LINUX MAGAZINE

Wildlife Watcher

Find out who's visiting your garden with a Rasp Pi photo trap

FREE DVD

LINUX MAGAZINE

ISSUE 270 – MAY 2023

Green Coding

Techniques for more sustainable software

The MIPI Debacle Why doesn't my webcam work?

Tapping the ChatGPT API

Coreboot A better solution for firmware

ELF Anatomy: Inside an executable file

Sniffnet: Intuitive network monitoring tool

10 Tantalizing Tools for eager FOSS appetites!

LINUX NEW MEDIA
The Pulse of Open Source

WWW.LINUX-MAGAZINE.COM

Linux installation

With the WebFAI you can set up your Linux installation fully automated and exactly as fully functional as it is done in our production at TUXEDO Computers!

Linux distribution

You can choose from all Linux distributions that we also offer for selection in our online store configurator. Depending on the distribution, you have the choice between different desktop environments.

always up-to-date

With the WebFAI you receive our latest, tested Linux distribution versions from our servers.

TUXEDO **WebFAI**

Fully Automated Installation

100% Linux

5 Year Warranty

Lifetime Support

Built in Germany

Germany Privacy

Local Support

TUXEDO

tuxedocomputers.com

BETTER BOUNDARIES

Dear Reader,

Is the whole high-tech scene imploding? If so, it won't be the first time, but every time it implodes, it comes back a little different, so even temporary implosions are relevant. The layoffs continue, with more pink slips circulating through the tech giants. Meta announced another round. Microsoft sent away the remnants of its ethics and society team. Members of both parties in the fractious US Congress are clamoring to reign in the power of big tech. Where will all this lead? At least for this news cycle, all eyes are on Amazon.

The number of ongoing investigations into Amazon practices is really quite extraordinary [1]. The probes include antitrust, privacy violations, deceptive advertising, and so-called dark patterns (manipulative tricks played within a user interface to make the user spend money, sign up, or stay signed up when they want to quit). Of course, in such cases, the subject is always innocent until proven guilty; however, given the current state of public opinion, the pressure is for the government to make something stick.

The antitrust issues are the most uncertain, and they could have the biggest implications. Amazon has been under scrutiny for years with its massive share of the online retail market. What is it? A free market for all products, or a captive market for its own products? The problem is, Amazon is both, which isn't illegal, but if they use their power to preference their own products over others, it could mean trouble. The US Federal Trade Commission (FTC) is currently reviewing whether to nix Amazon's recent purchase of iRobot, a company that makes robot vacuum cleaners that are sold over Amazon's platform, as well as its purchase of One Medical.

Privacy investigations are also nothing new for Amazon. According to reports, current investigations include a look at the Ring doorbell camera, which tells you who is at your front door (and also, allegedly, tells Amazon who is at your front door). And Alexa, that cute little desktop assistant that plays favorite pop songs on demand? She is under scrutiny for alleged violations of the Children's Online Privacy Protection Act.

An investigation from the FTC is not exactly like a lawsuit from the US Department of Justice Antitrust Division, but it could signal a more hawkish stance regarding the venerable tech tradition of big fish getting bigger by swallowing little fish. On the other hand, the Antitrust Division seems to be on the trail also.

Amazon's reach into our lives is truly breathtaking. From the retail items you buy online, to the colossal Amazon Web Services business, to Amazon Prime Video and Music services. The company also owns lots of other companies and products you don't even know about, and they even recently purchased MGM Studios. Their massive size and penetration into so many different markets makes it quite difficult to keep an eye on them. Businesses, in general, often push the limits to maximize profit. Ideally, they don't break the law, but sometimes, especially in industries that are still relatively new, the distinction between pushing the limits and breaking the law is not always clear. It is up to the government to define where the limits are, and that can take lots of energy, resources, and patience. These kinds of investigations help to stake out those limits, and in the long run, that could be good for Amazon and the industry in general, because clearer boundaries lead to better behavior – and ultimately, better protection from future investigations.

Joe

Joe Casad,
Editor in Chief

Info

[1] "Washington Prepares for War with Amazon": *https://www.politico.com/news/2023/03/20/ftc-amazon-irobot-antitrust-00087711*

LINUX MAGAZINE
MAY 2023

ON THE COVER

24 Coreboot
Boot faster with this innovative firmware alternative.

32 Sniffnet
You don't have to be an expert to analyze network traffic.

36 MIPI Cameras
Linux, MIPI cameras, and the problem with Intel's IPU6 image processing unit.

44 ELF File Format
Explore the structure of a Linux executable file.

54 ChatGPT
Build the power of AI chat into your homegrown scripts with the ChatGPT API.

63 Wildlife View
Rig up a photo trap using a motion sensor and a RaspPi Zero.

NEWS

8 News
- GNOME 44 Release Candidate Now Available
- Flathub Vying to Become the Standard Linux App Store
- Debian 12 to Ship with KDE Plasma 5.27
- Planet Computers Launches ARM-Based Linux Desktop PCs
- Ubuntu No Longer Shipping with Flatpak
- openSUSE Leap 15.5 Beta Available
- Linux Kernel 6.2 Released with New Hardware Support
- Kubuntu Focus Team Releases New Mini Desktop
- US National Cybersecurity Strategy Released

12 Kernel News
- Reader Reactions
- Tracking Up-to-Date Kernel Documentation
- Linux in Carspace
- Filesystems and Cryptography

COVER STORIES

16 Green Coding Initiatives
A number of open source projects offer tools for analyzing energy usage in software.

18 Energy-Efficient Software
Go has a reputation for producing energy-saving applications, but you still have to know what you are doing.

@linux_pro
Linux Magazine
@linuxpromagazine
@linuxmagazine

REVIEWS

24 Distro Walk – coreboot
Coreboot is an open source firmware alternative with an emphasis on speed and simplicity.

28 TUXEDO OS 2
The popular Linux PC forge TUXEDO extends Ubuntu for its in-house TUXEDO OS distribution.

IN-DEPTH

32 Sniffnet
Network traffic remains a closed book for many users. Sniffnet lets less experienced users monitor their network traffic with ease.

36 MIPI Cameras and Linux
Recently, things have gotten dicey for webcams supporting the MIPI specification. We'll tell you why and what to do about it.

42 dict
With the dict client, you can quickly search dozens of natural language dictionary databases for the perfect word.

44 ELF File Format
Linux and other Unix-based systems use the ELF file format for executables, object code, and shared libraries. Take a peek inside to learn how an ELF file is organized.

50 Systemd Flatpak Updates
You can automate Flatpak updates without a package manager using systemd's services and timers.

Green Coding

A sustainable world will need more sustainable programming. This month we tell you about some FOSS initiatives dedicated to energy efficiency, and we take a close look at some green coding techniques in Go.

IN-DEPTH

54 Programming Snapshot – ChatGPT
The electronic brain behind ChatGPT is amazingly capable when it comes to chatting with human partners. Mike Schilli picks up an API token and uses it to code some small practical applications.

60 nftables
Filter rules for firewalls can be tricky. As the successor to iptables, nftables simplifies the process of creating and maintaining firewall rules.

MakerSpace

63 Wildlife View
Armed with no more than a Raspberry Pi photo trap, you can discover who pays a visit to your garden at night.

67 Whiptail
Add menus and information pages to your Raspberry Pi projects.

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

LINUXVOICE

71 Welcome
This month in Linux Voice.

72 Doghouse – Old Farts
A new effort to record the history of open source is underway.

73 LinkAce/floccus
Synchronize and manage bookmarks while storing your data locally.

80 FSearch
In a crowded field of search applications, FSearch offers many interesting functions for quickly searching files and folders.

84 FOSSPicks
This month Graham looks at KiCad 7, Carbonyl, CliFM, markdownlint, MythTV 33.1, zelda3, and more.

90 Tutorial – ddgr
Since 2008, DuckDuckGo has been making waves as an efficient and much more private search engine alternative to Google. The unaffiliated command-line tool ddgr makes it easy to run DuckDuckGo searches from the terminal.

Fedora 37 Workstation and TUXEDO OS 2

Two Terrific Distros on a Double-Sided DVD!

Fedora 37 Workstation

64-bit

Fedora is a Red Hat-maintained community distro that is known for early adoption of new software components. The Fedora Workstation edition is focused on desktop operations in a professional setting. The Fedora 37 release includes the Gnome 43 desktop, which features a redesigned system status menu and a major upgrade for the Files file manager. The latest Fedora also comes with updates to the support for human languages, as well as key updates to computer language components, including Python, glibc, and LLVM.

TUXEDO OS 2

64-bit

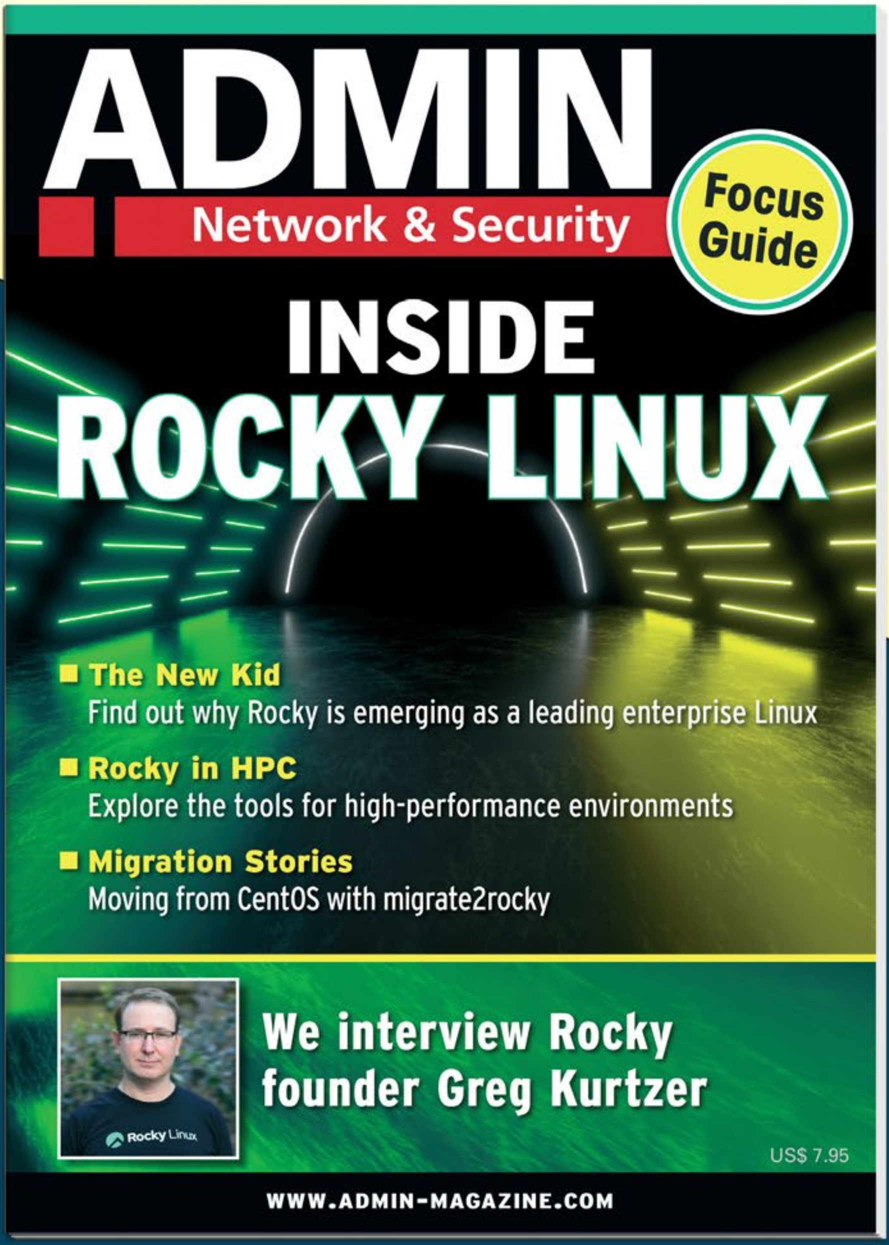
TUXEDO OS is an Ubuntu-based Linux variant maintained by the European hardware vendor TUXEDO for use with their native Linux laptop systems. TUXEDO unveiled an ISO for general purpose Linux systems back in September, and they have also made their new TUXEDO OS 2 release available to the public as a slim and responsive desktop system. TUXEDO OS 2 is based on Ubuntu 22.04. Users who choose TUXEDO OS over Ubuntu prefer the selection of native applications, the Flatpack package management system, and the attention to performance. In our tests, TUXEDO OS booted significantly faster than Ubuntu or Windows (see the article elsewhere in 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.

Go In-Depth with
Rocky Linux

Download this free focus guide, and learn why Rocky Linux is gaining a wide audience with users around the world.



ADMIN
Network & Security
Focus Guide

INSIDE
ROCKY LINUX

- **The New Kid**
  Find out why Rocky is emerging as a leading enterprise Linux
- **Rocky in HPC**
  Explore the tools for high-performance environments
- **Migration Stories**
  Moving from CentOS with migrate2rocky

We interview Rocky founder Greg Kurtzer

US$ 7.95

WWW.ADMIN-MAGAZINE.COM

Download free!

https://bit.ly/Inside-Rocky-Linux

NEWS

Updates on technologies, trends, and tools

THIS MONTH'S NEWS

08
- Gnome 44 Release Candidate Now Available
- Flathub Vying to Become the Standard Linux App Store

09
- Debian 12 to Ship with KDE Plasma 5.27
- Planet Computers Launches ARM-Based Linux Desktop PCs
- More Online

10
- Ubuntu No Longer Shipping with Flatpak
- openSUSE Leap 15.5 Beta Available
- Linux Kernel 6.2 Released with New Hardware Support

11
- Kubuntu Focus Team Releases New Mini Desktop
- US National Cybersecurity Strategy Released

Gnome 44 Release Candidate Now Available

Gnome 44 is upon us. Many Gnome fans have tested the beta version and found it to be the perfect next step for the open source desktop environment. And with the projected release, at the time of writing, of March 22, 2023, this release candidate arrives at the perfect time.

Surprisingly, however, the development team has added a few changes to the desktop. No, these are not new features but more bug fixes and cleanups.

For example, the team has added a bit more polish to the Epiphany web browser. There is also direct support for Wayland's `fractional_scale_v1`. What this means is that you'll be able to set fractional scaling for multiple monitor setups.

You'll also find new options in the Quick Settings menu, such as a list of currently connected Bluetooth devices. As well, the Quick Settings menu will now include a *Screenshot* button and a background apps list.

The expanding folders feature was removed in Gnome 43, but, thanks to a demanding community, it was added back for the Gnome 44 release candidate.

Other changes include: The Gnome Settings Daemon goes to sleep after 15 minutes of inactivity, the Gnome Control Center mouse and touchpad section has been slightly revamped, a good amount of code cleanup was done for the GDM login manager, Gnome Boxes now allows the creation of virtual machines without first selecting an operating system info entry, and Gnome Music now uses less memory when running in the "Songs" view.

Read more about the Gnome 44 release candidate here: *https://discourse.gnome.org/t/gnome-44-rc-released/14444*.

Flathub Vying to Become the Standard Linux App Store

Linux has a plethora of package managers and app stores. There's Apt, DNF, Yum, Zypper, Pacman, Gnome Software, Discover, and Synaptic.

For modern Linux distributions, however, you can also add Snap and Flatpak into the mix. Those last two have, for some time, struggled to gain much traction. However, over the past couple of years, those universal package managers have finally gained considerable popularity.

But only one of those tools is vying to become the de facto standard app store for Linux.

Flatpak has secured $100,000 in funding and is aiming for $150,000 more. Their goal is to prepare for higher operating costs and to bring in another full-time staffer.

This comes on the heels of Canonical announcing all official Ubuntu spin-offs will no longer ship with Flatpak installed by default.

Recently, Robert McQueen (Endless CEO and president of the Gnome board) penned a blog (*https://ramcq.net/2023/03/07/flathub-in-2023/*) to not only state how strong the Flatpak growth is (with more than 700,000 app downloads a day) but to state that they plan on establishing an independent legal entity to own and operate Flathub (which is currently hosted by Gnome).

McQueen says in his blog, "Flatpak has, in my opinion, solved the largest technical issue which has held back the mainstream growth and acceptance of Linux on the desktop … namely, the difficulty for app developers to publish their work in a way that makes it easy for people to discover, download (or sideload, for people in challenging connectivity environments), install and use." He adds, "Flathub builds on that to help users discover the work of app developers and helps that work reach users in a timely manner."

As far as what the future holds, the plan is to launch a new Flathub web experience, add verification features, turn on Flatpak repo subsets to enable users to select only verified and/or Flos apps, enable direct app uploads as well as donations and payments, and create Flatpak focus groups and an advisory board.

Debian 12 to Ship with KDE Plasma 5.27

In a break from tradition, Debian is opting to go with the newest version of the KDE Plasma desktop for the testing branch. This goes against their previous method of exhaustive testing before a piece of software is released with a distribution.

Debian is well known for shipping applications that seem to be out of date. The reason behind this is that the focus of the "mother of all" distributions is primarily on stability. To that end, the developers have always ensured each release includes only software that can be trusted to run without problems.

For those who prefer to use more up-to-date software, you can always install it manually, but in some instances – such as with a desktop environment – that can get a bit tricky.

So, if you prefer Debian with an up-to-date KDE Plasma desktop environment, you'll be able to enjoy that very thing by using the Debian testing branch.

That version is Debian 12 (Bookworm). And given that Bookworm is already in a soft freeze state, it's pretty certain KDE Plasma 5.27 will wind up being the official release once Bookworm is out of testing.

KDE Plasma 5.27 includes the new tiling window manager, a much-improved Discover app store, tons of bug fixes, and top-to-bottom refinements of the desktop. This release also ships with KDE Frameworks 5.103 and KDE Gear 22.12.3.

To find out more about the Debian 12 release, read the official release plan (*https://wiki.debian.org/PkgQtKde/BookwormReleasePlans*). At the time of writing, Bookworm is scheduled to hit the hard freeze state around March 12. The full freeze and release dates have yet to be announced.

Planet Computers Launches ARM-Based Linux Desktop PCs

Planet Computers has brought to market a new line of mini Linux desktop computers. The XR series is powered by ARM multicore processors and runs Ubuntu Linux 20.04.

According to Janko Mrsic-Flogel, CEO of Planet Computers, "By productizing Linux into a mini desktop PC, the XR is another step toward creating a more open and inclusive computing ecosystem."

The XR series will launch with two different units: XR1 (4-core CPU, 4GB of RAM, and 32 GB of internal storage) and XR2 (8-core CPU, a Mali-G610 GPU, from 4GB to 32GB of RAM, and from 32GB to 256GB of internal storage).

Both options offer dual gig Ethernet and WiFi 6, as well as built-in Internet relay support, 8K video encoding/decoding, and both 4k and 8k output. There is zero configuration necessary and ports include 2 x 1GB Ethernet, Bluetooth 5.0, 1 HDMI video in/2 HDMI video out, 4 x USB 4.0, 1 x USB 2.0, 1 x USB-C 3.0, 1 x USB-C 2.0, a headphone jack, and a microSD + SIM slot.

Both devices include a color touchscreen that gives you access to different configuration options, the time/date, and even the *Power* button.

You can pre-order the XR series now on the Planet Computer site (*https://store.planetcom.co.uk/*). The XR1 sells for $614.00 and the XR2 for $774.00.

MORE ONLINE

Linux Magazine

www.linux-magazine.com

ADMIN HPC

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

Warewulf 4 – Environment Modules
• Jeff Layton
Environment Modules are an indispensable tool for high-performance computing that allow you to switch between and among versions of compilers, libraries, and applications.

ADMIN Online

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

Encrypt and Decrypt Files with Age or Rage
• Matthias Wübbeling
Age and Rage are the Go and Rust implementations of a simple, modern, and secure file encryption tool.

Quick UDP Internet Connections
• Benjamin Pfister
The UDP-based Quick UDP Internet Connections (QUIC) protocol comes with mandatory TLS encryption and promises faster speeds.

Diving into Infrastructure Security
• Matthias Wübbeling
How to deal with threat intelligence on the corporate network when the existing security tools are not effective.

Ubuntu No Longer Shipping with Flatpak

Anyone who has followed the rise of universal package managers such as Snap and Flatpak is fully aware of the difference between the two technologies and how one is more distribution agnostic than the other.

I'm talking about Snap, which is driven by Canonical, the company that happens to be behind Ubuntu.

Both Snap and Flatpak are great methods for installing software, especially proprietary applications that do not have ports to the more traditional package managers such as Apt and DNF. Snap and Flatpak make the installation of tools such as Spotify, Slack, and Skype considerably easier.

Snap comes pre-installed with Ubuntu, the official Ubuntu spins, and many Ubuntu-based distributions. As well, some official spins have also included Flatpak. The combination of Apt, Snap, and Flatpak meant users had a plethora of applications to install.

Canonical has made the decision that Ubuntu spins are no longer allowed to ship with Flatpak installed (*https://discourse.ubuntu.com/t/ubuntu-flavor-packaging-defaults/34061*). You can still install Flatpak on Ubuntu and any one of its spins from the standard repositories with the command `sudo apt-get install flatpak -y`.

This change affects all Ubuntu spins, including Kubuntu, Lubuntu, Ubuntu Budgie, Ubuntu Kylin, Ubuntu MATE, Ubuntu Studio, Ubuntu Unity, and Xubuntu.

openSUSE Leap 15.5 Beta Available

You might not be surprised that the final release in the openSUSE Leap 15 series doesn't include a lot of new features. In fact, this new beta release is quite bereft in the new features category. As expected, Leap 15.5 is really just about bug fixes and new versions of the installed applications.

However, tucked inside the list of new packages, you will find the Leap version migration tool, which makes it possible to migrate from one release to another with the ease of a GUI.

The tool has been streamlined such that moving to the next available release can be done with a single click. So, if you're currently using a previous openSUSE Leap release (or are on the 15.5 Alpha), you can migrate to 15.5 with ease.

Some of the new package releases for openSUSE Leap 15.5 include Python 3.10, mdadm 4.2, KDE Plasma 5.27, Linux kernel 5.14.21, and the Mesa 22.3 graphics stack.

OpenSUSE Leap 15 began back in 2018, so it's had a pretty good run. For anyone who wishes to go with the latest, greatest version of openSUSE, consider the rolling release candidate, Tumbleweed (*https://get.opensuse.org/tumbleweed/*).

However, for those who prefer a more traditional release cycle, Leap is the way to go.

Download the beta version of openSUSE Leap (*https://get.opensuse.org/leap/15.5/*) and read the full blog post about the release (*https://news.opensuse.org/2023/02/21/leap-reaches-beta-phase/*).

Linux Kernel 6.2 Released with New Hardware Support

The latest, greatest Linux kernel has arrived and it includes a number of improvements for hardware support, performance, and security.

In a recent email to the LKML mailing list (*https://lkml.org/lkml/2023/2/19/309*), Linus Torvalds had this to say: "But in the meantime, please do give 6.2 a testing. Maybe it's not a sexy LTS release like 6.1 ended up being, but all those regular pedestrian kernels want some test love too."

Although 6.2 might not be "sexy," it still has plenty to offer. First up are hardware improvements that include out-of-the-box support for Intel Arc graphics and Intel's On Demand driver. As well, Skylake CPUs have gained a significant performance increase with an addition that's designed to address the Retbleed CPU vulnerability.

Linux 6.2 is also the first mainline kernel with support for Apple Silicon M1 Pro, Max, and Ultra. This particular work was upstreamed by the Asahi Linux team.

Get the latest news in your inbox every two weeks

Subscribe FREE to Linux Update

bit.ly/Linux-Update

The NTFS3 driver now includes the `hidedotfiles` option (to properly hide hidden files when mounting Windows directories), `nocase` (for case-insensitive files and folders), and `windows_names` (which prevents renaming files and folders with names not allowed in Windows).

Support also has been added for the Sony Dualshock 4 game controllers, improved support for the OneXPlayer handheld gaming console, support for the Habana Labs Gaudi2 AI accelerator, and a number of improvements for ASUS motherboards.

To find out more of what's gone into the 6.2 kernel, read parts one and two of the 6.2 Merge Window (part 1: *https://lwn.net/Articles/917733/*, part 2: *https://lwn.net/Articles/918146/*).

Kubuntu Focus Team Releases New Mini Desktop

The Focus NX GEN 2 is a small form factor desktop PC that comes with Kubuntu 22.04 pre-installed and offers plenty of power to help you be productive and entertained.

The power behind this tiny computer shouldn't surprise you, given the specs. The NX GEN 2 ships with a 12th-gen Intel Core i7-1260P CPU with 16 cores (running at 2.8GHz with a 4.7GHz turbo boost), configurable up to 64GB of DDR4-3200 RAM, and can house up to a 2TB NVMe boot SSD and a 4TB secondary SATA drive.

The rest of the specs include 10th-gen Iris Xe graphics, Intel Wi-Fi 6E AX211 Gig+, Bluetooth 5.3, 3 USB-A 3.2 ports, 1 USB-A 2.0 port, 1 audio/mic combo jack, 2 Thunderbolt 4 ports, 2 HDMI 2.1 ports, 1 RJ-45 jack, and a VESA monitor-mount bracket.

The NX GEN 2 has also been tested and configured for expansion with an eGPU. However, the built-in graphics should suffice for most use cases.

The pricing for the Focus NX GEN 2 starts at $995.00 for 8GB of RAM and a 250GB NVMe SSD. You can configure and purchase your Focus NX GEN 2 on the official Kubuntu Focus site (*https://kfocus.org/order/order-nx.html*).

US National Cybersecurity Strategy Released

The Biden administration has released a new National Cybersecurity Strategy (*https://www.whitehouse.gov/wp-content/uploads/2023/03/National-Cybersecurity-Strategy-2023.pdf*) "to secure the full benefits of a safe and secure digital ecosystem for all Americans." The 35-page document outlines the following "fundamental shifts" in how the United States approaches cybersecurity:

- We must rebalance the responsibility to defend cyberspace by shifting the burden for cybersecurity away from individuals, small businesses, and local governments, and onto the organizations that are most capable and best-positioned to reduce risks for all of us.
- We must realign incentives to favor long-term investments by striking a careful balance between defending ourselves against urgent threats today and simultaneously strategically planning for and investing in a resilient future.

The strategy also recognizes the current complex threat environment and emphasizes the need to build collaboration around the following five pillars:

- Defend critical infrastructure
- Disrupt and dismantle threat actors
- Shape market forces to drive security and resilience
- Invest in a resilient future
- Forge international partnerships to pursue shared goals

The document notes that "this Administration is also committed to improving Federal cybersecurity through long-term efforts to implement a zero-trust architecture (*https://www.fosslife.org/what-does-zero-trust-mean*) strategy and modernize IT and OT infrastructure."

See the official fact sheet for a summary (*https://www.whitehouse.gov/briefing-room/statements-releases/2023/03/02/fact-sheet-biden-harris-administration-announces-national-cybersecurity-strategy/*).

Zack's Kernel News

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

By Zack Brown

Reader Reactions

Recently, I covered a discussion in which the kernel developers talked about Spectre and Retbleed security issues cropping up on 32-bit kernels running on 64-bit hardware. The upshot was that by and large the developers seemed to feel that users shouldn't do this, and therefore there was no real need for the developers to worry too much about that specific case.

In response to that column, I received the following letter from Gary Dale, which articulated a point of view the kernel folks may not have considered. I'm including the letter here, with permission. Gary told me:

"I believe the kernel developers severely underestimate how many people may be running 32bit kernels on 64bit hardware. I, and a lot of other Raspberry Pi owners, are doing just that for a variety of reasons.

"In my particular case, I use my Pi 4B to run Kodi. However accessing Netflix, Prime and other streaming services requires the widevine library which only seems to exist in a 32bit version. Unfortunately, 32bit Kodi wouldn't install under multiarch. The 64bit Raspberry Pi OS is noticeably faster than the 32bit version but I'm stuck with the 32bit version due to digital restrictions.

"It can be argued that this is an artifact of the newness of 64bit Raspberry Pi OS and the ARM architecture. However this is just one case where it is necessary to run an OS that doesn't match the architecture. The kernel developers should probably be less dismissive of the potential problems."

I was really glad to get Gary's letter. If other readers start sharing thoughtful responses like this one, maybe "Reader Reactions" will become a regular part of this column. Reader reactions can always be sent to *edit@linux-magazine.com*. I read them all.

Tracking Up-to-Date Kernel Documentation

Carlos Bilbao recently took over maintainership of the `kernel-docs.rst` file in the kernel source tree, as well as the Spanish translation of that file. This file exists in the kernel's larger documentation directory hierarchy, which consists of many files that cover a wide array of developer activities, such as how to apply a patch or add a new system call.

The `kernel-docs.rst` file is a list of kernel documentation materials – starting with the larger documentation directory that the file is a part of. It also lists published books, online resources such as *LWN.net*, and so on. The top of the file explains, "The need for a document like this one became apparent in the linux-kernel mailing list as the same questions, asking for pointers to information, appeared again and again."

Carlos submitted a patch making himself the official maintainer of this file, and he also put out "a call for anyone interested in adding new, more up to date references to kernel-docs.rst. The document has been abandoned for a while but its original goal is still important."

Jonathan Corbet, maintainer of the *LWN.net*, replied, "I made an attempt to update this document a few years back and concluded that it was pretty much hopeless. What is there is ancient … what do you replace it with? There is a vast amount of content out there that will go obsolete just as quickly."

But having said that, Jonathan added, "I'm certainly not going to stand in the

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**.

way of anybody who wants to update and maintain this document, though."

Carlos attempted to clarify the potential issues involved in maintaining the file. First, he said, Jonathan was right about how quickly any resources it listed would go out of date. And second, he said, there was a real need to keep the file updated in a way that a single maintainer would find very difficult to do.

He suggested a standard process that he and others might follow together. To purge out-of-date material, he suggested that he could go into the file a couple times per year and remove anything more than three years old, "except for foundational books or active websites."

As far as updating the file, Carlos felt access should be opened up to the people producing the kind of documentation that would be listed in it. He said, "For example, Lorenzo Stoakes is writing a book on the Linux memory management subsystem. If Lorenzo reaches out to me when he is finished, I could add his work to this document, and also send a note informing subscribers of linux-docs about this new resource. I imagine that would be appealing to Lorenzo."

Essentially, his idea was to entice content creators to reach out to him directly, thus taking some of the burden of ongoing research off of his shoulders as the file's maintainer. He added, "Worst-case scenario, we end up with an equally outdated list of resources in a few years."

Jonathan had no objection, and Carlos started composing a new patch to document this approach within the file.

It's always good to see documentation efforts come to life and find new ways to stay relevant. Hopefully people maintaining or producing kernel-related documentation will get in touch with Carlos and help keep the `kernel-docs.rst` file up-to-date.

Linux in Carspace

Christoph Fritz wanted to support Local Interconnect Network (LIN) in Linux. LIN is a cheap, standardized networking protocol intended mostly for car manufacturers, to stop the proliferation of incompatible systems on the road.

Christoph posted some patches in the hopes of rallying kernel developers for a general discussion on the best way forward. His company, hexDEV, had already written a kernel driver called hexLIN. He said "its purpose is mainly to test, adapt and discuss different LIN APIs for mainline Linux kernel. But it can already be used productively as a Linux LIN node." He added, "We are looking for partners with Linux based LIN projects for funding."

Oliver Hartkopp replied, saying that the LIN-bus open source project already existed, "and implements the LIN protocol based on a serial tty adaption (which the serial LIN protocol mainly is)."

Christoph said he knew about that project and had even cc'd Pavel Pisa, one of its principal developers, in his original email. He pointed out, "When there is an internal kernel API for LIN, his sllin (tty-line-discipline driver for LIN) could be adjusted and finally go mainline."

Pavel himself also replied, saying he would love to get access to some actual hardware; he gave a brief technical explanation of some of the implementation difficulties he'd been having with the project so far.

Meanwhile, Ryan Edwards also said, "I've spent quite a bit of time trying to craft a solution for the LIN problem. Even with a TTY solution the best I was able to achieve was 40ms turnaround between the header and response which exceeded the timeout of the master. This was in userspace and I assume that a kernel solution would better be able to meet the timing but this solution would only work for devices with embedded UART."

Christoph, Pavel, and Ryan dug into the nitty gritty together, and Christoph tried to keep the discussion focused on choosing a target application programming interface (API) that they should all aim for. To which, Andrew Lunn cautioned, "for networking in general, we try very hard to make offload to hardware not special at all. It should just transparently work. One example of this is Ethernet switches which Linux controls. The ports of the switch are just normal Linux interfaces. You can put an IP address onto the ports in the normal way, you can add a port to a linux bridge in the normal way. If the switch can perform bridging in hardware, the linux bridge will offload it to the hardware.

But for the user, it's just a port added to a bridge, nothing special."

Andrew added in conclusion, "please try to avoid doing anything special for hardware offload. We don't want one way for software, and then 42 different ways for 42 different offload engines. Just one uAPI [Universal API] which works for everything."

The discussion ended fairly abruptly shortly thereafter. But it does seem that at least several kernel developers care about automotive networking and the LIN standard in particular. And at least one of those developers favors identifying the friendliest and most generic possible API to go into the kernel.

It's interesting to think about. Many car companies use Linux as their car operating system, so features such as LIN support may feel relatively important for them to get into the kernel. Another interesting possibility, as Linux becomes more and more able to control vehicles, is that users might `root` their cars and install custom Linux systems on them. Certainly plenty of car owners are actively trying to reverse-engineer the various platform-specific protocols on their cars. I'm sure something interesting will come out of all that eventually.

Filesystems and Cryptography

Niels de Vos pointed out that a lot of filesystems were adding support for `fscrypt`, which encrypts user files, and that it might be useful for kernel users to be able to enable that kernel feature for some filesystems but not others. He proposed a simple kernel config option that would allow per-filesystem encryption. He remarked, "This RFC is mostly for checking the acceptance of this solution, or if an other direction is preferred."

Theodore Ts'o didn't like this idea at all. He remarked pointedly, "I'm not sure what's the motivation behind adding this configuration option. If memory serves, early in the fscrypt development we did have per-file system CONFIG's for fscrypt, but we consciously removed it, just as we no longer have per-file system CONFIG's to enable or disable Posix ACL's or extended attributes, in the name of simplifying the kernel config."

Niels confirmed that the Linux distributions were a big motivator for his initial post. He said, "This is mostly why I sent this RFC. We are interested in enabling fscrypt for CephFS (soonish) as a network filesystem, but not for local filesystems (we recommend dm-crypt for those). The idea is that functionality that isn't available, can also not (easily) cause breakage."

He pointed out that the kernel actually did have special per-filesystem config options already, for ACLs and other security features. He said, "Because these exist already, I did not expect too much concerns with proposing a CONFIG_EXT4_FS_ENCRYPTION."

But Ted replied, "Actually, I was thinking of getting rid of them, as we've already gotten rid of [EXT4_FS_XATTR]." He reiterated, "there are tons of file system features that have not and/or still are not supported for distros, but for which we don't have kernel config knobs. This includes ext4's bigalloc and inline data, btrfs's dedup and reflink support, xfs online fsck, etc., etc., etc. Heck, ext4 is only supported up to a certain size by Red Hat, and we don't have a Kernel config so that the kernel will absolutely refuse to mount an ext4 file system larger than The Officially Supported RHEL Capacity Limit for Ext4."

Eric Biggers remarked, regarding Niels' proposal, "as others have pointed out, it doesn't seem worth the complexity to do this." And he went on to offer some historical background:

"Before Linux v5.1, we did have per-filesystem options for this: CONFIG_EXT4_ENCRYPTION, CONFIG_F2FS_FS_ENCRYPTION, and CONFIG_UBIFS_FS_ENCRYPTION. If you enabled one of these, it selected CONFIG_FS_ENCRYPTION to get the code in fs/crypto/. CONFIG_FS_ENCRYPTION was a tristate, so the code in fs/crypto/ could be built as a loadable module if it was only needed by filesystems that were loadable modules themselves.

"Having fs/crypto/ possibly be a loadable module was problematic, though, because it made it impossible to call into fs/crypto/ from built-in code such as fs/buffer.c, fs/ioctl.c, fs/libfs.c, fs/super.c, fs/iomap/direct-io.c, etc. So that's why we made CONFIG_FS_ENCRYPTION into a bool. At the same time, we decided to simplify the kconfig options by removing the per-filesystem options so that it worked like CONFIG_QUOTA, CONFIG_FS_DAX, CONFIG_FS_POSIX_ACL, etc."

At around this point in the conversation, Niels remarked, "What you are explaining makes sense, and I am not sure if there is another good reason why splitting out fscrypt support per filesystem would be required. I'm checking with the folks that suggested doing this, and see where we go from there." He added, "I understand that there is a preference for reducing the number of Kconfig options for filesystems. That indeed would make it a little easier for users, so I am supportive of that as well."

And that was the end of the thread.

It's a tough balancing act! Obviously, it would be great if every feature had its own config option, as well as maybe more options to simplify having multiple features together, perhaps with exceptions specified by yet more config options. In some cases, I'm sure the kernel developers feel that a given set of features really does need such fine-grained control by system administrators, while in others they would prefer to keep the config system as simple as possible, even if it means sacrificing some behaviors system administrators might actually want. ■■■

AKADEMY 2023

Akademy is the annual world summit for KDE Community members, developers, translators, designers, and friends. Come join us!

Akademy 2023 will be a hybrid event, in **Thessaloniki, Greece & online**

July 15 - 21

Repositories for energy-saving software

Going Green

A number of open source projects offer tools for analyzing energy usage in software. *By Kristian Kißling*

Word is starting to get out: Software often unnecessarily wastes a huge amount of energy. Several open source projects are developing tools for studying and measuring exactly how much. This article tours some of the most promising tools.

Taking a Tour

The Green Software Foundation (GSF) [1], which the Linux Foundation launched in 2022, has the stated mission to "Build a trusted ecosystem of people, standards, tooling, and best practices for creating and building green software." For this purpose, the GSF, which includes companies such as Microsoft, Intel, and Accenture, is organized into working groups. There are already several repositories with software, tools, and documents available on *https://github.com/Green-Software-Foundation*.

For example, the Open Source Working Group (opensource-wg) maintains a repository called Awesome Green Software [2] with links to a whole host of tools and research articles on energy conservation, as well as links to useful free software. On top of this, you will find an overview of climate organizations, as well as access to free articles, papers, and books.

The GSF also highlights software for the cloud and machine learning. For example, there is a link to the Green Cost Explorer [3], a JavaScript tool that docks onto Amazon's AWS Cost Explorer, but it also takes into account the energy types of the AWS regions as well as the costs. It ultimately shows you how much renewable (*Green*) and fossil (*Grey*) energy your cloud application requires.

Kepler [4], the Kubernetes Efficient Power Level Exporter, relies on eBPF to extract power-related statistics from Kubernetes. The Experiment Impact Tracker [5] and the Carbon-Tracker [6] measure the expected energy overhead for machine and deep learning jobs. The GitHub repository also has operating system-specific tools. For example, the *Linux* entry takes you to PowerAPI [7] and PowerTOP [8].

Green Patterns

Anyone who is looking to make their own IT landscape greener will find what they are looking for in the Green Software Patterns repository [9]. The GSF Patterns project collects tips and best practices for different fields of IT (AI, cloud, web) [10]. The spectrum ranges from very generic tips (e.g., run cloud jobs geographically as close to the user as possible) to concrete recommendations for action (e.g., compress transferred data). The patterns reference the Software Carbon Intensity (SCI [11]) specification software and use simple formulas to show how admins and developers computationally implement the recommendations.

The specification helps to attach an SCI score to an application based on its CO2 emissions. The lower the value, the less climate-damaging gas the software causes. The SCI score maps CO2 emissions C per unit R, where a unit can be an API call, an additional user, or an ML training run. So the rough formula is $SCI = C/R$, where C can be further subdivided into more specific sources, such as O (operational emissions) and M (embodied emissions). O is the energy required by the software itself, and M is the proportional energy required to produce and dispose of the underlying hardware.

The Carbon Aware SDK [12], on the other hand, is something that developers can integrate into their own applications to take advantage of fluctuating energy prices in an environmentally smart way. The Carbon Aware SDK, which you can use either at the command line or via web API, is often used with non-time-critical machine learning jobs. The SDK helps move workloads to other cloud regions where renewable energy is cheap at the moment. Alternatively, it helps to throttle back operations as a function of the type of energy. The people behind the project refer to CO2 reductions in the double-digit percentage range.

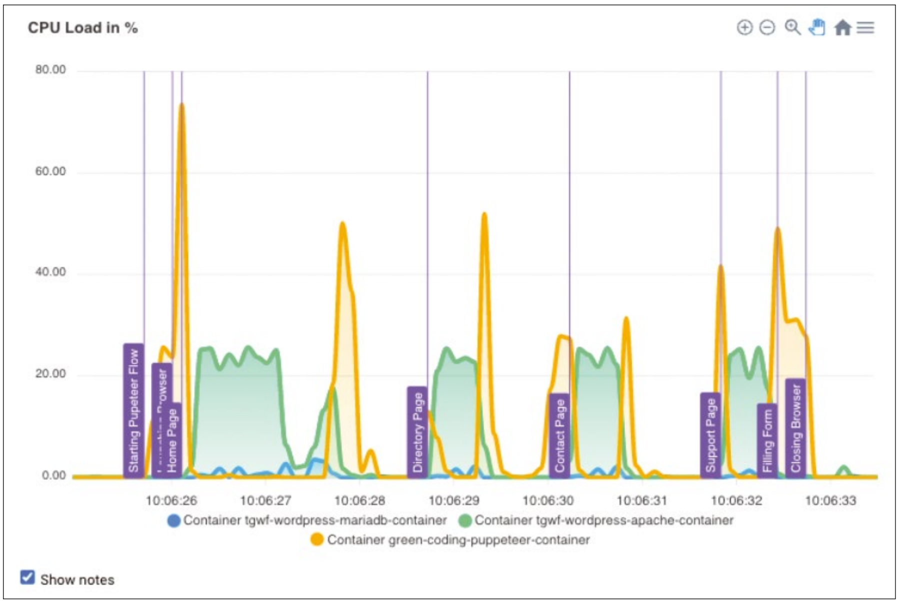


Figure 1: The Green Metrics tool takes measurements on containers and then visualizes the readings.

Green Coding

The tools of the GSF are available to everyone, but the foundation itself is based in the US and is primarily funded and maintained by US-based corporations. However, similar efforts are taking place in Europe. Green Coding Berlin, for example, is a German-based organization that believes digitalization will only have a positive impact on society if it is sustainable. Green Coding is looking to develop open-source measurement tools that determine the energy requirements of software and infrastructure. The aim is to create a community and an ecosystem centered around green software. To this end, Green Coding cooperates with the Sustainable Digital Infrastructure Alliance (SDIA [13]), an alliance of several companies with a roadmap for a sustainable digital infrastructure.

Green Coding offers the Green Metrics Tool [14], a container-based utility that is currently only intended for experimentation and testing. Green Metrics can help you measure the storage space per container, the CPU utilization as a percentage, or the system-wide power consumption. A Python API fields the metrics, and a JavaScript web front end visualizes the results (Figure 1).

KDE Eco

Green Coding collaborates with KDE Eco [15], a fledgling KDE initiative that aims to bring more sustainability to free and open software. The initial aim is to measure the energy requirements of free and open source software with a view to helping developers optimize software later on. The KDE Eco project offers two central repositories [16].

The FOSS Energy Efficiency Project (FEEP [17]) is fundamentally based on the Blue Angel methodology. (Blue Angel is a German initiative that assigns a special label to designate environmentally friendly products.) The FEEP website describes a measurement setup [18] that requires a reference computer, an energy meter, and software. This equipment helps to automate the execution of the software under test conditions and to record and process the measured data.

The BE4FOSS repository [19] collects Blue Angel information and provides links to accompanying workshops and manuals. Anyone thinking of certifying their own software with the Blue Angel label can find help in the Blue Angel Application [20] repository.

Conclusions

If you are seriously interested in software and energy saving, you will find many resources on the web; the examples presented in this article are a few of the projects working on the critical problem of energy efficiency. What is clear, however, is that many software efficiency projects are still in their infancy. As of now, the software community has no fixed standards for the measurement process or toolchains for integrating these capabilities into large infrastructures. Once these hurdles are removed, energy measurement tools could have a noticeable impact on software energy usage. ■■■

Info

[1] Green Software Foundation: *https://greensoftware.foundation*

[2] Awesome Green Software: *https://github.com/Green-Software-Foundation/awesome-green-software*

[3] Green Cost Explorer: *https://github.com/thegreenwebfoundation/green-cost-explorer*

[4] Kepler: *https://github.com/sustainable-computing-io/kepler*

[5] Experiment Impact Tracker: *https://github.com/Breakend/experiment-impact-tracker*

[6] CarbonTracker: *https://github.com/lfwa/carbontracker*

[7] PowerAPI: *https://github.com/powerapi-ng/*

[8] PowerTOP: *https://github.com/fenrus75/powertop*

[9] Green Software Pattern repository: *https://github.com/Green-Software-Foundation/green-software-patterns*

[10] Green Software Patterns: *https://patterns.greensoftware.foundation/*

[11] Software Carbon Intensity specification: *https://grnsft.org/sci*

[12] Carbon Aware SDK: *https://github.com/Green-Software-Foundation/carbon-aware-sdk*

[13] Sustainable Digital Infrastructure Alliance: *https://sdialliance.org*

[14] Green Metrics Tool: *https://github.com/green-coding-berlin/green-metrics-tool*

[15] KDE Eco: *https://eco.kde.org/de/*

[16] KDE Eco GitLab repositories: *https://invent.kde.org/teams/eco*

[17] FEEP repository: *https://invent.kde.org/teams/eco/feep*

[18] Measurement setup: *https://invent.kde.org/teams/eco/feep/-/blob/master/measurement_setup.md*

[19] BE4FOSS repository: *https://invent.kde.org/teams/eco/be4foss*

[20] Blue Angel Application: *https://invent.kde.org/teams/eco/blue-angel-application*

Energy-efficient programming with Go and beyond

Greener Coding

Go has a reputation for producing energy-saving applications, but you still have to know what you are doing.

By Tim Schürmann

Applications use the processor in different ways, and those differences are sometimes reflected in the power bill. The programming language you choose has a significant influence on energy consumption. But developers still need to go the extra mile to leverage the language's capabilities.

At first glance, Go comes with everything you need for energy efficiency. It has a lean syntax, and smart Goroutines distribute parallel tasks efficiently to the processor cores, thus avoiding a bloated runtime environment that needs to manage complex class hierarchies or juggle classic threads. And the compiler translates the source code into a native and therefore fast program, which means the processor can go back to sleep sooner and save energy. On top of that, Go statically links all external modules into the finished binary, eliminating the administrative overhead of dynamic libraries during execution.

This contrasts strongly with Java, where a compiler converts the program into intermediate code, which is then executed by a virtual machine. This additional software layer slows down execution and costs unnecessary energy. The situation is even worse for interpreted languages: PHP and Python parse each line of code step-by-step during execution. Numerous optimizations and just-in-time compilers are intended to make the resulting applications faster, but there is still some loss, depending on the task. If you want to develop energy-efficient programs, you need to go for Go, right? Or maybe not? The following look at energy efficiency in Go is a useful entry point for examining some more general rules for more efficient coding.

SLE Research

Back in 2017, a team of researchers wanted to know which programming language produces particularly energy-efficient applications. They presented the results of their study at the International Conference on Software Language Engineering (SLE [1]). According to the study, compiling programming languages such as C and Go work far more efficiently than interpreted counterparts such as PHP or Python. The results are less clear for languages that generate intermediate code. Java programs, for example, run faster and in a more energy-efficient way on average than their Go counterparts (Figure 1). Surprisingly, the virtual machine doesn't seem to slow things down – at least not in the trial, which has a few bugs on closer inspection.

The SLE study serves as a wake-up call for those who think they are automatically writing more efficient code just by choosing Go. However, a closer look also points to the complexity of these energy efficiency issues – and the danger of making assumptions based on a single study.

Initially, the researchers used only one possible compiler or interpreter for each language. Another Go compiler named TinyGo harnesses the LLVM framework to create highly optimized machine code for microcontrollers [2]. The C code in the study was compiled exclusively using the GCC.

The Go compiler used was version 1.6, and this was already considered obsolete at the time of the study. Of course, the other compilers and interpreters have also been revised and further optimized in the meantime. For example, PHP 8 can handle significantly more concurrent requests than its predecessors.

No Clouds

The researchers used the Computer Language Benchmark Game (Figure 2) for their measurements. This benchmark [3] consists of

10 individual tests implemented in a total of 28 languages. Among other things, these small programs calculate Mandelbrot sets and binary trees. The study only investigated which languages perform these individual manageable calculations in an energy-efficient manner. In contrast to this, real applications need to solve completely different tasks. Encryption tools, for example, struggle with extremely tricky mathematical problems, whereas web servers merely push data packets across the network.

Complex Internet and cloud applications, such as the ownCloud Infinite Scale (oCIS [4]) file sharing platform implemented in Go, were not included in the tests. OCIS was created as a more powerful alternative to the classic ownCloud PHP application, which reached its limits when faced with higher loads and many simultaneous users. The oCIS developers made their own attempt to optimize oCIS, and in the meantime, they ended up learning a lot about how to optimize Go. Unlike the small benchmark programs, however, oCIS consists of numerous services that work together on a network. The oCIS developers had to take a slightly different approach to their assessment.

If an application solves a task as quickly as possible, the processor can switch back to one of its energy-saving modes faster. It is therefore not surprising that, in the SLE study, the compiler languages were also the most energy-efficient. The oCIS developers also currently assume that high-performance programs consume less energy. This assumption allows for a focus on performance measurements and optimizations for which established tools and processes exist.

Go helps its developers with a built-in profiler [5]. The profiler measures the execution times for code parts to detect inefficient operations. According to Klaas Freitag, CIO of ownCloud GmbH, the Go profiler is usually only suitable for tracking down bugs, especially in distributed systems. This is why the oCIS team used an in-house development for its performance measurements: a tool named `cdperf` [6], which in turn harnesses the well-known `k6` test tool [7] (Figure 3). The `k6` test tool simulates multiple users accessing the file sharing platform simultaneously.

If so desired, the Go compiler can run customized benchmarks that you can use to determine the performance of

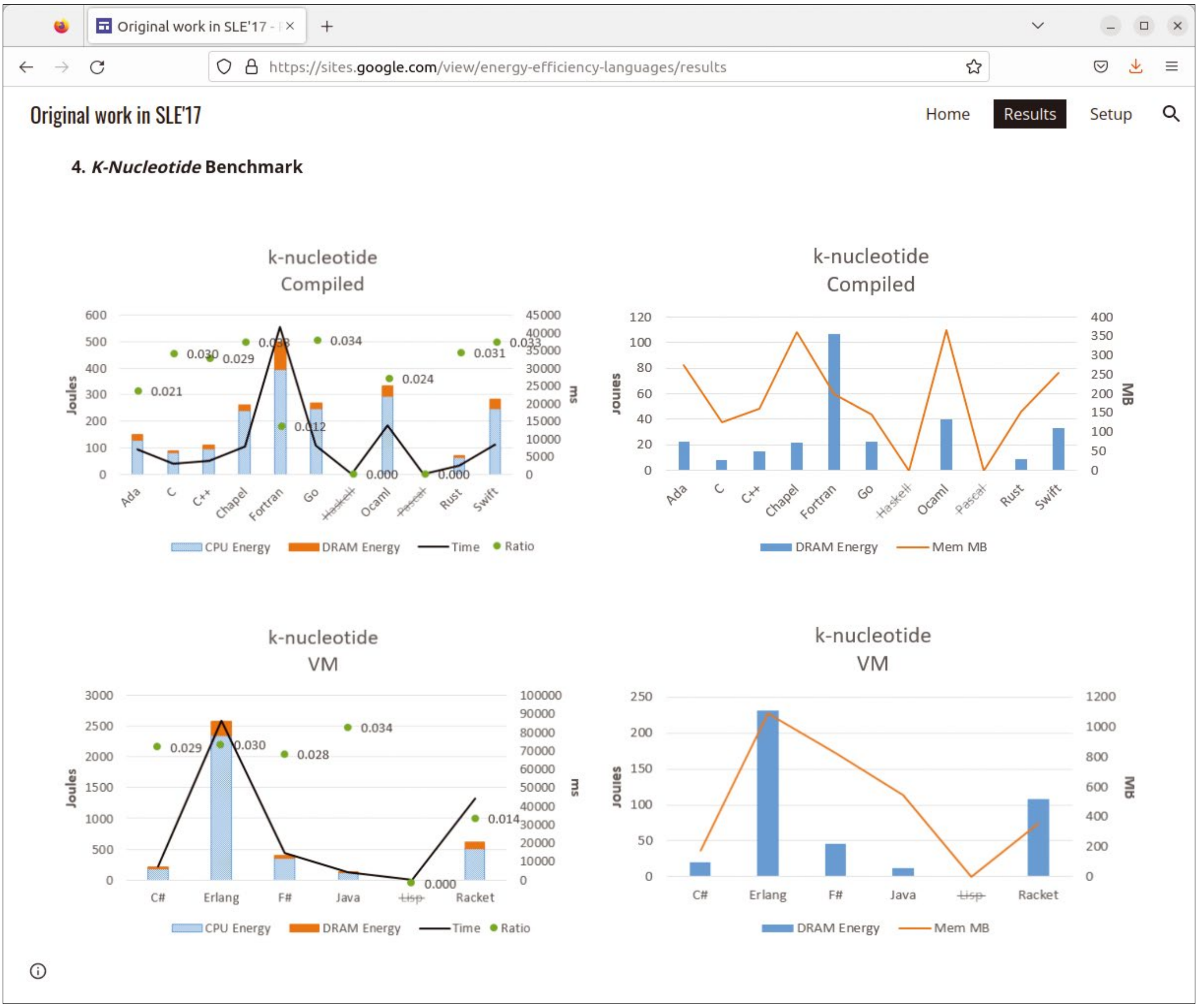
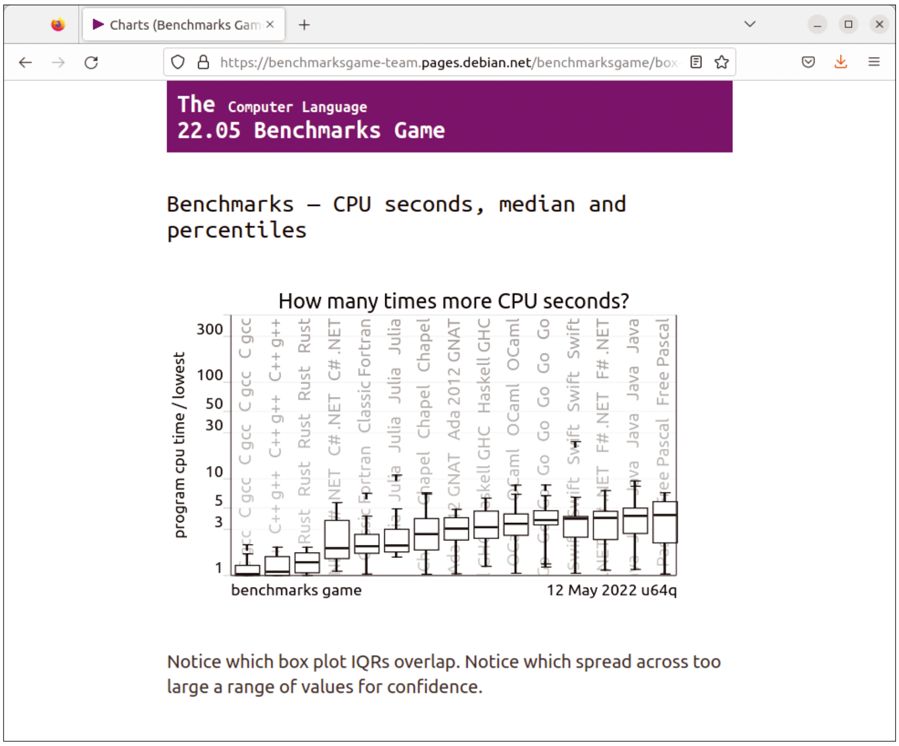


Figure 1: In the `k-nucleotide` benchmark, Go consumes more CPU power and main memory energy than other languages.

code changes [8]. For example, if a function calculates the greatest common divisor, you could call a DIY benchmark individually for thousands of different pairs of numbers. If the benchmark delivers a shorter runtime after a code change, you have successfully optimized the task of computing the greatest common divisor. The benchmarks can be maintained along with the (unit) tests and included in the development from the beginning.



The Computer Language
22.05 Benchmarks Game

Benchmarks – CPU seconds, median and percentiles

Notice which box plot IQRs overlap. Notice which spread across too large a range of values for confidence.

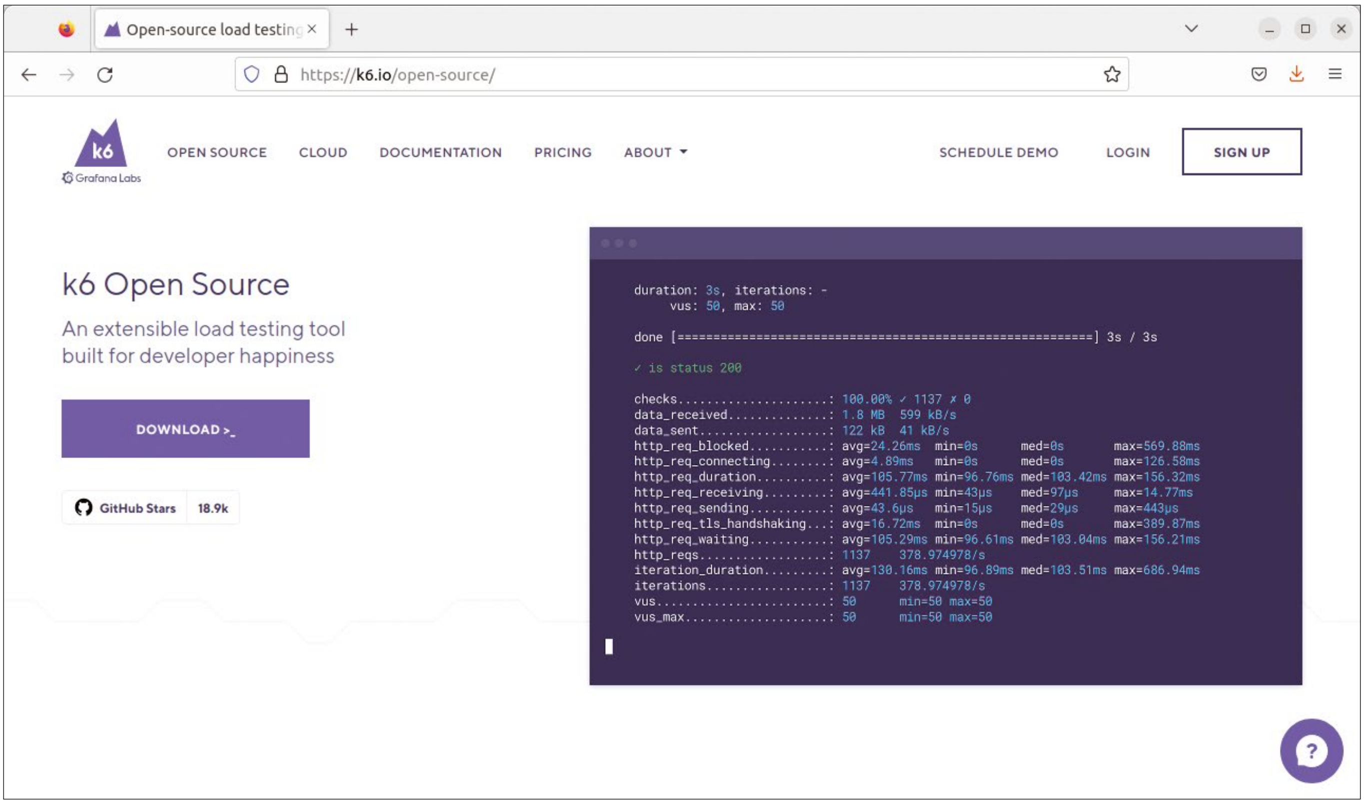
Figure 2: The Computer Language Benchmark Game is intended to compare the speed of programming languages.

Know Your Language

Once you have identified inefficient parts of the program, you need to optimize the source code; more specifically, you need to find the most efficient algorithms. Many programming languages support efficient development by evaluating constructs. For example, the Go compiler can already evaluate constant expressions, which saves computation time later during execution. However, this feature requires the developers to correctly integrate the constant expressions into the source code.

Furthermore, Go is a typed language that distinguishes between floating point and integer numbers. Because many processors calculate faster with integers, you should give them preference if possible. This is especially true if you use them as loop counters. Speaking of loops: Classic brakers and power guzzlers include nested but actually superfluous loops, sprawling recursions, and unnecessary repetitions. If a program has to recalculate all the thumbnails every time the image gallery is called up, for example, this repetitive work eats up energy unnecessarily.

The Go developers provide many more tips for efficient programming [9] on a



k6 Open Source

An extensible load testing tool built for developer happiness

```
duration: 3s, iterations: -
     vus: 50, max: 50

done [==========================================================] 3s / 3s

✓ is status 200

checks.....................: 100.00% ✓ 1137 ✗ 0
data_received..............: 1.8 MB  599 kB/s
data_sent..................: 122 kB  41 kB/s
http_req_blocked...........: avg=24.26ms  min=0s      med=0s       max=569.88ms
http_req_connecting........: avg=4.89ms   min=0s      med=0s       max=126.58ms
http_req_duration..........: avg=105.77ms min=96.76ms med=103.42ms max=156.32ms
http_req_receiving.........: avg=441.85µs min=43µs    med=97µs     max=14.77ms
http_req_sending...........: avg=43.6µs   min=15µs    med=29µs     max=443µs
http_req_tls_handshaking...: avg=16.72ms  min=0s      med=0s       max=389.87ms
http_req_waiting...........: avg=105.29ms min=96.61ms med=103.04ms max=156.21ms
http_reqs..................: 1137    378.974978/s
iteration_duration.........: avg=130.16ms min=96.89ms med=103.51ms max=686.94ms
iterations.................: 1137    378.974978/s
vus........................: 50      min=50 max=50
vus_max....................: 50      min=50 max=50
```

Figure 3: For their performance and energy measurements, the oCIS developers use the open source k6 tool.

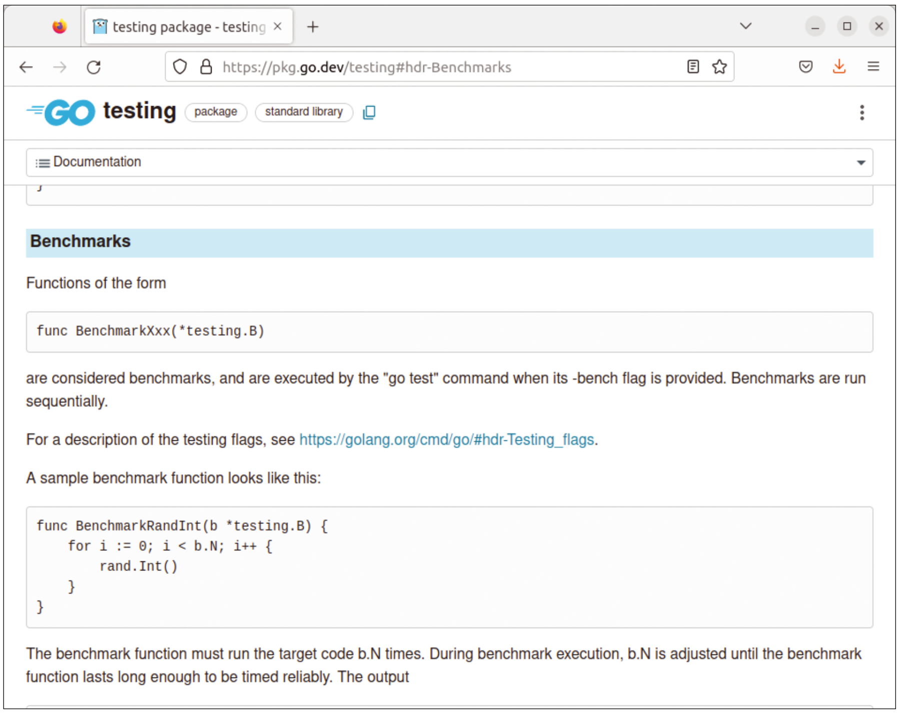
separate page (Figure 4). Comparable collections exist for other languages. In any case, you should not rush into every single file operation. From Freitag's point of view, such simple operations are already optimized. For example, PHP saves a file to disk just as quickly as a Go program would.

Efficiency Spoilers

Freitag acknowledges that efficient code does not necessarily lead to an energy-saving program. For example, a pass-through NAS solution draws more power from the grid than server-based services that start up time and time again. If an application demands hard work from the CPU for a short period of time and then has to clock up or switch on its power-hungry high-performance cores, the energy balance could be worse than that of a slow program that is satisfied with a power saver. Go developers can use the `NumCPU()` function from the standard library to query the number of available processor cores, but they cannot target work to a specific CPU core. You have to rely on the operating system or manipulate the Linux scheduler in order to target a specific core.

Special computing units can reduce energy consumption. For example, crypto units help build low-power SSL connections, and AI accelerators evaluate neural networks in an agile way. Go programs use libraries to pass their computations to these accelerators. For example, the GoCV package supports the use of NVIDIA's CUDA interface [10].

More energy guzzlers are hidden away in what are now numerous intermediate layers. For example, ownCloud, which is written in PHP, requires a complete LAMP stack. Accordingly, the power bill includes at least one database, a web server, and the PHP interpreter. On top of this, the individual services are often locked into a complex container environment or virtualization solution. OCIS, on the other hand, like most Go programs, runs natively on the server, making the PHP interpreter and some other intermediate layers obsolete.


testing package - testing
https://pkg.go.dev/testing#hdr-Benchmarks
GO testing
package
standard library
Documentation

Benchmarks

Functions of the form

```
func BenchmarkXxx(*testing.B)
```

are considered benchmarks, and are executed by the "go test" command when its -bench flag is provided. Benchmarks are run sequentially.

For a description of the testing flags, see https://golang.org/cmd/go/#hdr-Testing_flags.

A sample benchmark function looks like this:

```
func BenchmarkRandInt(b *testing.B) {
    for i := 0; i < b.N; i++ {
        rand.Int()
    }
}
```

The benchmark function must run the target code b.N times. During benchmark execution, b.N is adjusted until the benchmark function lasts long enough to be timed reliably. The output

Figure 4: Go comes with a test framework, which also gives developers the ability to run benchmarks.

Another task that often goes under the radar is software development. Developing software generates energy costs. In the development phase, complex CI systems repeatedly build and test the source code. OCIS is created using Drone [11], for example. Automatically starting the compiler and running the constantly recurring (unit) tests causes the software's footprint to grow, although the Go compiler itself is considered quite fast and economical.

Data Flood

Energy consumption depends not just on the processor load but also on the processed data. For example, an online store that is constantly bombarded with orders consumes more energy than it takes to deliver a text-centric website once an hour. Fetching large volumes of data from RAM costs significantly less power than requesting the data from a database or extracting the data from disk storage. Caches also catch intermediate results, such as preview images once they are generated. Once again, you don't have to reinvent the wheel: Many libraries provide efficient caching. Go developers can take a look at Gocache, for example [12].

However, RAM also consumes power. If the required data fits into the processor cache, the main memory goes to sleep. So if you're juggling sparse matrices, for example, you will want to compress them using efficient methods. By the way, the study also looked at RAM consumption. The programs generated by Go take up less space on average than their counterparts from competitors. Only Pascal takes less space.

Programmers also play an important role in energy-efficient RAM usage. When you pass a value to a function (calling by value), Go and many other languages copy it in memory. If the function is called frequently, many copies of the same value are created. But if you pass the function a reference to the value (calling by reference), this removes the need to access memory. Much like C, Go offers pointers (Figure 5) for this purpose, meaning that you only pass in references to the actual data. This approach saves energy in the case of repeated access and large data structures.

The reference implementation of the current Go 1.19 uses a garbage collector [13] to manage memory. The garbage collector cleans up in the background and automatically releases memory space that is no longer needed. However, its activity consumes compute time and therefore power. Although you could disable the garbage collector, you might end up running out of RAM or experiencing some other unpleasant side effects. The results of the study indicate that the garbage collector in Go uses very little energy.

Energy Ping-Pong

The Computer Language Benchmark Game includes the `k-nucleotide` benchmark program, which relies on Goroutines in the Go implementation to compute the results. The JavaScript version also runs concurrently, thanks to the

Node.js runtime environment. The TypeScript version, on the other hand, has to perform all the computations one after the other and is therefore slower than the JavaScript and Go implementations. The SLE study reveals unsurprisingly that TypeScript uses 17 times more energy than the almost identical JavaScript.

Parallel calculations mean power savings for a program. A similar thing happens if a process needs to actively wait for an action to finish. For example, if a client polls a non-responsive server at short intervals, this causes power consumption. Callback functions or push messages ensure greater efficiency. The program sleeps until the computed data or an event wakes it up. Go supports this approach through its channels and with listeners from the `net` package in network programming.

OCIS consists of numerous microservices, each of which performs a clearly defined task. According to Freitag, communication between the services is responsible for most of the cost in terms of performance and, in-turn, energy. For example, if two microservices constantly send requests to each other, this game of ping-pong drives up energy consumption. The electricity bill also increases if all connected systems have to apply new settings. For example, if several services need to be informed of a change in authorizations, numerous messages will inevitably cross the network, and these messages need to be coordinated and evaluated. Consequently, you can save energy in distributed systems with a reduced communication frequency.

Hungry AI

Training neural networks burns large amounts of energy. The longer you train a neural network for its task, the more accurately it will work later on. But, due to the computing power required for this, in extreme cases, you could emit even more CO2 than five cars in their entire service life [14]. If you train a neural network beyond what is actually necessary for the task, you are wasting a great deal of energy. An environmentally harmful AI like this is known as a red AI.

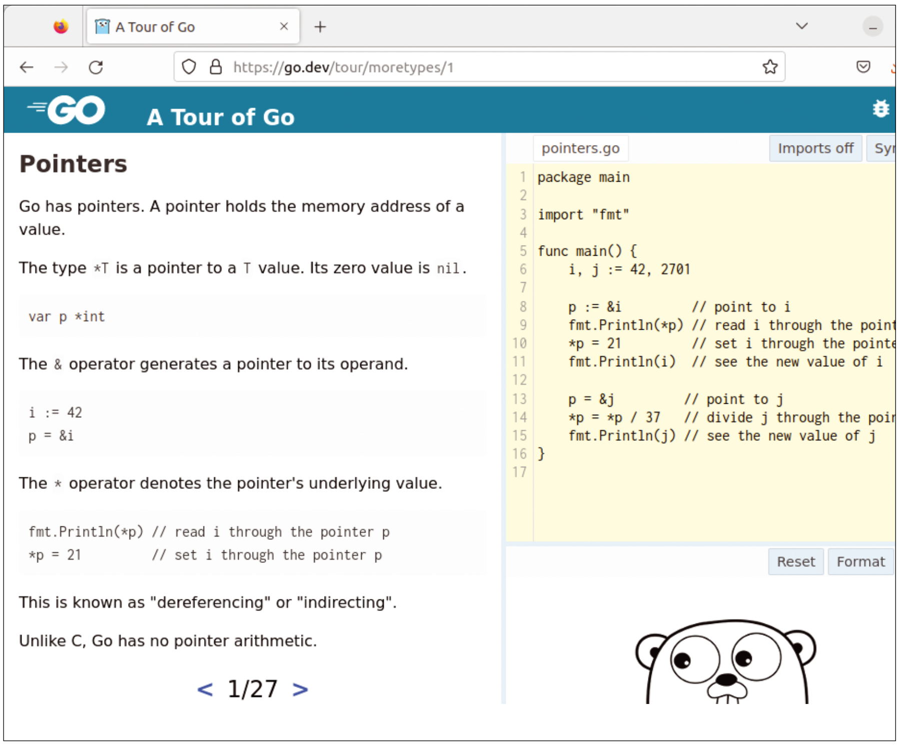
In addition, many scientists in AI research rely on Python. They resort to some tricks to speed up the computations. One of them is using the Cython compiler, which converts Python code into its C counterpart. In addition, many Python libraries are implemented directly in C. If you develop AI models, you need to design and train them to be as energy-efficient as possible. If you use pre-configured examples, go for variants that are already energy-efficient.

Strengths and Weaknesses

You can only use a programming language efficiently if you know its advantages and disadvantages. For example, the source code of some scripting languages is quite short. If the interpreter processes these instructions quickly, the scripts run as efficiently as their compiled counterparts and are also far easier to maintain.

In addition, some languages are tailored for very specific purposes. For example, PHP is primarily used to create dynamic web applications. Only a few programmers are likely to use it for complex numerical analysis. And if you need numerical analysis, the Julia functional programming language is a good choice. Today, it is mainly server-based services and command-line programs that are created in Go. The language offers pretty efficient string manipulations, but some tasks that execute in parallel can be solved efficiently with Goroutines.

Because Go has only been around for a few years, the programs written in it draw on libraries that are also quite young. The situation is different with established programming languages such as C, Python, or PHP. All too often, inefficient legacy systems lurk in software developed years ago. Refactoring and retiring old libraries can help save energy. The oCIS developers used the reimplementation of their file sharing platform to dump some legacy material. For instance, unlike ownCloud, oCIS does not require a classic database.



Pointers

Go has pointers. A pointer holds the memory address of a value.

The type `*T` is a pointer to a `T` value. Its zero value is `nil`.

```
var p *int
```

The `&` operator generates a pointer to its operand.

```
i := 42
p = &i
```

The `*` operator denotes the pointer's underlying value.

```
fmt.Println(*p) // read i through the pointer p
*p = 21         // set i through the pointer p
```

This is known as "dereferencing" or "indirecting".

Unlike C, Go has no pointer arithmetic.

```
package main

import "fmt"

func main() {
    i, j := 42, 2701

    p := &i         // point to i
    fmt.Println(*p) // read i through the point
    *p = 21         // set i through the pointe
    fmt.Println(i)  // see the new value of i

    p = &j         // point to j
    *p = *p / 37   // divide j through the poi
    fmt.Println(j) // see the new value of j
}

```

Figure 5: Pointers speed up memory access and data transfer in Go. In this way, you can quickly exchange the content of a variable without having to copy it.

If You Perform Your Own Measurements …

At the end of the day, only your own measurements will help you find out how much energy is going to which parts of your code. These measurements should take place during the development phase and in real time. On Linux, for example, you can check energy consumption using Intel's RAPL interface.

In the case of Go, however, you need to use auxiliary applications to check energy consumption, because Go programs lack this capability. A proposal for a function that checks energy consumption, dating back to 2019, was rejected by the developers because, in their view, neither operating systems nor CPUs provide precise consumption values for the short pieces of code that are executed [15]. At the time of going to press, there were no libraries that Go developers could use to measure energy consumption.

But Go at least comes with some debugging functions in its standard library. For example, the `debug` package contains a function that you can use to trigger garbage collection. The `ReadMemStats()` function prints information about memory usage from the `runtime` package, and you can generate your own metrics with the `metrics` package.

… Make Sure You Measure Something Meaningful

When measuring energy consumption, there is also the question of which program paths your code executes. As Freitag points out, errors are not allowed to occur during the measurements. Errors often cause an operation to abort and the program to generate less load.

You also need to measure under real-life conditions if possible. Developers sometimes get in a hurry and simplify the test environment in ways that could affect the measurement results. During their measurements, the oCIS developers resorted to the simplest authentication method (Basic Authentication). As a result, the system repeatedly decoded the same security token in a computationally intensive way. After switching to OpenID Connect, the entire system was measurably faster.

Energy measurements need to cover as many of the software parts as possible. Freitag recommends identifying several practical use cases and testing them. With file sharing software, it makes a difference whether a company shares only a few very large files or an extremely large number of small files. You need to take an average value into account in your measurements.

You can read the results of the SLE study online if you are interested [16], and you will find the test programs on GitHub [3]. OCIS developers are hoping to disclose their comparisons of ownCloud, oCIS, and Nextcloud. At the time this article was written, an internal validation was still in progress. But you can measure the performance yourself with the open source `cdperf` tool.

Conclusions

The C, Pascal, and Go languages help developers achieve an optimal compromise between runtime, memory usage, and energy efficiency – at least that's what the SLE study suggests. Nevertheless, the Go compiler does not automatically produce energy-efficient applications. Too many other factors play a role.

According to Freitag, it is primarily the software architecture that influences the performance and, indirectly, the energy efficiency of an application. For example, communication among Internet services can consume more energy than saving a file. In addition, efficient implementation is important, and that – in turn – depends on the use case.

There is no doubt that it is worthwhile to get started with resource-efficient programming now. See the box entitled "Tips for Energy-Efficient Implementation" for some tips on greener coding. ■■■

Tips for Energy-Efficient Implementation

- All stakeholders need to keep energy efficiency in mind at all times during the development process.
- Go for a compiler language such as Go, C, or Rust rather than an interpreted language such as Python.
- Choose a software architecture that efficiently solves the task at hand. For example, a blog should not consist of numerous microservices constantly communicating with each other.
- Use efficient algorithms. Eliminate costly polling and avoid unnecessary loops. Using a profiler can reveal inefficient pieces of code.
- If you use third-party libraries and code, extend your testing to ensure the third-party components also run as efficiently as possible.
- Reduce the volume of data to be processed, for example, by making do with smaller preview images, and store intermediate results in a cache.
- Make trade-offs in accuracy. A neural network does not always have to deliver perfect results. Integers are more efficient than floating point numbers as counter variables in loops.
- Use low-power specialist hardware such as AI accelerators whenever possible.
- As part of the testing process, measure power requirements, or at least performance, continuously. Optimize the inefficient operations identified in the process.
- Pay attention to the energy consumption of the software pipeline. Does the CI system really have to rebuild the software once an hour outside working hours and run through all the tests?

Info

[1] "Energy Efficiency across Programming Languages": *https://sites.google.com/view/energy-efficiency-languages/*

[2] TinyGo: *https://tinygo.org*

[3] "Energy Efficiency in Programming Languages (Benchmarks)": *https://github.com/greensoftwarelab/Energy-Languages*

[4] oCIS: *https://owncloud.dev/ocis/*

[5] "Profiling Go Programs": *https://go.dev/blog/pprof*

[6] cdperf: *https://github.com/owncloud/cdperf*

[7] k6: *https://k6.io/open-source/*

[8] Benchmarks: *https://pkg.go.dev/testing#hdr-Benchmarks*

[9] Effective Go: *https://go.dev/doc/effective_go*

[10] GoCV: *https://pkg.go.dev/gocv.io/x/gocv*

[11] Drone: *https://www.drone.io*

[12] Gocache: *https://github.com/eko/gocache*

[13] "A Guide to the Go Garbage Collector": *https://go.dev/doc/gc-guide*

[14] "Training a single AI model can emit as much carbon as five cars in their lifetimes": *https://www.technologyreview.com/2019/06/06/239031/training-a-single-ai-model-can-emit-as-much-carbon-as-five-cars-in-their-lifetimes/*

[15] "proposal: testing: report energy consumption figures": *https://github.com/golang/go/issues/30108*

[16] "Energy Efficiency across Programming Languages, Results": *https://sites.google.com/view/energy-efficiency-languages/results*

Booting up the coreboot firmware alternative

Core Strength

Coreboot is an open source firmware alternative with an emphasis on speed and simplicity.

By Bruce Byfield

In the next few years, how you boot a computer could change drastically. A firmware alternative known as coreboot [1] is well on the way to becoming readily available, and it is likely to become more popular as development accelerates. For one thing, coreboot is a giant leap forward for open hardware. Just as importantly, on hardware where it can be implemented, coreboot boots three to four times faster than the familiar BIOS or UEFI, and, depending on how it is configured, sometimes even faster.

Originally known as LinuxBIOS, coreboot was founded in 1999 at Los Alamos National Laboratory, the research and development facility best known for its role in the development of the atomic bomb. Today, coreboot's major contributors include manufacturers such as AMD and SiS, and motherboard vendors such as MSI, as well as various participants in the Google Summer of Code.

Supported CPU architectures include x86-64, ARM, ARM64, and RISC-V, as well as AMD's Geode and other software-on-a-chip platforms. In addition to Linux, coreboot directly supports BSD, OpenBSD, and Windows 2000 or later. You can also use coreboot indirectly with other operating systems specified in the payload. In addition, coreboot has inspired a number of spin-off projects, such as the Libreboot distribution [2] and librecore [3], which places a heavy emphasis on software freedom and non-x86 architectures. As these lists show, coreboot is supported across a wide spectrum of hardware and developed by an alliance of academics, manufacturers, and community members. Although reverse engineering is sometimes required, in a growing number of cases, coreboot developers can get schematics directly from manufacturers.

Developers have long recognized the growing need for a new firmware solution. According to the coreboot site, the size of a BIOS once averaged about 100KB but the average size is now closer to 8MB, and it almost certainly contains obsolete and redundant code, which seriously slows boot time. By contrast, the coreboot site claims [4] that "For desktops and laptop machines, coreboot can frequently boot to the start of the operating system in under a second. For servers, it can cut minutes off of the boot time." Not only that, but coreboot is designed to meet modern security standards, and, being smaller than a conventional BIOS, it provides a smaller target for security breaches. Yet another advantage is that, in keeping with open source tenets, "The architecture of coreboot is designed to have an unbrickable update process. Updating firmware should be no more dangerous than installing your favorite app on your mobile phone" [4]. In every way, coreboot is a timely overhaul of the conventional concept of computer firmware.

How Coreboot Works

Coreboot is designed to provide the absolute minimum of instructions to launch a modern operating system. The minimum structure means that coreboot must be modified for each chipset and

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/*.

Photo by Victor Freitas on Unsplash

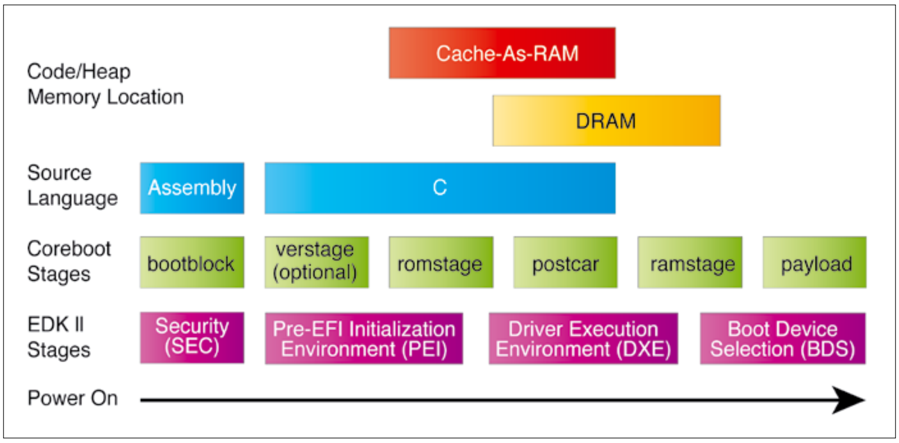


Figure 1: The stages of coreboot start-up. Source: modified from https://www.coreboot.org/

motherboard it supports, which delays progress but tends to increase efficiency.

Coreboot runs in five required stages plus an optional stage on x86 machines [5] (shown in timeline form in Figure 1):

- bootblock: The first stage is written in assembly language and is intended to set up the C environment used for the rest of the coreboot process. Tasks include initializing the Cache-as-RAM, which uses the CPU cache as memory for the heap and stack space required by the C environment. On x86 systems, the bootblock stage also switches the CPU from 16-bit real mode to 32-bit protected mode.
- verstage: An optional stage that starts the root of trust if verified boot is used.
- romstage: Prepares the system to access DRAM directly.
- postcar: Tears down the Cache-as-RAM memory and loads the ramstage.
- ramstage: Initializes hardware, including PCI, on-chip, and graphics devices, trusted platform modules, and the CPU. Initialization tables are prepared for the operating system, and hardware and firmware are locked down.
- payload: Loads a chunk of software carried in firmware storage that initializes the process of launching the operating system. Two commonly used payloads are SeaBIOS, an implementation of the x86 BIOS, and TianoCore, an open source version of UEFI. The GRUB2 bootloader can also serve as a coreboot payload. Given coreboot's speed, long-time Linux users might be bemused to find that it is impossible to read the output of the boot process as it happens. They will need instead to open `/var/log/boot.log`.

Figure 1 also illustrates how the coreboot stages fit within the stages of the EDK II cross-platform firmware specification.

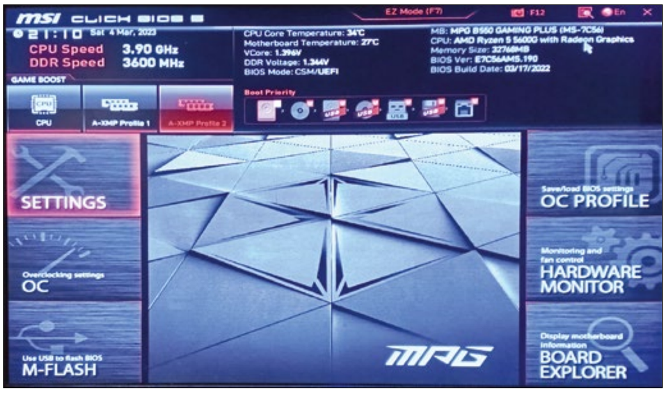
System76: A Sample Implementation

According to Wikipedia, computers with coreboot are available, including some x86-based Chromebooks, and from One Laptop per Child, Minifree (formerly Gluglug), PC Engines, Purism, System76, and Star Labs. Most of these computers are laptops, and some are refurbished, but the list has grown steadily over the past few years.

My hands-on experience with coreboot comes from a recently purchased Darter Pro from System76. In the past few years, System76 has emerged as a major manufacturer of Linux computers. Originally, System76 shipped its computers with a standard BIOS. However, gradually, it has been switching to coreboot for its laptops. As of March 2023, all of its six laptop models use coreboot. No official word yet of when coreboot will come to the company's desktops, servers, and minis, but a reasonable guess is that it is only a matter of time.

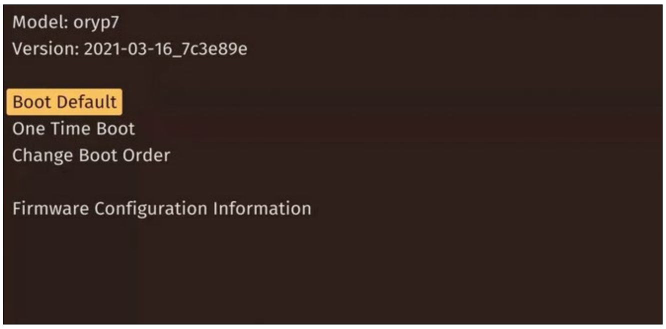
After the traditional BIOS (Figure 2) and UEFI implementations, coreboot on the Darter Pro comes as something of a shock. The BIOS of an earlier version of the Darter Pro had 38 top-level items, including specifications, as well as support settings for Secure Boot, Thunderbolt 3, and virtualization. By contrast, the current Darter Pro coreboot menu shown in Figure 3, which is accessed by holding down the Esc key at start-up, is a radical simplification.

System76's coreboot adoption seems a work in progress, so perhaps more items will be added in the months to come. The GitHub development pages seem at the least to keep that possibility open. However, this implementation of the firmware offers only the information that users are most likely to want, and not all of that can be changed. If System76's coreboot menu is compared with coreboot's build configuration (Figure 4), you can see that System76 chooses relevance and simplicity in its coreboot build.



21:10 Sat 4 Mar, 2023
CPU Speed 3.90 GHz
DDR Speed 3600 MHz
CPU Core Temperature: 34°C
Motherboard Temperature: 27°C
VCore: 1.396V
DDR Voltage: 1.344V
BIOS Mode: CSM/UEFI
MB: MPG B550 GAMING PLUS (MS-7C56)
CPU: AMD Ryzen 5 5600G with Radeon Graphics
Memory Size: 32768MB
BIOS Ver: E7C56AMS.190
BIOS Build Date: 03/17/2022

Figure 2: A typical BIOS – what coreboot replaces. Source: https://system76.com/


Model: oryp7
Version: 2021-03-16_7c3e89e

Boot Default
One Time Boot
Change Boot Order

Firmware Configuration Information

Figure 3: System76's coreboot menu is a simplification that shows only the items that matter most to typical users. Source: https://system76.com/

Much of the usual information in the BIOS is available in System76 Open Firmware, along with instructions on how to customize the firmware and flash it – as well as suggestions on how to recover if things go wrong. Control of hardware such as the keyboard, fan, and battery is available through System76 Open EC Firmware (Embedded Controller), sometimes in the form of keyboard shortcuts and sometimes in the form of desktop applications. Other implementations of coreboot will differ to some degree; Purism's Librem 14 laptop [6], for instance, is advertised as having "Disabled the Intel Management engine" and "Less binary blob firmware," with an emphasis on security. But no matter what the priorities are, the ultimate goal of a coreboot implementation is to provide a simpler, more efficient way to start a computer.

Welcome to the Revolution

Coreboot faces serious obstacles. Too often, it has to work around proprietary code, which takes time and sometimes compromises the project's open source philosophy. Perhaps, too, the conservatism of corporations favors the established structure of BIOS and UEFI, for no better reason than its familiarity. Probably, though, the greatest obstacle is the fact that every chipset and motherboard requires its own implementation. Still, the use of separate payloads eases that restriction and has led to another advantage: well-organized, clear, and complete documentation, both in the project itself [7] and among early retailers such as System76.

Coreboot's natural advantages are so obvious that to predict its future dominance is far from rash. Just to see a computer boot at a fraction of the time you expect, or to realize how easy flashing firmware can be, is enough to make a believer out of the most cynical. One way or the other, expect to hear more about coreboot in the near future. ■■■

Info

[1] coreboot: *https://coreboot.org/*

[2] Libreboot: *https://libreboot.org/*

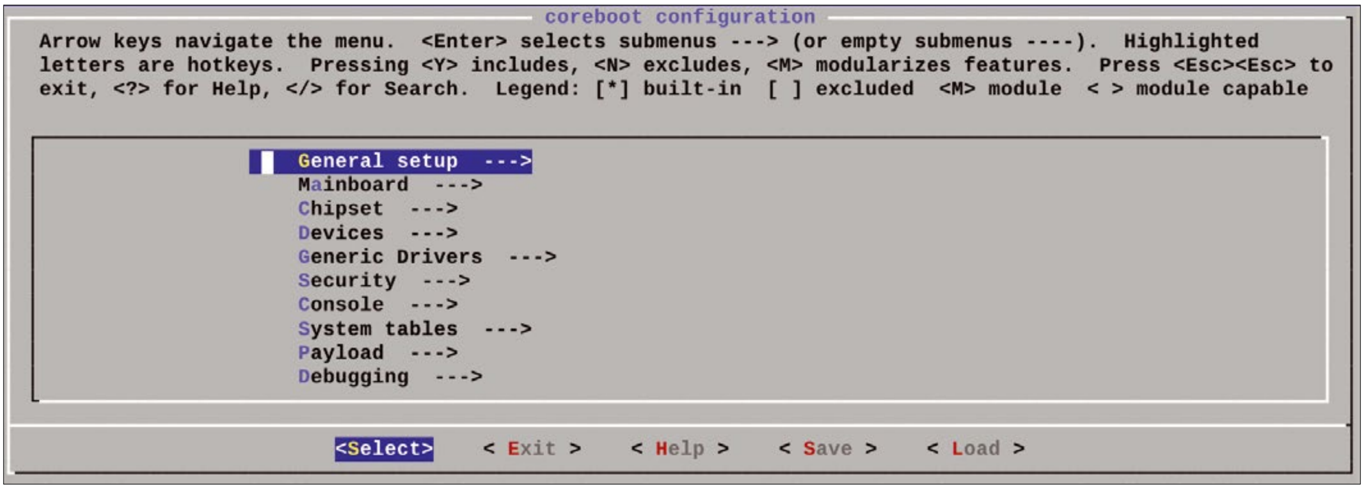
[3] librecore: *https://firmwaresecurity.com/tag/libreboot/*

[4] Advantages for end users: *https://www.coreboot.org/users.html*

[5] coreboot architecture: *https://doc.coreboot.org/getting_started/architecture.html*

[6] Purism's Librem 14: *https://puri.sm/products/librem-14/*

[7] Documentation: *https://doc.coreboot.org/getting_started/index.html*



```
                              coreboot configuration
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted
  letters are hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to
  exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

                  General setup  --->
                  Mainboard  --->
                  Chipset  --->
                  Devices  --->
                  Generic Drivers  --->
                  Security  --->
                  Console  --->
                  System tables  --->
                  Payload  --->
                  Debugging  --->

                <Select>    < Exit >    < Help >    < Save >    < Load >
```

Figure 4: Coreboot includes a wealth of configuration options, although System76 is selective about which ones users are offered.

The GNOME™ Conference

GUADEC

Riga, Latvia
July 26–31, 2023

GUADEC is GNOME’s main annual event. Held since 2000, it brings together Free Software enthusiasts and professionals from all over the world.

Join us at GUADEC 2023, in Riga or online, to hear about the latest technical developments, attend talks, participate in workshops, and celebrate GNOME!

Learn More

GUADEC.ORG

Photo by hani Pirzadian on Unsplash

TUXEDO OS 2 Preview in test

New Tuxedo

The popular Linux PC forge TUXEDO extends Ubuntu to include the latest KDE packages and says goodbye to Snap for its in-house TUXEDO OS distribution. The latest version is suitable for any PC. *By Hans-Georg Eßer*

It is hard to predict whether installing Linux on a new PC will work without detours. Individual components might only work after some time-consuming research and extensive efforts to install the required drivers. In fact, the result might even turn out to be a non-booting total failure. That's why Linux-friendly hardware vendors offer computers with a preconfigured Linux system, where everything is guaranteed to work. Some go so far as to put together their own distributions, which include all the required drivers, kernel patches, and additional tools for all the computers they offer.

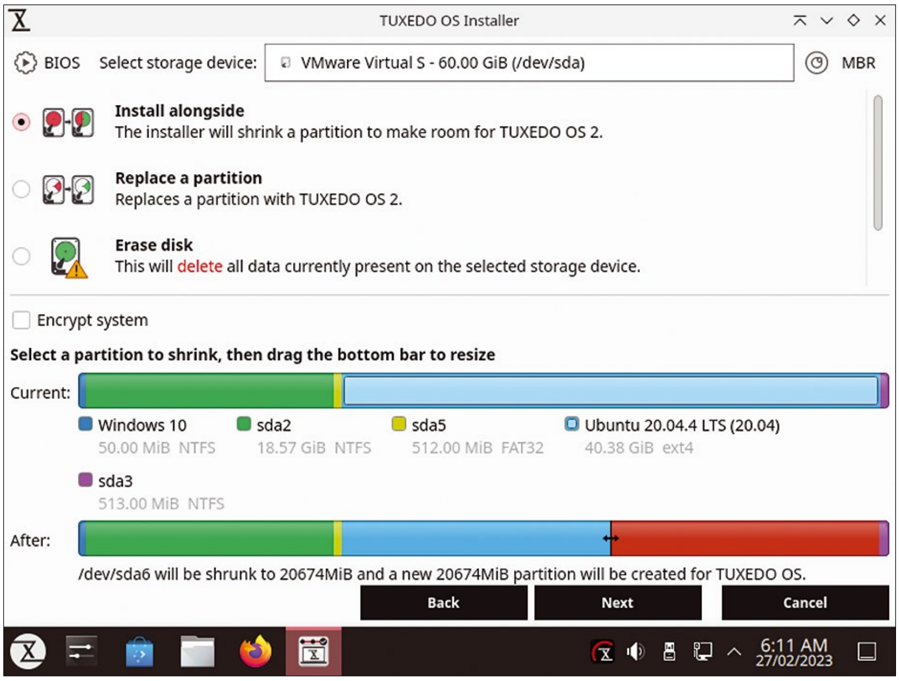
This is true, for example, of US supplier System76 or the German computer manufacturer TUXEDO. Both these vendors provide their own signature Linux distributions, but, whereas System76's Pop!_OS has been freely available for some time, TUXEDO's TUXEDO OS was intended only for TUXEDO customers until September 2022, when developers released TUXEDO OS 1 as an ISO image. The version 2 release was imminent at the time this article was written. We decided to take a look at TUXEDO OS 2 Preview [1].

TUXEDO OS 2 is based on Ubuntu 22.04, but it comes up as TUXEDO OS 2 when you type `lsb_release -a` or display the `/etc/os-release` configuration file. Nothing changes in terms of the typical Ubuntu-style conventions: For example, there is no root account, and you need Sudo to gain administrative privileges.

Installation

If you slot in the 3.3GB installation disc with `TUXEDO-OS-2-202302231702.iso` (you can also write this to a USB stick), you are first taken to a live system after selecting *TUXEDO OS 2 Try & Install* in the boot loader. After prompting you for the language, region/time zone, and keyboard mapping, the installer spends a few seconds updating the localizations before launching a KDE desktop.

A click on the sole desktop icon (*Install TUXEDO OS*) lets you start the setup routine. TUXEDO OS relies on the popular Calamares [2] installer framework. If you're installing on a non-TUXEDO device, a warning will appear to



TUXEDO OS Installer

BIOS Select storage device: VMware Virtual S - 60.00 GiB (/dev/sda) MBR

● **Install alongside**
The installer will shrink a partition to make room for TUXEDO OS 2.

○ **Replace a partition**
Replaces a partition with TUXEDO OS 2.

○ **Erase disk**
This will delete all data currently present on the selected storage device.

☐ Encrypt system

Select a partition to shrink, then drag the bottom bar to resize

Current:

Windows 10 50.00 MiB NTFS
sda2 18.57 GiB NTFS
sda5 512.00 MiB FAT32
Ubuntu 20.04.4 LTS (20.04) 40.38 GiB ext4
sda3 513.00 MiB NTFS

After:

/dev/sda6 will be shrunk to 20674MiB and a new 20674MiB partition will be created for TUXEDO OS.

Figure 1: The partitioner can shrink existing Windows and Linux partitions to make room for TUXEDO OS.

remind you that any special tools intended for TUXEDO hardware won't work.

The installer prompts you for the language again and then jumps directly to the most important part, partitioning the disk. You have four options: You can select a partition to shrink in *Install alongside*. A slider (Figure 1) helps you divide the disk space between the old and new systems. *Replace a partition* lets you define a partition to delete. In addition, you can *erase* the hard disk and choose *Manual partitioning*. This works for legacy partition tables (MBR) and for the newer GPTs. You can also elect to encrypt the TUXEDO OS partition with LUKS by clicking on *Encrypt System* and assigning a password.

When you erase the disk, you can choose between *No swap*, *Swap (without hibernation)*, *Swap (with hibernation)*, and *Use swap file* for the swap area. By default, the system uses an installation without swap space, and when shrinking an existing partition, the installer does not even ask you for swap space.

As the file system for the root partition, you can choose ext4 (the default), Btrfs, or XFS – but only if you use the whole disk or partition manually. The install continues with creating the standard user account. The *Require strong passwords* option can be disabled, and TUXEDO OS will even log you in automatically if so desired. That's all it takes; the installer now unpacks some files and waits for you to confirm the reboot.

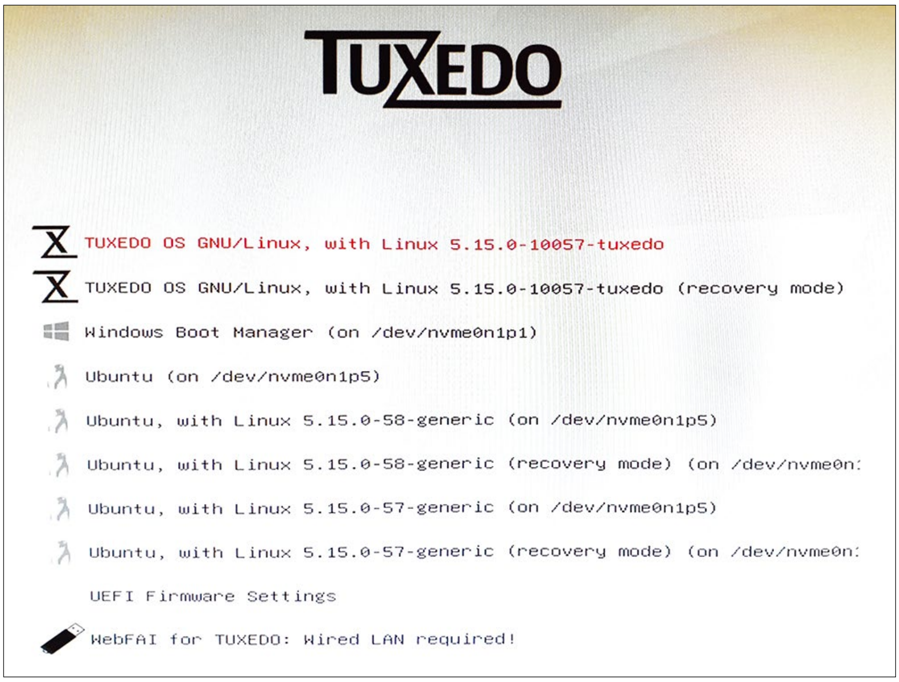
The freshly installed GRUB boot loader has a boot menu from which you can boot other Linux and Windows installations (Figure 2). This worked without problems in the test; in fact, a triple boot configuration with TUXEDO OS, Ubuntu 22.04, and Windows 11 worked without further ado.

Interface

On the test machine, a compact mini-PC with a quad-core Celeron N5105 and 16GB RAM, TUXEDO OS 2 booted into a usable KDE desktop in 13 seconds. For comparison's sake: Ubuntu 22.04 took 18 seconds on the same machine, and Windows 11 Professional 20 seconds. All three systems were configured to automatically log in the default user.

The current TUXEDO OS KDE packages come from the KDE-Neon repository [3]. The look, placement, and structure of the start bar and menu are sensibly chosen for Linux newcomers. The interface responds quickly to all actions. Initially, the desktop has no additional workspaces. You can drop files directly on the desktop, and they end up in the `~/Desktop/` folder.

By default, applications use light colors, and the start bar and menu have a dark color scheme. If you change the colors to *TUXEDO Dark* in the settings in *Appearance*, give the browser a darkening add-on, and change the background, you can make things really dark (Figure 3).



```
TUXEDO OS GNU/Linux, with Linux 5.15.0-10057-tuxedo
TUXEDO OS GNU/Linux, with Linux 5.15.0-10057-tuxedo (recovery mode)
Windows Boot Manager (on /dev/nvme0n1p1)
Ubuntu (on /dev/nvme0n1p5)
Ubuntu, with Linux 5.15.0-58-generic (on /dev/nvme0n1p5)
Ubuntu, with Linux 5.15.0-58-generic (recovery mode) (on /dev/nvme0n:
Ubuntu, with Linux 5.15.0-57-generic (on /dev/nvme0n1p5)
Ubuntu, with Linux 5.15.0-57-generic (recovery mode) (on /dev/nvme0n:
UEFI Firmware Settings
WebFAI for TUXEDO: Wired LAN required!
```

Figure 2: The boot manager lets you launch other Linux and Windows installations in addition to TUXEDO OS.

Software Selection

The developers pre-install many useful programs directly, including Firefox, Thunderbird, and KTorrent in the *Internet* category. You can use LibreOffice for your daily office work and play videos with VLC. VirtualBox – somewhat unusually – comes with the proprietary Oracle extension in place. Even the *build-essential* developer base package is already on board. Table 1 shows a small selection of the version numbers of the packages included with the TUXEDO OS 2 Preview.

Administrators who love legacy tools will be happy to see the terminal-based Midnight Commander file manager and the htop process viewer. TUXEDO OS does not automatically install an OpenSSH daemon. If you want to log in to the computer remotely, install OpenSSH Server with:

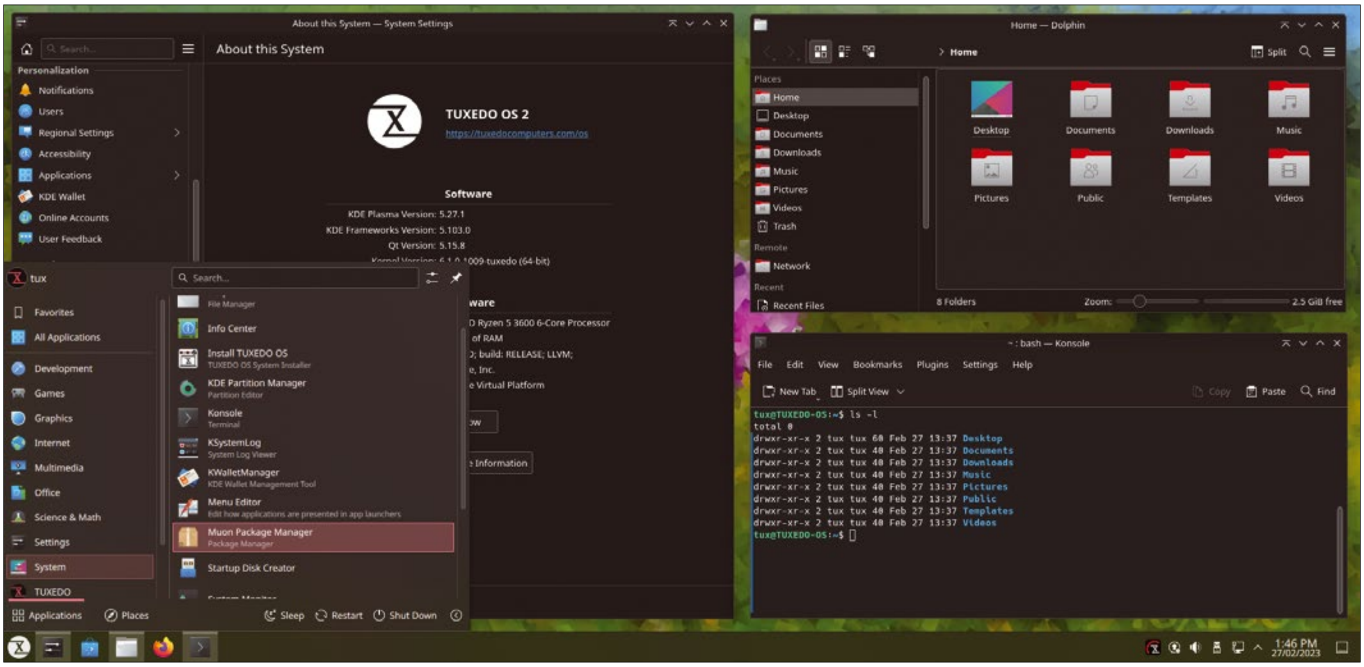
```
sudo apt install openssh-server
```

Although TUXEDO OS is an Ubuntu derivative, the developers are obviously not fans of the Snap [4] package manager favored by the Ubuntu project: The service you would need for using Snap (snapd) doesn't even make it onto the disc. Applications such as Firefox, which Ubuntu prefers or exclusively provides as a Snap package, are sourced from TUXEDO's own repository (Listing 1).

The Plasma Discover software manager looks like the modern app stores of other operating systems and offers detailed descriptions, screenshots, and reviews for many applications. However, it does not source all installable packages

Table 1: Preinstalled software

Software	Version
Kernel	6.1.0-1009
KDE Plasma	5.27.1
KDE Frameworks	5.103.0
Qt	5.15.8
LibreOffice	7.3.7.2
Firefox	110.0.1
Thunderbird	102.7.1
VLC	3.0.16
GCC	11.2.0



TUXEDO OS 2
https://tuxedocomputers.com/os

Software

KDE Plasma Version: 5.27.1
KDE Frameworks Version: 5.103.0
Qt Version: 5.15.8

```
tux@TUXEDO-OS:~$ ls -l
total 0
drwxr-xr-x 2 tux tux 60 Feb 27 13:37 Desktop
drwxr-xr-x 2 tux tux 40 Feb 27 13:37 Documents
drwxr-xr-x 2 tux tux 40 Feb 27 13:37 Downloads
drwxr-xr-x 2 tux tux 40 Feb 27 13:37 Music
drwxr-xr-x 2 tux tux 40 Feb 27 13:37 Pictures
drwxr-xr-x 2 tux tux 40 Feb 27 13:37 Public
drwxr-xr-x 2 tux tux 40 Feb 27 13:37 Templates
drwxr-xr-x 2 tux tux 40 Feb 27 13:37 Videos
tux@TUXEDO-OS:~$ 
```

Figure 3: If you prefer it dark, you can easily adjust the desktop.

Listing 1: TUXEDO OS repo

```
$ apt info firefox
[...]
APT-Sources: https://deb.tuxedocomputers.com/ubuntu jammy/main amd64 Packages
```

from the repositories, only applications in the narrower sense. On the other hand, you're not limited to .deb packages – Plasma Discovery also insludes Flatpak packages in the list of installable applications. You can add more Flatpak repositories in the Discover settings. You don't need root privileges to do so, by the way, because Flatpak packages are not installed globally across the system. For advanced users there is Muon (*System* | *Muon Package Manager*), a classical package manager, which also supplies library packages and smaller tools.

TUXEDO OS uses X11 as the basis for the desktop and not Wayland. However, detecting the monitor size proved problematic in the test. The fonts and controls turned out to be way too big. To find out how to fix the problem, see the "X11 Settings" box.

Gaming

The package repository maintained by TUXEDO Computers contains Lutris [5] and the Heroic Games Launcher [6]. Lutris lets you access your game libraries on Steam, Humble Bundle, Epic Games, and GOG in a common interface. Lutris also integrates various emulators. The package description does not tell you about the steps needed to set it up; however, a post on the Lutris forum [7] can help.

You won't be able to download Steam games on Lutris until you set up the Steam client. To access the Steam client directly, set up the *Steam installer* package in Discover and then running *Games* | *Steam*. An installer downloads the latest Steam version and launches the interface, letting you log in and download games. As soon as this is working, games can also be launched via the game overview in Lutris. Further notes on gaming can be found in an article on the TUXEDO website [8].

TUXEDO Tools

The start menu contains a *TUXEDO* category with two programs, TUXEDO Control Center and TUXEDO WebFAI Creator. These applications are accompanied by a command line tool named

X11 Settings

TUXEDO OS relies on X11 instead of Wayland, but you can add Wayland support retroactively. The developers justified their decision to go with X11 by arguing that X11 provides better compatibility with hybrid graphics setups (with a low-powered integrated graphics card and a powerful add-on card), which you will commonly find in TUXEDO notebooks.

Very large fonts and controls were preset on the test computer with a 27-inch monitor. The obvious way to fix this – right-clicking on the desktop, selecting Configure Display Settings, and resetting Global scaling from 150 to 100 percent – didn't improve things even after a reboot. KDE offers another possibility to adjust the font sizes: In the system settings under Appearance | Fonts you can Force font DPI. After enabling this option (with a value of 96 dpi) and logging in again, we were finally able to use the desktop.

However, what was actually responsible for the display issues in our lab was the following entry in the /etc/sddm.conf.d/kde_settings.conf file:

```
[X11]
EnableHiDPI=true
ServerArguments=-dpi 144
```

It told the X server to start with the -dpi 144 option. The installer obviously mistook the monitor for a high-resolution laptop display. To change to a good resolution, we just needed to remove the three offending lines or hash-tag them out (#).

TUXEDO Tomte. However, the use of these applications only makes sense when TUXEDO OS is used for the original purpose, i.e., when it is run on a computer made by TUXEDO.

The Control Center displays hardware information such as the CPU and GPU temperatures, as well as the fan speed (Figure 4). Profiles let you define settings for different scenarios (fan, power consumption, or computer performance). In the *System monitor* area you can define, for example, how many CPU cores the system uses and at what minimum and maximum clock frequencies. This also worked on our lab computer.

Two of these profiles can be enabled automatically when you switch to mains or battery operation on a laptop. Owners of TUXEDO hardware with external water cooling (TUXEDO Aquaris) can also control the fan speed and color LEDs of the device connected via Bluetooth in a separate tab [9].

The WebFAI Creator creates a USB stick that fully automates the process of installing TUXEDO OS and other distributions, along with different desktop variants of Ubuntu, OpenSuse, Manjaro, and elementary OS (with an encrypted or unencrypted file system in each case) on a computer. WebFAI downloads the files required for the installation from the web after you choose a system. You cannot change the configuration with this kind of installation. WebFAI is available in the boot menu of a TUXEDO OS installation so that, for example, you can reinstall a crashed system even without a USB boot stick.

TUXEDO tomte is a driver update tool for computers from TUXEDO Computers; it automatically searches for new driver versions for the device when booting the computer and automatically installs them.

If you are running TUXEDO OS on an unsupported machine, you can remove these components with the command from Listing 2. However, the menu entries in the separate *TUXEDO* submenus are retained, and calling these entries will cause error messages. To get rid of the menu items, launch the *Menu Editor*, select the *TUXEDO* entry, and press Delete. Confirming and then saving alleviates these worries.

Conclusions

TUXEDO OS cut a fine figure on our lab computer, booting faster than either Ubuntu or Windows on our test device. KDE and the standard applications run smoothly, and the interface looks well thought-out and preconfigured.

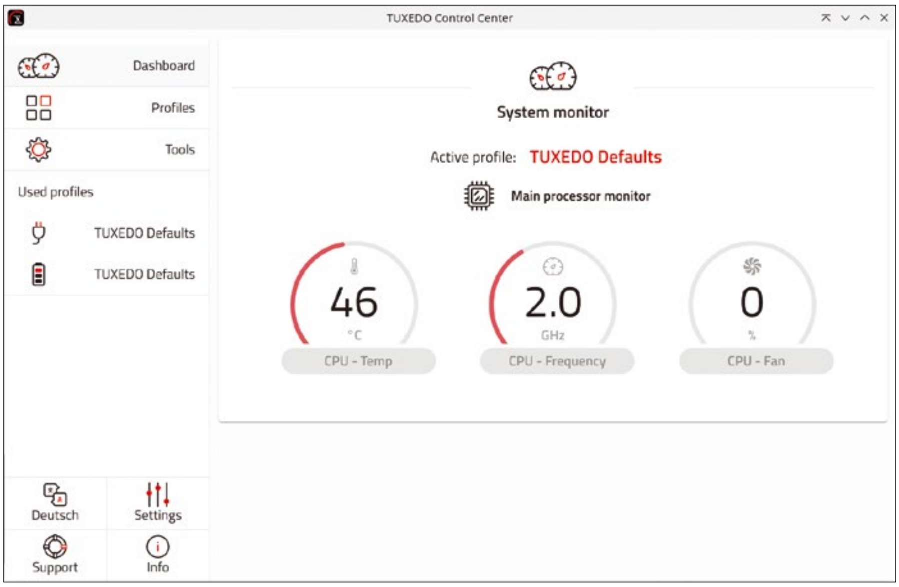
Some extras such as the TUXEDO Control Center cannot be used meaningfully on third-party devices but this was only to be expected. And you can uninstall the additional programs without any problems. We had no objections to the choice of pre-installed programs. If you don't like Ubuntu's Snap containers, you will be happy about their absence.

The Lutris gaming platform requires the presence of the Steam client to be fully functional. The steps required to integrate Steam into your game collection require some extended searching on the web.

Apart from this, the preview version of the latest Tuxedo operating system is already very usable, even without a Tuxedo laptop or PC. If you like to experiment with alternative Linux distributions, Tuxedo OS is an interesting option. ■■■

Listing 2: Removing TUXEDO Tools

```
$ sudo apt remove tuxedo-control-center tuxedo-webfai-creator tuxedo-tomte
```



Active profile: TUXEDO Defaults

Figure 4: The TUXEDO Control Center is intended for systems running on TUXEDO hardware.

Info

[1] TUXEDO OS ISO downloads: *https://os.tuxedocomputers.com*

[2] Calamares installer framework: *https://calamares.io*

[3] KDE Neon: *https://neon.kde.org*

[4] Canonical Snap: *https://snapcraft.io*

[5] Lutris: *https://lutris.net*

[6] Heroic Games Launcher: *https://github.com/Heroic-Games-Launcher/HeroicGamesLauncher*

[7] Linking Steam and Lutris: *https://forums.lutris.net/t/can-not-sync-to-steam-public-profile-error-at-first/4180*

[8] "Gaming with TUXEDO OS": *https://www.tuxedocomputers.com/en/Gaming-with-TUXEDO-OS.tuxedo*

[9] TUXEDO Aquaris: *https://www.tuxedocomputers.com/en/TUXEDO-Aquaris.tuxedo*

Analyze network traffic with Sniffnet

Traffic Monitor

Network traffic remains a closed book for many users. Sniffnet lets less experienced users monitor their network traffic with ease.

By Erik Bärwaldt

Monitoring network traffic is part of the admin's daily grind. Most admins use the graphical Wireshark tool because of its wealth of functions as well as its availability in most distributions' respositories. Because Wireshark is primarily aimed at professional users, you need in-depth network knowledge to use it. The new Sniffnet project targets less experienced users, providing meaningful results in a simple, intuitive, and clear-cut interface.

Installation

Sniffnet is a cross-platform Rust application [1]. If you don't have the Rust programming language on your system, first set it up with the command from line 2 of Listing 1. The installation routine requires some interactive input and prints some status messages at the prompt. You install Sniffnet itself in the terminal using the Rust package manager, Cargo (line 4).

If you are using a Linux distribution with a DEB package manager, you can install the precompiled DEB package [2] from the project's GitHub page instead; this removes the need to implement Rust first (line 6).

For the program to work correctly, you will definitely need to resolve some dependencies [3] (line 8), regardless of the installation path. When done, call Sniffnet with the `sudo sniffnet` command.

Listing 1: Install Sniffnet

```
01 ### Install Rust
02 $ curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
03 ### Set up Sniffnet via Rust
04 $ cargo install sniffnet
05 ### Install Sniffnet as a DEB
06 $ sudo apt install sniffnet_Linux.deb
07 ### Install dependencies
08 $ sudo apt install libpcap-dev libfontconfig libfontconfig1-dev
```

The start-up window opens with the filter options (Figure 1). On the right, you can specify the desired Internet protocol and transport protocols versions by enabling the appropriate radio buttons. In both cases, all available versions are enabled by default (i.e., IPv4 and IPv6, as well as TCP and UDP). On the far right, select the application protocol from more than a dozen options in a drop-down list. Again, Sniffnet takes all protocols into account in the default setting.

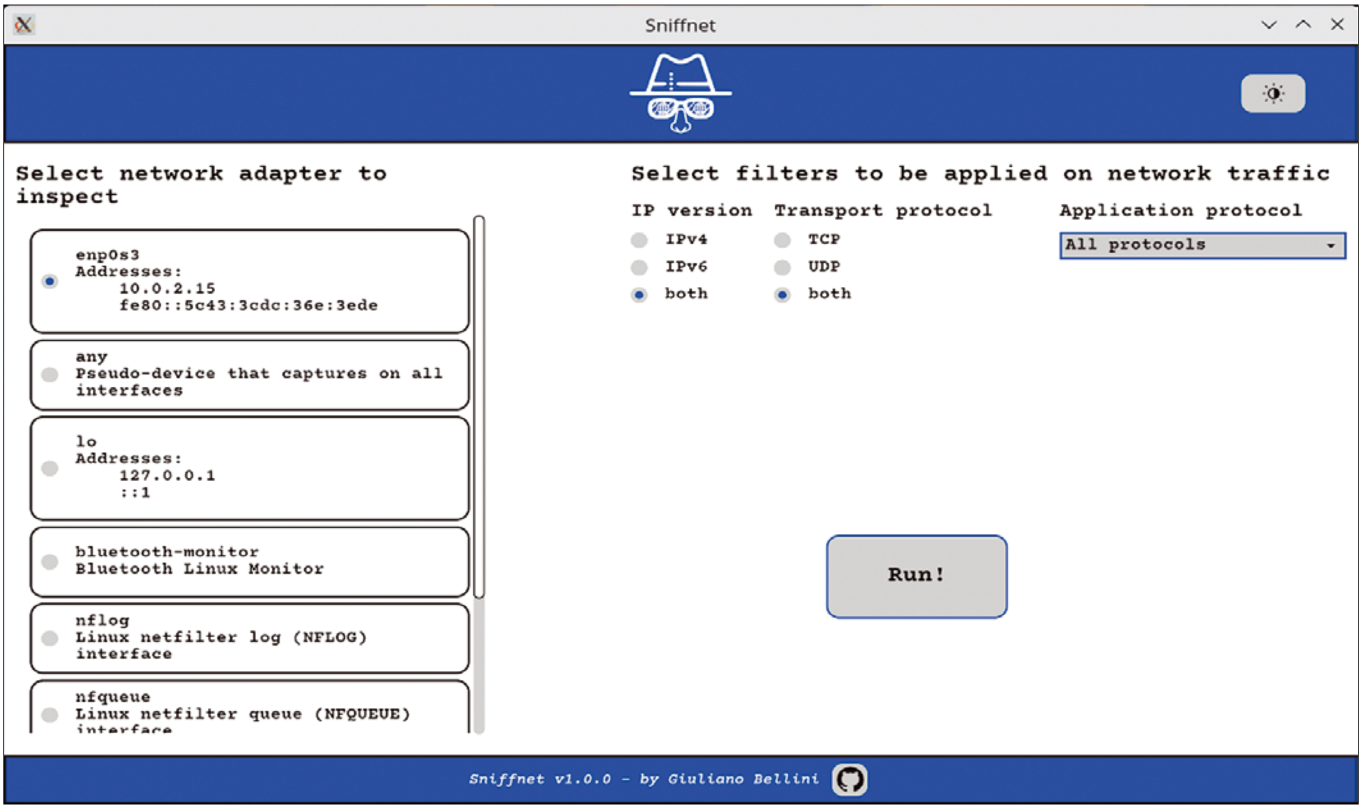
On the left of the start-up window, you will find the network interfaces available on the system. Sniffnet lists both the physical and virtual interfaces; Sniffnet always enables the first physical interface. Consequently, you may need to select the desired interface if this doesn't meet your needs. After adjusting the basic settings, click *Run!*

Sniffnet now takes a moment to acquire the available data and then opens the actual monitoring window (Figure 2). At the top, you will see a constantly updated graphical display showing the number of data packets passing through the active network interface per second. This display uses colors to differentiate between incoming and outgoing packets. Alternatively, you can display the transferred data volume in bytes per second by clicking the appropriate radio

Photo by Joey Kyber on Unsplash

button under *Plotted data*. In a small box to the right of the graphical display, you will find some statistical data about the network connection, including the number of filtered packets, their size in megabytes, and the filtered packets per application protocol.

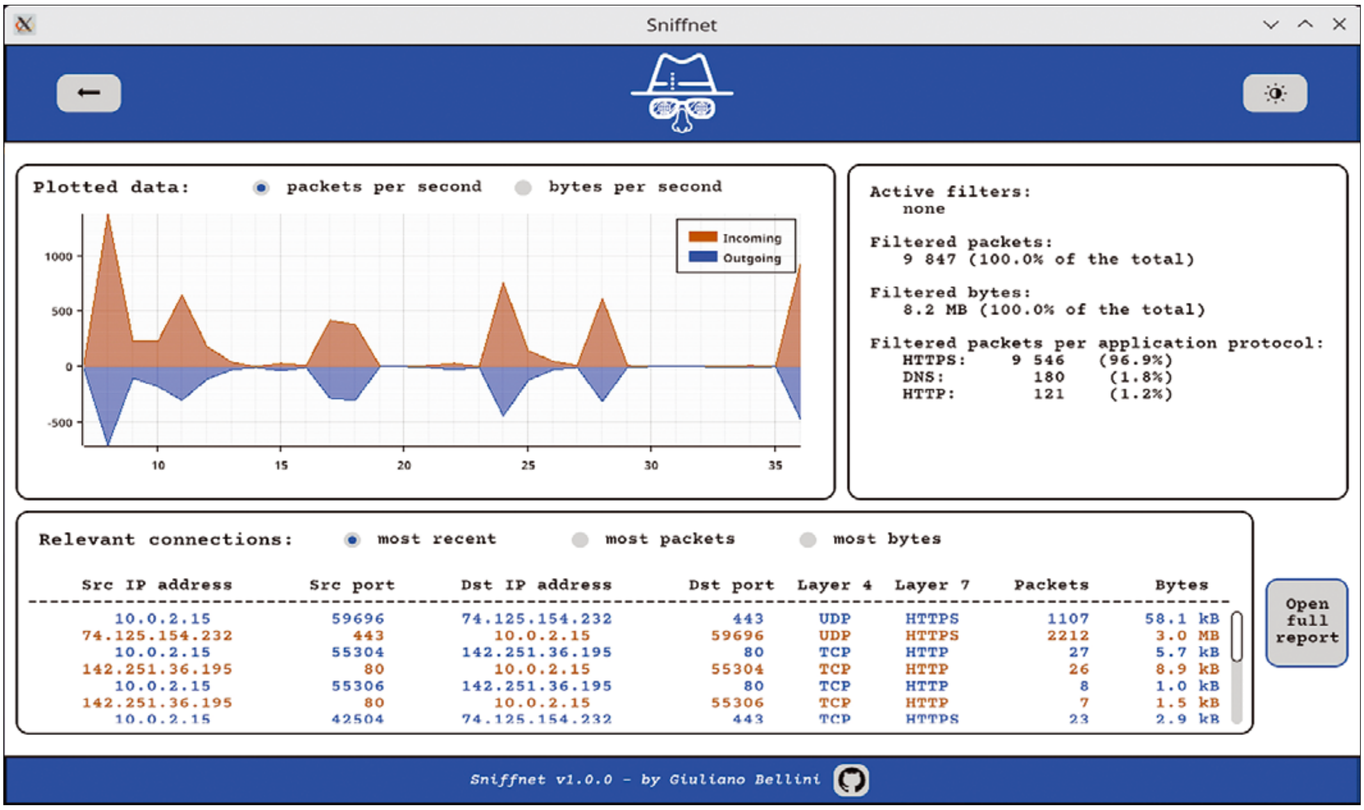
Perhaps the most interesting thing, the *Relevant connections* box beneath the graphical traffic display contains exact data on the individual connections, consisting of source and target addresses and the ports involved. In addition, you can see the transport and application protocol used for each connection, as



```
Select network adapter to
inspect

● enp0s3
  Addresses:
      10.0.2.15
      fe80::5c43:3cdc:36e:3ede

○ any
  Pseudo-device that captures on all
  interfaces

○ lo
  Addresses:
      127.0.0.1
      ::1

○ bluetooth-monitor
  Bluetooth Linux Monitor

○ nflog
  Linux netfilter log (NFLOG)
  interface

○ nfqueue
  Linux netfilter queue (NFQUEUE)
  interface
```

```
Select filters to be applied on network traffic
IP version     Transport protocol     Application protocol
○ IPv4         ○ TCP                  All protocols
○ IPv6         ○ UDP
● both         ● both
```

Figure 1: The very tidy Sniffnet start-up window only offers the essentials.



```
Active filters:
   none

Filtered packets:
   9 847 (100.0% of the total)

Filtered bytes:
   8.2 MB (100.0% of the total)

Filtered packets per application protocol:
   HTTPS:     9 546   (96.9%)
   DNS:         180    (1.8%)
   HTTP:        121    (1.2%)
```

```
Relevant connections:     ● most recent     ○ most packets     ○ most bytes

   Src IP address      Src port     Dst IP address     Dst port  Layer 4  Layer 7    Packets     Bytes
---------------------------------------------------------------------------------------------------
     10.0.2.15           59696      74.125.154.232        443     UDP      HTTPS       1107    58.1 kB
   74.125.154.232          443        10.0.2.15         59696     UDP      HTTPS       2212     3.0 MB
     10.0.2.15           55304      142.251.36.195         80     TCP      HTTP          27     5.7 kB
   142.251.36.195           80        10.0.2.15         55304     TCP      HTTP          26     8.9 kB
     10.0.2.15           55306      142.251.36.195         80     TCP      HTTP           8     1.0 kB
   142.251.36.195           80        10.0.2.15         55306     TCP      HTTP           7     1.5 kB
     10.0.2.15           42504      74.125.154.232        443     TCP      HTTPS         23     2.9 kB
```

Figure 2: Sniffnet summarizes all network traffic information in a single window.

well as the number of packets transmitted. On the far right, under the *Bytes* column, you will also find the volume of data that has been transferred per connection.

Sniffnet also color codes the data in the monitoring window. Incoming connections are shown in a reddish-brown, while outgoing ones are colored blue. You can change this view by clicking a radio button: Click the *most packets* radio button to display the connections by the number of packets sent or received, or click the *most bytes* button to sort all packets in descending order by the volume of data transferred.

Complete

In its graphical display, Sniffnet always shows you the most recent data transfers. To track the network traffic over time, Sniffnet logs all connections and their activities as a table in a text file named `$HOME/sniffnet_report/report.txt`. To view this text file, click on *Open full report* to open it in your desktop environment's standard text editor (Figure 3). The application updates the log in near real time.

If you want to change something in the settings during a run, click on the arrow icon in the upper left corner (Figure 2); this will take you back to the main window. It is important to note that Sniffnet deletes the logfile on every restart. If needed, you can save the logs after terminating the application or before restarting the software.

Conclusions

If you want to see what's happening with your network traffic, the largely self-explanatory Sniffnet offers a simple solution for all levels of users. However, the tool lacks analysis capabilities; if you are troubleshooting problems with your network connection, you will need to check the logs manually.

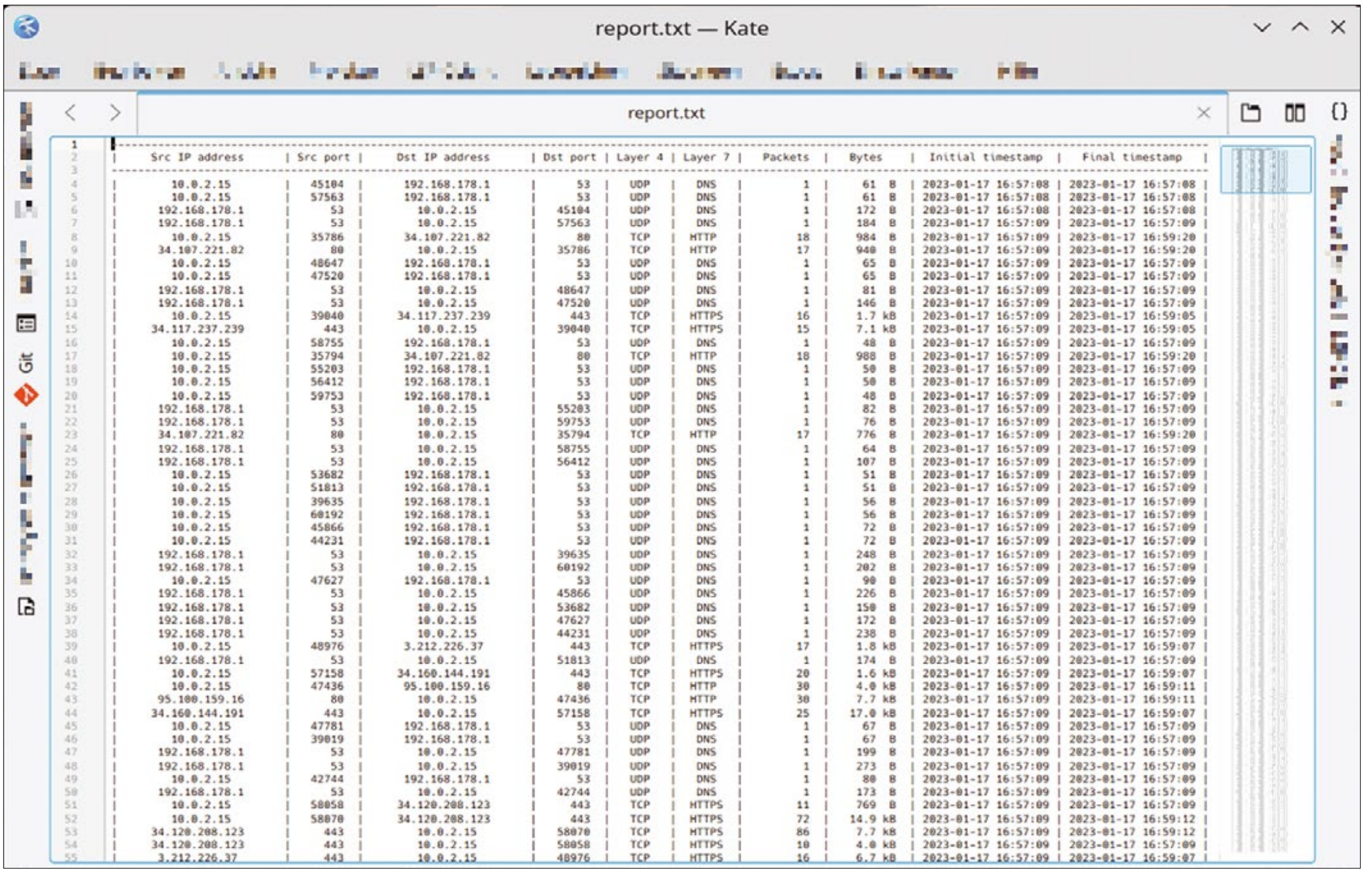
During testing, I found a few bugs. On Kubuntu 22.04, the software refused to open the report when I pressed the *Open full report* button, but you can work around this by manually opening the file without any problems. A more serious issue: Sniffnet deletes the logfiles almost arbitrarily. Furthermore, Sniffnet does not resolve IP addresses into names – you have to manually find the name for `104.26.7.95`. Finally, you cannot copy sequences from the live log; you need to extract them after generating a logfile (if it survives the session). Clearly, Sniffnet still has great potential for improvement. ■■■

Info

[1] Sniffnet: *https://github.com/GyulyVGC/sniffnet*

[2] DEB package: *https://github.com/GyulyVGC/sniffnet/releases*

[3] Installation instructions: *https://github.com/GyulyVGC/sniffnet#installation*



```
-----------------------------------------------------------------------------------------------------------------------------------------------
|   Src IP address   |Src port|   Dst IP address   |Dst port |Layer 4|Layer 7| Packets |  Bytes   |  Initial timestamp  |   Final timestamp   |
-----------------------------------------------------------------------------------------------------------------------------------------------
|     10.0.2.15      |  45104 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    61 B  | 2023-01-17 16:57:08 | 2023-01-17 16:57:08 |
|     10.0.2.15      |  57563 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    61 B  | 2023-01-17 16:57:08 | 2023-01-17 16:57:08 |
|   192.168.178.1    |     53 |     10.0.2.15      |   45104 |  UDP  |  DNS  |       1 |   172 B  | 2023-01-17 16:57:08 | 2023-01-17 16:57:08 |
|   192.168.178.1    |     53 |     10.0.2.15      |   57563 |  UDP  |  DNS  |       1 |   184 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  35786 |   34.107.221.82    |      80 |  TCP  | HTTP  |      18 |   984 B  | 2023-01-17 16:57:09 | 2023-01-17 16:59:20 |
|   34.107.221.82    |     80 |     10.0.2.15      |   35786 |  TCP  | HTTP  |      17 |   940 B  | 2023-01-17 16:57:09 | 2023-01-17 16:59:20 |
|     10.0.2.15      |  48647 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    65 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  47520 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    65 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   192.168.178.1    |     53 |     10.0.2.15      |   48647 |  UDP  |  DNS  |       1 |    81 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   192.168.178.1    |     53 |     10.0.2.15      |   47520 |  UDP  |  DNS  |       1 |   146 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  39040 |   34.117.237.239   |     443 |  TCP  | HTTPS |      16 |   1.7 kB | 2023-01-17 16:57:09 | 2023-01-17 16:59:05 |
|   34.117.237.239   |    443 |     10.0.2.15      |   39040 |  TCP  | HTTPS |      15 |   7.1 kB | 2023-01-17 16:57:09 | 2023-01-17 16:59:05 |
|     10.0.2.15      |  58755 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    48 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  35794 |   34.107.221.82    |      80 |  TCP  | HTTP  |      18 |   988 B  | 2023-01-17 16:57:09 | 2023-01-17 16:59:20 |
|     10.0.2.15      |  55203 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    50 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  56412 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    50 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  59753 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    48 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   192.168.178.1    |     53 |     10.0.2.15      |   55203 |  UDP  |  DNS  |       1 |    82 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   192.168.178.1    |     53 |     10.0.2.15      |   59753 |  UDP  |  DNS  |       1 |    76 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   34.107.221.82    |     80 |     10.0.2.15      |   35794 |  TCP  | HTTP  |      17 |   776 B  | 2023-01-17 16:57:09 | 2023-01-17 16:59:20 |
|   192.168.178.1    |     53 |     10.0.2.15      |   58755 |  UDP  |  DNS  |       1 |    64 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   192.168.178.1    |     53 |     10.0.2.15      |   56412 |  UDP  |  DNS  |       1 |   107 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  53682 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    51 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  51813 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    51 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  39635 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    56 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  60192 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    56 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  45866 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    72 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  44231 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    72 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   192.168.178.1    |     53 |     10.0.2.15      |   39635 |  UDP  |  DNS  |       1 |   248 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   192.168.178.1    |     53 |     10.0.2.15      |   60192 |  UDP  |  DNS  |       1 |   202 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  47627 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    90 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   192.168.178.1    |     53 |     10.0.2.15      |   45866 |  UDP  |  DNS  |       1 |   226 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   192.168.178.1    |     53 |     10.0.2.15      |   53682 |  UDP  |  DNS  |       1 |   150 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   192.168.178.1    |     53 |     10.0.2.15      |   47627 |  UDP  |  DNS  |       1 |   172 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   192.168.178.1    |     53 |     10.0.2.15      |   44231 |  UDP  |  DNS  |       1 |   238 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  48976 |    3.212.226.37    |     443 |  TCP  | HTTPS |      17 |   1.8 kB | 2023-01-17 16:57:09 | 2023-01-17 16:59:07 |
|   192.168.178.1    |     53 |     10.0.2.15      |   51813 |  UDP  |  DNS  |       1 |   174 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  57158 |   34.160.144.191   |     443 |  TCP  | HTTPS |      20 |   1.6 kB | 2023-01-17 16:57:09 | 2023-01-17 16:59:07 |
|     10.0.2.15      |  47436 |   95.100.159.16    |      80 |  TCP  | HTTP  |      30 |   4.0 kB | 2023-01-17 16:57:09 | 2023-01-17 16:59:11 |
|   95.100.159.16    |     80 |     10.0.2.15      |   47436 |  TCP  | HTTP  |      30 |   7.7 kB | 2023-01-17 16:57:09 | 2023-01-17 16:59:11 |
|   34.160.144.191   |    443 |     10.0.2.15      |   57158 |  TCP  | HTTPS |      25 |  17.0 kB | 2023-01-17 16:57:09 | 2023-01-17 16:59:07 |
|     10.0.2.15      |  47781 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    67 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  39019 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    67 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   192.168.178.1    |     53 |     10.0.2.15      |   47781 |  UDP  |  DNS  |       1 |   199 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   192.168.178.1    |     53 |     10.0.2.15      |   39019 |  UDP  |  DNS  |       1 |   273 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  42744 |   192.168.178.1    |      53 |  UDP  |  DNS  |       1 |    80 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|   192.168.178.1    |     53 |     10.0.2.15      |   42744 |  UDP  |  DNS  |       1 |   173 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  58058 |   34.120.208.123   |     443 |  TCP  | HTTPS |      11 |   769 B  | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|     10.0.2.15      |  58070 |   34.120.208.123   |     443 |  TCP  | HTTPS |      72 |  14.9 kB | 2023-01-17 16:57:09 | 2023-01-17 16:59:12 |
|   34.120.208.123   |    443 |     10.0.2.15      |   58070 |  TCP  | HTTPS |      86 |   7.7 kB | 2023-01-17 16:57:09 | 2023-01-17 16:59:12 |
|   34.120.208.123   |    443 |     10.0.2.15      |   58058 |  TCP  | HTTPS |      10 |   4.0 kB | 2023-01-17 16:57:09 | 2023-01-17 16:57:09 |
|    3.212.226.37    |    443 |     10.0.2.15      |   48976 |  TCP  | HTTPS |      16 |   6.7 kB | 2023-01-17 16:57:09 | 2023-01-17 16:59:07 |
```

Figure 3: All connections and activities on the network interface can also be viewed in text form.

DrupalCon
PITTSBURGH**2023**
5-8 JUNE

David L. Lawrence Convention Center
events.drupal.org/pittsburgh2023

Meet thousands of individuals from around the world who use, develop, design and support the Drupal platform

The value of DrupalCon is in the perspectives, energy, and diversity of experiences participants share—**join us!**

The disaster of MIPI cameras on Linux

Tinker Time

Linux users have long gotten used to the standard hardware in their systems working perfectly. Recently, however, things have gotten dicey for webcams supporting the MIPI specification. We'll tell you why and what to do about it. *By Martin Loschwitz*

Linux and hardware haven't always been the best of friends. Just 20 years ago, you couldn't just buy a mainstream computer and expect it to work when you installed Linux. If wanted to be sure that your new acquisition would play nicely with Linux, you often had weeks of very detailed research ahead of you. Which chipsets were supported? Which modems, printers, sound cards, and network adapters had the necessary drivers?

If you didn't check carefully, the worst case is that you would be left with hardware that was practically impossible to get to work with Linux. Just remember the notorious winmodems (Figure 1) or printers that did nothing without binary drivers that the vendor didn't bother to provide for Linux systems. If you are looking for proof that not everything was better in the past, the Linux hardware saga is a rich source.

Fortunately, the times are changing. Individual computer components are increasingly merging into a single logical unit. Sound cards became standard components at some point and supported use on Linux with appropriate drivers. Graphics chipsets, also an eternal nuisance, now come with support from NVIDIA or AMD. If you don't play games, you can make do with the chipsets integrated in AMD and Intel CPUs, which also perform reliably on Linux. Hardly anyone needs modems anymore, but LTE or 5G modems can usually be fed `at` commands via an emulated serial

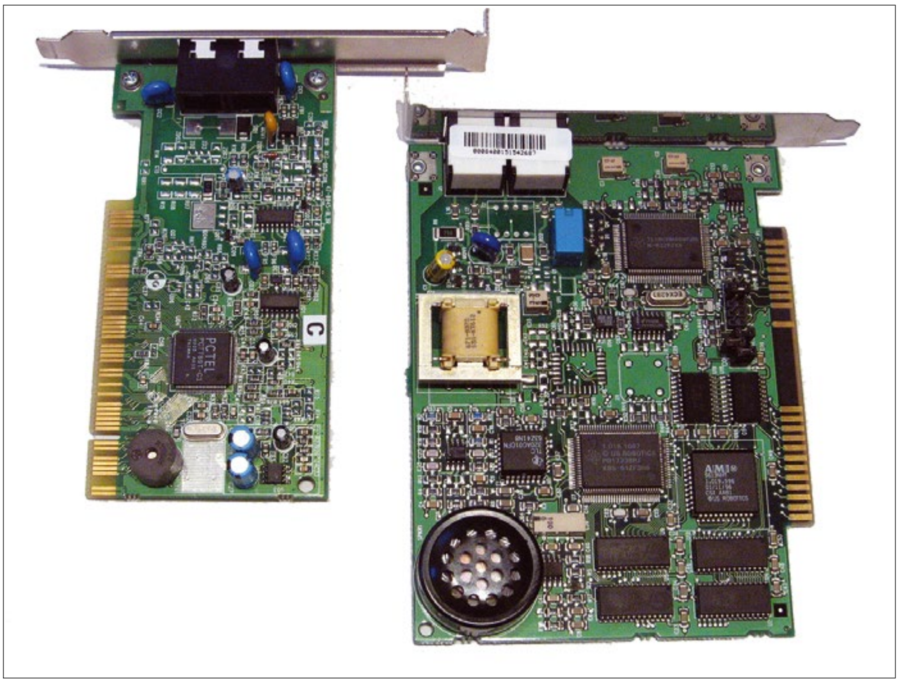


Figure 1: Softmodems or winmodems were a nuisance in the early years of Linux development because they didn't work without proprietary drivers. Over 20 years later, Intel has done the same stunt with webcams. Wikipedia, Douglas Whitaker, CC BY-SA 2.5

Photo by Clark Young on Unsplash

interface. Printers usually have their own network connection and offer support for KPL/PD or PostScript. Even for devices connected locally via USB, wide ranging driver support exists today. All is well, you might think, in the world of Linux and modern hardware.

In 2023, it seems strange that Linux users are again scrambling to compile kernel modules, battling with broken drivers on their desktops, and using elaborate work-arounds. This epidemic has been triggered by hardware that hardly anyone suspected would have come up lacking in support: webcams.

The Unknown Giant

The recent turn of events in webcams has its origins in something that took place more than 20 years ago: the founding of the Mobile Industry Processor Interface Alliance, or MIPI for short. MIPI was established in 2003 with the aim of creating protocol and interface standards for the interaction of processors and peripheral devices.

Most admins assume the term *peripherals* refers to mice, keyboards, printers, scanners, and similar devices, but the common definition also includes monitors and attachments such as external sound cards or webcams. For quite a while, MIPI didn't get much publicity or media coverage. But today, the consortium includes more than 300 manufacturers, so MIPI has definitely become a force in the industry.

One of MIPI's main fields of activity is defining protocols for the exchange of images and sounds between a host and the peripheral device. The CSI protocols are some of the most widely used protocols in the MIPI Alliance. CSI stands for Camera Serial Interface, and three versions of it are currently available. The most popular is undoubtedly 2019, the latest revision of MIPI CSI-2. CSI-2 is still used in many MIPI-compatible cameras today.

But how does a "normal" webcam differ from a MIPI device? In simple terms, CSI simply defines the interface that the camera and host use to exchange data. Like all interface definitions, CSI has had to adapt to the fact that data volumes have grown over the years. Today's webcams often need to handle 4K streams. MIPI CSI-2 from 2019 already supports significantly higher bandwidths than legacy USB 3.0 with its realistic 3.6 GB/s throughput. But the usable bandwidth is not the only difference between typical old-style webcams and the MIPI system.

The MIPI standard explicitly stipulates that the camera sensors and the components responsible for image evaluation must not be combined in the same assembly. This means that, in MIPI setups, the camera uses its sensors to produce raw audio and image data, which it then passes through to the host. The host handles the brunt of the work, computing an image with sound, which can then be used for video conferencing or other activities.

Several factors play an important role. On the one hand, decoupling the sensor from the chip enables a variety of combinations that can be adapted to the application. Inexpensive notebooks contain equally inexpensive video processing chips, mid-range devices use inexpensive chips with expensive sensors, and high-end machines use the best they can get – or so the theory goes (Figure 2).

On the other hand, this setup means you can integrate special chips that not only assemble an image for the webcam but also process the data, for example, with AI support. The famous AI problem that automatically calculates the number of bees in a hive using a webcam benefits massively from the MIPI interface: The data arrives in precisely the raw format that the computer needs to process directly, instead of fiddling around with a mess of pixels from a mid-range webcam.

The Crux of the Matter

Thus far, the gradual migration to MIPI technology seems very promising, but appearances are deceiving. The reason for the problems Linux users are having with their webcams is related to the image processing chips that comply with the MIPI specification. The CSI-2 protocol itself causes virtually no problems. The Linux kernel has supported it for quite some time and has no trouble supporting a data exchange between peripherals and the MIPI chip (Figure 3).

The situation is not quite so harmonious with the image processing chips, which have been making inroads into the current Intel and AMD CPU specifications for some time now. The IPU6 image processing unit included with Intel's "Tiger Lake" and "Alder Lake" CPU series specifications is explicitly intended as a potential CPU component for a MIPI-compatible camera based on the CSI-2 standard. The crux of the matter is that, to use the IPU6 chip, you need proprietary firmware and a special driver.

Firmware is unlikely to faze most Linux users; after all, there are quite a few drivers that you can only use with proprietary firmware. The kernel developers and the major distributors have long since found ways to cushion this problem from the user's point of view in the form of sensible workarounds. But things are different with IPU6.

Although Intel provides a driver for the cameras, this driver does not work well. For the current Linux kernel,

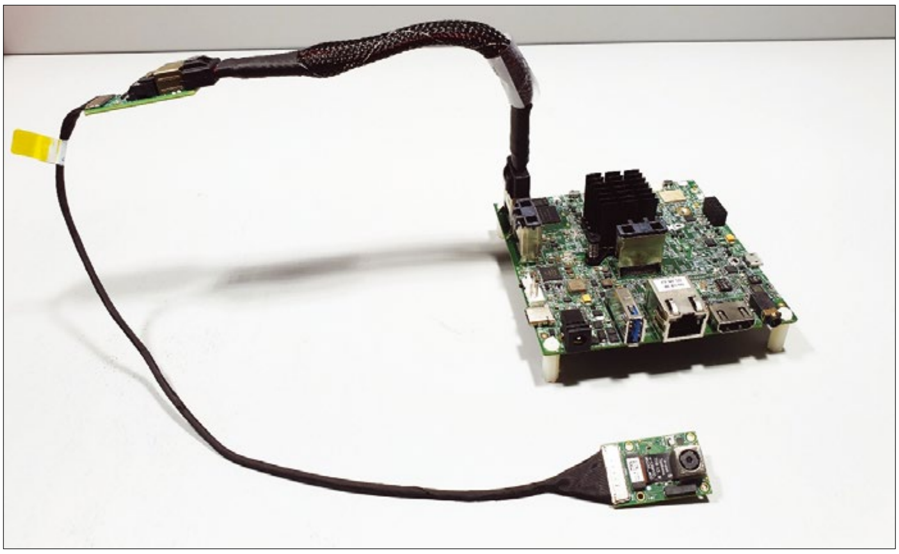


Figure 2: MIPI CSI-2 describes an interface for communication between peripheral devices and host chips so that they can be logically separated from each other and combined as required. © NXP Semiconductors

there is no trouble-free way to build it. To make things worse, the way it is written means that it cannot be integrated into the official Linux kernel in the foreseeable future and must therefore be considered an out-of-tree development. This prompted kernel veteran Greg Kroah-Hartman to explicitly warn users against buying new devices with Intel's "Alder Lake" CPUs and MIPI webcams [1]. Usable support under Linux is unlikely in the foreseeable future, he emphasized.

If you take a closer look at the situation on Lenovo's current 10th-generation flagship X1 Carbon, the extent of the disaster quickly becomes clear. After booting a recent desktop Linux, such as openSUSE or Fedora, the device's built-in webcam is simply not available. The webcam appears in the list of integrated hardware, but no driver is loaded. A usable webcam is also conspicuously absent in tools such as Google Meet. If you have a lemon like this on your desk, you can only take part in video conferences without sending video images – which is farcical given that we are living in the third decade of the 21st century.

To make things even worse, without resorting to a compiler, this debacle is practically impossible to clean up. One of the Linux community's greatest achievements over the past two decades has been to make messing around with drivers on consumer hardware unnecessary. When I first started exploring Linux in 1998, it was nothing out of the ordinary to build a custom kernel because of a special driver or even a single important patch. However, distribution kernels have steadily become more reliable and complete in recent years. Kernel building skills are likely to be rusty even among old hands.

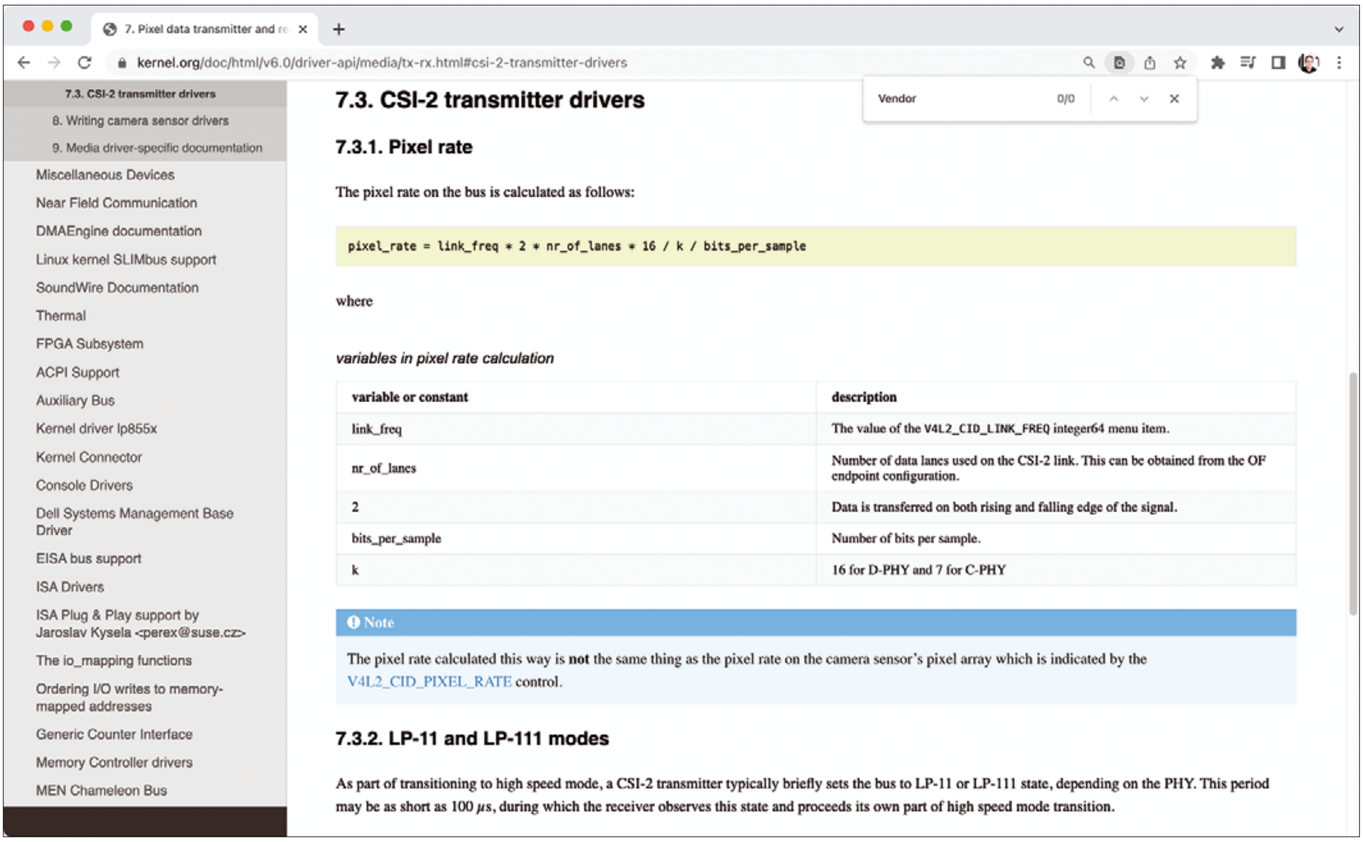
Ways Out

All told, the sparsely documented approaches to getting an IPU6 chip to work on Linux are not very encouraging. Users who are brave enough to tackle this problem have a real marathon ahead of them if they ever want to use the webcam.

Martin Sofaru, who owns a seventh-generation Lenovo X1 Yoga, took on the task of solving the IPU6 problem out of personal interest. He describes the process that ultimately led to success in several bug reports on GitHub.

Everything starts with the Linux kernel. The kernel is not always able to read the correct CSI pin assignment from the system's ACPI information and requires a patch [2] by Red Hat employee Hans de Goede (Figure 4). You then need to add another code patch [3] for current kernels to ensure that Linux uses the IPU6 chip's IOMMU interface instead of unsuccessfully placing it under the aegis of the CPU's IOMMU management. Finally, a third patch [4] ensures that the Linux kernel detects ACPI device LATT2021, which is installed in some laptops with an IPU6 chip and without which the camera cannot be controlled.

First, pick up the source code for your current kernel and check whether it already contains the three code additions, or if you need to apply them. Canonical has now included at least the first two patches in its own kernel for Ubuntu 22.10. This was not the case for openSUSE and Fedora, at least at the time this article went to press. If in doubt, your only option is to pick up


7. Pixel data transmitter and re
kernel.org/doc/html/v6.0/driver-api/media/tx-rx.html#csi-2-transmitter-drivers
Vendor
0/0
7.3. CSI-2 transmitter drivers
8. Writing camera sensor drivers
9. Media driver-specific documentation
Miscellaneous Devices
Near Field Communication
DMAEngine documentation
Linux kernel SLIMbus support
SoundWire Documentation
Thermal
FPGA Subsystem
ACPI Support
Auxiliary Bus
Kernel driver lp855x
Kernel Connector
Console Drivers
Dell Systems Management Base Driver
EISA bus support
ISA Drivers
ISA Plug & Play support by Jaroslav Kysela <perex@suse.cz>
The io_mapping functions
Ordering I/O writes to memory-mapped addresses
Generic Counter Interface
Memory Controller drivers
MEN Chameleon Bus

7.3. CSI-2 transmitter drivers

7.3.1. Pixel rate

The pixel rate on the bus is calculated as follows:

```
pixel_rate = link_freq * 2 * nr_of_lanes * 16 / k / bits_per_sample
```

where

variables in pixel rate calculation

variable or constant	description
link_freq	The value of the `V4L2_CID_LINK_FREQ` integer64 menu item.
nr_of_lanes	Number of data lanes used on the CSI-2 link. This can be obtained from the OF endpoint configuration.
2	Data is transferred on both rising and falling edge of the signal.
bits_per_sample	Number of bits per sample.
k	16 for D-PHY and 7 for C-PHY

Note

The pixel rate calculated this way is **not** the same thing as the pixel rate on the camera sensor's pixel array which is indicated by the V4L2_CID_PIXEL_RATE control.

7.3.2. LP-11 and LP-111 modes

As part of transitioning to high speed mode, a CSI-2 transmitter typically briefly sets the bus to LP-11 or LP-111 state, depending on the PHY. This period may be as short as 100 μs, during which the receiver observes this state and proceeds its own part of high speed mode transition.

Figure 3: The Linux kernel is quite capable of handling the CSI-2 specification and also implements a transport path for it. However, it has trouble with the Intel drivers. Users are therefore forced to start tinkering with their systems.

the source code for the kernel you are currently running and compare the code patches item by item.

Unfortunately, after customizing the kernel, you have only completed about 10 percent of the work. Once the kernel is up and running, the next step is to compile the required drivers and install the proprietary firmware. Intel delivers the firmware in its own Git directory [5], and an installation guide tells you what to do: Copy the files from the `include/` and `/lib` folders to `/usr/include/` and `/usr/lib/`.

This illustrates another dimension of the mess. The proprietary components for IPU6 include not only the firmware itself, but also a bunch of userland libraries for accessing IPU6. Ubuntu and Arch Linux offer corresponding packages, but Ubuntu's packages are provided externally by the OEM Solutions Group [6]. openSUSE and Fedora, on the other hand, require far more manual work. You need to retrieve the files directly from GitHub and unpack them on your system.

The story is similar for the driver packages. Ubuntu and Arch Linux have packages available for building the modules for your own kernel with just a few commands. Ubuntu unsurprisingly uses DKMS (Dynamic Kernel Management Support), which offers multiple benefits. Complementing the module sources themselves, the OEM Solutions Group has also included some patches from the Intel module code bug reports, without which the kernel module simply won't work.

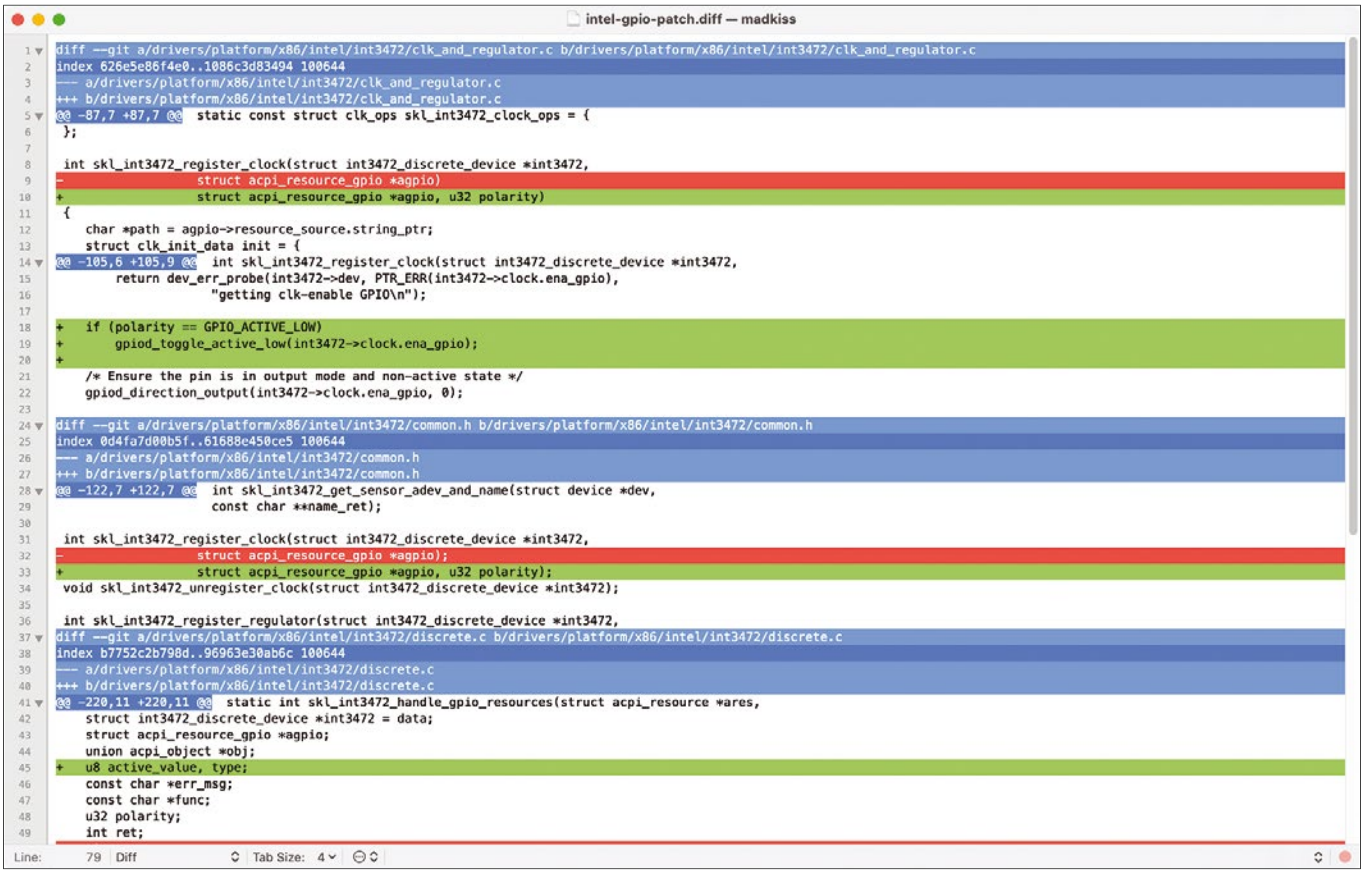
Intel's official GitHub repository is obviously not a hive of activity and the company doesn't seem to want to make it easy to use its hardware on Linux. Once again, Fedora and openSUSE users will have more work to do than users of some other distributions.

Kernel Modules

If you have made it this far, you now face the next challenge: building IPU6 modules that are suitable for your own kernel from Intel's sources. Again, Hans de Goede has been extremely helpful with patches and many comments, which ultimately give you semi-finished to-go solutions for individual distributions. Ubuntu has all the patches required and available for the current drivers in its program as DKMS packages (again in the OEM Solutions Repository); you can use these packages to build the essential kernel module in the usual way.

If your distribution does not offer this solution, once again the only option is to download the sources from GitHub and complete the manual build process. Bitter pill: It is precisely because Intel puts so little effort into maintaining its IPU6 drivers that many IDs of chipsets for IPU6 cameras that are quite common in the wild are missing from the headers. For example, quite a few Dell models use IPU6 chips, which the driver theoretically supports but fails to identify due to ID changes.

Again, you might very well be left out in the cold and have no choice but to do it all yourself. This requires additional knowledge, of course; not only do you have to patch the kernel; you also have to develop the patch yourself. After all these setbacks, if you do get to the point where the driver actually loads, you still have a long path ahead of you.



```
diff --git a/drivers/platform/x86/intel/int3472/clk_and_regulator.c b/drivers/platform/x86/intel/int3472/clk_and_regulator.c
index 626e5e86f4e0..1086c3d83494 100644
--- a/drivers/platform/x86/intel/int3472/clk_and_regulator.c
+++ b/drivers/platform/x86/intel/int3472/clk_and_regulator.c
@@ -87,7 +87,7 @@ static const struct clk_ops skl_int3472_clock_ops = {
 };

 int skl_int3472_register_clock(struct int3472_discrete_device *int3472,
-                  struct acpi_resource_gpio *agpio)
+                  struct acpi_resource_gpio *agpio, u32 polarity)
 {
    char *path = agpio->resource_source.string_ptr;
    struct clk_init_data init = {
@@ -105,6 +105,9 @@ int skl_int3472_register_clock(struct int3472_discrete_device *int3472,
        return dev_err_probe(int3472->dev, PTR_ERR(int3472->clock.ena_gpio),
                    "getting clk-enable GPIO\n");

+   if (polarity == GPIO_ACTIVE_LOW)
+       gpiod_toggle_active_low(int3472->clock.ena_gpio);
+
    /* Ensure the pin is in output mode and non-active state */
    gpiod_direction_output(int3472->clock.ena_gpio, 0);

diff --git a/drivers/platform/x86/intel/int3472/common.h b/drivers/platform/x86/intel/int3472/common.h
index 0d4fa7d00b5f..61688e450ce5 100644
--- a/drivers/platform/x86/intel/int3472/common.h
+++ b/drivers/platform/x86/intel/int3472/common.h
@@ -122,7 +122,7 @@ int skl_int3472_get_sensor_adev_and_name(struct device *dev,
                    const char **name_ret);

 int skl_int3472_register_clock(struct int3472_discrete_device *int3472,
-                  struct acpi_resource_gpio *agpio);
+                  struct acpi_resource_gpio *agpio, u32 polarity);
 void skl_int3472_unregister_clock(struct int3472_discrete_device *int3472);

 int skl_int3472_register_regulator(struct int3472_discrete_device *int3472,
diff --git a/drivers/platform/x86/intel/int3472/discrete.c b/drivers/platform/x86/intel/int3472/discrete.c
index b7752c2b798d..96963e30ab6c 100644
--- a/drivers/platform/x86/intel/int3472/discrete.c
+++ b/drivers/platform/x86/intel/int3472/discrete.c
@@ -220,11 +220,11 @@ static int skl_int3472_handle_gpio_resources(struct acpi_resource *ares,
    struct int3472_discrete_device *int3472 = data;
    struct acpi_resource_gpio *agpio;
    union acpi_object *obj;
+   u8 active_value, type;
    const char *err_msg;
    const char *func;
    u32 polarity;
    int ret;
```

Figure 4: If your own distributor has not already integrated the patches needed for IPU6, the first step on the way to using your webcam is to patch the kernel.

Helper Construct

This is far from the end of the journey to the absurd for Ubuntu and Arch Linux users. Even if a fully patched kernel with all required libraries, firmware files, and correctly patched drivers is available, and the kernel successfully displays the new camera in dmesg, if you try to access it from Firefox or Chrome, for example, you will see only a black screen. The reason for this is that Intel has decided that the in-house IPU6 drivers do not need a compatibility interface for Video4Linux (V4L). Unfortunately, the V4L framework is almost exclusively used on Linux to integrate external video components such as webcams.

As if you hadn't done enough work, the next step is to install a relay daemon for V4L. What this does is, on the one hand, pick up the signal from the IPU6 chip, which is now hopefully working, and, on the other, pass it to a V4L pseudo device where the other programs can grab it in the usual way. This helper construct is reason enough for a moderately severe fit of rage, but getting over-excited will not help. If you want to use the integrated MIPI cam on your new current laptop, you have to tinker on.

But for all of this messing around in userspace to be fruitful, you first need yet another component. `ipu6-camera-hal` makes the IPU6 interface visible to the V4L relay daemon on the host's filesystem, and this, in turn, makes the interface usable. Again, Ubuntu and Arch Linux offer the software as a package via their repository paths, whereas Fedora and openSUSE users have to go through another round of tinkering.

Particularly grotesque: Patches for the V4L2 loopback pseudo-device, which ultimately allows conventional tools to access the camera, are circulating in bug reports for IPU6. The same is true for the relay daemon, which is actually a Ubuntu tool, but even there it will not interact with the remaining components out of the box. In its default configuration, the parameters in its systemd unit do not fit the bill; it launches but does not identify the IPU6 camera as a potential communication partner.

Just before the editorial deadline, the right packages with all the necessary patches to get an IPU6 up and running on a 10th-generation Lenovo X1 Carbon arrived in the OEM Solutions PPA [6] from Ubuntu that I mentioned earlier on. A look at the code in this repository could possibly help you rebuild the setup on another distribution.

On Shaky Legs

The fact that it was possible to use the camera to its full extent on Linux at some point in my test can ultimately only be viewed as a Pyrrhic victory. The entire construct stands on shaky legs. For example, you need to block automatic system updates for your home-built kernel in the package manager. Otherwise, the distribution will replace the work of art you so painstakingly cobbled together with the standard kernel the next time a vulnerability is fixed.

On the other hand, you don't want to be running an old kernel, especially on desktops and laptops. The risk of becoming a victim of an attack through an unfixed vulnerability increases significantly with each attack that becomes public. The fact that kernel guru Greg Kroah-Hartman has already given the thumbs down to the driver in its current form is not encouraging for the future. Intel would probably have to do an extensive code redesign to integrate the driver into the mainline kernel at some point. However, it seems unlikely that this will happen in the foreseeable future given the current state of the driver.

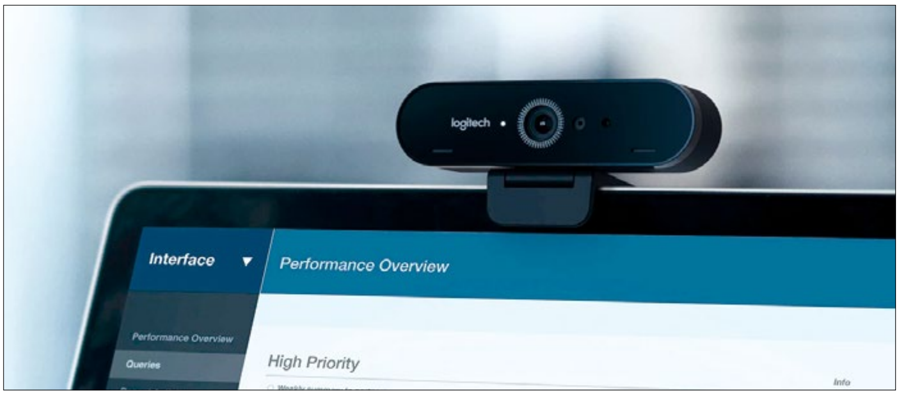


Figure 5: If you have a laptop with an IPU6 MIPI chip by Intel, you will be far better off with an external webcam than trying to set up the drivers on the system. © Logitech

Conclusions

Devices with the IPU6 chipset are definitely something to avoid as a Linux user, even if it feels wrong to even make such a statement in the year 2023. It is Intel's duty to develop suitable drivers, maintain them, and integrate them into Linux if the company is looking to release hardware onto the market by citing specifications.

If you already own a computer with an IPU6 chip and a MIPI-based cam, it is best to switch to a basic external webcam that offers better support for Linux (Figure 5). Webcams are readily available for little money, and they usually run as plug-and-play solutions on Linux without needing special drivers outside the kernel.

We can only hope that Intel will quickly change its mind and get a grip on the situation. Currently, the use of IPU6 cameras on Linux is an outrageous imposition on the part of the vendor. ■■■

Info

[1] Greg Kroah-Hartman on IPU cameras: *https://lore.kernel.org/lkml/YvUaEDMbZD70x+hD@kroah.com/*

[2] Required kernel patch 1: *https://github.com/intel/ipu6-drivers/blob/2b8a592dc63e117d8f5af5f32a10d7e0651832d2/patch/IOMMU-passthrough-for-intel-ipu.diff*

[3] Required kernel patch 2: *https://github.com/intel/ipu6-drivers/blob/2b8a592dc63e117d8f5af5f32a10d7e0651832d2/patch/int3472-support-independent-clock-and-LED-gpios-5.17%2B.patch*

[4] Required kernel patch 3: *https://lore.kernel.org/all/20221106170345.1579893-30-sashal@kernel.org/*

[5] Firmware files and libraries: *https://github.com/intel/ipu6-camera-bins*

[6] Ubuntu OEM Solutions Group: *https://launchpad.net/~oem-solutions-group/+archive/ubuntu/intel-ipu6*

Cybersecurity Maturity Model Certification
VPN with SoftEther
Git Versioned Backups
ADMIN
Network & Security
ISSUE 72
OpenStack
• Sensible deployments
• Small business alternatives
• What's new in OpenStack?
Windows Subsystems for Linux and Android
Parity Declustering RAID
Decrease resilvering times
Siege
Website stress and benchmarking tool
NetTools
Simplified AD troubleshooting
FREE DVD
Tracking MITM
systemd Security
ADMIN
Network & Security
ISSUE 67
systemd Security
Tricks for locking down your system
FREE DVD
AlmaLinux
PS Crescendo
Automation
ADMIN
Network & Security
Automation
in the Enterprise
RPort
Manage and automate remote devices
PowerShell Crescendo
From Linux shell commands to PS cmdlets
Cloud Automation with open source tools
FREE DVD
OSDBuilder
Terraform
ADMIN
Network & Security
ISSUE 69
Terraform
Multicloud orchestrator milestone
PowerShell OSDBuilder
Maintain system images while enforcing security
KUBERNETES FOR SMEs
FREE DVD
libiotrace
Kubernetes
ADMIN
Network & Security
ISSUE 71
Kubernetes
Scale up and stay safe with a containerized environment
Open Policy Agent
A flexible way to manage user rights
Rancher
A Kubernetes management platform
Azure AD Guests
Manage guest accounts with Access Reviews
microVMs
Lightweight VMs with a reduced resource footprint
VXLAN
Overlay networks in virtualized data centers
libiotrace
A ptrace-based tracing mechanism for syscalls
DNS over HTTPS
Advantages and disadvantages
QUIC
UDP-based protocol with mandatory TLS encryption
WWW.ADMIN-MAGAZINE.COM
Container Databases
MS365 Desired State Configuration
Binary Cyberattack Analysis
ADMIN
Network & Security
ISSUE 73
Databases
In clouds, in containers, and scaled out
Kubernetes StatefulSet
Containerize and scale DBs
Purview
Data loss prevention
MS365 DSC
Desired state configuration
Scale Out Cloud DBs
YugabyteDB and Vitess
Win Server 2022 GPOs
Templates and recommendations
ESXi Ransomware Attacks
LINUX NEW MEDIA

REAL SOLUTIONS *for* REAL NETWORKS

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

Improve your admin skills with practical articles on:

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

GET IT FAST
with a digital subscription!

@adminmagazine
@adminmag
ADMIN magazine
@adminmagazine

6 issues per year!

ORDER NOW

shop.linuxnewmedia.com

A command-line dictionary tool

Wordsmith

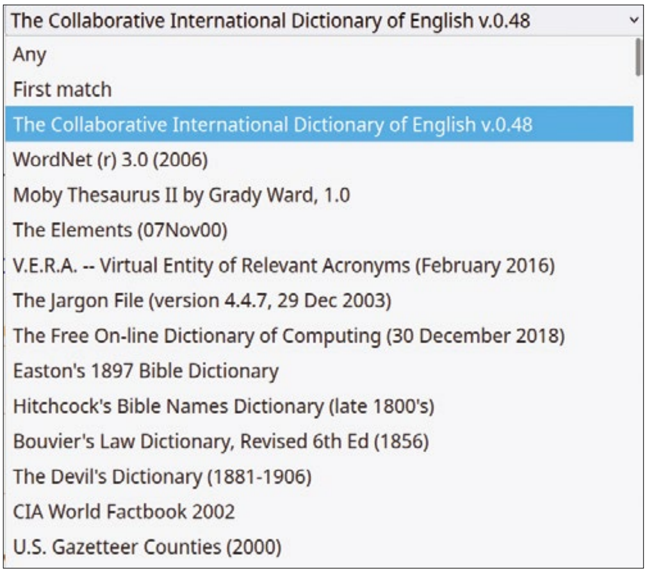
With the dict client, you can quickly search dozens of natural language dictionary databases for the perfect word. *By Bruce Byfield*

As a writer, I rarely need to look up the definition of a word. If I don't know a word, I'm not going to risk using it. However, I frequently check that a word has the exact nuance I want. If it does not, I turn to a thesaurus to search for a closer alternative. Unfortunately, a regular dictionary is next to useless for these purposes. That is why I was pleased to discover `dict` [1], which not only has the information needed for such purposes, but runs from the command line for added efficiency.

A client for the DICT [2] dictionary network protocol, `dict` essentially searches several dozen databases and returns comprehensive results in seconds. It is available for major distributions, but note that each database is often in a separate package in distro repositories and must be installed separately from the basic command. The option `--dbs` (`-D`) will show a list of these databases, as will the web version [3] of `dict` (Figure 1). The databases supported by `dict` include a number of dictionaries, a thesaurus, a collection of acronyms, the Jargon File, the CIA World Factbook, and 34 other languages besides English – far more than the average dictionary offers. The results include not only the range of literal meanings, but numerous contexts as well. Some default results for the word "red" from my current installation are shown in Table 1.

This amount of information makes for intensive scrolling, so `dict` used with only the basic command and a search term should usually be used with a pager like `less` or `cat`. Another option is to specify a specific database so that only the relevant results are given, using the option `--database` (`-d`) `DATABASE`. For instance

```
dict -d moby-thesaurus red
```



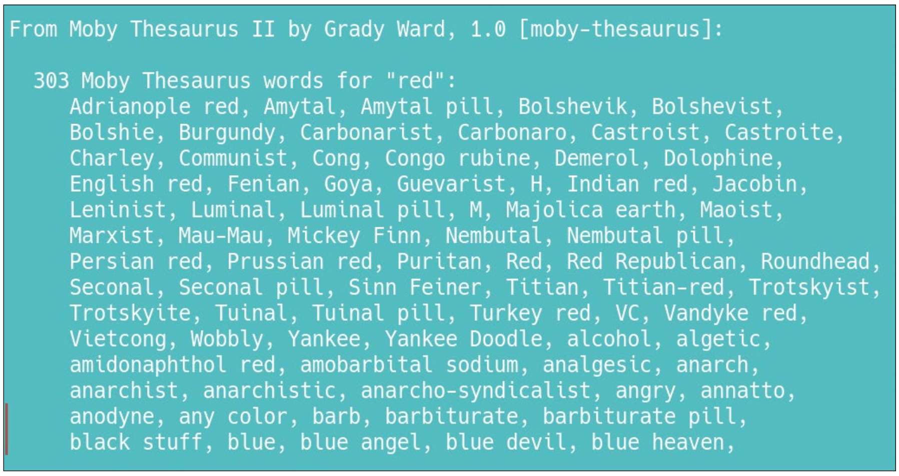
The Collaborative International Dictionary of English v.0.48 ⌄
Any
First match
The Collaborative International Dictionary of English v.0.48
WordNet (r) 3.0 (2006)
Moby Thesaurus II by Grady Ward, 1.0
The Elements (07Nov00)
V.E.R.A. -- Virtual Entity of Relevant Acronyms (February 2016)
The Jargon File (version 4.4.7, 29 Dec 2003)
The Free On-line Dictionary of Computing (30 December 2018)
Easton's 1897 Bible Dictionary
Hitchcock's Bible Names Dictionary (late 1800's)
Bouvier's Law Dictionary, Revised 6th Ed (1856)
The Devil's Dictionary (1881-1906)
CIA World Factbook 2002
U.S. Gazetteer Counties (2000)

Figure 1: A list of `dict`'s installed databases.

Table 1: Sample Results

Basic definitions:	Red \Red\, v. t. To put on order; to make tidy; also, to free from entanglement or embarrassment; – generally with up; as, to red up a house. [Prov. Eng. & Scot.]
Notes on how to use the word:	Red is often used in the formation of self-explaining compounds; as, red-breasted, red-cheeked, red-faced, red-haired, red-headed, red-skinned, red-tailed, red-topped, red-whiskered, red-coasted.
Contexts:	{Red admiral} (Zool.), a beautiful butterfly ({Vanessa Atalanta}) common in both Europe and America. The front wings are crossed by a broad orange red band. The larva feeds on nettles. Called also {Atalanta butterfly}, and {nettle butterfly}.
Note:	Etymologies, synonyms, and translations may also be given, depending on the databases installed.

Photo by Joshua Hoehne on Unsplash



```
From Moby Thesaurus II by Grady Ward, 1.0 [moby-thesaurus]:

  303 Moby Thesaurus words for "red":
     Adrianople red, Amytal, Amytal pill, Bolshevik, Bolshevist,
     Bolshie, Burgundy, Carbonarist, Carbonaro, Castroist, Castroite,
     Charley, Communist, Cong, Congo rubine, Demerol, Dolophine,
     English red, Fenian, Goya, Guevarist, H, Indian red, Jacobin,
     Leninist, Luminal, Luminal pill, M, Majolica earth, Maoist,
     Marxist, Mau-Mau, Mickey Finn, Nembutal, Nembutal pill,
     Persian red, Prussian red, Puritan, Red, Red Republican, Roundhead,
     Seconal, Seconal pill, Sinn Feiner, Titian, Titian-red, Trotskyist,
     Trotskyite, Tuinal, Tuinal pill, Turkey red, VC, Vandyke red,
     Vietcong, Wobbly, Yankee, Yankee Doodle, alcohol, algetic,
     amidonaphthol red, amobarbital sodium, analgesic, anarch,
     anarchist, anarchistic, anarcho-syndicalist, angry, annatto,
     anodyne, any color, barb, barbiturate, barbiturate pill,
     black stuff, blue, blue angel, blue devil, blue heaven,
```

Figure 2: A search in a synonym database for synonyms for "red."

will return only synonyms (Figure 2). Similarly, if you wanted to look up the Old Testament character Jezebel, you could specify `easton` to get results only from *Easton's Bible Dictionary* (1897) (Figure 3) – an encyclopedia entry, rather than a dictionary definition.

Another alternative for limiting results is to choose a strategy from those displayed using the options with `--strats` (`-S`). These strategies are analogous to regular expressions, but actually include both basic and modern options (Figure 4). These strategies are not documented, but most are self-explanatory if sometimes a bit obscure. For example, you can specify the prefix, suffix, and substrings to find a family of related words and use `soundex` to search for similar sounds, which might be useful in a rhymed poem. However, before you choose a limiting strategy, you can use `--match` (`-m`) to see how many results a search will return.

If a search is not useful, you can trying working more closely with a database. With `--serverinfo` (`-I`) and `--serverhelp` (`-H`), you can see more information about a database's server. Alternatively, `--info DATABASE` (`-i`) displays information about the database itself. However, note that no information may be available for some databases, in which case `dict` returns the misleading message, `invalid database`.

Depending on your preferences or how you plan to use search results, you can also use `--format` (`-f`) `FORMAT`. Four different formats are available, as described in the man page: `I`, `S`, `D`, and `m`. Annoyingly, these formats are diagrammed but not documented, so you may have to do more than one test before finding the one you prefer.

Limitations

Linux has other dictionary tools besides `dict`. On the desktop, you can choose such applications as GoldenDict, Artha, the WordNet browser, and Gnome Dictionary. However, so far as I can determine, `dict` is the only one available from the command line and by far the most flexible and thorough. In fact, its potential is so great that its lack of documentation and examples is irksome, if often surmountable. It is not clear whether `dict` is still being actively developed. Still, with patience, what is available is so useful that no writer on Linux should be without it. ■■■

Info

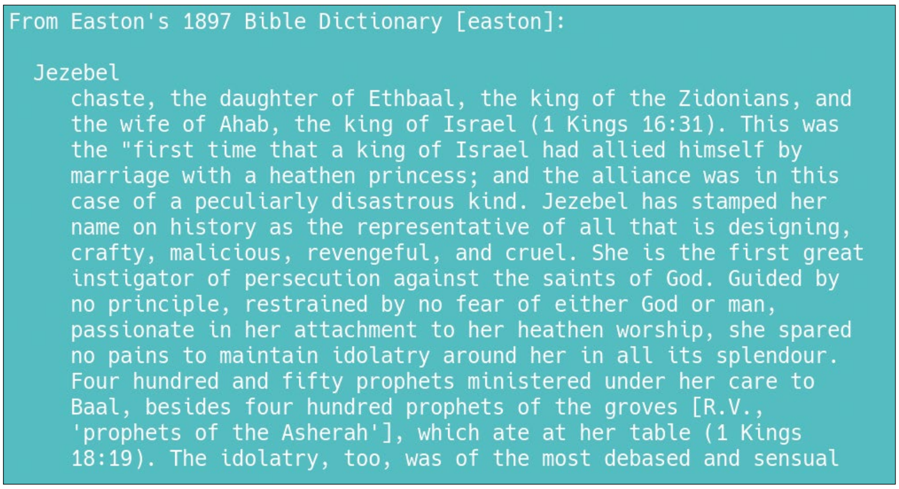
[1] dict: *https://man.archlinux.org/man/dict.1.en*

[2] DICT dictionary network protocol: *https://en.wikipedia.org/wiki/DICT*

[3] dict web version: *http://dict.org/bin/Dict*

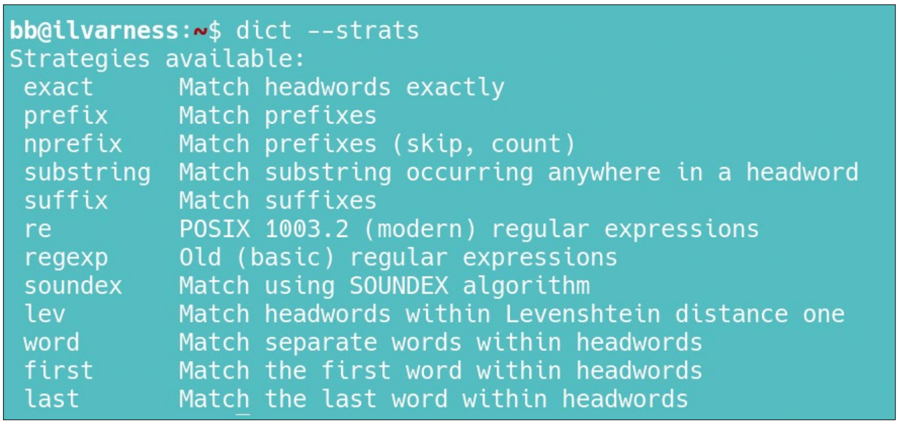
Author

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/*.



```
From Easton's 1897 Bible Dictionary [easton]:

  Jezebel
     chaste, the daughter of Ethbaal, the king of the Zidonians, and
     the wife of Ahab, the king of Israel (1 Kings 16:31). This was
     the "first time that a king of Israel had allied himself by
     marriage with a heathen princess; and the alliance was in this
     case of a peculiarly disastrous kind. Jezebel has stamped her
     name on history as the representative of all that is designing,
     crafty, malicious, revengeful, and cruel. She is the first great
     instigator of persecution against the saints of God. Guided by
     no principle, restrained by no fear of either God or man,
     passionate in her attachment to her heathen worship, she spared
     no pains to maintain idolatry around her in all its splendour.
     Four hundred and fifty prophets ministered under her care to
     Baal, besides four hundred prophets of the groves [R.V.,
     'prophets of the Asherah'], which ate at her table (1 Kings
     18:19). The idolatry, too, was of the most debased and sensual
```

Figure 3: A search in a Biblical database for "Jezebel."



```
bb@ilvarness:~$ dict --strats
Strategies available:
 exact      Match headwords exactly
 prefix     Match prefixes
 nprefix    Match prefixes (skip, count)
 substring  Match substring occurring anywhere in a headword
 suffix     Match suffixes
 re         POSIX 1003.2 (modern) regular expressions
 regexp     Old (basic) regular expressions
 soundex    Match using SOUNDEX algorithm
 lev        Match headwords within Levenshtein distance one
 word       Match separate words within headwords
 first      Match the first word within headwords
 last       Match the last word within headwords
```

Figure 4: Results can be filtered by a choice of strategies.

A Deep Dive into the ELF File Format

Elf Quest

Linux and other Unix-based systems use the ELF file format for executables, object code, and shared libraries. Take a peek inside to learn how an ELF file is organized.

By Alan Guy Pilbeam

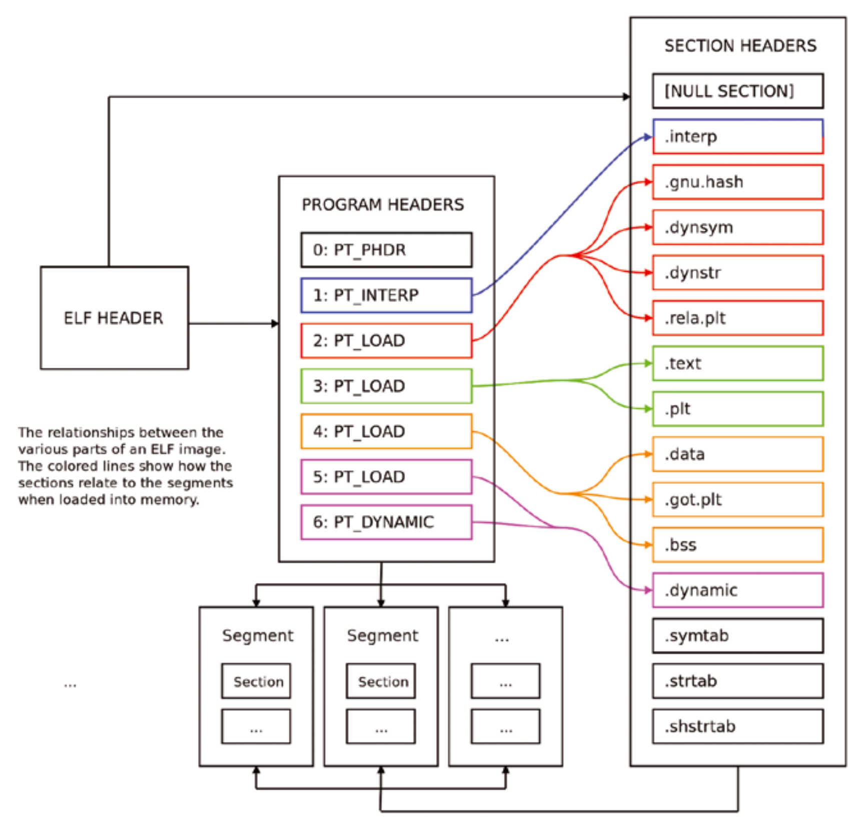
An object file is a black box that you can never really see inside. You can't just open it and read it. We all have some understanding that it contains instructions for the computer, coded in a format that only a computer can understand, but we're used to thinking of compiled program code as something of a mystery. If you take a closer look, however, the structure is not as opaque as you might imagine.

Linux, like most other Unix-like operating systems, stores program code in the Executable and Linkable Format (ELF). ELF superseded the less-flexible `a.out` format and has been the standard in Linux since 1995 and in FreeBSD [1] since 1998. An ELF object file contains the processor-specific instructions that the CPU actually executes. It also contains other information relating to

- how and where to load the object into memory
- dynamic linking
- any public functions exported (or imported) by the object
- debugging

Much of this functionality is only documented in the source code for kernels and toolchains themselves or in various obscure parts of the web. Understanding the ELF format is essential for anyone writing a compiler, assembler, or linker for the Linux platform.

The best way to study the structure of an ELF file is to watch one come together. In this article, I'll show how to construct an extensible ELF file from scratch using flat assembler (fasm) [2], an x86 macro-assembler. Fasm supports the creation of "flat" binary files, meaning that the programmer determines (broadly speaking)



The relationships between the various parts of an ELF image. The colored lines show how the sections relate to the segments when loaded into memory.

Figure 1: **Inside an ELF File.**

Photo by erin mckenna on Unsplash

Listing 1: The ELF Header

```
; Magic.
db 0x7f,"ELF"
; Class (32- or 64-bit).
db ELFCLASS64
; Endian-ness (least significant bytes
; first).
db ELFDATA2LSB
; Version of the ELF spec.
db EV_CURRENT
; ABI (Application Binary Interface) -
; we use ELFOSABI_LINUX = 3 or
; ELFOSABI_FREEBSD = 9
if OS eq "Linux"
  db ELFOSABI_LINUX
else if OS eq "FreeBSD"
  db ELFOSABI_FREEBSD
end if
; ABI version (always 0).
db 0
rept 7 {db 0} ; Padding.
; Executable type (2) could also be
; ET_REL = 1 for a relocatable object or
; ET_DYN = 3 for a shared library.
dw ET_EXEC
; Machine is x86-64.
dw EM_AMD64
; File version (always set to EV_CURRENT
; = 1).
dd EV_CURRENT
; Entry point.
dq LOAD_BASE + PLANE1 + main
; Program headers offset.
dq PROGRAM_HEADERS
; Section headers offset.
dq SECTION_HEADERS
; Flags (always 0).
dd 0
; Size of this ELF header.
dw ELF_HEADER_SIZE
; Size of one program header (56 bytes).
dw SIZEOF_PROGRAM_HEADER
dw NUM_PROGRAM_HEADERS
; Size of one section header (64 bytes).
dw SIZEOF_SECTION_HEADER
dw NUM_SECTION_HEADERS
; This is the section header table index
; of the entry associated with the
; section name string table.
dw SHSTRTAB_INDEX
ELF_HEADER_SIZE = $
```

every byte that appears in the output file. The ELF file will run on both Linux and FreeBSD; the full code is about 500 lines, and you can download it from the associated Git repository [3]. You'll need to install fasm from your operating system's package repository.

In your daily life, it is rarely necessary to build an ELF file manually. This exercise is intended for purposes of illustration. In the real world, a developer tool such as a compiler generates the ELF file based on the contents of source code and information provided by the developer.

The Layout of an ELF File

An ELF file usually has four main parts:

- the ELF header
- one or more program headers
- one or more section headers
- one or more sections

Assembly languages operate at a lower level than a language such as C, so assembly language corresponds more closely with the commands the computer actually executes. I'll therefore use assembly language as a means for describing the ELF format. Figure 1 provides a graphical representation of an ELF file and shows how this file relates to the ELF image when loaded into memory.

The ELF Header

In Listing 1, I declare the ELF header (for those new to assembly language, `db`

Listing 2: The Program Header Macro

```
macro PROGRAM_HEADER type,permissions, offset,virtual_
address,physical_address, disk_size,mem_size,alignment
{
  ; Segment type - common types are
  ; PT_NULL = 0; PT_LOAD = 1;
  ; PT_DYNAMIC = 2; PT_INTERP = 3;
  ; PT_PHDR = 6.
  dd type
  ; Permissions - readable/writable/
  ; executable: an ORed combination
  ; of values from PF_R = 0x4; PF_W =
  ; 0x2; PF_X = 0x1.
  dd permissions
  ; Offset in file.
  dq offset
  ; Offset at runtime.
  dq virtual_address
  ; Unused; set to 0.
  dq physical_address
  dq disk_size
  dq mem_size
  dq alignment
  ; The following variable lets us count
  ; program headers as we declare them.
  CPROGRAM_HEADER = CPROGRAM_HEADER + 1
}
```

means “declare byte,” `dw` means “declare word” (two bytes), and so on; `rept n {}` repeats the enclosed code `n` times).

As you can see in Listing 1, the ELF header contains general information describing the program’s context. For instance, the header contains settings for the class (32-bit or 64-bit), the Endianness, the operating system, the file type (executable, relocatable object, or shared library), and the processor. The ELF header also marks out the space for the program and section headers, which you’ll learn about later in this article.

Table 1: Common Section Names

Name	Description
`interp`	A (possibly) null-terminated string requesting a program interpreter (aka runtime linker/loader). On Linux, this would typically be `/lib64/ld-linux-x86-64.so.2`; on FreeBSD it’s `/libexec/ld-elf.so.1`.
`.gnu.hash`	When the dynamic linker combines all the objects in a process, it needs a way to discover, rapidly, whether a particular symbol is present in a given object file. The GNU hash section provides a precomputed hash table to facilitate this.
`.dynsym`	A list of symbols that the object provides or requires. The first symbol should be a null symbol. For symbols we’re providing ourselves, we need to supply the section index of the section where the symbol’s storage is located, and a virtual address for the symbol.
`.dynstr`	Null-terminated strings, usually the names of libraries and functions needed in the link. This section, like `.strtab` and `.shstrtab`, is defined to begin and end with a null character.
`.rela.plt`	Relocations. Each relocation contains the address of a slot the dynamic linker needs to fill in, as well as the offset of the corresponding symbol in `.dynsym`, the type of relocation (we’ll only be using `R_X86_64_JMP_SLOT = 7`), and a constant addend. These fields are all quad-words.
`.text`	The actual executable code of the program; the address of this section is typically the program’s entry point.
`.plt`	Contains code used as a springboard to functions in other ELF images loaded in the same address space.
`.got.plt`	Contains the absolute addresses of functions in other ELF images loaded in the same address space.
`.bss`, `.data`	These sections contain only program dataholding variables expected by libc.
`.dynamic`	This section contains an array of pairs of quad-words providing extra information to help with dynamic linking. The first quad-word can be thought of as a configuration option and the second, its value. For example, `DT_NEEDED` followed by the offset of the string `libc.so.7` indicates that this library is needed, and `DT_GNU_HASH` followed by a virtual address tells the linker where to find the `.gnu.hash` section.
`.symtab`	Non-dynamic symbols; not usually loaded into memory at runtime.
`.strtab`	Non-dynamic strings referenced by `.symtab`.
`.shstrtab`	Contains the section names used by the section headers.

Listing 3: The Section Header Macro

```
macro SECTION_HEADER name,name_string,
type,flags,virtual_address,offset,size,
link,info,alignment,entry_size
{
  ; An index into .shstrtab giving this
  ; sections's name.
  dd name_string

  ; Type - we will make use of the
  ; following section types:
  ; SHT_NULL = 0 ; initial null section
  ; SHT_PROGBITS = 1 ; .interp, .text,
  ; .plt, .data, .got.plt
  ; SHT_SYMTAB = 2, ; .symtab
  ; SHT_STRTAB = 3, ; .dynstr, .strtab,
  ; .shstrtab
  ; SHT_RELA = 4, ; .rela.plt
  ; SHT_DYNAMIC = 6, ; .dynamic
  ; SHT_NOBITS = 8, ; .bss
  ; SHT_DYNSYM = 11, ; .dynsym
  ; SHT_GNU_HASH = 0x6ffffff6 ; .gnu.hash
  dd type
  ; Flags - a combination of values from
  ; SHF_WRITE = 0x1, SHF_ALLOC = 0x2,
  ; SHF_EXECINSTR = 0x4 and SHF_STRINGS =
  ; 0x20.
  dq flags
  ; Offset at runtime.
  dq virtual_address
  ; Offset in file
  dq offset
  dq size
  ; The index of some other, related,
  ; section.
  dd link
  ; Usually 0 but can indicate
  ; another related section.
  dd info
  dq alignment
  ; Entry size - for sections that
  ; contain a list of similarly-sized
  ; items, this gives the size. Otherwise
  ; 0.
  dq entry_size
  ; Link - so that we can refer to this
  ; section by its name.
  LINK_#name = CSECTION_HEADER
  ; The following variable lets us count
  ; section headers as we declare them.
  CSECTION_HEADER = CSECTION_HEADER + 1
}
```

Program Headers

Each program header describes a memory segment in the final loaded image. Program headers don't have any raw data associated with them; they just tell the loader to allocate memory that will be populated with data from one or more sections. Segments can overlap; you can use the overlap to mark the `.dynamic` section as both `PT_DYNAMIC` and `PT_LOAD`. Because I'm declaring multiple segments, I wrap the declarations in a macro (Listing 2).

I can then declare a program header with a single macro invocation:

```
PROGRAM_HEADER PT_LOAD,PF_R or PF_X,↩
SECTION_TEXT,LOAD_BASE + PLANE1 +↩
SECTION_TEXT,0,TEXT_PLUS_PLT_SIZE,↩
TEXT_PLUS_PLT_SIZE,0x1000
```

A segment's offset and alignment must obey the following constraint:

```
offset mod alignment ==
virtual_address mod alignment &&
offset mod page_size ==
virtual_address mod page_size
```

(where `mod` is the integer modulo operator and `page_size` is usually 4096 for Linux and FreeBSD). The `PLANE*` constants in the source are there to maintain an appropriate minimum distance of `page_size` between segments.

The Section Headers

Each section header describes one section, including its type, permissions, size, and location (on disk and in memory), and other miscellaneous information. The raw data for the sections appears at the end of the ELF file. This raw data consists of executable code, program data such as global objects, and information used in the link. Some ELF images could also contain debugging data in DWARF format.

Table 1 shows some common section names and their descriptions. Most of these need little explanation. Note that the capitalized section names used in the source code don't make it into the final ELF file; they're just labels for the benefit of the assembler. The real section names are by convention lowercase and start with a dot. For more information on the `.gnu.hash` section, see "GNU Hash ELF Sections" online [4]; the accompanying Git repository contains a small C utility that will generate a GNU-style hash for any symbol.

As with the program headers, section headers have a common format, so I wrap the declarations in a macro (Listing 3).

We can then declare a section header with just one macro invocation:

```
SECTION_HEADER TEXT,SHSTRTAB.S6,↩
SHT_PROGBITS,SHF_ALLOC ↩
or SHF_EXECINSTR,
LOAD_BASE + PLANE1 + SECTION_TEXT,↩
SECTION_TEXT,TEXT_SIZE,0,0,0x10,0x0
```

Listing 4: The main Function

```
SECTION_TEXT:

main:

; Align the stack (if needed) to prevent
; segfault.
and rsp, -16
mov rax, 0
; Place string pointer in RDI:
mov rdi, LOAD_BASE + SECTION_DYNSTR + greeting
call puts@plt ; Print our message.
mov rax, 0
mov rdi, 0
call exit@plt ; Exit, returning 0.
```

Listing 5: The PLT and GOT

```
macro PLT_LINKAGE symbol_name,plane
{
  ; Evaluates to puts@plt, exit@plt, etc.
  symbol_name#@plt:
  ; Likewise puts@got, etc.
  jmp qword [plane + symbol_name#@got]
  ; Unresolved symbols cause a JMP to
  ; here.
  symbol_name#@plt2:
  ; Current symbol index.
  push CPLT
  jmp stitchup
  CPLT = CPLT + 1
}

macro GOT_ENTRY symbol_name,plane
{
  ; Evaluates to puts@got, etc.
  symbol_name#@got:
  ; Placeholder until symbol resolution:
  dq LOAD_BASE + plane + symbol_name#@plt2
}

...


SECTION_PLT:
; Instructions executed when the symbol
; is first resolved:
stitchup:
push qword [SECTION_GOT_PLT + PLANE1 + 8]
; Jump into linker.
jmp qword [SECTION_GOT_PLT + PLANE1 + 16]
; We list our imported functions here:
BEGIN_PLT_LINKAGE
PLT_LINKAGE puts,PLANE1
PLT_LINKAGE exit,PLANE1

...


SECTION_GOT_PLT:
; For the linker.
dq LOAD_BASE + PLANE3 + SECTION_DYNAMIC
; The linker fills these slots in:
dq 0
dq 0
; We list our imported functions here:
GOT_ENTRY puts,PLANE1
GOT_ENTRY exit,PLANE1
```

Section Contents

This simple example does not require all the sections described in Table 1, but a brief description of the `.text`, `.plt`, `.got.plt`, and `.rela.plt` sections will give you an indication of how the sections are structured.

The `.text` section contains the executable code for the program proper (Listing 4). In this case, the code calls `puts` to print a string to the terminal and `exit` to return control to the operating system.

Listing 6: Relocations

```
macro RELOCATION symbol_name,plane
{
  ; The address of the qword the linker
  ; must fill in:
  dq LOAD_BASE + plane + symbol_name#@got
  ; Symbol index and type, encoded in a
  ; single qword:
  dq (symbol_name#_index shl 32) or R_X86_64_JMP_SLOT
  ; No addend.
  dq 0
}

...


; some actual relocations
SECTION_RELA_PLT:

RELOCATION puts,PLANE2
RELOCATION exit,PLANE2
```

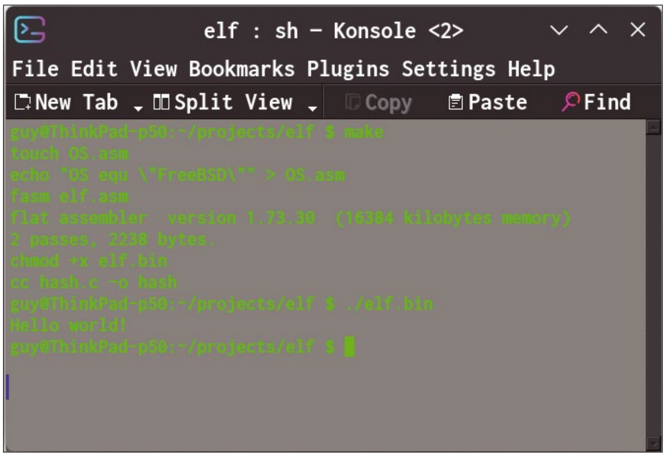
x86-64 instructions often use relative addressing. This means that, for example, a `CALL` instruction is encoded differently depending on its distance from the code it's calling (the destination is encoded as a signed 32-bit value). This makes it impossible to encode an absolute address or call a function whose offset won't fit in 32 bits. The solution is the Procedure Linkage table (PLT) and the Global Offset Table (GOT), which are described in the `.plt` and `got.plt` sections. The PLT provides call destinations that are local to the ELF image, so all of its labels can be reached by a 32-bit relative call. It then uses a `JMP` instruction to jump to the real function, whose address the dynamic linker has placed in the GOT. The PLT and GOT also allow the dynamic linker to resolve function addresses only when required, speeding up the loading process.

Listing 5 defines some convenience macros for the `.plt` and `got.plt` sections and then lists the sections themselves.

Next I use the `.rela.plt` section to define a macro for a single relocation, using the 24-byte structure described earlier (Listing 6).

Exported and Imported Symbols

The executable exports two symbols, `environ` and `__progname`, expected by libc,



```
guy@ThinkPad-p50:~/projects/elf $ make
touch OS.asm
echo "OS equ \"FreeBSD\"" > OS.asm
fasm elf.asm
flat assembler  version 1.73.30  (16384 kilobytes memory)
2 passes, 2238 bytes.
chmod +x elf.bin
cc hash.c -o hash
guy@ThinkPad-p50:~/projects/elf $ ./elf.bin
Hello world!
guy@ThinkPad-p50:~/projects/elf $
```

Figure 2: Building and running an ELF file.

Listing 7: Imported and Exported Symbols

```
macro EXPORT_OBJECT name,index,value
{
  dd name
  ; The symbol's visibility and type
  ; encoded in a byte:
  db (STB_GLOBAL shl 4) + (STT_OBJECT and 0xf)
  ; The "other" field is unused.
  db 0
  ; Section where our symbol resides.
  dw index
  ; Virtual address of symbol.
  dq value
  ; Symbol size.
  dq 8
  name#_index = CSYMBOL
  CSYMBOL = CSYMBOL + 1
}

macro IMPORT_FUNCTION name
{
  dd name
  ; Scope and type.
  db (STB_GLOBAL shl 4) + (STT_FUNC and 0xf)
  ; We're not providing any more
  ; information for this symbol as it is
  ; defined in another ELF object.
  db 0
  dw 0
  dq 0
  dq 0
  name#_index = CSYMBOL
  CSYMBOL = CSYMBOL + 1
}

...


SECTION_DYNSYM:
BEGIN_SYMBOLS
NULL_SYMBOL
IMPORT_FUNCTION puts
IMPORT_FUNCTION exit
EXPORT_OBJECT environ,LINK_BSS,LOAD_BASE + PLANE2 + SLOT_environ
EXPORT_OBJECT __progname,LINK_DATA, LOAD_BASE + PLANE2 + SLOT___progname
```

and imports `puts` and `exit`. In Listing 7, I wrap these declarations in some convenience macros.

Debugging a Handmade ELF Image

Writing ELF files by hand is error prone, and the `readelf` utility is useful for checking the validity of ELF files. A corrupted ELF file will often crash `readelf`, but it usually provides some helpful information before crashing. Table 2 shows some important switches when using `readelf`.

Table 2: readelf Switches

Option	Description
`-h`	Prints the ELF header.
`-l`	Prints the program headers and the relationships between sections and segments.
`-t`	Prints detailed information about the sections.
`-s`	Prints the symbol tables.
`-r`	Prints the relocations.
`-x`	Prints a section's contents in hexadecimal.
`-d`	Prints the contents of the `.dynamic` section.

Conclusion

I can now build and run the ELF file to confirm that it works (Figure 2).

You could easily expand the ELF image I've just described with more handwritten assembly code placed in its `.text` section. All that's needed is to add references to any extra shared libraries and functions to the relevant sections (although certain features, such as thread-local storage (TLS), and accessing the program's arguments may require additional boilerplate code). The main use case, however, might be to combine the ~500 lines of assembler code with the output of a compiler for a higher level language. Fasm gives you ultimate control over the contents of the final binary without having to resort to a blunt instrument such as a hex editor. ■■■

Info

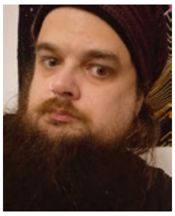
[1] FreeBSD: *https://www.freebsd.org*

[2] flat assembler: *https://www.flatassembler.net*

[3] Git repository for this article: *https://github.com/SanctaMaria1997/elf.git*

[4] GNU Hash ELF Sections: *https://blogs.oracle.com/solaris/post/gnu-hash-elf-sections*

Author



Alan Pilbeam is a programmer and writer from Oxfordshire, England. His passion for computers began when he was gifted a Commodore 64 (together with the legendary Simons' BASIC cartridge) in 1987. He has an endless fascination for compilers (especially BASIC-flavored), assemblers, debuggers, software, and hardware synths, and the history of Unix and Linux. He also likes vegan food and cats.

Linux Magazine Subscription

Print and digital options

12 issues per year

SUBSCRIBE
shop.linuxnewmedia.com/subs

Expand your Linux skills:

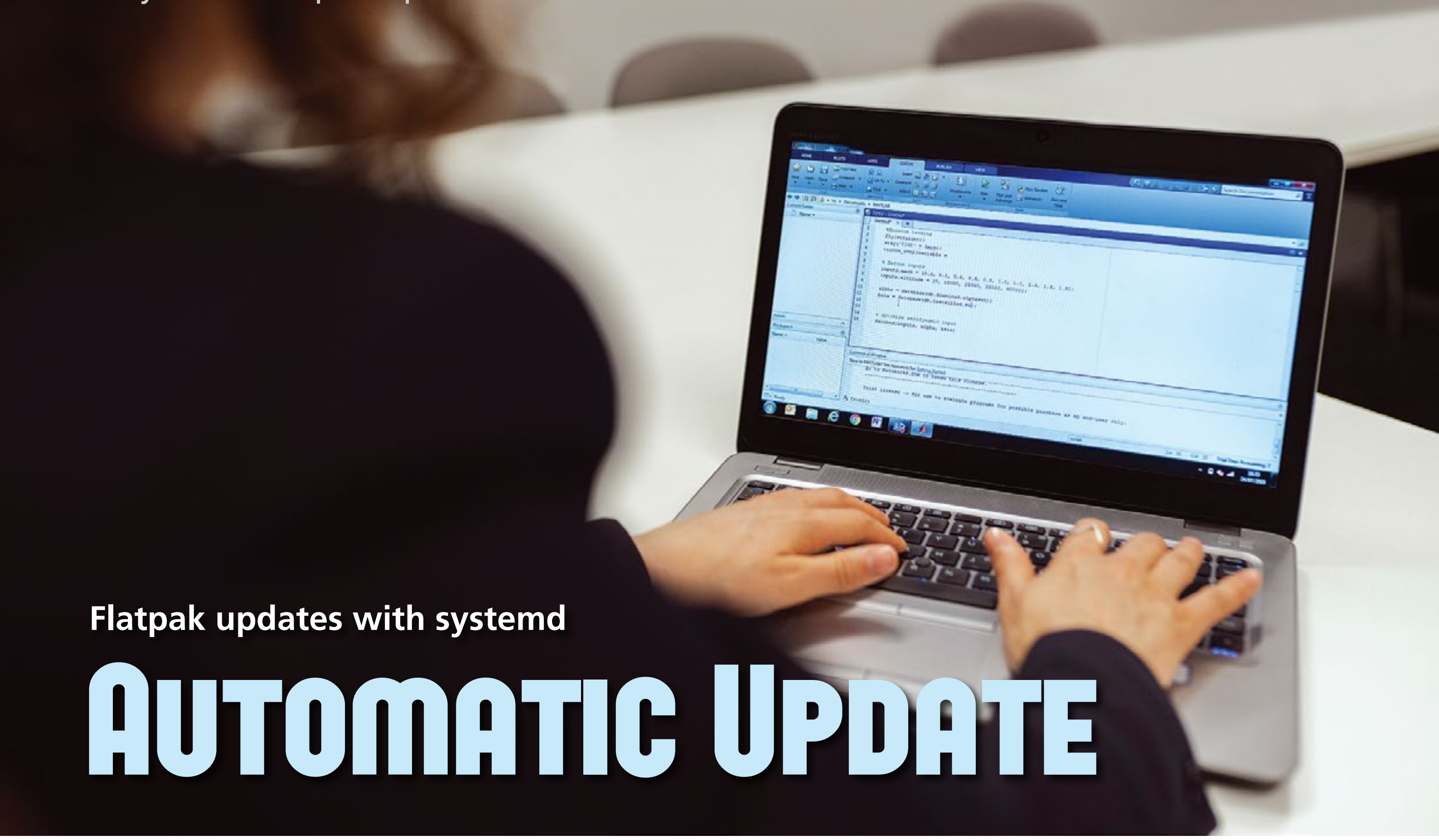
- In-depth articles on trending topics, including Bitcoin, ransomware, cloud computing, and more!
- How-tos and tutorials on useful tools that will save you time and protect your data
- Troubleshooting and optimization tips
- News on crucial developments in the world of open source
- Cool projects for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

Go farther and do more with Linux, subscribe today and never miss another issue!

Need more Linux?
Subscribe free to Linux Update

Our free Linux Update newsletter delivers insightful articles and tech tips to your inbox every week.

bit.ly/Linux-Update



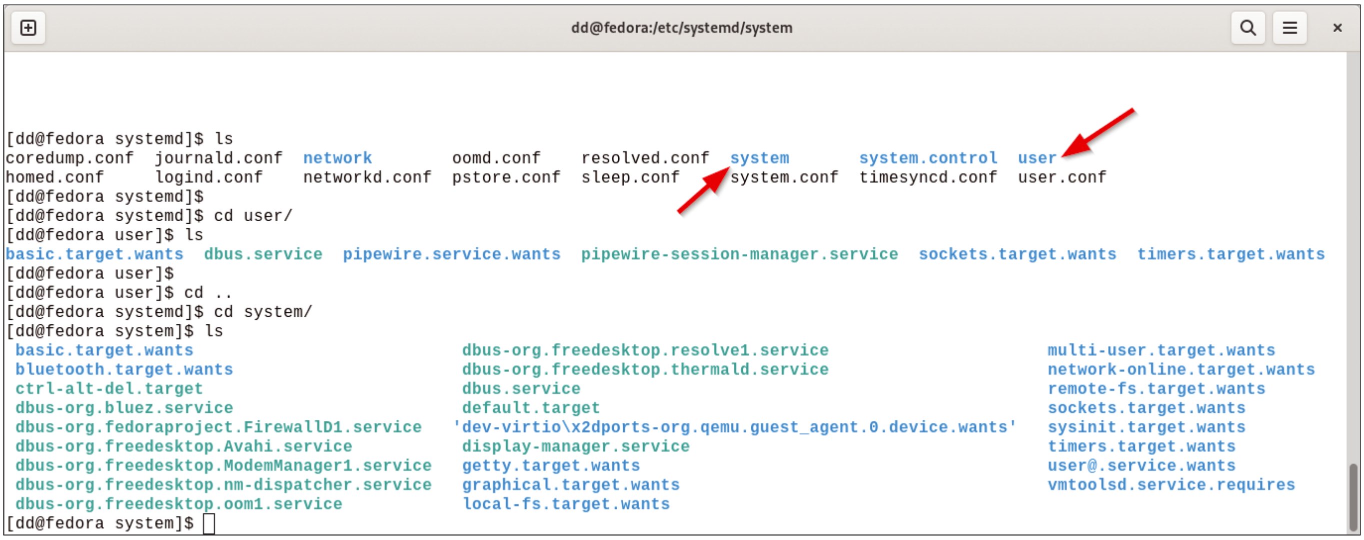
Flatpak updates with systemd

AUTOMATIC UPDATE

You can automate Flatpak updates without a package manager using systemd's services and timers. *By Ferdinand Thommes*

Alternative package systems such as Flatpak, Snap, and AppImage are becoming increasingly popular. They offer a number of advantages, for both developers and users. For example, experimental software versions can be tested without interfering with the distribution's package management system. You can also run several versions of a software tool for comparison purposes without them interfering with each other. Distribution-agnostic Flatpaks can simplify a developer's job because software only needs to be built in a single format.

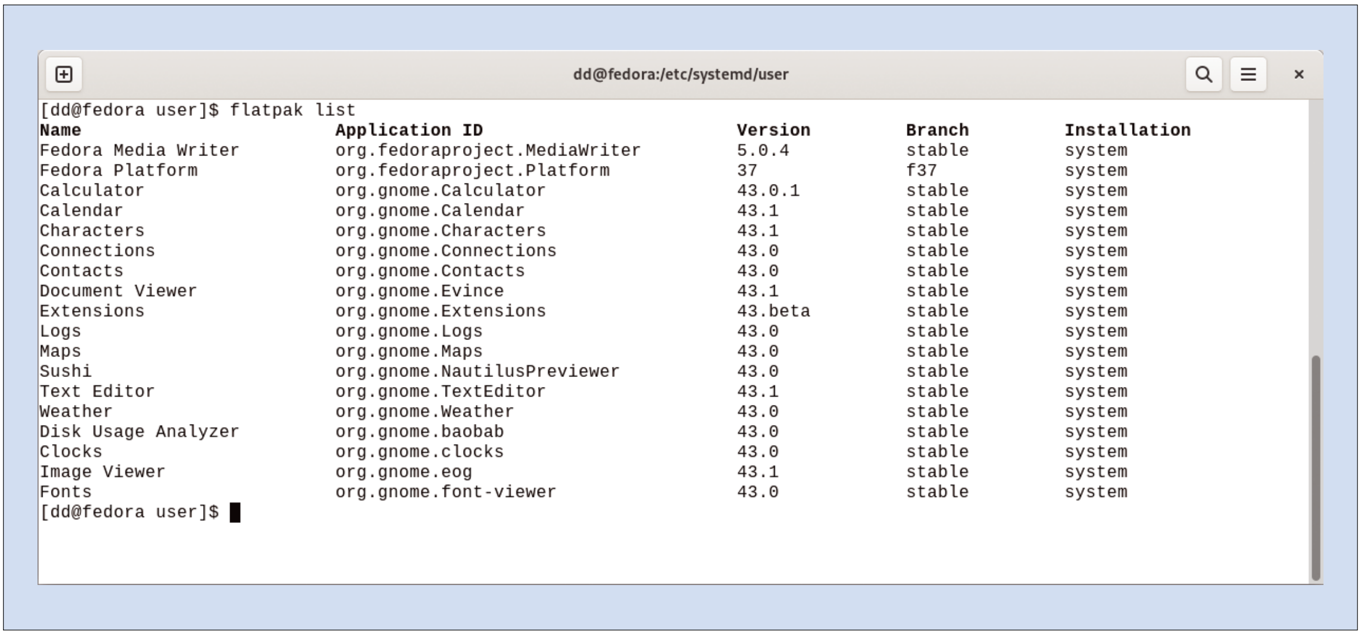
Flatpaks also present some disadvantages. In view of the low price of storage media, it is up to you whether Flatpak's greatly increased space requirement poses a disadvantage. However, the update behavior of Flatpaks and AppImages does pose a clear disadvantage.



```
[dd@fedora systemd]$ ls
coredump.conf  journald.conf  network        oomd.conf    resolved.conf  system       system.control  user
homed.conf     logind.conf    networkd.conf  pstore.conf  sleep.conf     system.conf  timesyncd.conf  user.conf
[dd@fedora systemd]$
[dd@fedora systemd]$ cd user/
[dd@fedora user]$ ls
basic.target.wants  dbus.service  pipewire.service.wants  pipewire-session-manager.service  sockets.target.wants  timers.target.wants
[dd@fedora user]$
[dd@fedora user]$ cd ..
[dd@fedora systemd]$ cd system/
[dd@fedora system]$ ls
 basic.target.wants                           dbus-org.freedesktop.resolve1.service                     multi-user.target.wants
 bluetooth.target.wants                       dbus-org.freedesktop.thermald.service                     network-online.target.wants
 ctrl-alt-del.target                          dbus.service                                              remote-fs.target.wants
 dbus-org.bluez.service                       default.target                                            sockets.target.wants
 dbus-org.fedoraproject.FirewallD1.service   'dev-virtio\x2dports-org.qemu.guest_agent.0.device.wants'   sysinit.target.wants
 dbus-org.freedesktop.Avahi.service           display-manager.service                                   timers.target.wants
 dbus-org.freedesktop.ModemManager1.service   getty.target.wants                                        user@.service.wants
 dbus-org.freedesktop.nm-dispatcher.service   graphical.target.wants                                    vmtoolsd.service.requires
 dbus-org.freedesktop.oom1.service            local-fs.target.wants
[dd@fedora system]$ 
```

Figure 1: Systemd distinguishes between system-wide services and those controlled by users. The same principle applies to Flatpak.

Photo by ThisisEngineering RAEng on Unsplash



```
[dd@fedora user]$ flatpak list
Name                        Application ID                        Version         Branch         Installation
Fedora Media Writer         org.fedoraproject.MediaWriter         5.0.4           stable         system
Fedora Platform             org.fedoraproject.Platform            37              f37            system
Calculator                  org.gnome.Calculator                  43.0.1          stable         system
Calendar                    org.gnome.Calendar                    43.1            stable         system
Characters                  org.gnome.Characters                  43.1            stable         system
Connections                 org.gnome.Connections                 43.0            stable         system
Contacts                    org.gnome.Contacts                    43.0            stable         system
Document Viewer             org.gnome.Evince                      43.1            stable         system
Extensions                  org.gnome.Extensions                  43.beta         stable         system
Logs                        org.gnome.Logs                        43.0            stable         system
Maps                        org.gnome.Maps                        43.0            stable         system
Sushi                       org.gnome.NautilusPreviewer           43.0            stable         system
Text Editor                 org.gnome.TextEditor                  43.1            stable         system
Weather                     org.gnome.Weather                     43.0            stable         system
Disk Usage Analyzer         org.gnome.baobab                      43.0            stable         system
Clocks                      org.gnome.clocks                      43.0            stable         system
Image Viewer                org.gnome.eog                         43.1            stable         system
Fonts                       org.gnome.font-viewer                 43.0            stable         system
[dd@fedora user]$ █
```

Figure 2: The `flatpak list` command shows you (on the far right) the context in which an application was installed as a Flatpak.

Gnome Software, Plasma Discover, and Mint Update will let you automatically update Flatpaks, but this kind of package management is not to everyone's taste. If you rely on the command line to keep your software up to date, Flatpak's update behavior leaves you completely out in the cold. In fact, you will not even be notified of Flatpaks that need to be updated.

Because the Flatpak developers have not planned any further steps for automated updates, the responsibility for updates lies with the desktop environments. For this reason, I put forth a method that lets systemd users automatically update the Flatpaks installed on their systems based on a schedule with the help of the system and session manager. This method not only relieves you of the need to update manually, but it also demonstrates how easy it is to create systemd services and matching timers.

System and User

Flatpaks can be installed in both the user and the system context. Accordingly, you need to create separate services and timers for both contexts.

The `/etc/systemd/` directory contains two subdirectories (among others) named `user/` and `system/`, which contain the service files depending on their purposes (Figure 1).

Only the system administrator can manage the system-wide systemd instance. In addition, there is one instance per user. The instance starts and stops on demand when users log in or log out. A maximum of one user instance per user runs, regardless of how often a user logs in. Only the user (or an administrator) is authorized to

Listing 1: User-Specific Service

```
# update-user-flatpaks.service
# in the folder /etc/systemd/user/
[Unit]
Description=Update user Flatpaks
[Service]
Type=oneshot
ExecStart=/usr/bin/flatpak --user update -y
[Install]
WantedBy=default.target
```

Listing 2: Editing the File

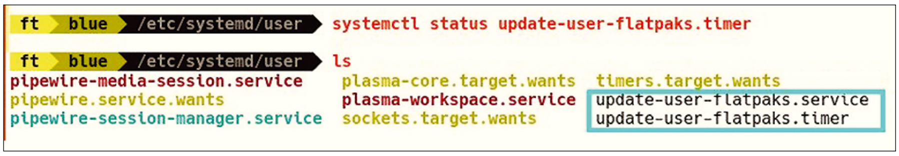
```
$ sudo nano /etc/systemd/user/update-user-flatpaks.service
```

Listing 3: System-Wide Service

```
# update-systems-flatpaks.service
# in the folder /etc/system/system/
[Unit]
Description=Update system Flatpaks
After=network-online.target
Wants=network-online.target
[Service]
Type=oneshot
ExecStart=/usr/bin/flatpak --system update -y
[Install]
WantedBy=multi-user.target
```

Listing 4: User-Specific Timer

```
# update-user-flatpaks.timer
# in /etc/systemd/user/
[Unit]
Description=Update user Flatpaks daily
[Timer]
OnCalendar=daily
Persistent=true
[Install]
WantedBy=timers.target
```



```
ft  blue  /etc/systemd/user  systemctl status update-user-flatpaks.timer

ft  blue  /etc/systemd/user  ls
pipewire-media-session.service   plasma-core.target.wants  timers.target.wants
pipewire.service.wants           plasma-workspace.service  update-user-flatpaks.service
pipewire-session-manager.service sockets.target.wants      update-user-flatpaks.timer
```

Figure 3: After completing the setup, `/etc/systemd/user/` contains a service file and a timer. You should also find the counterparts to these two files in `/etc/system/system/`.

manage these user-specific services. The Arch Linux Wiki [1] explains this concept in more detail.

The context in which the services for updating the installed Flatpaks are created depends on whether the Flatpaks were installed in the user or system context. You can discover the details using the `flatpak list` command (Figure 2), which shows all installed Flatpaks and runtimes. The right-hand Installation column shows the context in which the Flatpaks were installed. If you have both contexts, you need to create services for both the user and the system.

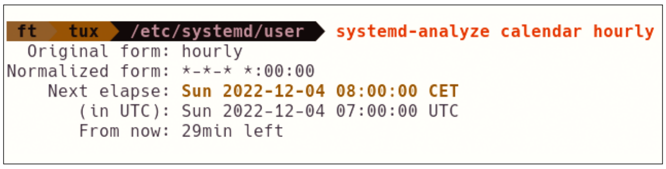
Quickly Installed

Listing 1 shows a systemd service for a user, which you need to store in the systemd user context as `update-user-flatpaks.service`. It can be stored in two locations. If several users use the computer and you want all of them to use the service, `/etc/systemd/user/` is the right choice. For single-user systems, store the service in `~/.config/systemd/user/` instead.

The easiest way to create the service file is to use the command shown in Listing 2. Enter the text from Listing 1 in your choice of editor, and then save. Follow the same steps for the system-wide variant shown in Listing 3. Name this service file `update-system-flatpaks.service` and store it in `/etc/system/system`.

Now you need to create timers to control the services; you will find the timers in Listing 4 (user-specific) and Listing 5 (system-wide). The two timers, named `update-user-flatpaks.timer` and `update-system-flatpaks.timer`, are stored in `/etc/systemd/user/` and `/etc/systemd/system/` respectively.

The `OnCalendar` specification in Listings 4 and 5 lets you adjust for desired update times (Figure 3). If the `daily` update option you entered for `OnCalendar` does not meet your needs, then you could use, for example, `OnCalendar=weekly` for a weekly run.

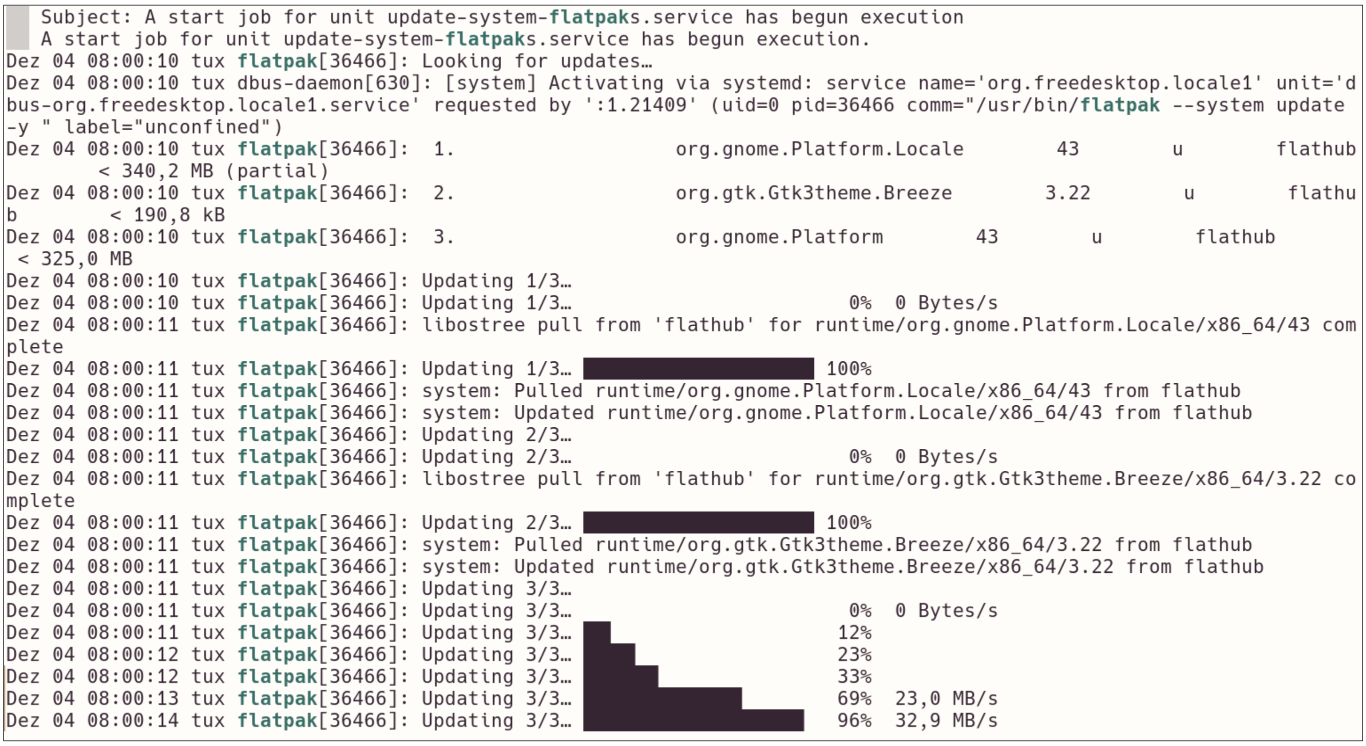


```
ft  tux  /etc/systemd/user  systemd-analyze calendar hourly
  Original form: hourly
Normalized form: *-*-* *:00:00
    Next elapse: Sun 2022-12-04 08:00:00 CET
       (in UTC): Sun 2022-12-04 07:00:00 UTC
       From now: 29min left
```

Figure 4: Want to know when the next update will take place? This command will tell you.

Listing 5: System-Wide Timer

```
# update-system-flatpaks.timer
# in /etc/systemd/system/
[Unit]
Description=Update system Flatpaks daily
[Timer]
OnCalendar=daily
Persistent=true
[Install]
WantedBy=timers.target
```



```
   Subject: A start job for unit update-system-flatpaks.service has begun execution
   A start job for unit update-system-flatpaks.service has begun execution.
Dez 04 08:00:10 tux flatpak[36466]: Looking for updates…
Dez 04 08:00:10 tux dbus-daemon[630]: [system] Activating via systemd: service name='org.freedesktop.locale1' unit='d
bus-org.freedesktop.locale1.service' requested by ':1.21409' (uid=0 pid=36466 comm="/usr/bin/flatpak --system update
-y " label="unconfined")
Dez 04 08:00:10 tux flatpak[36466]:  1.                   org.gnome.Platform.Locale        43        u        flathub
        < 340,2 MB (partial)
Dez 04 08:00:10 tux flatpak[36466]:  2.                   org.gtk.Gtk3theme.Breeze        3.22        u        flathu
b        < 190,8 kB
Dez 04 08:00:10 tux flatpak[36466]:  3.                   org.gnome.Platform       43        u        flathub
 < 325,0 MB
Dez 04 08:00:10 tux flatpak[36466]: Updating 1/3…
Dez 04 08:00:10 tux flatpak[36466]: Updating 1/3…                        0%  0 Bytes/s
Dez 04 08:00:11 tux flatpak[36466]: libostree pull from 'flathub' for runtime/org.gnome.Platform.Locale/x86_64/43 com
plete
Dez 04 08:00:11 tux flatpak[36466]: Updating 1/3… ████████████████████ 100%
Dez 04 08:00:11 tux flatpak[36466]: system: Pulled runtime/org.gnome.Platform.Locale/x86_64/43 from flathub
Dez 04 08:00:11 tux flatpak[36466]: system: Updated runtime/org.gnome.Platform.Locale/x86_64/43 from flathub
Dez 04 08:00:11 tux flatpak[36466]: Updating 2/3…
Dez 04 08:00:11 tux flatpak[36466]: Updating 2/3…                        0%  0 Bytes/s
Dez 04 08:00:11 tux flatpak[36466]: libostree pull from 'flathub' for runtime/org.gtk.Gtk3theme.Breeze/x86_64/3.22 co
mplete
Dez 04 08:00:11 tux flatpak[36466]: Updating 2/3… ████████████████████ 100%
Dez 04 08:00:11 tux flatpak[36466]: system: Pulled runtime/org.gtk.Gtk3theme.Breeze/x86_64/3.22 from flathub
Dez 04 08:00:11 tux flatpak[36466]: system: Updated runtime/org.gtk.Gtk3theme.Breeze/x86_64/3.22 from flathub
Dez 04 08:00:11 tux flatpak[36466]: Updating 3/3…
Dez 04 08:00:11 tux flatpak[36466]: Updating 3/3…                        0%  0 Bytes/s
Dez 04 08:00:11 tux flatpak[36466]: Updating 3/3… ██                    12%
Dez 04 08:00:12 tux flatpak[36466]: Updating 3/3… █████                 23%
Dez 04 08:00:12 tux flatpak[36466]: Updating 3/3… ███████               33%
Dez 04 08:00:13 tux flatpak[36466]: Updating 3/3… ██████████████        69%  23,0 MB/s
Dez 04 08:00:14 tux flatpak[36466]: Updating 3/3… ███████████████████   96%  32,9 MB/s
```

Figure 5: Check the log with `journalctl -b | grep flatpak` to discover exactly what happened in terms of updates since the last time the computer rebooted.

Listing 6: Timer and Systemd

```
01 $ sudo systemctl --user enable --now update-user-flatpaks.timer

02 $ sudo systemctl --system enable --now update-system-flatpaks.timer

03 $ sudo systemctl daemon-reload

04 $ sudo systemd-analyze calendar weekly

05 $ sudo journalctl -b | grep flatpak
```

If needed, however, you can do this in a far more targeted way. To start the update at 6:00pm on the first Saturday of each month, you would use the following entry:

```
OnCalendar=Sat *-*-1..7:*18:00:00
```

You can read all about systemd timers in the Arch Linux Wiki [2] if you are interested in a more in-depth approach.

Enabling Timers

Now it's time to enable the timers. To do this for the user, you need the command in line 1 of Listing 6. For the system-wide setup, use the command in line 2 instead. Finally, the call in line 3 tells systemd to reload all configuration files and restart all units. You also run this command after making changes to the files.

The call in line 4 lets you determine when the next update will occur (Figure 4). If you get the correct output, you also know immediately that the service is set up correctly.

If you want to know after the execution date whether the setup was successful, use the command from line 5. You will see the output here, if successful; otherwise you will see `flatpak update` (Figure 5).

Conclusions

Systemd is suitable for many control tasks on Linux systems. You can manage your Flatpak updates with systemd with just half an hour of work. Once the services and timers are set up, you can leave your future Flatpak updates to the computer.

This article is based on a tutorial by Jordan Williams. You can copy the code for the listings from his website [3] to save time typing. ■■■

Info

[1] Systemd user: *https://wiki.archlinux.org/title/systemd/User*

[2] Systemd timers: *https://wiki.archlinux.org/title/systemd/Timers*

[3] Jordan Williams tutorial and code: *https://www.jwillikers.com/automate-flatpak-updates-with-systemd*

Shop the Shop

shop.linuxnewmedia.com

Missed an issue?

You're in luck.

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

shop.linuxnewmedia.com

GET IT NOW!

SAVE TIME ON DELIVERY WITH OUR ALTERNATIVE PDF EDITIONS

Let an AI chatbot do the work

Machine Language

The electronic brain behind ChatGPT from OpenAI is amazingly capable when it comes to chatting with human partners. Mike Schilli picked up an API token and has set about coding some small practical applications. *By Mike Schilli*

Lead Image © Andrea Danti, 123RF.com

Every month just before *Linux Magazine* goes to press, I hold secret rites to conjure up an interesting topic at the last minute. So it is with interest that I have followed the recent meteoric rise of the AI chatbot ChatGPT [1], which – according to the alarmist press – is so smart that it will soon outrank Google as a search engine. Could this system possibly help me find new red-hot article topics that readers will snap up with gusto, greedily imbibing the wisdom I bundle into them?

The GPT in ChatGPT stands for Generative Pre-trained Transformer. The AI system analyzes incoming text, figures out what kind of information is wanted, mines appropriate responses from a massive data model trained with information from the Internet, and repackages it as a text response. Could the electronic brain possibly help me find an interesting topic for this column?

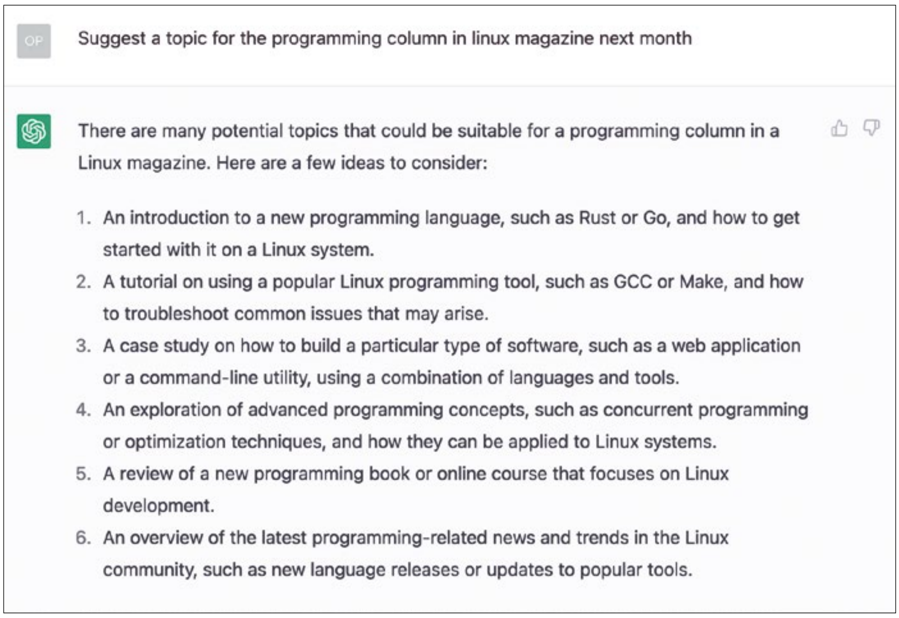
Author

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.

I put it to the test and directed the question to the chatbot. To do this, I typed the question in the search box of ChatGPT's website (Figure 1), and, to my amazement, the AI actually did come up with a couple of usable topics. Introductions to Rust and Go or a deep dive into parallel programming – that sounds interesting!

ELIZA 2.0

The interface feels like a modern version of ELIZA [2], the standard program from the stone age (1960) of this genre. In fact, after ELIZA, almost no progress was made in terms of AI and word processing for half a century, despite long-winded optimistic announcements. In the wake of deep learning and the associated neural networks, however, the field has experienced such a tremendous boost in the past 10 years that it is now difficult to tell whether you are chatting with a

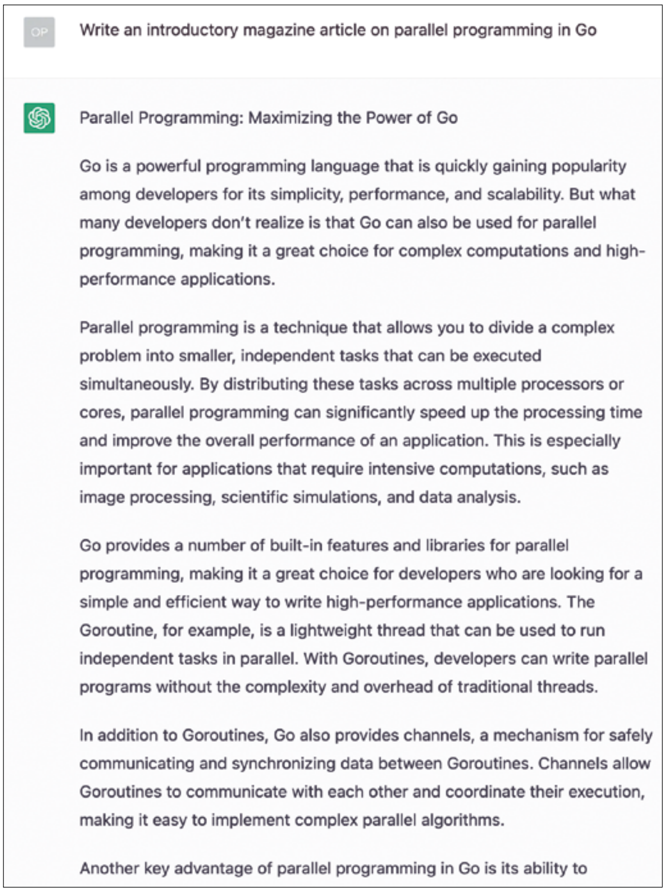


Suggest a topic for the programming column in linux magazine next month

There are many potential topics that could be suitable for a programming column in a Linux magazine. Here are a few ideas to consider:

1. An introduction to a new programming language, such as Rust or Go, and how to get started with it on a Linux system.
2. A tutorial on using a popular Linux programming tool, such as GCC or Make, and how to troubleshoot common issues that may arise.
3. A case study on how to build a particular type of software, such as a web application or a command-line utility, using a combination of languages and tools.
4. An exploration of advanced programming concepts, such as concurrent programming or optimization techniques, and how they can be applied to Linux systems.
5. A review of a new programming book or online course that focuses on Linux development.
6. An overview of the latest programming-related news and trends in the Linux community, such as new language releases or updates to popular tools.

Figure 1: ChatGPT helps Mike find topics for new issues of this column.



Write an introductory magazine article on parallel programming in Go

Parallel Programming: Maximizing the Power of Go

Go is a powerful programming language that is quickly gaining popularity among developers for its simplicity, performance, and scalability. But what many developers don't realize is that Go can also be used for parallel programming, making it a great choice for complex computations and high-performance applications.

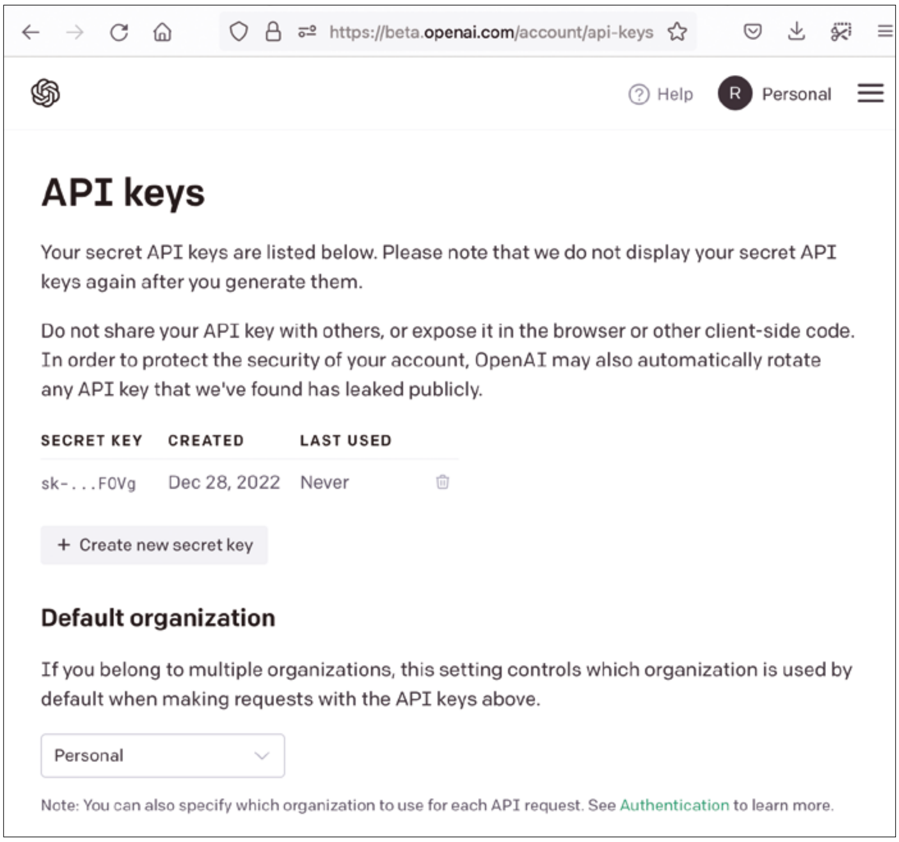
Parallel programming is a technique that allows you to divide a complex problem into smaller, independent tasks that can be executed simultaneously. By distributing these tasks across multiple processors or cores, parallel programming can significantly speed up the processing time and improve the overall performance of an application. This is especially important for applications that require intensive computations, such as image processing, scientific simulations, and data analysis.

Go provides a number of built-in features and libraries for parallel programming, making it a great choice for developers who are looking for a simple and efficient way to write high-performance applications. The Goroutine, for example, is a lightweight thread that can be used to run independent tasks in parallel. With Goroutines, developers can write parallel programs without the complexity and overhead of traditional threads.

In addition to Goroutines, Go also provides channels, a mechanism for safely communicating and synchronizing data between Goroutines. Channels allow Goroutines to communicate with each other and coordinate their execution, making it easy to implement complex parallel algorithms.

Another key advantage of parallel programming in Go is its ability to

Figure 2: If so desired, the AI system will write complete articles on the fly.



API keys

Your secret API keys are listed below. Please note that we do not display your secret API keys again after you generate them.

Do not share your API key with others, or expose it in the browser or other client-side code. In order to protect the security of your account, OpenAI may also automatically rotate any API key that we've found has leaked publicly.

SECRET KEY	CREATED	LAST USED
sk-...FOVg	Dec 28, 2022	Never

Default organization

If you belong to multiple organizations, this setting controls which organization is used by default when making requests with the API keys above.

Personal

Note: You can also specify which organization to use for each API request. See Authentication to learn more.

Figure 3: You can get an API token for free to avoid the hassle of constantly signing into the OpenAI website.

computer or a human being. Today, professional novelists [3] and newspaper writers even freely admit to letting their computer co-drivers take the wheel for wordy passages and plot development [4].

If you type questions for the bot in ChatGPT's browser interface, the system answers after a brief consideration with a jerky text flow, just like a call center person talking to an inquiring customer. The AI not only answers questions about general knowledge, but also writes entire newspaper articles on command. For example, within half a minute of the command "Write an introductory magazine article …," as shown in Figure 2, the text beneath, introducing parallel programming with Go, trickled out of the browser. The results seem amazingly professional. And you can do this in other languages, too. You might suspect that hordes of sports and financial journalists have been using similar systems for years to produce the same article formats on minor league games or stock market fluctuations by the dozen.

Terminal as a Life Coach

Now, if you don't want to keep signing into *OpenAI.com* with your email address, password, and annoying I-am-not-a-robot challenges before you can enter your requests, you can register on the developer page and retrieve an API key [5] (Figure 3). If a program provides this key as an authentication token in API requests, *OpenAI.com* responds via HTTP response, and simple command-line tools can snap up and print the response in the terminal. This gives you a ready-to-rumble command-line interface (CLI) to pretty much all of today's common knowledge.

The auth token for playing around is available after registration with a valid email address and phone number. OpenAI additionally offers a payment model [6], with token-based charges to your credit card. Because limits in free mode are so generous, you won't incur costs for normal use, so you can usually skip entering your credit card data in the first place.

Using the *go-gpt3* package from GitHub, the Go program in Listing 1 communicates with GPT's AI model with just a few lines of code. The program sends questions – known as prompts – to the GPT server as API requests under the hood. If the API server understands the content and finds an answer, it bundles what is known as a completion into an API response and returns it to the client.

Listing 1 wraps the logic for communication in a constructor `NewAI()` and a function `init()` starting in line 15; this fetches the API token from the environment variable `APIKEY` and uses it to create a new API client in line 21. To allow other functions in the package to use the client, line 21 drops it in a structure of the `openAI` type defined in line 8. The constructor gives the calling program a pointer to the structure, and Go's receiver mechanism includes this with the function calls.

Object Orientation in Go

The context object created in line 16 is used to remotely control the client if it times out or otherwise cancels an active request. This is not needed in the simple application in Listing 1, but the `gpt3` library needs it anyway, so line 16 also stores the context in the OpenAI structure so that the `printResp()` function can access it later, starting in line 23.

Prompts and Completions

This is where the actual web access takes place. Line 24 creates a structure of the type `CompletionRequest` and stores the request's text in its `Prompt` attribute. The `MaxTokens` parameter sets how deep the speech processor should dive into the text. The price list on OpenAI.com has a somewhat nebulous definition of what a token is [6]. Allegedly, 1,000 tokens are equivalent to about 750 words in a text; in subscription mode, the customer pays two cents per thousand tokens.

The value for `MaxTokens` refers to both the question and the answer. If you use too many tokens, you will quickly reach the limit in free mode. However, if you set the value for `MaxTokens` too low, only part of the answer will be returned for longer output (for example, automatically written newspaper articles). The value of 1,000 tokens set in line 26 will work well for typical use cases.

The value for `Temperature` in line 27 indicates how hotheaded you want the chatbot's answer to be. A value higher than `0` causes the responses to vary, even at the expense of accuracy – but more on that later. The actual request to the API server is made by the `CompletionStreamWithEngine` function. Its last parameter is a callback function that the client will call whenever response packets arrive from the server. Line 32 simply prints the results on the standard output.

Forever Bot

To implement a chatbot that endlessly fields questions and outputs one answer at a time, Listing 2 wraps a text scanner listening on standard input in an infinite `for` loop and calls the function `printResp()` for each typed question in line 16. The function contacts the OpenAI server and then prints its text response on the standard output. Then the program jumps back to the beginning of the infinite loop and waits for the next question to be entered.

Figure 4 shows the output of the interactive chat program that fields the user's question on the terminal, sends it to the OpenAI server for a response, and prints its response to the standard output. The bot waits for the Enter key to confirm sending a question, prints the incoming answer, and jumps to the next input

Listing 1: openai.go

```
01 package main
02 import (
03   "context"
04   "fmt"
05   gpt3 "github.com/PullRequestInc/go-gpt3"
06   "os"
07 )
08 type openAI struct {
09   Ctx context.Context
10   Cli gpt3.Client
11 }
12 func NewAI() *openAI {
13   return &openAI{}
14 }
15 func (ai *openAI) init() {
16   ai.Ctx = context.Background()
17   apiKey := os.Getenv("APIKEY")
18   if apiKey == "" {
19     panic("Set APIKEY=API-Key")
20   }
21   ai.Cli = gpt3.NewClient(apiKey)
22 }
23 func (ai openAI) printResp(prompt string) {
24   req := gpt3.CompletionRequest{
25     Prompt:      []string{prompt},
26     MaxTokens:   gpt3.IntPtr(1000),
27     Temperature: gpt3.Float32Ptr(0),
28   }
29   err := ai.Cli.CompletionStreamWithEngine(
30     ai.Ctx, gpt3.TextDavinci003Engine, req,
31     func(resp *gpt3.CompletionResponse) {
32       fmt.Print(resp.Choices[0].Text)
33     },
34   )
35   if err != nil {
36     panic(err)
37   }
38   fmt.Println("")
39 }
```

Listing 2: chat.go

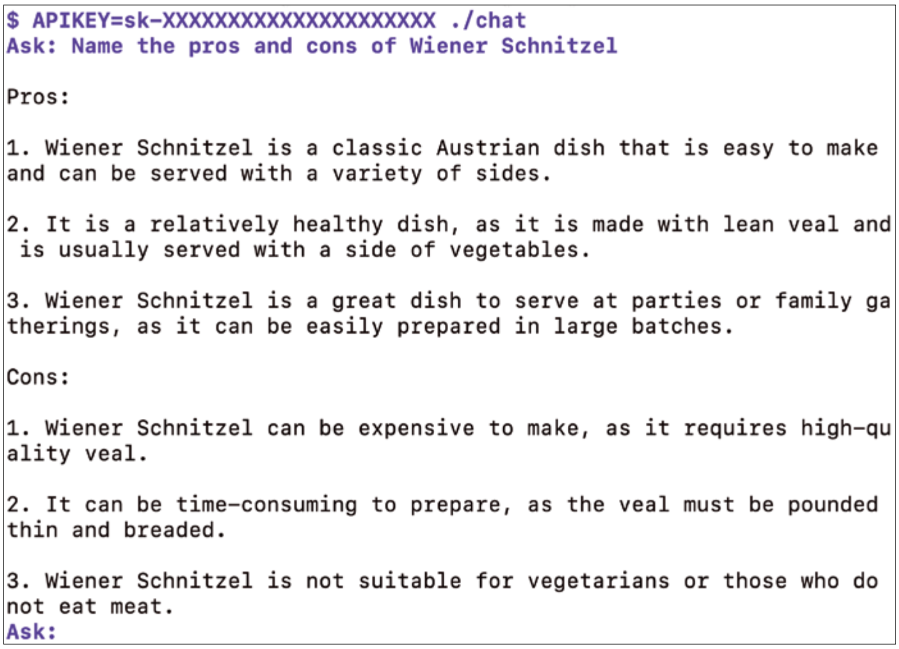
```
01 package main
02 import (
03   "bufio"
04   "fmt"
05   "os"
06 )
07 func main() {
08   ai := NewAI()
09   ai.init()
10   scanner := bufio.NewScanner(os.Stdin)
11   for {
12     fmt.Print("Ask: ")
13     if !scanner.Scan() {
14       break
15     }
16     ai.printResp(scanner.Text())
17   }
18 }
```

prompt. Pressing Ctrl+D or Ctrl+C terminates the program.

It is important to install the API token in the `APIKEY` environment variable in the called program's environment; otherwise, the program will abort with an error message. In Figure 4, the user asks the AI model about the advantages and disadvantages of Wiener schnitzel [7] and receives three pro and con points each as an answer. It turns out that the electronic brain has amazingly precise knowledge of everything that can be found somewhere on the Internet (and preferably on Wikipedia). It is actually capable of analyzing this content semantically, storing it in machine readable form, and answering even the most abstruse questions about Wiener schnitzel in a meaningful way. The `chat` binary with the ready-to-run chatbot is generated from the source code in the listings as usual with the three standard commands shown in Listing 3.

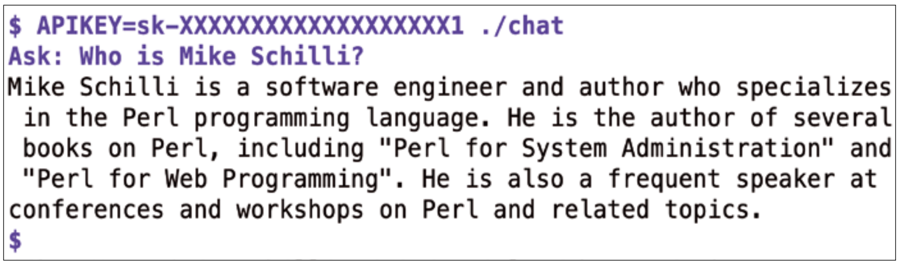
Listing 3: Creating the Binary

```
$ go mod init chat
$ go mod tidy
$ go build chat.go openai.go
```



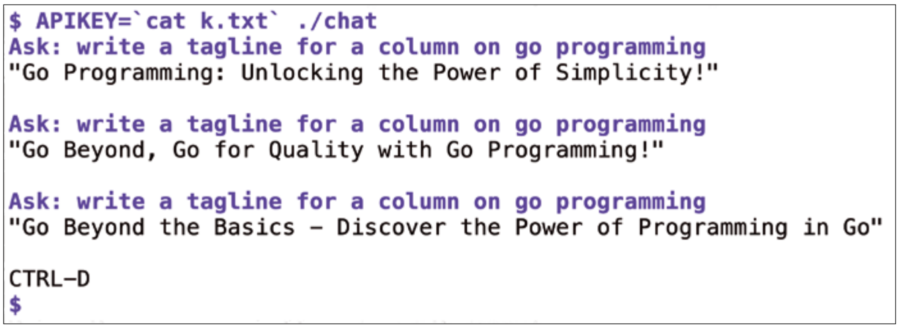
```
$ APIKEY=sk-XXXXXXXXXXXXXXXXXXXXX ./chat
Ask: Name the pros and cons of Wiener Schnitzel

Pros:

1. Wiener Schnitzel is a classic Austrian dish that is easy to make
and can be served with a variety of sides.

2. It is a relatively healthy dish, as it is made with lean veal and
 is usually served with a side of vegetables.

3. Wiener Schnitzel is a great dish to serve at parties or family ga
therings, as it can be easily prepared in large batches.

Cons:

1. Wiener Schnitzel can be expensive to make, as it requires high-qu
ality veal.

2. It can be time-consuming to prepare, as the veal must be pounded
thin and breaded.

3. Wiener Schnitzel is not suitable for vegetarians or those who do
not eat meat.
Ask:
```

Figure 4: The chatbot is shown here as a terminal application with OpenAI.com as the back end.



```
$ APIKEY=sk-XXXXXXXXXXXXXXXXXXX1 ./chat
Ask: Who is Mike Schilli?
Mike Schilli is a software engineer and author who specializes
 in the Perl programming language. He is the author of several
 books on Perl, including "Perl for System Administration" and
 "Perl for Web Programming". He is also a frequent speaker at
conferences and workshops on Perl and related topics.
$
```

Figure 5: The AI even knows the author of this column.



```
$ APIKEY=`cat k.txt` ./chat
Ask: write a tagline for a column on go programming
"Go Programming: Unlocking the Power of Simplicity!"

Ask: write a tagline for a column on go programming
"Go Beyond, Go for Quality with Go Programming!"

Ask: write a tagline for a column on go programming
"Go Beyond the Basics - Discover the Power of Programming in Go"

CTRL-D
$
```

Figure 6: With a `temperature` value of `1`, the AI continually comes up with new suggestions.

By the way, I've found that the AI system's knowledge is not always completely correct. In the answer to the question about who the author of this column is, it lists two Perl books that I have never written (Figure 5). For reference, the correct titles would have been *Perl Power* and *Go To Perl 5*. More importantly, there's no good way to find out what's right and what's wrong, because the bot never provides any references on how it arrived at a particular answer.

Setting the Creative Temperature

Programs can also ask the API to increase the variety of completions provided by the back end. Do you want the answers be very precise or do you prefer to have several varying answers to the same question in a more playful way, even at the risk of them being not 100 percent accurate?

This behavior is controlled by the `Temperature` parameter in line 27 of Listing 1. With a value of `0` (`gpt3.Float32Ptr(0)`), the AI robotically gives the same precise answers every time. But even with as low a value as 1, things start to liven up. The AI constantly rewords things and comes up with interesting new variations. At the maximum value of `2`, however, you get the feeling that the AI is somewhat incapacitated, causing it to output slurred nonsense with partly incorrect grammar. In Figure 6, the robot is asked to invent a new tagline for the Go programming column. With a `temperature` setting of 1, it provides several surprisingly good suggestions.

Translation at Your Command

Because users communicate with the AI back end via questions as text blocks instead of differently named API calls, you can now easily build more specialized tools that follow the pattern of Listing 2. How about a translation chatbot that translates texts from different languages into French? Listing 4 reads texts and sends them to the API with the instruction to "Translate to French" in the prompt.

Listing 4: french.go

```
package main
import (
 "bufio"
 "os"
)
func main() {
 ai := NewAI()
 ai.init()
 scanner := bufio.NewScanner(os.Stdin)
 text := ""
 for scanner.Scan() {
 text += scanner.Text()
 }
 ai.printResp("Translate to French:\n" + text)
}
```

Listing 5: var.sh

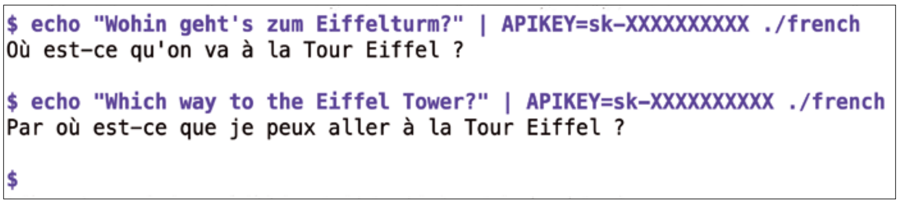
```
curl https://api.openai.com/v1/images/variations \
 -H "Authorization: Bearer sk-XXXXXXXXXXXXX" \
 -F image='@test.png' -F n=3 -F size="1024x1024"
```

Because the server can parse requests in many different languages, the API request does not even need to specify the source language of the provided text to be translated. The AI discovers this automatically on reading the question. Figure 7 shows that the electronic brain translates questions from German and English tourists concerning the route to the Eiffel Tower into very usable French. You can compile the `french` binary by typing

```
go build french.go openai.go
```

This links the new program to the library from Listing 1.

In the examples so far, we've used the el cheapo mode of chatbot with `MaxTokens` set to `1000`, giving us relatively brief answers. For longer texts, however, Listing 1 needs to set `MaxTokens` to a higher value. At some point, though, this will lead to OpenAI wanting to be compensated for the work. You will then need to add a means of payment to the account in the form of a credit card.



```
$ echo "Wohin geht's zum Eiffelturm?" | APIKEY=sk-XXXXXXXXXX ./french
Où est-ce qu'on va à la Tour Eiffel ?

$ echo "Which way to the Eiffel Tower?" | APIKEY=sk-XXXXXXXXXX ./french
Par où est-ce que je peux aller à la Tour Eiffel ?

$
```

Figure 7: The electronic brain also translates from and to a wide variety of languages.

Hard Limits

In summary, for the `Davinci003` model that Listing 1 set as the electronic brain to use in line 30, the AI is surprisingly knowledgeable about real-world facts. However, this knowledge stops abruptly in the year 2021, because more recent events have not yet been fed in. For example, the server has to pass on questions about the Soccer World Cup 2022 in Qatar, and when asked about the Secretary of Defense, the server will respond with the appointee from the Trump administration. The next update to the model will hopefully get it up to speed with current events.

Disturbing Worlds

The AI behind ChatGPT can understand texts, analyze problems, answer questions, and produce output. But it is also capable of image processing. For example, using the API, it can analyze uploaded images, assign them to a category, or subject them to amazing transformations.

For example, Listing 5 runs a `curl` command to contact the OpenAI API endpoint `images/variations`. The back end then goes ahead and disturbingly modifies the

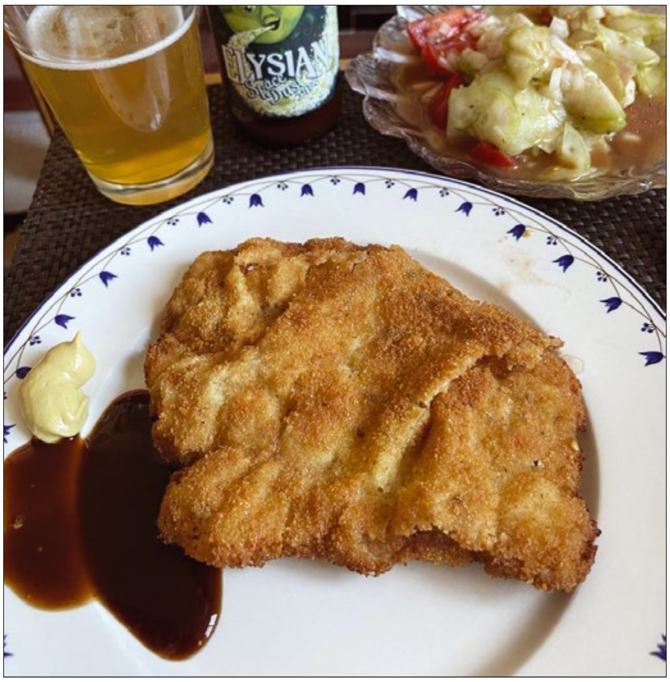


Figure 8: From this schnitzel image, the AI generated …

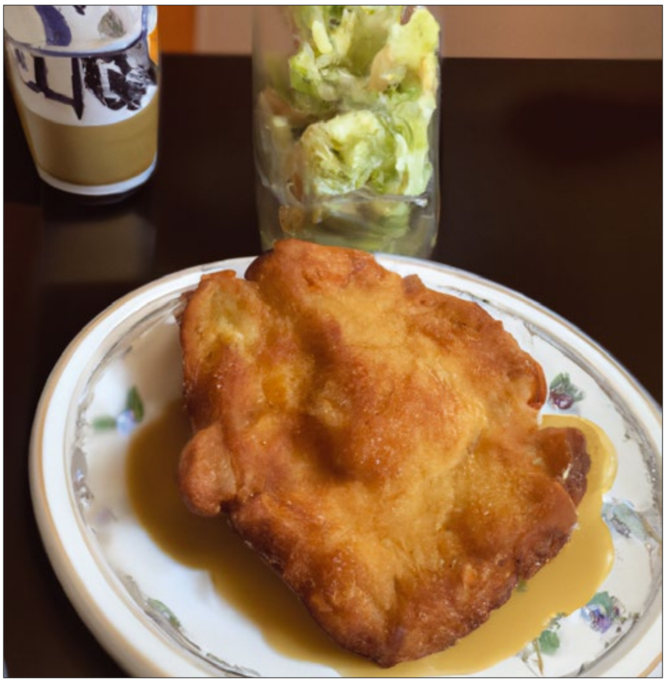


Figure 9: … this rather disturbing variation.

original photo of a Wiener schnitzel that I cooked myself [7]. It is important to note that this particular end point only accepts square photos that are in PNG format and are no larger than 4MB.

To use it, I spun up the trusted `convert` tool from the ImageMagick collection to reformat my schnitzel photo to create a PNG image measuring 1000x1000 pixels, using the `-crop` and `-resize` options. When called with a valid API token, the `curl` command – which parses and uploads the `test.png` file – returns three different URLs in the JSON response. Each of them results in a photo variation generated from the original, which `curl` can subsequently retrieve from the server and store on the local drive.

Figure 8 shows the uploaded original, and Figure 9 shows one of the three variants OpenAI offered for download. You can see that the schnitzel is now on a different plate and surrounded by a light brown sauce. Next to the plate there is a beer glass full of salad. And instead of good old American India Pale Ale, you get what looks like a sake rice wine bottle top left.

On closer inspection, even the schnitzel is no longer the original. Instead, the algorithm used by the AI seems to have combined my photo with one from its archive, probably from a Japanese tonkatsu restaurant. We live in amazing times. ■■■

Info

[1] ChatGPT: *https://openai.com/blog/chatgpt/*

[2] ELIZA: *https://en.wikipedia.org/wiki/ELIZA*

[3] "Collaborative Creative Writing with OpenAI's ChatGPT": *https://andrewmayneblog.wordpress.com/2022/11/30/collaborative-creative-writing-with-openais-chatgpt/*

[4] "How Kindle novelists are using ChatGPT": *https://www.theverge.com/23520625/chatgpt-openai-amazon-kindle-novel*

[5] Retrieving an API key for ChatGPT: *https://beta.openai.com/account/api-keys*

[6] OpenAI price list: *https://openai.com/api/pricing/*

[7] The author making Wiener schnitzel: *https://www.youtube.com/watch?v=88HkqLmx07o*

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

Photo by Andrew Neel on Unsplash

A simpler packet filter

Traffic Rules

Filter rules for firewalls can be tricky. As the successor to iptables, nftables simplifies the process of creating and maintaining firewall rules. *By Frank Hofmann*

Whether you are training to become an IT specialist, managing networks, or preparing for the Linux Professional Institute second certification (LPIC-2) [1], you can't avoid the topic of firewalls, especially rules for filtering packets on the network. After ipchains and Ipfw, the netfilter project's [2] iptables is mostly commonly used for configuring firewall rules on Linux, while FreeBSD/NetBSD and the two Solaris successors Illumos and OpenIndiana use IPFilter [3].

However, iptables is getting a little long in the tooth. In particular, the program code has become increasingly complex. Small changes in the project core have tended to affect all related tools. Iptables, ip6tables, ebtables, and arptables all come from the same codebase, but not in the form of modules. Instead they rely on code duplication, which has resulted in the four tools drifting apart over time. Iptables has been the best maintained, while ebtables has been neglected. Bugs fixed in iptables are still unfixed in ebtables years later.

This situation prompted the netfilter project to launch the development of an iptables successor, nftables [4], as early as 2009. The first two letters in nftables are derived from the project (netfilter). The stated development goals include higher data throughput, greater scalability with a view to changing requirements, and – in particular – a modular structure leading to improved maintainability [5]. Starting with Linux 3.13 (January 2014), nftables is part of the Linux kernel [6], using proven in-house components by the netfilter project.

Since the release of Debian 10 "Buster" in early July 2019, Debian also uses nftables [7], which in turn has affected derivatives such as Ubuntu and Linux Mint, as well as Red Hat Enterprise Linux (RHEL) and CentOS v7 [8] and later. All major distributions have included nftables for quite some time. While they don't necessarily enable nftables by default, it is at least operational.

Removals and Conversions

To create firewall rules, you need the `iptables` (IPv4), `ip6tables` (IPv6), `arptables` (ARP packets), and `ebtables` (Ethernet frames) command-line tools. Nftables replaces all four with a single command-line tool named `nft`, which you can use to set all the rules for accepting, forwarding, modifying, or dropping packets from the network on the system.

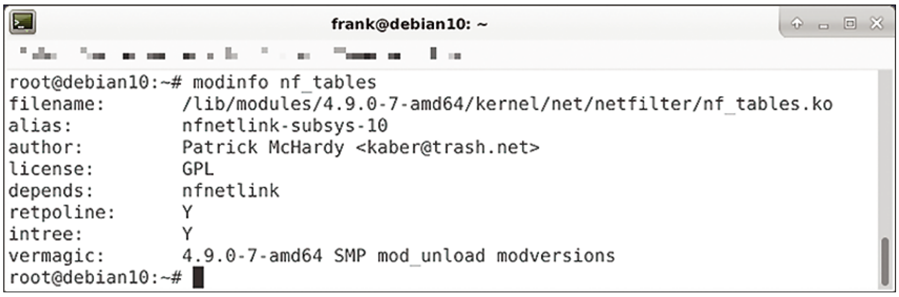
While iptables uses various filters and the three processing chains `INPUT`,

Photo by Ian Taylor on Unsplash

`FORWARD`, and `OUTPUT` to forward the packets, you have to define these yourself for the nftables framework to recognize them.

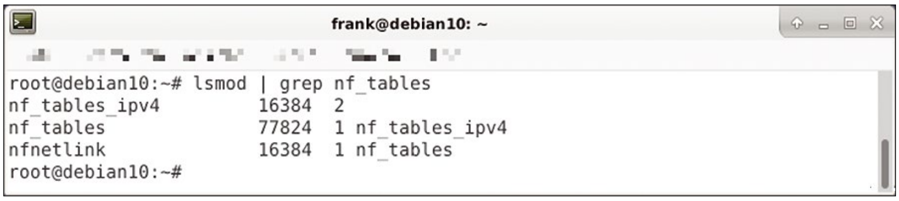
The `nft` tool draws on two libraries: `libnml`, a minimalist Netlink library [9], and `libnftml`, a Netlink userspace library [10]. This results in a reduction of code size in the Linux kernel, and minor changes to `nft` do not prompt the need to rebuild the kernel [11].

To ensure that the correct kernel module has been loaded into the system kernel, check the output from the `modinfo` (Figure 1) and `lsmod` (Figure 2) commands. Once this has been confirmed (as shown in Figures 1 and 2), you can start using `nft` directly.



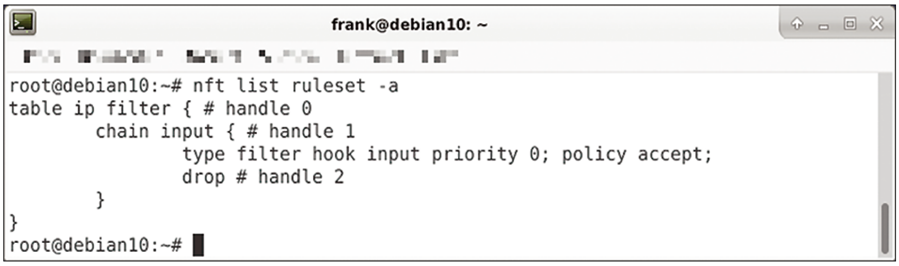
```
root@debian10:~# modinfo nf_tables
filename:       /lib/modules/4.9.0-7-amd64/kernel/net/netfilter/nf_tables.ko
alias:          nfnetlink-subsys-10
author:         Patrick McHardy <kaber@trash.net>
license:        GPL
depends:        nfnetlink
retpoline:      Y
intree:         Y
vermagic:       4.9.0-7-amd64 SMP mod_unload modversions
root@debian10:~# █
```

Figure 1: The `modinfo` command's output provides information about the kernel module.



```
root@debian10:~# lsmod | grep nf_tables
nf_tables_ipv4         16384  2
nf_tables              77824  1 nf_tables_ipv4
nfnetlink              16384  1 nf_tables
root@debian10:~#
```

Figure 2: You can use `lsmod` to find out if the system has the kernel module in place.



```
root@debian10:~# nft list ruleset -a
table ip filter { # handle 0
        chain input { # handle 1
                type filter hook input priority 0; policy accept;
                drop # handle 2
        }
}
root@debian10:~# █
```

Figure 3: The `nft list ruleset -a` command lists all enabled rules.

Basic Configuration

Initially, nftables starts with a completely empty ruleset; there are no predefined tables, chains, or rules. As a user (or admin), you first create the tables, add chains to the tables that hook into the Linux kernel as netfilter hooks, and then fill the chains with the appropriate rules. All of the above steps are done using the `nft` command, which you execute as root.

Listing 1 demonstrates how to define a firewall that prevents (currently) any packets from passing through. Line 1 creates a table for IP packets of the `filter` type. Line 2 adds a chain to the `filter` table. Line 3 adds a rule to the chain that drops all packets (`drop`).

Line 4 provides an overview with all the firewall's rules (Figure 3). Next to the entries, you will see comments formatted as `# handle NUMBER`; you use these comments to reference the entries. This is especially interesting if you want to delete or change existing definitions or insert new definitions in front of or after the entries. For example, line 5 deletes the `drop` rule.

Listing 1: Defining a Firewall

```
01 # nft add table ip filter
02 # nft add chain ip filter input {type filter hook input priority 0\;}
03 # nft add rule ip filter input drop
04 # nft list ruleset -a
05 # nft delete rule ip filter input handle 2
```

Basic Operation

When it comes to writing the rules, the `nft` developers rely on the Berkeley Packet Filter (BPF) [12] and use the classic `tcpdump` [13] as a guide, so you don't have to learn everything from scratch [14].

In addition, `nft` also provides a number of address families. The predefined familes are `arp` (ARP), `bridge` (previously provided by ebtables), `inet` (covers IPv4 and IPv6), `ip` (for IPv4), `ip6` (for IPv6), and `netdev` (which is used to filter incoming packets before they reach Layer 3 according to the ISO/OSI specification [15]).

The `nft` tool acts as a translator of the rules and keeps them in a small virtual machine (nftables core) for communication with the Linux kernel.

Listing 2: Enabling Port 22 for Incoming Packets

```
### Allow incoming packets on port 22.
### With Iptables:
# iptables -A INPUT -p tcp --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
# iptables -A OUTPUT -p tcp --sport 22 -m conntrack --ctstate ESTABLISHED -j ACCEPT
### With Nft:
$ nft add rule inet filter input tcp dport 22 ct state new,established accept
```

Listing 3: Adding Two Additional Ports

```
# nft add rule inet filter input tcp dport { 22, 80, 443 } ct state new,established accept
```

Where appropriate, I will compare the spelling and calls in iptables and `nft` based on practical examples. Listing 2 shows how to enable port 22 for incoming packets, just as you would for access via SSH, for both iptables and `nft`. You will notice that `nft` reduces this to a single command with simpler syntax.

If you want to add ports 80 and 443 (i.e., HTTP and HTTPS), you need two more lines for iptables per port. With `nft`, on the other hand, it is sufficient to extend the existing line to combine all three protocols in one go. All three ports are enclosed in curly brackets, starting with port 22 followed by ports 80 and 443 separated by commas (Listing 3).

Please note that the spaces inside the brackets in Listing 3 must be exactly as shown – otherwise Bash will choke and protest. Users of Zsh run into the same problem, which can be solved by quoting appropriately.

Save and Restore

Similar to iptables, the nftables configuration can be saved to a file. Line 1 in Listing 4 writes the current ruleset to the `firewall.config` file, and line 2 reads the configuration back in.

To make sure that there are no other (possibly interfering) rules left in the cache before initializing the firewall, you should add the line `flush ruleset` at the beginning of the configuration file `firewall.config`.

Listing 4: Saving nftables Configuration

```
01 # nft list ruleset > firewall.config
02 # nft -f firewall.config
```

Creatures of habit, humans have a hard time with change. To help out with the transition from iptables to `nft`, the `iptables-translate` and `ip6tables-translate` commands convert the spelling of iptables firewall rules to those of nftables (Listing 5). This works for both individual instructions and complete rulesets.

Conclusions

Nftables helps to group several complex tools under a common umbrella, making it easier to secure the network. To thoroughly test the new firewall ruleset, you can, for example, use a bunch of Raspberry Pis on a small, dedicated network. Alternatively, you can create a virtual test network using VirtualBox or the smart Mininet [16] application. ■■■

Author

Frank Hofmann works on the road, preferably in Berlin, Geneva, and Cape Town, as a developer, LPI-certified trainer, and author. He is the coauthor of the Debian package management book (*http://www.dpmb.org/index.en.html*).

Acknowledgements

The author would like to thank Axel Beckert and Werner Heuser for feedback during the preparation of this article.

Info

[1] LPIC-2: *https://www.lpi.org/our-certifications/lpic-2-overview*

[2] netfilter project: *https://netfilter.org*

[3] IPFilter: *https://en.wikipedia.org/wiki/IPFilter*

[4] nftables: *https://netfilter.org/projects/nftables*

[5] "What comes after iptables?": *https://developers.redhat.com/blog/2016/10/28/what-comes-after-iptables-its-successor-of-course-nftables*

[6] "Why you will love nftables": *https://home.regit.org/2014/01/why-you-will-love-nftables*

[7] Debian relies on nftables: *https://wiki.debian.org/nftables*

[8] "Hello nftables, Goodbye iptables": *https://linuxnewbieguide.org/hello-nftables-goodbye-iptables*

[9] libmnl: *https://git.netfilter.org/libmnl*

[10] libnftnl: *https://git.netfilter.org/libnfnetlink*

[11] "IPtables vs. nftables": *https://digitalglarus.ch/en-us/cms/blog/2018/08/19/iptables-vs-nftables*

[12] BPF: *https://en.wikipedia.org/wiki/Berkeley_Packet_Filter*

[13] tcpdump: *https://www.tcpdump.org*

[14] "Differences between iptables and nftables explained": *https://linux-audit.com/differences-between-iptables-and-nftables-explained/*

[15] ISO/OSI specification: *https://en.wikipedia.org/wiki/OSI_model*

[16] Mininet: *http://mininet.org*

Listing 5: Converting Rules

```
$ iptables-translate -A INPUT -p tcp --dport 22 -m conntrack --ctstate NEW -j ACCEPT
nft add rule ip filter INPUT tcp dport 22 ct state new counter accept
$ ip6tables-translate -A FORWARD -i eth0 -o eth3 -p udp -m multiport --dports 111,222 -j ACCEPT
nft add rule ip6 filter FORWARD iifname eth0 oifname eth3 meta l4proto udp udp dport { 111,222} counter accept
```

MakerSpace

Viewing wildlife with a Pi Zero photo trap

Garden Safari

Armed with no more than a Raspberry Pi photo trap, you can discover who pays a visit to your garden at night. *By Swen Hopfe*

You might have a sneaking suspicion that wild animals are partying in your garden at night. If you want to know who's come to visit, I can show you how to set up a wildlife monitoring system that is based on a Raspberry Pi and can be completely tailored to your individual needs.

Photo traps are nothing new; you will find professional systems and small boxes for hobbyists online. I wanted one that was just as compact (Figure 1), but with more innovative inner workings. The Raspberry Pi control center shoots the photos with a connected camera. However, it only starts its work when a passive infrared (PIR) sensor, which registers the thermal radiation of living beings, detects motion. Afterward, the electronics revert to power-saving mode, so the photo trap can remain in the field for weeks.

A Raspberry Pi Zero W delivers enough power and takes up so little space that everything fits into a compact case. To get the Zero W up and running, you need to download a new Pi OS image and transfer it to a microSD card. With a monitor and keyboard attached, you can get started by configuring the boot options and network settings.

You want to select automatic login over the command-line interface (CLI) with SSH enabled and then configure all the remaining settings in a terminal window from a computer on your network. The Pi Zero will have the hostname `phototrap`, which sums up its task. The next step is to create a separate folder for the project files in your home directory.

Functionality

When the camera is triggered by the PIR sensor, the image is written to the Pi Zero's SD card along with a timestamp in the file name. To check whether you have any new images, you can read the memory card on the spot or log the device into your home network and copy what you find there over a WiFi connection.

For the camera module, I decided on a model with a filter mechanism and photoresistor that responds to ambient brightness so I could take photos at night without IR blocking, as with a conventional Raspberry Pi NoIR camera, but also take photos in daylight that don't look overexposed; however, the colors are a little off. Ultimately, that does not make much difference, nor does the slightly lower resolution compared with the current Pi camera or the high-quality variant. As with other wildlife cameras, it's all about detection and documentation. The larger lens also does a good job of fitting into the body and adjusting the fixed focal length.

To remain functional for as long as possible in the wild despite a powerful battery and an optional solar panel, it's

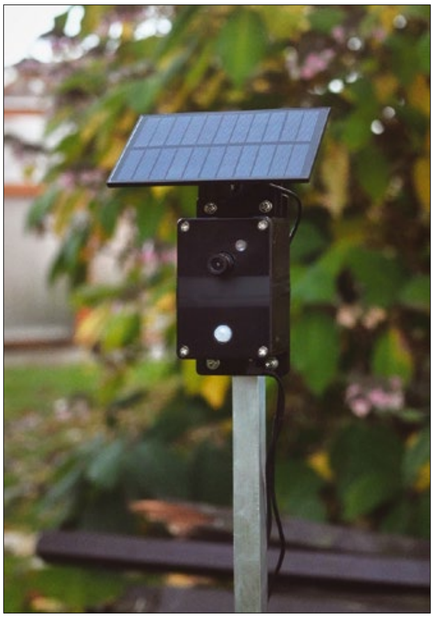


Figure 1: The photo trap automatically switches back to power-saving mode after a shot.

Lead Image © Diana Rich, Fotolia.com

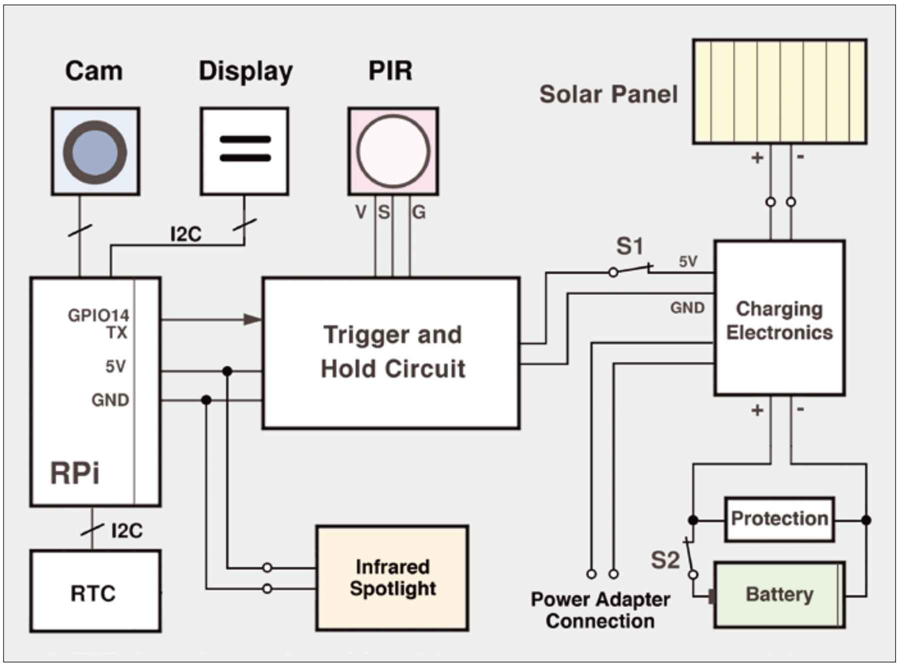


Figure 2: The block diagram shows how the Pi Zero can be switched on and off.

not a good idea to let the Raspberry Pi run constantly. Instead, you will want to wake it up with a small electronic system triggered by the PIR sensor and then switch it off again when the job is done (Figure 2).

The required software is included in the Python `fotofalle.py` script, which starts after booting the Zero, triggers the camera, and then shuts it down. The control electronics detect the sleep state by querying the Pi Zero serial port and shutting off the power supply completely. After that, only the frugal monitoring electronics of the photo trap remain active to react to the next motion detection event, which repeats the process. The script, a wiring diagram, and other useful information for building the device are available from my GitHub project [1], and English versions of the schematics can be downloaded from the *Linux Magazine* code site [2].

Structure

All of the hardware components integrate smoothly into a ready-made case with a rubber seal. The lid has a hole for the camera lens, the photoresistor, and the dome of the motion detector. A real-time clock is at the back of the lid, with an opening for the OLED display on the side. On the inside, I located the battery on the left, with the Pi Zero, charger board, and other electronics sandwiched to the right (Figure 3). The "Parts List" box summarizes all components.

On the protected underside of the housing, the photo trap has connections for a 5V power supply for external charging, an infrared light source, and the plug connection to the solar panel, which has an insert at the very top of the case and acts as a small canopy for rain protection.

One switch is integrated with the power supply. For longer periods

Parts List

- Raspberry Pi Zero W (1 or 2)
- PIR sensor
- RTC module
- OLED display
- Relay module
- Battery
- Overvoltage protection
- Charging electronics
- Circuit board, various resistors
- Capacitor, diodes, transistor
- Housing, wiring
- Solar panel (optional)
- IR spotlight (optional)

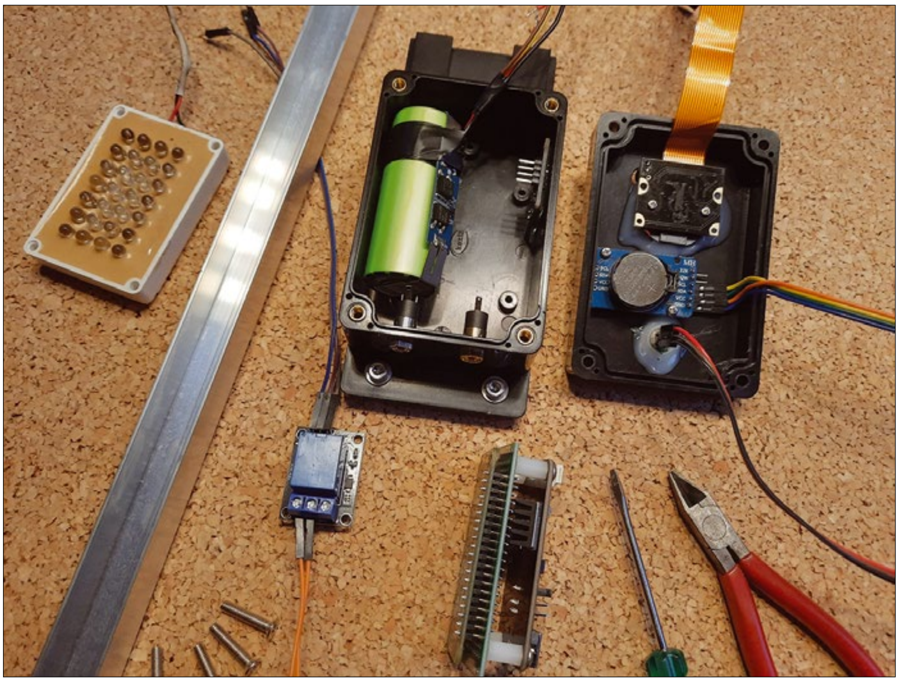


Figure 3: All components to be installed fit easily into a small housing.

without use, another switch disconnects the battery from the protective circuit and the charging electronics. A U bracket on the back of the housing accommodates a ground spike or strap to allow for other installation possibilities of the camera in the wild.

Real-Time Clock

The combination of a timestamp from the real-time clock (RTC) and a random string is used to tag the captured images. However, the Pi Zero does not have a built-in RTC and usually has to retrieve the correct time from a Network Time Protocol (NTP) server. To generate a timestamp for photos shot in the field, even without an Internet connection, you need to add additional hardware to the photo trap.

I used an RTC module with a DS3231 chip that can be powered from the 3V3 pin of the Zero. Older boards are designed for operation with a coin battery. If you want to use a conventional CR2032 cell, you have to remove the resistor at top right to avoid long-term damage. When running on a Raspberry Pi, you can optionally remove the pull-up resistors opposite the VCC pin.

Check the I2C bus with

```
sudo i2cdetect -y 1
```

and you will see two entries for the module. By default, the output contains the value *57* at position 50:7 and *68* at position 60:8.

You will need to enable the RTC during the boot process by adding the line

```
dtoverlay=i2c-rtc,ds3231
```

to your `config.txt` file for the overlay. The entry must not contain spaces to the right of the equals sign. After rebooting and calling `i2cdetect` again, a value of *UU* should now be found at position 60:8 (Listing 1), which confirms that the driver for the DS3231 chip is active.

Now that an RTC is running, you need to disable the previous service provided by the Raspberry Pi:

```
$ sudo systemctl ⮐
  disable fake-hwclock.service
```

After the next reboot, the Zero will use the hardware clock. Now you need to set up a udev rule to make sure the RTC is updated automatically over the web when an Internet connection is opened:

```
$ sudo nano /etc/udev/rules.d/⮐
  60-rtc_synch.rules
```

This command creates the `60-rtc_synch.rules` file with the content:

```
ACTION=="add", SUBSYSTEM=="rtc", ⮐
  ATTRS{hctosys}=="0", ⮐
  RUN+="/sbin/hwclock -s --utc"
```

As a result, the RTC should synchronize when the Pi Zero is briefly connected to the Internet and then will always display the current time when offline. You can check whether this is the case in the test setup at any time with:

```
$ sudo hwclock -r; date
```

If required, the built-in clock can be set manually:

```
$ sudo hwclock ⮐
  --set --date="10 ⮐
  Dec 2022 10:00"
```

Now your photo trap has an independent and reliable clock onboard. In

Listing 1: Active Driver

```
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- 57 -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- UU -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

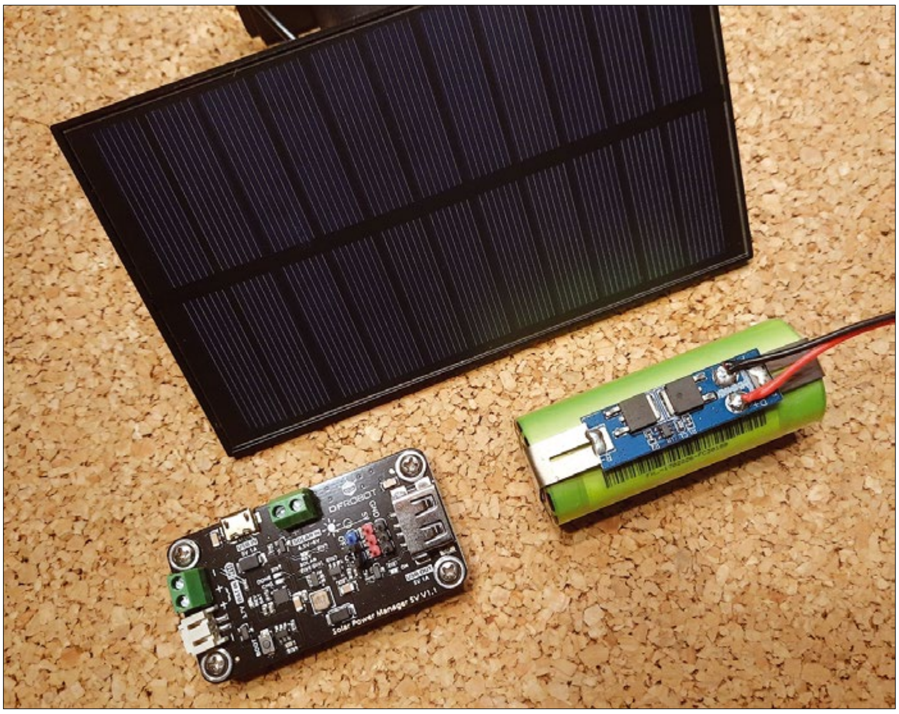


Figure 4: The charging electronics include a battery with a protection circuit.

addition to the timestamp, the file names of the captured images include a random string, ensuring that no images are overwritten, even in the event of an error.

OLED Display

The OLED connects to the Pi Zero over the I2C bus, too. It serves as an informational display to output details of the device's status and the recording process when setting up in the field. The screen is switched on along with the Pi Zero and switched off accordingly when the wildlife camera is idle.

To specifically trigger a photo, swipe your hand over the motion detector to see whether a new photo is taken and what timestamp it has. The recommended approach here is to check the test shot directly afterward to realign the camera, if necessary.

Battery and Charging Electronics

I used a lithium iron phosphate (LiFePO4) battery, which has a very low self-discharge rate and a good power-to-weight ratio. It doesn't suffer from a memory effect and works at low temperatures. All of this is ideal for a photo trap in the wild.

Although you can find chargers for these lithium batteries, you can equip existing charging modules with a protective circuit to limit their end of charging voltage to 3.6V. I used a solar power manager by DFRobot with connections for a solar module with 4V to 6V, an external charging option, and a battery connection. I also added a protection board mounted directly on the battery (Figure 4). The small battery management board offers deep discharge and short circuit protection in addition to overcharge detection.

Conclusions

Because of the shutter lag caused by the power-saving mechanism, this wildlife camera is not suitable for capturing animals moving quickly through the focus area. However, experience has shown that only very few garden visitors are in such a hurry, which is why I think, after a period of use in my own garden, that this DIY model is up to the task. Because I regularly have animal visitors on my property, the photo trap is triggered fairly often; it works satisfactorily across the board and will remain in operation to deliver many interesting shots.

Info

[1] Code on GitHub: *https://github.com/swenae/fotofalle*

[2] English schematics: *https://linuxnewmedia.thegood.cloud/s/5Rzx9tQW2FJ6N3Z*

Author

Swen Hopfe works for a medium-sized company with a focus on smart cards and near-field communication (NFC). When he is not taking photos, in the great outdoors, or in his garden, he focuses on topics such as the Raspberry Pi, Internet of Things, and home automation.

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

MakerSpace

Text-based menus and information pages

Whip It Up

Whiptail interfaces add menus and information pages to your Raspberry Pi projects. *By Pete Metcalfe*

Text interfaces can be extremely useful for Windows or secure shell (SSH) clients that need to connect to your Linux or Raspberry Pi systems. One of the simplest options for creating text interfaces is Whiptail [1], which comes preinstalled on Raspbian and many Linux systems. The Raspberry Pi configuration tool, `raspi-config` (Figure 1), is a good illustration of how Whiptail can be used.

In this article, I introduce Whiptail with three small projects. The first creates a menu interface for some common Linux diagnostic tools, the second controls Raspberry Pi output pins, and the final uses Python to display Pi sensor data.

Getting Started

If your system does not have Whiptail, you can install it on Raspbian/Debian/Ubuntu machines with the command:

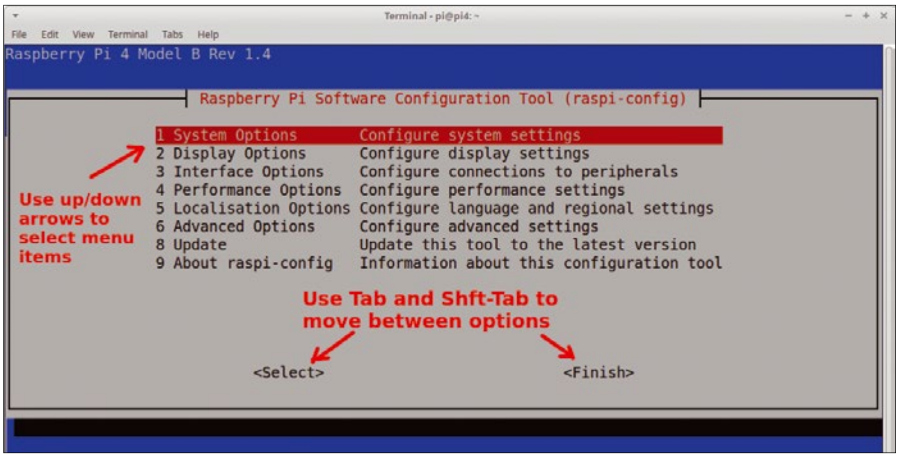
```
sudo apt install whiptail
```

A simple Whiptail test shows the date/time in a message box:

```
whiptail --title "Time/Date" ⮐
          --msgbox "$(date)" 0 0
```

This statement clears the terminal screen and shows a default background with a message box centered in the window (Figure 2). The last two parameters in the statement define the height and width of the dialog. The *0 0* at the end of the last line autosizes the message box.

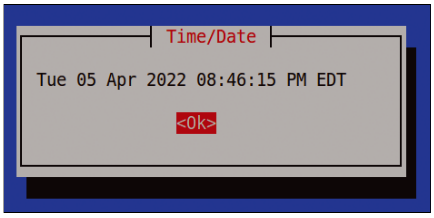
The Whiptail utility supports a variety of controls, such as: `checklist`, `gauge`,



```
Raspberry Pi 4 Model B Rev 1.4

        ┤ Raspberry Pi Software Configuration Tool (raspi-config) ├

                1 System Options       Configure system settings
                2 Display Options      Configure display settings
                3 Interface Options    Configure connections to peripherals
                4 Performance Options  Configure performance settings
                5 Localisation Options Configure language and regional settings
                6 Advanced Options     Configure advanced settings
                8 Update               Update this tool to the latest version
                9 About raspi-config   Information about this configuration tool

                     <Select>                          <Finish>
```

Use up/down arrows to select menu items
Use Tab and Shft-Tab to move between options

Figure 1: The Raspberry Pi configuration tool `raspi-config` uses Whiptail.



```
        ┤ Time/Date ├

 Tue 05 Apr 2022 08:46:15 PM EDT

             <Ok>
```

Figure 2: Whiptail message box.

Lead Image © PazMadrid, morguefile.com

Listing 1: Whiptail Menus

```
01 #!/bin/bash
02 #
03 # wmenu.sh - Wipetail Menu example with diagnostic tools
04 #
05
06 while true
07 do
08 # Create a whiptail menu, pass the tool name if selected
09 selection=$(whiptail --title "Diagnostics" --nocancel
     --menu "Select Tool:" 0 0 4
       "df" " Disk Space"
       "vmstat" " Memory Stats"
       "lsusb" " USB Devices"
       "Exit" " Close and Exit" 3>&1 1>&2 2>&3 )
10
11   # Do an action for the selected menu item
12   case $selection in
13     "df" )
14       whiptail --title "Disk Space" --msgbox "$(df -h)" 0 0
15       ;;
16     "vmstat" )
17       whiptail --title "Memory Stats" --msgbox "$(vmstat --stats)" 0 0
18       ;;
19     "lsusb" )
20       whiptail --title "USB Devices" --msgbox "$(lsusb)" 0 0
21       ;;
22     "Exit" )
23       exit
24       ;;
25   esac
26 done
```

`infobox`, `inputbox`, `menu`, `msgbox`, `passwordbox`, `radiolist`, `textbox`, and `yesno`.

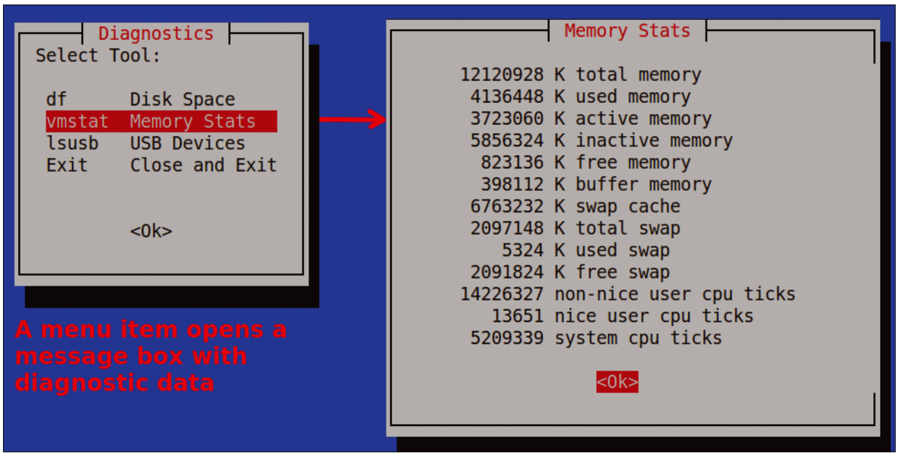
A Menu of Diagnostic Tools

Menuing is one of most useful features in Whiptail. A main script can present a top-level menuing interface that links to submenus, information pages, or applications. The syntax for the menu options is:

```
[...] --menu <text> <height> ↩
            <width> <listheight> ↩
            [<tag> <item>]
```

The `<item>` parameter is the text string shown on the menu line. The `<tag>` parameter is the variable passed when a menu is selected; this parameter is typically a number, but you can also use strings. In my menuing example, I will be passing the diagnostic utility name as the tag.

Listing 1 is a Whiptail menuing script that shows three common Linux diagnostic tools, `df` (disk space usage), `vmstat` (virtual memory statistics), and `lsusb` (list USB devices). For this example, a fourth menu line exits the script (Figure 3). The `--menu` option is defined to autosize with four menu items (`0 0 4`; line 9). Each menu has a line description and a tag that is passed when that line is selected.



```
    12120928 K total memory
     4136448 K used memory
     3723060 K active memory
     5856324 K inactive memory
      823136 K free memory
      398112 K buffer memory
     6763232 K swap cache
     2097148 K total swap
        5324 K used swap
     2091824 K free swap
    14226327 non-nice user cpu ticks
       13651 nice user cpu ticks
     5209339 system cpu ticks
```

A menu item opens a message box with diagnostic data

Figure 3: This Whiptail menu shows diagnostic data.

For menuing applications the Whiptail utility requires a set of redirection statements, (3 > &1 1 > &2 2 > &3) at the end of the main command. These redirections are a Bash trick to swap standard output, (file 1, stdout), and standard error (file 2, stderr). By adding these statements Whiptail has a clean, error free, mechanism for user input and menu output.

This menu example only uses an *Ok* button, the *Cancel* button is disabled by the `--nocancel` parameter (line 9). In the next example the *Cancel* button will exit the script.

Raspberry Pi Radiolist Project

The goal of the Raspberry Pi radiolist project is to create a simple user interface that controls general purpose input/output (GPIO) pins. For my setup I am using a Pimoroni Explorer HAT (hardware attached on top) that has four LEDs; however, other hardware arrangements could also be used.

The first step for this project is to install wiringPi [2], so that GPIO pins can be accessed and controlled from Bash scripts. This utility is installed on a Raspberry Pi with:

```
sudo apt-get install git-core
git clone https://github.com/↩
  WiringPi/WiringPi.git
cd WiringPi
git pull origin
./build
```

The *wiringpi* library includes the `gpio` command-line utility that you can use to read, write, toggle, and set the mode of GPIO pins. My hardware arrangement has four LEDs on pins 0, 2, 7, and 21 that I need to set up for output:

```
# Set the LED pins as outputs
gpio mode 7 output
gpio mode 0 output
gpio mode 2 output
gpio mode 21 output
```

Once the GPIO pins are set up as outputs, I can toggle their states manually:

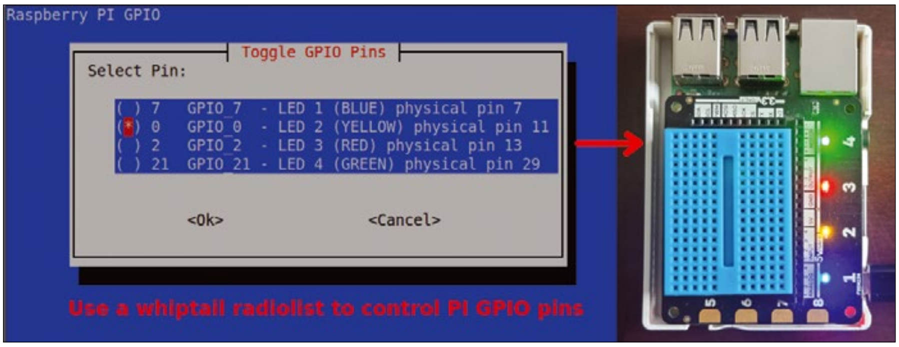
```
# Toggle the output state on pin 7
gpio toggle 7
```

The syntax of a `radiolist` is similar to a menu:

```
--radiolist <text> <height> ↩
  <width> <list-height> ↩
  [ <tag> <item> <status> ]
```

The only difference with a checklist is that you can indicate which entry is currently selected by setting its status to `on`. The default `radiolist` has two buttons, an *Ok*, which returns a `0` when selected, and a *Cancel* button, which returns a 1.

The Raspberry Pi radiolist script (Listing 2) shows a list of GPIO pins the user can toggle (Figure 4). A `while` statement loops as long as the *Ok* button is selected (lines 9 and 14). The radiolist is defined to show a GPIO pin in the `<tag>` parameter, and the `<item>` parameter displays the LED number, color, and physical pin number (lines 11). An `if` statement checks a GPIO pin (line 17) and then passes a valid pin number to the `gpio toggle` statement (line 19).



```
Raspberry PI GPIO

Toggle GPIO Pins
Select Pin:

( ) 7   GPIO_7  - LED 1 (BLUE) physical pin 7
(*) 0   GPIO_0  - LED 2 (YELLOW) physical pin 11
( ) 2   GPIO_2  - LED 3 (RED) physical pin 13
( ) 21  GPIO_21 - LED 4 (GREEN) physical pin 29

<Ok>                <Cancel>
```

Use a whiptail radiolist to control PI GPIO pins

Figure 4: Raspberry Pi radiolist project.

Listing 2: Whiptail Radiolist

```
01 #!/usr/bin/bash
02 #
03 # wradio.sh - toggle a GPIO output pin
04 #
05 OK=0
06 response="$OK"
07
08 # Cycle OK entered, Enter Cancel to exit
09 while [ "$response" == "$OK" ] ; do
10
11   thepin=$(whiptail --title "Toggle GPIO Pins"
       --backtitle "Raspberry PI GPIO"
       --radiolist "Select Pin:" 0 0 4
         7 "GPIO_7 - LED 1 (BLUE) physical pin 7 " off
         0 "GPIO_0 - LED 2 (YELLOW) physical pin 11" off
         2 "GPIO_2 - LED 3 (RED) physical pin 13" off
         21 "GPIO_7 - LED 4 (GREEN) physical pin 29 " off
       3>&1 1>&2 2>&3 )
12
13   # Get the response from the radio dialog
14   response=$? ; # Get the output from the dialog
15
16   # Check if a radio item returned a value
17   if [ ${#thepin} > 0  ] ; then
18     # Toggle the selected GPIO pin number
19     gpio toggle $thepin
20   fi
20 done
```

This project could be enhanced with a `checklist` box that shows the present status of all GPIO pins; then, rather than setting one pin, multiple pins could be set or reset in a single pass.

Pi Sensor Python Example

For the final project, I want to show Raspberry Pi sensor data in a Whiptail message box. My setup uses a low-cost DHT11 temperature and humidity sensor (about $3-$10), but the more accurate DHT22 module could also be used. These sensors have a three-pin configuration with power, ground, and signal connections.

The Python library for both the DHT11 and DHT22 sensors is installed by:

```
pip install Adafruit-DHT
```

A Python wrapper library offers all the Bash Whiptail functionality. Note that this library requires that the basic Whiptail utility be installed as a prerequisite. To install the Python library, enter:

```
pip install whiptail-dialogs
```

Once the DHT and Whiptail libraries are loaded, you can create some lightweight projects that connect the sensor data with text interfaces.

The example in Listing 3 imports the *Adafruit_DHT* and *Whiptail* libraries

Listing 3: Python DHT11 Sensor Project

```
#!/usr/bin/python3
#
# dht11_whip.py - DHT11 sensor data in a Whiptail message box
#
import Adafruit_DHT
from whiptail import Whiptail

# Get the sensor value for DHT11 on pin 4
humid, temp = Adafruit_DHT.read_retry(11, 4)

# Create a message string with sensor data
msg = f"{temp} C\n{humid} %"

# Create a whiptail object with titles and autosize
w = Whiptail(title="Temperature and Humidity",
             backtitle="Raspberry PI DHT11 Sensor",
             height = 0, width = 0)

# Show the sensor data in a message box
w.msgbox(msg)
```

(lines 5 and 6) and then reads the DHT11 sensor on pin 4 (line 9). The humidity and temperature values are put into a message string (line 12) that is shown in a Whiptail message box (line 15).

An alternative approach to the Python *Whiptail* library is to write all the code in a Bash script and then call Python in command-line mode. The Python command-line option `-c` executes a string of Python statements; for example:

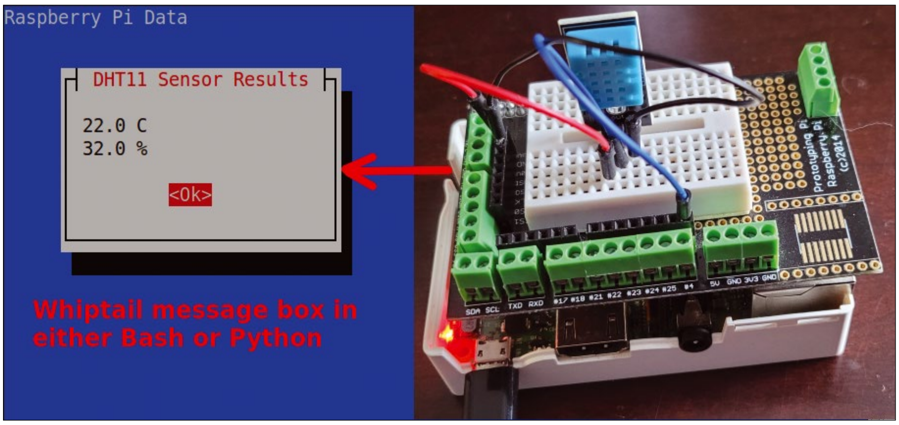
```
$ python3 -c ↩
  "a=4;b=5;print(f'{a}+{b}={a+b}')"
4+5=9
```

The Bash script in Listing 4 calls Python to return the sensor data as a Bash variable (`msg`; lines 8-15). Python imports the DHT library (line 11), reads the sensor (line 12), and then prints the results as a string (line 13). The Bash variable (`msg`) is then shown in a Whiptail message box (line 18).

Bash and Python scripts each have their own benefits and drawbacks, but the end result produces the same text interface. Figure 5 shows the Raspberry Pi hardware and the sensor data in a Whiptail message box interface.

Listing 4: Bash DHT11 Sensor Project

```
01 #!/usr/bin/bash
02 #
03 # dht11_whip.sh - Show DHT11 sensor data in a Whiptail message box
04 #               - Use Python to get the sensor data
05 #               - Python output is saved as a Bash variable
06
07 # Create a Bash variable based on Python output
08 msg=$(python3 -c """
09
10 # Use Adafruit_DHT Library with a DHT11 sensor on pin 4
11 import Adafruit_DHT
12 humid, temp = Adafruit_DHT.read_retry(11, 4)
13 print(f'{temp} C\n{humid} %')
14
15 """)
16
17 # Show a whiptail message box with the sensor data
18 whiptail --backtitle "Raspberry Pi Data"
         --title "DHT11 Sensor Results"
         --msgbox "$msg" 0 0;
```



```
Raspberry Pi Data
DHT11 Sensor Results
22.0 C
32.0 %
<Ok>
```

Figure 5: Raspberry Pi sensor project.

Summary

Whiptail, preinstalled on Raspbian, is a simple tool for creating text-based interfaces. If you need more functionality, take a look at the Dialog [3] utility, which supports all the Whiptail features and about a dozen more presentation boxes. Two of the features I especially liked about Dialog are inline custom colors and the ability to refresh pages with a time-out parameter. ■■■

Info

[1] Whiptail: *https://whiptail.readthedocs.io/*

[2] wiringPi: *https://github.com/WiringPi/WiringPi*

[3] Dialog: *https://invisible-island.net/dialog/*

Author

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

■■■

Linux users are always looking for ways to make things better: better performance, smoother operations, safer surfing. One largely overlooked but most satisfying way to make things better is through those little tweaks that add personality and elegance to the user experience. You don't have to take what comes up on first boot – make it your own! In this month's Linux Voice, we feature some cool desktop tools that might not change your life, but they could add a little incremental lift to your desktop experience. We start with a look at a pair of handy tools for managing your browser bookmarks, floccus and LinkAce. Then we take a stroll with FSearch, a fast and handy desktop search app. Finally, this month's tutorial shows how to search DuckDuckGo from the command line with ddgr.

Image © Olexandr Moroz, 123RF.com

LINUXVOICE

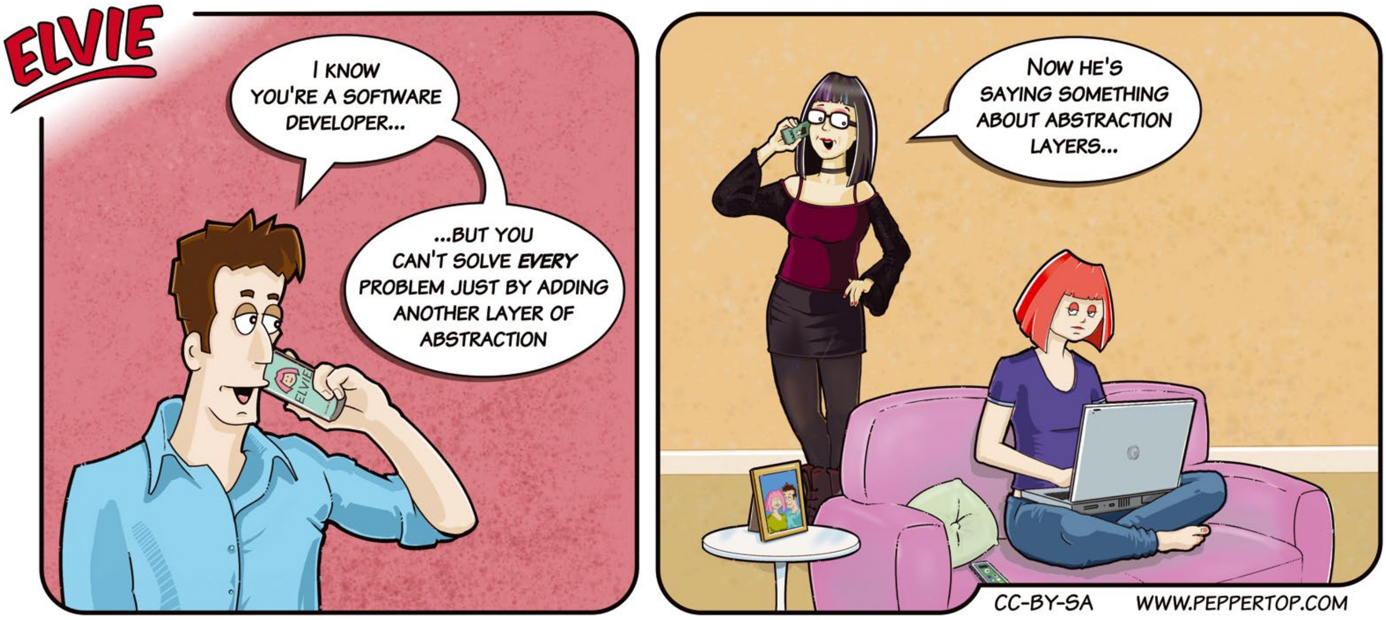
Doghouse – Old Farts 72
Jon "maddog" Hall
A new effort to record the history of open source is underway.

LinkAce/floccus 73
Ferdinand Thommes
Synchronize and manage bookmarks while storing your data locally.

FSearch 80
Ferdinand Thommes
In a crowded field of search applications, FSearch offers many interesting functions for quickly searching files and folders.

FOSSPicks 84
Graham Morrison
This month Graham looks at KiCad 7, Carbonyl, CliFM, markdownlint, MythTV 33.1, zelda3, and more.

Tutorial – ddgr 90
Nate Drake
Since 2008, DuckDuckGo has been making waves as an efficient and much more private search engine alternative to Google. The unaffiliated command-line tool ddgr makes it easy to run DuckDuckGo searches from the terminal.


ELVIE
I KNOW YOU'RE A SOFTWARE DEVELOPER...
...BUT YOU CAN'T SOLVE EVERY PROBLEM JUST BY ADDING ANOTHER LAYER OF ABSTRACTION
ELVIE
NOW HE'S SAYING SOMETHING ABOUT ABSTRACTION LAYERS...
CC-BY-SA
WWW.PEPPERTOP.COM

MADDOG'S DOGHOUSE

A new effort to record the history of open source is underway. BY JON "MADDOG" HALL

Historical Record

Recently a friend of mine posted about a new initiative to engage "Open Source Pioneers" to record their "legacies" with digital recordings about open source history. It seems to be a well-conceived (and even well-funded) effort to capture these stories, and timely, because (as my friend pointed out) many of the people who were in their 30s when the Linux kernel was started (1991) are now (GASP!) in their 60s and are obviously headed toward senility or death.

The creators of this project, called the Free and Open Source Stories Digital Archive Foundation (FOSSDA) [1], did manage to link in Richard M. Stallman and his efforts to start the GNU project and the Free Software Foundation, so the whole "FOSS Era" managed to be moved back a few more years to 1983, which happened to be the year that I joined Digital Equipment Corporation (DEC) to start work on their proprietary Unix systems … and yes, I acknowledge that Ultrix was closed source and proprietary to DEC.

Open source, for me, started in 1960 when I was 10 years old. My father subscribed to a number of magazines named "Popular <Something>." *Popular Science*, *Popular Mechanics*, and *Popular Electronics* were some of them, with *Popular Mechanics* and *Popular Electronics* being my two favorite. These magazines would talk about the technologies of the day and often would include blueprints and circuit diagrams for building things described in the articles – sometimes printed in parts over several months of the magazine.

As I got older, *Popular Electronics* [2] was the magazine I leaned towards. Electronic parts were very expensive in those days, and often I had to take apart old radios and TVs to get most of the parts I needed (resistors, capacitors, tubes, etc.) to build the things I wanted to build. A single transistor cost $1.50, and that was when you could fill up the tank of your car for $3.50.

This interest was increased when I started taking electronics classes in the last three years of high school. I was determined to study electrical engineering when I went to college.

Computers were not really in my view then, as computers were talked about in the magazines, but they were not in most high schools or even in many universities because they were so expensive.

However, when I went to university in 1968, my path to electrical engineering was diverted to software by the discovery that I could "build" things with software and digital computer logic easier than I could with the analog hardware of the day, and locating two small DEC PDP-8 minicomputers allowed me to learn interactive assembly language programming. I was hooked.

There were few "commercial" programs available to me at the time. Computing was still in its infancy and computer models were still measured in hundreds or thousands of units, not the millions and billions of today. Generating and distributing your programs in binary format was typically not worth the time and effort, and the software cost so much on a per-unit basis that often the software was delivered to the end user with the developer "attached" to the magnetic tape, and they would spend a couple of days getting their source code compiled and working on your computer. Software IP was protected by "trade secret" and contract law, not by copyright. Buying software was expensive.

However there were system vendor user groups, such as the Digital Equipment Corporation User's Society (DECUS) and IBM's SHARE, who had libraries of software contributed by their users and available to other users for the cost of duplicating (often to paper tape) and distribution. As a student I ordered a lot of the software and learned how to program in assembly language from studying it.

In the mid 1970s, as home computers like the Altair and other simple computers (some in kit form) were produced, there were "bulletin board" servers accessed by users through dial-up modems, and magazines like *Kilobaud Microcomputing* [3], *BYTE* [4], and *Dr. Dobb's Journal* [5] [6] would publish source code for programs, and readers would painstakingly type these into their home computers. As operating systems like CP/M emerged, some of these programs were in binary-only form, but many were delivered in source code, written in BASIC, assembly, or other languages.

And these magazines, including *Popular Electronics*, started publishing circuit diagrams that you would build if you were good with a soldering iron or a "perf board" and wire-wrapping tool.

So you see, "open source" did not start with coining the term, or GNU. There was a long and proud history of people sharing their skill and knowledge before that. From my viewpoint, open source started way before the GNU project; it is just that we did not call what we were doing "open source."

We just called it "code." ■■■

Info

[1] FOSSDA: *https://fossda.org/*

[2] *Popular Electronics*: *https://archive.org/details/Pop195710/Popular%20Electronics/Pop-1960-01/*

[3] *Kilobaud Microcomputing*: *https://archive.org/details/kilobaudmagazine*

[4] *BYTE*: *https://archive.org/download/BYTE-MAGAZINE-COMPLETE*

[5] *Dr. Dobb's Journal*: *https://archive.org/search?query=Dr.+dobbs+journal*

[6] "Dr. Dobb's Bites the Dust After 38 Years": *https://www.i-programmer.info/news/152-epub/8096-dr-dobbs-bites-the-dust-after-38-years.html*

Bookmark organization with floccus and LinkAce

Saved and Sorted

LinkAce and floccus synchronize and manage bookmarks while storing your data locally. BY FERDINAND THOMMES

Anyone who spends a lot of time on the web will repeatedly come across content that they want to save for further evaluation at a later stage. Traditionally, you would use your web browser's bookmark function for this. For example, if you want to continue reading a web page you discover on your way home from work on your computer at home, and you use the browser's bookmarks, this requires synchronization between the devices.

All modern browsers offer synchronization services for this. As an example, think of Firefox Sync [1]. The drawback of these services is that data such as bookmarks, passwords, open tabs, and more are stored on the browser provider's servers. Without wanting to imply anything about Mozilla, what better way can you imagine than this if you want to profile user behavior?

This is remedied by applications such as the floccus [2] browser extension or the self-hosted LinkAce [3] URL manager. Floccus focuses on the secure synchronization of bookmarks between different mobile and stationary platforms and different browsers using a private cloud instance over WebDAV, Nextcloud, or Google Drive. LinkAce's focus is more on archiving bookmarks and improving how they are organized on your own hardware.

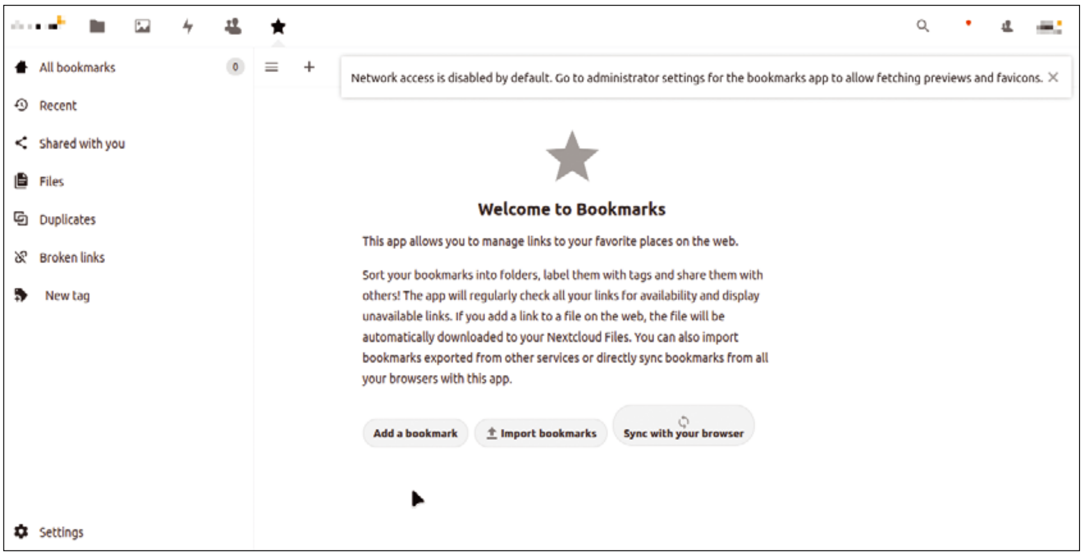
floccus

Imagine you want to sync the bookmarks you stored in Firefox between your devices, without using the Firefox account or other external services, by integrating your Nextcloud instance. This requires some preliminary work.

In Nextcloud, you need to install and activate an app named Bookmarks, version 0.14 or newer (Figure 1), via the app manager or the Nextcloud app store [4]. And you need to set up the *floccus bookmarks sync* extension in your browser of choice. The extension is also available for Android via F-Droid or the Google Play Store.

Backup

Before starting to synchronize, it is a good idea to back up your bookmarks. This can be done in Firefox via *Settings* | *Bookmarks* | *Manage bookmarks* | *Import and Backup*. When you get there,



Network access is disabled by default. Go to administrator settings for the bookmarks app to allow fetching previews and favicons.

Welcome to Bookmarks

This app allows you to manage links to your favorite places on the web.

Sort your bookmarks into folders, label them with tags and share them with others! The app will regularly check all your links for availability and display unavailable links. If you add a link to a file on the web, the file will be automatically downloaded to your Nextcloud Files. You can also import bookmarks exported from other services or directly sync bookmarks from all your browsers with this app.

Figure 1: The Bookmarks app is available for installation in the Nextcloud instance, as well as in the Nextcloud app store, as a prerequisite for running floccus.

you can choose to back up as a JSON file or export as HTML. In Chromium, the first step is also to go to *Bookmarks* in the settings and then to *Export bookmarks* via the hamburger menu in the upper right corner. Exporting as HTML makes sense because the file can be imported directly into Nextcloud Bookmarks via *Settings* at the bottom left (Figure 2).

At the same location you will find the *Add to Nextcloud* entry, which you can grab with the mouse pointer and drag to the browser's bookmark bar as a bookmarklet (Figure 3). This will allow bookmarks to be created directly in Nextcloud in the future. After installing floccus, the first thing to do is to set up one or more accounts. An account with floccus is only valid for one bookmark folder at a time. If you want the

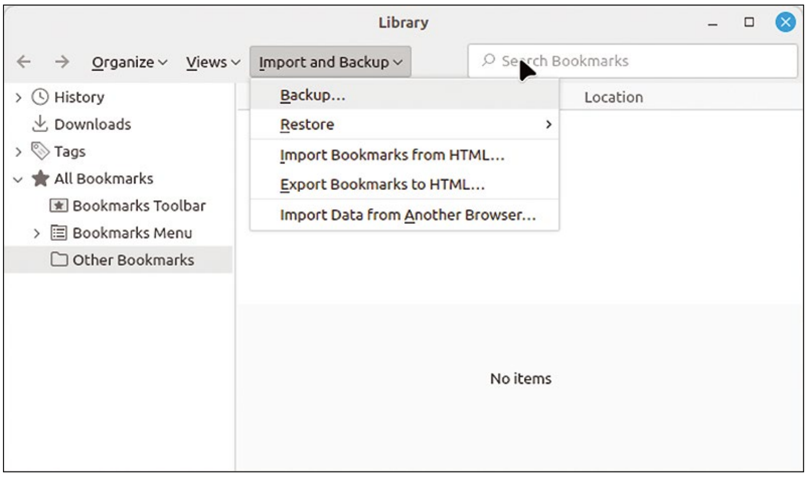


Figure 2: In the Firefox and Chromium settings, your first step will be to back up the existing bookmarks in HTML format.

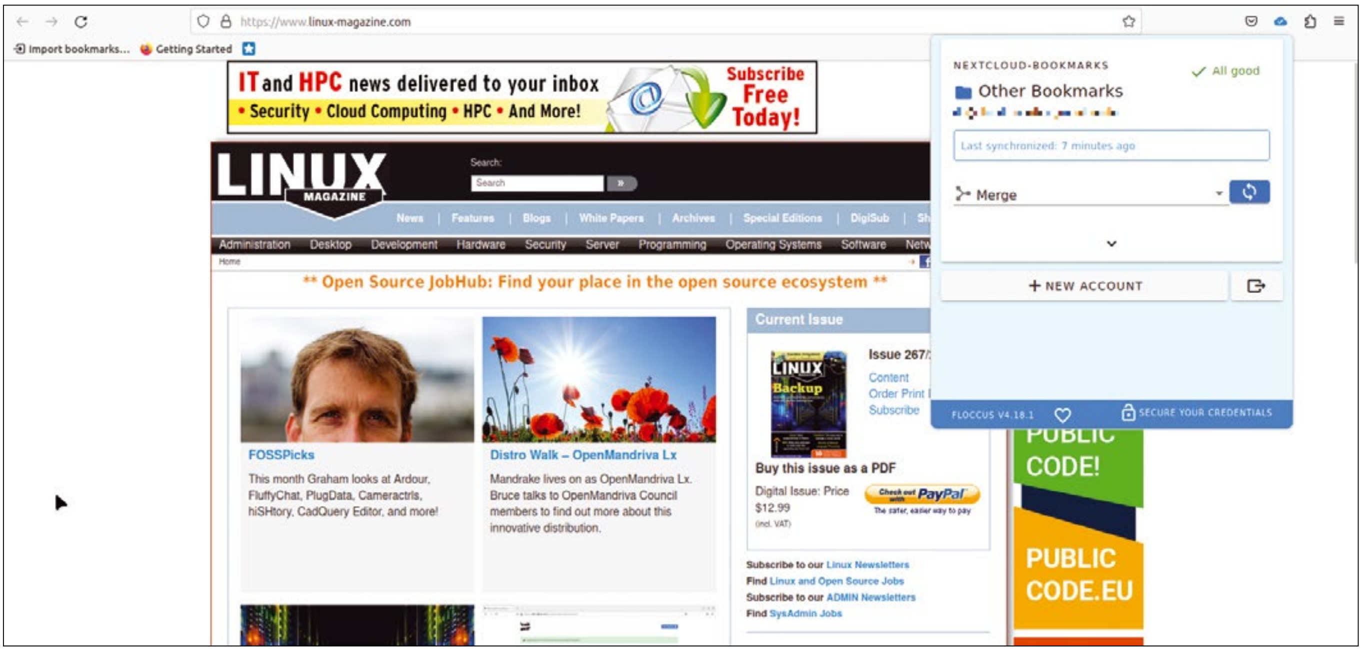
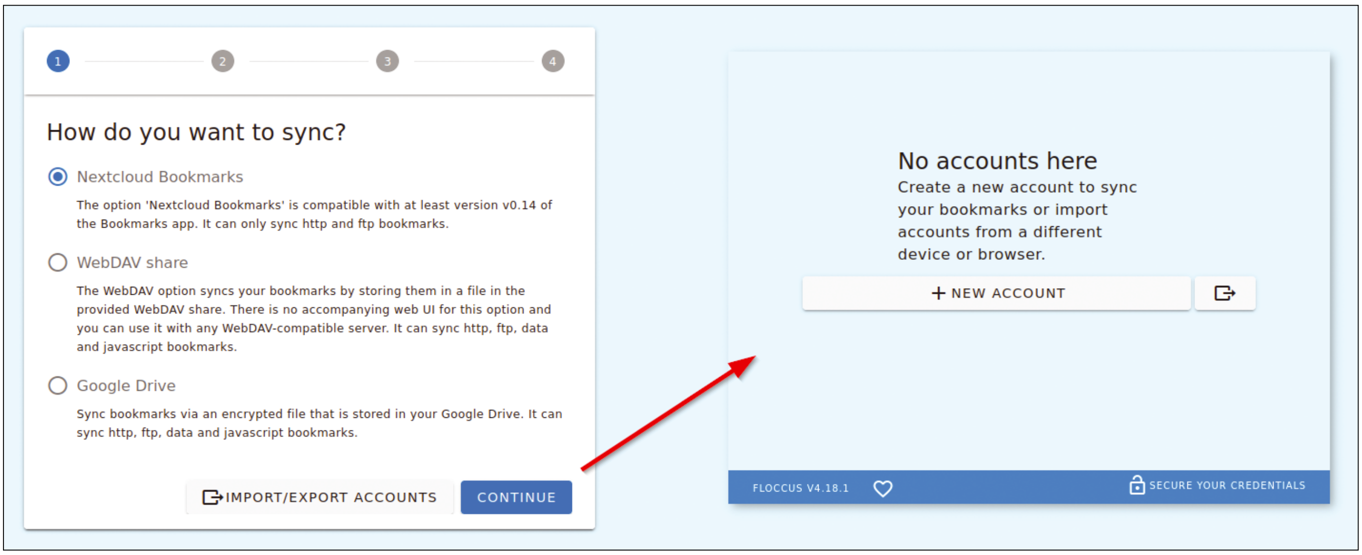

https://www.linux-magazine.com
Import bookmarks…
Getting Started
IT and HPC news delivered to your inbox
Security • Cloud Computing • HPC • And More!
Subscribe Free Today!
LINUX MAGAZINE
Search
News
Features
Blogs
White Papers
Archives
Special Editions
DigiSub
Administration
Desktop
Development
Hardware
Security
Server
Programming
Operating Systems
Software
** Open Source JobHub: Find your place in the open source ecosystem **
FOSSPicks
This month Graham looks at Ardour, FluffyChat, PlugData, Cameractrls, hiSHtory, CadQuery Editor, and more!
Distro Walk – OpenMandriva Lx
Mandrake lives on as OpenMandriva Lx. Bruce talks to OpenMandriva Council members to find out more about this innovative distribution.
Current Issue
Buy this issue as a PDF
Digital Issue: Price $12.99
Check out with PayPal
Subscribe to our Linux Newsletters
Find Linux and Open Source Jobs
Subscribe to our ADMIN Newsletters
Find SysAdmin Jobs
PUBLIC CODE!
PUBLIC CODE.EU
NEXTCLOUD-BOOKMARKS
All good
Other Bookmarks
Last synchronized: 7 minutes ago
Merge
+ NEW ACCOUNT
FLOCCUS V4.18.1
SECURE YOUR CREDENTIALS

Figure 3: Using the bookmarklet, which is stored in the bookmarks toolbar, any web pages you visit can easily be stored in Nextcloud or on another storage medium for later use.



How do you want to sync?

● Nextcloud Bookmarks

The option 'Nextcloud Bookmarks' is compatible with at least version v0.14 of the Bookmarks app. It can only sync http and ftp bookmarks.

○ WebDAV share

The WebDAV option syncs your bookmarks by storing them in a file in the provided WebDAV share. There is no accompanying web UI for this option and you can use it with any WebDAV-compatible server. It can sync http, ftp, data and javascript bookmarks.

○ Google Drive

Sync bookmarks via an encrypted file that is stored in your Google Drive. It can sync http, ftp, data and javascript bookmarks.

No accounts here

Create a new account to sync your bookmarks or import accounts from a different device or browser.

Figure 4: Floccus offers synchronization via Nextcloud, WebDAV, or with automatic encryption via Google Drive.

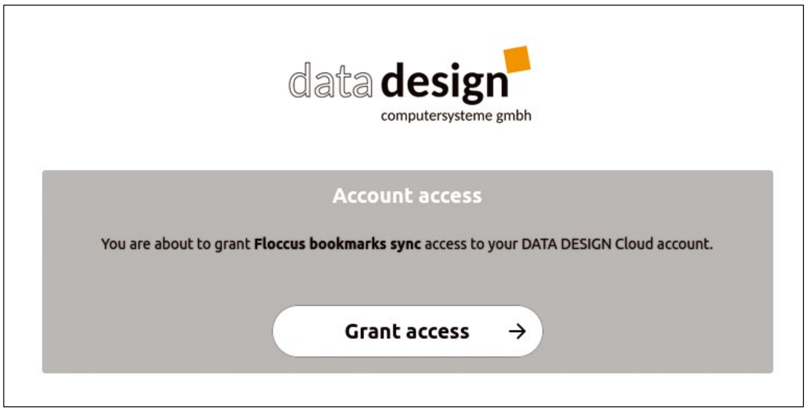
bookmarks to behave the same way as in the browser, you need at least two accounts. One is for the bookmarks themselves, the other for the bookmarks toolbar.

Creating Accounts

In the account setup window, your first choice is between the generic WebDAV protocol, Google Drive, or Nextcloud as the central location for synchronization. I opted for Nextcloud instance for my example (Figure 4).

In the next step, enter the URL of your Nextcloud instance in the address bar (Figure 5) to connect. Then specify the Nextcloud folder in which you want to store the bookmarks and the browser directory that contains the bookmarks. To do this, first create folders at the top level in Nextcloud and enter them with your name as the server folder, for example, `/Bookmarks-FF`. The initial slash is important here.

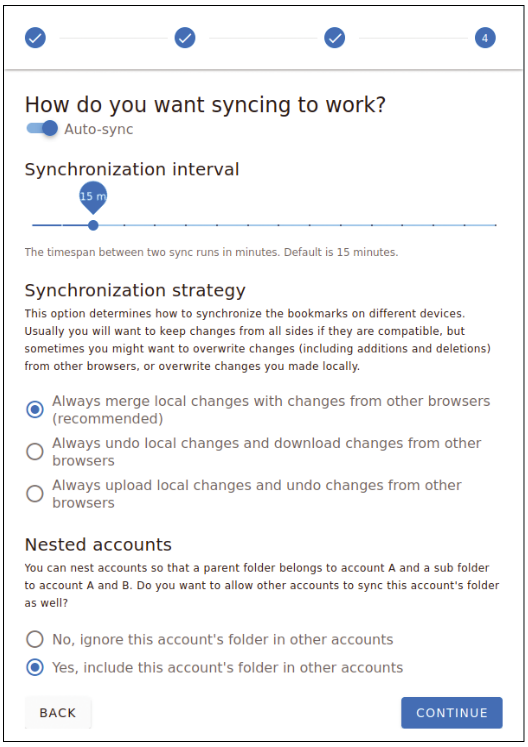
When you are prompted for the local folder, select the bookmark folder to be synchronized. For Firefox select *Bookmarks menu*, for Chromium *Other bookmarks*. A second account for the toolbars will have the entry *Bookmark Toolbar* for Firefox or *Bookmark Bar* for Chromium. In the next window, you specify whether to synchronize automatically and at what time intervals (Figure 6). On top of this, you can decide


data design computersysteme gmbh

Account access

You are about to grant **Floccus bookmarks sync** access to your DATA DESIGN Cloud account.

Figure 5: Connecting to your Nextcloud instance is just a matter of a few clicks. You only need the URL and your access credentials.



How do you want syncing to work?

● Auto-sync

Synchronization interval

15 m

The timespan between two sync runs in minutes. Default is 15 minutes.

Synchronization strategy

This option determines how to synchronize the bookmarks on different devices. Usually you will want to keep changes from all sides if they are compatible, but sometimes you might want to overwrite changes (including additions and deletions) from other browsers, or overwrite changes you made locally.

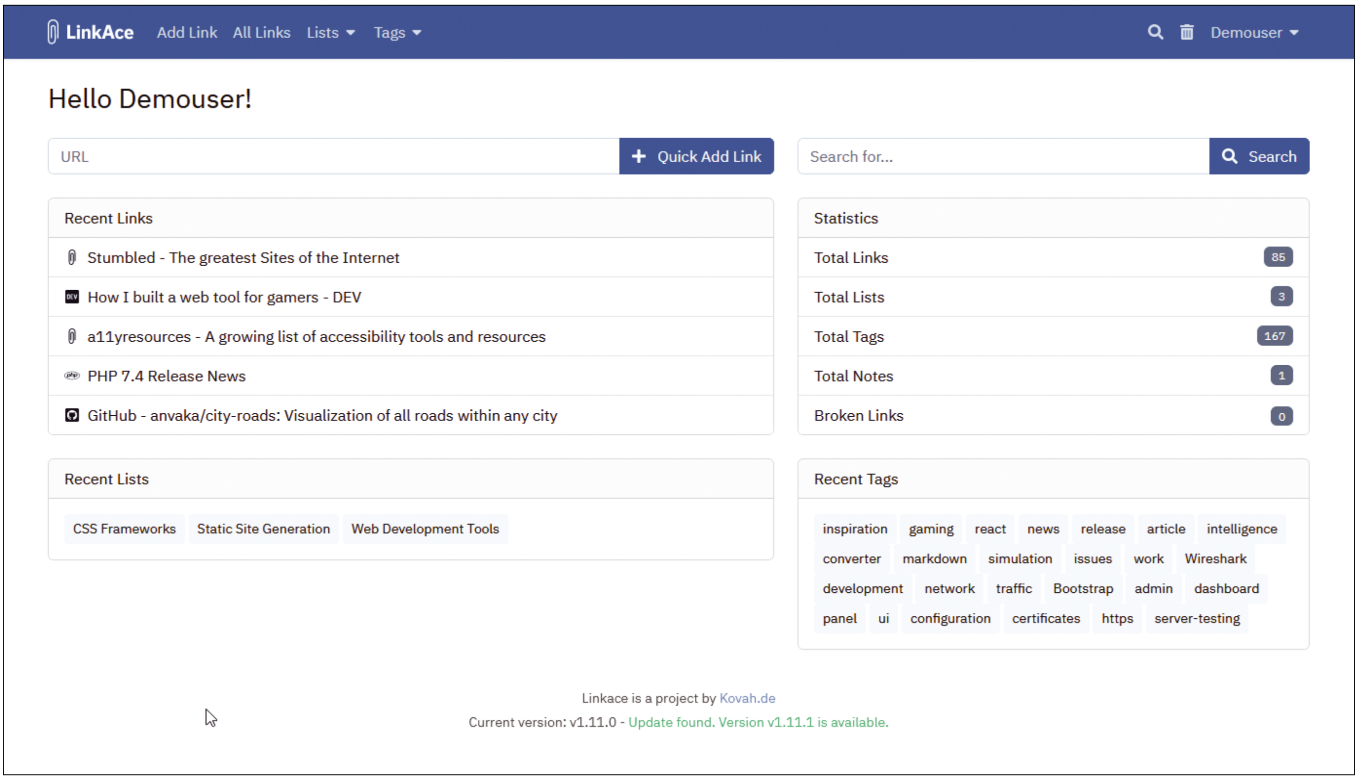
- ● Always merge local changes with changes from other browsers (recommended)
- ○ Always undo local changes and download changes from other browsers
- ○ Always upload local changes and undo changes from other browsers

Nested accounts

You can nest accounts so that a parent folder belongs to account A and a sub folder to account A and B. Do you want to allow other accounts to sync this account's folder as well?

- ○ No, ignore this account's folder in other accounts
- ● Yes, include this account's folder in other accounts

Figure 6: In the floccus settings, you can define how and how often synchronization will occur.



Hello Demouser!

URL

Search for...

Recent Links

- Stumbled - The greatest Sites of the Internet
- How I built a web tool for gamers - DEV
- a11yresources - A growing list of accessibility tools and resources
- PHP 7.4 Release News
- GitHub - anvaka/city-roads: Visualization of all roads within any city

Statistics	
Total Links	85
Total Lists	3
Total Tags	167
Total Notes	1
Broken Links	0

Recent Lists

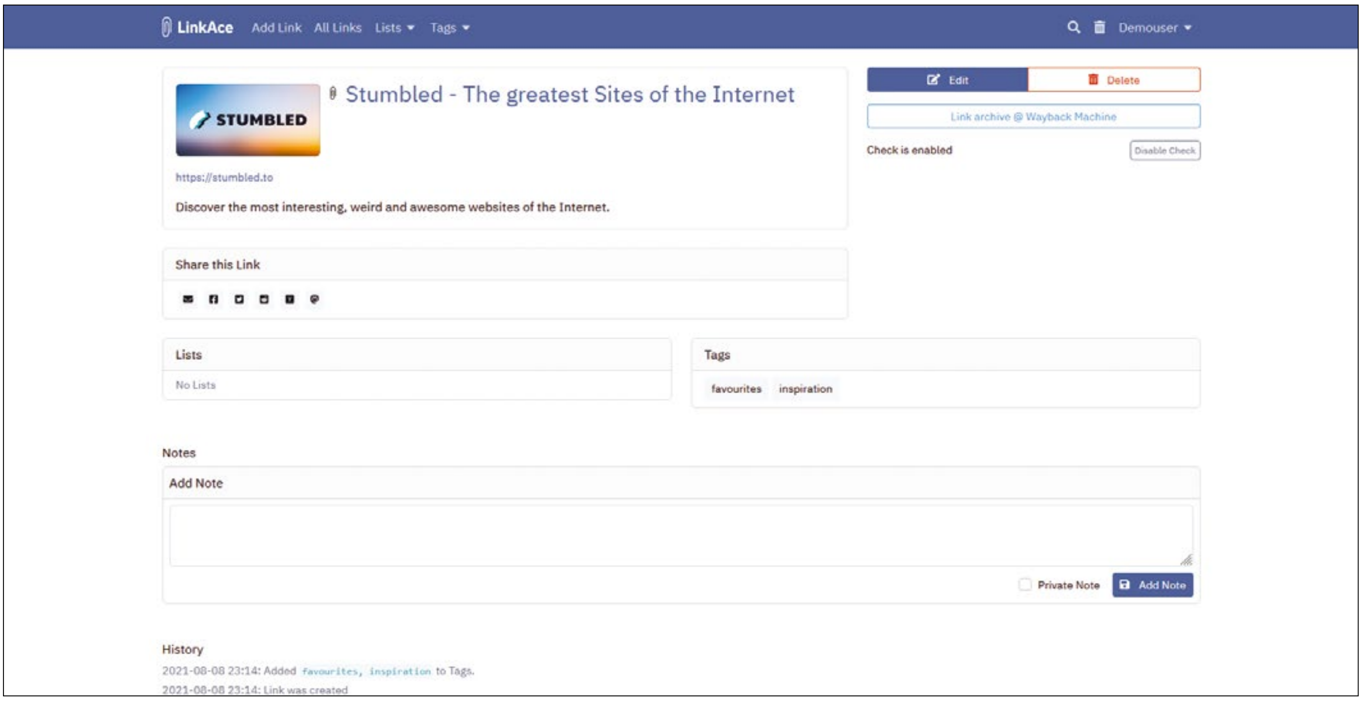
CSS Frameworks · Static Site Generation · Web Development Tools

Recent Tags

inspiration gaming react news release article intelligence converter markdown simulation issues work Wireshark development network traffic Bootstrap admin dashboard panel ui configuration certificates https server-testing

Linkace is a project by Kovah.de
Current version: v1.11.0 - Update found. Version v1.11.1 is available.

Figure 7: LinkAce's standard interface clearly displays the most recentl links as well as an overview of links, lists, and tags. A broken link will have a red icon.



Stumbled - The greatest Sites of the Internet

https://stumbled.to

Discover the most interesting, weird and awesome websites of the Internet.

Check is enabled

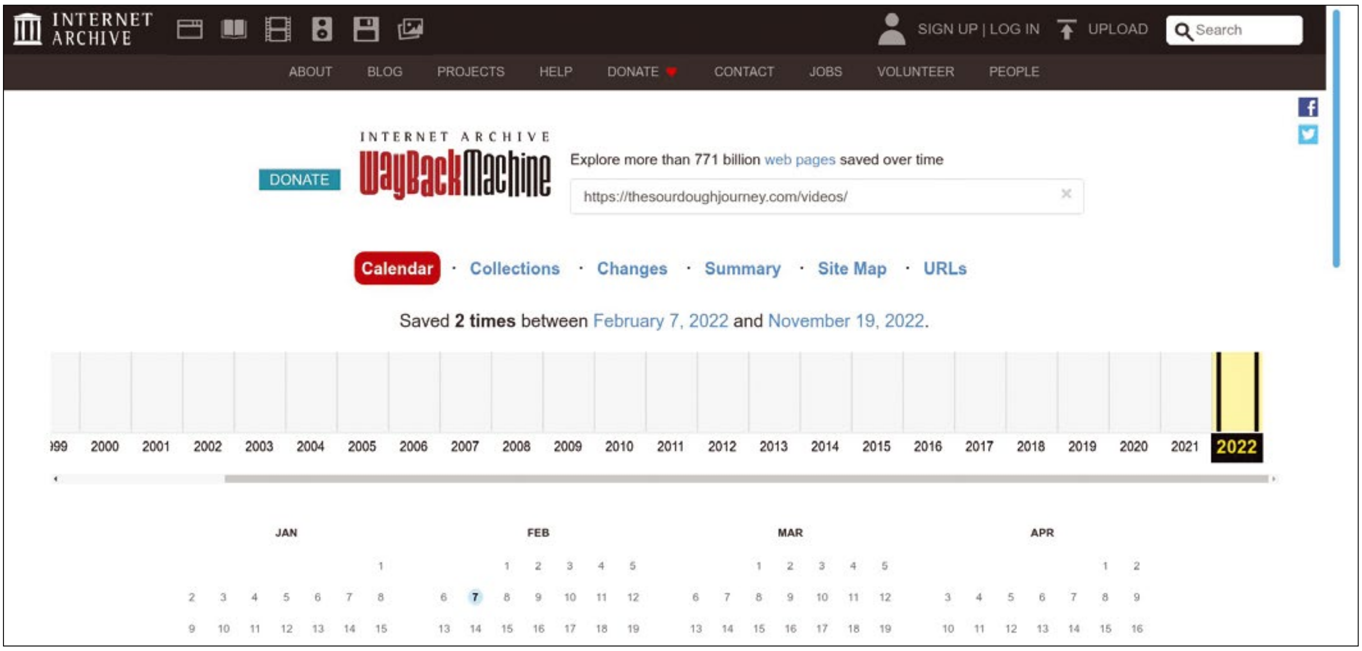
No Lists

favourites inspiration

2021-08-08 23:14: Added `favourites, inspiration` to Tags.

2021-08-08 23:14: Link was created

Figure 8: Clicking on *All Links* shows you an overview of all saved links with a thumbnail of the corresponding web page.

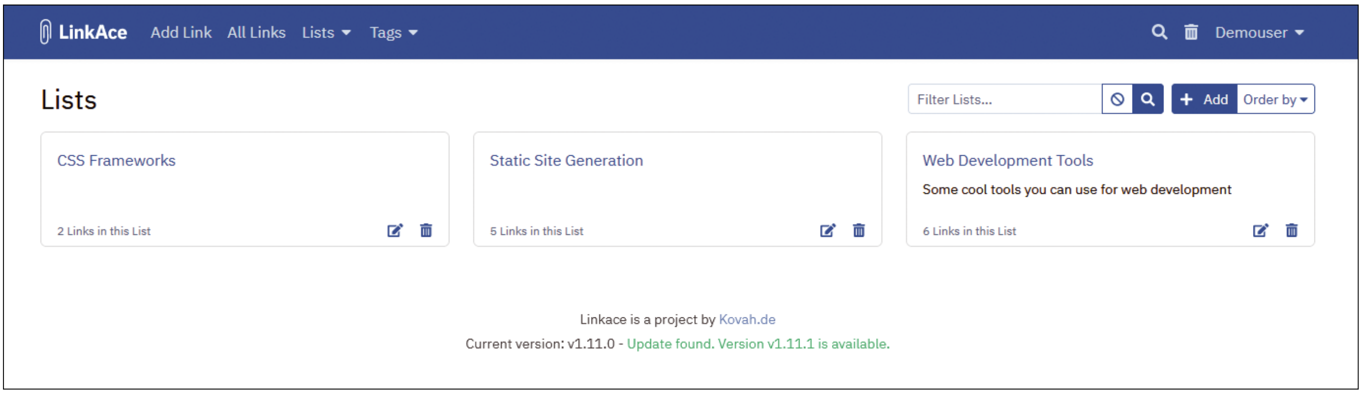


Explore more than 771 billion web pages saved over time

https://thesourdoughjourney.com/videos/

Saved **2 times** between February 7, 2022 and November 19, 2022.

Figure 9: You can share, edit, delete, send to the Wayback Machine, or add a note to an opened link. The history for this link is displayed bottom left.



Lists

CSS Frameworks
2 Links in this List

Static Site Generation
5 Links in this List

Web Development Tools
Some cool tools you can use for web development
6 Links in this List

Linkace is a project by Kovah.de
Current version: v1.11.0 - Update found. Version v1.11.1 is available.

Figure 10: Clicking on *Lists* shows all the lists you have created so far; lists can be filtered, edited, or deleted.

whether this will be bidirectional. Floccus can be protected with a password, which you set at the bottom of the window via *Secure credentials*.

Do the same with other devices by installing floccus and configuring it accordingly. The software offers a manual [5] and an FAQ [6] for support. If you want to synchronize your add-ons, history, and open tabs along with your bookmarks, a self-hosted Firefox sync server is the better choice [7].

LinkAce

The LinkAce web application is not about synchronizing the existing bookmarks, but about an independent collection of bookmarks for later evaluation and archiving. A Raspberry Pi is all you need for this. I used a Proxmox container in testing. LinkAce uses PHP and requires a database in the background. You can choose between MySQL, MariaDB, PostgreSQL, or SQLite. The fastest way to set up LinkAce is through Docker Compose.

But before I get to the installation, I will first look at what LinkAce aims to do and how it goes about it. The intention of the German open source developer Kevin Woblick [8] is to make it easier for users to store, archive, mark, classify, and retrieve bookmarks. LinkAce's interface is reduced to the bare essentials, well structured and clearly laid out (Figure 7).

Clear Cut

The app offers three ways to add new URLs. First, the existing bookmarks in the browsers can be imported as HTML. You can also enter URLs directly in the interface (Figure 7). There is also a bookmarklet that you can drag into your web browser's bookmarks bar to interactively store the URLs of the web pages you visit in the database.

LinkAce then takes care of the saved URLs. It will also perform a periodic Line Check if enabled (Figure 8). If a link is no longer available or has been moved, you will be notified and given the details. In addition, saved links can be backed up to the Internet Archive's Wayback Machine (Figure 9) [9]. All saved links can be exported as HTML, imported into a browser, or kept as a backup. Other supported backup targets include the commercial AWS [10] and S3 [11].

Tags and Lists

Tags and lists are used to classify links. Lists serve the purpose of bundling multiple links relating to a specific topic (Figure 10), while tags are used for general categorization. LinkAce also lets you enable optional guest access; the guest can see all links that you do not classify as *private*. Access can be controlled separately for each link, tag, and list.

Another way to share links is through link sharing, which is currently available for 20 different services. With an API key for the LinkAce API, the app can also be connected to other tools, which can then access and edit the data in LinkAce.

Installation

If you like the idea, you can test the LinkAce demo [12] without any obligations before proceeding to install. There are three options to choose from for setting up on your own system. The easiest way is to use a prepared Docker Compose file that installs all the required components [13]. If you want more control over the process, you can use the Docker image from Docker Hub [14] as an alternative. As a third option, you can install all the components manually on your server [15].

I implemented the Docker Compose method for my test; this is also a good choice for

Listing 1: Setting Up Docker

```
### Setting up Docker
$ sudo apt update
$ sudo apt install apt-transport-https ca-certificates curl softwareproperties-common
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu jammy stable"
### Set up Docker CE
$ sudo apt update
$ sudo apt install docker-ce
$ sudo systemctl status docker
$ sudo usermod - aG docker ${USER}
### Set up Docker Compose
$ sudo curl -L "https://github.com/docker/compose/releases/download/v2.13.0/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
### Determine the Docker version
$ sudo chmod +x /usr/local/bin/docker-compose
$ docker-compose --version
```

newcomers. I used an Ubuntu Server 22.04 as the basis. To set up Docker and its associated components on your machine, proceed as shown in Listing 1. If you are using a different Ubuntu version, replace the `jammy` specification from line 5 with the codename of the Ubuntu release you are using.

You can check on *https://github.com/docker/compose/releases* if version 2.13.0 (line 12) is still the latest. The entries from lines 14 and 15 let you determine the currently installed version.

The further procedure is described in the LinkAce documentation [16]. First of all, you need to choose between the basic and advanced version of LinkAce. The basic version, which delivers the application and the web server in a container, is fine if you only want to use LinkAce locally on your network. For the advanced variant, you need to install the software, web server, and database in separate containers. Among other things, this simplifies the use of a reverse proxy for securing via HTTPS if you want the application to be accessible from the outside.

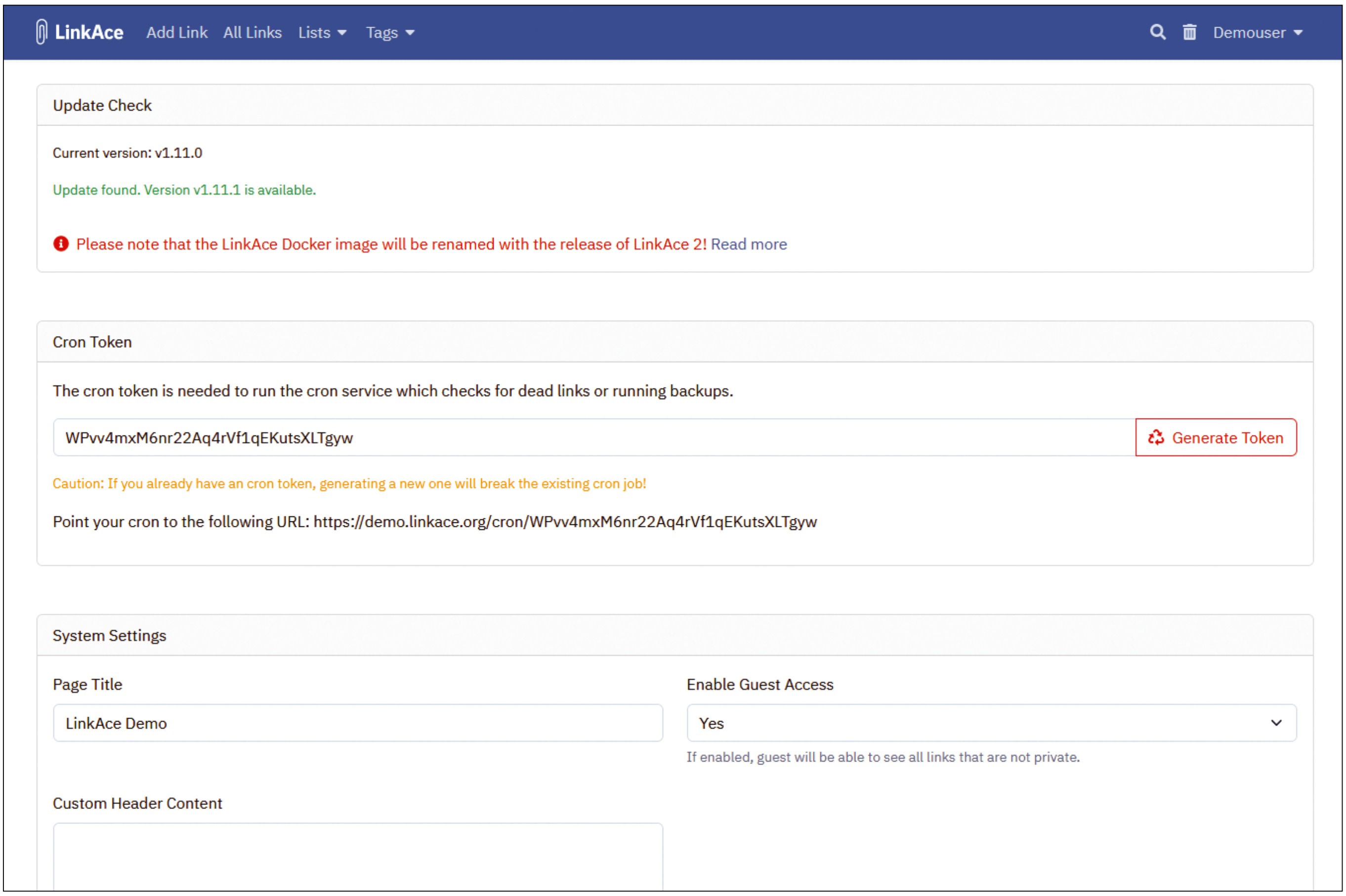
In terms of the installation procedure, both variants are the same, except that you need to download a different Compose file. I will use the basic method here. The procedure for using a reverse proxy or a load balancer in the extended variant can also be found in the documentation [16].

First, download `linkace-docker-simple.zip` or `linkace-docker-advanced.zip` to a previously created folder in your home directory and unzip the ZIP file. The basic variant contains the files `docker-compose.yml` and the hidden file `.env` in addition to license information and a README. The extended variant adds the `nginx.conf` and `nginx.ssl.conf` files.

Setting Passwords

Before proceeding, you need to at least change the `POSTGRES_PASSWORD` and `REDIS_PASSWORD` entries in the `.env` file. If you use a database other than MySQL, then replace the name there to match. After saving, the `docker-compose up -d` command is used for the final setup work. The setup can take a few minutes – after all, the call rolls out the application, web server, and database in line with the specifications in the compose file. When done, you will see a URL that lets you launch the LinkAce web interface.

Some more steps to take before using LinkAce, partially optional, are summarized in the Post Setup [17] section of the documentation. I will look at the most important of these. First, switch to *Settings* in the top right corner. You will see the bookmarklet, which you need to drag into your bookmarks bar. Just below, you have the functions for creating an API token.



Update Check

Current version: v1.11.0

Update found. Version v1.11.1 is available.

Please note that the LinkAce Docker image will be renamed with the release of LinkAce 2! Read more

Cron Token

The cron token is needed to run the cron service which checks for dead links or running backups.

WPvv4mxM6nr22Aq4rVf1qEKutsXLTgyw

Caution: If you already have an cron token, generating a new one will break the existing cron job!

Point your cron to the following URL: https://demo.linkace.org/cron/WPvv4mxM6nr22Aq4rVf1qEKutsXLTgyw

System Settings

Page Title: LinkAce Demo

Enable Guest Access: Yes

If enabled, guest will be able to see all links that are not private.

Custom Header Content

Figure 11: Among other things, you can generate a token in the settings; you need this for time-controlled cron jobs to work correctly.

Create a user including a password and an email address. If you want to add more security with two-factor authentication, you will find instructions for this in the User Settings documentation [18].

You can then set your preferences for the language, time format, privacy, and some preferences for the web interface. At the bottom of the page, you decide whether you prefer Light or Dark Mode, or whether you want to follow your desktop environment's default settings. Last but not least, choose the social media platforms on which you want to share your links, if any.

In addition to the *Settings*, there are the *System Settings*, which partly overlap. The first thing to do here is to create a cron token. It is used to periodically check your links, for backups, and for a backup in the Wayback Machine. The procedure for setting up a cron job for LinkAce is shown in the "Cron Job for LinkAce" box (Figure 11).

LinkAce offers more options than standard bookmark management in the web browser when it comes to sorting, searching, and categorizing. You can change individual links from public to private and vice versa at will. A description of any length can be added to each link. Deleted links remain in the Recycle Bin until you empty it. In addition, there are RSS feeds for private and public links [19]. Some commands can be used for administration via a command-line interface (CLI) interface [20]. Upgrades from LinkAce are also easy, as evidenced by the documentation [21].

Conclusions

Floccus and LinkAce both aim to improve the bookmark handling experience, but with different strategies. Floccus provides secure synchronization where the data remains under your own control. LinkAce instead focuses on uncomplicated management of even the largest link collections. The developer is happy to accept suggestions on GitHub for additional features in the upcoming version 2. This is also where you will find the forum for the application [22]. No matter which application you choose, the good thing is that no data ends up in clouds belonging to Google, Mozilla, or other providers. Instead, everything remains under your own control. ■■■

Cron Job for LinkAce

LinkAce needs a cron job to check the links, add them to the Wayback Machine, and create backups. The easiest way to create one is to use the `crontab -e` command. Then create a job as in the following:

```
* * * * * wget -qO- ⮐
  http(s)://linkace-examplecom/cron/⮐
  WPvv4mxM6nr22Aq4rVf1qEKutsXLTgyw
```

The part following `.../cron/` is created as a cron token in the system settings.

Info

[1] Firefox Sync: *https://www.mozilla.org/en/firefox/sync/*

[2] floccus: *https://floccus.org*

[3] LinkAce: *https://www.linkace.org*

[4] Bookmark app for Nextcloud: *https://github.com/nextcloud/bookmarks*

[5] floccus Guide: *https://floccus.org/guides*

[6] floccus FAQ: *https://floccus.org/faq*

[7] Self-hosted Firefox sync server: *https://mozilla-services.readthedocs.io/en/latest/howtos/run-sync-1.5.html*

[8] Kevin Woblick: *https://kovah.de/*

[9] Wayback Machine: *https://en.wikipedia.org/wiki/Internet_Archive#Wayback_Machine*

[10] AWS: *https://aws.amazon.com*

[11] S3: *https://cloud.support.dracoon.com/hc/en/articles/360001402540-ber-S3-Simple-Storage-Service-*

[12] LinkAce demo: *https://demo.linkace.org/*

[13] Installation via Docker Compose: *https://www.linkace.org/docs/v1/setup/setup-with-docker/*

[14] Installation via Docker Hub: *https://hub.docker.com/r/linkace/linkace/*

[15] Installing the server: *https://www.linkace.org/docs/v1/setup/setup-without-docker/*

[16] LinkAce setup with Docker: *https://www.linkace.org/docs/v1/setup/setup-with-docker/*

[17] LinkAce post-setup steps: *https://www.linkace.org/docs/v1/setup/post-setup/*

[18] LinkAce user settings: *https://www.linkace.org/docs/v1/configuration/user-settings/*

[19] LinkAce RSS feeds: *https://www.linkace.org/docs/v1/application/rss-feeds/*

[20] LinkAce commands via CLI: *https://www.linkace.org/docs/v1/cli/*

[21] LinkAce upgrades: *https://www.linkace.org/docs/v1/upgrade*

[22] LinkAce forum on GitHub: *https://github.com/Kovah/Linkace/discussions?page=3*

Find files and directories with FSearch

Quick Finder

In a crowded field of search applications, FSearch offers many interesting functions for quickly searching files and folders, with more promised in the future.

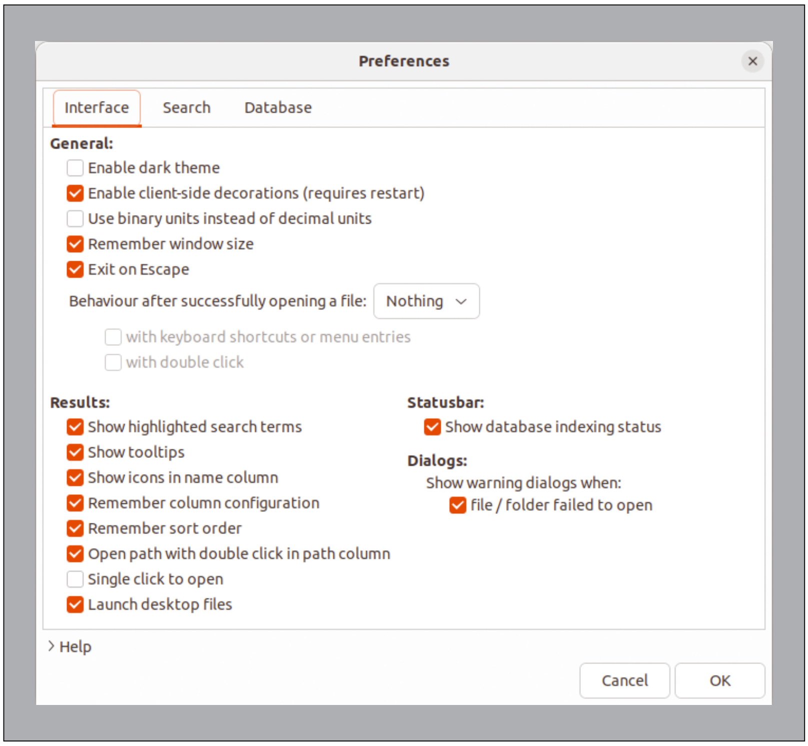
BY FERDINAND THOMMES

Data has the bad habit of hiding away in the depths of the filesystem precisely when you need it urgently. That's why there are tools at different levels to help with searching, including FSearch v0.2.2. Similar to the Findutils for `locate`, FSearch first builds a database of the selected data, which can then be searched at lightning speed. The search begins as soon as you start entering the search term.

Great Selection

It is important to distinguish between desktop environment search tools and generic search tools. In terms of user interfaces, KDE Plasma offers the KRunner, KFind, and Baloo programs, the latter being the underpinnings for searching with the Dolphin file manager. Gnome comes with Tracker integrated into the Nautilus file manager by default. In addition, there are many search apps that do not depend on a desktop and can be set up retroactively. The best-known candidates here include Catfish, ANGRYsearch, Recoll, and FSearch. At the command line, there are the tried-and-trusted `find` and `locate` tools, and others.

Figure 1: The *Interface* tab in the settings gives users options for designing the interface and outputting the results.



General:
☐ Enable dark theme
☑ Enable client-side decorations (requires restart)
☐ Use binary units instead of decimal units
☑ Remember window size
☑ Exit on Escape
Behaviour after successfully opening a file: Nothing
☐ with keyboard shortcuts or menu entries
☐ with double click

Results:
☑ Show highlighted search terms
☑ Show tooltips
☑ Show icons in name column
☑ Remember column configuration
☑ Remember sort order
☑ Open path with double click in path column
☐ Single click to open
☑ Launch desktop files

Statusbar:
☑ Show database indexing status

Dialogs:
Show warning dialogs when:
☑ file / folder failed to open

This article covers the open source FSearch tool, whose search speed is hard to beat and which comes with some interesting features. German developer Christian Boxdörfer was inspired to create FSearch [1] by the speed of the Everything search engine [2] on Windows. First released in 2016, the C program is based on the GTK3 toolkit. The developer is already planning a conversion to the Qt framework, but the app already integrates quite well with Qt-based systems.

FSearch has a clear-cut graphical interface. If you enable client-side decorations in the Preferences dialog's *Interface* tab (Figure 1), the menubar will then appear in a hamburger menu top right (Figure 2).

Installation

Despite its good qualities, FSearch has not yet arrived in the archives of all distributions. Only MX Linux, Solus, PCLinuxOS, GNU Guix, and FreeBSD have the official binaries. Arch Linux offers the tool in its Arch User Repository (AUR).

Fortunately, the Boxdörfer provides packages for Debian and openSUSE via SUSE's Open Build Service [3]. Fedora and RHEL users will find the program in the Copr repository [4], while Ubuntu users can access it via a Personal Package Archive (PPA) [5]. A Flatpak can be found on Flathub, but with limited functionality [6]. The developer also offered a Snap package, but the package format had so many restrictions that he switched to a PPA.

The current version, FSearch v0.2.2, was released in August 2022. The alpha version 0.3 is available for Ubuntu as a daily build, from the AUR as a Git dump, and for Fedora as a nightly build. The application can also be compiled from the source code without major issues [7]. I will stick to the stable version in this article, using Debian and Fedora for testing. The end of the article reveals what Boxdörfer has planned for upcoming releases.

First Launch

When you launch the application for the first time, you will see a notice that the database is empty. A switch lets you specify the folders you want to include or exclude from the search (Figure 3). Then indexing begins – this can take some time, depending on the size of the filesystem and the number of directories you add. However, on subsequent launches, loading the database only takes a moment.

On my fairly large test system with over 2.3 million entries in the FSearch database, the index is 104MB, and the first indexing run took about 10 minutes. The FSearch configuration file is located in `~/.config/fsearch.conf` and the database in `~/.local/share/fsearch.db`.

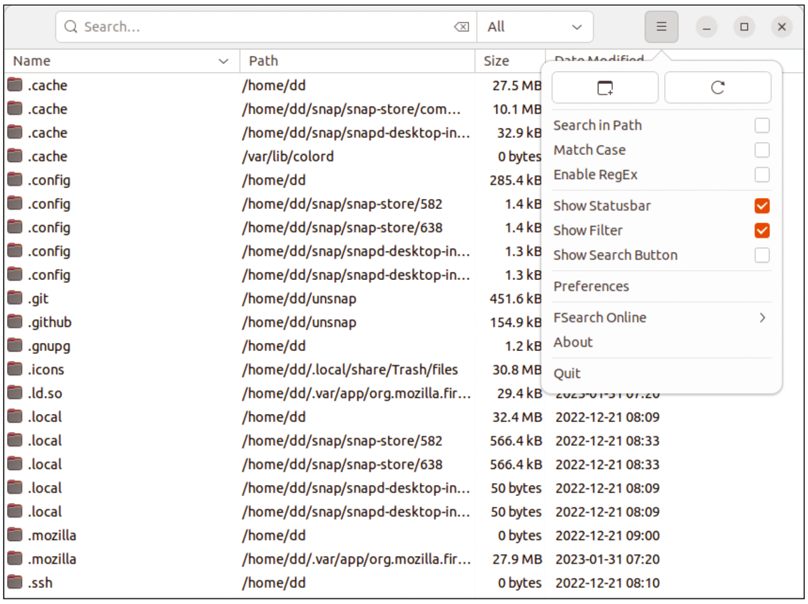
The top line in the FSearch window is occupied by the search mask. To the right, a drop-down menu lets you limit the search to specific file types, even after the search results are made available. The hamburger menu or, depending on the setting, the menubar, lets users open another window, manually update the database, and perform other tasks. There are also various options for disabling the search button, filters, or status bar. The status bar is located at the bottom of the window and displays the number of matches for the current search on the left and the total number of indexed files on the right.

Why FSearch?

So what makes FSearch such an excellent search tool? First and foremost, probably the speed. By default, the software starts searching as soon as you type the first letters of the search word and usually finishes displaying the matches before you complete the search term (in case you are wondering why the search button can be turned off). This instant search simply makes an additional button superfluous in the default setting.

The matches can be sorted at the top of the results list by file name, path, size, or modification time. However, the software does not support full-text searching; it only searches for the names of files and directories. If you need full-text searching, try Recoll or DocFetcher.

In the settings (Figure 3) below *Database*, you can include or exclude paths or hidden files and folders from the search. For example, you can use a wildcard to exclude all files with "flatpak" in their name by specifying `flatpak*` (Figure 4). If you are including a wildcard, a checkbox labeled *One Filesystem* appears after each entry (otherwise it is grayed out), which lets you exclude a network drive that may or may not be mounted in `/home` from the search.



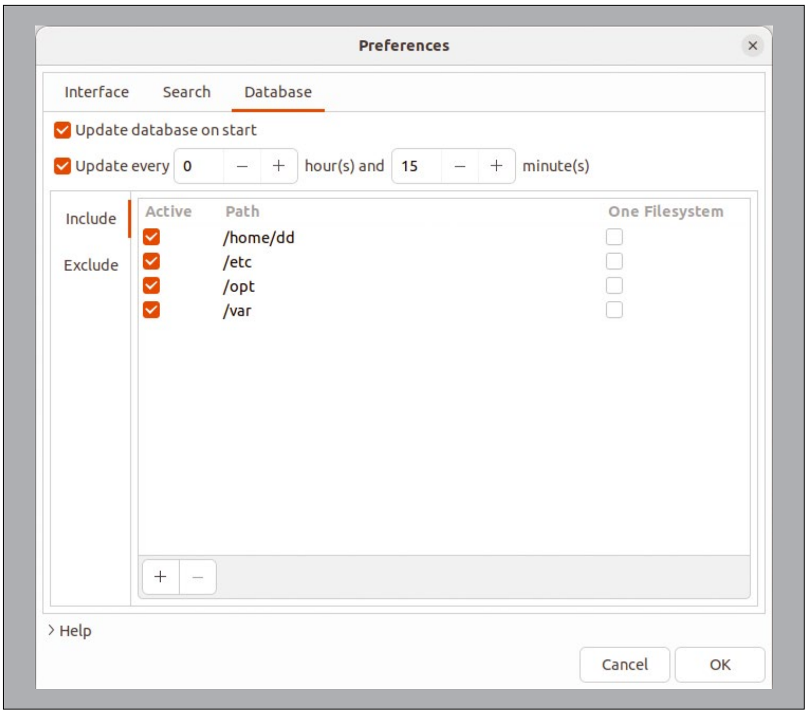
Name	Path	Size	Date Modified
.cache	/home/dd	27.5 MB	
.cache	/home/dd/snap/snap-store/com...	10.1 MB	
.cache	/home/dd/snap/snapd-desktop-in...	32.9 kB	
.cache	/var/lib/colord	0 bytes	
.config	/home/dd	285.4 kB	
.config	/home/dd/snap/snap-store/582	1.4 kB	
.config	/home/dd/snap/snap-store/638	1.4 kB	
.config	/home/dd/snap/snapd-desktop-in...	1.3 kB	
.config	/home/dd/snap/snapd-desktop-in...	1.3 kB	
.git	/home/dd/unsnap	451.6 kB	
.github	/home/dd/unsnap	154.9 kB	
.gnupg	/home/dd	1.2 kB	
.icons	/home/dd/.local/share/Trash/files	30.8 MB	
.ld.so	/home/dd/.var/app/org.mozilla.fir...	29.4 kB	
.local	/home/dd	32.4 MB	2022-12-21 08:09
.local	/home/dd/snap/snap-store/582	566.4 kB	2022-12-21 08:33
.local	/home/dd/snap/snap-store/638	566.4 kB	2022-12-21 08:33
.local	/home/dd/snap/snapd-desktop-in...	50 bytes	2022-12-21 08:09
.local	/home/dd/snap/snapd-desktop-in...	50 bytes	2022-12-21 08:09
.mozilla	/home/dd	0 bytes	2022-12-21 09:00
.mozilla	/home/dd/.var/app/org.mozilla.fir...	27.9 MB	2023-01-31 07:20
.ssh	/home/dd	0 bytes	2022-12-21 08:10

Search in Path ☐
Match Case ☐
Enable RegEx ☐
Show Statusbar ☑
Show Filter ☑
Show Search Button ☐

Figure 2: FSearch offers a clear-cut interface and a powerful search engine that returns results as you type.

You can extend the predefined filters for file types such as *Folders*, *Files*, *Pictures*, *Documents*, or *Audio*, with your own filters in the settings on the *Search* tab (Figure 5). However, the search can also be narrowed down even further in the search mask by entering `~/documents pdf`, for example. If you know the file name, `~/Documents Li` will find all documents with this combination of initial letters.

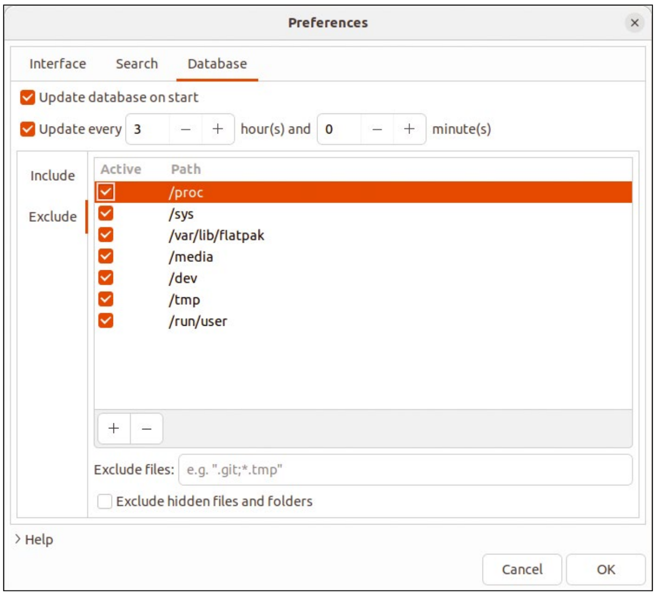
If you enable the *Search in path when query contains path separators* option in the *Search* tab, FSearch will recognize when you want to follow a



☑ Update database on start
☑ Update every 0 hour(s) and 15 minute(s)

Active	Path	One Filesystem
☑	/home/dd	☐
☑	/etc	☐
☑	/opt	☐
☑	/var	☐

Figure 3: When first launched, FSearch needs to fill the database with content. A click on *Add folders* lets you select the folders to be indexed; you can also exclude folders.



☑ Update database on start

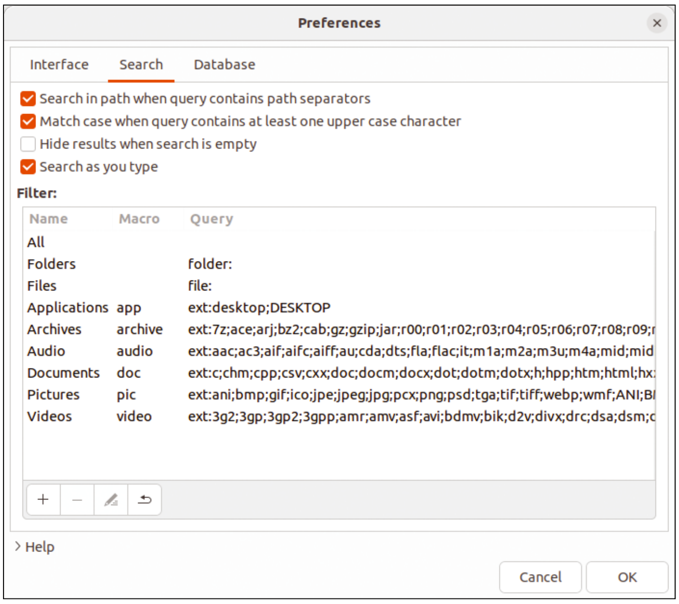
☑ Update every 3 hour(s) and 0 minute(s)

Active	Path
☑	/proc
☑	/sys
☑	/var/lib/flatpak
☑	/media
☑	/dev
☑	/tmp
☑	/run/user

Exclude files: e.g. ".git;*.tmp"

☐ Exclude hidden files and folders

Figure 4: In the settings, you can exclude entire directories or certain files from indexing (and thus from the search), if required.



☑ Search in path when query contains path separators

☑ Match case when query contains at least one upper case character

☐ Hide results when search is empty

☑ Search as you type

Filter:

Name	Macro	Query
All		
Folders		folder:
Files		file:
Applications	app	ext:desktop;DESKTOP
Archives	archive	ext:7z;ace;arj;bz2;cab;gz;gzip;jar;r00;r01;r02;r03;r04;r05;r06;r07;r08;r09;
Audio	audio	ext:aac;ac3;aif;aifc;aiff;au;cda;dts;fla;flac;it;m1a;m2a;m3u;m4a;mid;mid
Documents	doc	ext:c;chm;cpp;csv;cxx;doc;docm;docx;dot;dotm;dotx;h;hpp;htm;html;hx
Pictures	pic	ext:ani;bmp;gif;ico;jpe;jpeg;jpg;pcx;png;psd;tga;tif;tiff;webp;wmf;ANI;B
Videos	video	ext:3g2;3gp;3gp2;3gpp;amr;amv;asf;avi;bdmv;bik;d2v;divx;drc;dsa;dsm;

Figure 5: The *Search* tab in the Preferences dialog lets you define your own filters and associated file types for the search.

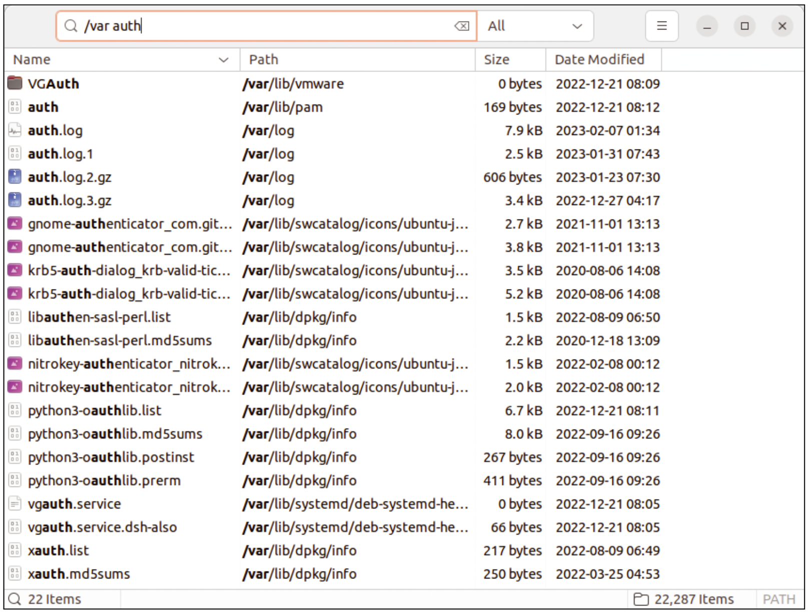
path. If you enter `home` in the search mask, the result shows all files with `home` in the name. By adding the path separator to the term (e.g., `home/`), the software performs a path search.

The *Behaviour after successfully opening a file* option (Figure 1) with *Minimize* or *Close* selected from the drop-down list proves useful if you only intend to browse for a file, open it from the context menu, and then process it elsewhere.

New Operators

The developers updated the search engine in FSearch 0.2. The syntax now offers many more possibilities than before. Operators and functions can be used to specify the search term or you can extend it using wildcards. In addition, the search evaluates regular expressions, based on the Perl Compatible Regular Expressions (PCRE) library.

Figure 6: This is where the *AND* operator comes into play. Because the spelling distinction is not activated, the match list also contains files with "AUTH" in their names.



/var auth

Name	Path	Size	Date Modified
VG**Auth**	**/var**/lib/vmware	0 bytes	2022-12-21 08:09
auth	**/var**/lib/pam	169 bytes	2022-12-21 08:12
auth.log	**/var**/log	7.9 kB	2023-02-07 01:34
auth.log.1	**/var**/log	2.5 kB	2023-01-31 07:43
auth.log.2.gz	**/var**/log	606 bytes	2023-01-23 07:30
auth.log.3.gz	**/var**/log	3.4 kB	2022-12-27 04:17
gnome-**auth**enticator_com.git...	**/var**/lib/swcatalog/icons/ubuntu-j...	2.7 kB	2021-11-01 13:13
gnome-**auth**enticator_com.git...	**/var**/lib/swcatalog/icons/ubuntu-j...	3.8 kB	2021-11-01 13:13
krb5-**auth**-dialog_krb-valid-tic...	**/var**/lib/swcatalog/icons/ubuntu-j...	3.5 kB	2020-08-06 14:08
krb5-**auth**-dialog_krb-valid-tic...	**/var**/lib/swcatalog/icons/ubuntu-j...	5.2 kB	2020-08-06 14:08
lib**auth**en-sasl-perl.list	**/var**/lib/dpkg/info	1.5 kB	2022-08-09 06:50
lib**auth**en-sasl-perl.md5sums	**/var**/lib/dpkg/info	2.2 kB	2020-12-18 13:09
nitrokey-**auth**enticator_nitrok...	**/var**/lib/swcatalog/icons/ubuntu-j...	1.5 kB	2022-02-08 00:12
nitrokey-**auth**enticator_nitrok...	**/var**/lib/swcatalog/icons/ubuntu-j...	2.0 kB	2022-02-08 00:12
python3-o**auth**lib.list	**/var**/lib/dpkg/info	6.7 kB	2022-12-21 08:11
python3-o**auth**lib.md5sums	**/var**/lib/dpkg/info	8.0 kB	2022-09-16 09:26
python3-o**auth**lib.postinst	**/var**/lib/dpkg/info	267 bytes	2022-09-16 09:26
python3-o**auth**lib.prerm	**/var**/lib/dpkg/info	411 bytes	2022-09-16 09:26
vg**auth**.service	**/var**/lib/systemd/deb-systemd-he...	0 bytes	2022-12-21 08:05
vg**auth**.service.dsh-also	**/var**/lib/systemd/deb-systemd-he...	66 bytes	2022-12-21 08:05
x**auth**.list	**/var**/lib/dpkg/info	217 bytes	2022-08-09 06:49
x**auth**.md5sums	**/var**/lib/dpkg/info	250 bytes	2022-03-25 04:53

22 Items
22,287 Items

Like previously, FSearch interprets a space in the search term as an *AND*. For example, if you are looking for all occurrences relevant to AUTH in the `/var` directory, type `/var auth`. If you don't select *Match case when query contains at least one upper case character* (Figure 5) in the *Search* tab, FSearch will find both "AUTH" and "auth" (Figure 6).

You can also use the new operators to specify match criteria such as size, modification date, folder depth, extension, content type, and much more. For example, to find any MP4 file larger than 1GB, you would type

```
ext:mp4 size:>1gb
```

The program also lets you create queries with the *OR* and *NOT* operators. For example, to find all JPEG and PNG files modified last month with "cat" or "dog" in their name, type:

```
(cat OR dog) ext:jpg;png dm:lastmonth
```

On GitHub, the project explains the extended syntax and provides a couple of examples [8].

Outlook

Boxdörfer provides users with a sneak peak of planned extensions and the versions in which they will be incorporated into FSearch on GitHub [9].

In the near future, he plans, among other things, to introduce live updates of the database and to use `inotify` to automatically detect changes to directories or files [10]. In addition, it will be possible to filter for certain file types during indexing according to properties such as file size or last change.

FSearch 0.5 is expected to support bookmarks as well as a search history, and version 0.6 is expected to show thumbnails for supported file types. An implementation for the command line is also on the roadmap. That would make it easier to write extensions for KDE's KRunner, for example.

Conclusions

There are many ways to search the filesystem on Linux depending on your requirements. FSearch offers most of the functions you would expect from a comfortable search application. Above all, it is fast and starts searching as soon as you enter the term. The results can be filtered and sorted in many ways. It is hard to understand why FSearch is still missing from the repositories of many distributions, because in my humble opinion, the search engine is extremely capable. If you want to report bugs or need support, you can write to Christian Boxdörfer. You also can support his work with a small donation via PayPal. ■■■

Info

[1] FSearch: *https://github.com/cboxdoerfer/fsearch*

[2] Everything: *https://en.wikipedia.org/wiki/Everything_(software)*

[3] Download for Debian and openSUSE: *https://software.opensuse.org//download.html?project=home%3Acboxdoerfer&package=fsearch#manualopenSUSE*

[4] Download for Fedora and RHEL: *https://copr.fedorainfracloud.org/coprs/cboxdoerfer/fsearch/*

[5] Launchpad repository for Ubuntu: *https://launchpad.net/~christian-boxdoerfer/+archive/ubuntu/fsearch-stable*

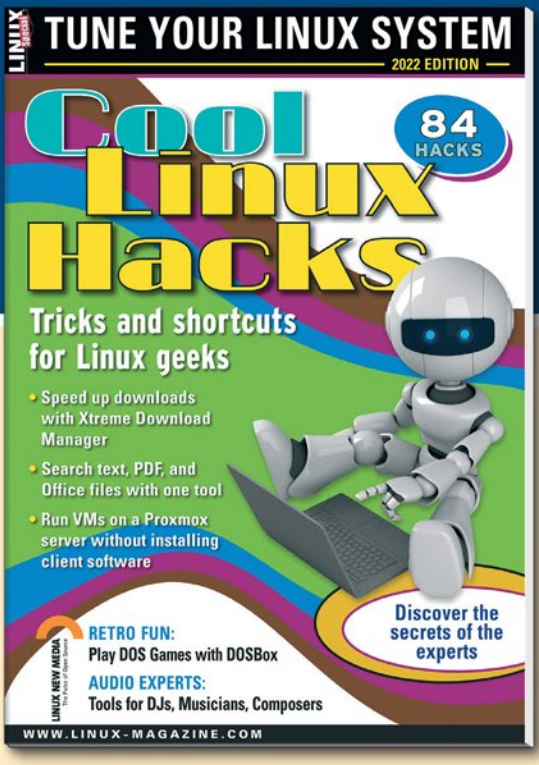
[6] FSearch Flatpak: *https://flathub.org/apps/details/io.github.cboxdoerfer.FSearch*

[7] Compile FSearch yourself: *https://github.com/cboxdoerfer/fsearch/wiki/Build-instructions*

[8] New syntax: *https://github.com/cboxdoerfer/fsearch/wiki/search-syntax*

[9] FSearch roadmap: *https://github.com/cboxdoerfer/fsearch/wiki/Roadmap*

[10] inotify: *https://en.wikipedia.org/wiki/inotify*


LINUX Special
TUNE YOUR LINUX SYSTEM
2022 EDITION
Cool Linux Hacks
84 HACKS
Tricks and shortcuts for Linux geeks
• Speed up downloads with Xtreme Download Manager
• Search text, PDF, and Office files with one tool
• Run VMs on a Proxmox server without installing client software
Discover the secrets of the experts
RETRO FUN: Play DOS Games with DOSBox
AUDIO EXPERTS: Tools for DJs, Musicians, Composers
LINUX NEW MEDIA
WWW.LINUX-MAGAZINE.COM

SHOP THE SHOP
shop.linuxnewmedia.com

GET PRODUCTIVE WITH
COOL LINUX HACKS

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

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

ORDER ONLINE:
shop.linuxnewmedia.com/specials

FOSSPicks

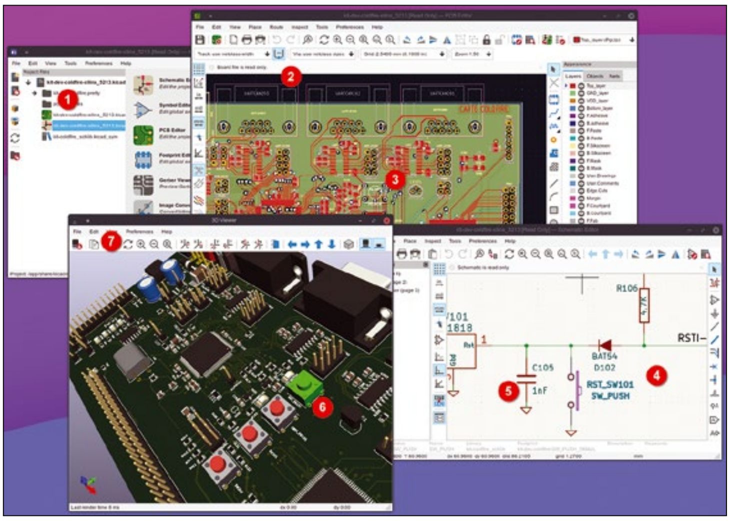
Sparkling gems and new releases from the world of Free and Open Source Software

Graham has finally taken delivery of a couple of RISC-V development boards for Linux experimentation. Expect plenty of cross-build tools in future issues! BY GRAHAM MORRISON

Electronics designer

KiCad 7

KiCad is a huge cross-platform application that's used by thousands of designers, from hobbyists to professional engineers. It's remarkable that it remains open source and free, and it is capable of far more than we could describe briefly here. It encapsulates all you need to create and test circuit schematics, a process known as schematic capture. A huge symbol library lets you drag and drop almost any component you could ever require into your circuit, from 50 different audio amplifiers to 30 different video encoders and decoders. If you do discover something missing, you can generate and drop your own creations into the library to create your own symbols. Circuit simulation makes the application feel a little like programming, testing, and debugging within an IDE before you arrange circuits into a layout that can be built or printed, or even viewed in 3D – with a 3D mouse. Finally, a Bill of Materials (BoM) can be generated to include everything you've used so you can easily create an order to deliver everything you need alongside your PCB. KiCad doesn't quite negate the need to understand the first principles of electronics, but it goes a long way to make it more accessible and as approachable for anyone with an interest. It's even powerful enough to stay with you when your skills eventually take you to a professional level. There are only a handful of open source applications that can say the same.



1. Demos: KiCad includes several excellent demo projects to help you get started. **2. PCB Editor:** This is where your schematics are laid out onto a PCB, and from where they can be exported. **3. Component linking:** Clicking on an item in any view will highlight it on any other. **4. Schematics:** Create a circuit and even run a simulator to test its validity with plugins to calculate resistance, current, and capacitance. **5. Component library:** Add items from a huge list of recognized components or add your own. **6. 3D View:** See how your PCB will look, and even manipulate the view with a 3D mouse. **7. Export:** Send your designs to be printed and create a BoM of the components you'll need to solder in.

KiCad has now been in development for over 30 years. It's a project we often mention in passing, most recently when we covered a tool called Pinion that used KiCad 6 in the background to create wonderful interactive electrical circuits and circuit pinout diagrams. With this major release of KiCad 7, we have the perfect excuse to look at this all-conquering electrical circuit design tool in more detail. It's become a hugely influential project, helping to kickstart and maintain the burgeoning DIY circuit design and printing boom. And thanks to its new annual release policy, version 7 comes almost exactly a year after the previous major release. It includes fixes for over 1,200 issues alongside some significant new features. These new features include custom fonts that can be used in schematic, PCB, and worksheet editors, along with the ability to create text boxes, which are likely an ideal location to test out the new fonts. There's also support for 3D mice, which can help you navigate the metaverse of your own constructions without turning them into physical objects. When you do want a physical object, there's a new command-line interface to help automate schematics export and file conversion, and there are new rectangle and circle primitives for drawing pretty patterns all over your circuits. To help with simulation mode, some of the configuration options for a model now have their own graphical properties rather than requiring you to enter the configuration text manually. This helps expose what is often an overlooked and very powerful part of what KiCad offers: its ability to simulate components in circuits. The best feature, however, is being able to draw a wired connection at 45 degrees. You could previously draw at right angles, or across whatever angle worked for a start and end point, but fixing an angle to 45 degrees is very pleasing and works well when optimizing circuits for a specific space.

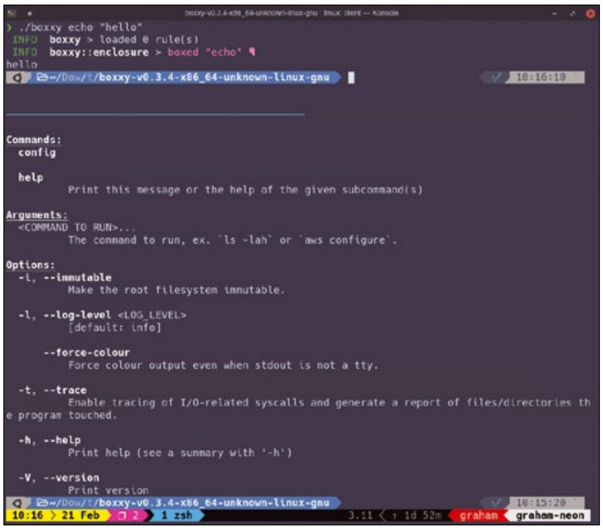
Project Website
https://www.kicad.org

Isolation tool

boxxy

Linux has become so good at containers, virtual machines, and isolated package formats that you're probably running more instances of Linux in containers than you are natively. But you often don't need a full-blown container just to keep a few things separated from your day-to-day system. For those occasions, boxxy is the perfect solution. Boxxy is a clever, low-resource tool that corrals directories and files into a location you specify, and optionally, only under the circumstances you write into its configuration file. Thanks to a Linux kernel feature called namespaces, boxxy accomplishes all of this without using symbolic links (symlinks), which are typically used all the time for ad hoc links between locations. But symlinks can make it difficult to understand where your files are, whether something has truly been backed up, and just how many chained links are sitting on top of one another. Boxxy is better.

Boxxy compartmentalizes all of this using a separate namespace for each configuration you create. A configuration can be as simple as a source target (such as `~/.tmux.conf`) and a destination rewrite (`~/.config/tmux/tmux.conf`) with a mode setting of `file` for simple file redirection, or `directory`, which is the default. A `context` can optionally be set so that a rule only applies to one location, so you can build a set of rules for different locations with the same configuration. With that out of the way, you need to launch or alias the command you want run with `boxxy`, followed by the configuration name. Each command will think it is accessing resources from whatever path their developers have hard-coded into their executable while you're secretly redirecting input and output to a location of your choice. It's a brilliant way to clean up wayward configuration files, or a home directory cluttered with dozens of files from utilities that have no regard for Marie Kondo's tidying up philisophy.



```
❯ ./boxxy echo "hello"
 INFO  boxxy > loaded 0 rule(s)
 INFO  boxxy::enclosure > boxed "echo" ❚
hello
 ~/Dow/t/boxxy-v0.3.4-x86_64-unknown-linux-gnu  ▌                    ✓  10:16:10

Commands:
  config

  help
          Print this message or the help of the given subcommand(s)

Arguments:
  <COMMAND TO RUN>...
          The command to run, ex. `ls -lah` or `aws configure`.

Options:
  -i, --immutable
          Make the root filesystem immutable.

  -l, --log-level <LOG_LEVEL>
          [default: info]

      --force-colour
          Force colour output even when stdout is not a tty.

  -t, --trace
          Enable tracing of I/O-related syscalls and generate a report of files/directories th
e program touched.

  -h, --help
          Print help (see a summary with '-h')

  -V, --version
          Print version
 ~/Dow/t/boxxy-v0.3.4-x86_64-unknown-linux-gnu                          ✓  10:15:20
10:16 ❯ 21 Feb ❯ 2 ❯ 1 zsh ❯                      3.11 ❮ ↑ 1d 52m ❮ graham ❮ graham-neon
```

Boxxy takes advantage of low-level kernel features to create an accessible and useful tool to isolate apps you want to run.

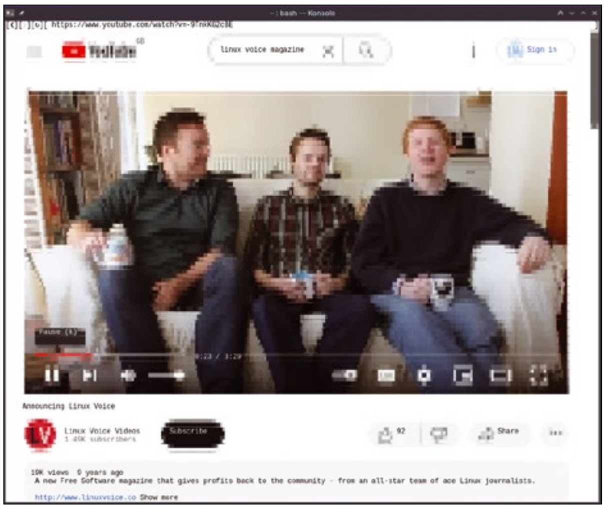
Project Website
https://github.com/queer/boxxy

Terminal browser

Carbonyl

We've looked at several terminal-based web browsers over the years, and most can trace their origins back to when a terminal was a necessity rather than a choice, because desktops didn't exist. This meant their focus was on getting access to a website and allowing you to navigate without too much difficulty, rather than emulating a nonexistent desktop web browser. Those apps are still brilliant if you need to download a graphics driver in an emergency or search the web for instructions on resetting a broken Wayland session. But for accessing the modern JavaScript-ridden interactive web, they're less than ideal. Carbonyl, however, is a terminal web browser that takes the terminal experience to a whole new level. Its output is a mixture of desktop FFmpeg-to-ASCII conversion (our guess) with perfect text rendering for the parts of a web page you need to read. And it's amazing.

Installation is easy via npm, and there's a Docker image, too, for simple cloud deployments. The compatibility and rendering come from a hidden Chromium-based renderer, but the results are remarkable. Video previews play perfectly, drop-down text is rendered natively, and websites are almost as easy to navigate as they are on the desktop, especially if you're already familiar with them. Everything just works, from SSL-dependent email to interactive 3D through WebGL. It even runs through SSH at 60 frames per second and takes very few resources when sitting idle. Video and WebGL do challenge your CPU, however, and there are too few keyboard bindings to remove the need for a mouse, which seems incongruent with the command line. It's also missing an advert blocker. But these are all small issues in something that seems to single-handedly solve a decades-old problem and even make command-line web browsing preferable to the desktop for coolness and readability factors.



A new Free Software magazine that gives profits back to the community - from an all-star team of ace Linux journalists.

Carbonyl renders web pages so brilliantly into your terminal it looks more like a low-resolution screen grab than pixel art.

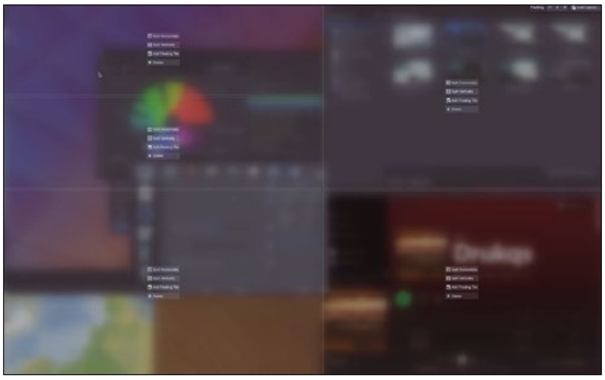
Project Website
https://github.com/fathyb/carbonyl

Window tiling

KDE Plasma 5.27

KDE Plasma 5.27 is a significant milestone, not just because it contains so many new features and fixes, but because it's the final release before the migration to Qt 6 and Plasma 6. As with several major open source projects, including Firefox, Krita, and Blender, a rapid cadence makes it difficult to track anything but major new features, and that's certainly the case with Plasma 5.27. Nestled alongside the new Discover features and the Guided setup is "native window tiling." This is something we've wanted to see for years, and we've covered an almost endless supply of scripts and add-ons that bring native-like tiling to Plasma. Some have been more successful than others, with the austere kwin-tiling being the tool to beat thanks to its powerful i3 keyboard shortcuts. The KDE team, however, have supplanted kwin-tiling with their first tentative steps into tiling, while at the same time massively broadening its appeal to the many of us who can't remember a single keyboard shortcut.

The new tiling feature is difficult to find, but it's triggered by enabling the *Tiling Editor* in the *Desktop Effects* pane. With this done, you can now press Meta+T (by default) to open the editor and create some zones for your windows to lock into. The editor is beautifully designed, showing a small pop-up that prompts you to split an area horizontally, vertically, or to enable floating windows. An area can be further subdivided into as many horizontal and vertical splits as you need, and area boundaries can be dragged into whatever column or row for any size you need. You can also load a predefined layout. Pressing the shortcut again will return you to the desktop, and if you press Shift when dragging a window, the tiles you created are highlighted. Releasing a window into one of these will lock it in place, but you can still resize tiles and borders even when the screen is full, and all windows in the same row or column will also resize. The only features missing are keyboard controls and the ability to save a new layout, but we're sure these will come with an update.



There are no keyboard shortcuts yet for native Plasma tiling, but Alt+dragging a window makes it almost as intuitive when relocating windows.

Project Website

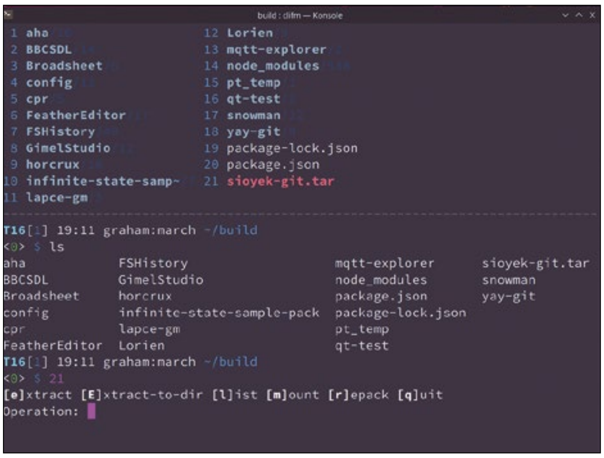
https://kde.org/et/announcements/plasma/5/5.27.0/

3D design

CliFM

Despite the terminal being the place for copying, renaming, deleting, and organizing files, text-based file managers have always been popular. They can help remove some of the jeopardy in using commands to do important things to your data, especially when multiple files or directories are involved, and a file manager is obviously more visual. This can make it easier to navigate and explore filesystems you may not be familiar with. Miguel de Icaza famously started his career by writing Midnight Commander for Linux in 1994, before starting the Gnome project in 1997, and his venerable `mc` command is still the one many of us resort to when we want to visualize files and directories on the command line. But as entrenched in our collective history as Midnight Commander is, there's always room for a command-line tool that tries to take a different approach.

The big difference with CliFM is that it doesn't use TUI to build a text-based representation of a graphical interface. As a result, its interface feels a lot more authentic and true to traditional Unix tools and philosophy – and it does this without forgoing either intuitiveness or capabilities. The main view presents a numbered list for the directories and files in the current location. Selecting a number and pressing Return will either enter a directory or present a context-sensitive menu for files. If you select an archive, for example, you can choose between extracting, listing, mounting, and repacking the files it contains, and it's the same for lots of other supported file types. You can also issue commands, such as `m`, `c`, or `r` to move, copy, or delete a numbered file or directory respectively. Multiple entries can be selected interactively, or with the `s` command, and it all feels like playing an ancient interactive fiction game with your filesystem. Which isn't a bad thing!



```
 1 aha                    12 Lorien
 2 BBCSDL                 13 mqtt-explorer
 3 Broadsheet             14 node_modules
 4 config                 15 pt_temp
 5 cpr                    16 qt-test
 6 FeatherEditor          17 snowman
 7 FSHistory              18 yay-git
 8 GimelStudio            19 package-lock.json
 9 horcrux                20 package.json
10 infinite-state-samp~   21 sioyek-git.tar
11 lapce-gm

T16[1] 19:11 graham:march ~/build
<0> $ ls
aha            FSHistory                  mqtt-explorer      sioyek-git.tar
BBCSDL         GimelStudio                node_modules       snowman
Broadsheet     horcrux                    package.json       yay-git
config         infinite-state-sample-pack package-lock.json
cpr            lapce-gm                   pt_temp
FeatherEditor  Lorien                     qt-test
T16[1] 19:11 graham:march ~/build
<0> $ 21
[e]xtract [E]xtract-to-dir [l]ist [m]ount [r]epack [q]uit
Operation: █
```

The prompt in CliFM gives you access to its file manager features, but also passes commands through to your shell for the best of both worlds.

Project Website

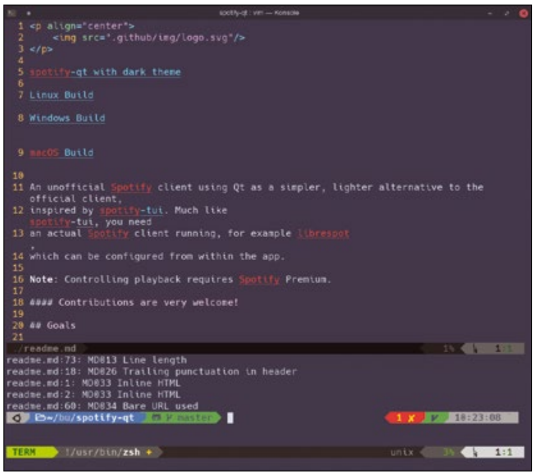
https://github.com/leo-arch/clifm

Markdown linting

markdownlint

Using the Markdown syntax for writing text documentation has become almost the default for many projects and writers. It's easy to learn and understand, and it treads lightly on the readability of the raw document when you don't have a Markdown preview. You don't need to understand Markdown to know which parts are titles, lists, or links to other pages, and Markdown can be ingested by many services and turned into beautifully rendered web pages or structured documentation. It's GitHub's default for good reason. But there also are a couple of big problems with Markdown, and they're related: There's no agreed global Markdown standard and there are many tweaked derivatives of definitions that become standards in their own domain. Even GitHub supports two, kramdown and its own GitHub Flavored Markdown (GFM).

Despite this confusion and complexity, and its use in development projects with their own CI systems, there are very few tools that can check the validity of Markdown according to your own whims and requirements. However, markdownlint is one of these few, and it's also the most adaptable for multiple standards of Markdown. It's a command (typed as `mdl`) that takes a Markdown file as an argument and reports back on any problems with the syntax used within the file.



```
<p align="center">
    <img src=".github/img/logo.svg"/>
</p>

spotify-qt with dark theme

Linux Build

Windows Build


macOS Build


An unofficial Spotify client using Qt as a simpler, lighter alternative to the
official client,
inspired by spotify-tui. Much like
spotify-tui, you need
an actual Spotify client running, for example librespot
,
which can be configured from within the app.

Note: Controlling playback requires Spotify Premium.

#### Contributions are very welcome!

## Goals

./readme.md                                                    1%   1:1
readme.md:73: MD013 Line length
readme.md:18: MD026 Trailing punctuation in header
readme.md:1: MD033 Inline HTML
readme.md:2: MD033 Inline HTML
readme.md:60: MD034 Bare URL used
 ~/bu/spotify-qt   master                              1 ✗   10:23:08

TERM  !/usr/bin/zsh +                                  unix  3%   1:1
```

If you ever work with text-based Markdown files, markdownlint helps you make sure you've got the syntax correct.

Each problem is reported with a line number, error code, and text description, such as "Inline HTML" or "Lists should be surrounded by blank lines." There are 47 such rules, from correctly incrementing headings to ending a file with a single newline character, and they comprehensively cover the majority of issues you're likely to encounter. Broken rules are called style issues by markdownlint and, like Markdown itself, styles can be added or created to match your preferred flavor of Markdown. The syntax for this is also straightforward, such as `rule 'MD013', :line_length => 500` to test for lines longer than 500 characters. All of this is ideal for integration with a CI system that can check every commit, but it's equally useful on the command line and a great solution if you ever do any work with Markdown.

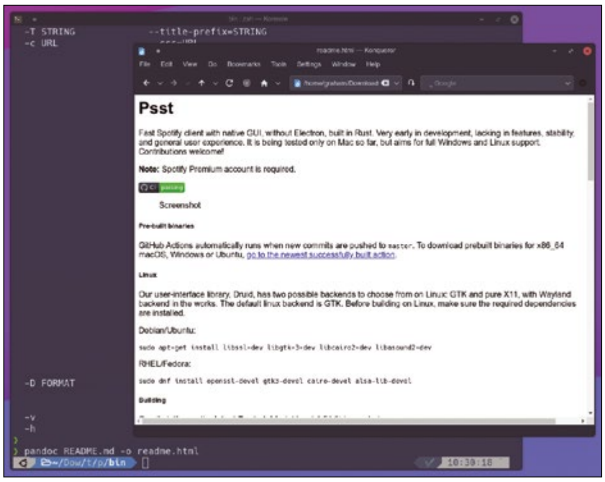
Project Website
https://github.com/markdownlint/markdownlint

Text processor

pandoc 3.1

While we're in the realm of text processing, there's one tool above all others that writers turn to when they have to convert documentation in one format to documentation in another. That's pandoc. Pandoc has been around since 2006 and has just celebrated the major milestone release of 3.0, formalizing more than five years of development work. Pandoc is the text equivalent to FFmpeg, capable not just of converting a huge number of input formats to a huge number of output formats, but also transforming the text on its way through its various processors. It can convert between Markdown formats, HTML, LaTeX and DOCX, and output to PDF while transforming tables, definition lists, footnotes, and even mathematical and R notation. We used it extensively when converting magazine articles in XML to ePub.

What makes pandoc isn't so much the input and output formats it supports, but the way you can finely control how the conversion operates, often iterating over finer points in substantial sets of documentation that might otherwise take considerable effort to convert any other way. This power is thanks to what pandoc calls filters, a program module that can modify the text as it passes through. Many are included, and you can easily write your own using JSON or Lua. But the big change for this release is how pandoc runs, because you now have a choice in how you access its functionality, either using a traditional command-line tool, as a server, or via a library. This makes it easier to incorporate into whatever system you use to translate your docs. It can even be used as an interpreter, like Python, so you can quickly check command-processing effects without editing and running a script. This is a great way to get started if you've ever been put off by pandoc's complexity and definitely worth the time investment.



```
  -T STRING            --title-prefix=STRING
  -c URL

  -D FORMAT

  -v
  -h
❯
❯ pandoc README.md -o readme.html
 ~/Dow/t/p/bin                                              10:30:18
```

Psst

Fast Spotify client with native GUI, without Electron, built in Rust. Very early in development, lacking in features, stability and general user experience. It is being tested only on Mac so far, but aims for full Windows and Linux support. Contributions welcome!

Note: Spotify Premium account is required.

Screenshot

Pre-built binaries

GitHub Actions automatically runs when new commits are pushed to `master`. To download prebuilt binaries for x86_64 macOS, Windows or Ubuntu, go to the newest successfully built action.

Linux

Our user-interface library, Druid, has two possible backends to choose from on Linux: GTK and pure X11, with Wayland backend in the works. The default linux backend is GTK. Before building on Linux, make sure the required dependencies are installed.

Debian/Ubuntu:

```
sudo apt-get install libssl-dev libgtk-3-dev libcairo2-dev libasound2-dev
```

RHEL/Fedora:

```
sudo dnf install openssl-devel gtk3-devel cairo-devel alsa-lib-devel
```

Building

If you have a set of documentation in one format and you need it in another, pandoc can nearly always get the job done.

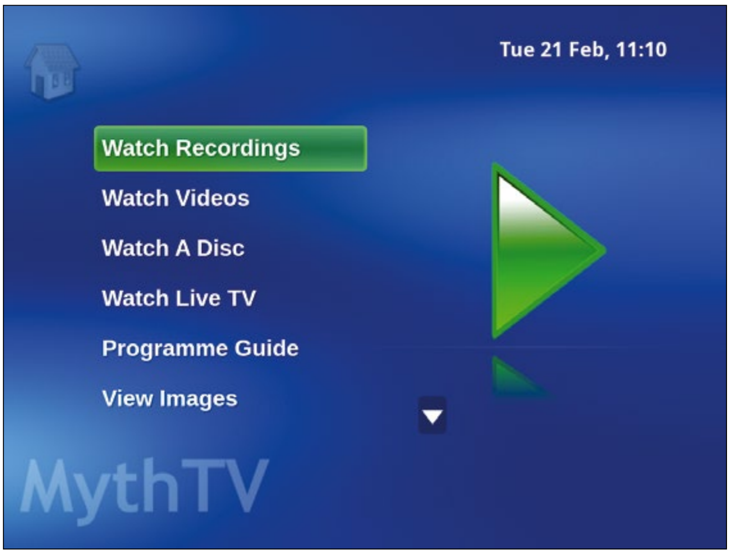
Project Website
https://pandoc.org/

TV recorder

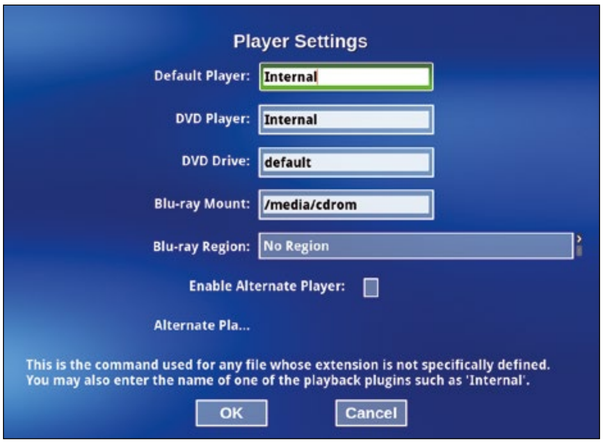
MythTV 33.1

MythTV is one of those few remarkable projects that have been around for so long they've become part of the open source canon (a little like KiCad!). It was first released over 20 years ago, and I personally used it as a daily television-recording solution, video player, and music center for five years, between 2004 and 2009, until my interest in live broadcasts waned and the Internet took over. Back then, I was using a video grabber to digitize the analog output from a satellite receiver in coordination with an LIRC infrared beamer to send channel signals. A successfully recorded program required a delicate balance between show timings carefully scraped from a website, the infrared beamer successfully sending a channel change code at the right time, the receiver interpreting that code correctly, and the grabber starting correctly with enough back end storage for whatever was being recorded. Remarkably, after a steep learning curve and considerable configuration effort, it worked faultlessly for years, running on a big-box PC hidden inside a clothes chest.

Since then, the DVB standard for digital television took over those analog transmissions, and the Internet replaced everything with streaming. We now have terabytes of solid-state storage, the Raspberry Pi, and few devices accepting infrared remote control. Projects such as Tvheadend, the DVR digital video recorder, Kodi, and OpenELEC make my old MythTV setup feel like a relic. But MythTV survives, and even thrives, because it still does things all those other tools can't do – like run forever. Its front end is still perhaps the closest match to the graphical front ends you get with a commercial cable or satellite service, and can be installed as a module in Kodi. The back end controls your hardware, and remains bulletproof after a not-insubstantial setup, with a separate database back end for channel and program guide data. Each element can be run on a separate device, with many people running the front end on their screens, for example, with a back end tucked away to manage recording schedules and video files.



MythTV is a brilliant digital television recording platform, with powerful ad-removal capabilities and a standalone front end that integrates with Kodi.



Default Player: Internal
DVD Player: Internal
DVD Drive: default
Blu-ray Mount: /media/cdrom
Blu-ray Region: No Region
Enable Alternate Player: ☐
Alternate Pla...

This is the command used for any file whose extension is not specifically defined.
You may also enter the name of one of the playback plugins such as 'Internal'.

Another benefit from MythTV's long history is that it has mature MediaWiki-based documentation that the community keeps up-to-date.

The plugin system remains one of MythTV's best features, letting you fine-tune your system to your exact needs, including gaming, news, weather, web browsing, home security, image gallery, and music plugins. MythWeb is worth the entry price alone, because it implements what remains the best program guide and recording front end in open source, plus it enables convenient access to your movies and music and other plugins, all through a web page. And the latest releases keep adding new features, along with a longer term goal of modernizing and restructuring the code. Recent updates have added a new waveform view in the music player, new versions of FFmpeg, a new start-up page, better DVD and Blu-ray playback, and hundreds of individual fixes. Best of all, and thanks to the project still being actively developed, you can be sure the time you invest getting everything set up will be worth it. You'll probably be able to run MythTV for another 20 years. It will quietly record whatever you're searching for, removing any advertising, and converting the video into whatever format is most convenient for you. It may be unglamorous, but it's still the best way of taming the wild frontiers of digital television.

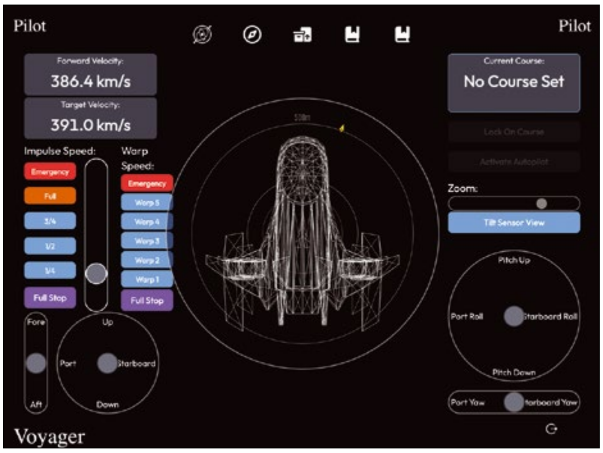
Project Website
https://www.mythtv.org

Bridge simulator

Thorium Nova

When you read that this game is a "bridge simulator," your first thought might be that it's a recreation of the tricky card game. It's not, nor is it a game like Bridge Constructor, where you're tasked with traversing some kind of chasm or river. This bridge simulator takes its inspiration from Star Trek, rather than structural engineering, and the bridge in question is the command center for a spaceship. Thorium Nova is a sequel to Thorium, which was the developers' first attempt at a space-base bridge simulator. True to a real command bridge, there are different stations for controlling different aspects of the ship, including captain, navigation, tactical planning, engineering, and communications. Each role has its own interface, its own skill set, and its own set of challenges. Most importantly, this is a multiplayer game, with a different player taking control of a different station, and everyone needs to work cooperatively to make the ship work.

The game itself runs on a server and other players connect through a web browser. By default, all of this is automatically set up when you first run the game. One of the main aims of the game is to be as accessible and as beginner-friendly as possible, and being browser-based means almost anyone can join. When players do join, they can choose their own station, or be assigned one by the flight director controlling the server part of the game. Due to the alpha state of the game, there are currently no missions, but it's the flight director and the team's task to set their own objectives. The pilot and navigation station is the most fully developed, letting you navigate to different space bodies and align your ship to targets. You can also move inventory through the cargo holds, but the game is still a work in progress, and this is an ideal time to get involved if this is something you feel you'd enjoy.



While much of the narrative structure in Thorium Nova has yet to be implemented, the basic game engine is fully functional.

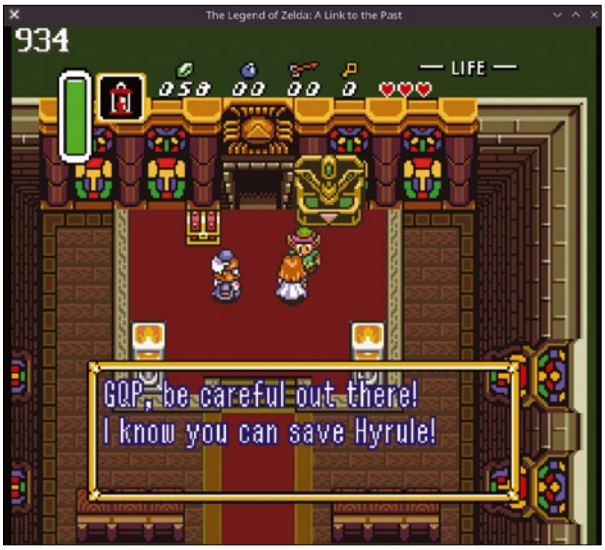
Project Website
https://thoriumsim.com

Reverse-engineered Zelda

zelda3

The Legend of Zelda: A Link to the Past is one of the best games ever made. It was originally released by Nintendo for its Super Nintendo/Super Famicon in 1991, and it's still available for the latest Switch console. The game was one of the first to incorporate a wonderful story, challenging puzzles, and intuitive combat mechanics within an immersive 2D world that's been often copied but never surpassed. It's a game whose intellectual property is obviously closely guarded by Nintendo, but it's also a game that contains the indelible DNA of success, and the nostalgic hopes and dreams of millennials everywhere. Which is why it's so significant that the zelda3 project has reverse-engineered the entire game into 80 thousand lines of C code, without infringing on any of the original game assets.

You will need the original game to retrieve the original assets, and these are needed if you want to build the game into a playable version of the original. The process is simple and quick, however, with included Python scripts used to extract the tables and compile them into a format the project can use when building itself. The entire build takes only a few minutes on modern hardware and the end result is A Link to the Past running on modern hardware. Even more impressively, the new version can include pixel shaders, widescreen ratios, a higher quality world map, better audio, and an extra item to hold with the *X* button. There's also snapshot saving and loading, and cheat shortcuts for unlimited health/magic and rupees, bombs, and arrows. It's a fun way to replay an old game, but it's the source code that's most important. This contains a direct, modern interpretation of the original game engine, tested frame-by-frame and state-by-state against the original.



GQP, be careful out there!
I know you can save Hyrule!

Who would have thought that decades later Zelda would be reverse-engineered and rebuilt to run at 800 FPS on a 64-bit ARM CPU?

Project Website
https://github.com/snesrev/zelda3

DuckDuckGo from the Terminal

ddgr Search

Since 2008, DuckDuckGo has been making waves as an efficient and much more private search engine alternative to Google. The unaffiliated command-line tool ddgr is designed to make running DuckDuckGo searches from the terminal a breeze.

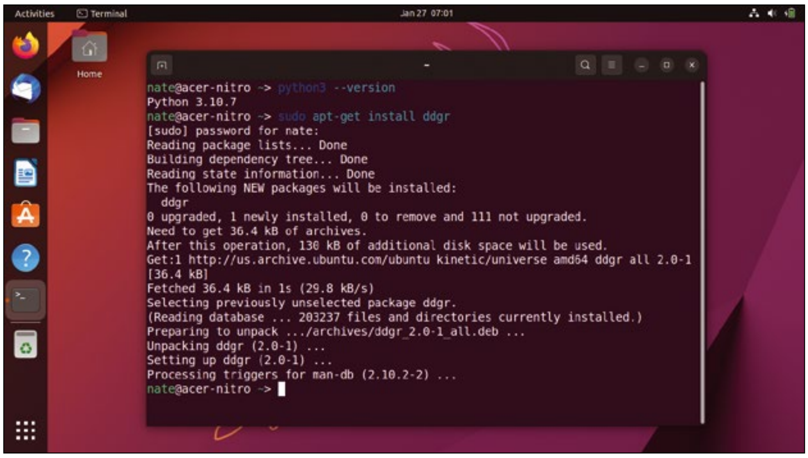
BY NATE DRAKE

Linux users do have reasons to be grateful to the almighty search engine Google. After all, Google gave the world the (mostly) open source Android operating system [1]. Still, many phones and devices come preinstalled with Google's app – a firm whose bottom line is largely based on serving you more and more targeted ads. This isn't good for your privacy, and as recently as late December 2022, the search engine was fined around $57 million in France due to its handling of customers' data [2].

As Linux users, we're used to finding workarounds, so in theory it's not much trouble to "de-Google" your life. In practice, this is very difficult given Google's market dominance. DuckDuckGo was originally created by Gabriel Weinberg to address these kind of privacy worries. Out of the box, the search engine does not serve you up personalized ads, nor does it rely on content farms to display results. The results come from the search engine's own crawler as well as partnerships it has with other search sites such as Bing and Yahoo! Search.

Unlike Google, DuckDuckGo doesn't use any hidden trackers. There are adverts, but these are personalized for you. It doesn't place you in a "filter bubble" by listing search results based on your location, nor does it try to trap you in an elaborate "ecosystem" of apps for mapping, email, travel, and so on [3].

Figure 1: Make sure you have at least Python 3.7 before installing ddgr.



```
nate@acer-nitro -> python3 --version
Python 3.10.7
nate@acer-nitro -> sudo apt-get install ddgr
[sudo] password for nate:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  ddgr
0 upgraded, 1 newly installed, 0 to remove and 111 not upgraded.
Need to get 36.4 kB of archives.
After this operation, 130 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu kinetic/universe amd64 ddgr all 2.0-1
[36.4 kB]
Fetched 36.4 kB in 1s (29.8 kB/s)
Selecting previously unselected package ddgr.
(Reading database ... 203237 files and directories currently installed.)
Preparing to unpack .../archives/ddgr_2.0-1_all.deb ...
Unpacking ddgr (2.0-1) ...
Setting up ddgr (2.0-1) ...
Processing triggers for man-db (2.10.2-2) ...
nate@acer-nitro -> 
```

You may wonder then, given the clear privacy advantages, why everyone isn't using this challenger search engine. The short answer is that many people do: DuckDuckGo has been adopted as the default search engine in Trisquel GNU/Linux and Linux Mint, as well as the Midori and Tor web browsers. Most major web browsers such as Firefox also support it out of the box, allowing users to switch easily or allow installation via a simple extension [4].

Still, no matter how well a browser is locked down, there are still privacy issues. Even running a simple search can expose you to adverts (personalized or not) and "super cookies" (which can't be removed from your device), and there's always a risk of your search results being stored on your device, even if you use "incognito mode." As Linux users, we're at home with the command line, so it's not surprising that developer jarun created ddgr, which allows you to segue around all these privacy concerns by running DuckDuckGo searches from your terminal. If the name sounds familiar, you may remember that jarun is also the creator of googler, which operates on similar lines but with the Google search engine [5].

Although jarun has no official connection with DuckDuckGo, the utility is still one privacy lovers will want to use: Not only is it quick and easy, but it doesn't record any data about searches. Tracking is disabled by default. You can turn off user-agent strings and can even connect via an HTTPS proxy, which means you can even run searches over the Tor network. The interface is very well laid out and you can tidy it up even further using ddgr's command-line parameters. There's even support for text-based web browsers.

Installing ddgr

Assuming that we've piqued your interest, the first step is to download and run ddgr (Figure 1). The utility is available in most Linux repositories already: For instance, on Ubuntu-based systems, it's just a case of running `sudo apt-get install ddgr`. There's also a version in the Snap Store, which you can install with `snap install ddgr`. Because ddgr is written in Python, you can also use the pip3 installer. Just run `sudo pip3 install ddgr`.

If these options aren't available or you're very privacy conscious, you can also clone the Git repository and run it as a standalone executable with:

```
git clone https://github.com/jarun/ddgr
```

Use `cd` to switch to the ddgr directory and then run `./ddgr`.

You'll need to run at least Python 3.7 to use ddgr. You can check your Python version in your terminal by running `python3 --version`. If you want to be able to copy website URLs from ddgr to the clipboard, you'll also need to have installed either xsel, xclip, or termux-clipboard-set. If you're running a Debian- or Ubuntu-based flavor [6] of Linux, it should already have xsel installed.

The utility will open search results in your default browser unless you tell it otherwise, so make sure you have this set up correctly in Linux before continuing.

Taking Flight

Once ddgr is installed, simply run your first search with `ddgr KEYWORD`, for instance, type `ddgr lighthouses` to view all DuckDuckGo search results for the term "lighthouses." By default, the utility will list the first 10 results from DuckDuckGo.

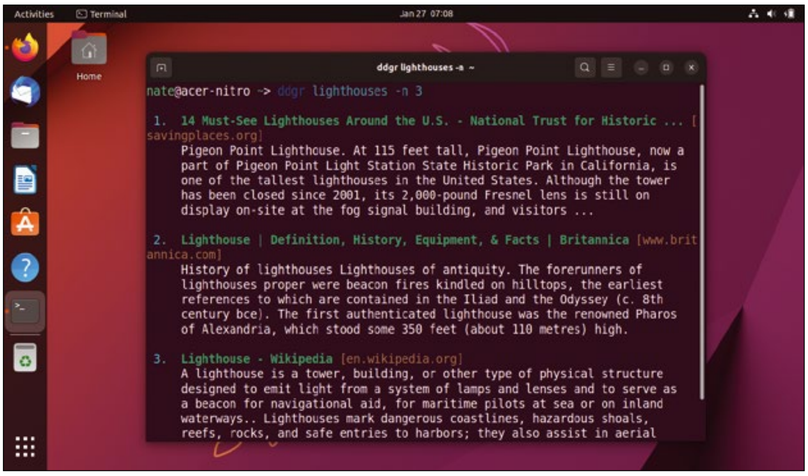
You'll next be prompted to type an omniprompt key. This takes a little getting used to at first but is actually designed very logically. Type `?` and press Enter the first time you run a search to see all the available options.

If you simply want to open a search result in your browser, just type the index number (e.g., `3`) and hit Return. This will open the page.

If you haven't found exactly what you're looking for, you only need to enter `n` once a search is complete to see the next x-number of search results. Use `p` to return to the previous list of results or `f` to return to the very start.

Use `x` to toggle displaying URLs within search results. You can also copy URLs to the clipboard using c+Space+the index number (e.g., `c 3`) provided you have a compatible program (see above).

If you've run one search and now want to run another, just enter the keywords at the omniprompt and hit Enter. If you add the flag `d` before running another search, ddgr will retain the same search parameters as for the original. This can be very helpful if you want to fine-tune your search results.



```
nate@acer-nitro -> ddgr lighthouses -n 3

 1.  14 Must-See Lighthouses Around the U.S. - National Trust for Historic ... [
savingplaces.org]
     Pigeon Point Lighthouse. At 115 feet tall, Pigeon Point Lighthouse, now a
     part of Pigeon Point Light Station State Historic Park in California, is
     one of the tallest lighthouses in the United States. Although the tower
     has been closed since 2001, its 2,000-pound Fresnel lens is still on
     display on-site at the fog signal building, and visitors ...

 2.  Lighthouse | Definition, History, Equipment, & Facts | Britannica [www.brit
annica.com]
     History of lighthouses Lighthouses of antiquity. The forerunners of
     lighthouses proper were beacon fires kindled on hilltops, the earliest
     references to which are contained in the Iliad and the Odyssey (c. 8th
     century bce). The first authenticated lighthouse was the renowned Pharos
     of Alexandria, which stood some 350 feet (about 110 metres) high.

 3.  Lighthouse - Wikipedia [en.wikipedia.org]
     A lighthouse is a tower, building, or other type of physical structure
     designed to emit light from a system of lamps and lenses and to serve as
     a beacon for navigational aid, for maritime pilots at sea or on inland
     waterways.. Lighthouses mark dangerous coastlines, hazardous shoals,
     reefs, rocks, and safe entries to harbors; they also assist in aerial
```

Figure 2: Use the **-n** flag to limit the number of search results. Enter the index number for a result to open in your browser.

On the Wing

By default, ddgr will show you the top 10 results for any search term. This can produce huge amounts of text and makes scrolling difficult. Fortunately there are a number of command-line parameters you can use to tweak your DuckDuckGo searches.

One of the most useful flags is `-n`, which you can use to change the number of search results displayed (Figure 2). For instance, to show just three search results for "lighthouses" at a time, use `ddgr Lighthouses -n 3`.

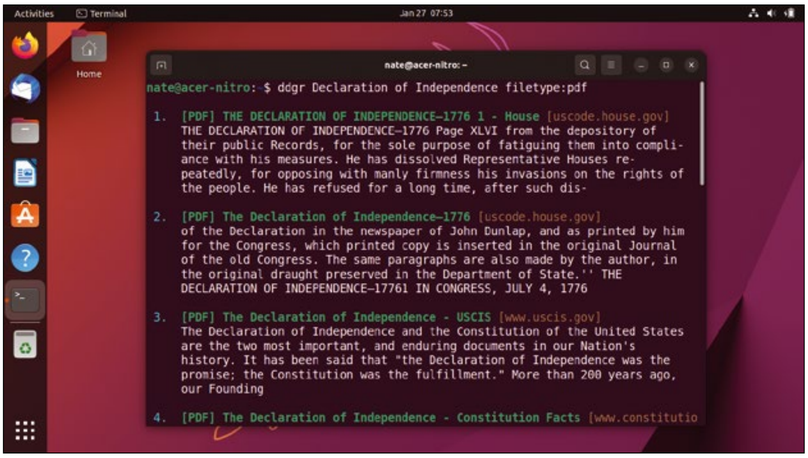
If you're looking for a specific search term, then you can include keywords in quotes, just as you would when searching on a browser-based search engine such as Google, for example:

```
ddgr "We hold these truths to be self-evident, ↩
  that all men are created equal"
```

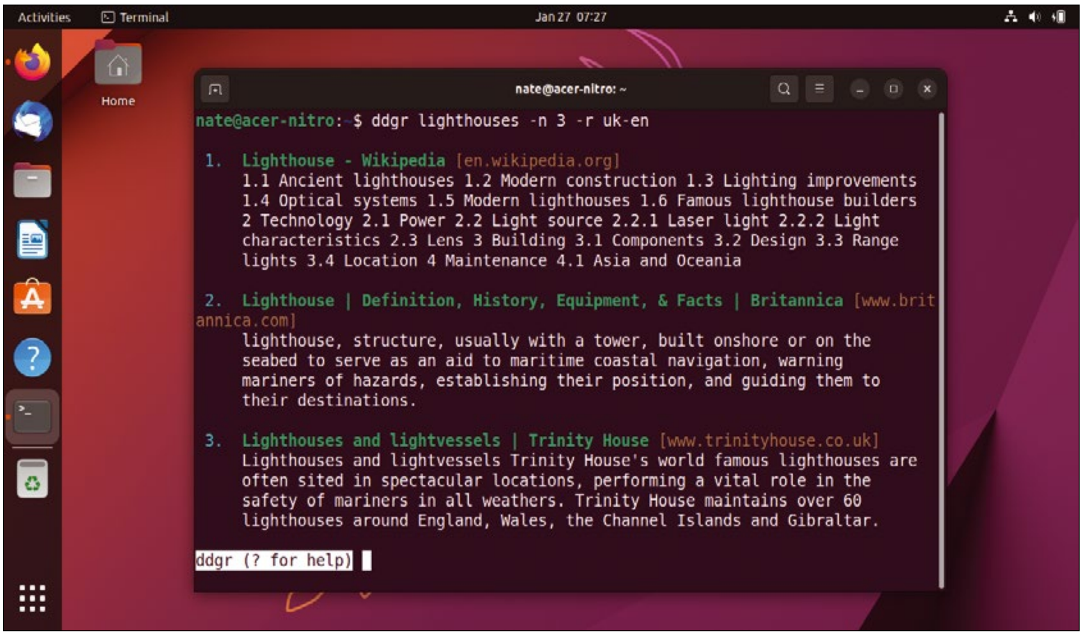
You can also ask ddgr to search only specific types of files using the `filetype:` parameter. For instance, in the example above where we look for a specific extract from the Declaration of Independence, you could also instruct ddgr to look only for PDFs containing this phrase using (Figure 3):

```
ddgr "We hold these truths to be self-evident, ↩
  that all men are created equal" filetype:pdf
```

Figure 3: The utility even supports searching for certain types of files such as PDFs.



```
nate@acer-nitro:~$ ddgr Declaration of Independence filetype:pdf

 1.  [PDF] THE DECLARATION OF INDEPENDENCE—1776 1 - House [uscode.house.gov]
     THE DECLARATION OF INDEPENDENCE—1776 Page XLVI from the depository of
     their public Records, for the sole purpose of fatiguing them into compli-
     ance with his measures. He has dissolved Representative Houses re-
     peatedly, for opposing with manly firmness his invasions on the rights of
     the people. He has refused for a long time, after such dis-

 2.  [PDF] The Declaration of Independence—1776 [uscode.house.gov]
     of the Declaration in the newspaper of John Dunlap, and as printed by him
     for the Congress, which printed copy is inserted in the original Journal
     of the old Congress. The same paragraphs are also made by the author, in
     the original draught preserved in the Department of State.'' THE
     DECLARATION OF INDEPENDENCE—17761 IN CONGRESS, JULY 4, 1776

 3.  [PDF] The Declaration of Independence - USCIS [www.uscis.gov]
     The Declaration of Independence and the Constitution of the United States
     are the two most important, and enduring documents in our Nation's
     history. It has been said that "the Declaration of Independence was the
     promise; the Constitution was the fulfillment." More than 200 years ago,
     our Founding

 4.  [PDF] The Declaration of Independence - Constitution Facts [www.constitutio
```



```
nate@acer-nitro:~$ ddgr lighthouses -n 3 -r uk-en

 1.  Lighthouse - Wikipedia [en.wikipedia.org]
     1.1 Ancient lighthouses 1.2 Modern construction 1.3 Lighting improvements
     1.4 Optical systems 1.5 Modern lighthouses 1.6 Famous lighthouse builders
     2 Technology 2.1 Power 2.2 Light source 2.2.1 Laser light 2.2.2 Light
     characteristics 2.3 Lens 3 Building 3.1 Components 3.2 Design 3.3 Range
     lights 3.4 Location 4 Maintenance 4.1 Asia and Oceania

 2.  Lighthouse | Definition, History, Equipment, & Facts | Britannica [www.brit
annica.com]
     lighthouse, structure, usually with a tower, built onshore or on the
     seabed to serve as an aid to maritime coastal navigation, warning
     mariners of hazards, establishing their position, and guiding them to
     their destinations.

 3.  Lighthouses and lightvessels | Trinity House [www.trinityhouse.co.uk]
     Lighthouses and lightvessels Trinity House's world famous lighthouses are
     often sited in spectacular locations, performing a vital role in the
     safety of mariners in all weathers. Trinity House maintains over 60
     lighthouses around England, Wales, the Channel Islands and Gibraltar.

ddgr (? for help)
```

Figure 4: Ddgr can also show results for a particular language or region.

Documents such as the Declaration of Independence are easy to find, but if you're looking for more recent news, ddgr also lets you filter search results to within a particular time frame using the `-t` parameter followed by the time in (d)ays, (m)onths, or (y)ears. For instance, if you wanted to see what the World Wide Web has had to say about Bitcoin in the past three months, you could enter

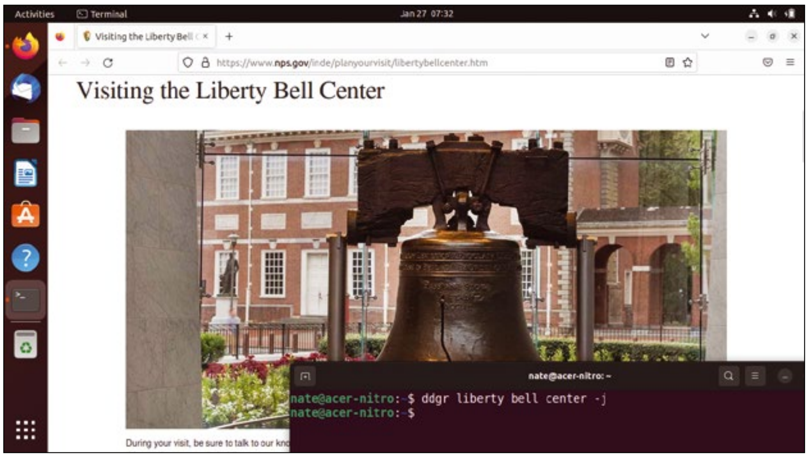
```
ddgr -t m 3 bitcoin
```

You can refine these results still further with `-w` flag, which instructs ddgr to list only results from a specific website. For instance, if you wanted to read only what *Linux Magazine* has had to say about Bitcoin recently, you could enter

```
ddgr -t m 3 bitcoin -w linux-magazine.com
```

You can also restrict your search results to particular regions if you want. The default is the US, but you can change this with the `-r` flag followed by the correct parameter for the country and language. For example, to see only British English pages on the subject of lighthouses, you'd type (Figure 4):

```
ddgr lighthouses -n 3 -r uk-en
```


Activities
Terminal
Jan 27 07:32
Visiting the Liberty Bell C
https://www.nps.gov/inde/planyourvisit/libertybellcenter.htm
Visiting the Liberty Bell Center
During your visit, be sure to talk to our kno
nate@acer-nitro: ~

```
nate@acer-nitro:~$ ddgr liberty bell center -j
nate@acer-nitro:~$
```

Figure 5: If you're feeling lucky, just use `-j` to open the first search result in your browser.

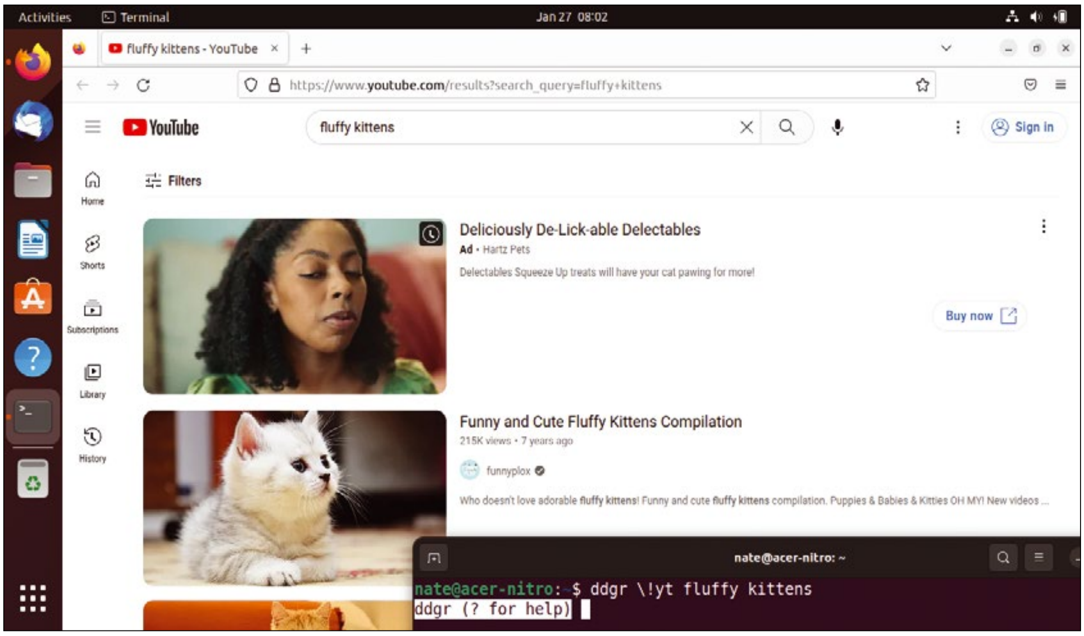
A full list of DuckDuckGo's regional parameters are available on the main website [7].

The utility also has its own equivalent of Google's "I'm feeling lucky," in that you can tell it just to open the first search result that appears in your browser. You can do this with the `-j` flag. For example, running `ddgr liberty bell center -j` will open the official page for the Liberty Bell Center on the National Parks Service website because that's the first search result that appears in DuckDuckGo (Figure 5).

If you're still having trouble finding what you need, you can disable safe search to see results uncensored using the `--unsafe` flag at your own risk (e.g., `ddgr --unsafe piratebay films`). You can also disable the user agent using `--noua`, but I found in testing this that ddgr threw up a 403 (Forbidden) error when I tried to search for new sites.

Bangs and Browsers

The main DuckDuckGo search engine offers a very useful feature called bangs. This allows you to search certain websites without using a search engine at all. This is done using a *!* plus DuckDuckGo's particular code for that website. For example, if you wanted to search YouTube for videos of cute, fluffy kittens, you'd enter *!yt fluffy kittens* into the search bar. Searches are run in exactly the same way on ddgr, with only a backslash added (Figure 6), for example:


Activities
Terminal
Jan 27 08:02
fluffy kittens - YouTube
https://www.youtube.com/results?search_query=fluffy+kittens
YouTube
fluffy kittens
Sign in
Home
Shorts
Subscriptions
Library
History
Filters
Deliciously De-Lick-able Delectables
Ad • Hartz Pets
Delectables Squeeze Up treats will have your cat pawing for more!
Buy now
Funny and Cute Fluffy Kittens Compilation
215K views • 7 years ago
funnyplox
Who doesn't love adorable fluffy kittens! Funny and cute fluffy kittens compilation. Puppies & Babies & Kitties OH MY! New videos ...
nate@acer-nitro: ~

```
nate@acer-nitro:~$ ddgr \!yt fluffy kittens
ddgr (? for help)
```

Figure 6: Just add a `\` in ddgr to use DuckDuckGo bangs just like on the website.

```
ddgr \!yt fluffy kittens
```

Bangs are available for other mainstream websites, too, such as Wikipedia, Amazon, and even Google itself. If you're not familiar with DuckDuckGo, their blog has a full list of available websites [8].

If you already used a text-based browser, you can pass search results to it from ddgr. This is done using the `BROWSER` variable. To do this on a one-time basis, just run your search with your browser name, e.g., `BROWSER=lynx ddgr lighthouses`. For this session, any search results you open in ddgr will be displayed in the Lynx browser (Figure 7).

This is where the `d` flag can come in handy, because if you want to keep passing ddgr search results to the browser for this session, you can just enter `d` with the search term at the omniprompt (e.g., `d museums`).

If you want ddgr to always pass results to your text-based browser, you'll need to add this to the `.bashsrc` file and specify its name, for example:

```
echo export BROWSER=lynx >> ~/.bashrc
```

Currently ddgr supports ELinks, Links, Lynx, w3m, and www-browser out of the box.

A Word on Tor

DuckDuckGo has a good relationship with the Tor Project, given that their search engine is used by the Tor Browser. The Tor Project said they chose DuckDuckGo because the search engine doesn't log or collect your personal information [9]. DuckDuckGo even has its own Tor hidden service (`.onion`) address and operates its own Tor exit enclave [10].

In light of this, if you do want to run DuckDuckGo searches over Tor, by far the simplest way is just to download and use the Tor Browser, which also doesn't record any activity and is optimized for your privacy. Still, ddgr does also support searches over Tor, with some extra help. The utility itself can connect to the Internet via an HTTPS proxy. The Tor command-line utility uses a SOCKS5 proxy, so in order to get the two talking, you need to use a non-caching web proxy such as Privoxy.

This isn't too hard to set up. If you're running a Debian-based installation, just open a new terminal instance and enter

```
sudo apt-get install tor privoxy
```

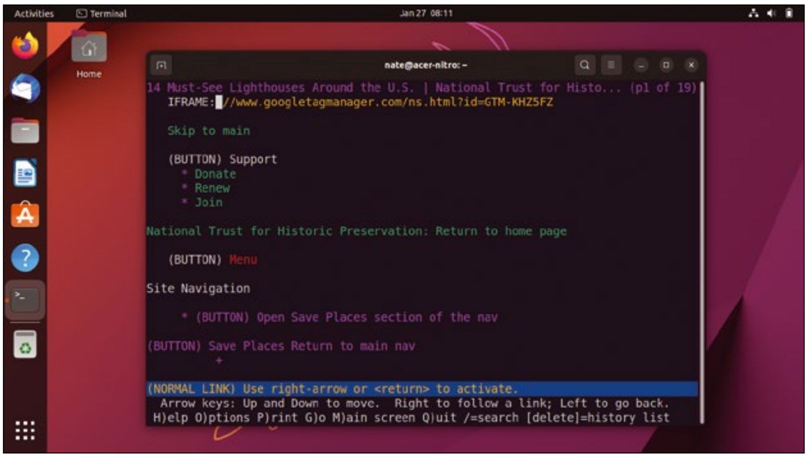
Next, configure Privoxy to use Tor by editing `/etc/privoxy/config`:

```
sudo nano /etc/privoxy/config
```

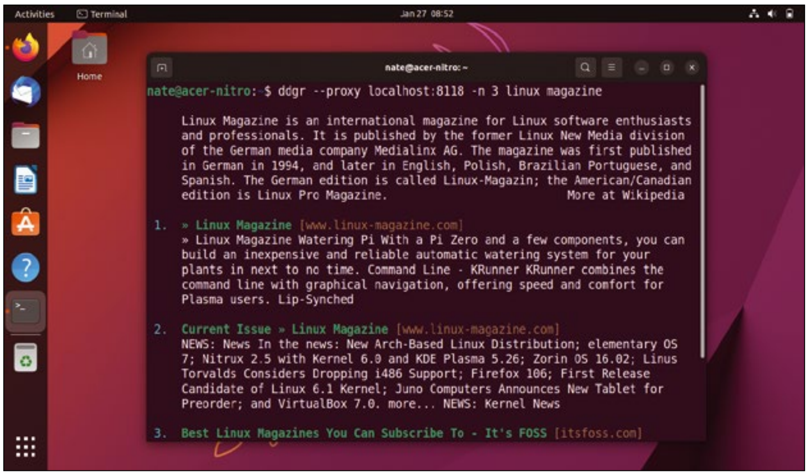
Scroll to the end and add the text

```
forward-socks4a / localhost:9050 .
```

Figure 7: Search results can be passed to a text-based browser such as Lynx.



```
14 Must-See Lighthouses Around the U.S. | National Trust for Histo... (p1 of 19)
   IFRAME: //www.googletagmanager.com/ns.html?id=GTM-KHZ5FZ

   Skip to main

   (BUTTON) Support
     * Donate
     * Renew
     * Join

National Trust for Historic Preservation: Return to home page

   (BUTTON) Menu

Site Navigation

     * (BUTTON) Open Save Places section of the nav

(BUTTON) Save Places Return to main nav
         +

(NORMAL LINK) Use right-arrow or <return> to activate.
  Arrow keys: Up and Down to move.  Right to follow a link; Left to go back.
 H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list
```



```
nate@acer-nitro:~$ ddgr --proxy localhost:8118 -n 3 linux magazine

      Linux Magazine is an international magazine for Linux software enthusiasts
      and professionals. It is published by the former Linux New Media division
      of the German media company Medialinx AG. The magazine was first published
      in German in 1994, and later in English, Polish, Brazilian Portuguese, and
      Spanish. The German edition is called Linux-Magazin; the American/Canadian
      edition is Linux Pro Magazine.                          More at Wikipedia

  1.  » Linux Magazine [www.linux-magazine.com]
      » Linux Magazine Watering Pi With a Pi Zero and a few components, you can
      build an inexpensive and reliable automatic watering system for your
      plants in next to no time. Command Line - KRunner KRunner combines the
      command line with graphical navigation, offering speed and comfort for
      Plasma users. Lip-Synched

  2.  Current Issue » Linux Magazine [www.linux-magazine.com]
      NEWS: News In the news: New Arch-Based Linux Distribution; elementary OS
      7; Nitrux 2.5 with Kernel 6.0 and KDE Plasma 5.26; Zorin OS 16.02; Linus
      Torvalds Considers Dropping i486 Support; Firefox 106; First Release
      Candidate of Linux 6.1 Kernel; Juno Computers Announces New Tablet for
      Preorder; and VirtualBox 7.0. more... NEWS: Kernel News

  3.  Best Linux Magazines You Can Subscribe To - It's FOSS [itsfoss.com]
```

Figure 8: Here I've used ddgr with Tor and Privoxy to run a search via the Tor network.

Save and exit. You'll next need to restart both Privoxy and Tor for these changes to take effect:

```
sudo /etc/init.d/privoxy restart
```

```
sudo service tor restart
```

You'll now be the proud owner of a local HTTP proxy using Tor on port 8118. This means you can now use Tor with ddgr with the `--proxy` parameter to run searches via the Tor network (Figure 8), such as:

```
ddgr --proxy localhost:8118 lighthouses
```

To my disappointment during testing, I wasn't able to use this feature to search within `.onion` domains such as the address for the *New York Times* website using the `-w` flag. This would have been a useful feature, but in fairness, the developer doesn't claim that the utility can access Tor hidden services, just that it can run searches using Tor. Naturally if you want this extra level of privacy, be prepared for much longer search times because your data will be routed through various Tor relays.

DuckDuckGo for It

There's a lot to be said for searching the web via your command line: It's clean, it's quick, and with time, very easy to navigate. You can link parameters together to fine-tune your search results, all without handing your personal information over to the gods of Google.

We've also only touched briefly on the multitude of command-line parameters available in ddgr. There are so many, in fact, that the utility does away with the usual config file because everything is right there in the program. Other features include displaying results in different colors, listing searches in reverse order, and even exporting them in JSON format (presumably for research purposes).

Although ddgr and googler have the same creator, the search engines they represent have very different business models. One is committed to harvesting all your data to serve your targeted advertising; the other might show you the occasional ad but isn't collecting any information in doing so.

Ridding yourself of a conventional browser when searching is also a great way to prevent snoopers who want to "fingerprint" your browser through nefarious methods such as playing sounds or drawing complex, invisible images on your web page. Still, at some point you'll need to open your browser to view these results.

If you're happy to use a text-based browser all the time, you'll be shielded from most of the nastier browser fingerprinting and tracking techniques out there. For those who still prefer a GUI, make sure to check your browser settings to ensure Do Not Track is enabled and use a reliable ad blocker to filter out most malicious URLs. Consider also installing an extension to block third-party/cross-site cookies such as Ghostery [11]. ∎∎∎

The Author

Nate Drake is a tech journalist specializing in cybersecurity and retro tech.

Info

[1] Android Open Source Project: *https://source.android.com/*

[2] "Google Fined $57M by Data Protection Watchdog Over GDPR Violations": *https://digitalguardian.com/blog/google-fined-57m-data-protection-watchdog-over-gdpr-violations*

[3] "Why Should I Use DuckDuckGo Instead of Google?": *https://spreadprivacy.com/why-use-duckduckgo-instead-of-google/*

[4] "Adding DuckDuckGo to Your Browser": *https://help.duckduckgo.com/duckduckgo-help-pages/desktop/adding-duckduckgo-to-your-browser/*

[5] jarun (Arun) on GitHub: *https://github.com/jarun*

[6] Ubuntu man page – xsel: *https://manpages.ubuntu.com/manpages/xenial/en/man1/xsel.1x.html*

[7] DuckDuckGo parameters: *https://help.duckduckgo.com/duckduckgo-help-pages/settings/params/*

[8] "DuckDuckGo Bangs – Search Directly On Your Favorite Websites": *https://ostechnix.com/duckduckgo-bangs-search-directly-on-your-favorite-websites/*

[9] Tor Project – Support: *https://support.torproject.org/tbb/tbb-41/*

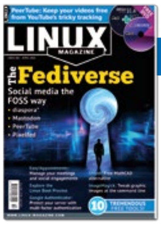
[10] Tor Exit Enclave – DuckDuckGo Help Pages: *https://help.duckduckgo.com/privacy/tor-exit-enclave/*

[11] Ghostery free ad blocker extension: *https://www.ghostery.com/ghostery-ad-blocker*

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.

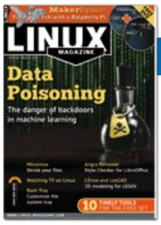


#269/April 2023

The Fediverse

Social media tools connect the world, bringing us the latest news and commentary from politicians, movie stars, community leaders, and remote friends. But the tracking and data mining of the commercial social media platforms has left many users searching for a better option. This month we dive down into the alternative universe for social media users: the Fediverse.

On the DVD: EndeavourOS Cassini 22.12 and Debian 11.6 “bullseye”

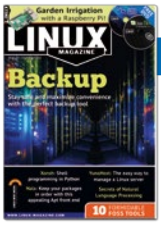


#268/March 2023

Data Poisoning

Think computers don’t make mistakes? If you slip some doctored-up training samples into the mix, you can get a fancy machine-learning system to think a dog is a cat or a 1 is an 8 – and you can trigger this bad behavior through a hidden signal no one else will notice.

On the DVD: MX Linux 21.3 and Puppy Linux FossaPup 9.5



#267/February 2023

Backup

In theory, everyone could use the same backup tool, but the best way to build regular and systematic backups into your life is to find a solution that fits with your own habits and methods. This month, we preview some popular backup apps in the Linux space so you can find one that works for you.

On the DVD: Linux Mint 21 Cinnamon and Kali Linux 2022.4

#266/January 2023

Generative Adversarial Networks

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

On the DVD: Ubuntu 22.10 and AlmaLinux 9.0

#265/December 2022

Quantum Computing

Most Linux users know that this futuristic technology leverages the weird power of quantum mechanics. But how does it really work? What can I do with it? Are there tools available today that will help me experiment? This month we take a deep dive into quantum computing.

On the DVD: Manjaro 21.3.7-220816 and Arch Linux 2022.10.01

#264/November 2022

Artificial Intelligence

Machine learning remains shrouded in mystery even though it is now an integral part of our everyday world. This month we look behind the curtain at some popular techniques for supervised and unsupervised learning.

On the DVD: Debian 11.5 and Rocky Linux 9.0

FEATURED EVENTS

Users, developers, and vendors meet at Linux events around the world. We at *Linux Magazine* are proud to sponsor the Featured Events shown here.

For other events near you, check our extensive events calendar online at *https://www.linux-magazine.com/events.*

If you know of another Linux event you would like us to add to our calendar, please send a message with all the details to *info@linux-magazine.com*.

DrupalCon Pittsburgh

Date: June 5-8, 2023

Location: Pittsburgh, Pennsylvania

Website: *https://events.drupal.org/pittsburgh2023*

Don't miss your chance to attend the in-person DrupalCon North America 2023 in the incredible city of Pittsburgh, Pennsylvania. This year's event will be full of valuable insights, information, and connections. Whether you're new to Drupal or a long-time member of the Drupal community, you'll find new insights and connections to advance your career and your business.

SUSECON 2023

Date: June 20-22, 2023

Location: Munich, Germany

Website: *https://www.suse.com/susecon/*

SUSECON is the annual global conference for SUSE customers, partners, and community enthusiasts, providing information to meet the technical needs and business challenges of the enterprise IT customer. Here you will find open source solutions for your business needs based in Linux, Kubernetes and Edge computing. Choose from sessions and practical tutorials June 20-22 in Munich, Germany.

Akademy 2023

Date: July 15-21, 2023

Location: Thessaloniki, Greece & Online

Website: *https://akademy.kde.org/2023/*

Akademy is the annual world summit of KDE, one of the largest free software communities in the world. It is a free, non-commercial event organized by the KDE community.The conference is expected to draw hundreds of attendees from the global KDE Community. Akademy features a 2-day conference followed by 5 days of workshops, Birds of a Feather (BoF) and coding sessions.

Events

KubeCon + CloudNativeCon Europe 2023	Apr. 18-21	Amsterdam, Netherlands	https://events.linuxfoundation.org/
PyCon US 2023	Apr. 19-27	Salt Lake City, Utah	https://us.pycon.org/2023/
Linux App Summit 2023	Apr 21-23	Brno, Czech Republic	https://linuxappsummit.org/
DeveloperWeek Europe	Apr 26-27	Virtual Event	https://www.developerweek.com/europe/
Cloud Expo Europe	May 10-11	Frankfurt, Germany	https://www.cloudexpoeurope.de/en
Open Source Summit North America	May 10-12	Vancouver, Canada + Virtual	https://events.linuxfoundation.org/
DORS/CLUC 2023	May 11-12	Zagreb, Croatia	https://www.dorscluc.org/
Icinga Camp Berlin	May 17	Berlin, Germany	https://icinga.com/community/events/icinga-camp-berlin-2023/
ISC High Performance	May 21-25	Hamburg, Germany	https://www.isc-hpc.com/about-overview.html
Percona Live 2023	May 22-24	Denver, Colorado	https://www.percona.com/live/conferences
openSUSE Conference 2023	May 26-28	Nürnberg, Germany	https://events.opensuse.org/conferences/oSC23
CloudFest USA	May 31-Jun. 3	Austin, Texas	https://www.cloudfest.com/usa/
DrupalCon Pittsburgh	June 5-8	Pittsburgh, Pennsylvania	https://events.drupal.org/pittsburgh2023
OpenSouthCode 2023	June 9-10	Málaga, Spain	https://bit.ly/opensouthcode2023
OW2con'23	June 14-15	Paris-Chatillon, France	https://www.ow2con.org/view/2023/
DevConf.CZ 2023	June 16-18	Brno, Czech Republic	https://www.devconf.info/cz/
SUSECON 2023	June 20-22	Munich, Germany	https://www.suse.com/susecon/
KCDC 2023 (Kansas City Developer Conference)	June 21-23	Kansas City, Missouri	https://www.kcdc.info/

Images © Alex White, 123RF.com

Contact Info

Editor in Chief
Joe Casad, jcasad@linux-magazine.com
Copy Editors
Amy Pettle, Aubrey Vaughn
News Editors
Jack Wallen, Amber Ankerholz
Editor Emerita Nomadica
Rita L Sooby
Managing Editor
Lori White
Localization & Translation
Ian Travis
Layout
Dena Friesen, Lori White
Cover Design
Dena Friesen
Cover Image
© Ledda Maria Rita, 123RF.com
Advertising
Brian Osborn, bosborn@linuxnewmedia.com
phone +49 8093 7679420
Marketing Communications
Gwen Clark, gclark@linuxnewmedia.com
Linux New Media USA, LLC
4840 Bob Billings Parkway, Ste 104
Lawrence, KS 66049 USA
Publisher
Brian Osborn
Customer Service / Subscription
For USA and Canada:
Email: cs@linuxnewmedia.com
Phone: 1-866-247-2802
(Toll Free from the US and Canada)

For all other countries:
Email: subs@linux-magazine.com

www.linux-magazine.com

While every care has been taken in the content of the magazine, the publishers cannot be held responsible for the accuracy of the information contained within it or any consequences arising from the use of it. The use of the disc provided with the magazine or any material provided on it is at your own risk.

Copyright and Trademarks © 2023 Linux New Media USA, LLC.

No material may be reproduced in any form whatsoever in whole or in part without the written permission of the publishers. It is assumed that all correspondence sent, for example, letters, email, faxes, photographs, articles, drawings, are supplied for publication or license to third parties on a non-exclusive worldwide basis by Linux New Media USA, LLC, unless otherwise stated in writing.

Linux is a trademark of Linus Torvalds.

All brand or product names are trademarks of their respective owners. Contact us if we haven't credited your copyright; we will always correct any oversight.

Printed in Nuremberg, Germany by Zeitfracht GmbH.

Distributed by Seymour Distribution Ltd, United Kingdom

Represented in Europe and other territories by: Sparkhaus Media GmbH, Bialasstr. 1a, 85625 Glonn, Germany.

Published monthly as Linux Magazine (Print ISSN: 1471-5678, Online ISSN: 2833-3950) by Linux New Media USA, LLC, 4840 Bob Billings Parkway, Ste 104, Lawrence, KS 66049, USA. Periodicals Postage paid at Lawrence, KS and additional mailing offices. Ride-Along Enclosed. POSTMASTER: Please send address changes to Linux Magazine, 4840 Bob Billings Parkway, Ste 104, Lawrence, KS 66049, USA.

WRITE FOR US

Linux Magazine is looking for authors to write articles on Linux and the tools of the Linux environment. We like articles on useful solutions that solve practical problems. The topic could be a desktop tool, a command-line utility, a network monitoring application, a homegrown script, or anything else with the potential to save a Linux user trouble and time. Our goal is to tell our readers stories they haven't already heard, so we're especially interested in original fixes and hacks, new tools, and useful applications that our readers might not know about. We also love articles on advanced uses for tools our readers *do* know about – stories that take a traditional application and put it to work in a novel or creative way.

Topics close to our hearts include:

- Security
- Advanced Linux tuning and configuration
- Internet of Things
- Networking
- Scripting
- Artificial intelligence
- Open protocols and open standards

If you have a worthy topic that isn't on this list, try us out – we might be interested!

Please don't send us articles about products made by a company you work for, unless it is an open source tool that is freely available to everyone. Don't send us webzine-style "Top 10 Tips" articles or other superficial treatments that leave all the work to the reader. We like complete solutions, with examples and lots of details. Go deep, not wide.

We have a couple themes coming up that we could use your help with. Please send us your proposals for thoughtful and practical articles on:

- **Cryptocurrencies**
- **Systemd hacks**

Describe your idea in 1-2 paragraphs and send it to: **edit@linux-magazine.com**.

Please indicate in the subject line that your message is an article proposal.

Authors

Erik Bärwaldt	32	Kristian Kißling	16
Zack Brown	12	Martin Gerhard Loschwitz	36
Bruce Byfield	24, 42	Vincent Mealing	71
Joe Casad	3, 6	Pete Metcalfe	67
Mark Crutch	71	Graham Morrison	84
Nate Drake	90	Alan Guy Pilbeam	44
Hans-Georg Eßer	28	Mike Schilli	54
Jon "maddog" Hall	72	Tim Schürmann	18
Frank Hofmann	60	Ferdinand Thommes	50, 73, 80
Swen Hopfe	63	Jack Wallen	8

Approximate

UK / Europe	May 05
USA / Canada	Jun 09
Australia	Jul 21

On Sale Date

Please note: On sale dates are approximate and may be delayed because of logistical issues.

Issue 271 / June 2023

Smart Home

For many Linux users, home remote control is the ultimate hacker quest. Turn off lights? Close the blinds? Adjust the thermostat? If you know Linux, you can have home automation without giving up your privacy.

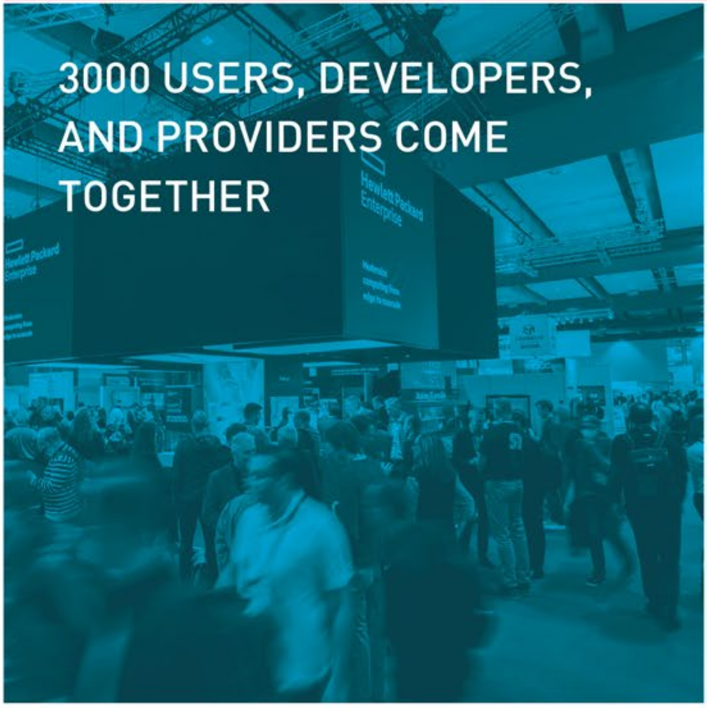
Preview Newsletter

The Linux Magazine Preview is a monthly email newsletter that gives you a sneak peek at the next issue, including links to articles posted online.

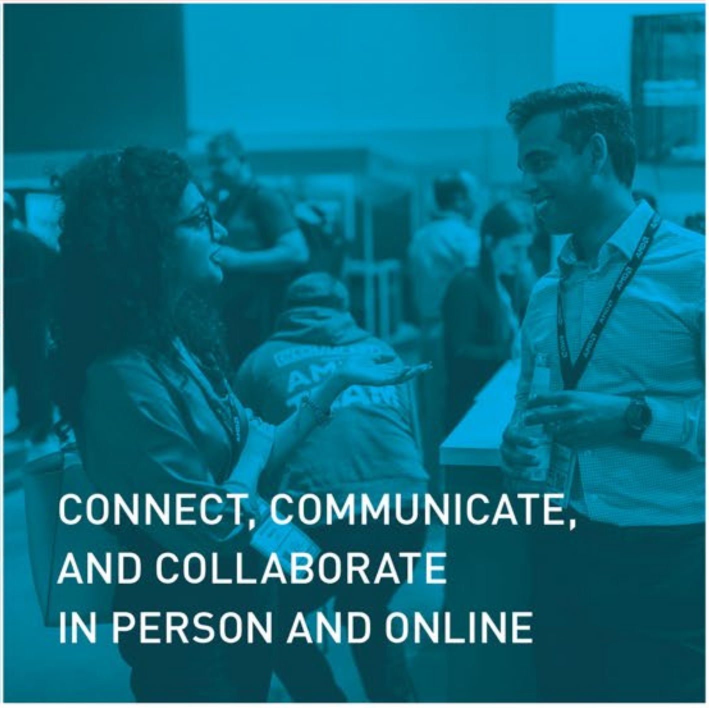
Sign up at: ***https://bit.ly/Linux-Update***

Image © pavel shlykov, 123RF.com

HPC
MACHINE LEARNING
DATA ANALYTICS
QUANTUM COMPUTING



3000 USERS, DEVELOPERS, AND PROVIDERS COME TOGETHER



CONNECT, COMMUNICATE, AND COLLABORATE IN PERSON AND ONLINE

isc-hpc.com

ISC 2023 | MAY 21 – 25 | HAMBURG, GERMANY

IMAGINE
TOMORROW

ISC
High Performance

REGISTRATION IS OPEN
EARLY-BIRD RATES UNTIL APRIL 19

HETZNER

HIGH POWER - HIGHER EFFICIENCY
LOWER COSTS!

DEDICATED SERVER EX-LINE
incl. Intel® Core™ i9-13900 or i5-13500

DEDICATED SERVER EX101

- ✓ Intel® Core™ i9-13900
- ✓ 64 GB DDR5 ECC RAM
- ✓ 2 x 1.92 TB Gen4 NVMe SSD
- ✓ 100 GB Backup Space
- ✓ Traffic unlimited
- ✓ Location Finland and Germany
- ✓ No minimum contract
- ✓ Setup Fee $41.90

intel CORE i9

monthly from $ 90.20

DEDICATED SERVER EX44

- ✓ Intel® Core™ i5-13500
- ✓ 64 GB DDR4 RAM
- ✓ 2 x 512 GB Gen4 NVMe SSD
- ✓ 100 GB Backup Space
- ✓ Traffic unlimited
- ✓ Location Germany
- ✓ No minimum contract
- ✓ Setup Fee $41.90

intel CORE i5

monthly $ 47.30

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