in the use of cryptographic methods, as well as the strengths and limitations of provable security. And, last but not least, I'll use practical examples to illustrate



current research topics in the field of modern cryptography.

Provable Security: The 6 Steps

According to the principles of security-by-design, the provable security process has six steps, as shown in Figure 1.

Step 1: Describe the Functionality

The first step is to describe the functionality of the system – What is the intended purpose? This description often takes the form of a requirements specification.

Step 2: Define the Security Properties

The task of defining the security characteristics usually has two parts. First, you need to describe the security objectives in a way that everyone can understand. After that, you need to formalize the whole thing precisely in order to avoid ambiguous interpretations.

Note that the formal security case applies only to the formal description. That is why this formalization must break down the real protection goals as precisely as possible. As a rule, this step covers the following aspects: What security property does the system need to provide? What system(s) with what setup are we talking about? What class of attackers do we need to fend off? What are their capabilities? When is an attack deemed successful?

Identifying and formalizing the security properties is a challenging task, and a suitable security model has not yet been found for many cryptographic schemes. For example, the research community has still not agreed on a definition for authenticated encryption [1]. In addition, there is always a natural state of competition between efficiency and security.

Ideally, you would want to provide protection against all types of adversaries and rely on the best possible security guarantees. However, security is not free, and stronger security guarantees are often less efficient. Inefficient systems are not usually adopted in practice.

Step 3: Implement the Functions Using Generic Procedures

This step refers to the basic cryptographic building blocks, such as private key encryption, message authentication codes, hash functions, or signature schemas. Defining generic procedures helps you understand which security property of the underlying building block is used to achieve a certain security property of the new system.

The design has to work independently of specific procedures, since it is always possible for individual procedures to be broken. For example, researchers recently found the first attacks on the SHA-1 hash function. Developing a completely new system because individual components have been broken is simply too expensive. Moreover, the insecurity of individual instances does not alter the fact that the concept as such is secure. To illustrate this point, just briefly think about the example of fire protection. Just because the material of an individual door proved not to be fireproof does not mean that the entire strategy, which envisaged a fireproof door at a particular location, is wrong.

Step 4: Formal Mathematical Proof of Security

After the formal specification of the security properties and the design, formal mathematical proof of security follows in the fourth step. This step confirms that the design satisfies the desired safety properties. Formal proof provides a one-to-one mapping between the security properties of the underlying cryptographic building blocks and the security properties that the system is intended to achieve. Formally verifying the security uncovers design flaws. If the security properties of the underlying cryptographic building are not applied, there is a flaw in the design, and there is most likely a more efficient solution.

Step 5: Instantiate the Implementation

Once the formal security of the system has been established, it is necessary to instantiate the generic cryptographic building blocks with specific cryptographic procedures. For example, a generic encryption schema with a private key is introduced, and this building block is implemented in practice using AES. Since generic building blocks work with abstract objects (such as “a private key” or “a ciphertext”), you need to translate these objects into concrete instances. For example, the object “public key encryption scheme” is instantiated with an ElGamal encryption scheme [2].

Step 6: Security Analysis of the Implementation

The sixth and last step calls for security evaluation of the concrete instantiation. This is required because transitioning from an abstract description to a concrete implementation has many pitfalls. For example, the basic cryptographic element used with Internet Protocol Security (IPsec), which is known as “encrypt-then-MAC” is an authenticated encryption scheme with weak security [3]. It is considered secure against a chosen-plaintext attack (CPA) [4], and it is also secure against spoofing given that the encryption scheme is CPA-secure and the message authentication code (MAC) cannot be spoofed. However, the concrete instantiation used in IPsec, for example, was found to be insecure. The security of the implementation can also be proven by penetration tests, which reveal weaknesses in the implementation of the security concept that could allow an attacker to penetrate the system.

Example: Secure Email

To illustrate the principle of provably secure software development, consider the example of an encrypted email message (Figure 2). In this case, Alice wants to send an encrypted message to Bob. Bob needs to be able to decrypt the email and verify the signature.

Alice has Bob’s public key pk_B . For simplicity’s sake, I will assume that Alice has verified the key and ensured that it is Bob’s. Furthermore, Alice has her own private key sk_A .

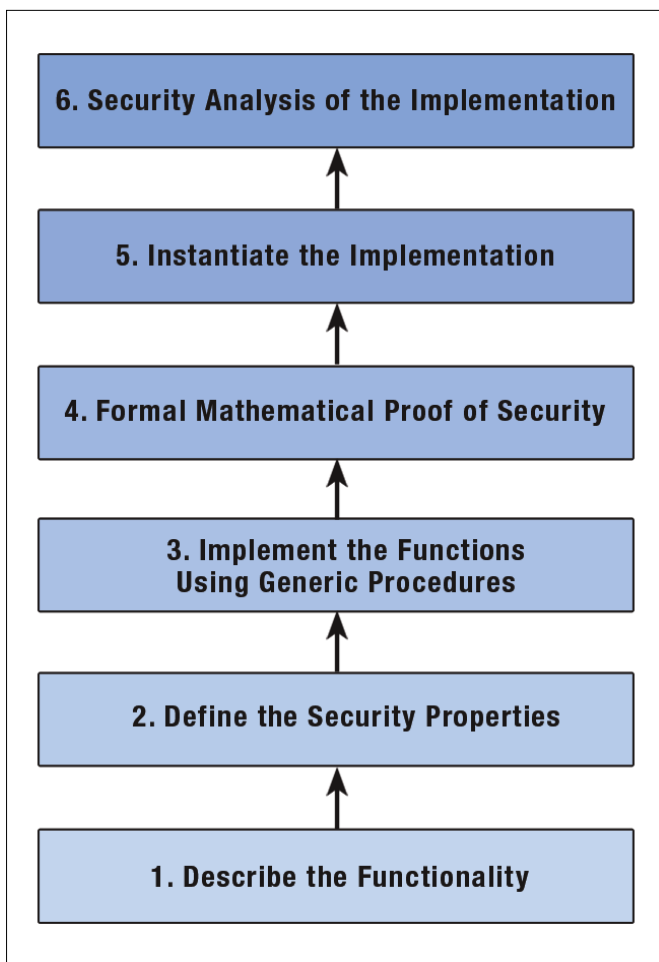


Figure 1: The six steps leading to provable security.

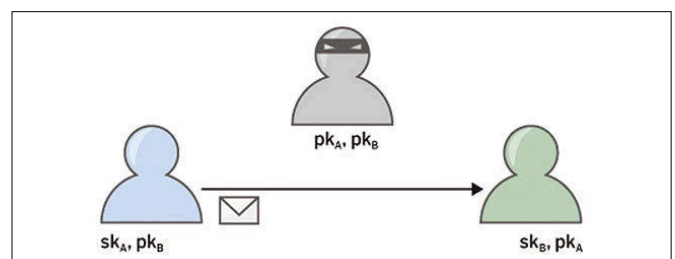


Figure 2: Secure email communication from Alice to Bob.

Similarly, Bob knows Alice's public key pk_A and has his own private key sk_A . In this case, I will also assume that Bob has verified the authenticity of the key pk_A .

The first step is to specify the functionality. In this case we are dealing with authenticated email; consequently, I have two tasks: First, I need to compute an encrypted and authenticated email. This interface is designated $Enc + Auth$. Second – assuming that the authentication is valid – the encrypted email has to be decrypted. Keeping to the same pattern, I will refer to this interface as $Vrfy + Dec$.

The second step is to determine the (fairly simple) security properties. The attacker knows Alice and Bob's public keys and all the methods used. Nevertheless, the attacker must not be able to obtain information from the email. Furthermore, the attacker must not be able to send emails on behalf of Alice (i.e., messages verified with the pk_A public key) to Bob.

Although these features seem to make sense at an intuitive level, they still leave too much scope for interpretation. Can the attacker interact with Bob? Is there a learning phase where the attacker is allowed to use Alice as an interface to forward the encrypted and authenticated emails from Alice to Bob? Is the attacker allowed to impersonate Bob and read encrypted emails for a period of time? How are the messages to Bob generated? In a formal description of the security properties, it is important to eliminate these ambiguities so that a precise and unambiguous description results.

After formalizing the interfaces and the security properties, the third step is to instantiate the individual components using cryptographic methods. The two interfaces $Enc + Auth$ and $Vrfy + Dec$ need to be constructed with cryptographic procedures such that they provide precisely the desired functions. In the case of secure email, this step is relatively simple (Figure 3).

To implement the $Enc + Auth$ interface, you first need to encrypt the message with a public key method and then sign the ciphertext with a digital signature method. I'll use c to denote the ciphertext and the lowercase Greek letter sigma (σ) to denote the signature. In line with this, you would implement the $Vrfy + Dec$ interface with the following operations. First, the signature procedure's verification algorithm is used to check the validity of the signature on the ciphertext c . If this signature is valid, the public key procedure's decryption algorithm decrypts the ciphertext c and outputs the resulting message m .

In the security validation, I can now assume in the fourth step that both the encryption method and the signature method are secure according to formal definitions. You then formally verify that an attack that breaks the security of $Enc + Auth$ as well as $Vrfy + Dec$ (according to the formal model) can be used to break the security of both the public key and the signature

procedure. However, since both procedures were assumed to be secure in the first step, a contradiction arises, from which it follows that such an attack cannot exist.

In the last two steps, you now use concrete procedures, for example, RSA-based encryption, implement them, and prove the security of the implementation through penetration tests.

Strengths and Limitations

Since the discovery of this method, provably secure cryptographic methods such as El Gamal encryption [4] have been virtually unbreakable. Provable security reduces the proof for an encryption method to a simple computational problem, such as the very well-known factorization problem.

In this problem, the attacker is given a number $N = p \cdot q$ calculated as the product of two primes of equal length p and q . The attacker's task is now to compute the two prime factors p and q . To date, no efficient solution algorithm has been discovered for this well-known computational problem. If the security of an encryption scheme is now based on the factorization problem, then the formal proof shows the following. If there is an efficient attacker who breaks the encryption scheme, then there is also an efficient attacker who solves the factorization problem. In other words, the problem of breaking the encryption is as difficult to solve as the factorization problem.

At first glance, the term "provably secure" suggests that there are provably no longer any attacks. However, this is not true. Attacks on systems that are *supposedly* probably secure repeatedly occur. Where does this contradiction come from, and how can it be resolved? First of all, it is important to note that the security of the proof always refers to a formal model. However, if the model does not truly reflect practice, then it cannot rule out potential attacks because these attacks simply do not occur in the model. Consequently, accurately modeling reality is one of the biggest challenges in this area, and even the smallest errors in modeling can lead to an insecure solution.

The security of the Secure Sockets Layer (SSL) protocol is a great example of incomplete modeling. In practice it was possible for a message not to be decrypted; instead, a *Padding Error* message appeared. At first glance, this appears harmless, as it does not seem to reveal any direct information about the plaintext. This is why it was not included in the formal model. At second glance, however, researchers were able to show that it was possible to exploit this error message to decrypt the full message – and to do so without breaking the actual security of the encryption scheme.

Typical Errors

A widespread mantra from the IT security field is that you should never develop your own cryptographic method. But at what point

do you construct your own procedure? In fact, this mantra falls short in many places and should instead read: "Never use cryptographic procedures unless you can prove what you are doing." The following example demonstrates that this statement is by no means exaggerated and that even the simple use of two cryptographic procedures can lead to a completely insecure construction.

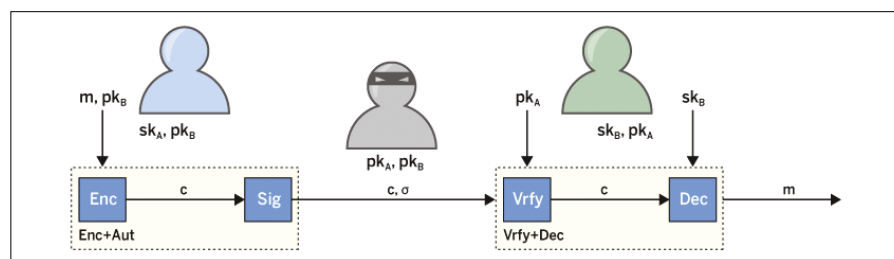


Figure 3: Schema for secure email communication.

In this example, assume that Alice wants to store a file m with encryption in the cloud. To do this, she uses the message and a public key. The cloud operator does not want to store duplicates of the same message for efficiency reasons. However, since the file has been encrypted, the operator cannot readily determine whether the message already exists in an encrypted form in the cloud.

For this reason, the ciphertext is extended to include the hash value of the message: $c = (Enc(pk, m), H(m))$. A hash function is a deterministic function that usually compresses the input. The security of this function is based on collision resistance. This property means that it is (efficiently) impossible to find two different messages m_1 and m_2 that produce the same output ($H(m_1) = H(m_2)$).

Intuitively, you might expect that the hash value does not reveal any information about the message m thanks to the compression properties. However, there is no security property that covers this assumption. In fact, analysis of the construction shows that it is completely insecure. In particular, the output of the hash function can be used to completely override the security properties of the encryption.

This simple example illustrates that a combination of two secure cryptographic methods can lead to an insecure construction. Cryptographic methods should only be used together if you actually need the security properties of each method. It is precisely this relationship between the underlying properties and the properties you want to achieve that is demonstrated by formal proof.

Current Research Topics

There are still countless unsolved problems in the cryptography research field. Thematically, it moves between basic theoretical research and applied practical research. This section presents two areas of current research.

One is password-based cryptography. Passwords are the Achilles' heel of many modern systems because they combine everything a cryptographer usually doesn't want to have: They are short and have little entropy, which means that an attacker can break most passwords by simply trying them out. This type of attack is known as an offline brute force attack. After the invention of public-key cryptography, many speculated that passwords would die out and everyone would exclusively use a private key for authentication. That has not happened to date,

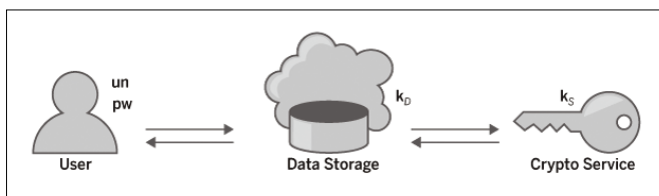


Figure 4: Flow for password-hardened encryption.

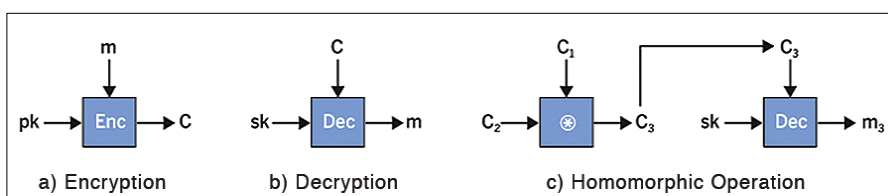


Figure 5: Computations on encrypted data.

and it appears that passwords will remain the most common technique for authentication on the Internet in the long term. In research, this leads to the following question: How can the security of a system be guaranteed even if it uses weak secrets like passwords?

Password hardening and password-hardened encryption (PHE) were developed in this context (Figure 4). The idea is mainly based on the following considerations. First, the user's interface must not change, so they continue to use their username and password for authentication. Second, the user's data must be resistant to offline brute force attacks – even if the database has been stolen. And third, to achieve that, you need an external crypto service that does not have direct access to the data, but only provides operations to encrypt and decrypt the data.

Password-hardened encryption achieves a level of security that was previously unthinkable. For the first time, a user's data can be stored secured against offline brute force attacks, and without the user having to use complicated public key techniques.

A second research focus is on computations on encrypted data. To better understand how computations can be performed on encrypted data, consider how conventional encryption works.

Regardless of the type of encryption method, each method essentially provides key generation, encryption, and decryption. Since key generation does not play a role in the following considerations, I will not go into the details of this operation.

In cryptography, an encryption operation is a procedure that converts a readable text – the plaintext – into a form that no longer reveals any information about the plaintext – ciphertext. Figure 5 visualizes the encryption and decryption operation. The encryption algorithm Enc receives a message m and a public key pk as input. The result of the computation is a ciphertext c . The decryption algorithm Dec is used to decrypt this again. As input, it needs the secret key sk and the ciphertext c .

In order to describe a homomorphic encryption method, you first need to understand the concept of homomorphism. It comes from mathematics and means structure-preserving mapping. Put simply, this means that you can map the execution of a computation in a structure to a computation on another structure. The structure itself is preserved.

The concept of homomorphism is now applied to encryption methods. This means performing a computation on the ciphertext without actually decrypting the ciphertext. The special thing about this operation is that the computation on the plaintext has an effect on the ciphertext, although the plaintext is not available during this computation. Depending on the type of encryption, the combination of the ciphertexts is applied to the plaintext as an addition or multiplication. The addition operation is known as additive homomorphic encryption, and the multiplication operation is called multiplicative homomorphic encryption.

The right half of Figure 5 shows this process. You can see two ciphertexts C_1 and C_2 on the left. They were computed in the conventional way: that is, C_1 is the result of encrypting a message m_1 with the public key pk , while C_2 is the result of encrypting a message m_2 with the public key pk . These two

ciphertexts are now combined by a computation x .

By way of an example. I'll first encrypt the number 2 with the public key pk :

$C_1 := \text{Enc}(pk, 2)$. The resulting ciphertext is referred to as C_1 . In the second step, I'll follow the same steps with the number 3:

$C_2 := \text{Enc}(pk, 3)$ to generate the ciphertext C_2 . The third step now concatenates the

two ciphertexts by applying the operation x : $C_3 := C_1 \times C_2$. If the encryption method is an additive homomorphic method, then the ciphertext $C_3 := C_1 \times C_2 := \text{Enc}(pk, 2) \times \text{Enc}(pk, 3) = \text{Enc}(pk, 2 + 3) = \text{Enc}(pk, 5)$ contains a value of 5. If the process is multiplicatively homomorphic, then $C_3 := C_1 \times C_2 := \text{Enc}(pk, 2) \times \text{Enc}(pk, 3) = \text{Enc}(pk, 2 \times 3) = \text{Enc}(pk, 6)$, contains a value of 6.

In both cases, only the owner of the secret key sk has access to the result of the computation, since only they can decrypt the ciphertext C_3 . An example of additive homomorphic encryption is provided by the ElGamal method [2], whereas the Paillier method [5] is multiplicatively homomorphic.

In contrast to the previous methods, fully homomorphic encryption allows arbitrary calculations to be performed on the ciphertext. For a better understanding of how it works, Figure 6 shows the individual steps. First, a plaintext m is encrypted with the public key pk (a), resulting in the ciphertext C . Now the computation (b) is performed; the ciphertext C and the public key pk and the description of the program P flow into this. The computation results in a ciphertext C_p and the result of the computation is $\text{Eval}(pk, C, P) = C_p = \text{Enc}(pk, P(m))$.

Just as in the previous examples, anyone can perform the computation on the ciphertext C , since it does not require any secret information. However, only the owner of the associated secret key sk can compute the result. Fully homomorphic encryption is an incredibly powerful tool because there are no constraints on the P program. This means that arbitrary computations can be performed on the encrypted data. Performing computations on encrypted data dramatically reduces the need to trust the party storing the data. For instance, you could outsource the data to a cloud service that holds the data and performs computations without the need to decrypt. The service performing the computations would not need to see the plaintext data or even the result of the computation. The data owner remains in control over the data.

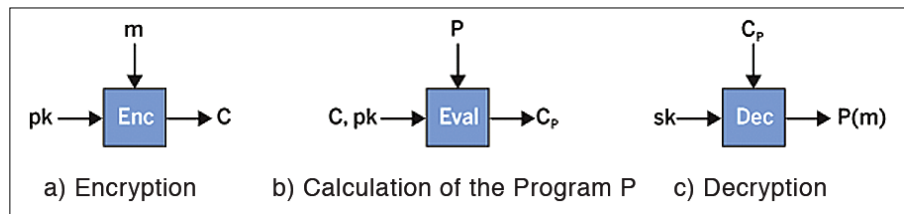


Figure 6: Fully homomorphic encryption.

Conclusions

This article has introduced you to provable security, one of the cornerstones of modern cryptography. You also learned about some recent research topics that are exciting cryptographers today.

Cryptography has been around for thousands years, but it still manages to excite, and state-of-art encryption techniques still sound like pure magic. For example, who would have thought that computations on encrypted (genetic) data would work? But cryptography is still in its infancy, and the crypto technologies of the future will no doubt continue to surprise us. ■■■

Info

- [1] Katz, Jonathan and Yehuda Lindell, *Introduction to Modern Cryptography: Principles and Protocols*, CRC Press, 2007, https://archive.org/details/Introduction_to_Modern_Cryptography
- [2] ElGamal encryption: <https://www.geeksforgeeks.org/elgamal-encryption-algorithm/>
- [3] Encrypt-then-MAC: [https://en.wikipedia.org/wiki/Authenticated_encryption#Encrypt-then-MAC_\(EtM\)](https://en.wikipedia.org/wiki/Authenticated_encryption#Encrypt-then-MAC_(EtM))
- [4] CPA: https://en.wikipedia.org/wiki/Chosen-plaintext_attack
- [5] Paillier encryption: https://en.wikipedia.org/wiki/Paillier_cryptosystem

Author

Dominique Schröder heads the Chair of Applied Cryptography at the Friedrich-Alexander University Erlangen-Nuremberg, Germany and advises companies on the analysis and development of IT security concepts. He has received numerous awards, including the Feodor-Lynen Research Fellowship from the Alexander von Humboldt Foundation and the Intel Early Career Faculty Award.



What?!

I can get my
issues
SOONER?

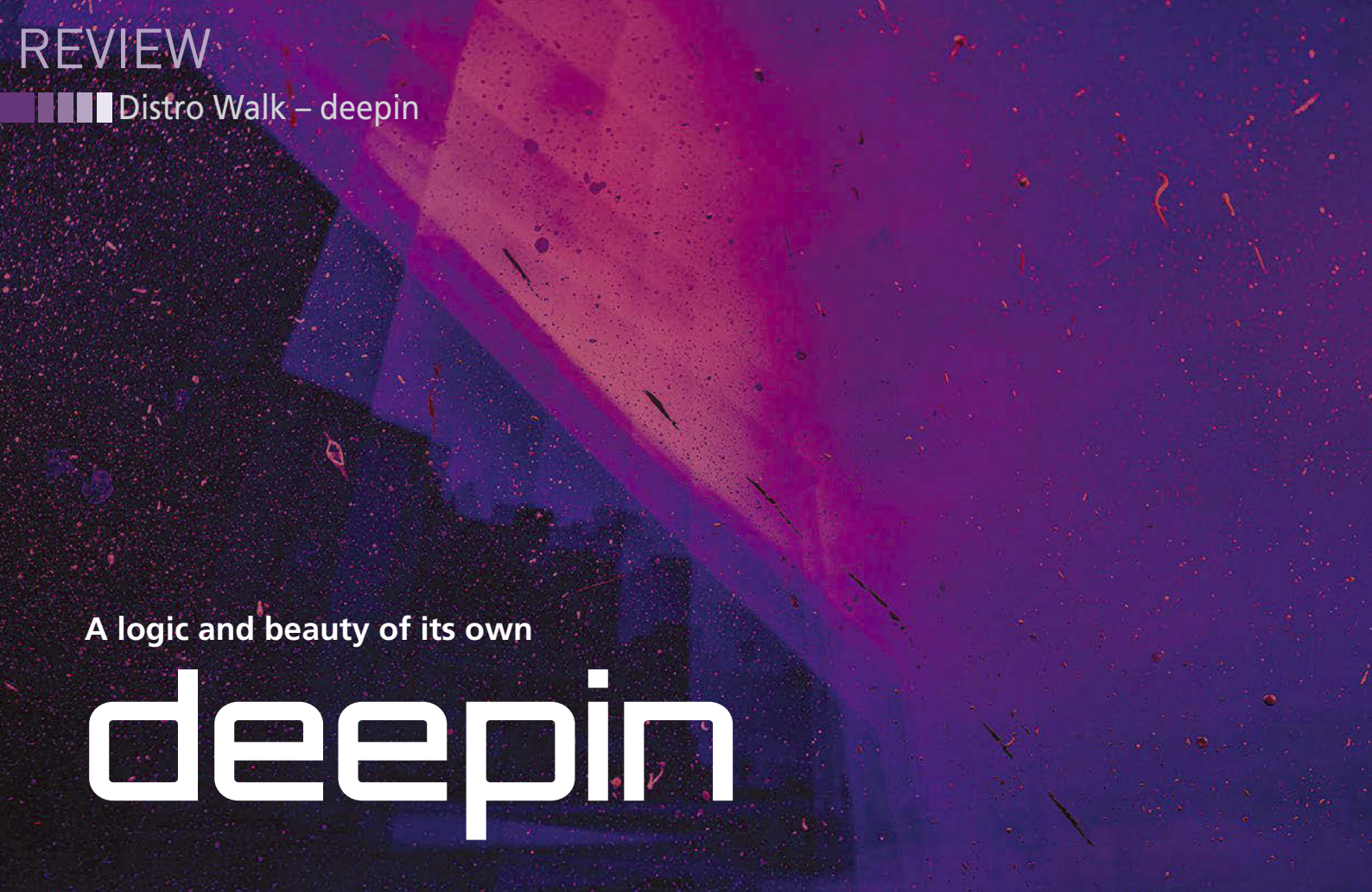


Available anywhere, anytime!

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

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

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



A logic and beauty of its own

deepin

Deepin offers a visually stunning OS with a few unique quirks. *By Bruce Byfield*

Deepin [1] is a distribution developed in Wuhan, China by Deepin Technology. Its homepage proclaims it “the top Linux distribution from China,” although similar claims are sometimes made for Ubuntu Kylin. Deepin has a longstanding reputation for attention to visual detail that is more reminiscent of macOS than Windows. Like elementary OS, deepin is visually stunning right out of the box, without any customization. However, when you start to install, you soon realize that its aesthetics come at a cost.

For one thing, the installer suggests a minimum of 64GB of disk drive and recommends 128GB to enable all of the effects, although these requirements are not mentioned until you choose a disk to partition. By contrast, Ubuntu requires a minimum of 512MB for a perfectly functional Gnome desktop and can function on older computers that deepin cannot.

For another, the organization of the Deepin Desktop Environment (DDE) has a logic all its own. This logic is not difficult to figure out but does take some adjustment when working with it, to an extent that Gnome or Plasma do not.

The Endless Installation

Ordinarily, I comment on a distribution’s installer as little as necessary. After all, most users will only deal with it a few times – often, only once. However, given the unique logic, deepin’s installer should be described in some detail.

To start with, you should read and agree to the EULA and privacy policy – even if you typically accept policy statements without first reading them. The extensive EULA is uncommon for the Linux space, and the privacy policy goes into some detail about the types of information they collect – not just browser history, but information on when you use your computer and the applications installed on your system. If you are concerned about privacy and license restrictions, you might decide to forego the installation; however, if you are easy going about such things, check the box and proceed.

If you do decide to install, do not be deceived by the installer. At first, it looks deceptively simple consisting of only three items: the choice of language and partitions, followed by installation (Figure 1).

After you install and reboot, you are faced with the same page that appeared when you booted the first time. It looks like the installation has failed, but eventually the installation continues with the second stage, which includes everything omitted from the first stage. If you are using a virtual machine, you are asked to choose the installation mode, either Effect, which is described as offering (in slightly fractured English) “a delicate mode,” or Normal. The installation continues with the selection of the time zone from a map that does not indicate time zones, along with some unusual restrictions, such as a limit of 3-32 characters on account names. Yet even then, the installation is not finished.

The third stage involves customization, with choices of a Fashion (large dock) or Efficient (small dock) desktop mode, an Effect or Normal running mode, and several icon themes. These choices are generally clear, but no details of exactly what each entails are offered during installation. Fortunately, the running mode can be changed later from the desktop’s dock and the icon mode from the Personalization app in the Control

Photo by Jr Korpa on Unsplash

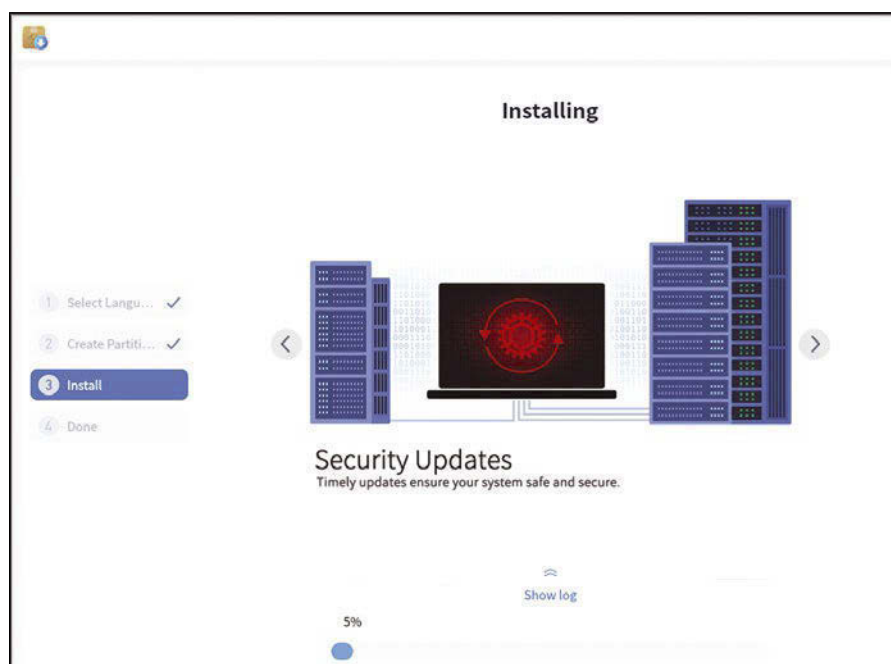


Figure 1: The first stage of deepin's four-stage installation process.

Center. Apparently, the desktop mode cannot be changed, presumably because it requires the disk image, although the App store also gives the option of Eagle Mode, in which a change in zoom changes the information displayed on the desktop.

Then, just as the desktop becomes available, exploring it reveals a fourth

stage. To access many of the default apps, such as the app store, the email client, and the cloud services, you must create a deepin account.

Desktop Logic

The DDE is written in Qt, and dde-kwin, its window manager, is a modification of Plasma's KWin. The whole of DDE is available in Fedora and Arch Linux, while UbuntuDDE is an unofficial Ubuntu flavor. Despite requiring a large amount of disk space, the DDE's

performance is roughly equivalent to Gnome or Plasma, even when run on a virtual machine. Deepin's appearance is universally praised and is popular enough that deepin has been on DistroWatch's top 100 distributions for page hits in the past seven years, usually somewhere in the top 20, although in the past six months it has slipped to 48 [2].

Much of the DDE will be familiar to the users of other desktops, such as the menu and the combined dock with favorite applications and the panel with its system tools and notifications. These widgets are concisely arranged using icons with mouseover labels and organized into sections. Clicking the system tool results in windows sliding from the edges of the screen, like the one for notifications, which is perhaps the most useful I have seen in any desktop environment (Figure 2). However, as soon as you start using it, the DDE reveals some characteristics of its own.

Probably the most noticeable characteristic is the DDE's default simplicity. For example, at login, DDE defaults to a tiling-like structure, with only one app displaying at a time (Figure 3), with larger apps occupying the entire screen. Some windows – although not all – can be resized by dragging on their edges, as in most desktops, but only one window remains visible, no matter what you do. To see all open windows, you

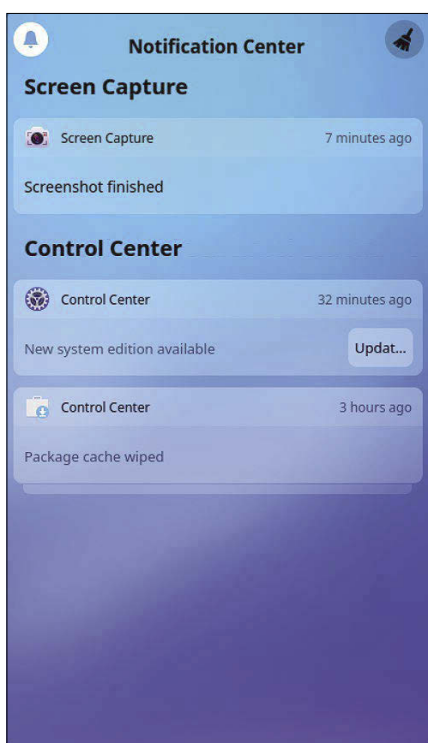


Figure 2: The notification pane is typical of the minimalist design of deepin's windows.

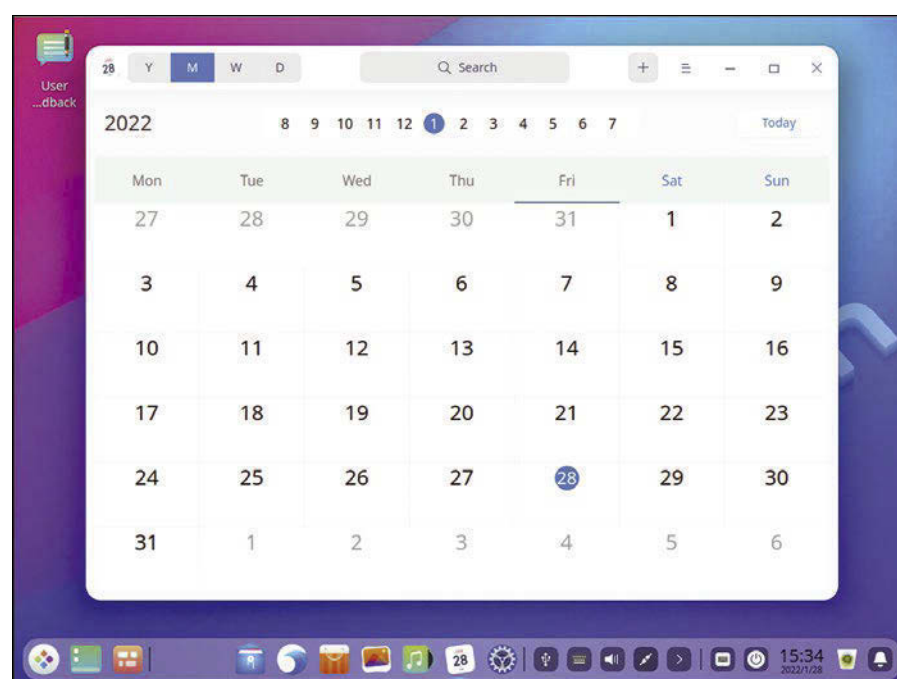


Figure 3: The main desktop displays one window at a time.

must click the multitasking view icon, the third from the left on the dock. From the multitasking view (Figure 4), you can see all the open windows arranged by the virtual desktop they are on, although none can be resized. Pressing the ESC key returns you to the desktop proper. Similarly, if you right-click the desktop, you can only add document and folder icons. To add an application's icon to the desktop, you need to right-click on it in the menu. Dock apps are added similarly. No doubt in the interest of avoiding clutter, little redundancy is included, leaving users to hunt for the logical place for a function. The function will usually be there and organized in a logical way, but you may have to search for it. Because the icons are more visible than the mouseover help, navigating is easier if you have a visual memory rather than a text-based one.

Still another outstanding feature is that, unlike many modern distributions, deepin develops an entire ecosystem of applications designed specifically to work together, much like GNOME or Plasma. In the last handful of releases, the DDE has replaced standard free software like Chrome and Thunderbird with its own equivalents. With version 2.4, the process is nearing completion, with only a few apps like Simple Scan borrowed from other distributions. The most noticeable exception is LibreOffice, which is too ubiquitous and too fully featured to have any serious substitute. Other standard apps like Gimp or digiKam are available in its app store, but deepin defaults to its own apps. Many of its apps, like the file manager, text editor, or music player, are minimally designed versions of what are available in most distributions. Others, like the

Boot Maker are basic tools for operations done at the command line. However, deepin also includes some unique apps, particularly in its utilities. For example, the dock installs a virtual keyboard by default, and the menu includes a recorder for voice notes as well as a combination screenshot and desktop recorder (Figure 5). Such apps have few, if any direct equivalences. In recent years, many of deepin's apps have found their way into distributions such as Debian separately from the complete desktop.

Debunking the Controversy

If you are a long-time Linux user, you might have stayed away from deepin. In 2018, deepin's app store was discovered to be sending general user information such as screen resolution to CNZZ, a Chinese analytics company. The collection of such information is common to most websites, but some users were concerned that personal information was harvested and even given to the Chinese government. However, no evidence was given that the deepin open source code did either of these things. Moreover, although it is true that information was gathered without user consent, the 20.4 release's installer now begins with a EULA and privacy policy that details exactly what anonymous information is collected. As I mentioned earlier, it is a good idea to read this policy so you know what will happen with your data. If you're comfortable with the uses outlined in the policy, there is no reason to avoid deepin because of the past controversy. To do so would be to miss out on one of the cutting edge distributions. As for the required disk space and peculiar logic, whether they are worth the aesthetics or the accompanying degree of customization is something that each user must decide for themselves. ■■■



Figure 4: The multitask view displays each window open on each virtual desktop.



Figure 5: Screen Capture, a combination screenshot and desktop recorder utility, is one of deepin's original applications.

Info

- [1] deepin:
<https://www.deepin.org/en/>
- [2] DistroWatch:
<https://distrowatch.com/>

Turn your ideas into reality!

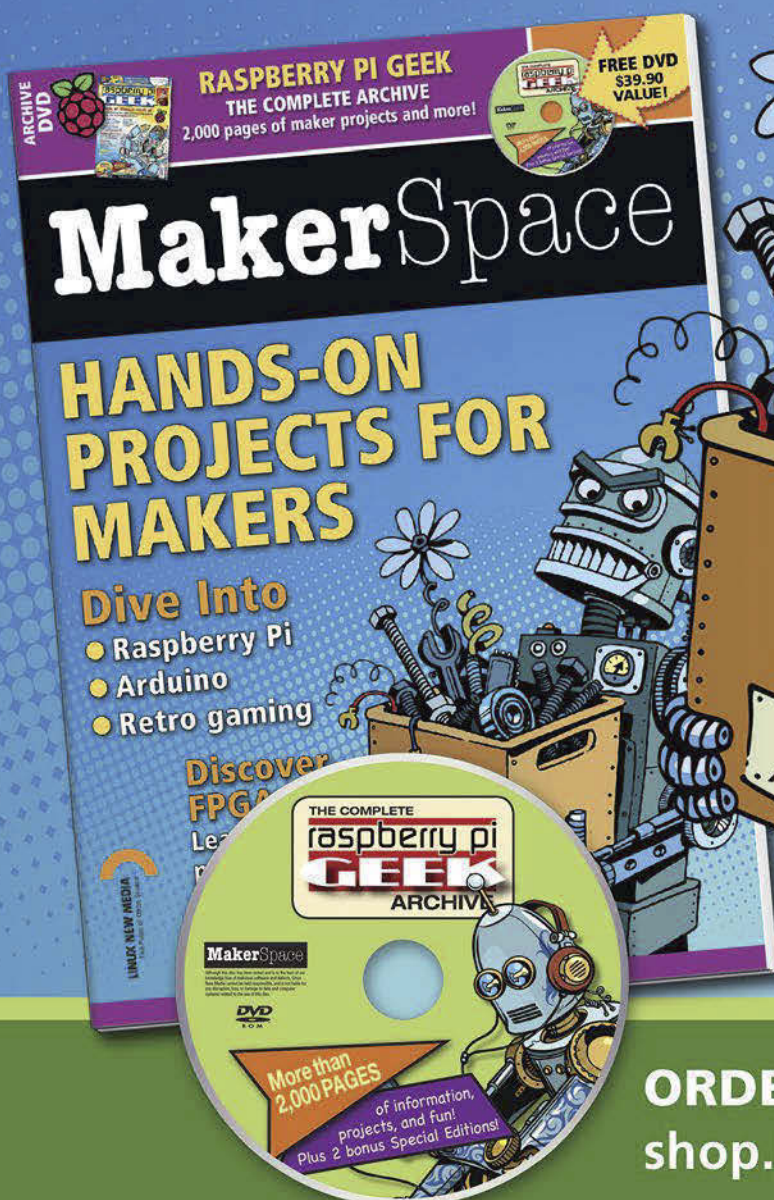
This is not your ordinary computer magazine! MakerSpace is a new special issue that focuses on technology that you can use to build your own stuff.

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

This special issue will help you dive into:

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

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



ORDER ONLINE:
shop.linuxnewmedia.com/specials



A professional DAW for Linux

Fat Beats

Linux users looking for a professional digital audio workstation can now take advantage of Reaper's large feature set. *By Claudius Grieger*

While no digital audio workstation (DAW) can do everything, Reaper [1] offers an extremely comprehensive feature set without compromising on quality. Reaper's strength lies in its attractive pricing, flexibility, low resource consumption, and possibly the largest DAW feature set on the market.

Until recently, Cockos, the development team behind Reaper, resisted the temptation to launch a Linux version, instead pointing to useful performance with Wine. However, with the Reaper 6 release, an unsupported alpha for Linux unexpectedly appeared. With the following minor release, Reaper now offers an official Linux version.

Fat Beats for Linux

Because Cockos does not bother with advertising, Reaper's price is low. A normal license costs \$60; if you make more than \$20,000 per year with Reaper, the identical program version costs \$225. One license is valid for two major versions. So, if you buy Reaper 6 now, you will also get Reaper 7 later on. Before you purchase Reaper, you can try out the program for 60 days without any restrictions or newsletter sign-up obligations.

The Linux version supports VST, VST3, and LV2 plugins, as well as ALSA, PulseAudio, and JACK. In principle, PipeWire is also compatible with Reaper, but there are some peculiarities. Reaper's interface seems quite rudimentary with a large amount of text and few effects or useless elements, which is another reason why

the program is very frugal in terms of resources. The installation only occupies about 100MB on your hard disk. Alternatively, you can install the software on a USB stick as a portable Live DAW.

Reaper doesn't come with a full feel-good package of plugins and virtual sound generators like Apple Logic Pro. Instead, Reaper offers a handful of VST plugins (Figure 1) and JS effects, Reaper's own plugin format (usually without a graphical interface).

Although Cockos offers some programs under a free license (GPL), Reaper is not one of them. According to lead developer Justin Frankel, who co-developed Winamp many years ago and then sold it to AOL, Reaper won't be offered as a GPL in the future either. For the core team, managing third-party code would involve a huge amount of extra work. However, Frankel also fought against a native Linux version for many years, so anything still seems possible.

Despite the non-free license, Reaper can be customized by the user more extensively than any other DAW on the market, without the user needing to master a programming language. While other audio programs may at most let you change the colors, Reaper lets you develop or exchange the complete theme yourself, including different views for the mixer and editor. You also can assign your own keyboard shortcuts to features and adapt them to your own preferences with macros and scripting.

The Reaper development team also responds to requests from the community.

The development team usually implements feature requests and bug reports quickly, as long as they are reported via the in-house forum [2]. It's not uncommon for several decimal point updates to be released in one day, and rarely more than a month passes between two updates. If you are having trouble using Reaper, the multilingual forum is usually the first point of contact. Much like the Ubuntu Users community, the Reaper community is large, helpful, mostly friendly, and accessible around the clock due to Reaper's international distribution.

Unique Selling Points

Reaper does a few things differently while retaining the classic work approach or basic user interface that may be familiar from Ardour, Pro Tools, Samplitude, or Cubase. In other words, the track control elements are located on the left in the editor, and the usual channel structure in the mixer remains. The workflow and look can be customized more comprehensively than in any other of the well-known DAWs. In addition, Reaper supports macros and several scripting languages (the native ReaScript/EEL, as well as Lua and Python). For this reason, Reaper has the reputation of being the "Linux of DAWs" in the audio world.

Unlike other DAWs, Reaper only uses one track type. Both audio and MIDI items can be stored and recorded on a track. Keeping the types separate still makes sense, but you don't need to choose the type up front. Each track can become an audio, MIDI, instrument, FX,

Lead Image © panupong_lithkai, 123RF.com

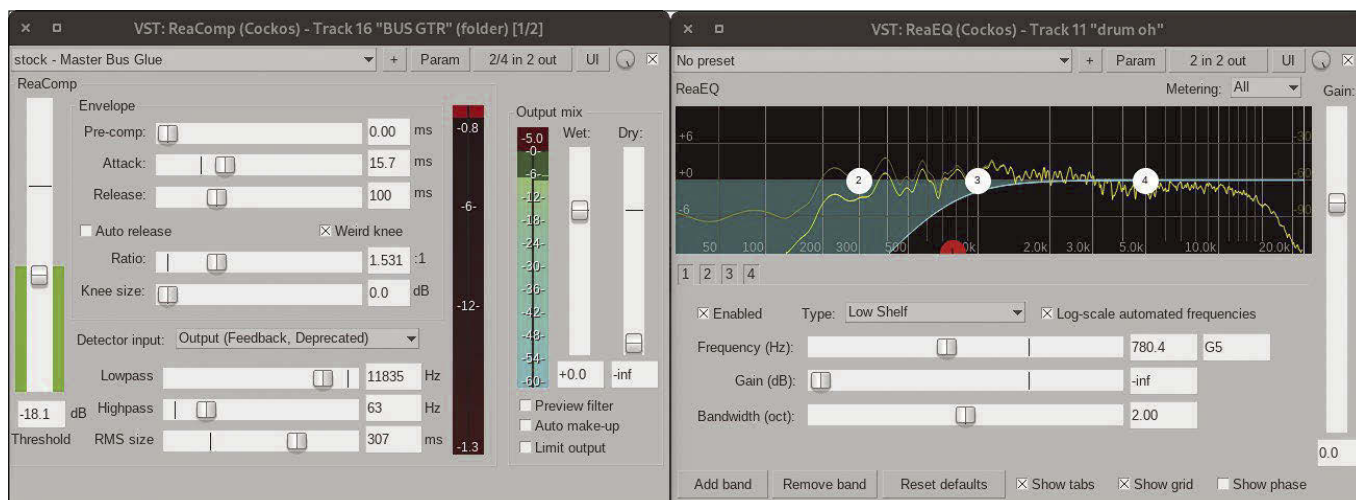


Figure 1: The plugins included with Reaper look fairly functional.

bus, VCA, or folder track – this takes some time getting used to. From my point of view, this artificial separation of track types in other DAWs only causes unnecessary worries anyway. Freely assignable track and item colors, track icons, and names give users a clear overview. There are also no separate areas for audio, MIDI, or instrument plugins in the mixer's respective plugin slots.

An item can be an audio or MIDI event that can be provided with item effects (FX). This way, you don't have to link effects to tracks as track FX, although you can if necessary. Once you get used to item FX, you won't want to do without them. Also, each item has its own volume setting, in addition to automation (Figure 2).

Reaper's resource consumption is comparatively low. When installed, Reaper occupies hardly more than 100MB; you can also set up a portable installation on a USB stick as a mobile DAW if required. When running, Reaper also consumes less RAM and CPU power than its competitors – which is definitely due in part to the no-frills interface. Plugins can also optionally be run as separate processes. If you are experimenting or using alpha versions, a buggy plugin won't take the entire DAW down with it.

All of this makes Reaper an excellent DAW for less powerful hardware or older systems. Even a 15-year-old laptop can easily handle less challenging audio work. The Linux CPU architecture choices include x86-64, i686, AArch64 (ARM64), and Armv7l (ARM32). Reaper even runs on a

Raspberry Pi 4, with some limitations due to the lack of compute power. Multitrack recording works better than overdubbing with real-time monitoring.

Operation

You control Reaper in the usual way with single clicks. Alternatively, you can improve the workflow experience with “new” methods like drag and drop. To do this, hold down the left mouse button and simply drag the mouse pointer over the corresponding controls. You can enable Solo, Mute, and Record Ready in the Editor or Mixer, while you can fade tracks in and out in the Track Manager (Ctrl + Shift + M). You can quickly set sends to the buses in the Routing Matrix (Alt + M) (Figure 3). Tracks that are selected together can also be controlled simultaneously, for example, using the volume or pan controls. In the end, this leaves you with more time for the creative process.

Reaper either launches directly into the last project, displays a selection box, or loads a new project. For a new project, the DAW records the tracks directly in a temporary directory. When you save, Reaper then dumps or copies the files into the desired directory. It is no problem to drag audio files with a 44.1 or 48 kHz sample rate into a project that is defined for some other setting – Reaper resamples on the fly in the background.

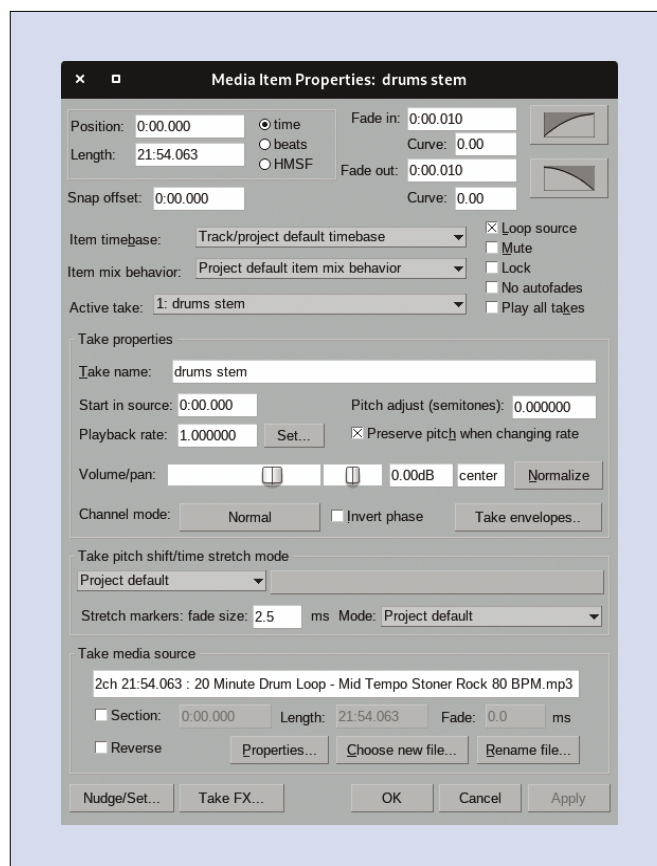


Figure 2: The Media Item Properties dialog lets you adjust item effects.

You can use modifier keys to make operations even faster. Other DAWs also use Shift, Alt, or Ctrl. Reaper, however, is more consistent and assigns at least one additional function to each control element, often even two or three. One of my favorites is Alt plus a left-click to delete markers, FX plugins, or automation points and enable the eraser in the MIDI Editor. Ctrl temporarily enables the pencil and draws empty MIDI items in the editor, manual curves in the automation tracks, or changes the velocity in one fell swoop in the MIDI Editor.

Right-click also offers a context menu for virtually all functions that are otherwise hidden in the menubar. Controls such as the play button, solo button, or toolbar buttons let you choose from a number of additional functions. In this sense, the motto for getting started is: Just try everything out and be surprised at what happens.

Reaper also has a very clever solution for tempo changes. An item's playback rate can be set manually in the item preferences by double-clicking or by pressing F2. Alt plus a left-click at the edge of an item lets you smoothly stretch or compress the item. Depending on the magnet function's setting in the standard toolbar in the upper left corner, the item is then glued to the visible grid, which can be temporarily deactivated via the

Shift key if required. Once again, freedom of choice is paramount here.

Themes

Reaper gives users the ability to develop entirely bespoke themes, providing a playground for people who enjoy designing their own desktops. The default interface is good and consistently usable, but unspectacular. Depending on your needs, you can either develop your own themes from scratch or use existing themes provided by Reaper users (Figure 4). Instructions for building your own themes can be found on Reaper Forums as a pinned post in the themes section. On the Reaper homepage, the *Stash* link takes you to a repository [3] where you can exchange other relevant goodies besides themes.

Some developers also offer their themes for sale via the forum. Some take wholly individual approaches, but there are also replicas of Pro Tools, Logic Pro, or, in the case of the House of White Tie Imperial [4] theme, a replica of an analog mixing desk (Figure 5). The Star Trek terminals look utterly wild. Ultimately, you have to decide for yourself if these themes are useful in everyday life.

Three themes are pre-installed: *Default*, *Reaper 1*, and *Reaper 5* (Figure 6). You can select one of these in the menubar below *Options | Themes*. New themes are

delivered in the .ReaperThemeZip format (a renamed ZIP file with a defined folder structure). To install, simply drag and drop the file into an active Reaper window or simply double-click on it in the file manager.

In the menubar under *Options | Layouts*, you'll also find alternative views for the mixer and editor, such as wider channel strips in the mixer, metering only, and fader only – whatever the theme developer wants is allowed here, too. The integrated screenset tool lets you store your own presets for recording, mixing, mastering, or other applications and call them up again as needed using a keyboard shortcut.

Actions

One the most important terms for Reaper newcomers is "actions." Everything in Reaper is an action: cutting audio, zooming, creating MIDI notes, copying elements, moving faders, switching tracks to record ready, and loading plugins. All actions can be assigned their own key combinations and combined as desired with custom actions or macros.

The Action List (Figure 7) can be opened via *Actions | Action List* or by pressing the ? key. A search box then helps you find actions and view, modify, and delete existing shortcuts, as well as create new ones using the *Add* button

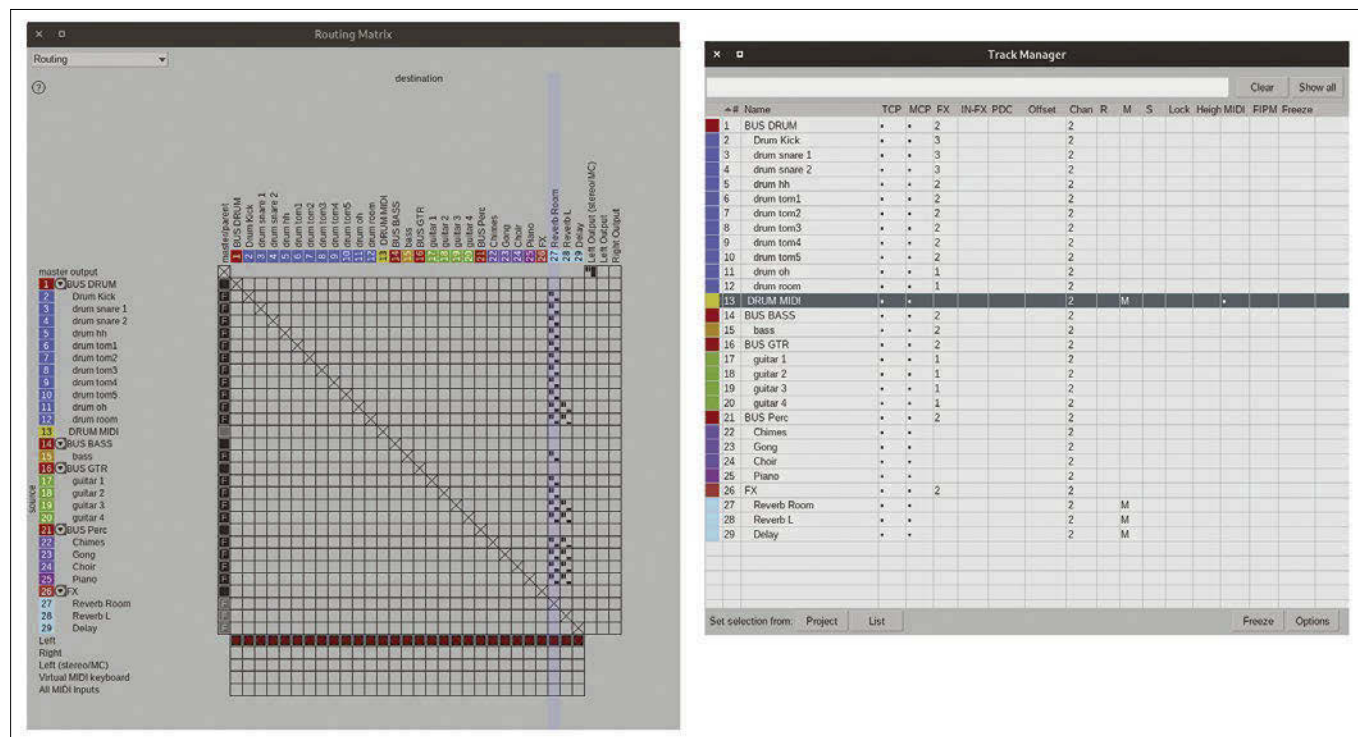


Figure 3: Routing and tracks are quickly set up thanks to drag and drop.

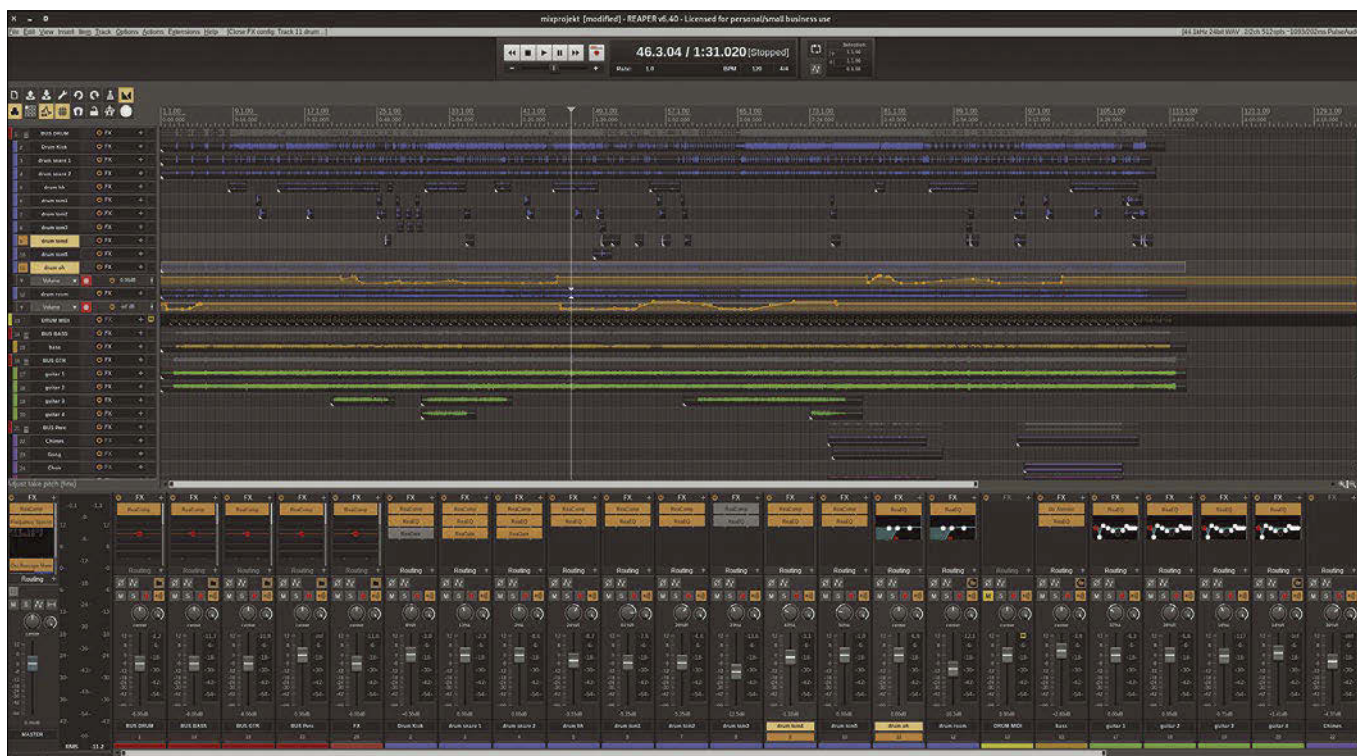


Figure 4: Reaper with the optional Echolot theme.

(see the “Tips for Beginners” box). It is also possible to search for the corresponding action based on the combination next to the search field. The biggest problem is finding out the function name. In addition, Reaper creates the lists for the *Main*, *Media Explorer*, and *MIDI Editor* sections separately, and they

cannot be edited across groups. To switch between Action List sections, use the toggle menu in the top right corner of the window.

Menus

Reaper leaves a lot to be desired when it comes to sensible menu structuring,

making it confusing for digital audio newcomers. As the number of features increases, the developers have simply filled up the menus without ever restructuring them with any kind of plan in mind. Fortunately, however, the menu structure can also be freely customized using *Options | Customize*

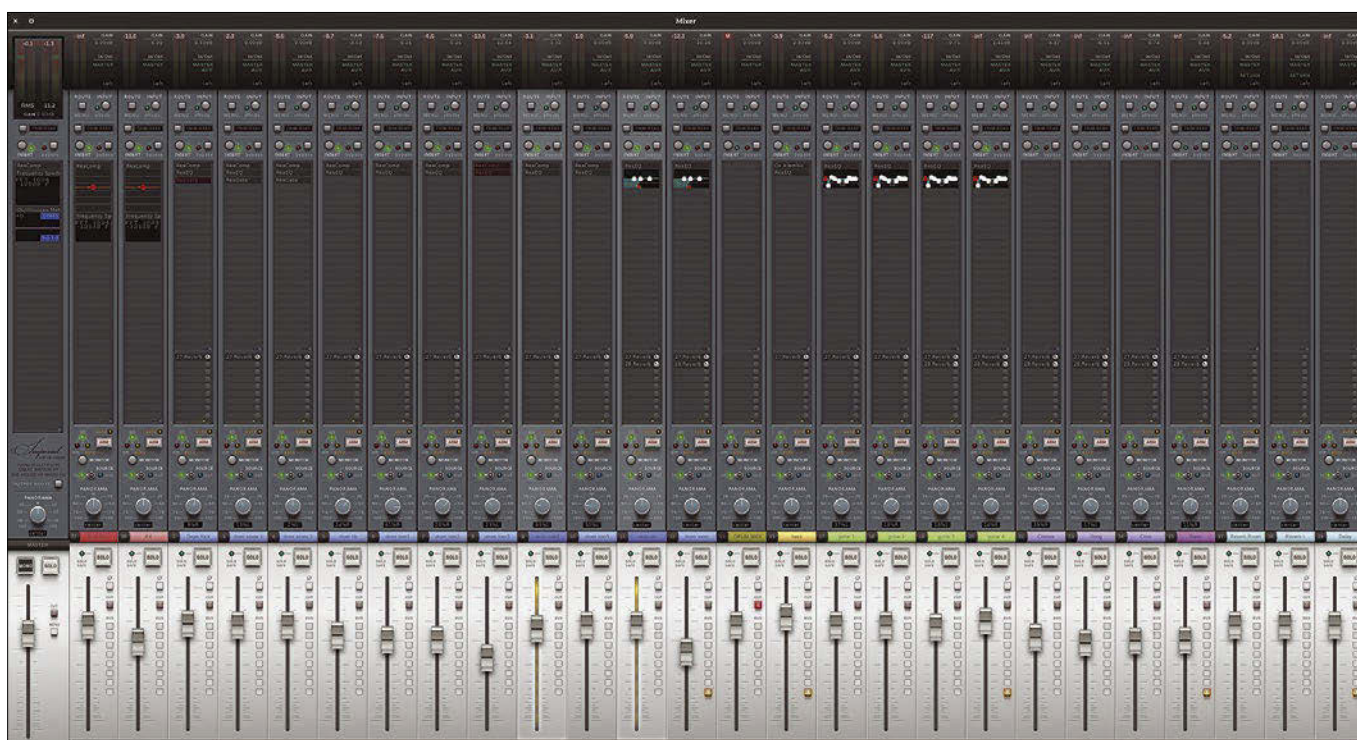


Figure 5: The House of White Tie Imperial theme is shown here with a feature-rich extended mixer.

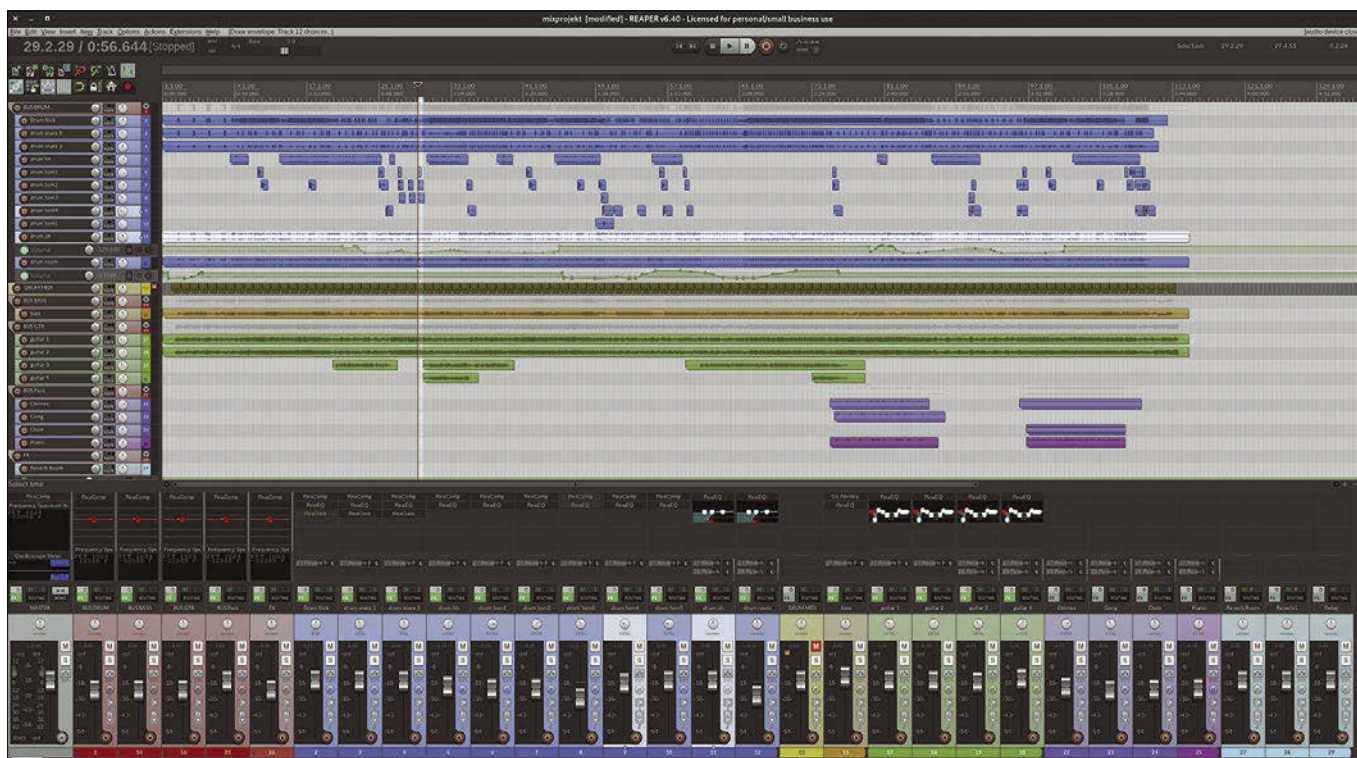


Figure 6: The Reaper 5 theme included in the default configuration.

menu/toolbar with drag and drop support (Figure 8). You can let your own preferences and typical use scenarios define the structure.

The free ReaMenus [6] add-on installs a fully structured menu into the DAW in the form of a configuration file. However, the independent ReaMenus tends to lag behind Reaper's development status by several versions. Consequently, you might find some functions missing from the overview. However, the missing functions can still be accessed via the menu editor or action list.

Tips for Beginners

After installing Reaper on a new system, you should disable vertical zooming via the mouse wheel. Almost all programs scroll when you rotate the mouse wheel, but Reaper does not. This is difficult to get used to, so you should assign vertical zooming to Ctrl plus the mouse wheel and vertical scrolling directly to the mouse wheel. Select Shift plus the mouse wheel for horizontal scrolling and Alt plus the mouse wheel for horizontal zooming. Actions can also be executed via a button in the toolbar. The free SWS Extension [5] not only adds some really useful tools to Reaper, but it also comes with some very useful actions – a must-install for Reaper pros.

Reaper also lets you assign individual actions to buttons in the toolbars. You can use right-click plus *Customize Toolbar* on the Main toolbar in the top left corner to add new buttons and optionally define your own icon.

Once you have customized Reaper, it is a good idea to export the configuration.

You can do this via *Options | Preferences* (or use Ctrl + P) and selecting *General*. With this backup, you can restore all program details with just a few clicks.

Linux Specifics

Since version 6, Reaper is now available for Linux as a standalone

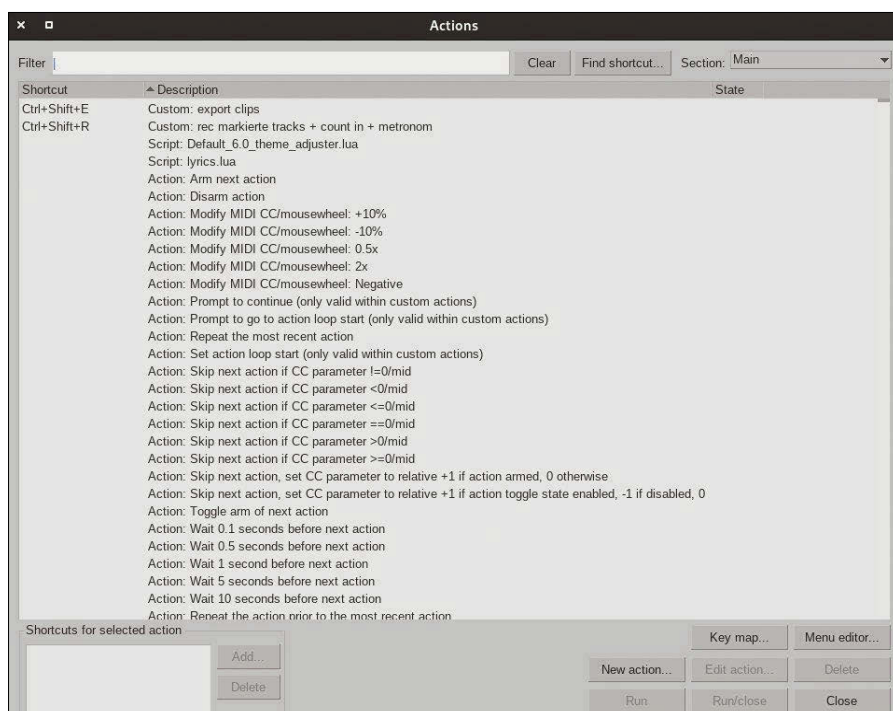


Figure 7: The Action List lets you define your own keyboard shortcuts for almost all actions.

program. As with Ardour, the Reaper installer is a script, which is run in a terminal window and guides you through the installation. In principle, this means that the developers offer compatibility with all distributions. However, depending on your system constellation, there may also be compatibility issues.

Basically, Reaper is a good companion for everyday audio work on Linux, as well as production environments, because Reaper can easily interact with ALSA, PulseAudio, and JACK. I gained Reaper experience and always achieved good results with kernels 5.4, 5.10 LTS, and 5.13. The test recordings I created amounted to 20 tracks, each five minutes in length and with up to five effects in the mix and bounce.

However, Reaper has issues with some system combinations. Specifically, Wayland and PipeWire are problematic. For example, Manjaro with GNOME 41 and Wayland doesn't allow you to drag and drop files from the file manager into the program. Fortunately, Reaper's built-in Media Explorer lets you do this (*View | Media Explorer* or `Ctrl + Alt + X`). The Media Explorer even offers some

additional functions, such as preview, tempo and pitch adjustment, and time selection of samples. If you use the classic X server, the problem does not occur on the same system.

Fedora relies on the PulseAudio successor PipeWire. As well thought out as this is at the desktop level, the audio interface is currently a poor choice for professional audio. Long often recordings lead to error messages and dropouts due to buffer overflow. Also, when mixing, there are repeated problems with oversampling of some plugins or crashes without an intelligible error message. However, this is probably not a specific Reaper problem because PipeWire also causes similar trouble with Ardour and all too often takes the DAW down with it into a proverbial black hole.

Xfce and various other desktop environments and window managers use `Alt` as the system's modifier key (e.g., to move or resize windows). This clashes with Reaper's defaults. For Reaper, you will want to replace this key on the system with the `Super` key (the Windows key), which Reaper does not otherwise use. Alternatively, re-

route all key combinations for actions from `Alt` to `Super`. It may be worthwhile for you to create your own audio user with an appropriately customized desktop configuration.

Reaper also has problems with high-resolution 4K displays and does not scale with the corresponding pixel density. Fonts and controls are

therefore displayed at an illegibly small size if you have a 4K resolution, and scaling the display size to 200 or 300 percent compromises the overview. The system font also has an influence on Reaper's display. Reaper takes advantage of the fact that Windows and macOS do not give users any real freedom in font selection. However, Linux users have a harder time. There are not typically any difficulties out of the box with KDE and GNOME. On the other hand, the display on Openbox, i3, and Xfce could do with some improvement, to say the least, if you don't have the right font settings.

Conclusions

Reaper is ready for use with Linux. Compared to the alternatives, Reaper offers a comprehensive, customizable, resource-saving, stable, and at the same time, inexpensive DAW with a classic concept. For a production environment, I recommend a recent installation of Manjaro or Ubuntu with kernel version 5.4 or newer and ALSA/JACK as the sound server. Depending on your typical work area and your own usage history, you may need to make some adjustments to your own preferences. And, don't forget that Reaper is not free software. If this doesn't bother you, you finally have one more choice on the professional audio market for Linux. ■■■

Info

- [1] Reaper: <https://www.reaper.fm>
- [2] Reaper Forums: <https://forum.cockos.com/forumdisplay.php?f=20>
- [3] Reaper repository: <https://stash.reaper.fm>
- [4] Imperial theme: https://www.houseofwhitetie.com/reaper/imperial/wt_imperial.html
- [5] SWS Extension: <https://www.sws-extension.org>
- [6] ReaMenus: <https://github.com/RCJach/reamenus>

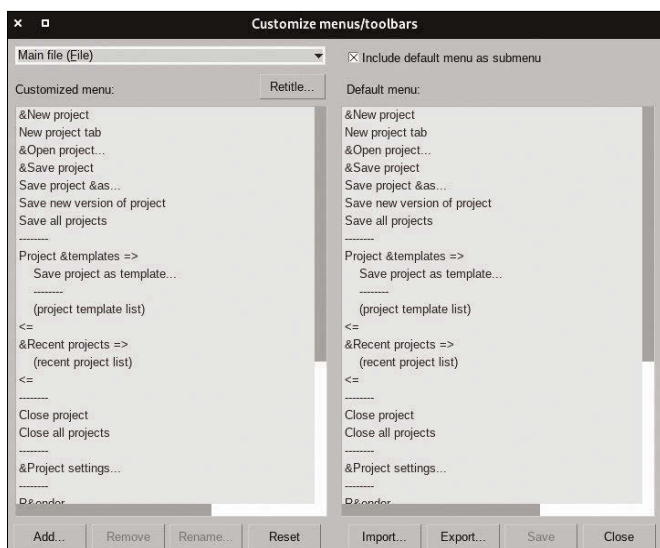
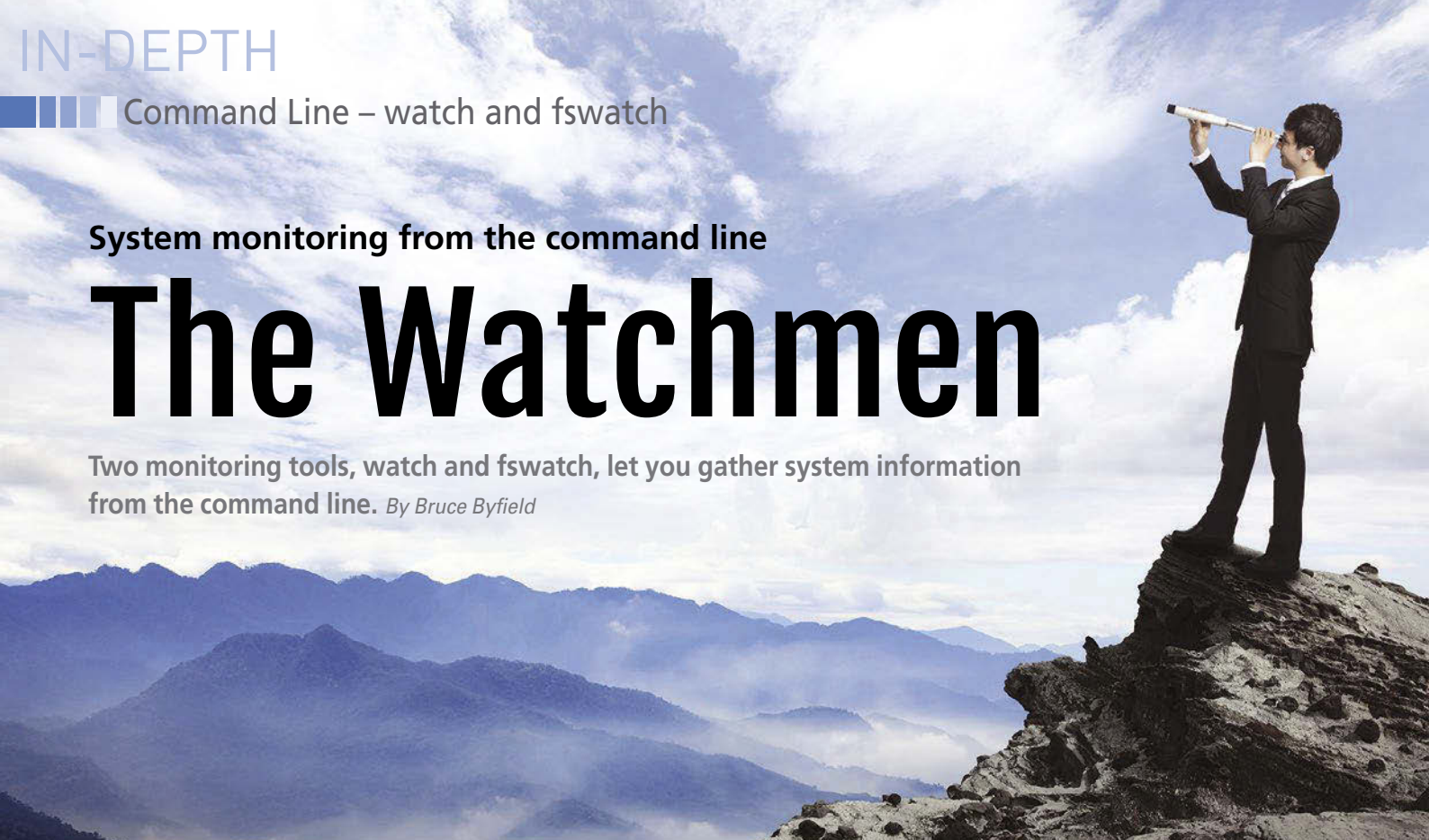


Figure 8: You can easily redesign confusing menus to meet your needs.



System monitoring from the command line

The Watchmen

Two monitoring tools, `watch` and `fswatch`, let you gather system information from the command line. *By Bruce Byfield*

Most users familiar with Linux have probably used `cron` or `at` to schedule the running of commands. Both can be useful in their place: `cron` for repeated scheduling of events and `at` for scheduling an event once. However, what both lack is the ability to gather system information and respond to it unless you write a specific script. Usually, it is much easier to use `watch` [1] and `fswatch` [2] to do both these things. While `watch` and `fswatch` can be used simply to gather information or to check for possible security incursions, both can be tweaked to act like a scheduler with little effort and minimal script-writing ability.

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/>.

watch

The purpose of `watch` is to follow how a command's output changes over time (Figure 1). This information can be used for troubleshooting, as well as for keeping a root or regular user informed about system changes as new packages are installed or updated. In limited circumstances, it could also be used as a simpler replacement for `at` or `cron`. Several other common uses are shown in Table 1. By default, `watch` runs every two seconds until closed or interrupted. The basic command structure is:

```
watch OPTIONS COMMAND
```

Depending on the command's contents, `watch` may need to be inside quotation marks. For example, a command would need quotes if it uses a pipe in order to run `less` or `grep`. Alternatively, instead of quotes, you could run `--exec (-x)`, so that a new process is not needed when the command contains multiple commands.

Two options set the nature of `watch`'s behavior. The most important is `--interval SECONDS (-n SECONDS)`. The `--interval` option overrides the default 2 seconds

between each time the command is run – an interval obviously chosen for immediate troubleshooting. However, on a computer that is always running, setting the interval to 86,400 would make `watch` run once per day, and setting the interval to 604,800 would make it run weekly, making `watch` serve the same function as `at` or `cron`. Either a comma or a period can be used to write large intervals; the minimal interval is .1 second. The only difference between `watch` and other schedulers is that you would need to remember to restart `watch` if the computer was ever shut down, which is a problem that `at` or `cron` do not have. For reasons that are not clear, the interval can be supplemented with `--precise (-p)` to make sure that the interval is precise – perhaps some testing might require that precision.

`watch` also supports options to customize output and exit behavior. With `--color (-c)`, output is color-coded. With `--no-linewrap (-w)`, long lines are truncated, while `--differences (-d)` highlights the latest output that differs from previous output. You can also remove the header showing the interval, command, current date, and time with

```
Every 60.0s: ls -l ./watch                                nanday: Sat Jan 22 13:06:04 2022
total 4
-rw-r--r-- 1 bb bb 136 Jan 22 13:04 distros.txt
-rw-r--r-- 1 bb bb   0 Jan 22 13:01 test.txt
```

Figure 1: Using `watch` to track the changes in a directory every 60 seconds.

Table 1: Everyday Uses for Watch

<code>watch -n 5 date</code>	Display the date every five seconds
<code>watch -n 60 from</code>	Watch for mail every 60 seconds
<code>watch -d ls -l</code>	Watch changes in a directory
<code>watch -d 'ls -l fgrep joe'</code>	Watch files owned by the joe account
<code>watch uname -r</code>	Watch for installation of a new kernel
<code>watch -d free -m</code>	Watch changes in disk spaces

`--no-title (-t)`. Exit options are equally varied. With `--chxexit (-g)`, `watch` exits when the output changes, which can be an obvious and handy indicator. Possibly, too, you may want `--beep (-b)` for a noise to indicate that `watch` has just exited with an error or `--erexit (-e)`, which stops output after an error occurs but waits to exit until any key is pressed.

fswatch

`fswatch` monitors changes to directories or files. The simplest way to use it is to run `fswatch` in one terminal and edit files in another.

As you start to use `fswatch`, you need to know something about how the command is structured and operates. `fswatch` is capable of using several different utilities. On macOS, it reports on information gathered by `FSEvents`. On BSD, it relies on the `kqueue` monitor. On Linux, it uses `inotify`, a Linux kernel subsystem, by default with the option of the `poll` monitor, which saves the time at which files were modified. All these monitors give similar information, although `fswatch`'s `man` and `info` pages warn that each has its own strengths and weaknesses, as well as its own bugs, all of which are described in detail in the help pages. You can use the `--list-monitor (-M)` option to see a list of available monitors and select which one to use with `--monitor NAME (-m NAME)`. However, the output, which displays in the terminal in which the command is running, generally differs little with the monitor.

Without any options, `fswatch` only records the files that have changed, but other options can add additional information, such as the event detected, and, optionally, the time the event was detected. Event types are self-explanatory. One action may have more than one event type. `fswatch` event types include:

- Created
- Updated
- Removed
- Renamed
- OwnerModified
- AttributeModified
- MovedFrom
- MovedTo
- IsFile
- IsSymLink
- Link

To help organize the output, you can use `--batch-marker CHARACTER` to separate out each loop of the command. In addition, `--print0 (-0)` can be used to ensure that lines are separated for easier reading.

The basic command structure is

```
fswatch OPTIONS PATHS
```

As well as specific paths, you can use select paths with regular expressions using `--include REGEX (-i REGEX)` or `--exclude REGEX (-e REGEX)`. Searches can be made case insensitive with `--insensitive (-I)` and include subdirectories with `--recursive (-r)`. If the watched files include symbolic links, `fswatch` will follow them if the `--follow-links (-L)` option is added. You can also use `--timestamp (-t)`

to add the local time to the output or `--utf-time (-u)` to add the time in UTC format. With either time option, you can structure the date using `--format-time FORMAT (-f FORMAT)`, using the `strftime` codes [3]. Other useful options are `--one-event (-1)`, which exits `fswatch` after one set of events, and `--latency SECONDS (-l SECONDS)`, which must be at least .1 seconds. Unlike `watch`, `fswatch` does not give any output, except for briefly outlining the tab of another terminal whose present working directory is open.

Often, the basic information generated by `fswatch` is useful by itself. However, like `watch`, `fswatch` can be used to issue commands. It does so by piping it through `xargs`, whose purpose is to issue other commands. Table 2 shows four common examples cribbed from `fswatch`'s online help [4].

Two More For the Toolbox

If you prefer to work from a desktop environment, Gnome offers `command-runner-applet` with approximately the same functionality as `watch` and `fswatch` [5]. But `command-runner-applet` is not a single command; according to its GitHub page, it takes over the desktop while running, although mouse and keyboard actions will run after it completes.

Both `watch` and `fswatch`, on the other hand, offer a wider range of functionality within a single command, and `fswatch` in particular offers in-depth reporting options. The main difference, of course, is that `watch` provides a unified way to monitor with commands, while `fswatch` is concerned mainly with the management of directories and files. Each, though, is yet another example of how the command line can offer more than the desktop. Although relatively unknown, each is a useful addition to the administrative toolbox. ■■■

Table 2: fswatch and xargs

Action	Command	Comments
Run a Bash command	<code>fswatch FILE-PATH xargs -n 1 COMMAND</code>	Usually for creating, updating, or deleting files
Watch one or more files and/or directories	<code>fswatch PATHS **/*.js xargs -n 1 bash_command</code>	–
Print the absolute paths of the changed files	<code>fswatch PATH xargs -n 1 -I {} echo {}</code>	–
Filter by event type	<code>fswatch --event DIRECTORY-PATH xargs -n 1 bash_command</code>	Usually for creating, updating, or deleting directories

Info

- [1] `watch`: <https://linux.die.net/man/1/watch>
- [2] `fswatch`: <https://www.mankier.com/1/fswatch>
- [3] `strftime` code: <https://strftime.org/>
- [4] `fswatch` examples: [https://www.mankier.com/1/fswatch#Examples_\(TL;DR\)](https://www.mankier.com/1/fswatch#Examples_(TL;DR))
- [5] `command-runner-applet`: <https://github.com/porridge/command-runner-applet>

The sys admin's daily grind: Last words

Time to Say Goodbye

After two decades as a sys admin columnist, Charly bids a fitting farewell with a login banner created with FIGlet. By Charly Kühnast

When I log in to servers or network components, I am often greeted by a large font banner with variable information content. Sometimes it simply tells me the hostname, sometimes the kernel version, and sometimes it reads *DO NOT REBOOT BETWEEN 8AM AND 8PM*.

Of course, it is debatable whether banners make sense or are just nonsense, and I use them sparingly on my own systems. However, servers that I can't just play around with warn me with a banner that reads *Abyssus abyssum invocat* (one mistake leads to another).

Test systems for evaluating new software, on the other hand, cheerfully greet me with *Morituri te salutant* (those who are about to die, salute you). I've fared quite well with that banner all these years, but how do you create one?

I use FIGlet [1] for this. All of the popular distros include this tool, which supports several fonts and offers a variety of design options. I don't use any of them – I just want to get an unambiguous message across when logging in. That's why I only use the `-c` parameter to center the banner, which would otherwise be left-justified. For my farewell banner, I make an exception and also use the `-D`

parameter, which lets you display German umlauts. I can usually ignore these with my Latin calendar sayings. FIGlet replaces special characters such as `[`, `\`, and `]` with `Ä`, `Ö`, and `Ü` while `{`, `|`, and `}` give you the corresponding lowercase letters. The call

```
figlet -c -D Vielen Dank f}r 20 Jahre
```

produces the banner shown in Figure 1 (see Figure 2 for the English version).

So, that's all folks. I'm saying goodbye to my work on this column, which I started with the February 2002 issue of the *Linux Magazine* German edition and am now ending with the February 2022 issue. I am taking this step because I would like to write less frequently in the future, but in greater detail. I would like to thank the entire editorial team for their support, especially Jan Kleinert and Jörg Luther, the editors-in-chief for the *Linux Magazine* German edition. But the biggest thanks go to you, dear Linux friends! ■■■



Figure 1: Charly bids a fitting farewell with a FIGlet banner in German ...

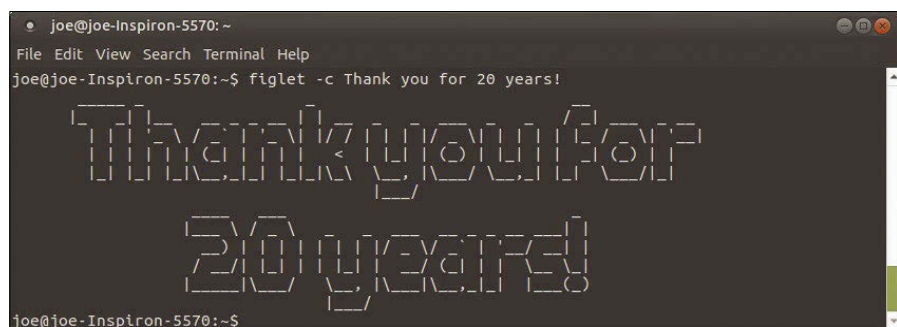


Figure 2: ... and then in English.

Info

[1] FIGlet: <http://www.figlet.org>

Author

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





Easy monitoring with Checkmk

Check It

We'll show you how to use the Checkmk open source monitoring tool to monitor your home router.

By Martin Hirschvogel

Every organization needs monitoring to make sure that servers, networks, applications, cloud assets, and other elements work as they should. Monitoring also provides timely alerts, and it helps IT teams track down the causes of (potential) problems. Having a suitable monitoring tool will help you resolve problems faster, ideally before they have an impact on operational systems. Checkmk [1] is an open source monitoring tool that is ideally suited for modern hybrid environments, combining enterprise-grade scalability and security with the extensibility of open source software.

In most cases, Checkmk runs on a dedicated server or virtual machine (VM). As I am just running a small monitoring setup for my home office environment, my host will be my computer with Ubuntu 18.04. The computer has four CPU cores and 4GB of RAM, which is more than enough to get started. Checkmk also runs on other Linux distributions, such as Debian, Red Hat Enterprise Linux, CentOS, or SUSE Linux Enterprise Server, and you can also run Checkmk on a Docker container or virtual appliance. If you install Checkmk

on a dedicated host, the only additional steps you need are transferring the files to the server.

Checkmk can monitor anything with an IP address, including servers, cloud assets, and network devices, as well as systems that belong to the Internet of Things (IoT). For this article, I chose to monitor my TP-Link router. Checkmk also comes ready-equipped with reasonable thresholds for alerts. These pre-configured thresholds will come in handy, because your monitoring will be up and running within a few minutes without you having to worry about setting up alerts.

Choose a Checkmk Edition

You can use Checkmk for free. For this tutorial, I chose the Checkmk Raw Edition. You could also start with the Checkmk Free Edition, the free version of the Checkmk Enterprise Edition, which provides additional features. After 30 days, the Free Edition is limited to 25 hosts. The Checkmk Raw Edition is completely open source and permanently free.

Checkmk started as a front-end for the Nagios [2] open source monitoring tool several years ago, but it became a stand-alone monitoring solution in 2012 and has replaced almost all Nagios components. The Checkmk Raw Edition still uses the Nagios monitoring core as a utility. To get started with Checkmk, go to the download section on the project website [3], where you will find the latest versions. Select the Checkmk Raw Edition, choose the latest stable version (this tutorial is based on version 2.0.0p13), and then choose your platform and your OS version. Click the download button and download the package.

Once the download has finished, you will need to install the package, including all of dependencies. Installation steps vary depending on your distro. In my case, I will switch to the terminal and go to my download folder, and then run `apt install` as root (see Figure 1):

```
apt install \
./check-mk-raw-2.0.0p13_0.
bionic_amd64.deb
```



```

root@FerdinandPC:/home/ferdinand/Downloads# apt install ./check-mk-raw-2.0.0p13_0.bionic_amd64.deb
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'check-mk-raw-2.0.0p13' instead of './check-mk-raw-2.0.0p13_0.bionic_amd64.deb'
The following NEW packages will be installed:
  check-mk-raw-2.0.0p13
0 upgraded, 1 newly installed, 0 to remove and 46 not upgraded.

```

Figure 1: Installing Checkmk.

Once the installation is completed, you can perform a test. The command `omd` should be accessible now. The following command:

```
omd version
```

should return the details of the Checkmk version you are currently using.

The `omd` command stands for Open Monitoring Distribution (OMD), an open source project created by Mathias Kettner, the founder of Checkmk [4]. The OMD project includes monitoring solutions assembled from various components.

Create Your Checkmk Monitoring Site

The next step is to start an initial monitoring site (a site is also known as an instance). Make sure you are still root and use `omd create` to create a new Checkmk site and name it as you wish (Figure 2). This tutorial uses `cmk_tutorial`:

```
omd create cmk_tutorial
```

At the end of the output, you should now see the information about how to start and access your Checkmk site (Figure 3). You can follow the steps to change your admin password in the terminal, but I prefer to do that in the Checkmk user interface.

Copy the randomly generated password (you will need it in the next step), and start your monitoring site with `omd start`:

```
omd start cmk_tutorial
```

Should you want to drill deeper into Checkmk later on, it is important to understand what has just taken place. By creating a new Checkmk site, you have created a new user, which is known as the site user, and a group with the name of the site on your server. A directory for the site will have been created under `/omd/sites`, (e.g., `/omd/sites/cmk_tutorial`). Checkmk also copied its default configuration into the new directory, and a user with the name `cmkadmin` has been created for the Checkmk web interface.

Start Monitoring with Checkmk

Before you go to the Checkmk user interface, think about the systems you want to monitor. For this tutorial, I will monitor my TP-Link TD-W9960v router with the classic protocol Simple Network Monitoring Protocol (SNMP). If you also want to monitor systems with SNMP, you should make sure the SNMP agent is active on your device. In my case, I had to activate it in my router's web interface, and I decided to keep the SNMP community set to the default value `public`. (In a real-world setting, you will want to change this community string rather than using the default value.)

Once you are sure the host you want to monitor is ready to provide the monitoring data, it is time to switch to the Checkmk user interface in your browser. Every Checkmk site has its own URL, which consists of your monitoring server's IP address or host name and the name of the Checkmk site. In my example, this is `127.0.0.1/cmk_tutorial`.

```

root@FerdinandPC:/home/ferdinand# omd create cmk_tutorial
Adding /opt/omd/sites/cmk_tutorial/tmp to /etc/fstab.
Creating temporary filesystem /omd/sites/cmk_tutorial/tmp...OK
Updating core configuration...
Generating configuration for core (type nagios)...Precompiling host checks...OK
OK
Executing post-create script "01_create-sample-config.py"...OK
Restarting Apache...OK

```

Figure 2: Creating a monitoring site.

```
Created new site cmk_tutorial with version 2.0.0p13.cre.
```

```

The site can be started with omd start cmk_tutorial.
The default web UI is available at http://FerdinandPC/cmk_tutorial/

```

```

The admin user for the web applications is cmkadmin with password: E34JgAQl
For command line administration of the site, log in with 'omd su cmk_tutorial'.
After logging in, you can change the password for cmkadmin with 'htpasswd etc/htpasswd cmkadmin'.

```

Figure 3: The installation ends with instructions for how to access the site. Be sure to change the password – either at the command line or through the Checkmk user interface.

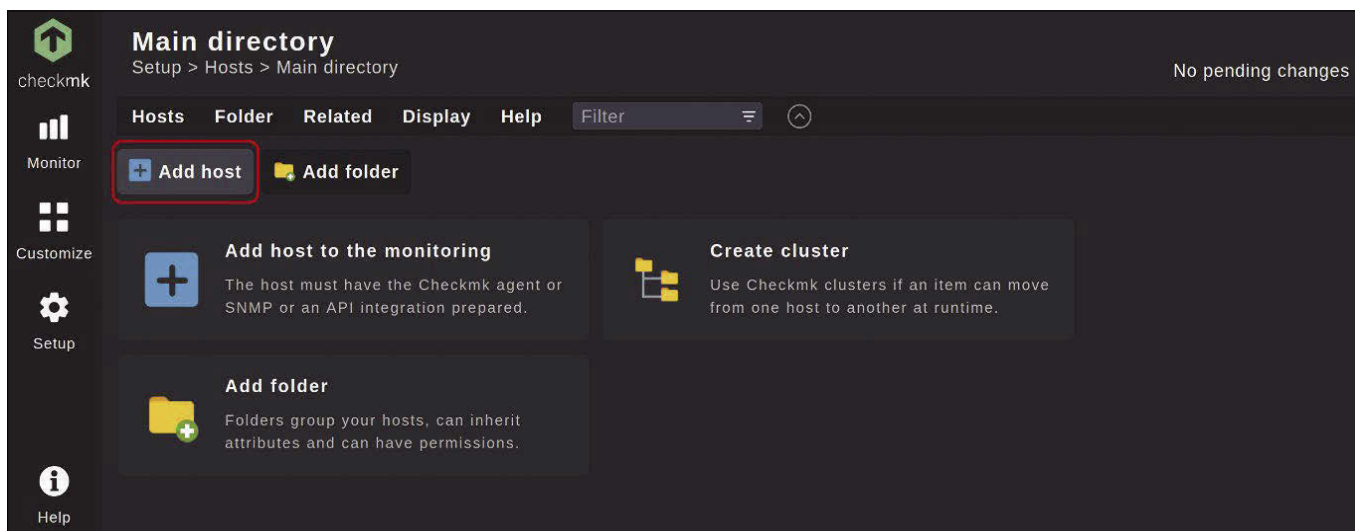


Figure 4: Adding a host for Checkmk to monitor.

Open the link to your Checkmk site in your browser. You can open the link shown on your terminal. Log in with `cmkadmin` and the password you copied from the terminal. You will see an empty dashboard. Go to *User* in the sidebar on the left and, under *Profile*, click on *Change password* to change your password (recommended).

After that you can start adding hosts to your monitoring (Figure 4). In my case, I will add my router. Go to *Setup* | *Hosts* and click on *Add host*

In my example, I only have to add the hostname `tplinkmodem.net`, because TP-Link set up the DNS for their devices and, thus, Checkmk can resolve the IP address for your hostname

automatically. I recommend using this option, if available. If that is not the case, you will also need to add your device's IP address by activating the checkbox next to *IPv4 address* and entering the IP address.

For my router, I want to work with SNMP, but by default, Checkmk assumes that you use the Checkmk agents to monitor systems. The Checkmk agents are great for monitoring servers, but on most network devices, you cannot install additional software. Thus, go to *Monitoring agents*, check the box *Checkmk agent/API integrations*, and choose *no agent*.

Next, check the box labeled *SNMP* (Figure 5). Choose your SNMP

version – *SNMP v2 or v3* works in most cases. Because I did not change the SNMP community, I can leave the SNMP credentials box unchecked – by default, Checkmk assumes that the password is *public*. If that is not the case, you have to set the community string under SNMP credentials. Leave the other items unchanged.

Next, click on *Save & go to service configuration*. Checkmk will now automatically discover any relevant monitoring services on that host, and you should then see a list of *Undecided services*.

Click on *Fix all* to monitor all of these relevant services (Figure 6). That will add all of the detected services to the monitoring and will also remove

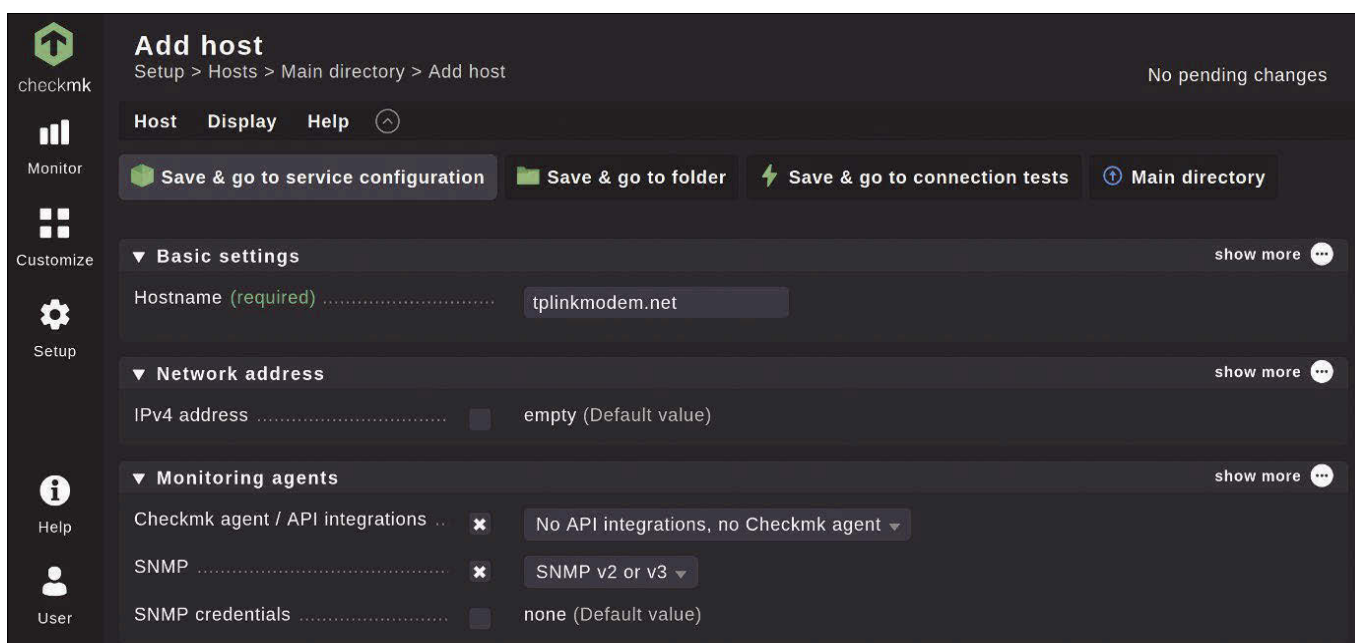


Figure 5: Choose a monitoring agent in the Add host dialog box.

services that have vanished. You can manage the services manually, of course, but the *Fix all* function makes it a lot easier.

In this case, Checkmk simply detected all of my router's ports that are currently online, plus the uptime and the SNMP info check – 10 services in total for me. If there is an official check plugin available for your device, Checkmk will detect relevant information automatically. The 2,000 official check plugins cover the equipment from most common professional network vendors. This collection of plugins makes Checkmk a great tool for network monitoring and saves you a lot of time.

I recommend the blog post “Network Monitoring with Checkmk” [5] for further information on monitoring with SNMP. But before you start with outside reading, follow the last step to complete this tutorial.

Activating Changes

Finally, you need to activate the changes by clicking on the highlighted field with the yellow ! at the top right corner that counts the number of changes. After that, click on *Activate on selected sites*, and you will have successfully added the first host to your monitoring.

The activation of changes is a safety mechanism. All changes made will be listed under *Pending changes* (Figure 7). You can review any listed changes before they affect your monitoring. Checkmk differentiates between the Setup menu as a configuration environment, in which you manage the hosts, services and settings, and the Monitor menu, in which the actual operational monitoring takes place. New hosts and other changes in the configuration initially have no effect on the monitoring. You need to activate these before they can go into production.

You find your host now under *Monitor* | *Overview* | *All hosts*. Click on a

Services of host tplinkmodem.net
Setup > Hosts > Main directory > Properties of host tplinkmodem.net > Services of host tplinkmodem.net 1 change !

Actions Host Settings Display Help

Full service scan Monitor undecided services Remove vanished services Properties of host tplinkmodem.net

Full scan finished after 14.3 s at 2021-10-27 15:42:52.
Job details ▶

10 rows

Undecided services: 10 | Vanished services: 0 | New host labels: 0 | Vanished host labels: 0 | Changed host labels: 0

Fix all

Undecided services (currently not monitored)

State	Service	Status detail
OK	Interface 06	[eth0], (up), MAC: D8:07:1E:1E:4B:7A, Speed: 100 MBit/s

Figure 6: Click the *Fix all* button to monitor all relevant services.

Activate pending changes
Setup > Activate pending changes 3 changes !

Changes Related Display Help

Activate on selected sites

Currently there are 3 changes to activate.

Activation status

Actions	Site	Status	Version	Changes	Progress	Details
✕	Local site cmk_geekflare	online	2.0.0p13	3		Activation needed

Pending changes 3 rows

Time	User	Change	Affected sites
2021-10-27 15:43:08	cmkadmin	Saved check configuration of host 'tplinkmodem.net' with 10 services	All sites
2021-10-27 15:43:06	cmkadmin	Updated discovered host labels of 'tplinkmodem.net' with 0 labels	All sites

Figure 7: You'll have a chance to review pending changes before you activate them.

host, and you will see an overview of all the monitoring services of that host. Figure 8 shows the overview of my router. You can click on each service to see more details.

Conclusion

This article is intended as a proof of concept. I used the SNMP monitoring agent in the example because SNMP is supported on most home routers. However, be aware that there are some security concerns with SNMP. The US government Cybersecurity and Infrastructure Security Agency recommends that you only use SNMPv3 and has outlined additional precautions [6].

This tutorial ends here, but your real experience with monitoring has only just begun. If you want to continue, you should add your host server to the monitoring as well. Checkmk provides some lightweight, but powerful monitoring agents for server monitoring.

You will find the agents for various operating systems, such as Windows and a number of Linux distributions, via the sidebar by clicking on *Setup | Agents | Linux*.

After you install, the procedure is similar: Add the host that you want to monitor, but you can leave the box *Checkmk agent/API integrations* unchecked and should not switch that to SNMP. By default, Checkmk assumes you are using Checkmk agents to monitor systems.

The Checkmk documentation [7], as well as in the official Checkmk

forum [8], will provide answers for all your Checkmk questions. ■■■

Info

- [1] Checkmk:
<https://checkmk.com>
- [2] Nagios:
<https://www.nagios.org/>
- [3] Checkmk download:
<https://checkmk.com/download?edition=cre&version=stable&dist=ubuntu&os=focal>
- [4] OMD:
<https://omdistro.org/>
- [5] "Network Monitoring with Checkmk":
<https://checkmk.com/blog/network-monitoring-with-checkmk-2-0>
- [6] Reducing the Risk of SNMP Abuse:
<https://www.cisa.gov/uscert/ncas/alerts/TA17-156A>
- [7] Checkmk documentation:
<https://docs.checkmk.com/latest/en/>
- [8] Checkmk user forum:
<https://forum.checkmk.com/>

Services of Host tplinkmodem.net
Monitor > Overview > All hosts > tplinkmodem.net > Services of Host

Commands Host Services Add to Export Display Help

Acknowledge problems Schedule downtimes Filter Show checkboxes tplinkmodem.net

State	Service	Icons	Summary	Age	Checked	Perf-O-Meter
OK	Check_MK		[snmp] Success, execution time 1.6 sec	22 m	42.0 s	1.62 s
OK	Check_MK Discovery		no unmonitored services found, no vanished services found, no new host labels	20 m	20 m	
OK	Interface 06		[eth0], (up), MAC: D8:07:B6:1E:1E:7A, Speed: 100 MBit/s, In: 0.00 B/s (0%), Out: 0.00 B/s (0%)	22 m	40.0 s	0.00 bit/s / 0.00 bit/s
OK	Interface 07		[eth0.2], (up), MAC: D8:07:B6:4B:4B:7A, Speed: 100 MBit/s, In: 0.00 B/s (0%), Out: 131 B/s (<0.01%)	22 m	40.0 s	0.00 bit/s / 1.05 kbit/s
OK	Interface 08		[eth0.3], (up), MAC: D8:07:B6:1E:1E:7A, Speed: 100 MBit/s, In: 0.00 B/s (0%), Out: 131 B/s (<0.01%)	22 m	40.0 s	0.00 bit/s / 1.05 kbit/s
OK	Interface 09		[eth0.4], (up), MAC: D8:07:B6:4B:4B:7A, Speed: 100 MBit/s, In: 0.00 B/s (0%), Out: 131 B/s (<0.01%)	22 m	40.0 s	0.00 bit/s / 1.05 kbit/s
OK	Interface 10		[eth0.5], (up), MAC: D8:07:B6:1E:1E:7A, Speed: 100 MBit/s, In: 0.00 B/s (0%), Out: 131 B/s (<0.01%)	22 m	40.0 s	0.00 bit/s / 1.05 kbit/s

Figure 8: Viewing all the monitoring services on a host.

Adjusting cell phone photo orientation with Go

Sunny Side Up

Cell phones often store photos upside down or sideways for efficiency reasons and record the fact in the Exif metadata. However, not all apps can handle this. Mike Schilli turns to Go to make the process foolproof. *By Mike Schilli*

Somehow, I seem to hold my cell phone wrong when taking pictures. It often happens that a picture that looks right in the phone's photo gallery is suddenly upside down or on its side when I want to send it to friends on WhatsApp. A quick look at the metadata of the photo in question reveals what's going wrong (Listing 1) [1].

It appears that my phone is saving images with the wrong orientation, but it is also storing the correction value, which shows up in the JPEG format's Exif header to tell you how the photo actually should be rotated. I don't know who came up with this idea, but assuming that any app looks in the

Exif header first and then aligns the image correctly seems like a fundamental flaw to me. There's no doubt in my mind that the original camera app should perform the rotation when the picture is taken, rather than having the work done downstream time and time again by an unmanageable number of photo apps.

Upside Down World

Figure 1 illustrates that the cell phone still displays the image the right way round in its photo app, even though it was stored incorrectly. The desktop version of the WhatsApp client in the browser, however, obviously ignores the associated Exif header and sends the picture upside down (Figure 2). Imagine the recipient's reaction to receiving a message about my surf trip



to Ocean Beach in San Francisco with a photo on its head!

High time to write a Go program that correctly orients a photo retrieved from a phone and then deletes the Exif header entirely, so that any downstream application will process the image as-is – very much like Gimp (Figure 3) does when it sees a rotated photo. While I'm at it, I'll throw in some algorithms for rotating images 90 or 180 degrees: After all, someone might ask about them in your next job interview.

Listing 1: Photo Metadata

```
$ exiftool pic.jpg | grep Rotate
Orientation: Rotate 180
```

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.

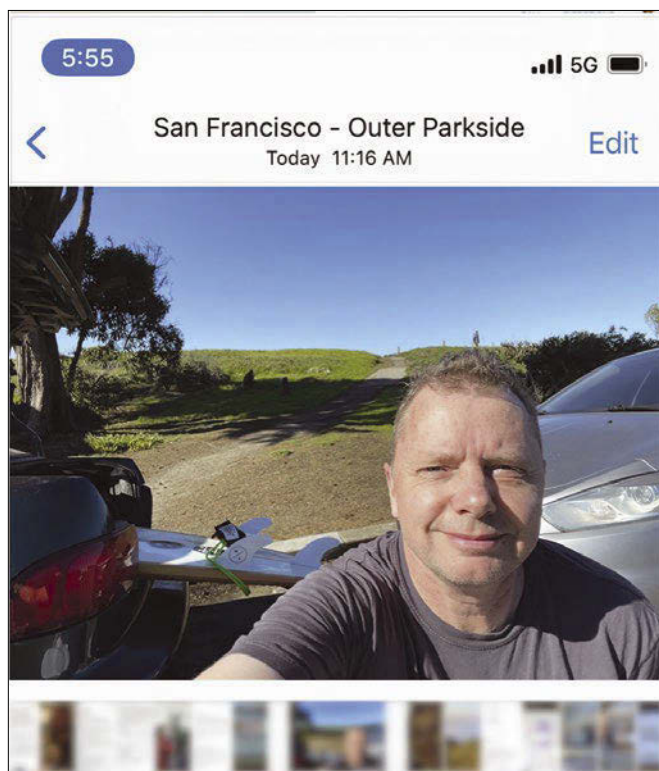


Figure 1: My smartphone shows the image the right way round ...

Photo by Girl with red hat on Unsplash

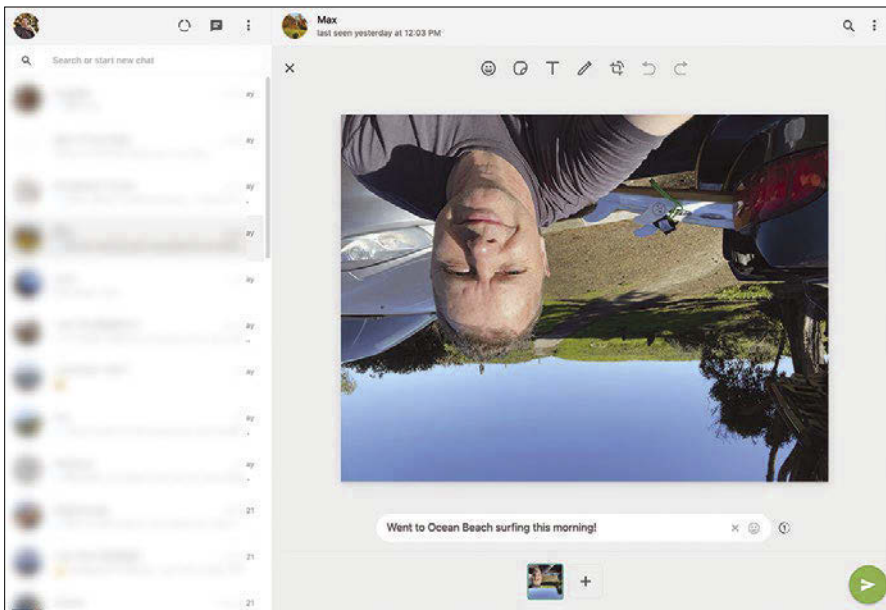


Figure 2: ... but WhatsApp turns the photo on its head.

A digital photograph is ultimately an m by n matrix of pixel values. To turn it upside down (i.e., to perform a 180-degree rotation), an algorithm simply swaps pixel values above and below the center line. The X values of the pixels in an image traditionally run from left to right, while the Y values increase from top to bottom. This means that the origin of the photo can be addressed as $(0,0)$ top left, while the bottom right corner is at the coordinates $(w-1, h-1)$, where w specifies the image width and h the image height, each in pixels.

Double Flip

But be careful: If you simply swap the Y values for the rotation, you will end up with a mirror image of the original photo. Instead, small X values (pixels in the upper left corner) need to be put in the lower right corner for a 180-degree rotation. This means that the rotation algorithm needs to mirror both the Y values and the X values. A pixel located in the upper half of the image with the coordinates $(x0, y0)$ thus ends up in the lower half of the image at position $x1, y1$. The distance from $y0$ to the center line is equal to the distance from $y1$ to the

center line. At the same time, $x0$ is as far from the left edge of the image as $x1$ is from the right.

Listing 2 shows the algorithm that turns a JPEG photo upside down. The two nested for loops starting in line 13 work their way through the Y values from 0 to the bottom of the image and through the X values from 0 to the right edge. These coordinates correspond to the $x0$ and $y0$ positions of the original pixels. Inside the double loop, line 15 retrieves the original pixel value in `jimg.At(x,y)`. To rotate the image, it stores the value at the mirrored position in the newly created, modifiable photo `dimg` at position $x1, y1$. The position is calculated as the distance of $x0$ from the right or $y0$ from the bottom edge of the image. The algorithm efficiently mirrors the image at both the horizontal and vertical center lines, turning it upside down as desired.

Writeable Copy

When Go reads a JPEG photo from disk, it usually ends up in a pixel array that can't be modified. But since the algorithm wants to manipulate the photo, line 11 in Listing 2 first creates a writable photo with the same dimensions as the original using `NewRGBA()` but leaves it empty. Then the `rot180` function keeps calling `jimg.At()` for all pixels of the original and transfers them to the target photo, mirrored one by one, with `Set(x,y,value)`. Line 19 returns the

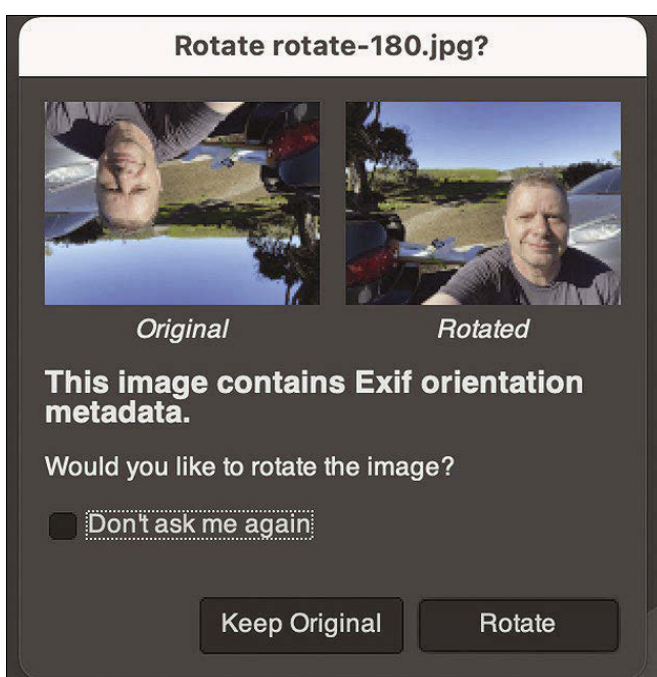


Figure 3: Gimp notices that something is wrong.

Listing 2: rotate-180.go

```
01 package main
02
03 import (
04     "image"
05 )
06
07 func rot180(jimg image.Image) *image.RGBA {
08     bounds := jimg.Bounds()
09     width, height := bounds.Max.X, bounds.Max.Y
10
11     dimg := image.NewRGBA(bounds)
12
13     for y := 0; y < height; y++ {
14         for x := 0; x < width; x++ {
15             dimg.Set(width-1-x, height-1-y, jimg.At(x, y))
16         }
17     }
18
19     return dimg
20 }
```


finished rotated image as a pointer to the calling main program.

This covers the algorithm for rotating a photo in memory by 180 degrees. But how does the image, which is compressed in JPEG format on disk, find its way into RAM as a pixel matrix in the first place? The `imgMod()` function in Listing 3 takes the name of the photo file in `srcFile`, the name of the target file as the second parameter, and, as the third parameter, a callback function that performs the desired rotation of the image in RAM.

Functions are full-featured data types in Go and can easily be passed to other functions, which basically tells Go: “Here’s the algorithm to apply to the

original image data.” Listing 3 opens the original file for reading in line 11, uses `Decode()` to decode the JPEG data from the standard `image/jpeg` package, and stores the results into a matrix pointed to by the `jimg` variable if there are no errors. Line 21 then calls the function previously passed in as a parameter with the variable `cb`, passes the image data in `jimg` to it, and receives the modified data of the then rotated image in `dimg`. All that remains is to create a new destination file, `dstFile`, and write the JPEG-encoded data for the modified photo in line 30.

Rotating 90 Degrees

Not all incorrectly saved images are upside down. Sometimes the images are also on their sides and need to be rotated by 90 degrees. In these cases, calling `exiftool` will display something like *Orientation: 90 CW* for the JPEG file. This indicates that you need to rotate the image 90 degrees

clockwise to display it with the correct orientation. In Figure 4, Gimp determines that a photo of the little-known Billy Goat Hill park in San Francisco needs to be rotated a quarter turn to the right and offers to complete the task right away.

But how does a quarter rotation of the pixels work in a 2D matrix? Figure 5 shows schematically how the first row of pixels with the values (1,2,3,4) ends up as the rightmost column in the result after rotating the matrix clockwise by 90 degrees.

While the algorithm converts rows into columns, the target image’s dimension also changes. Cell phone photos are typically rectangular rather than square, and a photo rotated 90 degrees not only changes its pixel values but also swaps the width and height of the resulting overall image. Listing 4 accounts for this by having line 10 swap the dimensions in bounds in the X and Y direction, so that the modifiable target image generated in line 12 by `NewRGBA()` already has the dimensions of the rotated rectangle rather than those of the original.

Changing X to Y

The double for loop starting in line 14 of Listing 4 iterates line by line through the source image, retrieves the current

Listing 3: `imgmod.go`

```
01 package main
02
03 import (
04     "image"
05     "image/jpeg"
06     "log"
07     "os"
08 )
09
10 func imgMod(srcFile string, dstFile string, cb
    func(image.Image) *image.RGBA) {
11     f, err := os.Open(srcFile)
12     if err != nil {
13         log.Fatalf("os.Open failed: %v", err)
14     }
15
16     jimg, _, err := image.Decode(f)
17     if err != nil {
18         log.Fatalf("image.Decode failed: %v", err)
19     }
20
21     dimg := cb(jimg)
22     if err != nil {
23         log.Fatalf("Modifier failed")
24     }
25
26     f, err = os.Create(dstFile)
27     if err != nil {
28         log.Fatalf("os.Create failed: %v", err)
29     }
30     err = jpeg.Encode(f, dimg, nil)
31     if err != nil {
32         log.Fatalf("jpeg.Encode failed: %v", err)
33     }
34 }
```

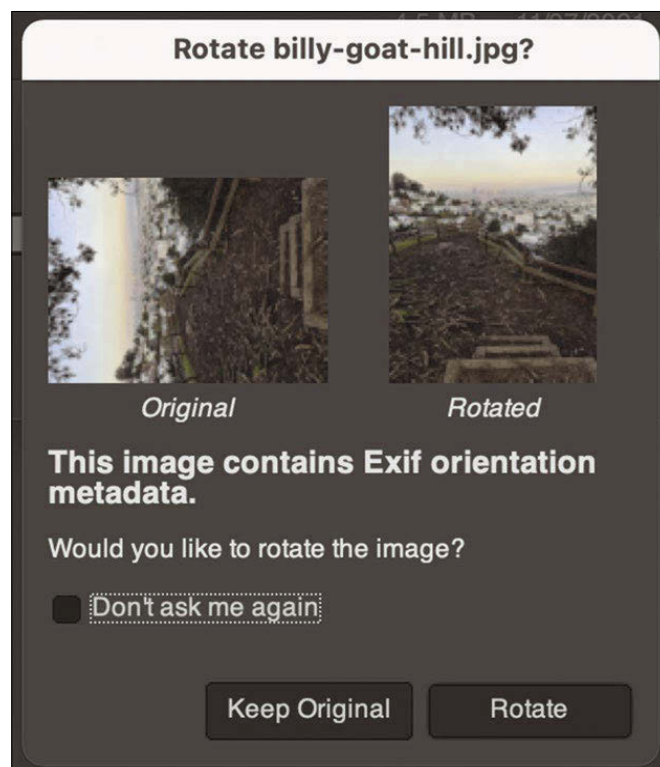


Figure 4: Gimp notices that the image needs to be rotated by 90 degrees.

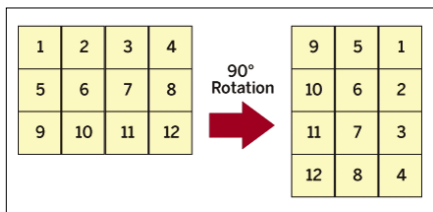


Figure 5: Matrix transformation after rotating an image 90 degrees clockwise.

pixel values by calling `jimg.At()`, and uses `dimg.Set()` to store them column by column from right to left in the target matrix – it's as simple as that.

With these two algorithms now cast in code, the main program in Listing 5 can fetch a photo from disk, read the Exif headers, determine whether there is an `Orientation` tag in the headers, and initiate rotation to the corrected format. If you have used `exiftool` to read the JPEG photos' Exif headers thus far, you might be surprised to find that the true value of an `Orientation` tag in the Exif headers is not a string with the degree information at all, but an integer value. Common settings are the values 6

(90 degrees clockwise), 3 (180 degrees), or 8 (90 degrees counter-clockwise). Other values of the standard, which also supports mirrored photos, are shown in Figure 6, but they occur less frequently in practice.

The Exif tags hidden in a JPEG photo are not easy to read, but fortunately the Go community on GitHub provides some libraries that simplify the task to calling a `Decode()` function. Listing 5 shows the main `autorot` program, which uses the `goexif` library provided by GitHub user `rwcarlsen`. I used a similar open source product for image manipulation in an earlier column [2].

Using `data.Get()`, Listing 5 in line 34 retrieves the value of the `Orientation` tag from the mess of Exif tag data. If the tag is not found, there is nothing to do, because the image already has the correct orientation. However, if the program finds a tag, line 40 fetches the first integer value of the tag (index 0), and the `switch` construct starting in line 45 determines what kind of corrective rotation (90 or 180 degrees) the image needs and calls the modifier function

Listing 4: rotate-90.go

```
01 package main
02
03 import (
04     "image"
05 )
06
07 func rot90(jimg image.Image)
08     *image.RGBA {
09     bounds := jimg.Bounds()
10     width, height := bounds.Max.X,
11         bounds.Max.Y
12     bounds.Max.X, bounds.Max.Y =
13         bounds.Max.Y, bounds.Max.X
14     dimg := image.NewRGBA(bounds)
15
16     for y := 0; y < height; y++ {
17         for x := 0; x < width; x++ {
18             org := jimg.At(x, y)
19             dimg.Set(height-y, x, org)
20         }
21     }
22     return dimg
23 }
```

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?

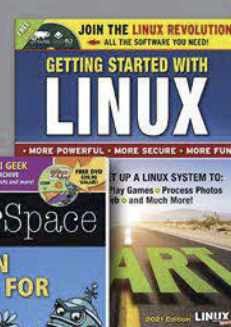
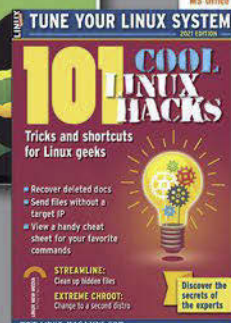
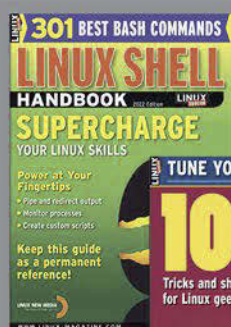
Searching for that back issue you really wish you'd picked up at the newsstand?

➤ shop.linuxnewmedia.com

DIGITAL & PRINT
SUBSCRIPTIONS



SPECIAL EDITIONS



Listing 5: autorot.go

```

01 package main
02
03 import (
04     "flag"
05     "fmt"
06     "github.com/rwcarlsen/goexif/exif"
07     "os"
08     "path"
09 )
10
11 func main() {
12     flag.Usage = func() {
13         fmt.Printf("Usage: %s jpg-file\n", path.Base
14             (os.Args[0]))
15     }
16
17     flag.Parse()
18     if len(flag.Args()) != 1 {
19         flag.Usage()
20     }
21
22     jpgFile := flag.Arg(0)
23
24     f, err := os.Open(jpgFile)
25     if err != nil {
26         panic(err)
27     }
28
29     data, err := exif.Decode(f)
30     if err != nil {
31         panic(err)
32     }
33
34     orient, err := data.Get(exif.Orientation)
35     if err != nil {
36         fmt.Printf("No orientation header found.\n")
37         os.Exit(0)
38     }
39
40     val, err := orient.Int(0)
41     if err != nil {
42         panic(err)
43     }
44
45     switch val {
46     case 3:
47         imgMod(jpgFile, jpgFile, rot180)
48     case 6:
49         imgMod(jpgFile, jpgFile, rot90)
50     default:
51         panic("Unknown orientation")
52     }
53 }

```

`imgMod` with the appropriate algorithm function as a parameter. The calls to `imgMod()` in Listing 5 use identical names for the source and target files so `autorot` simply overwrites the original

files. If that seems too dangerous, you can rename the target file to `.bak`, and you're guaranteed no accidents.

When Go's `image` library writes the JPEG image back to disk, it completely omits the original Exif data, so there's no longer an `Orientation` header there, indicating correct rotation.

Listing 6: Compiling `autorot`

```

$ go mod init autorot
$ go mod tidy
$ go build autorot.go rotate-90.go
rotate-180.go imgmod.go

```

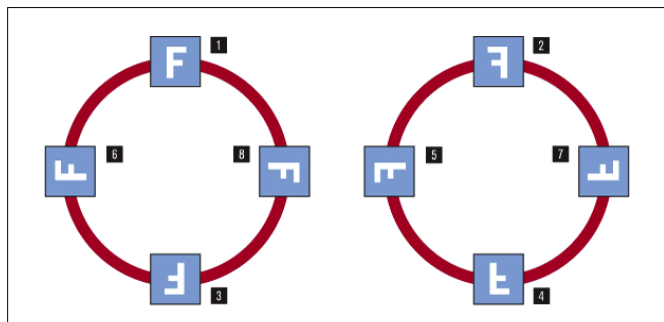


Figure 6: Integer values in the Exif header indicate the photo's orientation.

without any problems and will run there without any complaints, given a similar operating system and processor architecture.

Conclusions

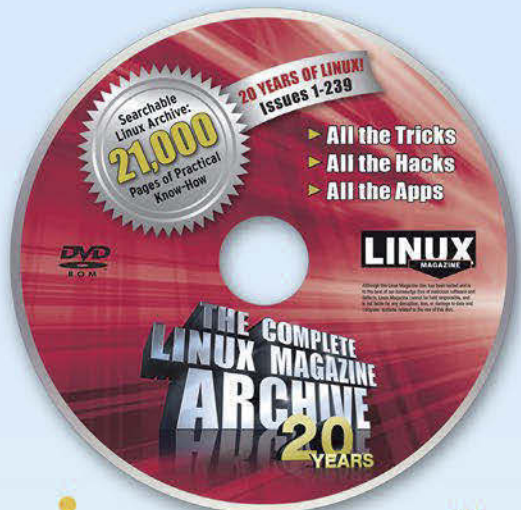
The program is by no means perfect yet. It'll work for most cases, but still lacks counterclockwise rotation and algorithms for more exotic Exif values. But with the described basics, these features can easily be added. As always, Go imposes no limits on the hobbyist here. ■■■

Info

[1] Listings for this article:

<ftp://ftp.linux-magazine.com/pub/listings/linux-magazine.com/257>

[2] "Manipulating Stored Geocoordinates in Cellphone Photos" by Mike Schilli, *Linux Magazine*, issue 240, December 2020, <https://www.linux-magazine.com/Issues/2020/240/Obfuscation-Filter>

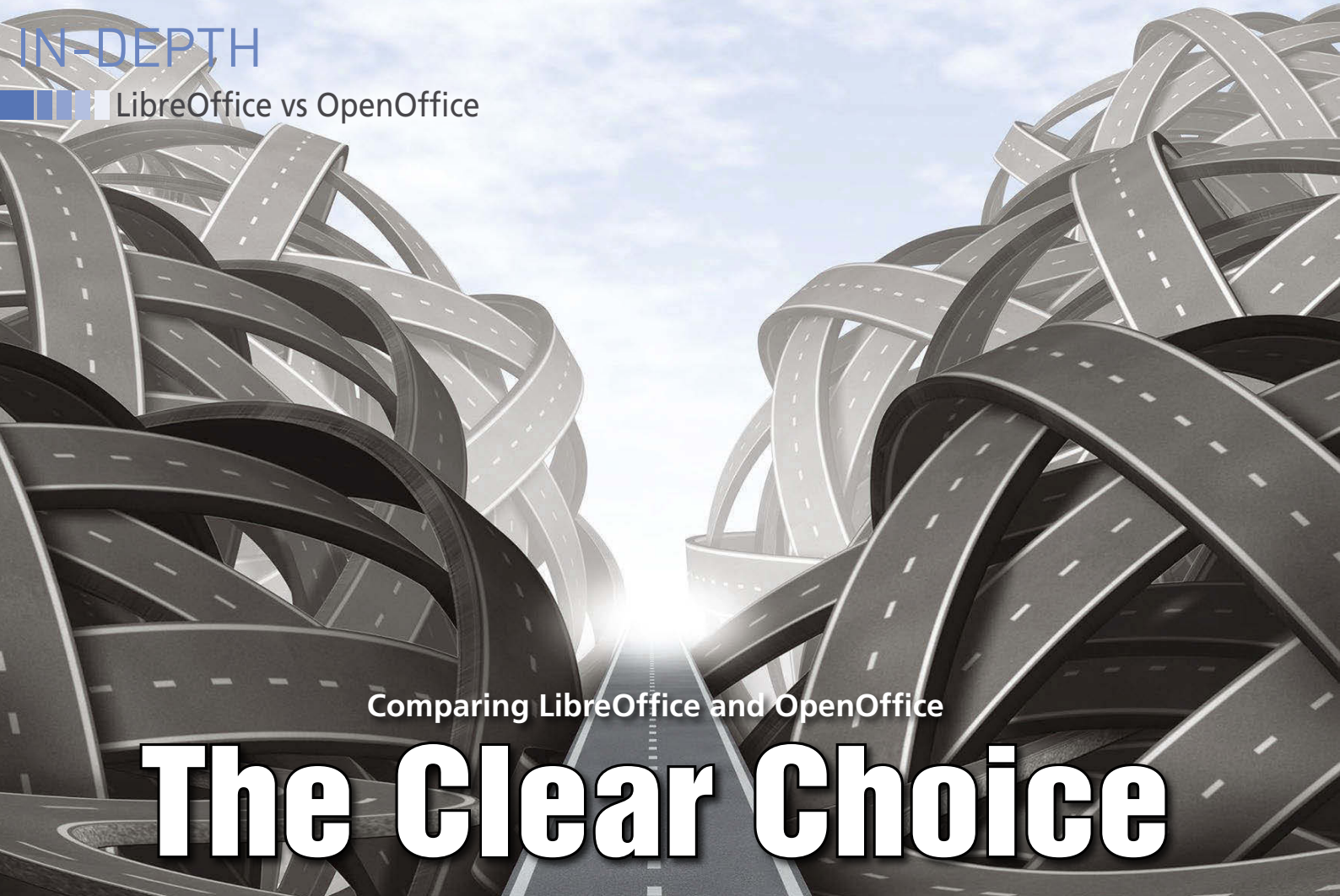


LINUX MAGAZINE ★ 20 YEAR ARCHIVE DVD ★

ORDER NOW!

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





Comparing LibreOffice and OpenOffice

The Clear Choice

While LibreOffice and OpenOffice have a shared past, LibreOffice outstrips OpenOffice in contributors, code commits, and features. *By Bruce Byfield*

A search for comparisons of LibreOffice and Apache OpenOffice returns over 8.3 million results. That number comes as no surprise, given that LibreOffice and OpenOffice are the best-known open source office suites and share a common past. However, what is surprising is how shallow many of the comparisons are. Many offer only a superficial glimpse at either office suite from the viewpoint of an unsophisticated and undemanding user. Often, the comparisons are obsolete. Also important to note is that many comparisons strive for a false sense of objectivity by declaring that any differences are minor. However, by every possible standard, LibreOffice outshines OpenOffice and shows OpenOffice to be outdated. To pretend otherwise is a distortion of the truth.

As you might know, the two office suites share a common history (Figure 1). Both originate in OpenOffice.org, a project run by Sun Microsystems from 2000-2011. In fact, OpenOffice claims to be the legal descendant of OpenOffice.org because in 2011 Sun

passed OpenOffice.org to Oracle, which in turn passed it on to the Apache Foundation – but the concept of ownership has little relevance in open source. At the same time, Go-oo, a semi-official fork that had operated quietly since 2007, created LibreOffice and its governing body, The Document Foundation. Go-oo had been contentious in OpenOffice.org because it advocated a faster pace of development, so the creation of the two new projects seemed to formalize a division that already existed. Certainly LibreOffice and OpenOffice showed little love for one another from the beginning.

Also from the start, OpenOffice faced challenges that LibreOffice did not. To start with, OpenOffice uses the Apache License, while LibreOffice uses a dual LGPLv3/Mozilla Public License (MPL). A major consequence of this difference in licenses is that LibreOffice can borrow code from OpenOffice, while OpenOffice cannot borrow code from LibreOffice. Just as importantly, from late March 2014 to March 2015, OpenOffice had a total of 16 contributors, 12 of

whom stopped contributing when IBM stopped supporting OpenOffice [1]. Since then, OpenOffice has been reluctant to publish the number of contributors, but, considering that OpenOffice claimed 120 contributors in 2012 [2], it is doubtful that the number today is more than a small fraction of LibreOffice's 1,722 [3]. The result is predictable: OpenOffice has not had a major release since 2014 and only 11 minor ones. In comparison, LibreOffice has had 13 major releases and 87 minor releases. And in 2019, LibreOffice had over 15,000 code commits and OpenOffice had 595 [4]. Equally inevitably, LibreOffice has attracted important corporate supporters such as Red Hat and Collabora while OpenOffice has been unable to fill the void left by IBM eight years ago. OpenOffice's continued existence has been quixotic, even admirable, but the project is simply not structured to compete.

The Feature Disparity

Aside from Go-oo's contributions a decade ago, the two projects initially

Lead Image © lightwise, 123RF.com

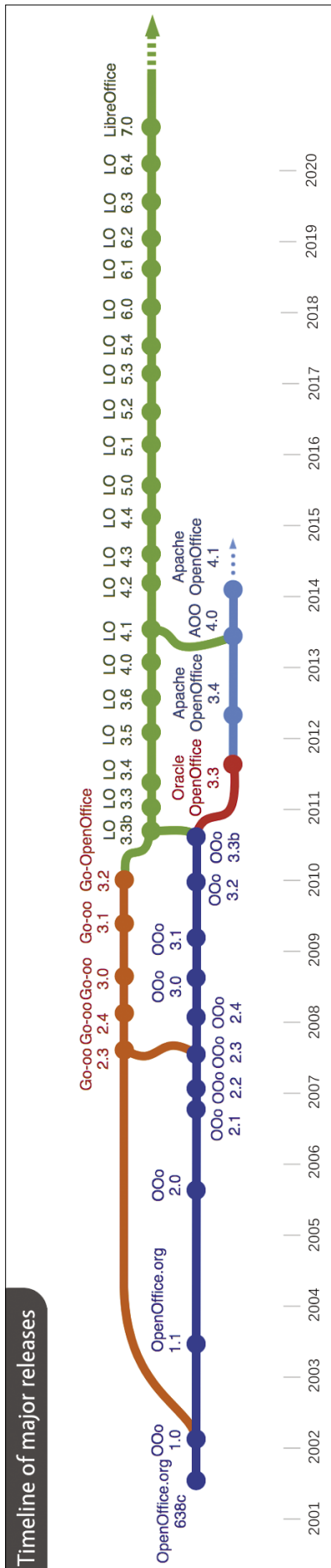


Figure 1: The OpenOffice.org family releases, as of October 2020. Source: The Document Foundation

shared a common codebase. Even today, enough code remains the same that extensions that run with one often run with the other's releases, although less often than a decade ago. No doubt the mutual use of Open Document Format (ODF) for files contributes to continued compatibility. However, the differences do seem to be growing – mostly due to LibreOffice's innovations. The difference can be seen in the download sizes: LibreOffice's DEB download file is 195MB, while OpenOffice's is 159MB. In other words, LibreOffice is close to one quarter larger, despite the fact that it has removed redundancies in the code and rewritten parts for greater efficiency, operations that OpenOffice has yet to duplicate.

About eight years ago, I saw a copy of a LibreOffice internal document that listed all the LibreOffice features that OpenOffice lacked. To my surprise, there were hundreds of line items. Even though many were minor, the differences are far too numerous to list, let alone to explain each one. However, among the more significant features, LibreOffice included an extensive overhaul of the interface, rearranging menu items, repositioning items in windows to make them more accessible, adding themes, and in general creating a more user-friendly look. These changes are especially noticeable in the Charts subsystem for spreadsheets, where the colors were updated from OpenOffice.org's stuck-in-the-1990s look. In addition, LibreOffice introduced themes and a choice of user interfaces that includes an MS Office-like ribbon toolbar. None of these changes have any equivalent in OpenOffice.

LibreOffice's new features are visible in all the modules, especially Writer and Calc. But in the interests of space, I will let Writer stand as an illustration of the others. Outstanding features in LibreOffice Writer that have no counterpart in OpenOffice are:

- The folding of Table Design into the other styles, where it is easy to find and can be copied over into other files.
- Default list styles with choices of bullet and numbering systems, saving users the need to set them up.
- Export to EPUB format (this feature is still being fully implemented, although it is often usable now).

- Support for HarfBuzz, a font-rendering app that allows the modification of OpenType and True Type files through a series of codes added in character fields, making LibreOffice a full-fledged desktop publisher.

However, for many users, the most important improvements in LibreOffice Writer are the continued enhancements for exporting to MS Word. This effort can be a moving target because Microsoft often alters its formats, but reliable import filters to MS Office are essential to many companies' adoption of LibreOffice. Much of this work is due to LibreOffice's strategic alliance with Collabora, which builds compatibility solutions for its customers and donates its improvements to LibreOffice, contributing over a third of LibreOffice's commits in 2019. In the recent 7.3 release, the MS Word import and export filters are at a stage where most routine business and academic documents can be reliably exchanged with those not using LibreOffice. By contrast, OpenOffice's MS Word compatibility is much more erratic.

The Latest Releases

For still another perspective, you have only to look at the latest release notes. On November 27, 2021, OpenOffice announced the release of OpenOffice 4.1.11 [5]. The complete list of improved or enhanced features for this release is:

- Enabled persistent adjustment to help text font size.
- Added Fontwork to Insert menu.
- Added missing PDF export icon to File menu.

In addition, four bugs fixes were thought worthy of mention:

- New security warning introduced with Apache OpenOffice 4.1.10 blocks useful functions.
 - The Windows installation file has missing properties and lacks a code signature.
 - Windows installation: Lost adapted texts after switch to NSIS 3.x.
 - Chart lost when saving in ODS format.
- Furthermore, no new or updated dictionaries or languages were added.

In comparison, the release notes for LibreOffice 7.3 [6], announced February 2, 2022, included so many new features and improvements that a complete list would be too long to include here.

In LibreOffice Writer, some of the improvements included:

- Support for shaped hyperlinks.
- Start of support for linked list, character, and paragraph styles.
- Tracked changes monitors moved text better.
- Faster PDF conversion for complex documents.

In LibreOffice Calc, highlights included:

- Improved speed for opening several formats.
- Improved speed for charts and calculations.

Both Draw and Impress had the following improvements:

- PowerPoint compatible screen sizes improved.
- New surfaces added to 3D objects.

Finally, the following general improvements were made:

- A dialog for barcodes and QR codes.
- Standardized line widths implemented throughout.
- Official binaries optimized for greater performance.
- Enhanced and updated help files.
- Various improvements to MS Office filters.
- Four new languages added, including English (Israel) and Klingon.
- New scripting libraries.

What more needs to be said? Even allowing for the differences in point

releases, the overall picture is clearly one of a faltering project versus a thriving one. The idea that OpenOffice can compete with LibreOffice simply does not hold up.

The Clear Choice

OpenOffice remains an adequate office suite. OpenOffice.org gave it, like LibreOffice, a firm foundation. In particular, OpenOffice's choice of styles gives it a decided advantage over many alternatives. If you only write short or one-off documents, OpenOffice may more than fit your needs. Yet the inescapable fact remains that with no major changes for over a decade, OpenOffice is showing its age. The project is barely keeping up with bug fixes and is not keeping up to date with modern demands the way that LibreOffice is. The only argument I have heard in favor of OpenOffice is that, like Debian's Stable repository, it uses older, more patched code, but given OpenOffice's coding resources, even that argument does not stand up. In fact, LibreOffice is probably better able to fix its bugs. All that OpenOffice has in its favor is name recognition, which only lures users into downloading a vastly stunted alternative to LibreOffice.

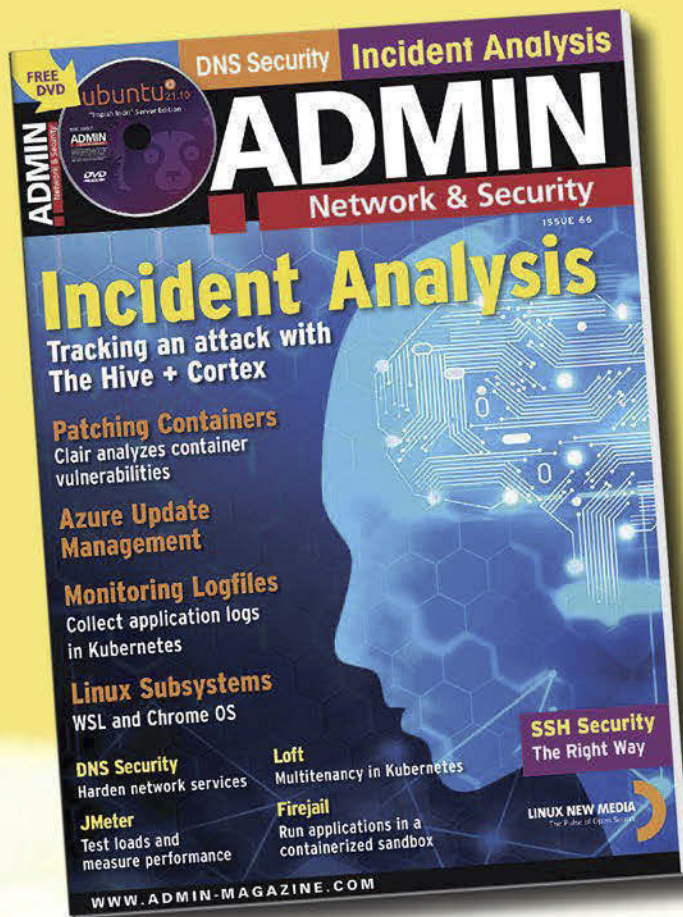
The sensible response to the current situation would be for OpenOffice either to stop limping along or merge with

LibreOffice. In fact, LibreOffice proposed a collaboration in 2020 [7], but stubbornness and perhaps old grievances ensured that it was angrily rejected. Free software means that OpenOffice supporters are free to develop anything they choose, but that does not mean that the rest of us are obliged to support them. When asked to choose between LibreOffice and OpenOffice, the only rational choice is LibreOffice. ■■■

Info

- [1] OpenOffice Contributors 2014-2015: <https://lwn.net/Articles/637735/>
- [2] OpenOffice Contributor Count Graph: www.openoffice.org/stats/committers.html
- [3] LibreOffice Contributors: <https://www.libreoffice.org/about-us/credits/>
- [4] 2019 Code Commits: <https://fossbytes.com/libreoffice-sends-open-letter-appealing-to-apache-openoffice/>
- [5] OpenOffice 4.1.11: <https://cwiki.apache.org/confluence/display/OOOUSERS/AOO+4.1.11+Release+Notes>
- [6] LibreOffice 7.3 Release Notes: <https://wiki.documentfoundation.org/ReleaseNotes/7.3>
- [7] Collaboration Proposal: <https://blog.documentfoundation.org/blog/2020/10/12/open-letter-to-apache-openoffice/%20>

REAL SOLUTIONS *for* REAL NETWORKS

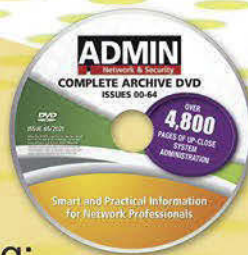


ADMIN is your source
for technical solutions
to real-world problems.

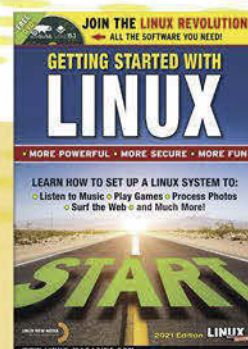
Improve your admin skills
with practical articles on:

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

SUBSCRIBE NOW!
shop.linuxnewmedia.com



Check out
our full catalog:
shop.linuxnewmedia.com





MakerSpace

Flexible HiFiBerry replaces old audio amplifier

Musical Pi

A HiFiBerry HAT, the matching system, and a Raspberry Pi combine with two old speakers to create a contemporary boombox. *By Tim Schürmann*

When my trusty audio amplifier bit the dust with crackling, hissing, and sound dropouts, the speakers and CD player from the hi-fi system still worked perfectly. My idea was to replace the broken module with a Raspberry Pi at a reasonable price. If it could receive music on Bluetooth, then all of my family members could feed it from their smartphones without leaving the couch.

A piggyback board from the HiFiBerry series combined with the matching system on a Raspberry Pi replaced this legacy audio amplifier. The system can play music sent over Bluetooth or other ways, and you can control it conveniently in a browser. However, the build and setup come with a few pitfalls, along with a few minor quirks in operation.

A setup like this is also useful for corporate meeting rooms: The presenter streams the audio output from the Raspberry Pi to the speakers, without wiring or even leaving their seat.

Hi-Fi Berries

Of course, the Raspberry Pi only has one analog audio output, and it is not renowned for particularly good musical quality. To the rescue comes the Swiss company Modul 9, which offers various amplifiers for the small-board computer

(SMB) under the HiFiBerry brand. You simply connect the small boards to the Raspberry Pi GPIO ports and plug the speaker cables into the sockets provided – that's all it takes.

The company also provides its own operating system (OS), HiFiBerryOS, which promises easy hardware startup and acts as a small audio center immediately after launching. Among other things, it accepts signals over Bluetooth and AirPlay or a from connected storage

Tricky Manual Steps

Raspberry Pi OS comes with all the drivers required for the HiFiBerry products. Instead of HiFiBerryOS, then, you could use Raspberry Pi OS and manually install all the required services. However, even setting up Bluetooth could cause you to break a sweat and lose half a weekend. If you still want to try it, you will find instructions online [1].

Although it makes sense to shut down Raspberry Pi OS gracefully, most people simply switch a boombox or hi-fi system on and off with the power switch. For this to work, you need to run the system completely in main memory. However, setting up this kind of configuration is time consuming and complex. Going for HiFiBerryOS will take you to your goal far faster and is easier on your nerves.

Lead Image © Author, 123RF.com

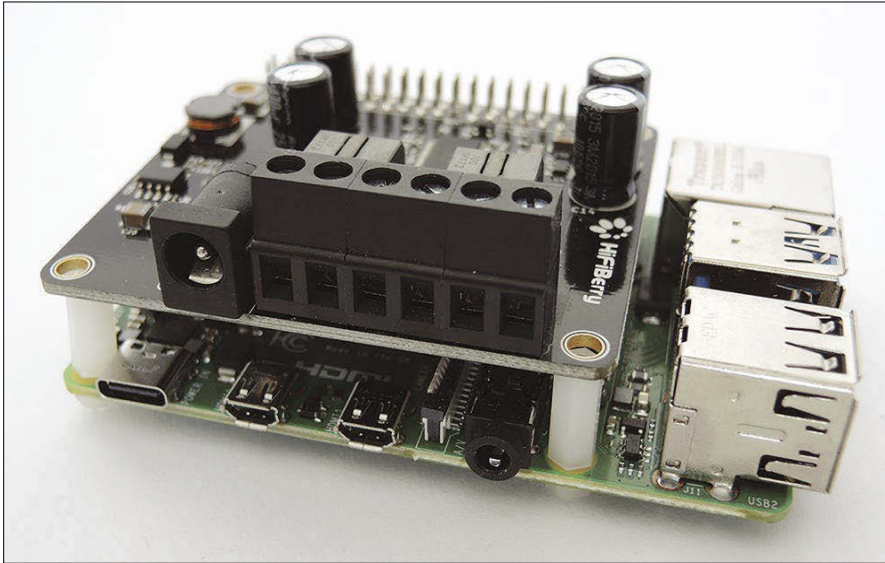


Figure 1: The Amp2 contains a small DAC+ sound module on-board, so all signal processing takes place on the board, without the need for any other hardware.

device, and you can control it in a browser. Under the hood is a customized Linux system with preconfigured services at work. Details can be found in the “Tricky Manual Steps” box.

Hardware Zoo

Modul 9 offers several audio amplifiers for different purposes. The MiniAmp [2] for \$22 (£20/EUR15) is designed for the small Pi Zero and only has two 3W outputs, which, although enough for small PC speakers, is not good enough for a sound system in your living room.

Fortunately, an all-purpose weapon named Amp2 (Figure 1) [3] comes in at \$55 (£54/EUR46). The amplifier has two outputs rated at 30W, which might not sound like much at first, but the small circuit board effortlessly managed to delight even my neighbors on the next floor with hard rock.

Both the MiniAmp and the Amp2 only support two speakers at a time. If you want to connect more, you can get the Beocreate 4 Channel Amplifier for \$190 (£175/EUR160) [4]. As the name suggests, it offers a total of four channels, with two connections even delivering 60W. Additionally, the Beocreate has a digital signal processor with which you can apply various effects to the audio signal in HiFiBerryOS, if required. Unfortunately, the company does not offer a solution for surround sound.

All of these boards come without an audio input, but to continue using an

old CD player, you need an analog sound card with input in the HiFiBerry universe. For this project, I went for the DAC+ analog-to-digital (ADC) converter [5], combined directly with the AAmp60 amplifier [6]. However, unlike the Amp2, it does not have its own sound card, so you will have to piggyback the two HiFiBerry boards on the Raspberry Pi (Figure 2).

The DAC+ ADC in the middle then generates the audio signals, which it passes on to the speakers through the AAmp60 amplifier on top. The pair currently costs \$95 (£110/EUR85). The performance of the AAmp60 is the same as that of the Amp2. Unlike the Amp2, however, the AAmp60 needs the DAC+ ADC or another compatible HiFiBerry sound card.

Other Accessories

The AAmp60, Amp2, and Beocreate each have a connector for an external power supply. Four-ohm speakers need a power supply rated at 12V-18V; for 8-ohm speakers, the developers advise you to go for 18V.

According to the data sheet, the Amp2 can handle a maximum of 24V and the AAmp60 20V. If you want to be on the safe side, purchase the respective board along with the matching power supply in the HiFiBerry store. The Amp2 and Beocreate supply the Raspberry Pi with power; the AAmp60 additionally supplies the DAC+ ADC. Consequently, the Raspberry Pi does not need its own power supply.

To play audio material over Bluetooth, you also need a USB dongle. The exceptions here are the Raspberry Pi 3B+

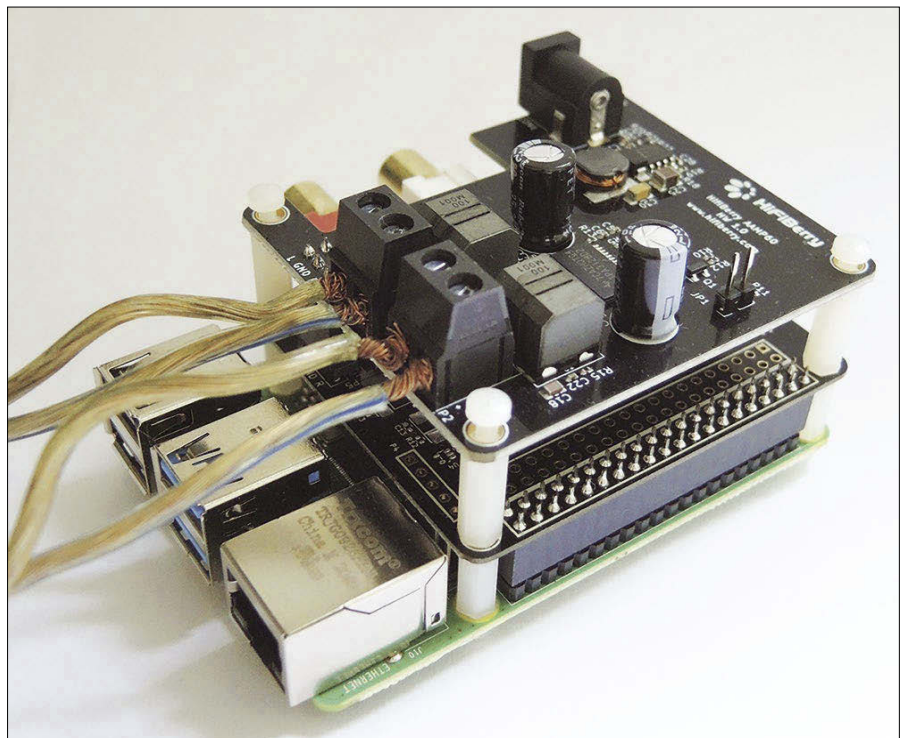


Figure 2: The AAmp60 requires a compatible HiFiBerry sound card. Here, a DAC+ ADC is plugged directly onto the GPIO ports at the bottom, with the AAmp60 sitting on top.

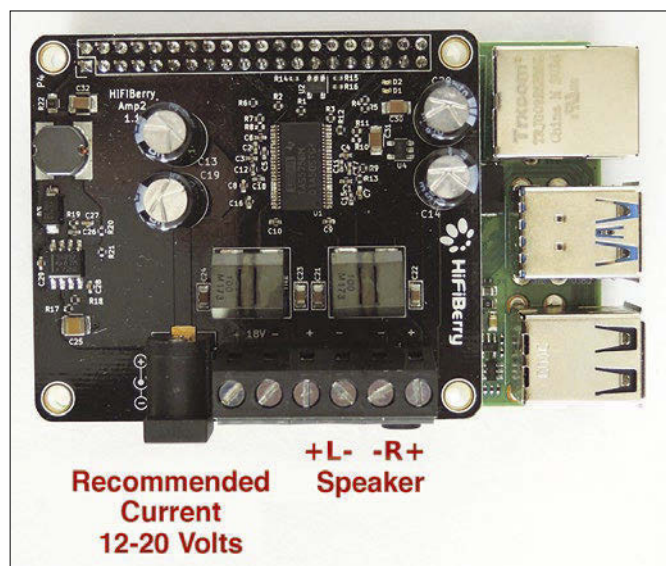


Figure 3: The two jacks on the Amp2 to the left of the connections for the speakers are an alternative power connection; the screw design for power and speakers is similar, so caution is advised.

and 4, which come with the chips in place. The 3B model (without a plus sign) has one, but it does not give you a stable Bluetooth connection, say the HiFiBerry developers, which is why HiFiBerryOS ignores the adapter. When buying a new adapter, make sure it is compatible with the Raspberry Pi and Linux.

The weight of the boards puts some strain on the GPIO pins, so you should first mount the supplied plastic spacers after unpacking (see the “Cases” box). When connecting the cables for the speakers, pay attention to the correct sockets; their assignments differ between the various HiFiBerry models (Figures 3 and 4).

Further peripherals for operation are not needed. However, a monitor, keyboard, and mouse will facilitate the initial setup, which can also be handled in a similarly smooth way in the browser. If you connect the Raspberry Pi to the network by cable, the setup is largely automated; otherwise, you can configure over WiFi, for which you need a smartphone or other device that connects to the WiFi network and has a browser.

Software

The system for your Raspberry Pi model is available on the HiFiBerry page [8]. Unpacking the archive on your hard drive gives you an image of about 1GB,

which you write to an empty microSD card of at least 4GB. The Etcher [9] program, which the HiFiBerryOS developers also recommend, is an intuitive tool for this task. When done, insert the card into the Raspberry Pi and turn it on.

The first launch takes about three minutes, and the SBC reboots once. If it is connected to the local network,

the system automatically retrieves an IP address from the DHCP server (e.g., from a router). Without a network cable, HiFiBerryOS sets up its own WiFi network, to which you connect your smartphone; its name will start with *HiFiBerry_Setup_*.

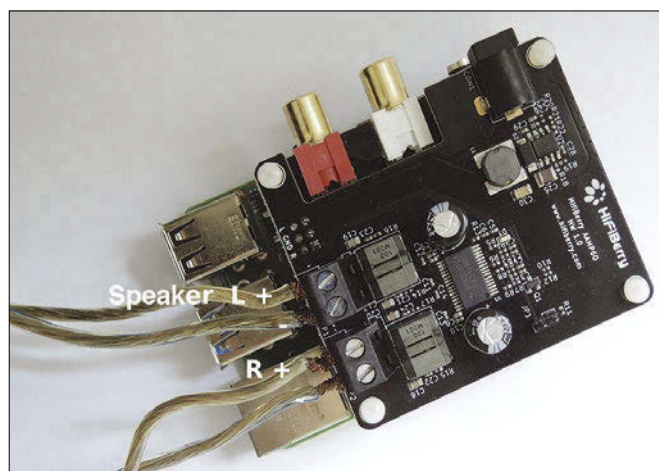


Figure 4: Plug the cables from the left speaker into the two left-hand terminal connections on the AAmp60; the cables from the right speaker are plugged into the next connections to the right.

Cases

Because of the height and the position of the connectors, the amplifiers do not fit into standard Raspberry Pi cases. However, at least for the Amp2, Modul 9 offers a compact steel housing [7], as well as other models, but not all will fit every Raspberry Pi variant.

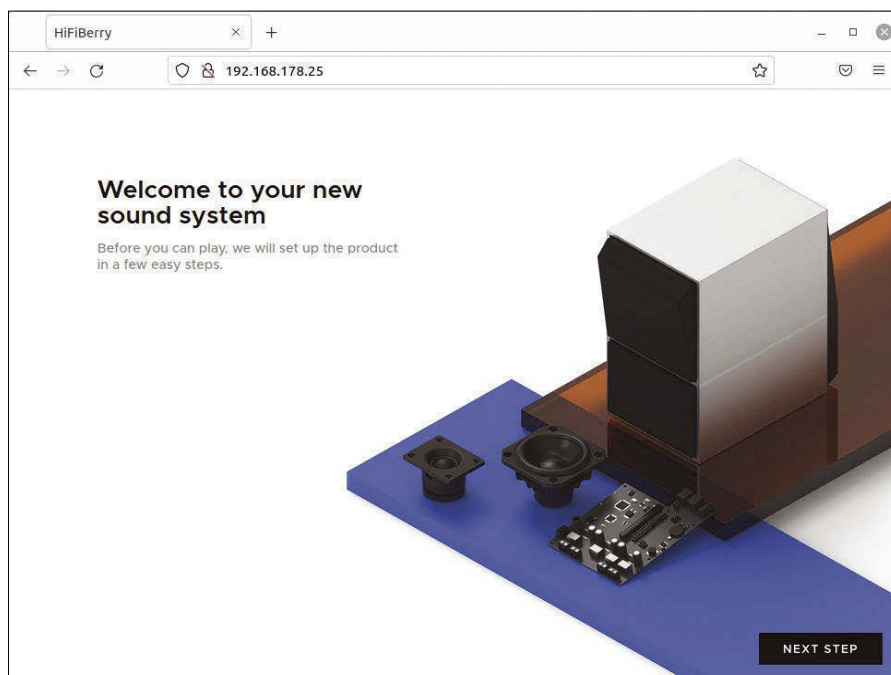


Figure 5: If you see this page, the first setup of HiFiBerryOS has completed correctly.

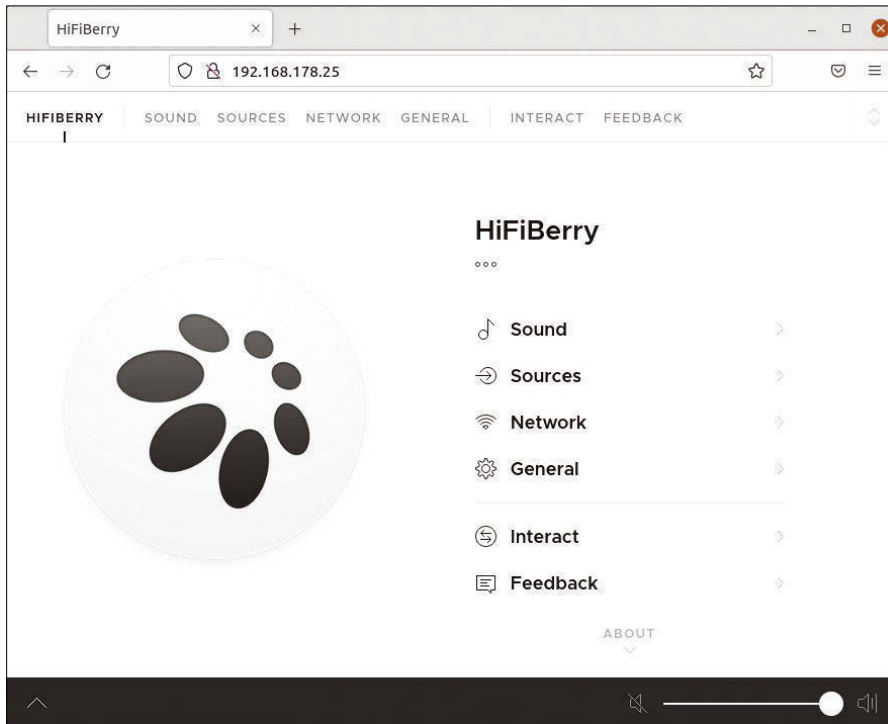


Figure 6: The command center lets you control the playback. Under *Sources* you can change the audio source. The interface automatically adjusts to the screen size.

If you have a connected screen, keyboard, and mouse, you are taken directly to the user interface. Otherwise, call <http://HiFiBerry.local> in the browser. If the connection fails, determine the current IP address of the SBC. If in doubt, the user interface of the router should provide the details. You will find the Raspberry Pi listed there as *hifiberry*. Enter the IP address you determined in this way in the browser. To access the interface with this IP address in the future, you might want to configure your router to always assign the same IP address to the Raspberry Pi.

If you cannot access the user interface after about five minutes, disconnect the computer from the power and then reconnect it. A connected screen might give you clues as to the problem. If the restart attempt fails, make sure the HiFiBerryOS image was written to the medium without errors; the card could be faulty.

Setup

Once the HiFiBerryOS web page (Figure 5) welcomes you, click *Next Step* and then *Finish Setup* until you see the interface (Figure 6).

Unless you want to connect a network cable permanently, go to the

Network menu item, select a WiFi network under the *Wireless* tab, and enter the access password. As soon as HiFiBerryOS establishes the connection, shut down the system in *General* | *Power* | *Shut Down*, wait 20 seconds, and then disconnect it from the power supply. You will not need the network cable from now on.

After a reboot, navigate to the very bottom of the interface. The system should show the board it detected; the Amp2 will be listed as something like *DAC+ /Amp2*. However, a combination of the AAmp60 and DAC+ ADC might not be detected correctly; as a result, the audio input does not appear as an *Analog Input* under *Sources*.

In this case, shut down HiFiBerryOS again as before. Open the smallest partition on the SD card on your PC.

Under Windows you can access the required files directly. In a text editor, add the following line at the end of the *config.txt* file:

```
dtoverlay=hifiberry-dacplusadc
```

Restart the Raspberry Pi from the SD card and make sure in the web interface that the HiFiBerry has identified the system correctly.

Bluetooth

Once the system is running, switch to *Sources* (i.e., audio sources). *AirPlay* is enabled by default, and smartphones with support for this and other media players should identify the computer as storage and be able to play back music through it.

For Bluetooth access, select *Bluetooth* and then click the black *Bluetooth Pairing* button. After that, all devices in the vicinity with Bluetooth enabled will recognize the *HiFiBerry* as a speaker. Pair it with your smartphones, tablets, and other devices – the procedure will differ in each case (Figure 7) – then stop pairing on HiFiBerryOS.

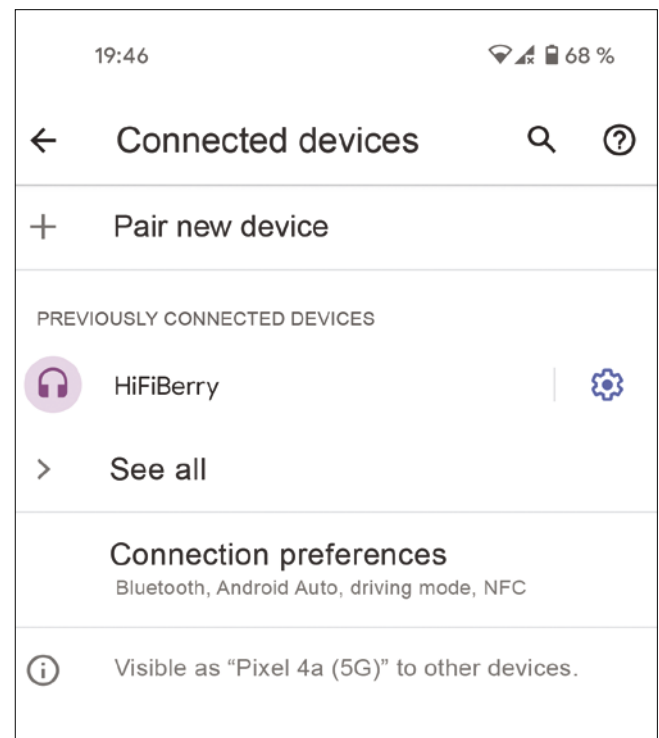


Figure 7: To introduce Android 11 to HiFiBerryOS, select *Connected devices* and *Pair new device* in the settings.

Now you can use these devices to control the volume. To avoid getting on your neighbors' nerves, you might want to turn the volume down when playing back the first time. During playback, HiFiBerryOS displays the title at the bottom edge, and to the right, you will find the volume control.

To play back music from the CD player, switch to *Sources | Analog Input*. This item may be hidden under *Disabled Sources* (Figure 8). Turn on the input, which will simultaneously turn off the other sources. If you later want the Bluetooth players to provide the sound, disable the audio input and enable the corresponding items under *Sources | Bluetooth*.

Tricky Stuff

Your Raspberry Pi is now working as a small audio amplifier. In practice, however, you will probably find yourself continually tripping over shortcomings. For example, the Amp2 had massive problems with volume control in a first installation in the test. As soon as the volume exceeded or fell below certain thresholds, audible dropouts occurred in the playback. With the current version of the system in a new installation, everything worked flawlessly. The AAmp60 did not exhibit any of these problems.

In practice, new Bluetooth devices are likely to enter your household from time to time (e.g., if you have visitors). To grant them access to the amplifier, you need to enable the pairing feature in each case. Turning pairing on permanently is not a good idea – or do you really want your neighbor to play back their favorite opera on your speakers?

The user interface also lacks password protection and other security measures. Users on the network can therefore reconfigure the Raspberry Pi or play music without restrictions.

To switch to the CD player, you have to enable the appropriate audio input from the user interface. At the time of writing, HiFiBerryOS does not switch over automatically as soon as a signal is present. Unlike a standalone amplifier, listening pleasure requires a few clicks up front, which, in my experience, is too much for many users.

Shutdown

The intended way to shut down HiFiBerryOS after use is by selecting *General | Power | Shut Down*. In practice, however, people often simply unplug the power cord or turn off the powerstrip for devices of this type – especially users who find the user interface too complex. In the test, cutting the power did not cause any permanent damage, and the

Raspberry Pi repeatedly booted without any problems.

The developers of the HiFiBerryOS also point out in a blog post that the system does not write very much data to the SD card and that damage is very rare [10]. Just in case, I keep a second SD card with a copy of the system next to my DIY boombox.

Modul 9 now also offers a power controller, still in the testing or beta phase [11], that lets you shut down the Raspberry Pi in a controlled manner at the push of a button.

Conclusions

HiFiBerry amplifiers quickly transform a Raspberry Pi into an oversized boombox, allowing you to extend the service life of your legacy hi-fi components at a low cost. HiFiBerryOS also removes the need to set up Bluetooth and other complex services manually.

The system also offers access to numerous services, including Spotify and various internet radio stations. However, minor quirks make the use of multiple input sources difficult to control. ■■■

Info

- [1] "Configuring Bluetooth on Linux" by Bruce Byfield, <https://www.linux-magazine.com/Online/Features/Configuring-Bluetooth-on-Linux/>
- [2] MiniAmp: <https://www.hifiberry.com/shop/boards/miniamp/>
- [3] Amp2: <https://www.hifiberry.com/shop/boards/hifiberry-amp2/>
- [4] Beocreate 4 channel amplifier: <https://www.hifiberry.com/shop/boards/beocreate-4-channel-amplifier/>
- [5] DAC+ ADC: <https://www.hifiberry.com/shop/boards/hifiberry-dac-adc/>
- [6] AAmp60: <https://www.hifiberry.com/shop/boards/hifiberry-aamp60/>
- [7] Case for Amp2: <https://www.hifiberry.com/shop/cases/steel-case-for-hifiberry-dac-pi-4-2-2/>
- [8] HiFiBerryOS: <https://www.hifiberry.com/hifiberryos/>
- [9] Etcher: <https://www.balena.io/etcher/>
- [10] "Powering up/down your Pi with a button": <https://www.hifiberry.com/blog/powering-up-down-your-pi-with-a-button/>
- [11] HiFiBerry power controller: <https://www.hifiberry.com/shop/boards/power-controller/>

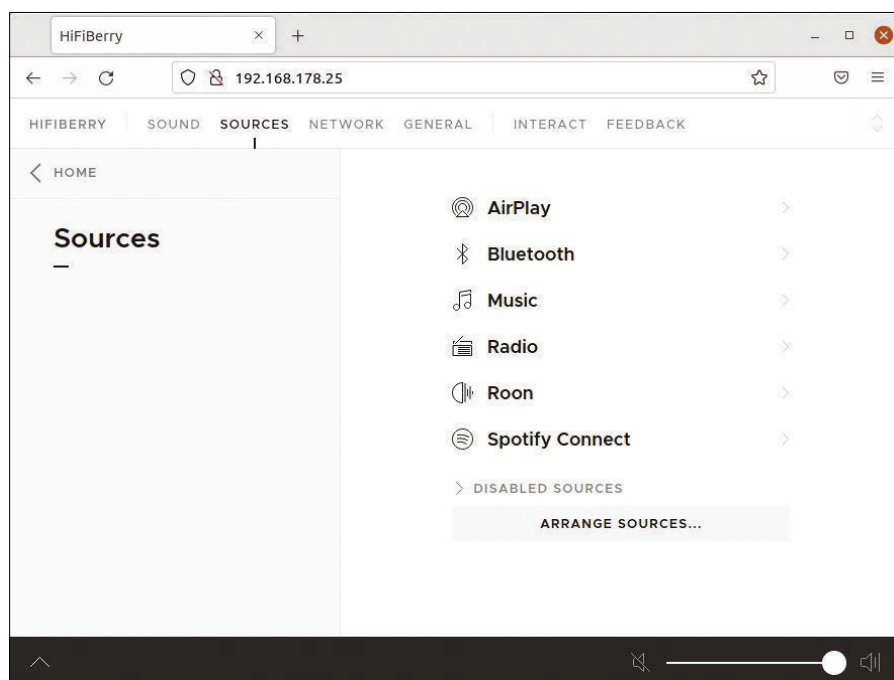
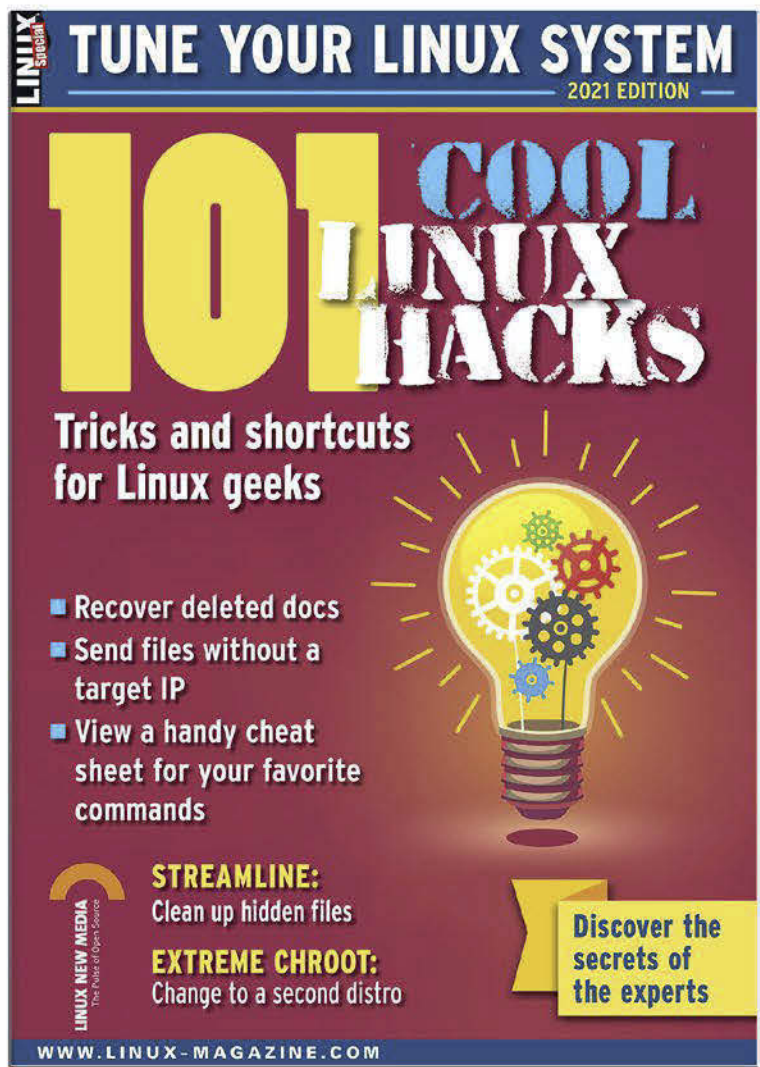


Figure 8: HiFiBerryOS not only plays back audio material via Bluetooth and AirPlay, but also from Spotify, radio stations, and other sources. You have to enable some settings first, though (e.g., for CD players).

SHOP THE SHOP
shop.linuxnewmedia.com

GET PRODUCTIVE WITH 101 LINUX HACKS



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

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

ORDER ONLINE:
shop.linuxnewmedia.com/specials



MakerSpace

DMX with the Kunbus Revolution Pi
Core 3+

Party Lights

The DMX protocol for controlling lighting technology has its roots in stage and event technology, but it can also be used in home automation. We show you how to control DMX devices with the Revolution Pi module. *By Martin Mohr*

In principle, digital multiplex (DMX) technology lets you control all DMX devices through a simple and robust bus. DMX uses the RS-485 specification (i.e., asynchronous serial transfer) for data transmission. The signal is transmitted over two lines simultaneously with inverted voltage levels ($\pm 5V$).

This type of data transfer is resistant to external interference signals and theoretically allows line lengths of up to 1,200 meters. However, the data transmission rate decreases with increasing line length. Up to a distance of 10 meters, a maximum data rate of 10Mbps can be achieved over RS-485.

DMX works with a data rate of 250Kbps by default; a DMX network is built of strings. One string can contain up to 32 DMX receivers. More receivers require the use of a repeater to extend the network. Network branches are implemented with special splitters. Extremely robust, three-pole XLR (Canon) connectors are used to connect all DMX devices, which is ideally suited for stage use.

The DMX512 standard supports up to 512 device addresses. Depending on how many addresses a component needs, the number of devices that can be operated on a bus varies.

DMX Lighting

In this article, I use two simple DMX lights that I ordered directly from China [1] some time ago for EUR6.58 each. However, ordering directly from China now might not be worth the effort and prices have increased, so find out in advance what additional costs you will have to pay.

The lamps have eight channels each, which means they occupy eight addresses on the DMX bus. If you assign address 1 to the first lamp, the second lamp is assigned address 9. Somewhat confusingly, addresses and channels are talked about in the DMX environment as if they were different things. The rule of thumb is that you should have an address for each channel in the light. Table 1 shows an overview of the channels for the hardware I used.

Most of the functions the DMX lamps offer are self-explanatory. Only a few items require special attention. For the spot to illuminate, you need a value greater than 0 on channel 1. You can't turn up just the red dimmer while the general dimmer is set to 0.

A value of 0 disables the strobe (channel 6). At a value of 1, it flashes at maximum rate and slows with an increase in value. Various predefined color scenarios can be selected in Auto Run operational

Photo by Sam Moqadam on Unsplash

Table 1: DMX Light Channels

Channel	Function	Value Range
1	Dimmer (all colors)	0-255
2	Red dimmer	0-255
3	Green dimmer	0-255
4	Blue dimmer	0-255
5	White dimmer	0-255
6	Stroboscope	0-255
7	Fade in/out	0-250
	Sound mode	251-255
8	Function speeds selected on channel 7	0-255

mode; the playback speed is controlled by channel 8.

At this point, the manufacturer's documentation is a bit vague; it is not clear what exactly is supposed to happen. However, this is not a problem because an external controller is being used.

The lamps have one DMX input and output each and have a power consumption of 36W, so the power can be supplied by a standard socket. The addresses of the lights can be set quite intuitively with a four-digit, seven-segment display on the back. Just make sure the display shows an *A* in the first position. Assign address 1 to the first lamp and address 9 to the second. If the address settings do not work at first, take a look in the enclosed manufacturer's manual. Figure 1 shows the two lamps with the wiring and the Revolution Pi I used as a controller.

Revolution Pi

The Revolution Pi Core 3+ from Kunbus (hereafter, RevPi Core) [2] is a small industrial controller with a Raspberry Pi Compute Module at its heart. The RevPi Core can be extended with different modules and adapted to different requirements. The manufacturer's extensive treasure trove of modules also contains a module the RevPi Core can use to connect to the DMX bus.

The DMX module can operate in both primary and worker mode and control up to 512 DMX addresses. The supply voltage is 24V DC with a current consumption during use of 100mA. The module indicates the current operating status with three LEDs. The exact meaning of the displays is shown in Table 2.

An eight-pin plug on the front side of the module lets you establish a connection to the DMX bus. The pin

Table 2: Operating States

LED Name	Signal	Meaning
Power	Off	Gateway inactive
Power	Flashing (green)	Initializing
Power	On (green)	System working and OK
Power	Flashing (red)	Fixable error
Power	On (red)	Serious error/gateway failure
Term	On (green)	Bus termination active
Term	Off	Bus termination inactive
Traffic	Flashing (green)	DMX data active (primary and worker mode)
Traffic	On (red)	DMX data inactive (worker mode)

assignment of the connector is shown in Table 3. You will also find it – and the pin assignment for the three-pin XLR connector – on the side of the module. If you need more information about the RevPi DMX module, it is best to take a look at the user manual [3].

Configuration and Test

The DMX module can be configured in two ways: either from the switches on the front of the module or by setting the corresponding values in software. Here, I look at how to configure the module in software. To begin, log in to the RevPi Core (Figure 2). The user is *admin*, and the password can be found on the right side of the device's housing.

After successfully logging in, start the PiCtory app (Figure 3) and drag and drop the RevPi Core and DMX module to the current configuration. The DMX module is configured by 2 bytes. An overview of the supported parameters can be found in Table 4. The address and channel settings can differ slightly depending on the mode (transmit/receive). If necessary, consult the manual [3].

In the experimental setup, the first configuration byte under *Value Editor* (Config_1) is set to 0x10 and the

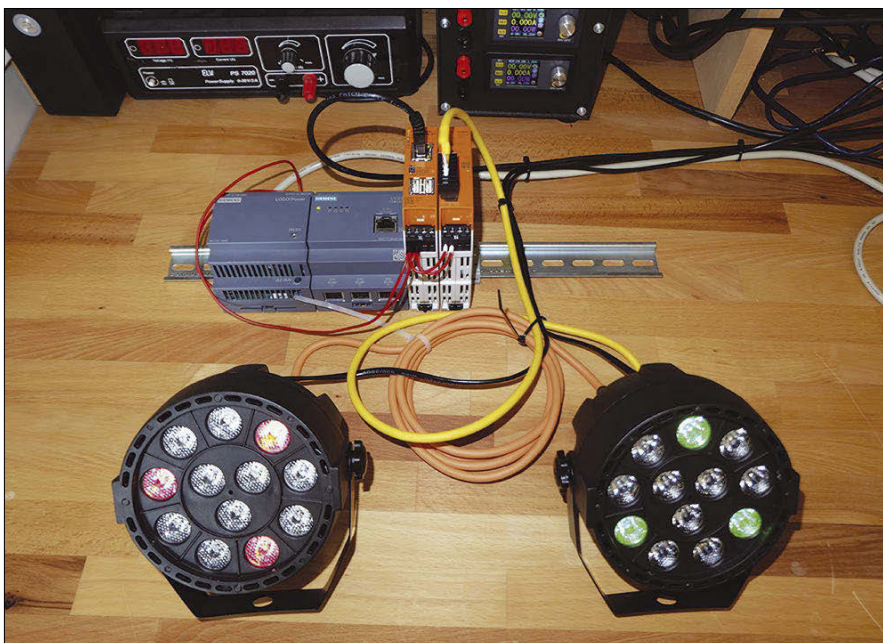


Figure 1: The test setup with two DMX lights and the Revolution Pi as the controller is fairly simple to emulate.

Table 3: Pin Assignments

Module Pin	Function	XLR Connector Pin
1	None	–
2	None	–
3	None	–
4	Earth (shield)	1
5	DMX +	3
6	DMX –	2
7	None	–
8	None	–

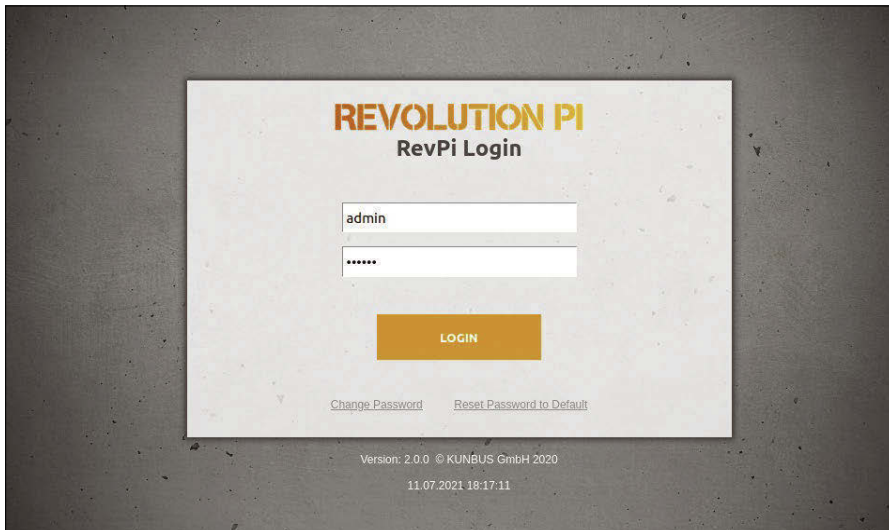


Figure 2: RevPi Core login screen.

second (Config_2) to 0x80 (Figure 4). To access the individual bytes of the DMX module in programs, they need to be published by checking the *Export* column. For the test setup you need to set the first 16 bytes (DMXOut_1 to DMXOut_16).

Figure 4 shows the complete configuration of the RevPi Core, including the DMX module. To keep the settings after a restart, save the configuration by selecting *File | Save as Start-Config*. Now reboot the RevPi Core to check that the settings have all taken effect.

After the reboot, connect to the RevPi Core console over SSH. On Linux use the ssh command-line tool (Windows users

can install PuTTY [4]). The username for the login is *pi*, the password is the same as for logging in from a browser.

The piTest program briefly tests the setup; it gives you read and write access to all of the RevPi Core's interfaces. If you launch the tool without any parameters, it provides a listing of all your options. Listing 1 gives you the commands you need to make the first DMX light show red and the second one green.

The first two commands fully fade up the master dimmers of the two lights; the following two commands do the same for the colors. Although values between 0 and 255 can be written to the channels, the lighting intensity only changes every 16 values, although this is not generally the case with DMX, just with these cheap lights.

The piTest program outputs the offset in the process image of the RevPi Core every time. The process image represents all of the device's interfaces. You can

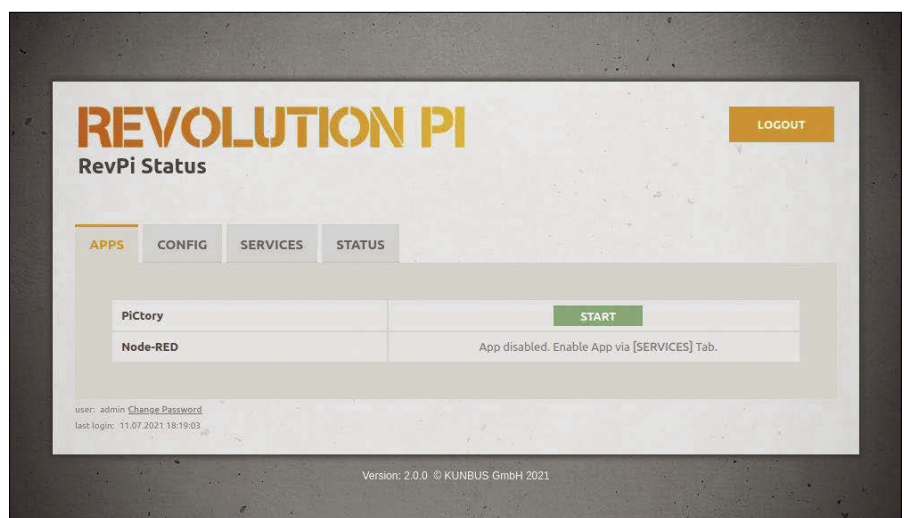


Figure 3: Launch the PiCtory app by pressing the matching button.

Table 4: Configuration Bits

Byte	Bit	Function
0	0	Address/channel bit 0
	1	Address/channel bit 1
	2	Address/channel bit 2
	3	Address/channel bit 3
	4	Address/channel bit 4
	5	Address/channel bit 5
	6	Address/channel bit 6
1	7	Address/channel bit 7
	0	Address/channel bit 8
	1	Address/channel bit 9
	2	Not used
	3	Not used
	4	Not used
	5	Cancel behavior (0 = keep data, 1 = delete data)
	6	Mode (0 = transmitter, 1 = receiver)
	7	Termination (0 = inactive, 1 = active)



Figure 4: Finished configuration for the DMX module.

Listing 1: Testing DMX Lights

```
$ piTest -w DMXOut_1,255
Write value 255 dez (=ff hex) to offset 525.
$ piTest -w DMXOut_9,255
Write value 255 dez (=ff hex) to offset 533.
$ piTest -w DMXOut_2,255
Write value 255 dez (=ff hex) to offset 526.
$ piTest -w DMXOut_11,255
Write value 255 dez (=ff hex) to offset 535.
```

Listing 2: Switching Off the Lights

```
$ piTest -w 525,1,0
Write value 0 hex (=0 dez) to offset 525.
$ piTest -w 526,1,0
Write value 0 hex (=0 dez) to offset 526.
$ piTest -w 533,1,0
Write value 0 hex (=0 dez) to offset 533.
$ piTest -w 535,1,0
Write value 0 hex (=0 dez) to offset 535.
```

Listing 3: dmx_demo.py

```
01 #!/usr/bin/python3
02 import time
03 import struct
04
05 DEBUG = False
06
07 DMX1_MASTER = 525
08 DMX1_RED = 526
09 DMX1_GREEN = 527
10 DMX1_BLUE = 528
11
12 DMX2_MASTER = 533
13 DMX2_RED = 534
14 DMX2_GREEN = 535
15 DMX2_BLUE = 536
16
17 class RevPi:
18     _rev_pi = open('/dev/piControl0', 'wb+', 0)
19
20     def __init__(self):
21         pass
22
23 class DMX(RevPi):
24     _green = 0
25     _red = 255
26     _blue = 0
27
28     def __init__(self, master, adr_green, adr_red, adr_blue):
29         self.adr_green = adr_green
30         self.adr_red = adr_red
31         self.adr_blue = adr_blue
32         self._rev_pi.seek(master)
33         self._rev_pi.write(struct.pack('<H', 255))
34
35     def update_color(self, adr, val):
36         self._rev_pi.seek(adr)
37         self._rev_pi.write(struct.pack('<H', val))
38
39     def set_step(self, mode_val):
40         if mode_val == 1:
41             self._green += 16
42         elif mode_val == 2:
43             self._red += -16
44         elif mode_val == 3:
45             self._blue += 16
46         elif mode_val == 4:
47             self._green += -16
48         elif mode_val == 5:
49             self._red += 16
50         elif mode_val == 6:
51             self._blue += -16
52
53     def do_green(self):
54         if DEBUG:
55             print('GREEN: %d' % self._green)
56         self.update_color(self.adr_green, self._green)
57
58     def do_red(self):
59         if DEBUG:
60             print('RED: %d' % self._red)
61         self.update_color(self.adr_red, self._red)
62
63     def do_blue(self):
64         if DEBUG:
65             print('BLUE: %d' % self._blue)
66         self.update_color(self.adr_blue, self._blue)
67
68 DMX_1 = DMX(DMX1_MASTER, DMX1_GREEN, DMX1_RED, DMX1_BLUE)
69 DMX_2 = DMX(DMX2_MASTER, DMX2_GREEN, DMX2_RED, DMX2_BLUE)
70
71 while 1:
72     for mode in range(1, 7):
73         if DEBUG:
74             print('mode=%d' % mode)
75         for i in range(0, 15):
76             if DEBUG:
77                 print('i=%d' % i)
78             DMX_1.set_step(mode)
79             DMX_2.set_step(mode)
80             DMX_1.do_red()
81             DMX_2.do_red()
82             DMX_1.do_green()
83             DMX_2.do_green()
84             DMX_1.do_blue()
85             DMX_2.do_blue()
86             time.sleep(0.05)
```


also use the offset to access the DMX addresses. Listing 2 uses this ability to switch off the lamps again. In principle, you could use a little shell script magic to control the lights automatically, but because shell programming is not everyone's cup of tea, I'll look at how to control the DMX lamps with a Python script in the next section.

Program

The Python script (Listing 3) first loads all the required libraries. The block that follows creates some variables and constants for the offset. As seen in the previous section, values on the RevPi Core can be accessed by their plain text names and by offsets.

With the help of the `fcntl` library, the script could be designed in such a way that it determines the offset in the background and uses it instead of the plain text name. The script here avoids this complex procedure by using the offsets directly. Because these values always increase by 1 in the example, you do not have to determine all values by trial and error. A code example demonstrating the use of the plain text names can be found on the Kunbus homepage [5].

Two classes (lines 17 and 23) establish the connection to the RevPi (RevPi) and control color changes of the lights (DMX). The `open()` command lets you open a connection to the hardware driver and store it in the `_rev_Pi` variable. All access to the hardware will then be through this variable.

In the DMX class, the `do_color` methods use the `update_color()` statement to ensure that the appropriate color is sent

to the light. The color is calculated by the `set_step()` method and controlled by the `mode_val` variable. Because the lamps can only handle fairly coarse dimming, the values for each color are always increased or decreased by 16.

Two lights and three colors allow six possible color transitions, which is why the `mode` variable is reset to 1 as soon as it reaches a value of 7 (line 72). The `time.sleep(0.05)` command controls the speed of the color transitions. The `__init__` method of the DMX class sets the master dimmer to 100 percent, to make full use of the dimmers for each color.

The `while 1` command tells the program to run in an infinite loop, which you can cancel with the Ctrl + C keyboard combination. Inside this loop, the code changes the variable `mode` and passes it to the objects for the two lights, turning a particular color on or off. When a color reaches its new value, the `mode` variable is incremented by 1. To cycle through a color transition fully, you need to call the `set_step()` method 16 times, as in the second for loop (line 75).

You can launch the demo program with the command

```
python3 dmx_demo.py
```

at the RevPi Core's command line. You can find a video on YouTube showing the demo program in action [6].

Conclusions

The DMX module for the Revolution Pi Core 3+ makes creating your own

lighting show easy. A little browsing on the Internet reveals many interesting devices that can be controlled by DMX. Besides classic stage lighting, the products also cover lighting systems for home automation. One genuine highlight among the DMX devices is a moving, talking skull [7]. Have fun with your own DMX experiments. ■■■

Info

- [1] DMX lights:
<https://www.aliexpress.com/item/1005002449259723.html?gatewayAdapt=glo2deu>
- [2] RevPi Core 3+:
<https://revolutionpi.com/revpi-core/>
- [3] DMX module user manual:
<https://www.kunbus.com/files/media/bedienungsanleitungen/DO0262R00-MGW-UM-DMX-EN.pdf>
- [4] PuTTY: <https://www.putty.org>
- [5] Python example:
<https://revolution.kunbus.com/download/1141/>
- [6] Video for the program:
<https://youtu.be/2i2Httbwc1E>
- [7] Talking skull:
<https://www.hauntedhousecreations.com/full-motion-talking-skull/>

Author

Martin Mohr has had a fondness for everything that flashes since his early youth, reinforced by an apprenticeship as an electronic technician. After studying computer science, he has mainly developed Java applications, but the Raspberry Pi rekindled his old love of electronics.

It would be an understatement to say that Linux and NVIDIA have had an uneasy relationship. The massive NVIDIA, famous for graphics cards and other computer hardware components, has a long history of recalcitrance when it comes to Linux compatibility. Things are a little better than they were – at least some Linux drivers exist today – but most of those drivers are proprietary, and the company doesn't invest much in showing Linux users how to work with their hardware. In this month's Linux voice, we'll help you swim through the NVIDIA driver soup using tools and resources available through Ubuntu. Also inside: You'll take your first steps with Matplotlib, an open source Python library for graphing and other data visualization tasks.



Image © Olexandr Moroz, 123RF.com

LINUXVOICE ►

Doghouse – IoT Need for Openness 75

Jon "maddog" Hall

Closed IoT devices can use unexpected bandwidth "reporting home," pointing to a need for free devices to allow the user more control over their household gadgets and WiFi use.

NVIDIA Drivers in Linux 76

Adam Dix

A terminal-based solution helps ease the frustration of installing NVIDIA drivers.

FOSSPicks 80

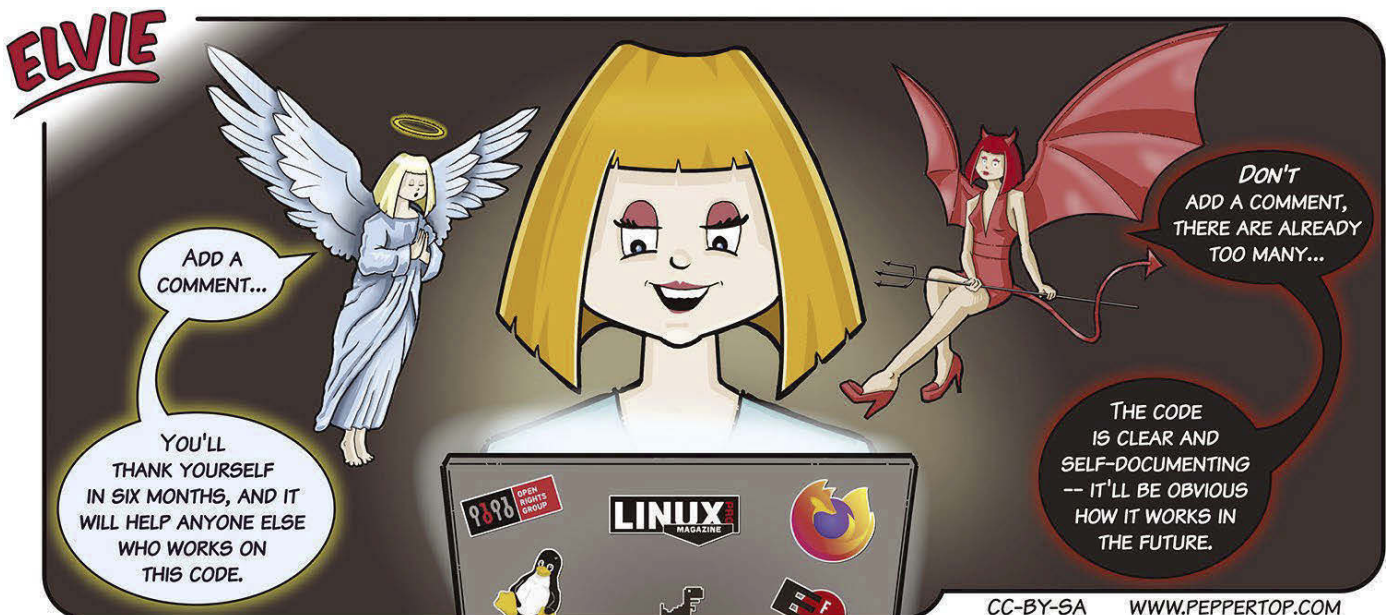
Graham Morrison

This month Graham looks at Blender 3.0, woob, Fokus, bat, GNU poke, Bladecoder Adventure Engine, Crispy Doom, and more!

Tutorial – Matplotlib 88

Marco Fioretti

Build illuminating graphs into your Python programs.



CC-BY-SA WWW.PEPPERTOP.COM

Public Money

Public Code



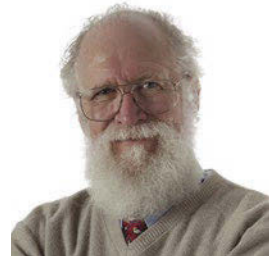
Modernising Public Infrastructure with Free Software



Free Software Foundation Europe

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

MADDOG'S DOGHOUSE



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

Closed IoT devices can use unexpected bandwidth "reporting home," pointing to a need for free devices to allow the user more control over their household gadgets and WiFi use.

BY JON "MADDOG" HALL

Freeing home IoT and WiFi

Recently, I upgraded my home Internet to fiber and 1Gbps symmetric service, so I theoretically get 1Gbps communication up and down from my provider.

For most people, that translates to a thought pattern of "I can move data at 1Gbps from my laptop/desktop/phone" which can be fairly far from the truth.

Because of overhead in communications, you can look for between five and 10 percent loss in your throughput, even in the best of all possible situations. Therefore you might see 995Mbps to 990Mbps as actual data throughput, even with a Cat 6 cable of four feet hard wired to the LAN ports of your WiFi router.

Recognizing that WiFi would probably be slower, and likewise, the server at the other end would help determine data speed, my partner and I used the fiber provider's speed test. We expected that it used a server (or generated the bits at their end) for the fastest test possible.

My partner and I measured 990Mbps as download speed through the router, but the upload speed was considerably worse, coming in at only 500Mbps, and that caused a temporary puzzle.

Fortunately, my provider offers an interesting application that keeps track of the bit traffic from every single hardware address associated with each device. Therefore I can see how many bytes are being uploaded and downloaded from each device – each day, each week, and each month.

From this, we could see that some devices were uploading, doing backups of data that were scheduled for a late-night time.

We could also see how the data was divided up among other sources that many people might not consider. In our case, it was a light bulb.

We use a lot of WiFi devices to control our home. WiFi-controlled speakers, Chromecast for video broadcasting, etc., but these were "off" at the time we were testing the speed. What we did not anticipate was the fact that a WiFi-controlled light bulb, turned on and off typically once in a 24-hour period, might download 14.41MB of data and upload 5.93MB of data each day. Wow. I might understand the 14.41MB down for a firmware update ... but every day (on the average)? And why close to 6MB

uploaded just to do what? Report back to some entity that the bulb had actually provided the light in the room ... or not?

We have only one light bulb like this in our home, but we do have a series of WiFi electric outlets. Each one had similar upload and download traffic.

Our Internet provider noted that "Homes that have many of these types of devices (many being in the dozens of units) may experience less than optimum download and upload speeds." Duh.

All of this is affecting the speeds that the Internet user sees. This does not take into account the sharing of WiFi inside the house by all these packets that are simply handled by the router itself but that take away bandwidth from other WiFi uses.

All of this melded into a discussion among some friends of mine about the need for symmetrical Internet speeds, Internet of Things (IoT) discussions, and how these devices are using both electricity and Internet even when they are "off."

A lot of these devices are sending *all* of their communications back to the company that created them. It would have been better if these devices had a centralized "server" unit that could process a lot of this data, called "edge programming," boiling down the raw data and distributing firmware updates to all the devices using the same Internet sub-network.

It would be great if these devices also used different frequencies (perhaps the emerging 60GHz WiFi), or perhaps I need to spend more time setting up quality of service (QoS) for the router and Internet. After all, if it takes a fraction of a second more to turn a light on or off it probably will not make much difference.

One friend of mine said that they isolate all of these devices on a separate network, blocking them from "calling home." If the devices stop working, they get rid of the devices. Of course, they lose both the money paid for them as well as the service of the device.

Despite being a free and open source advocate for many years, I have so far taken a laid back position on these IoT devices, but in the next year I am going to refresh my home of 40 years, spending a considerable amount of money, and part of that will be to only purchase "free" IoT devices. This could also include purchases of significant appliances.

It is too bad that manufacturers of closed devices will not be able to receive my money, and the money of many more people who feel the same way. ■■■

Simplifying NVIDIA driver installation on Ubuntu

NVIDIA Drivers Made Easy

A terminal-based solution helps ease the frustration of installing NVIDIA drivers.

BY ADAM DIX

Proprietary NVIDIA [1] drivers on Linux are an evil so necessary that even the staunchest of open source advocates will find a use case for them from time to time. Whether for gaming, mining, or some other hardware-accelerated task, the green giant is here to stay, for better or for worse, and sadly seems to be wholeheartedly opposed to going all in with open source. Unlike AMD [2] driver support – which is simply fantastic on Linux, as I am sure most of you know already – NVIDIA doesn't seem to be willing to play nicely. One has to wonder if they feel that they simply don't need to, that their sway over the market sets them apart from others and absolves them of the responsibility that the other players feel to their customers. Who knows. What I do know is that NVIDIA Linux driver support is worse for the wear despite the hard work of many folks in the community. The worst part about this is the effect that it has on beginners to Linux who have heard all of the wonderful things about it and give it a try, only to have their system either hobbled with Nouveau drivers or who have to go through what I am about to discuss below. It is a sad state with a clear and obvious solution that I am afraid we won't see come to fruition any time soon.

All may not be well, but not all is bad either. At least we have some kind of (albeit proprietary) solution to rely upon.

I will focus on Ubuntu [3] here as that is what I typically use nowadays as a simple, Linux-based install for whatever I need to do, but also because this is statistically likely to be what a beginner would first try out. While I started with Red Hat [4]

(pre-Fedora [5]), today most folks tend towards Ubuntu, a derivative of it, or perhaps another Debian derivative their first time out.

While Ubuntu itself has a driver's install section, not all Ubuntu derivatives necessarily do, and in some cases, even if your distribution does have an advanced driver install section, it may be difficult to find, especially for a beginner. Furthermore, there may be confusion about which group of packages to install and which version of the NVIDIA proprietary driver you want to use. I have personally also found that often after choosing a driver and clicking on *Apply*, the window becomes unresponsive, or at least appears that way. While the basic open source Nouveau [6] Linux driver is adequate for most tasks, hard-core users will often find a need for proprietary drivers. While I can't fix the terribly dated control panel interface that NVIDIA offers in their proprietary driver package installation, I can help you get it installed and walk you through how to deal with it when that install inevitably breaks.

In this article I'll show you how to figure out which graphics drivers to use on Ubuntu Linux and derivatives with a simple, built-in command which I find quicker and easier than opening up the GUI equivalent. I will also show you how to deal with the near certainty of the black screen that you'll encounter afterwards. You see, much like a YouTuber trying to install Steam [7] on Pop!_OS [8] for the first time, I too have had struggles with using the GUI for installing NVIDIA drivers. However, unlike the typical YouTuber [9], I have not had any issues doing the same task in the terminal.

It should be noted that more than one package may exist for NVIDIA proprietary drivers, and you may need to test a few different driver packages before you figure out which one works best for you. Furthermore, devices with Optimus [10] capability may or may not work properly in terms of Optimus switching. For those who don't know, don't recall, or weren't around in those days, Optimus is an NVIDIA and Intel [11] software scheme intended to allow the user to choose between NVIDIA or Intel graphics at will within the OS itself. While packages from the likes of system76 and others do make Optimus more capable than it was when it was first introduced, and in many cases fully functional, there still

The Author

Adam Dix is a mechanical engineer and Linux enthusiast posing as an English teacher after playing around a bit in sales and marketing. You can check out some of his Linux work at the EdUBudgie Linux website, <https://www.edubudgie.com>.



tend to be bugs in my experience. Don't let that deter you, though, from trying some of the different packages out there for your particular NVIDIA card and Intel iGPU combo. However, Optimus itself isn't something that I intend to get into here.

Everything that I will outline here was done on a Supermicro X8STi [12] with an entry-level NVIDIA GeForce GT 710 in the single PCIe X16 slot. (I have done the same on my Dell E6430 [13] with an NVS 5200M and Intel iGPU and have actually gotten Optimus to work in the past; though again, that is not something that I intend to go over here for the reasons already mentioned.) If your machine has NVIDIA and Intel, my suggestion is to simply install the NVIDIA drivers and use the NVIDIA card. For me, there was no advantage to having Optimus working, because the battery was already shot and I was better off simply running NVIDIA full time on that machine for the performance, regardless of the electron abuse. Let's go over how to find and install the correct drivers and deal with whatever aftermath may exist.

Determining System Component Information

Once you have the machine up and running, you can check which drivers are being used by simply running the following:

```
$ lspci -vv
```

This will show all of your PCI devices with details included (Figure 1). If you scroll down a bit, you will see one listed as *VGA compatible controller: NVIDIA Corporation GK208B [GeForce GT 710]*, with the particulars of your card listed here, instead of *GT 710*, obviously. When you look at the last two lines of that entry you will see *Kernel driver in use* and *Kernel modules*. Here you will see *nouveau* listed as the kernel driver in use, which, as mentioned, is the basic open source NVIDIA driver. If you are running Pop!_OS or a similar distribution which came with the proprietary NVIDIA drivers preinstalled, then you may see *nvidia* listed under *Kernel driver in use*, in which case this article isn't for you, but most of you will likely see *nouveau*. As a side note, after *Capabilities* you will see *access denied*. To see this field, simply run the same command with **sudo** and you will have access to that information. This will give some additional information, such as power management, but isn't needed for what we are doing.

Finding Out What Driver Packages Are Available

At this point you have confirmed that the card was recognized and is running with the open source driver. To view which proprietary NVIDIA driver packages are available, you will want to run the following command now:

```
$ ubuntu-drivers devices
```

```
adam@adam-X8STi-Ubuntu: ~/Desktop
01:00.0 VGA compatible controller: NVIDIA Corporation GF108GLM [NVS 5200M] (rev a1) (prog-if 00 [VGA controller])
Subsystem: Dell GF108GLM [NVS 5200M]
Control: I/O- Mem- BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B- DisINTx-
Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- >MAbort- <MAbort- >SERR- <PERR- INTx-
Latency: 0, Cache Line Size: 64 bytes
Interrupt: pin A routed to IRQ 11
Region 0: Memory at f4000000 (32-bit, non-prefetchable) [disabled] [size=16M]
Region 1: Memory at e0000000 (64-bit, prefetchable) [disabled] [size=256M]
Region 3: Memory at f0000000 (64-bit, prefetchable) [disabled] [size=32M]
Region 5: I/O ports at e000 [disabled] [size=128]
Expansion ROM at f5000000 [disabled] [size=512K]
Capabilities: [60] Power Management version 3
Flags: PMEClk- DSI- D1- D2- AuxCurrent=0mA PME(D0-,D1-,D2-,D3hot-,D3cold-)
Status: D0 NoSoftRst+ PME-Enable- DSel=0 DScale=0 PME-
Capabilities: [68] MSI: Enable- Count=1/1 Maskable- 64bit+
Address: 0000000000000000 Data: 0000
Capabilities: [78] Express (v2) Endpoint, MSI 00
DevCap: MaxPayload 128 bytes, PhantFunc 0, Latency L0s <4us, L1 <64us
ExtTag+ AttnBn+ AttnInd- PwrInd- RBE+ FLReset- SlotPowerLimit 75.00W
DevCtl: CorrErr- NonFatalErr- FatalErr- UnsupReq-
RlxdOrd- ExtTag+ PhantFunc- AuxPwr- NoSnoop+
MaxPayload 128 bytes, MaxReadReq 512 bytes
DevSta: CorrErr- NonFatalErr- FatalErr- UnsupReq- AuxPwr- TransPend-
LnkCap: Port #0, Speed 5GT/s, Width x16, ASPM L0s L1, Exit Latency L0s <25ns, L1 <4us
ClockPM+ Surprise- LLActRep- BwNot- ASPMOptComp-
LnkCtl: ASPM L0s L1 Enabled; RCB 128 bytes Disabled- CommClk+
ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-
LnkSta: Speed 5GT/s (ok), Width x16 (ok)
TrErr- Train- SlotClk+ DLActive- BWMgmt- ABWMgmt-
DevCap2: Completion Timeout: Not Supported, TimeoutDis+, NROPrPrP+, LTR-
10BitTagComp-, 10BitTagReq-, OBFF Not Supported, ExtFmt-, EETLPPrefix-
EmergencyPowerReduction Not Supported, EmergencyPowerReductionInit-
FRS-, TPHComp-, ExtTPHComp-
AtomicOpsCap: 32bit- 64bit- 128bitCAS-
DevCtl2: Completion Timeout: 50us to 50ms, TimeoutDis+, LTR-, OBFF Disabled
AtomicOpsCtl: ReqEn-
LnkCtl2: Target Link Speed: 5GT/s, EnterCompliance- SpeedDis-
Transmit Margin: Normal Operating Range, EnterModifiedCompliance- ComplianceSOS-
Compliance De-emphasis: -6dB
LnkSta2: Current De-emphasis Level: -6dB, EqualizationComplete-, EqualizationPhase1-
EqualizationPhase2-, EqualizationPhase3-, LinkEqualizationRequest-
Capabilities: [b1] Vendor Specific Information: Len=14 <?>
Capabilities: [100 v1] Virtual Channel
Caps: LPEVC=0 RefClk=100ns PATEntryBits=1
Arb: Fixed- WRR32- WRR64- WRR128-
Ctrl: ArbSelect=Fixed
Status: InProgress-
VC0: Caps: PATOffset=00 MaxTimeSlots=1 RejSnoopTrans-
Arb: Fixed- WRR32- WRR64- WRR128- TWRR128- WRR256-
Ctrl: Enable+ ID=0 ArbSelect=Fixed TC/VC=01
Status: NegoPending- InProgress-
Capabilities: [128 v1] Power Budgeting <?>
Capabilities: [600 v1] Vendor Specific Information: ID=0001 Rev=1 Len=024 <?>
Kernel modules: nouveau, nvidia_drm, nvidia
```

Figure 1: Terminal output of the **lspci -vv** command.

This will take a few seconds depending on your system. For me, it normally returns a complete response in about 15 seconds (Figure 2), so be patient here. My setup returns the following:

```
== /sys/devices/pci0000:00/0000:00:03.0/0000:03:00.0 ==
modalias : pci:v000010DEd0000128Bsv00001462sd00008C93bc032
sc00i00
vendor : NVIDIA Corporation
model : GK208B [GeForce GT 710]
driver : nvidia-driver-460 - distro non-free
driver : nvidia-driver-470 - distro non-free recommended
driver : nvidia-driver-418-server - distro non-free
driver : nvidia-driver-390 - distro non-free
driver : nvidia-driver-470-server - distro non-free
driver : nvidia-driver-450-server - distro non-free
driver : nvidia-driver-460-server - distro non-free
driver : xserver-xorg-video-nouveau - distro free builtin
```

Figure 2: Terminal output of the **ubuntu-drivers devices** command.

```
adam@adam-X8STi-Ubuntu: ~/Desktop$ ubuntu-drivers devices
WARNING:root: pkg_get_support nvidia-driver-390: package has invalid Support Legacyheader, cannot determine support level
== /sys/devices/pci0000:00/0000:00:01.0/0000:01:00.0 ==
modalias : pci:v000010DEd00000FCsv00001028sd00001534bc03sc00i00
vendor : NVIDIA Corporation
model : GF108GLM [NVS 5200M]
manual_install: True
driver : nvidia-340 - distro non-free
driver : nvidia-driver-390 - distro non-free recommended
driver : xserver-xorg-video-nouveau - distro free builtin

== /sys/devices/pci0000:00/0000:00:1c.4/0000:0c:00.0 ==
modalias : pci:v0000144dd000043B1sv0000103Csd00002154bc02sc00i00
vendor : Broadcom Inc. and subsidiaries
model : BCM4352 802.11ac Wireless Network Adapter
driver : bcmwl-kernel-source - distro non-free

adam@adam-X8STi-Ubuntu: ~/Desktop$
```



```
adam@adam-X8STI-Ubuntu: ~/Desktop
adam@adam-X8STI-Ubuntu:~/Desktop$ apt list *nvidia* --installed
Listing... Done
libnvidia-cfgi-470/nou 470.86-0ubuntu0.20.04.2 amd64 [installed,upgradable to: 470.103.01-0ubuntu0.20.04.1]
libnvidia-common-470/nou 470.86-0ubuntu0.20.04.2 all [installed,upgradable to: 470.103.01-0ubuntu0.20.04.1]
libnvidia-compute-470/nou 470.86-0ubuntu0.20.04.2 amd64 [installed,upgradable to: 470.103.01-0ubuntu0.20.04.1]
libnvidia-decode-470/nou 470.86-0ubuntu0.20.04.2 amd64 [installed,upgradable to: 470.103.01-0ubuntu0.20.04.1]
libnvidia-encode-470/nou 470.86-0ubuntu0.20.04.2 amd64 [installed,upgradable to: 470.103.01-0ubuntu0.20.04.1]
libnvidia-extra-470/nou 470.86-0ubuntu0.20.04.2 amd64 [installed,upgradable to: 470.103.01-0ubuntu0.20.04.1]
libnvidia-fbc1-470/nou 470.86-0ubuntu0.20.04.2 amd64 [installed,upgradable to: 470.103.01-0ubuntu0.20.04.1]
libnvidia-gl-470/nou 470.86-0ubuntu0.20.04.2 amd64 [installed,upgradable to: 470.103.01-0ubuntu0.20.04.1]
libnvidia-ifr1-470/nou 470.86-0ubuntu0.20.04.2 amd64 [installed,upgradable to: 470.103.01-0ubuntu0.20.04.1]
linux-modules-nvidia-470-5.13.0-28-generic/nou 5.13.0-28.31-20.04.1+1 amd64 [installed,upgradable to: 5.13.0-28.31-20.04.1+2]
linux-modules-nvidia-470-generic-hws-20.04/nou 5.13.0-28.31-20.04.1+1 amd64 [installed,upgradable to: 5.13.0-28.31-20.04.1+2]
linux-objects-nvidia-470-5.13.0-28-generic/nou 5.13.0-28.31-20.04.1+1 amd64 [installed,upgradable to: 5.13.0-28.31-20.04.1+2]
linux-signatures-nvidia-5.13.0-28-generic/nou 5.13.0-28.31-20.04.1+1 amd64 [installed,upgradable to: 5.13.0-28.31-20.04.1+2]
nvidia-compute-utils-470/nou 470.86-0ubuntu0.20.04.2 amd64 [installed,upgradable to: 470.103.01-0ubuntu0.20.04.1]
nvidia-driver-390/nou 470.86-0ubuntu0.20.04.2 amd64 [installed,upgradable to: 470.103.01-0ubuntu0.20.04.1]
nvidia-prime/focal-updates,nou 0.8.16-0.20.04.1 all [installed,auto-removable]
nvidia-prime/focal-updates,nou 470.57.01-0ubuntu0.20.04.2 amd64 [installed,auto-removable]
nvidia-utils-470/nou 470.86-0ubuntu0.20.04.2 amd64 [installed,upgradable to: 470.103.01-0ubuntu0.20.04.1]
xserver-xorg-video-nvidia-470/nou 470.86-0ubuntu0.20.04.2 amd64 [installed,upgradable to: 470.103.01-0ubuntu0.20.04.1]
adam@adam-X8STI-Ubuntu:~/Desktop$
```

Figure 3: Terminal output of the `apt list` command.

As you can see here, there is more than one package available from NVIDIA along with the Nouveau open source driver. You can double check if your system has any NVIDIA packages installed by running:

```
$ apt list *nvidia* --installed
```

If you have installed any of these, they will be listed along with any dependencies that are required for them to run which are also present on your system, such as libraries and the settings dialo box package (Figure 3).

It is here that things can sometimes get a bit difficult. While all of these packages are available for your card, not all of them will work well, properly, or for that matter, at all, with your specific card. With that in mind you should at this point back up everything that you need to save somewhere safe and on a different machine, external drive, cloud storage, or (better yet) all of the above. Personally, I know that with my GT 710 the best functioning package is listed as *nvidia-driver-390*, though frankly, I only know that through trial and error. Documentation on these packages is good and you can find more information about them at the NVIDIA website [1]. Personally, I had to learn this the hard way, by trying a package, uninstalling it when it failed, trying a different package, rinse and repeat. For that reason I would like to give a bit of a disclaimer here: If you install one of these packages and reboot to a black screen or have some other problem with your display, then be patient and don't fret just yet.

Device Driver Installation

To install the package that you would like to try, simply run:

```
$ sudo apt update
$ sudo apt install nvidia-driver-390
```

replacing the *nvidia-driver-390* with the package of your choice for your specific GPU. Using the terminal

here has the advantage that you can easily see what is going on, as opposed to using the GUI, which often leaves one wondering. Once the installation is completed and all dependencies have been installed, you will want to reboot your machine so the new software can be loaded at boot. If you are lucky, you will be greeted with the login screen or desktop as you normally would. You can then check that your installation worked by either using the `lspci -vv` command discussed earlier or by opening settings

and looking at the About section to see what is listed for Graphics. Mine shows *GeForce GT 710/PCIe/SSE2* while using the *nvidia-driver-390* packages. As mentioned before, using `lspci -vv` will show *nvidia* under the *Kernel driver in use* section. You will also have the NVIDIA control panel installed for changing the particulars of your setup with a graphical interface.

Troubleshooting and Resolving Issues

If you were *not* greeted with something graphical but rather a blank screen, then it is likely that the package is not in fact suitable for your GPU and will need to be removed. Don't panic just yet, however. This is a fairly easy and straightforward process as well. Reboot, and at the GRUB loading screen, instead of choosing a normal boot into Ubuntu, choose *Advanced Options for Ubuntu* and then choose the *Ubuntu, with Linux ... (recovery mode)* option. The ... will show the kernel that you are running and you can select the most recent kernel here. Make sure to select the *recovery mode* option regardless.

Once in recovery mode you will want to choose root, with the description of *Drop to root shell prompt*. This will boot into a command line and from here you can get rid of the offending packages. Run the following to find out which NVIDIA packages are installed:

```
$ apt list *nvidia* --installed
```

Once you have seen what packages are there, you can use the following two commands in the following order to rid yourself of the NVIDIA packages. First use:

```
$ apt remove nvidia-driver-390 nvidia-settings
nvidia-prime nvidia-dkms ... etc.
```

You will have many NVIDIA packages here, and listing all of them after the driver package itself with a space between them will remove all. This

will get rid of dependencies and libraries so that when you try the next option you are starting from a clean slate. In the example above I listed a few of the packages that show up when I run the `l1st` command, but there are about 20 or 22 of them in total. Once that is done, you will finalize the uninstallation by running the same but with `purge` instead of `remove`, à la:

```
$ apt purge nvidia-driver-390
nvidia-settings nvidia-prime
nvidia-dkms ... etc.
```

This gets rid of any settings and configurations that may still be lingering after removal of the packages themselves. You can also do a quick cleaning with the following command after all of that:

```
$ apt autoremove
```

just to be sure that you have gotten everything out of there that can be removed.

When this is complete, simply type `reboot` and hit Enter to reboot the system. At this point you should see that the *nouveau* drivers are loaded and running instead of the proprietary NVIDIA ones. You can now try a different driver package by listing and installing as you did before, and I would recommend going to the next available lower number. In my experience most of the problems that I have had with installing NVIDIA drivers stem from the fact that my old cards aren't well supported on newer drivers, even though those newer drivers will show up when queried. Going to older available packages one at a time ensures that I have the most up-to-date set of packages available for my card which will run without problems.

Concluding Remarks

I cannot express the amount of frustration I felt as a beginner around installing one of these only to have a blank screen be the result upon reboot. At that time, my solution was a complete reinstall of the OS, at which point I would try to use the open source driver package with its faults or go to a different NVIDIA driver and cross my fingers and hope for better results. Yes, I have reinstalled Ubuntu more than four times simply trying to get a driver package that worked. That is exactly why I think it is important not only to be comfortable with the terminal and with using the recovery mode, but also simply to know a bit about what is happening under the hood. Now that I have a method for recovery upon failure and have some

confidence that I can get my system back to a working desktop, I can say that I simply find it easier and faster to use the terminal for the entire process. The added benefit here in my opinion is that you can actually see what is happening in the terminal rather than waiting, guessing, and hoping with the GUI equivalent (Figure 4).

Hopefully one day we will see the NVIDIA gang team up with our beloved open source community to produce a non-proprietary solution with the performance and benefits of the closed source offering that currently exists. Until then I am afraid we are stuck with some of this hassle. Following the advice given above, however, you at least won't have to be stuck at a blank screen for too long after installing their solution onto your machine. I guess it really is as they say: You win some and you lose some. ■■■

Info

- [1] NVIDIA: <https://www.nvidia.com/>
- [2] AMD Radeon graphics cards: <https://www.amd.com/en/graphics/radeon-rx-graphics>
- [3] Ubuntu: <https://ubuntu.com/>
- [4] Red Hat Enterprise Linux: <https://www.redhat.com/en/technologies/linux-platforms/enterprise-linux>
- [5] Fedora Linux: <https://getfedora.org/>
- [6] Nouveau Xorg driver: <https://launchpad.net/nouveau>
- [7] SteamOS and Linux titles: <https://store.steampowered.com/linux>
- [8] Pop!_OS by system76: <https://pop.system76.com/>
- [9] YouTube: Gaming on Linux is NOT Ready: <https://www.youtube.com/watch?v=Rlg4K16ujFw>
- [10] NVIDIA Optimus technology: <https://www.nvidia.com/en-us/geforce/technologies/optimus/technology/>
- [11] Intel GPUs: <https://www.intel.com/content/www/us/en/products/discrete-gpus.html>
- [12] Supermicro X8STi: <https://www.supermicro.com/products/motherboard/Xeon3000/X58/X8STi.cfm>
- [13] Dell Latitude E-Series laptops: <https://www.dell.com/>

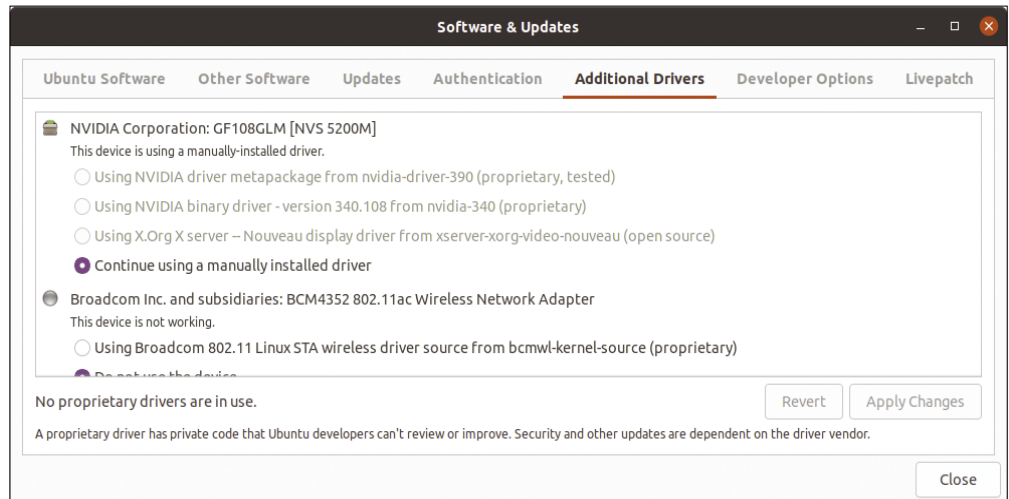


Figure 4: Choosing an additional driver in the GUI.

FOSSPicks

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



Graham had an interesting moral dilemma this month: whether to include an open source Wordle clone or not. Read on to see his decision.

BY GRAHAM MORRISON

Graphics studio

Blender 3.0

We've looked at updates to Blender before but never one as significant as this. Blender 3.0 comes 22 years after Blender 2.0. This period covers the project from directly after the source code was released as freeware in 2000 through the creation of the Blender Foundation in 2002 and its true open sourcing in 2003, before the incredible number of world-changing feature updates that have defined the 2.x release

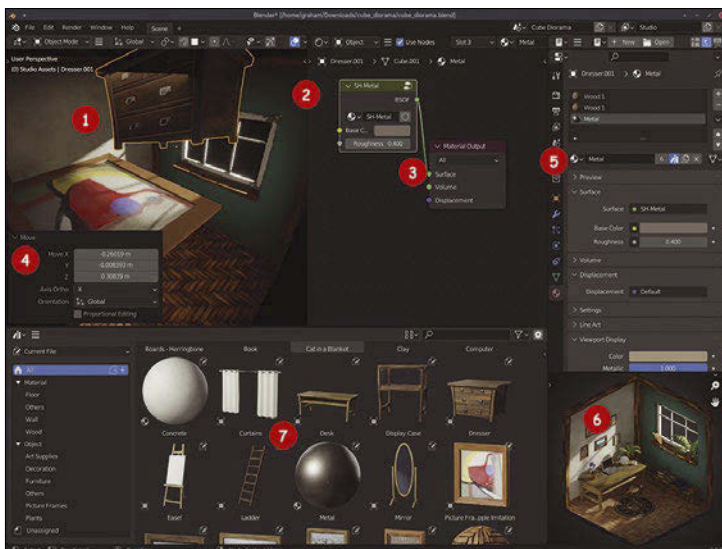
cycle. Hugely influential, Blender is a lighthouse project for what open source can achieve. From modeling, animation, movies, and 3D rendering to architecture, 3D printing, and game design, Blender has received unparalleled industry recognition and is used by hobbyists, students, and professionals alike.

Not just a token release, Blender 3.0 contains a significant number of new features and refinements, especially if you've skipped a few updates. The best of these is the

asset browser, a new kind of view type that appears in the menu alongside the 3D viewport, compositor, and all the other types. When enabled, the chosen pane will show thumbnails from a huge number of Blender asset types, including objects, materials, and most recently, poses. Poses are a way of storing character expressions for your models, such as a smile or frown. Blender 3.0 will let you tweak these poses without losing the original, with keying sets for both characters and properties. You can then add, blend, apply, and flip poses from the asset view to see how they'll fit on your chosen character. Key values can even be selected by dragging a square across the frame graph. Equally, assets can be dragged and dropped into your scene, which will update as quickly as your hardware allows, even when adding high-dynamic range image (HDRI) lighting assets. It's a great way to quickly drag a model with materials and lighting into your scene. You can contribute back to the library by marking scene elements as assets – which can also be linked, rather than copied – so that only a single instance is used across multiple projects.

Speed and quality improvements are everywhere. The knife tool can cut across multiple objects and is finally included in the undo history. Drawing strokes can be modified concurrently, and six degrees of freedom (6DOF) controllers have even been added to the virtual reality mode. When you generate final output, rendering is two to eight times faster thanks to the new Cycles X renderer. Viewport updates are faster, as is point-of-view (POV) mode for setting up a camera, which even in complex scenes remains interactive enough to line up shots, with further detail added the longer that view is maintained. Everything looks better, with improved shadows, hardware ray tracing, and light scattering on the surface of materials. Even the UI is much improved. If you've not used Blender for a few years, you'll find on-screen prompts, sliders, and controls are now available for almost every function. Each view is accompanied by icons or glyphs for the main functions, and everything is a lot more intuitive than it used to be. There are still many options and many menus, but you can get started without using 90 percent of them. This is helped by the default theme which now has more configurable contrast and even more rounded corners. Blender is now easier to use than ever and even more capable of producing amazing results.

Project Website
<https://www.blender.org>



1. Viewport rendering: The previewed output in Blender now looks almost identical to the final output. **2. UI improvements:** Mouse over an area to highlight its edges and more easily join or merge different panes. **3. Geometry nodes:** This release has new nodes for interaction with curves, text data, and instances. **4. Properties display:** Rather than guessing or remembering key combinations, properties can be changed with the UI in the viewport. **5. Improved icons:** Glyphs and icons across Blender are now much easier to understand and use. **6. Speed improvements:** Not just in the preview – everything is now faster, from nodes to rendering. **7. Asset manager:** Keep all your best ideas in one place across projects, including character poses.

Console web service access

woob

Woob is an acronym for “web outside of browsers,” and its mission is to provide access to specific websites without requiring a web browser. It does this by being a command-driven front end to a module-driven back end, with each module targeting a specific kind of web service. Type **woob** followed by **bank**, for example, and you’re presented with a huge list of modules designed to work with online banks. What’s most surprising is that while the project is young and relatively niche, there’s already a huge number of modules to support each of its capabilities, with more being added all the time. There’s support for 88 banks, for example, from Barclays to Swiss Life. Selecting one of these will walk you through configuring

the module with your own credentials so you can access whatever capabilities the module supports. For the bank module, these capabilities include viewing your interest rate, the ability to transfer funds, and retrieving your balance.

Of course, you might wisely not trust woob to negotiate your bank account without some kind of audit of its code, but woob also supports many less crucial services. The cinema capability lets you interact with film metadata grabbers and local download services. The gallery module does the same for picture services, while parcel helps you access tracking information from companies such as DHL and DPD. There are many more, including those that deal with local Linux command-line tools and sensors, remote web services such as weather and bug tracking, and even cooking recipes and music lyrics. You don’t even

```

Type "help <command>" for more info about a command.
recipes> search brie
1 - Brie chaud et pané (750g)
  description: Délicieux savoureux simple Brie pané à servir avec des légumes crus !
2 - Cake aux lardons et olives au Brie Cœur de Lion (750g)
  description: Pour toutes les occasions, ce cake aux lardons et olives au bon brie Cœur
  de Lion se dégustera entre amis ou en famille.
3 - Sauté de porc au brie et à ma façon (750g)
  description: Délicieuse façon d'allier la viande et le fromage.
4 - Quiche brie brocolis (750g)
  description: Une petite quiche, à déguster tiède ou froide avec une salade !
5 - Gratin de chou-fleur au Brie Cœur de Lion (750g)
  description: Autour d'un gratin de chou-fleur au Brie Cœur de Lion.
6 - Tatin parmentière au brie, miel et noisettes rapide (750g)
  description: La tarte tatin, version sucrée, c'est bon, même très bon ..... mais cette
  tatin salée. Cette recette remporte la 7ème place du concours Gûishons l'unique.
7 - Brie farci (750g)
  description: A vous faire fondre...
8 - Tarte tatin courgette brie-miel (750g)
  description: En entrée ou en plat cette tarte tatin est idéale. Simple et bonne aussi bi
  en chaude que froide.
9 - Mini pizzas courgette Brie (750g)
  description: Servez 2 pizzas par personne avec une salade verte aux tomates cerises pour
  équilibrer le repas. Retrouvez toutes les recettes MAGGI.
10 - Pizza wrap poivrons et brie aux olives (750g)
  description: Petits rouleaux de pizza simplissimes et goûteux, parfaits pour un snacking.
Hint: There are more results available for 750g (use option -n or count command)
recipes/#search>

```

Access some of your favorite web services from the command line
– without a web browser – with woob.

need to use the command line because many are accompanied by a Qt-based graphical front end, which can be used instead of the command line. These have the added advantage of showing images such as movie posters, album covers, or recipe photographs. It can take some effort to get started, but after you’ve used a couple of modules you realize it offers a better user experience than many of the web services it hopes to replace, and that’s definitely a good thing.

Project Website
<https://woob.tech>

Cat clone

bat

The Unix philosophy of having one simple tool to do one simple task is still alive and well on the command line, and the **cat** command is one of its best exemplars. It’s a command so venerable that it was included in the original first edition of “Research Unix,” developed by Ken Thompson and Dennis Richie in 1971 (though the term “Research Unix” was first used in 1978). As everyone knows, **cat** will print the contents of a text file to the standard command-line output, letting you easily see its contents. If more than one file is listed, it will concatenate the output of each in turn (the task that gives it its name) and let you view the contents as a whole or pipe them back into a

new, single file. It can optionally display line numbers, or highlight tabs or the end of a line, but it can’t do much more. It’s a simple tool for a simple job, and it works brilliantly.

But there are times when you sometimes need a little more from the humble **cat**, especially when rummaging through source code or text files with a specific format. This is where the equally zoomorphic **bat** can help. It extends **cat**’s simple premise with a few extra modern essentials, including syntax highlighting, built-in paging, and even **git** integration that can show only the modifications in a file after comparing it with the **Git** index. All of this happens when you simply replace your **cat** invocation with **bat**, but

```

// I added this as a test

std::string wrap_text(std::string str, const int line_width, char find = ' ')
{
    for (int i = 0; i < str.size(); ++i)
    {
        if (i % line_width == 0)
        {
            int n = str.find(find, i);
            if (n != std::string::npos)
                str[n] = '\n';
        }
    }
    return str;
}

bool vec_find(std::string key, std::vector<std::string> const & vec)
{
    return std::find(vec.begin(), vec.end(), key) != vec.end();
}

int get_terminal_width()
{
    struct winsize ws;
    ioctl(STDIN_FILENO, TIOCGWINSZ, &ws);
    return ws.ws_col;
}

```

bat is one of those rare commands that makes you wonder how you ever lived without after you start using it.

there are further options to customize the output too, including options to specify when to disable the pager, when to display only the differences, and how one file type maps to another. It’s fast, looks beautiful, and can be themed, while still remaining completely compatible with **cat**’s original basic output, making it just as useful for previewing **zfp** or finding results as it is previewing the contents of **main.cpp**.

Project Website
<https://github.com/sharkdp/bat>

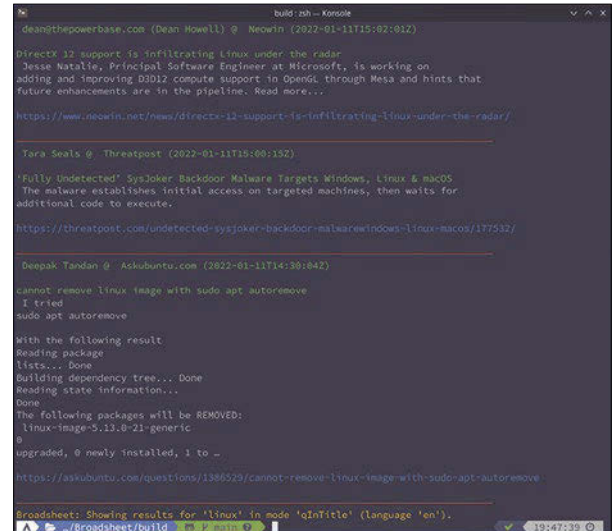
Terminal news

Broadsheet

The amount of news that's being generated has become overwhelming, regardless of how you feel about current events. We've all become "doom scrollers," incessantly skipping from one story to the next like unrepentant information addicts. There aren't many good solutions to this problem. The curated content that you find in magazines like this can help, especially if that means also getting away from the screen. Subscribing to an RSS feed is another possible solution because it lets you purposefully follow updates from a specific source and filter those updates from your own client. But RSS has been in decline for years, and many general news sources have a vested interest in not supporting it – making you visit their

sites, create accounts, and even subscribe.

Broadsheet, a new option in an early stage of development, could bridge the gap between too much news via your usual channels and the lack of support for less distracting methods. It's a command-line tool that outputs a very RSS-like synopsis of news stories, along with links to read the full story in a browser. Command-line arguments can be used to limit stories to a specific domain, country, or search results, and it works brilliantly. Because you're on the command line, clicking on a link is harder and less of a temptation than it might be in a browser. The only downside: Broadsheet requires a subscription to a news aggregation service, <http://newsapi.org>, which



While Broadsheet only currently works with a subscription-based news API, the developer hopes to add RSS support in the future.

provides an API key to use within the configuration file. It's currently reasonable to use the free developer tier, but this will change when Broadsheet becomes stable. Hopefully by then the developer will have added the proposed RSS support, resulting in the best of both worlds.

Project Website

<https://github.com/i-am-the-gabe/Broadsheet>

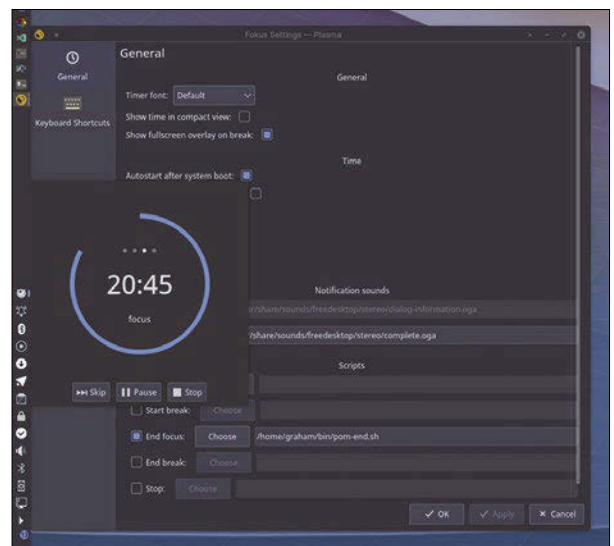
Pomodoro widget

Fokus

If Broadsheet has got you thinking about removing distractions from your work environment, another tried and tested method is the Pomodoro technique. This is a time management hack developed by Francesco Cirillo in the 1980s, and it's often accompanied by a timer in the shape of a tomato (pomodoro is Italian for tomato). The process simply asks you to focus on a single task for a set period of time, usually 25 minutes, without letting yourself be distracted by other things. The contemporary distractions of the '80s would be benign by modern standards, being perhaps a Rubik's cube, Newton's Cradle, or a fresh cup of coffee. Today there are more distractions than you have tabs open, and something

like Pomodoro can make a huge difference in your productivity. The main problem is how to integrate it into your work environment without wasting a week researching the options. Not wanting to procrastinate any further your search for a solution, we've found one for you.

The original tomato clock was designed to keep you away from the temptation of technology. Because it's likely you'll want to work on a computer, moving it to your desktop is an acceptable solution. There are many options for Linux, but one of the best we've come across is Fokus, which, as you might guess from the *k*, is a KDE Plasma application widget. It features both a small panel containing a timer and a panel widget that fits even small panels, and both will show you how much time you have left for each interval. Cleverly, it can also be configured to run a script at the beginning or end of a



Get on top of your work, and your self esteem, by being wholly productive for 25 minutes without getting distracted. It really works!

period, and it's easy to write something that adds the work for a period into a personal journal or work log. Fokus is beautifully designed and better than the equivalent applications we've seen for macOS and Windows, many of which are commercial or advertising driven.

Project Website

<https://gitlab.com/divinae/focus-plasmoid>

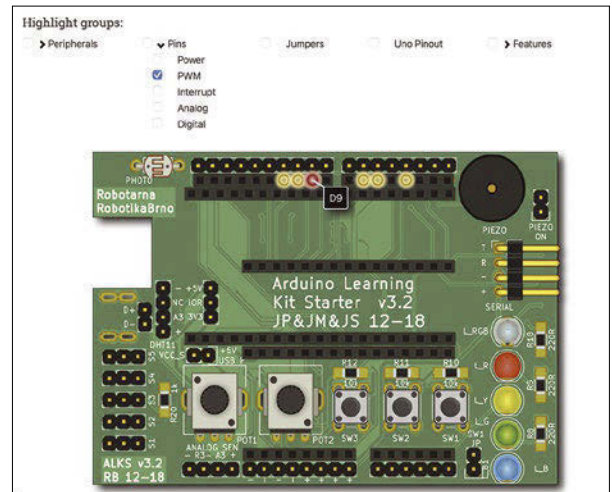
PCB illustrator

Pinion

One of the most recent revolutions in technology, alongside the likes of home 3D printing, virtual reality, and the Raspberry Pi, is the ability for any of us to create a circuit and get it printed. There's even a choice of software you can use to do this, and we've covered several in this magazine. Perhaps the most mature is KiCAD, a well-established schematics editor that lets you create a circuit, verify its design, simulate its execution, and test the electrical rules that govern its function. KiCAD will then automatically generate a PCB layout and export the results in a format that can be sent to a PCB manufacturer. It's simple enough that even beginners can use it to produce their first circuits, but it's

also advanced enough that many engineers use it for their product design.

Pinion is a tool that takes KiCAD files and turns them into beautiful illustrations that you can share or publish on a website. Not only do they look incredible, with a cell-shaded, pseudo-realistic style, they're also interactive. Text can be added to parts of the circuit or the board, and tracks can be highlighted by selecting them. The virtual card can also be flipped over, all thanks to the magic of JavaScript. This functionality is added via a simple YAML file that links the KiCAD components together with your descriptions, grouping, and pin annotations. The text you add can be as long as needed and even include links and basic



Pinion is a simple Python tool that can save you the trouble of drawing your own realistic circuit diagrams.

formatting. It uses a hand-drawn library called PcbDraw-Lib to illustrate the components, and while the library is currently relatively small, it's growing and easy to contribute to. The diagram is then built by executing the `pinion` command, together with arguments pointing to the YAML and the KiCAD file. The results are excellent!

Project Website

<https://yaqwsx.github.io/Pinion>

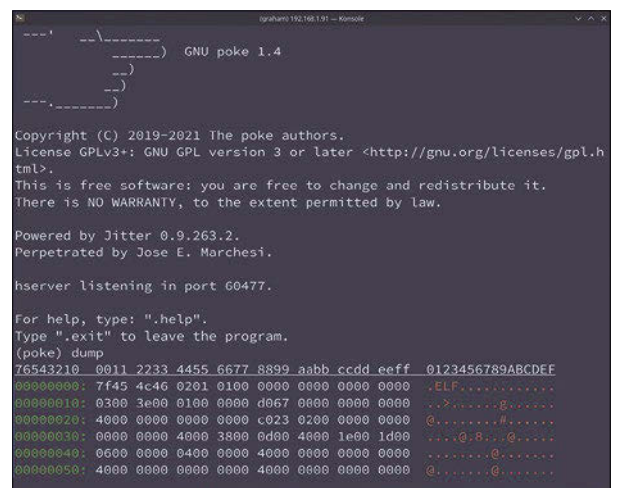
Binary editor

GNU poke

We often look at hexadecimal editors on these pages, and they can be useful for looking into binary files, but we don't often find an actual binary editor like GNU poke. Its name refers to a couple of ancient commands that originated with the DECsystem-10 but became infamous on the Commodore 64. The PEEK command would be used to view the contents of a memory location, and the POKE command was used to change its value. Both commands were crude, but they also allowed anyone to hack whatever was running in system memory. They were easy system exploits, and if you had a good understanding of the assembler and the hardware, you could achieve almost anything with lots of PEEK and POKE

commands, no Internet, and too much time over a long summer break.

GNU poke is a binary editor that will appeal to those old-school (now middle-aged) hackers, but also to anyone with an interest in poking around binary files. It describes itself as "spartan," which in this context means there's not much to see when it's running. Launching it with a file as an argument doesn't open an editor you'd recognize but instead will offer you a prompt and some primitive help. Typing `.dump` will generate output that looks like a hex editor, allowing you to see the first 128 bytes of your selected file. But the key to using poke is understanding the structure of the data you're looking at, because that allows you to use operators and even



Big and little endians, characters, strings, and even bitmaps can be inspected and modified with GNU poke.

variables to procedurally edit and modify your data. Data can be copied to additional buffers, and there's decent file input and output for your saved changes and discoveries. It's deep and complex and difficult to learn despite the excellent internal documentation, but it's easier than doing this stuff manually in 1986 with PEEK and POKE commands and a summer spent reading hardware reference manuals.

Project Website

<https://www.jemarch.net/poke>

Game creator

Bladecoder Adventure Engine

When video games first became popular and accessible, it seemed like each new month brought a new gaming genre. Arcade conversions, text adventure games, vertical shoot-'em-ups, horizontal scrolling shoot-'em-ups, platforms, strategy games, and point-and-click adventures all appeared over a few short years. As technology improved, it seemed these game types would be forgotten as progress supplanted them with more engaging and more immersive game types. But this hasn't happened. Instead, many of those original game types have survived and are even flourishing in an age where GPUs perform real-time raytracing and smartphones are millions of times more powerful than the Apollo 11 guidance computer. In particular, 2D graphics are still relevant, remaining popular in platformers, RPGs, and point-and-click adventure games. Even the original Sierra developers returned in the adventure games category, with the successful crowdfunding and release of Thimbleweed Park in 2017 and many similar titles released in the proceeding years.

There's more to the old game ideas than retro gaming and nostalgia. Many of them can be the basis of a modern legitimate title, while hopefully also being easier to produce on modern hardware than the original titles back in the day. The Bladecoder Adventure Engine is a perfect example of this. It's a set of modern tools that help you create interactive graphic adventures, much like Monkey Island or Thimbleweed Park, and it's even being used to create an Android and iOS game called The Goddess Robbery. Using the editor, this and other games can be downloaded and inspected in order to help you learn how to build your own games.

The engine itself consists of two parts: a graphical editor and the background engine for running the resulting game. The graphical editor (Adventure Editor) is where you're likely to spend the most time because it's where your project is created, as assets are added, and functionality implemented, all without requiring any programming. Projects consist of one or more chapters containing scenes. Scenes are analogous individual locations,



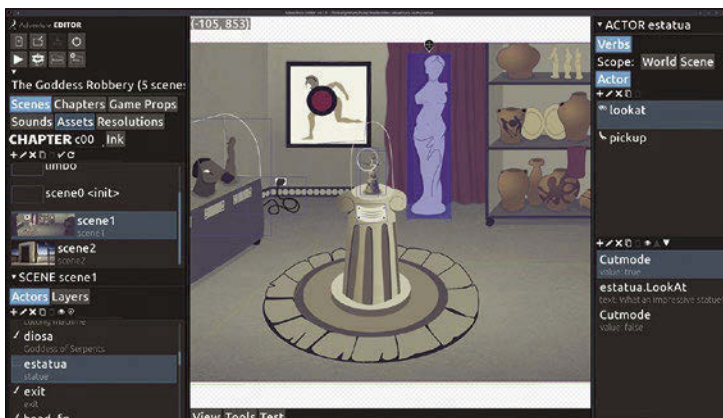
Actors and scenes can be fully animated with small previews, and full screen playback is easily accessible.

such as the Scumm Bar or Governor's Mansion in the original Monkey Island, and they're core to point-and-click gameplay. The editor enables you to easily import assets, including the graphics for a background, sprite actors for movement, and objects to show, hide, and collect. All of this is tied together by adding clickable areas in the background linked to verbs such as "lookat," "pickup," or "talkto" to perform actions. This is game creation via point-and-click, using the panels around the principal scene window, and the application has more in common with a graphical editor such as Gimp or Krita than an IDE. The end result can be exported and published as a standalone game without any programming or compilation.

There's initially a lot to learn, but there are some excellent example projects, good documentation, and even a few video tutorials to help get you started. It's easier than using Godot or the ancient Adventure Game Studio and could be a great way to get someone into game development who might be intimidated by programming or using something less-graphical. It's also perfect for collaboration because the work can so easily be split into artwork, sound design, script writing, and game design, with an end result that can be shared with anyone, even commercially.

Project Website

<https://github.com/bladecoder/bladecoder-adventure-engine>



The quality of the games you can produce on Bladecoder Adventure Engine is dependent on the quality of your artwork.

Multiplayer shooting platformer

Teeworlds

When Teeworlds describes itself as a multiplayer shooter, you might be forgiven for thinking it may be a game set in space where you zoom across a star field chasing ships being flown by online competitors. This game, however, is set in the world of a retro, 2D, side-scrolling platformer, and you play as a cartoon smiley face with a gun. The background artwork and animation is similarly cartoon-like, featuring seasonal changes and the usual mix of canyons, snow and ice, and intensely green foliage. You jump and zoom around these levels trying to avoid getting shot while also trying to shoot the other players in the same level. Rather than simply pointing left or right in

your direction of travel, the gun angle can be changed as you fly through the levels, helping you aim more accurately. This is an unusual mechanic that can feel a little like playing classic two-player tanks while on the move, but it also requires an additional level of skill as you try and take on all the variables, including gravity, special effects, and bullet type. Another unique element is a Bionic Commando-style grapple or swinging arm that helps you traverse the level like a caffeine-fueled Spiderman, leading to some very unique gameplay. The multiplayer aspect is core to the game, and there are several online modes to choose between. The most popular are Deathmatch and Capture the



Teeworlds' license is OSI compliant, but its bespoke wording is closer to a public domain game than we'd like.

Flag – where you hunt down the flag, capture it for yourself, and defend it against your adversaries for as long as you can. This all depends on the popularity of the game online, and thankfully, the game is well established and has a great following. There were over 600 servers when we checked and over 1,200 players playing online, making it easy to find a game. You can also run your own server locally for your own use or as part of the online community, and it's the same for the levels themselves thanks to a super-easy level designer.

Project Website

<https://www.teeworlds.com>

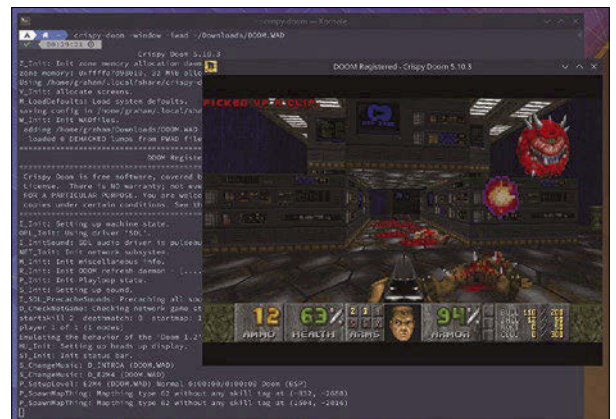
Doom clone

Crispy Doom

There will never be enough Doom. It's one of those games that hasn't really aged, despite technology advancing towards science fiction levels. Modern graphics capabilities eclipse those of a game nearly 30 years old, and modern gameplay is now far more complex and nuanced. And yet, Doom is still perfectly nuanced, and the original aesthetic, sprites, and control methods are part of that experience. Crispy Doom is a clone that plays with this idea slightly, lifting some of the limits on the original Doom to explore other possibilities without breaking any of the core mechanics that made the original game so successful. This is why it describes itself as a

friendly fork of Chocolate Doom, a clone that attempts to accurately reproduce Doom as it would have been in 1993 on 1993 hardware.

Crispy Doom differs from the original in several ways that make it more convenient on modern hardware. It can run with a widescreen aspect ratio for our modern monitors and at a 1993-eclipsing 640x400 display resolution. There's gamma correction for playing with the contrast and darkness, and options that can enable free vertical looking, a heads-up display, colored status bars, and many other options. All of these can be set from the in-game menu or via the configuration file installed when you first run the game, with none of the options breaking the

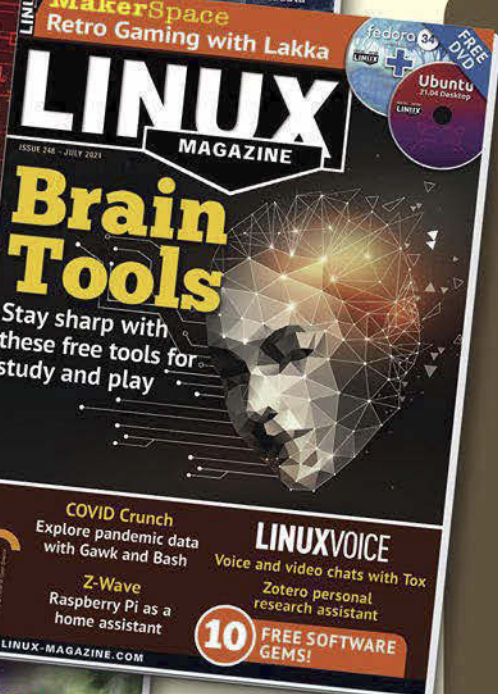
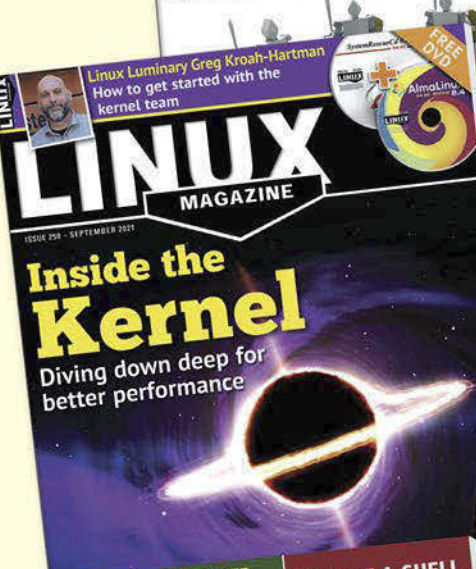


Crispy Doom requires the original WAD file to play the game, but the Doom demo WAD is free and others are easy to purchase or retrieve from your own copy.

essential playability of the game. It's actually remarkable how modern the game can feel by simply running it at 120 frames per second on a modern screen. It's perfect for playing on a Raspberry Pi, for example, and builds quickly on its AArch64 system architecture. Putting this together with a battery, controller, and screen could be the basis for a Raspberry Pi gaming handheld, and Doom would remain just as fun.

Project Website

<https://github.com/fabiangeffrath/crispy-doom>



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 plotting library

Data Visualization

Matplotlib offers so many options that you may have trouble deciding on which ones to use for your plots. **BY MARCO FIORETTI**

Matplotlib [1] is an open source Python library for high-quality data visualization, which you can use interactively or inside scripts. While Matplotlib might look like just a supercharged version of the venerable plotting utility Gnuplot [2], that's not the whole story. Matplotlib really shines in the integration of charting with heavy numeric processing. When combined with other Python libraries, such as NumPy and pandas (see the "Matplotlib's Partners: NumPy and pandas" box), Matplotlib is often considered an effective open source alternative to MATLAB [3].

Some tools have steeper learning curves than others. In my opinion, Matplotlib's learning curve is *wider*, not steeper than average. For example, Matplotlib has an entire tutorial devoted to colors specifications! In other words, Matplotlib has so many features and options that it looks much more complex than it actually is. For this reason, much of *this* tutorial uses simplified versions of some official examples to introduce the main Matplotlib concepts and features and connects the dots between them.

NumPy and pandas

NumPy [4] is the core library for scientific computing in Python. As far as Matplotlib is concerned, NumPy's most important components are the high-performance data structures used in some examples of this tutorial and the functions to manage them. Pandas [5] is another library, specializing in manipulation of numerical tables and time series. The combination of Matplotlib, NumPy, and pandas for numeric processing is really powerful, but it is also a relatively advanced topic that would be impossible to cover properly in just one tutorial for beginners. However, Matplotlib is really useful even alone. What you will learn with this tutorial will make it easier to add NumPy and pandas later on.

Install Procedure

You will find several ways to install the most recent stable version of Matplotlib, all properly documented on its website. However, unless you *really* need the LATEST stable version, I strongly suggest that you save time and frustration by installing whatever binary packages are available in your preferred distribution's official repository. On Ubuntu 21.04, for example, you can install a recent Matplotlib and all its dependencies by just searching for it in the Ubuntu Software Center or, as I did, by typing this command at the prompt:

```
#> sudo apt-get install python3-matplotlib
```

Coding Styles

There are two distinct coding styles for writing and (re)using Matplotlib code: object-oriented and pyplot.

The first style (Listing 1) uses classical object-oriented programming in any language: First you create objects (`fig` and `ax`, whose meaning I will explain shortly), and then you invoke "methods" associated to them (e.g., `ax.plot()` to do the actual plotting).

The second style (Listing 2) uses pyplot functions (in which the `plt` prefix means "use this function provided by the pyplot module") to create and manage charts similar to MATLAB.

Listing 1: Object-Oriented Style

```
fig, ax = plt.subplots(figsize=(5, 2.7))
ax.plot(x, x, label='linear') # Create a plot
ax.legend(); # Add a legend.
```

Listing 2: Pyplot Style

```
plt.figure(figsize=(5, 2.7))
plt.plot(x, x, label='linear')
plt.legend();
```

Listing 2, in fact, does exactly the same things as Listing 1 but by calling stateful functions such as `figure`, `plot`, or `legend` (instead of methods attached to an instance of some object). In this context, “stateful” means that each pyplot function call *adds* or changes something to the *current* state of a plot, without erasing what has been done by previous calls of other functions.

In practice, pyplot function calls have equivalent Matplotlib object-oriented methods, and vice versa, to such an extent that your choice of coding style is a matter of personal preference. However, the official Matplotlib documentation suggests using the pyplot style for “quick interactive work” and using the object-oriented style, which is more flexible, for code that will be reused often in many different ways.

Matplotlib Terminology

Before looking at any actual code, I need to explain some basic terminology, because Matplotlib gives a few words a slightly different meaning than they have in ordinary English.

Aside from what you see on the screen, Matplotlib uses “Figure” as a keyword that creates a container of objects called “Artists.”

Basically, everything visible inside a Matplotlib Figure is an Artist, from drawings to embedded images, text captions, and so on. Each Figure keeps track of, and connects, all its child Artists, including nested subfigures, non-graphic objects such as titles or legends, and, of course, “Axes.”

An Axes (plural!) is maybe the most basic and most essential Matplotlib Artist: It is a region, directly attached to a specific Figure, where you can plot some dataset. You can create Axes together with their parent Figure or add them later.

Each Axes has at least a title and two labels, one per axis. Most other Artists cannot be moved to, or shared with, an Axes (that is, one specific plot or subplot) different from the one they were originally attached to.

Another essential Matplotlib term, “Axis” (singular), has pretty much the same meaning it has in normal English. An Axis object sets all the components of one axis of a plot, from the scale (linear, logarithmic, or other) to the extreme points, marks (also called ticks), and their corresponding text labels. Each Axes includes at least two Axis objects, but it is possible to add more, as shown later in this tutorial.

From figure borders to colors, there are a zillion ways to style each single detail of any Matplotlib plot and two high-level ways to set them. The quick and dirty option, which may be enough in some scenarios, is to manually set all the options you need when you generate a plot. The other option is to use full styles, which are

predefined listings of all the default visual settings that your plots will need. You may load either one of the predefined styles shipped with Matplotlib using the `style.use` function:

```
plt.style.use('someMatplotlibStyle')
```

You can also load a custom style sheet from a file that you have created. In both cases, changing styles will only require you to change the one line of code that contains the style name or the path to the file that stores it. In addition, because both methods will ignore any non-visual setting that may be inside a style, styles are portable across machines with very different installations of Matplotlib.

Data Structures

As far as data structures are concerned, no matter how you define and fill a sequence of values, Matplotlib will internally convert it to the special arrays provided by NumPy, because those are the formats that its plotting functions can handle. In most cases, you should directly use the same arrays anyway, as shown in the examples that follow, because they are faster to process than normal Python arrays and consume less memory.

Matplotlib Gallery

With the general concepts presented above in mind, I will use some simple, but very different, examples to create plots with Matplotlib. You can easily adapt the following examples to a variety of use cases.

A Basic Plot with Line Styling

The Matplotlib code in Listing 3 generates the plot shown in Figure 1. After loading pyplot and NumPy in the first two lines, Listing 3 defines a variable `t` that increases in steps of 0.2 units in the interval from 0 to 5 (line 4). Lines 6 to 8 show how to plot three different curves, each with its unique style and label. The format of the strings used to style the lines

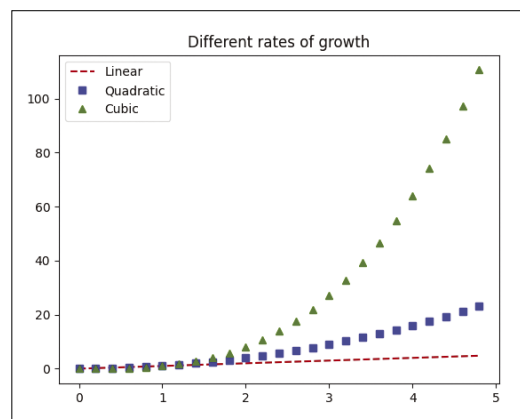


Figure 1: A really basic plot still looks nice thanks to Matplotlib styling options.

Listing 3: Basic Plot

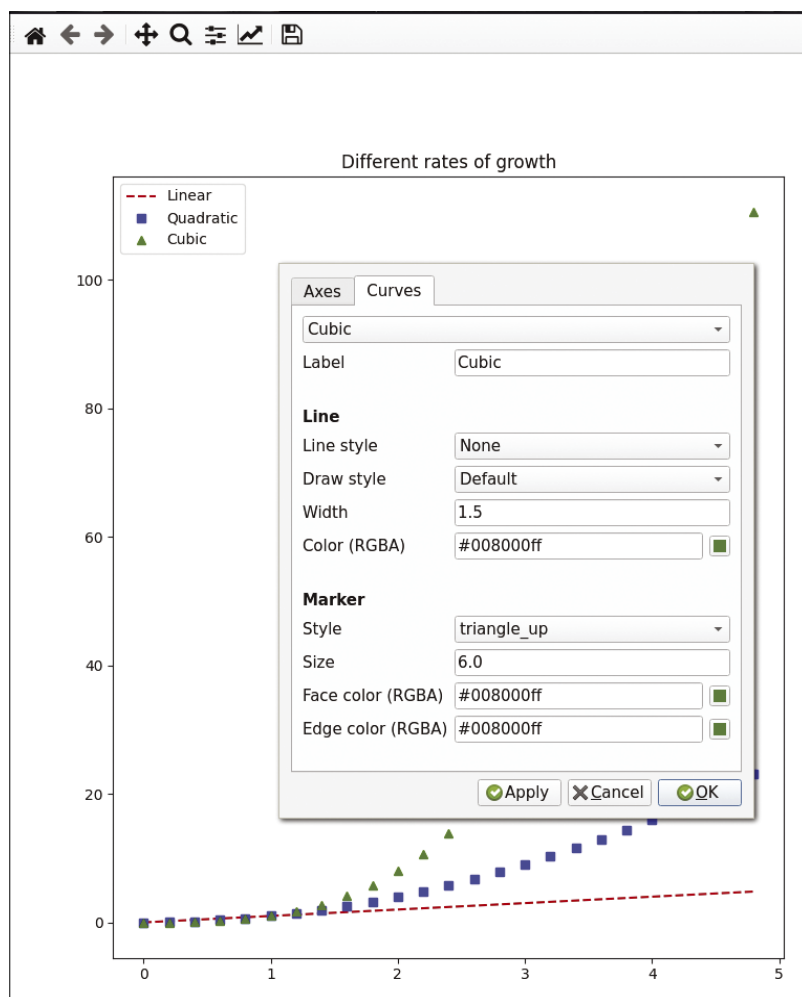
```
01 import Matplotlib.pyplot as plt
02 import NumPy as np
03
04 t = np.arange(0., 5., 0.2)
05
06 plt.plot(t, t, 'r--', label='Linear')
07 plt.plot(t, t**2, 'bs', label='Quadratic')
08 plt.plot(t, t**3, 'g^', label='Cubic')
09
10 plt.legend()
11 plt.title("Different rates of growth")
12 plt.savefig('basic-styling.png')
13 plt.show()
```

is hard to memorize because it has lots of options, but it's not cryptic once you know that the first letter is the color and the rest are a more or less symbolic alias for the line style: 'r--' creates a dashed red line, 'bs' creates another line marked with blue squares, and 'g^' creates a third line made of green triangles. The default format string is 'b-', which is a solid blue line. Lines 10 to 12 add the legend and title and save the plot to a file.

Line 13 of Listing 3

opens the chart in the graphical interface shown in Figure 2, which, at least on Ubuntu, comes with the standard Matplotlib package. In this interface, aside from modifying the styles, you can also zoom in, drag the figure around to look at a specific zone, change the size of any subplot, and save the results of your edits.

Figure 2: The basic Matplotlib interface for Linux lets you try out different formatting options.



Reading Data from a CSV File

You can use Matplotlib plots to visualize data created in another program and saved as a file, often as plain text in comma-separated values (CSV) format, such as `csv-data.csv` in Listing 4.

Figure 3 shows the data from the CSV file in Listing 4 in a histogram format. To create the histogram with Matplotlib, you can use the code in Listing 5 – after loading Matplotlib and another module (`csv`) to handle CSV data. Listing 5 creates two arrays, `x` and `y` (lines 4 and 5), and opens the data file (line 7). Lines 8-12 load the file's contents into a table called `plots` (using the comma as a column separator), copy the names from the first column into the `x` array and the ages (reformatted as integer numbers) into the `y` array. The rest of Listing 5 is self-explanatory, because each line corresponds to a specific, easy-to-spot visual feature in Figure 3.

A Random Data, Scattered Plot

While an in-depth overview of NumPy's capabilities is beyond the scope of this tutorial, Figure 4 gives an idea of the possibilities of using Matplotlib and NumPy together. The scatter plot in Figure 4 is generated from the code in Listing 6, which is taken (with minimal changes) from the official Matplotlib documentation. For my purposes, I will divide the code into two sections and discuss them in reverse order. The last section (lines 11-15) draws and saves a series of distinct points, scattered in random positions across the Axes. The coordinates, size, and

Listing 4: csv-data.csv

```
John, 32
Mary, 48
Ashok, 25
Mario, 41
Heather, 38
Ashley, 29
```

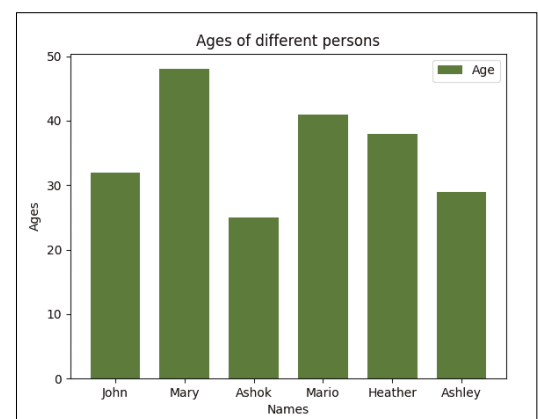


Figure 3: You can also plot data from a CSV file generated by another program.

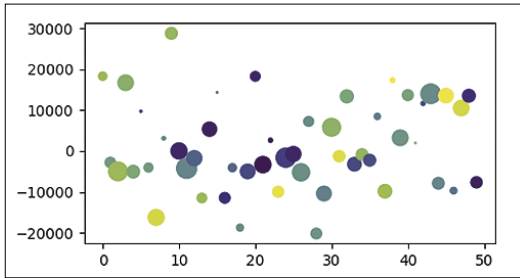


Figure 4: A scatter plot, showing the distribution and weights of a generic dataset.

color of each point are loaded in line 12 by the powerful pyplot function `scatter` (see [6] for an explanation of `scatter` syntax). The interesting part of Listing 6 is how all the arrays of values passed to `scatter` are generated. It is easy to guess, by looking at lines 4-9, that all those arrays are generated randomly, on the spot, with simple calls to the NumPy functions that generate random numbers. For the settings of those functions, please see the documentation for these functions on the NumPy website [4].

Subplots and Extra Axes

You can combine different plots into one Matplotlib Figure, add Axis elements, and link them to each other using the techniques in Listing 7.

Using more or less the same syntax shown in the previous examples, Listing 7 creates three distinct functions: a parabola (lines 5 and 8), a normal sinusoid (line 11), and a growing sinusoid (line 13). What is new is that these plots are grouped into two subplots (shown in Figure 5), whose Axis objects are indeed linked to each other. A full explanation of the code is beyond the scope of this article, but here are the main points you need to understand to create similar charts.

Line 7 splits the main Figure into two subplots, each with its own Axes, called `ax1` and `ax3`. Then, the first subplot gets the `ax2` of line 9, which is a “twin” of the `ax1` of line 8, for a very precise purpose: to show the very different scales of both the parabola and the sinusoid

Listing 5: Plotting Data from a CSV File

```
01 import Matplotlib.pyplot as plt
02 import csv
03
04 x = []
05 y = []
06
07 with open('csv-data.csv','r') as csvfile:
08     plots = csv.reader(csvfile, delimiter = ',')
09
10 for row in plots:
11     x.append(row[0])
12     y.append(int(row[1]))
13
14 plt.bar(x, y, color = 'g', width = 0.72, label = "Age")
15 plt.xlabel('Names')
16 plt.ylabel('Ages')
17 plt.title('Customers ages')
18 plt.legend()
19 plt.savefig('03-csv.png')
20 plt.show()
```

Listing 6: Scattered, Random Data

```
01 import Matplotlib.pyplot as plt
02 import NumPy as np
03
04 np.random.seed(20010911)
05 data = {'a': np.arange(50),
06         'c': np.random.randint(0, 50, 50),
07         'd': np.random.randn(50)}
08 data['b'] = data['a'] + 10000 * np.random.randn(50)
09 data['d'] = np.abs(data['d']) * 100
10
11 fig, ax = plt.subplots(figsize=(5, 2.7))
12 ax.scatter('a', 'b', c='c', s='d', data=data)
13 ax.set_xlabel('Speed')
14 ax.set_ylabel('Distance');
15 plt.savefig('random-scatter.png')
```

Listing 7: Connecting Subplots

```
01 import Matplotlib.pyplot as plt
02 import NumPy as np
03
04 t = np.arange(0.0, 40.0, 0.8)
05 s = t**2 #parabola
06
07 fig, (ax1, ax3) = plt.subplots(1, 2, figsize=(12, 5))
08 l1, = ax1.plot(t, s)
09 ax2 = ax1.twinx()
10 l2, = ax2.plot(t, np.cos(-1*t/2), 'C1')
11 ax2.legend([l1, l2], ['Parabola (left)', 'Sinusoid (right)'])
12
13 ax3.plot(t, 20 - t*np.sin(20 - t))
14 ax3.set_xlabel('Angle [°]')
15
16 ax4 = ax3.secondary_xaxis
17     ('top', functions=(np.rad2deg, np.deg2rad))
18 ax4.set_xlabel('Angle [rad]')
19 plt.savefig('extra-axes.png')
```

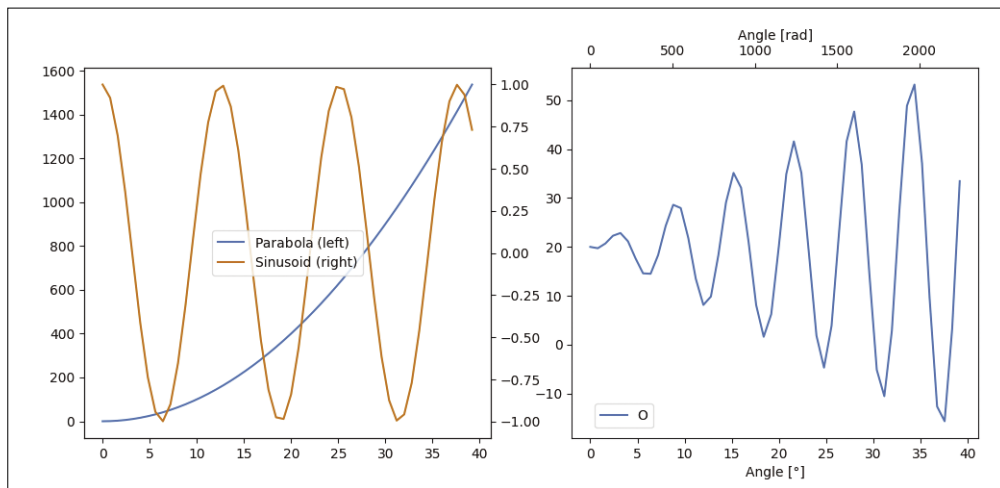


Figure 5: Plots can be combined in many ways with extra Axes that make them easier to read.

separately, but as clearly as possible, on opposite sides of the subplot. Meanwhile, the growing sinusoid is plotted in the right subplot, with its own labels and titles, but also using another feature of Matplotlib (line 16): the `secondary_xaxis` method creates an extra horizontal Axis (`ax4`) at the top of that subplot that shows the same variable as the bottom Axis (`ax3`) but with a different unit (radians instead of degrees).

Adding Text in Plots

You already know how to add legends to charts, but what about other text, maybe in nonstandard parts of a plot? Listing 8 shows the simplest way to add the captions and annotations shown in Figure 6.

Lines 11 and 13 of Listing 8 describe the function that generates the “local maximum” annotation shown in Figure 6. This syntax is relatively simple to describe: `plt.annotate` takes as

Listing 8: Inserting Text

```
01 import Matplotlib.pyplot as plt
02 import NumPy as np
03
04 t = np.arange(0., 50, 0.2)
05
06
07 plt.plot(t, (t**3)*np.cos(2*t))
08
09 plt.title("Combination of trigonometric and power factors")
10
11 plt.text(1, 80000, r'$Function: (t^\mu)*cos(\sigma*t), \text{ with } \mu = 3 \text{ and } \sigma = 2$')
12
13 plt.annotate('local maximum', xy=(19, 4100), xytext=(1, 45000),
14             arrowprops=dict(facecolor='red', shrink=0.05))
15
16 plt.savefig('text-in-plots.png')
```

arguments the string to insert, its starting position (`xytext`), the point where the arrow should end (`xy`), and its style (`arrowprops`). In line 11, `plt.text` has even fewer arguments: just the coordinates where a string should be inserted and then the string itself. The string is enclosed by single quotes and dollar signs and preceded by an `r` suffix, which means that what follows is “raw” text that uses LaTeX syntax and symbols (e.g., `\mu` and `\sigma`). Also note that in these

strings blank spaces must be escaped to tell Python that they are actual spaces that should just be printed instead of separators of function arguments.

Categorical Charts

Another place in a plot where you might want arbitrary text instead of numbers or dates is on its Axes. In Matplotlib, this type of diagram is called a categorical chart. Figure 7 shows a categorical chart of how cats and dogs consider certain activities.

Unless you have lots of categories to display or other special needs, plotting categorical charts is easier than it appears. Listing 9 only needs three arrays of strings and five lines of code to create Figure 7. The `activity` array lists all the activities, and the `cat` and `dog` arrays describe how those pets consider each activity (lines 3 to 5). The only critical part here is that each reaction of cats or dogs must be in the same position of the corresponding activity. That is, if `playing` is the fifth element of the `activity` array, and dogs are `SUPER-HAPPY` only when playing, then `SUPER-HAPPY` must be the fifth element of the `dog` array and the only element with the `SUPER-HAPPY` value. Then, all Matplotlib

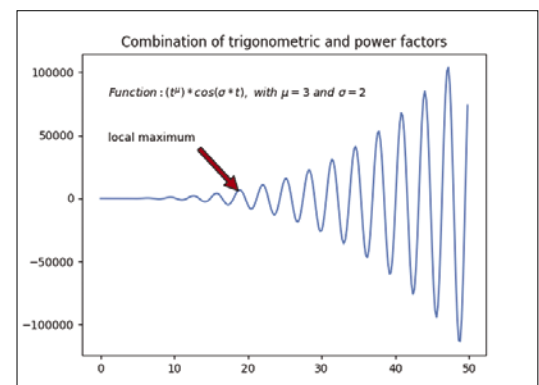


Figure 6: If labels and legends aren’t enough, you also can add captions and annotations where you want.

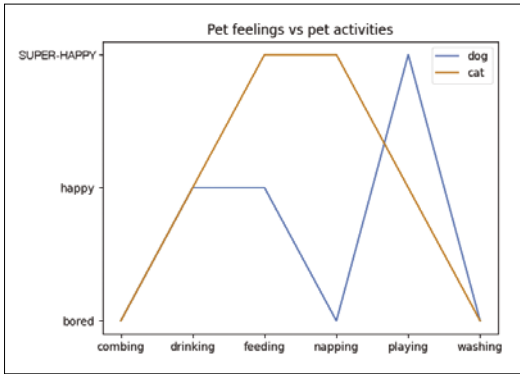


Figure 7: If they are properly categorized, charts can connect and display data of all kinds.

needs to plot everything correctly are the `ax.plot` methods shown in lines 8 and 9, which take the two arrays as the lists of x and y values to use to find all the points of each plot.

Sankey Diagrams

A Sankey diagram [7] represents the flow of certain variables in or out of a system by depicting each flow's width proportional to its quantity. While this type of diagram is not useful for everyone, a Sankey diagram showcases Matplotlib's flexibility and might also be a lifesaver for those in the fields of energy management, manufacturing, and science. Figure 8 and its code in Listing 10 are another example provided, but not completely explained, by the official Matplotlib documentation [8].

To draw a Sankey diagram, Matplotlib (or any other tool, for that matter) needs two things: a function, or object, that *knows* how to draw a Sankey diagram, and the parameters of each flow that enters or leaves the system. Listing 10 does just that by importing the Sankey function (line 2) and then telling Matplotlib to create and finish a Sankey object with the single long command in line 4. To do that, of course, each flow must be described by three parameters, namely the flow's intensity, orientation (i.e., if it enters or leaves the system), and an optional label. Lines

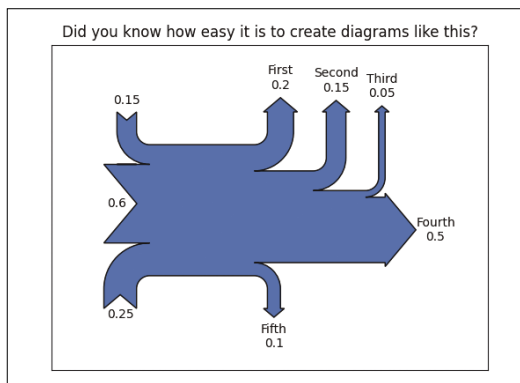


Figure 8: Sankey diagrams visualize input and output flows of materials, energy, or other quantities.

Listing 9: Categorical Chart

```
01 import Matplotlib.pyplot as plt
02
03 activity = ["combing", "drinking", "feeding", "napping", "playing",
             "washing"]
04 cat = ["bored", "happy", "SUPER-HAPPY", "SUPER-HAPPY", "happy", "bored"]
05 dog = ["bored", "happy", "happy", "bored", "SUPER-HAPPY", "bored"]
06
07 fig, ax = plt.subplots()
08 ax.plot(activity, dog, label="dog")
09 ax.plot(activity, cat, label="cat")
10 ax.legend()
11 plt.title("Pet feelings vs pet activities")
12 plt.savefig('categorical-names.png')
```

4 and 6 of Listing 10 pass this data passed to the Sankey object. As an example, the arrow pointing downward in Figure 8 (with a value of 0.1 and the label "Fifth") is the direct result of writing `-0.10, 'Fifth', and -1` in the last elements of the flows, labels, and orientation arrays (lines 4-6) passed to the Sankey object.

Polar Charts

You may be familiar with (and potentially loathe) the final Matplotlib example in this tutorial from performance evaluations: a polar diagram that resembles a radar screen. Figure 9 shows a polar chart with the scores achieved by one or more subjects in a series of activities (aka evaluations).

As in some other examples in this tutorial, the plotting part of Listing 11 (lines 11 to 19) is easier to explain than the portion that provides the data. Line 14 defines this diagram as `polar`, instead of using a Cartesian coordinate system like the other diagrams. Line 11 declares that there must be five labels (the length of the `superhero_1` array), located at equal intervals along the entire circumference (which is long pi times two), and line 19 says that the labels to place in those locations are in the `categories` array. Lines 15-17 add three distinct lines.

Listing 10: Sankey Diagram

```
01 import Matplotlib.pyplot as plt
02 from Matplotlib.sankey import Sankey
03
04 Sankey(flows=[0.25, 0.15, 0.60, -0.20, -0.15, -0.05, -0.50, -0.10],
        labels=['', '', '', 'First', 'Second', 'Third', 'Fourth',
               'Fifth'], orientations=[-1, 1, 0, 1, 1, 1, 0, -1]).finish()
05 plt.title("Did you know how easy it is to create diagrams like this?")
06 plt.savefig('sankey.png')
```

around the diagram, each with its own label, and the numeric values taken by the array of the corresponding superhero, defined in lines 7-9. As a

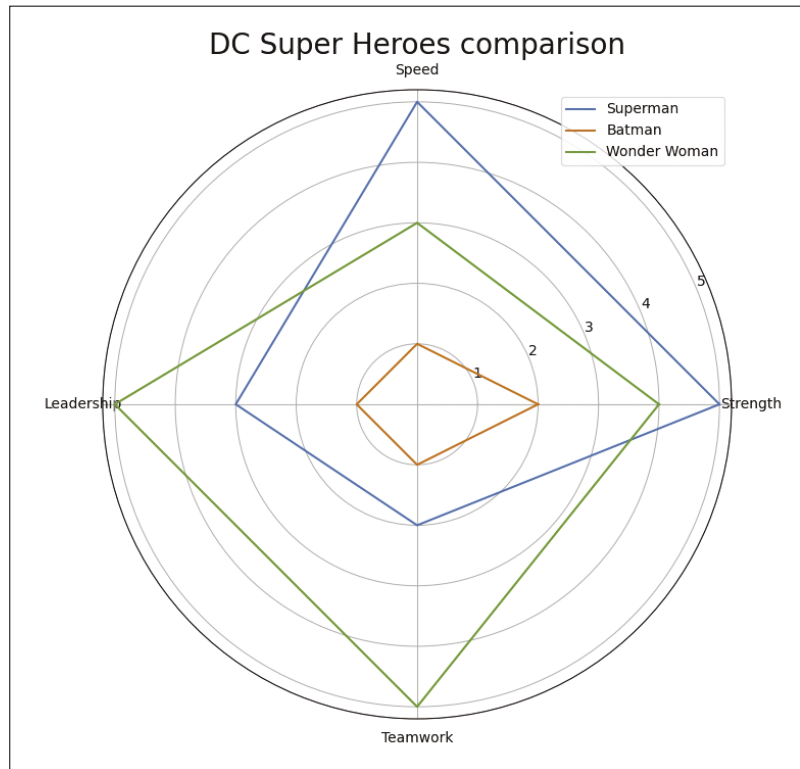


Figure 9: A superhero performance comparison using a polar diagram in Matplotlib.

Listing 11: Polar Comparison of DC Super Heroes

```
01 import NumPy as np
02 import Matplotlib.pyplot as plt
03
04
05 categories = ['Strength', 'Speed', 'Leadership', 'Teamwork', '']
06
07 superhero_1 = [5, 5, 3, 2, 5]
08 superhero_2 = [2, 1, 1, 1, 2]
09 superhero_3 = [4, 3, 5, 5, 4]
10
11 label_loc = np.linspace(start=0, stop=2 * np.pi, num=len(superhero_1))
12
13 plt.figure(figsize=(8, 8))
14 plt.subplot(polar=True)
15 plt.plot(label_loc, superhero_1, label='Superman')
16 plt.plot(label_loc, superhero_2, label='Batman')
17 plt.plot(label_loc, superhero_3, label='Wonder Woman')
18 plt.title('DC Super Heroes comparison', size=20)
19 lines, labels = plt.thetagrids(np.degrees(label_loc),
                                labels=categories)
20 plt.legend()
21 plt.savefig('../figures/radar.png')
```

result, Figure 9 shows that Superman has Strength=5 but Teamwork=2, Wonder Woman has Leadership=5, and so on.

The most important, but harder to notice, parts of Listing 11 are in lines 5 and 11: If you must only display four parameters (Strength, Speed, Leadership, and Teamwork), why do you need *five* locations for their labels and, correspondingly, a fifth, empty category ('')? The answer, which can easily be verified by altering those parameters, is that they are needed to “come full circle.” Without that extra empty category, each superhero’s line would not return to its initial point to close itself, resulting in an uglier diagram.

Conclusion

If you made it this far, you now know enough about Matplotlib to start creating many, wildly different charts and, above all, find your way around Matplotlib’s huge documentation without wasting too much time. Happy plotting! ■■■

Info

- [1] Matplotlib: <https://matplotlib.org>
- [2] Gnuplot: www.gnuplot.info
- [3] MATLAB: www.mathworks.com/products/matlab.html
- [4] NumPy: www.numpy.org
- [5] Pandas: <https://pandas.pydata.org>
- [6] Pyplot scatter syntax: https://matplotlib.org/3.1.1/api/_as_gen/matplotlib.pyplot.scatter.html
- [7] Sankey diagrams: www.ifu.com/e-sankey/sankey-diagram/
- [8] Sankey Matplotlib example: https://matplotlib.org/stable/gallery/specialty_plots/sankey_basics.html

The Author

Marco Fioretti (<http://mfioretti.com>) is a freelance author, trainer, and researcher based in Rome, Italy. He has been working with free/open source software since 1995 and on open digital standards since 2005. Marco also blogs about digital rights at <https://stop.zona-m.net>.



LINUX NEWSSTAND

Order online:
<https://bit.ly/Linux-Newsstand>

Linux Magazine is your guide to the world of Linux. Monthly issues are packed with advanced technical articles and tutorials you won't find anywhere else. Explore our full catalog of back issues for specific topics or to complete your collection.

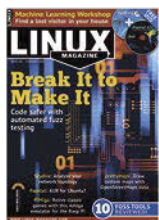


#256/March 2022

Facial Recognition

Biometrics got a boost recently with the arrival of Microsoft's Hello technology. Now the open source world is catching up, with an innovative tool appropriately called Howdy. Facial authentication might not be ready for the CIA yet, but we'll help you get started with Howdy and explore the possibilities of authenticating with a glance.

On the DVD: antiX 21 and Haiku R1/ Beta 3



#255/February 2022

Break It to Make It

Fuzz Testing: Ever wonder how attackers discover those "carefully crafted input strings" that crash programs and surrender control? Welcome to the world of fuzz testing. We introduce you to the art of fuzzing and explore some leading fuzz testing techniques.

On the DVD: Parrot OS 4.11 and Fedora Workstation 35



#254/January 2022

Phone Hacks

Eventually phone manufacturers just give up on supporting old hardware. If you're not ready to abandon that hardware yourself, you might find a better alternative with LineageOS — a free Android-based system that supports more than 300 phones, including many legacy models that are no longer supported by the vendor. We also explore PostmarketOS, a community-based Linux distribution that runs on several Android devices.

On the DVD: Ubuntu 21.10 and EndeavourOS 2021.08.27

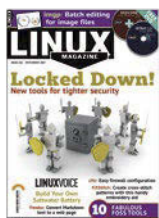


#253/December 2021

OpenBSD

BSD Unix has been around longer than Linux, and it still has a loyal following within the Free Software community. This month we explore the benefits of a leading BSD variant from the viewpoint of a Linux user.

On the DVD: Tails 4.22 and Q4OS 4.6



#252/November 2021

Locked Down!

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

On the DVD: Debian 11 and Redcore Linux 2101



#251/October 2021

Linux From Scratch

Building an operating system is not like compiling a desktop app. You'll need to create a complete development environment – and if you follow the steps carefully, you'll emerge with a deeper understanding of Linux.

On the DVD: Linux Mint 20.2 Cinnamon Edition and Garuda Linux KDE Dr460nized Edition

FEATURED EVENTS



Users, developers, and vendors meet at Linux events around the world. We at *Linux Magazine* are proud to sponsor the Featured Events shown here. For other events near you, check our extensive events calendar online at <https://www.linux-magazine.com/events>.

If you know of another Linux event you would like us to add to our calendar, please send a message with all the details to events@linux-magazine.com.

NOTICE

Be sure to check the event website before booking any travel, as many events are being canceled or converted to virtual events due to the effects of COVID-19.

DrupalCon Portland

Date: April 25-28, 2022

Location: Portland, Oregon

Website: <https://events.drupal.org/portland2022>

Come to the Drupal community's largest annual event – in person! Build your skills, learn more about what Drupal can do, and contribute to advancing the open source platform that organizations around the world rely on every day. This year's in-person event will be full of valuable insights, information, and connections.

Cloud Expo Europe Frankfurt

Date: May 11-12, 2022

Location: Frankfurt, Germany

Website: <https://www.cloudexpo-europe.de/en>

Cloud Expo Europe Frankfurt returns bigger and better than ever as a face-to-face event. Join us May 11-12 in Messe Frankfurt for cutting-edge content from leading end users. The program includes case studies and panel discussions on the latest topics related to cloud, cyber security, and DevOps.

Events

Storage Developer Conference	April 5	Virtual Conference	https://www.snia.org/events/sdcemea
Cephalocon 2022	April 5-7	Portland, Oregon + Virtual	https://events.linuxfoundation.org/
ODSC East 2022	April 19-21	Boston, MA + Virtual	https://odsc.com/boston/
LinuxFest Northwest 2022	April 22-24	Virtual Event	https://lfnw.org/conferences/2022
DrupalCon Portland 2022	April 25-28	Portland, Oregon	https://events.drupal.org/portland2022
Linux App Summit	April 29-30	Rovereto, Italy + Virtual	https://linuxappsummit.org/
Linux Storage, Filesystem, MM & BPF Summit	May 2-4	Palm Springs, California	https://events.linuxfoundation.org/lfsfmm/
Cloud Expo Europe Frankfurt	May 11-12	Frankfurt, Germany	https://www.cloudexpo-europe.de/en
The Open Source Infrastructure Conference	May 16-17	Berlin, Germany	https://stackconf.eu/
KubeCon + CloudNativeCon Europe 2022	May 16-20	Valencia, Spain	https://events.linuxfoundation.org/
ISC High Performance 2022	May 29-June 2	Hamburg, Germany	https://www.isc-hpc.com/
OpenJS World 2022	June 6-10	Austin, Texas	https://events.linuxfoundation.org/openjs-world/
cdCon	June 7-8	Austin, Texas + Virtual	https://events.linuxfoundation.org/cdcon/
ODSC Europe 2022 London	June 15-16	United Kingdom + Virtual	https://odsc.com/europe/
ITEXPO Florida	June 21-24	Fort Lauderdale, Florida	https://www.itexpo.com/east/
Open Source Summit North America	June 21-24	Austin, Texas + Virtual	https://events.linuxfoundation.org/
Xen Developer & Design Summit	June 28-30	Bucharest, Romania + Virtual	https://events.linuxfoundation.org/

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 Pettie, Aubrey Vaughn

News Editor

Jack Wallen

Editor Emerita Nomadica

Rita L Sooby

Managing Editor

Lori White

Localization & Translation

Ian Travis

Layout

Dena Friesen, Lori White

Cover Design

Lori White

Cover Image

© drizzd, 123RF.com

Advertising

Brian Osborn, bosborn@linuxnewmedia.com
phone +49 8093 7679420

Marketing Communications

Gwen Clark, gclark@linuxnewmedia.com
Linux New Media USA, LLC
4840 Bob Billings Parkway, Ste 104
Lawrence, KS 66049 USA

Publisher

Brian Osborn

Customer Service / Subscription

For USA and Canada:
Email: cs@linuxpromagazine.com
Phone: 1-866-247-2802
(Toll Free from the US and Canada)

For all other countries:
Email: subs@linux-magazine.com

www.linuxpromagazine.com – North America

www.linux-magazine.com – Worldwide

While every care has been taken in the content of the magazine, the publishers cannot be held responsible for the accuracy of the information contained within it or any consequences arising from the use of it. The use of the disc provided with the magazine or any material provided on it is at your own risk.

Copyright and Trademarks © 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 hofmann infocom GmbH.

Distributed by Seymour Distribution Ltd, United Kingdom

LINUX PRO MAGAZINE (ISSN 1752-9050) is published monthly by Linux New Media USA, LLC, 4840 Bob Billings Parkway, Ste 104, Lawrence, KS 66049, USA. Periodicals Postage paid at Lawrence, KS and additional mailing offices. Ride-Along Enclosed. POSTMASTER: Please send address changes to Linux Pro Magazine, 4840 Bob Billings Parkway, Ste 104, Lawrence, KS 66049, USA.

Published monthly in Europe as Linux Magazine (ISSN 1471-5678) by: Sparkhaus Media GmbH, Bialasstr. 1a, 85625 Glonn, Germany.

Authors

Zack Brown	12
Bruce Byfield	6, 34, 44, 58
Joe Casad	3
Mark Crutch	73
Adam Dix	76
Marco Fioretti	88
Claudius Grieger	38
Jon "maddog" Hall	75
Peer Heinlein	22
Martin Hirschvogel	47
Charly Kühnast	46
Martin Gerhard Loschwitz	16
Vincent Mealing	73
Martin Mohr	68
Graham Morrison	80
Mike Schilli	52
Dominique Schröder	28
Tim Schürmann	62
Jack Wallen	8

Issue 258 / May 2022

Green IT

Reducing energy usage saves money and helps to save the planet. Next month we study some recent developments in the quest for more energy-efficient computing.

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>

Approximate

UK / Europe

Apr 02

USA / Canada

Apr 29

Australia

May 30

On Sale Date

Please note: On sale dates are approximate and may be delayed because of logistical issues.



Image © frenta, 123RF.com

ISC IS BACK IN PERSON!

TRANSFORMING

THE FUTURE

CONFERENCE

PARALLEL PROGRAMMING

SYSTEM ARCHITECTURE

WORKSHOPS

MACHINE LEARNING

EXHIBITION

QUANTUM COMPUTING

APPLICATIONS & ALGORITHMS

AND MORE...

TUTORIALS

Come join 3,000 high performance computing (HPC) practitioners, vendors, and enthusiasts.

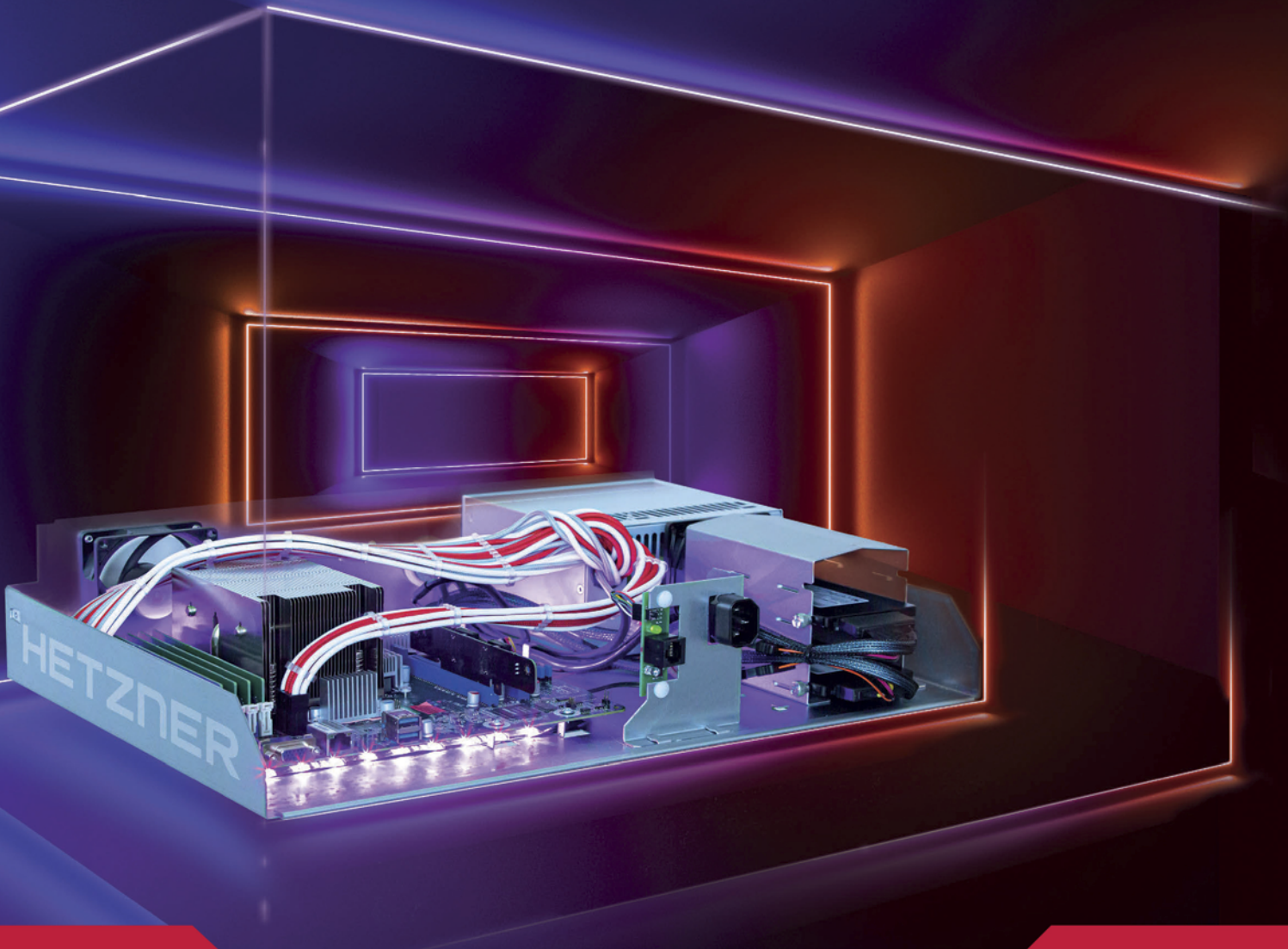
WHAT TO EXPECT?

- Technical program (in person and online)
- HPC exhibition (in person and online)
- Reunion with peers, friends, and collaborators
- Safe and health-protocol-compliant environment

After two years of an online presence ISC returns to you as an in-person event. If you're involved or interested in HPC, don't miss out on making valuable connections at ISC 2022.

HETZNER

DEDICATED ROOT SERVER DESIGNED FOR PROFESSIONALS



Dedicated Root Server AX41-NVMe

- ✓ AMD Ryzen™ 5 3600
Simultaneous Multithreading
- ✓ 64 GB DDR4 RAM
- ✓ 2 x 512 GB NVMe SSD
- ✓ 100 GB Backup Space
- ✓ Traffic unlimited
- ✓ Location Finland and Germany
- ✓ No minimum contract



monthly from \$ **39**

Setup Fee \$ **0**

Dedicated Root Server AX51-NVMe

- ✓ AMD Ryzen™ 7 3700X
Simultaneous Multithreading
- ✓ 64 GB DDR4 ECC RAM
- ✓ 2 x 1 TB NVMe SSD
- ✓ 100 GB Backup Space
- ✓ Traffic unlimited
- ✓ Location Finland and Germany
- ✓ No minimum contract



monthly from \$ **62**

Setup Fee \$ **0**

All prices exclude VAT and are subject to the terms and conditions of Hetzner Online GmbH. Prices are subject to change. All rights reserved by the respective manufacturers.

www.hetzner.com