



LINUX

MAGAZINE

ISSUE 268 – MARCH 2023

Data Poisoning

The danger of backdoors in machine learning



Minuimus
Shrink your files

Angry Reviewer
Style Checker for LibreOffice

Watching TV on Linux

LDraw and LeoCAD
3D modeling for LEGOs

Bash Tray
Customize the system tray

10 TIMELY TOOLS FOR THE FOSS SET





Qui(e)te powerful!

TUXEDO Pulse 15 - Gen2



AMD Ryzen 7 5700U-35W
8 cores | 16 threads



WQHD display
2560 x 1440 | 165 Hz



Up to 18 h of runtime
91 Wh battery



Rigid magnesium chassis
1,7 cm thin | 1,5 kg light



100%
Linux

5

Year
Warranty



Lifetime
Support



Built in
Germany



German
Privacy



Local
Support

TUXEDO

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

I'LL BE WATCHING YOU

Dear Reader,

The elite gathering known as Davos is not an ordinary business convention. The annual conference of the World Economic Forum is an uber-exclusive event, not a meeting of governments but rather a gathering of representatives from around 1,000 member companies – the biggest companies in the world – along with a smattering of political leaders and academics.

At this year's Davos event, Microsoft hosted a party for 50 guests and invited the rock star Gordon Matthew Thomas Sumner, otherwise known as Sting, to provide the entertainment for the party. Estimates vary for how much it costs for Sting to show up and play at your party, but the remuneration is thought to be well in the six-figure range.

The news of this musical mixer might easily have escaped notice if Microsoft had not arisen the next day to lay off 10,000 employees. The grim layoffs, and the accompanying call to austerity, seemed oddly discordant with the extravagant excess of hiring Sting to headline a meet-and-greet, and Microsoft was quickly denounced in the tech press and social media.

I have to admit, the Sting party is an easy mark for opinion pagers like me. The optics are just terrible. But the problem with really bad ideas like this one is that everyone guffaws about how bad it is and moves on, leaving the rest of the story unexamined.

One thing about this weird and unfathomable choice is that it lays bare the unspoken truth that technical budgets and marketing budgets come from totally different universes. All kinds of money gets poured into marketing and sales that seems totally irrational to everyone else. I talked to a marketing consultant once who said she had a client who spent \$1 million on a booth for a trade show, but the booth was so exotic and eye-catching that it won them \$10 million in air time because the booth was featured on a national news program. (News time is way more valuable than advertising time, she pointed out.) I was at a show once and walked past a booth that looked like the bridge of a starship. I said, "Wow that really looks like the bridge of the starship Enterprise," and they told me, "It doesn't just *look* like it, it *is* the bridge of the starship Enterprise. We rented it from the studio and shipped it here." (Don't ask me to fact check that one – I'm just telling you what they told me.) And then there was the trick-riding motocross street party, with motorcycles roaring up a ramp, doing flips in the air, and roaring down another ramp, while friendly helpers dispensed hits of pure oxygen in slushy flavors like "raspberry" and "wintergreen." I've seen a Donald Trump impersonator, Maori dancers, a mechanical bull, and 6-foot-tall identical twin runway models (promoting a mirrored storage drive, as I recall) ... None of these spectacles had *anything* to do with the products the com-

panies were trying to sell. It was all 100 percent for the purpose of getting attention. But attention is so valuable that executives are willing to shell out unbelievable sums for seemingly pointless things.

Why exactly *would* Microsoft want to hire Sting to play at a party for 50 people? One reason might be because they want everyone to know they *can* hire Sting to play at a party for 50 people. Microsoft has always been good at projecting power and dominance. Now they aren't as powerful and dominant as they used to be, but they can still look powerful and dominant by throwing money around in ways that make it seem like their money is an endless font. Whereas to us, Sting and the layoffs seem contradictory, to Microsoft, they counterbalance perfectly, because people will say, "They laid off 10,000, which implies they don't have enough money, but look at all this money they threw down on Sting, so maybe they really are still rich and powerful after all." The problem with this approach is that, despite what the C-suite executives in the company might think, people never did really like Microsoft's power-and-dominance act in the first place, and it is even more annoying if you have to base it on trick mirrors and gimmicks.

The other thing to know is that the kind of people who show up at Davos are really hard to talk to anywhere else. It is very difficult to catch the CEO of a company that does \$5 billion per year in business on an elevator. Think of all the computer services that a \$5 billion business could buy. Of course, nobody actually inks a service deal at this kind of event. They just make "connections," so the next time the CEO is in a board meeting and there is a question about the IT infrastructure, he can say "I know a guy at Microsoft. Let's call him ..."

If hiring Sting leads to a deal with even one of those 50 guests, it was probably worth it financially. But all the machinations on Mount Olympus have little meaning to the 10,000 workers who lost their jobs the day after the Sting soiree. Layoffs are inevitable in the high-tech industry, but a little respect for the dignity of those 10,000 workers would have been a good thing too. In fact, it might have even been good marketing. ■■■

Joe

Joe Casad,
Editor in Chief



ON THE COVER

34 **Bash Tray Scripts**

Write a script that generates a custom system tray app.

40 **Minuimus**

Use this handy tool to trim your bloated files – without changing the contents.

54 **Angry Reviewer**

Tighten up your prose with this discerning LO extension.

58 **Automatic Fish Feeder**

A Raspberry Pi Zero will keep your fish swimming while you're away.

80 **IPTVnator**

Explore the world of Internet Protocol Television.

90 **LDraw and LeoCAD**

Well, of course you should design your LEGO structures on a CAD system!

NEWS

08 **News**

- Nohara Project Is a Version of Fedora with Fixes
- Gnome 44 Now Has a Release Date
- Nitrox 2.6 Is Available with Kernel 6.1 and a Change
- Vanilla OS Initial Release Is Now Available
- Critical Linux Vulnerability Found to Impact SMB Servers
- Linux Mint 21.1 Now Available with Plenty of Changes
- Another Attempt at a Linux Tablet Is in the Works
- Now Available: Designing with LibreOffice 2nd Edition
- KaOS Linux 2022.12 Has Plenty to Be Excited About

12 **Kernel News**

- Bug Tracking

COVER STORIES

16 **Data Poisoning**

Machine learning can be maliciously manipulated – we'll show you how.

REVIEWS

22 **Distro Walk – NuTyX**

Thierry Nuttens, the developer of NuTyX, shares a behind-the-scenes look at a small Linux distribution.

95 Back Issues

97 Call for Papers

96 Events

98 Coming Next Month

IN-DEPTH

24 **DWS Remote Control**

DWS Remote Control offers convenient browser access to computers outside of your home network.

28 **Firmware in Debian**

The Debian project has a new direction on non-free firmware.

34 **Bash Tray**

YAD lets you customize your system tray with one-line Bash tray scripts.

40 **Minuimus**

Minuimus helps you save disk space by reducing the file size of PDF files.

42 **apt-clone**

In the right circumstances, apt-clone can be a simple option for cloning your Debian system.

44 **Programming Snapshot – Go and R**

Spotify collects data about its users and their taste in music. Mike decided to investigate.

50 **unsnap**

The unsnap script removes snaps from your computer and replaces them with flatpaks.

54 **LibreOffice Writer Angry Reviewer**

The Angry Reviewer style check can be used to evaluate and improve any type of writing, including academic articles and grant applications.



16 Data Poison

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.

MakerSpace

58 Automatic Fish Feeder

Whether at work or on vacation, every pet lover worries about how to take care of their little roommates. What every aquarium owner needs is an automatic feeder.

62 Solar Array

A forecast service and some Perl magic help predict the solar power yield of a residential photovoltaic array.



@linux_pro



@linuxpromagazine



Linux Magazine



@linuxmagazine



LINUXVOICE

71 Welcome

This month in Linux Voice.

72 Doghouse – Software Freedom

Restricting uses for FOSS may seem appealing, but it also might not be the solution some imagine.

73 Docker

Do you work with Ubuntu but need to test something on openSUSE? You don't need a second PC or a virtual machine – a single container is quite enough.

80 IPTVnator

The IPTV standard lets you view your favorite channels on Linux.

84 FOSSPicks

This month Graham looks at LosslessCut, Webcamoid, Sniffnet, and more!

90 Tutorial – LDraw and LeoCAD

LDraw and LeoCAD help you become a virtual LEGO architect.

TWO TERRIFIC DISTROS

DOUBLE-SIDED DVD!

SEE PAGE 6 FOR DETAILS

MX Linux 21.3 and Puppy Linux FossaPup 9.5

Two Terrific Distros on a Double-Sided DVD!

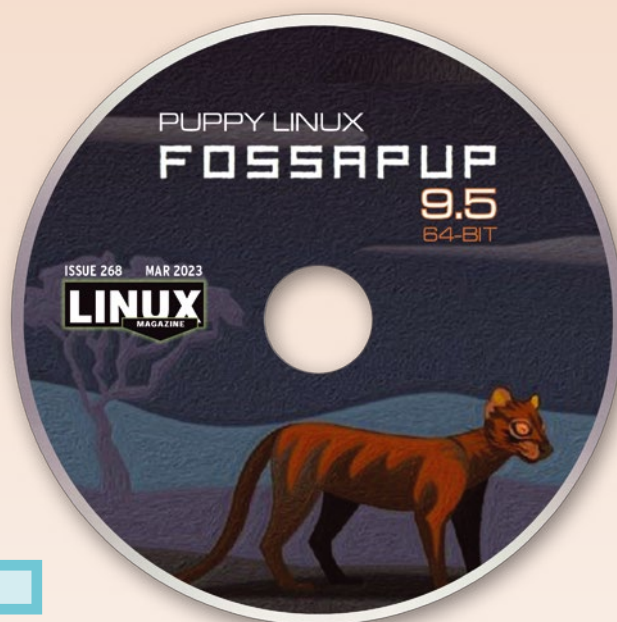


MX Linux 21.3 64-bit

Codenamed Wildflower, MX Linux 21.3 (MX-21.3) is the latest joint effort between the antiX and MX Linux communities. Based on Debian 11.6, Wildflower is available with the latest releases of the Xfce, KDE Plasma, and Fluxbox desktop environments.

Changes to Fluxbox include MX Rofi Manager, a tool for configuring the look and feel, while KDE now includes the 6.0 Advanced Hardware Support (AHS) kernel with advanced hardware configuration. All editions include a new deb-installer, menu editor, and updated firmware packages, as well as the latest antiX tool for creating live images and system snapshots.

As with all MX releases, the result is a distribution with extreme user-friendliness, suitable for all levels of users.



Puppy Linux FossaPup 9.5 64-bit

While Puppy Linux is usually listed as a distribution, it is actually a family of distributions. Originally aimed at small installations, Puppy Linux is now known for the features that they all have in common. Included on this month's DVD is FossaPup 9.5, one of Puppy's better known distributions. To try other Puppy Linux distributions, download them at <https://puppylinux-woof-ce.github.io/>.

All Puppy Linux distros are built using woof-CE, which creates a distro from an existing binary (usually Ubuntu or Slackware) and can be installed on a single directory or an entire filesystem. Each uses its own choice of widgets, applications, and features. However, each Puppy Linux loads system files into RAM and encrypts personal files rather than using passwords. Each, too, opens with a Quick Setup window and can save the current setup when shutting down.

Like all Puppy distros, FossaPup installs quickly and offers some unusual approaches to computing that can be appreciated by all users.

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.

CLOUDFEST

March 21-23, 2023
Europa-Park, Germany

6,000+ Participants
250+ Speakers

150+ Partners
65 Countries

The world's largest cloud industry event is ready to once again take over a spectacular European amusement park to facilitate new partnerships, deep knowledge sharing, and the best parties the industry has ever seen.



Start your CloudFest Journey
AND SAVE €399!

With your FREE Code: **CF23ADMIN**

scan me!



reg.cloudfest.com

NEWS

Updates on technologies, trends, and tools

THIS MONTH'S NEWS

- 08** • Nobara Project Is a Modified Version of Fedora with User-Friendly Fixes
- Gnome 44 Now Has a Release Date
- 09** • Nitrox 2.6 Is Available with Kernel 6.1 and a Major Change
- Vanilla OS Initial Release Is Now Available
- More Online
- 10** • Critical Linux Vulnerability Found to Impact SMB Servers
- Linux Mint 21.1 Now Available with Plenty of Look and Feel Changes
- Another Attempt at a Linux Tablet Is in the Works
- 11** • Now Available: Designing with LibreOffice 2nd Edition
- KaOS Linux 2022.12 Has Plenty to Be Excited About

Nobara Project Is a Modified Version of Fedora with User-Friendly Fixes

The Nobara Project's aim is to bring users a better gaming, streaming, and content-creation solution out of the box.

This new Linux distribution (which is not a Fedora Spin) is a completely independent project. This Linux operating system includes NVIDIA drivers, Wine dependencies, OBS Studio, third-party codecs, and a collection of package fixes that are geared toward making it easy for users to immediately be productive, without having to tweak, install, or patch anything.

Pre-installed packages include Blender, DaVinci Resolve, OBS Studio, Wine, Proton, Discord, Flatpak, Steam, Lutris, OnlyOffice, VapourSynth, and much more. You'll also find the RPM Fusion repository that includes both free and non-free software.

Unlike Fedora, Nobara uses AppArmor because it claims AppArmor is more user-friendly, less intrusive, and easier to manage. Nobara ships with the same kernel used in Fedora, with a few added patches.

The Nobara Project ships with the Gnome desktop, and there are no plans to include any other environments.

To read more about Nobara, check out the official website (<https://nobaraproject.org/>) and download an ISO for installation from the Nobara Project download page (<https://nobaraproject.org/download-nobara/>).

Gnome 44 Now Has a Release Date

Although it's a bit premature to talk about what features will be found in the 44th iteration of the Gnome desktop, it's safe to say that the Epiphany web browser will most likely get the GTK4 port we've been waiting for as well as WebExtensions support that will be enabled out of the box. There also may be wider libadwaita support in more apps.

The big question on my mind is will Gnome 44 finally feature the new terminal app, Console, or will it continue on with the current default Gnome terminal?

According to 9To5Linux (<https://9to5linux.com/gnome-44-release-date>), Gedit is making a comeback, although there is no indication it will return as the default text editor. My guess is that the new default will remain the same with the newly-updated Gedit available as an alternative.

The official release schedule for Gnome 44 looks like this:

- Gnome 44 Alpha: Jan 7, 2023
- Gnome 44 Beta: Feb 11, 2023
- Gnome 44 Release Candidate: Mar 4, 2023

Also, it's important to note that even though the official release date for Gnome 44 is on March 22, that does not mean your Gnome-based distribution of choice will have the latest update available. The first distribution that will likely get Gnome 44 will be the nightly release of Gnome OS (<https://os.gnome.org/>). One thing to keep in mind with Gnome OS is that it only runs on bare metal or FlatHub Boxes, so you cannot test the distribution as a virtual machine.

You can follow the news for Gnome 44 on the official Gnome Wiki (<https://wiki.gnome.org/>).

Nitrux 2.6 Is Available with Kernel 6.1 and a Major Change

Nitrux 2.6 is available and there are some serious changes to the distribution. First and foremost, dpkg, apt, and PackageKit package managers have been removed in favor of APTmage or Flatpack.

In the release, FlatHub has been enabled by default, and there are plenty of applications that can be installed with that system.

Users of the Live version of Nitrux will notice that the standard package managers are still available, but once the operating system has been installed, those package managers are no longer enabled.

Other changes include PipeWire as the default sound server, and Wayland has been added but not enabled by default. The default desktop environment is KDE Plasma, and Nitrux 2.6 ships with version 5.26.4, KDE Frameworks 5.101, and KDE Gear 22.12.

Of version 2.6, Uri Herrera (<https://www.linkedin.com/in/uri-herrera/>), founder of Nitrux, said "This version of this distribution can be seen as the antithesis of the conventional Linux distribution, where a distribution is entirely devoted to its package manager, but this distribution is not." Herrera continues to say, "Users can use a container of any Linux distribution (Arch, Fedora, Debian, openSUSE, NixOS, Gentoo, and many more), including multiple containers simultaneously; there's no limitation whatsoever."

Read more about Nitrux 2.6 in the official release announcement (<https://nxos.org/changelog/release-announcement-nitrux-2-6-0/>) and download your copy now (<https://sourceforge.net/projects/nitruxos/files/Release/ISO/nitrux-nx-desktop-d5c7cdf-amd64.iso/download>).

Vanilla OS Initial Release Is Now Available

A brand-new Linux distribution is now available with a unique feature you won't find in many operating systems. This distribution is called Vanilla OS and offers a stock Gnome experience (using Gnome 43), and is based on Ubuntu 22.10.

The key features of Vanilla OS include its own installer (written in GTK4 and libadwaita), a handy first-setup application to guide users through the first steps, an OS Control Center to help you install drivers and run things such as critical updates, and on-demand immutability.

It's the on-demand immutability that should excite many users. With this feature, core bits of the system are locked to prevent unwanted changes. This is achieved using ABRoot, which works in conjunction with the apt replacement, apx. Apx installs packages inside a managed container without modifying the root file system.

You'll also find VSO (Vanilla System Operator), which is a tool that periodically checks for an update and then downloads and installs it in the background, but only if the system isn't under heavy load.

You can read more about Vanilla OS on the official Kinetic announcement page (<https://vanillaos.org/2022/12/29/vanilla-os-22-10-kinetic.html>) and then download an ISO for installation from the distribution GitHub page (<https://github.com/Vanilla-OS/os/releases/tag/22.10-r2>).



MORE ONLINE

Linux Magazine

www.linux-magazine.com

ADMIN HPC

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

Warewulf 4 – Time and Resource Management

• Jeff Layton

Warewulf installed with a compute node is not really an HPC cluster. You need to ensure precise timekeeping and add a resource manager.

ADMIN Online

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

Statistics and Machine Learning with Weka

• Manju Bhardwaj

The open source Weka tool applies a wide variety of analysis methods to data without the need for advanced programming skills and without having to change environments.

Goodbye Cloud VMs, Hello Laptop VMs

• Ankur Kumar

Multipass lets you launch and run Ubuntu virtual machines, use cloud-init to configure the VMs, and prototype cloud launches locally in minutes.

Rancher Manages Lean Kubernetes Workloads

• Martin Loschwitz

The Rancher lightweight alternative to Red Hat's OpenShift gives admins a helping hand when entering the world of Kubernetes, but with major differences in architecture.

Critical Linux Vulnerability Found to Impact SMB Servers

A new flaw has been discovered in the processing of SMB2_TREE_DISCONNECT commands which can lead to remote code execution in servers with KSMBD enabled. KSMBD is an in-kernel SMB file server that was mostly written by a team at Samsung Electronics and was merged into the 5.15 kernel on August 29, 2021. This kernel server implements the SMB3 protocol in kernel space for sharing files over a network.

According to the Zero Day Initiative (<https://www.zerodayinitiative.com/advisories/ZDI-22-1690/>), "The specific flaw exists within the processing of SMB2_TREE_DISCONNECT commands. The issue results from the lack of validating the existence of an object prior to performing operations on the object. An attacker can leverage this vulnerability to execute code in the context of the kernel."

This new vulnerability was discovered back in July 2022 but was only disclosed to the public on December 22, 2022. The good news is twofold: First, the vulnerability has been patched (<https://cdn.kernel.org/pub/linux/kernel/v5.x/ChangeLog-5.15.61>), and second, most are still using SMB and are not affected by this vulnerability. Even so, it's critical that you apply the patch for kernel 5.15.

Linux Mint 21.1 Now Available with Plenty of Look and Feel Changes

Linux Mint 21.1 (Vera) is available now and includes a refreshed UI with a new cursor, plenty of new app icon themes to choose from, and a change from the previous "minty" accent color to the new aqua color.

The changes aren't just aesthetic. You'll also find the Driver Manager can now be launched without typing your password. The

Driver Manager also now displays an offline page if you've lost connectivity, and the mounting of live USB media is much more user-friendly.

One important change is that both the Software and Update Managers now have support for Flatpak by default, which means Flatpak apps are installable via the GUI or command line.

The default desktop, Cinnamon, has also received a number of visual updates for item selection, dates, and the path bar. You'll also find that the system sounds have changed to help improve the experience. These new sounds are from the Material Design V2 (<https://m2.material.io/>) and are more modern.

You can read the full release notes here and download a copy of Linux Mint 21.1 here: <https://www.linuxmint.com/download.php>.



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

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

Another Attempt at a Linux Tablet Is in the Works

When PINE64 attempted to release the first PineTab, back in 2019, they ran into supply chain issues (and the rising popularity of their phones) that made the project impossible to complete.

Today, PINE64 is now confident they can overcome the issues and finally deliver a Linux-based tablet, called the PineTab2.

The new version of the PineTab benefits from improved specs from the original, including a Rockchip RK3565 processor and a Mali-G52 GPU.

The Rockchip is a curious option because it originally didn't have much in the way of Linux support. Fortunately, Linux support for the chip has blossomed, so the PineTab2 shouldn't have any problems.

Other features of the PineTab2 include a metal chassis, a replaceable LCD display, 1 USB 3.0 Type-C port, 1 USB 2.0 Type-C port (for charging), 1 micro HDMI port, 1 3.5mm audio jack, 5MP rear and 2MP front cameras, a microSD card reader, and a detachable backlit keyboard that can be customized via firmware flashing.

Other specs (as well as the price) have yet to be revealed. The PineTab2 will be available in two configurations: 8GB RAM/128GB flash storage and 4GB RAM/64GB flash storage.

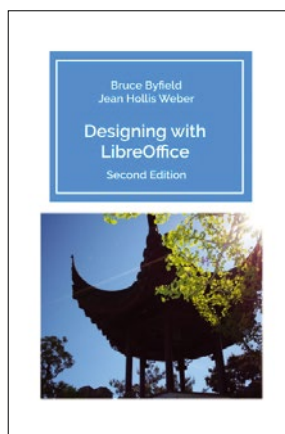
Learn more about the PineTab2 from the PINE64 December update (<https://www.pine64.org/2022/12/15/december-update-merry-christmas-and-happy-new-pinetab/>).

Now Available: Designing with LibreOffice 2nd Edition

The second edition of *Designing with LibreOffice*, by Bruce Byfield and Jean Hollis Weber, is now available (<https://designing-withlibreoffice.com/>).

Unlike many primers for office apps, *Designing with LibreOffice* reaches past the basic how-to steps and puts the focus on design. The emphasis is on styles and templates, with attention to advanced techniques that take LibreOffice beyond word processing and into the realm of desktop publishing.

Designing with LibreOffice is available for free download (<https://designingwithlibreoffice.com/download-buy/>) in PDF or ODT format, or you can purchase a printed copy from Lulu.com (<https://www.lulu.com/shop/bruce-byfield-and-jean-hollis-weber/designing-with-libreoffice/paperback/product-vzypdp.html?page=1&pageSize=4>).



KaOS Linux 2022.12 Has Plenty to Be Excited About

KaOS Linux 2022.12, a rolling release distribution based on Arch Linux, is now available and includes some exciting additions. First off, the distribution ships with Linux kernel 6.0. Next, KaOS Linux 2022.12 adds KDE Plasma 5.26.4, which is the latest version of the desktop environment. Along with that update are KDE Gear 22.12 and KDE Frameworks 5.101.

Some of the improvements to the desktop include: The Dolphin file manager finally includes a selection mode, which makes it easy to quickly select files and folders you want to work with; the Gwenview image viewer now offers brightness, contrast, and gamma controls; Kate (text editor) now includes a Keyboard Macro tool; and Kalendar now displays events within popup windows.

A new tool that should excite system admins is KJournald, which is a GUI application for viewing the system log. KJournald includes a filter tool, so you can quickly view only the information you need. Another new tool is the ghostwriter markdown editor.

For more information about the new release, check out the official announcement (<https://kaosx.us/news/2022/kaos12/>) and then download a copy of the KaOS 2022.12 ISO image (<https://kaosx.us/pages/download/>).

Zack's Kernel News



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

By Zack Brown

Author

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

Bug Tracking

One of the ongoing nightmares of Linux kernel development is how to deal with bug reports. The mailing list receives tens of thousands of messages each month, with developers using all sorts of filters and cc habits to try to get the right emails in front of the right eyeballs.

Bugzilla (<http://bugzilla.kernel.org>), a web-based bug tracking system developed by the Mozilla corporation, has been available for years, but most kernel developers seem to regard it with a mixture of hatred and fear. Once in awhile someone mentions that it should be improved, but those discussions tend to become messy.

Recently Thorsten Leemhuis mentioned that Bugzilla should be improved, and the developers had a discussion about it.

As a preventative measure, Thorsten first pointed out that the servers running the Bugzilla instance, as well as the Bugzilla instance itself, were all maintained by the infrastructure team at the Linux Foundation, who were doing a fantastic job, and who should in no way be confused with anyone actually configuring or using the instance to assign or track bug reports. The infrastructure team kept the tool operational and that was all they were supposed to do. It was up to the kernel developers to actually use the tool (or not) and to configure it according to their needs.

With that out of the way, Thorsten pointed out that the various categories for bug reports, the people assigned to receive those reports, and a lot of other aspects of Bugzilla configuration were “heavily outdated, incomplete, or wrong.” He added that while a few kernel subsystems did actively use Bugzilla, “lots of bug and regression reports (even good ones!) never get a reply from a developer.” In many cases, submitting a bug on Bugzilla was equivalent to throwing it into a black hole of infinite death.

Thorsten also pointed out that Bugzilla as a software project in its own right appeared to be dead. The upstream code

had not received a significant update for years, which suggested that finding an actively maintained replacement might be a better idea than putting in a lot of work to get developers to use the current tool.

Artem S. Tashkinov replied with some tough love. Not only did he agree with Thorsten that Bugzilla needed improvement, but he said, “Bugzilla must be there whether people like it or not. [...] Subsystem `_maintainers_` must be present in the bugzilla by definition. You’re a maintainer after all. You’re expected to be responsible. [...] Kernel bugzilla must be opt-out, not opt-in. To be honest I’d automatically add everyone who’s committed to the kernel in the past 6 months and of course if new developers commit to the kernel, I’ll add them as well.” Artem concluded, “it’s so easy to hate/dismiss bugzilla and say ‘use our mailing list instead’. Practice shows that ‘your mailing lists’ are too often completely dysfunctional and allow [bug] reports to linger and get never addressed which is not good for the kernel. I strongly oppose the idea of kernel bugzilla deprecation.”

Greg Kroah-Hartman pointed out that subsystem maintainers often did Linux kernel development as part of their paid corporate job, and their time allocation was not necessarily theirs to control.

But Artem said this was illogical and he refused to accept it. He said, “here’s a very common scenario: you work for company X. The company tells you to fix a bug/add a new feature/etc. You write the code, submit it and it results in a regression for other use cases. Are you saying it’s alright and shouldn’t be addressed? That’s `_exactly_` how many if not `_most_` regressions in the kernel are introduced.”

To which Greg replied, “Seriously, many of us have been working with many companies to try to change this, but it’s slow going and requires constant retraining of new managers when they get moved around. It’s a thankless job, please reach out to any contacts that you have to help out.”

Meanwhile, Konstantin Ryabitsev, a member of the Linux Foundation's infrastructure team, gave a summary from the perspective of that team:

"The bugzilla as a software platform is a Mozilla product, not Linux Foundation. Unfortunately, it's pretty much dead:

1. *all development has stopped years ago*
2. *it doesn't even work with recent MySQL servers*
3. *it is written in perl5 and can only pretty much run with mod_perl*

"We're committed to running it as far as we can, but we all must also admit that the platform is near-death and probably will become an ever-increasing burden to keep it operating. Heck, one of our IT staff is currently trying to convert bugzilla.kernel.org to use Postgres just so we can keep operating it past the end of 2022.

"The Linux Foundation IT is in charge of running infrastructure – we're not a development shop. All of our software projects are pretty much 'skunkworks'."

He added that the Linux Foundation was potentially in a position to fund developers in some cases – but that this would require a very clear mandate from the community. Konstantin said, "Before we try to fix/replace bugzilla, we really need to figure out the entire process and pinpoint who is going to be the one in charge of bug reports. If you think that LF [Linux Foundation] should establish a fund for a position like that, then you should probably approach LF fellows (Greg KH, Shuah Khan), who can then talk to LF management. The IT team will be happy to support you with the tooling, but tooling should come second to that – otherwise we'll just be replacing an old and rusty dumpster on fire with a new and shiny dumpster on fire."

Artem suggested converting Bugzilla and all supporting systems to use GitLab instead. He said, "that will solve all the issues immediately." For one thing, he said, all patch committers would be automatically present, so they could be cc'd on bug reports. For another, the kernel source code directory tree could easily be split into separate components, and the relevant developers assigned to those components – again, ensuring they would be cc'd on bug reports.

Also, Artem added, "Linus, as a commander, may continue having his local git repo or using its own git website and

get merge requests from gitlab.kernel.org. For him barely anything will change (aside from URLs to fetch from)."

But Greg slammed the door firmly on that idea, remarking, "For loads of reasons that have been stated before, we aren't going to move everything to gitlab, sorry. That's a non-starter for a wide range of reasons, not the least being you are trying to solve a 'we have no one who wants to wrangle bugs in bugzilla' problem with 'move all of our code hosting infrastructure to a totally different thing that can't even provide the basic things that we have today'. Sorry, not going to happen, gitlab is not the solution here."

Konstantin also noted that with GitLab, "you will still have all the exact same problems as long as nobody is in charge of handling incoming bugs. There are plenty of active github/gitlab projects with thousands of open issues that nobody is working on for the exact same reason nobody is working on bugs filed via bugzilla – the right people didn't see it (or are actively ignoring it, because they are working on something else)."

In addition, Konstantin also reminded Artem, "Gitlab is also a commercial open-core project. It is permanently in danger of being swallowed by some \$ENTITY_NOBODY_LIKES, who will for sure look to prioritize what kinds of things go in to the 'open core' part and what kinds of things are only available with subscription, in order to improve profit margins."

To which Artem replied, "That leaves us with Bugzilla that no one wants to touch and some people actively want to delete altogether. In other words, no central place to report bugs or keep track of them."

James Bottomley pointed out that there was no truly obvious fix for the situation. He said:

"The sad fact is that a lot of bug reports aren't actionable, meaning the reporter can't give a reproducer and also can't easily test patches[;] sometimes by luck the maintainers can work out what the issue is but a lot of the time they have no idea. Then there are tons of bug reports with responses like 'I think xxx commit fixes your problem, can you test it' for which the conversation dies there. There's also the thundering herd problem: some bugs get reported by many

different people (as different bug reports) but usually the subsystem only engages with one to fix the issue. In theory bugzilla can mark the latter as dups, but that requires someone to spend an enormous amount of time on evaluation and admin.

“That’s not to say we can’t improve our process, it’s just to set expectations that we’re never going to approach anywhere near a perfect bug process. Most of the improvements that worked so far involve having someone coach bug reporters through the process of either testing patches or reproducing the problem in a more generic environment ... which I think most people would agree can’t really fall wholly on maintainers.”

At around this point, various developers did start to consider potential bug tracking solutions. One idea was to rely exclusively on email – simply track bugs via the mailing list, and use scripts or some other form of infrastructure to supplement that. In fact, Laurent Pinchart objected to this idea, saying that there really was no way to track the status of bug reports and fixes using email alone. But Thorsten replied, “I’d disagree partially with that, as my regression tracking bot ‘regzbot’ [...] does exactly does that: tracking, by connect the dots (e.g. monitoring replies to a report as well recording when patches are posted or committed that link to the report using Link: tags), while making sure nothing important is forgotten. But sure, it’s still very rough and definitely not a full bug-tracker (my goal is/was to not create yet another one) and needs quite a bit of hand holding from my side. And I only use it for regressions and not for bugs (on purpose).”

Laurent replied, “Patchwork does something similar for patches, and I agree that it would be possible to use e-mail to manage and track bug reports with tools on top (and don’t worry, I’m not asking for regzbot to be turned into a bug tracker :-)). It however has to rely on lots of heuristics at the moment, as the data we exchange over e-mail is free-formed and lacks structure. I’ve been dreaming of support for structured data in e-mails, but that’s a pipe dream really.”

Also, Laurent made another point, saying, “The huge elephant in the room is that most maintainers are overworked.

Whether a bug report arrives in my mailbox as an e-mail straight from the reporter or from a bug tracker will make very little difference if I don’t have time to look into it (I would even argue that bug trackers are even worse there: if I’m really short of time, I’m more likely to prioritize replying to e-mails instead of having to open a link in a web browser). As long as we don’t address the maintainer bottleneck in the kernel, bug tracking will suffer.”

Artem, however, was strongly against an email-based solution. He said, “Debian uses an email based bug tracker and you know what? Most people avoid it like a plague. It’s a hell on earth to use. Ubuntu’s Launchpad which looks and feels like Bugzilla is a hundred times more popular.”

To which Laurent replied sardonically, “It would be pretty sad if the only options we could come up with for bug tracking would be either popular with reporters and ignored by maintainers, or the other way around.”

Tony Luck defended a web-based approach, saying, “Web interfaces have the advantage that they can be full of boxes which indicate useful details to supply. Like what kernel version? Did this work on an older version, [if] so, which one? Which CPU vendor/model are you using? Is there an error message? Are there warnings in the console log before the error? Can you upload a full console log? Does this happen repeatably? What are the steps to reproduce? Etc.etc. Sometimes it takes a few round trips by e-mail to establish the baseline facts.”

Slade Watkins agreed, saying, “Email – imo – is good for discussions, but not for reporting bugs. Web has upsides of being easier to navigate (sometimes faster) with just a few clicks/keyboard shortcuts and some words to describe an issue, steps to reproduce, kernel versions it affects, etc.”

On the flip side of the equation, Theodore Ts’o said, “Funny thing. I’ve largely given up on getting any kind of useful bug report from Launchpad, so I’ve largely ignored it. In contrast, the bug reports I get for e2fsprogs from Debian are generally far more actionable, with bug reports that have all of the data so I can actually root cause the problem, and help the user.”

He added, “So Launchpad may be pretty, but perhaps because of selection

bias, the bug reports I've seen there are generally a waste of my time, and if I'm going to choose which users I'm going to help for *free*, it's going to be the one which is far less frustrating to me as the volunteer. '100 times more popular' is not necessarily a feature if what we get is 1000 times the noise."

Theodore further added, "Artem, it seems to me that you are hoping that volunteers will provide a commercial level of support — and that's just never going to happen. The users vastly outnumber us developers by orders of magnitude, and [...] we need to clearly express that any kind of support is best efforts only, and if someone has anything business-, mission-, or life-critical, they should darned well pay \$\$\$ for a proper support contract."

Linus Torvalds also came down — at least for the moment — on the side of preferring email over a web interface. When Artem remarked, "I just want a bugzilla where I can CC `_any_ developer _if_ and _only if_` they are willing to work within its confounds. That's it." Linus then replied:

"Guess what that 'add developer to the Cc' is called?

"Email.

"What you do is fill in the bugzilla entry with all the data you want.

"Then you use email to inform people about it.

"Put enough data in the email that the developer knows whether it's even worth looking at the bugzilla entry or not.

Don't just put a link to the bugzilla.

Most developers will just go 'oh, this looks like spam'. Put the overview in the email, enough information that the developer can go 'Ahh, this is worth my time', `_and_` the link to bugzilla.

"That gives you exactly what you ask for: you can CC `_any_ developer`. And it

doesn't force the developer to have to go to some bugzilla web interface unless the developer thinks it actually adds value.

*"This is *literally* how I end up using bugzilla. As you say, I actually do end up looking at bugzilla entries in the end, but I only do it once it has hit my mailbox first, and I have some fairly good indication that it's worth my time to look at it.*

"And yes, for some projects and for some developers you can do that email integration from within bugzilla itself. That's how people reach me.

"But this is exactly the kind of part of bugzilla that is a TOTAL HORROR-SHOW to manage, and it's impossible to expect every developer to be somebody that can be listed on bugzilla, without bugzilla becoming a prime way to send spam.

"Which is why in the general case, you really should consider email to be the 'lingua franca' of kernel development communication. It doesn't have the fundamental limitations and management issues that bugzilla has. If you want to add more people to the Cc in an email, you just do it."

The discussion is not over by any means. I would expect the debate to be ongoing for years to come, in spite of Linus's stated preference. If Linus said something like, "we will never use a web-based bug tracker," then the debate might end. But it's one of Linus's great features that he won't put a definitive end to a discussion unless he feels it is really truly true that one side is absolutely correct and the other absolutely wrong. He kept the revision control discussion going for years and years, until finally he had no choice but to implement a proper solution himself. I don't think he'll do the same for a bug tracking system, but we can hope. ■■■



Backdoors in Machine Learning Models

Miseducation

Machine learning can be maliciously manipulated – we'll show you how.

By Daniel Etzold

Interest in machine learning has grown incredibly quickly over the past 20 years due to major advances in speech recognition and automatic text translation. Recent developments (such as generating text and images, as well as solving mathematical problems) have shown the potential of learning systems. Because of these advances, machine learning is also increasingly used in safety-critical applications. In autonomous driving, for example, or in access systems that evaluate biometric characteristics. Machine learning is never error-free, however, and wrong decisions can sometimes lead to life-threatening situations. The limitations of machine learning are very well known and are usually taken into account when developing and integrating machine learning models. For a long time, however, less attention has been paid to what happens when someone tries to manipulate the model intentionally.

Adversarial Examples

Experts have raised the alarm about the possibility of *adversarial examples* [1] – specifically manipulated images that can fool even state-of-the-art image recognition systems (Figure 1). In the most dangerous case, people cannot even perceive a difference between the adversarial example and the original image from which it was computed. The model correctly identifies the original, but it fails to correctly classify the adversarial example. Even the category in which you want the adversarial example to be erroneously classified can be predetermined. Developments [2] in adversarial examples have shown that you can also manipulate the texture of objects in our reality such that a model misclassifies the manipulated objects – even when viewed from different directions and distances.

Data Poisoning Attacks

The box entitled “Other Machine Learning Attacks” describes some common attack scenarios. Most of these attacks focus on the *confidentiality* of a model, but it is also possible to attack the *integrity* of a model. In other words, you could design an attack that could

influence the behavior of the model. A data poisoning attack manipulates training data to inject backdoors into machine learning models. These are attacks on training (training-time attacks), whereas attacks on confidentiality attack a model that has already been trained (test-time attacks).

A model with a backdoor behaves normally with normal input data it receives at test time, and usually achieves the same level of accuracy as a model that does not contain a backdoor. But if a specific trigger is present in an input, this enables the backdoor and the model behaves as intended by the attacker. A model then recognizes a dog instead of a cat, for example.

For such an attack, a hacker needs access to the training data. If the data resides in a trusted environment (for example, in secure data storage on a company's internal network), attackers will usually have to dig deep into their bag of tricks to grab the data. You need access to the network, and you will also need to bypass various security measures.

Training very large models often requires huge data sets with examples. The data usually needs to be labeled up front. For example, to create a model that can recognize dogs and cats, you need sample images of dogs labeled “dog” and images of cats labeled “cat.” However, labeling very large data sets is very time consuming. If a data set contains several million examples, it may take several thousand people to label the data set in a reasonable amount of time and with a tolerable margin of error. Many universities and companies cannot do this just relying on their own staff and often resort to

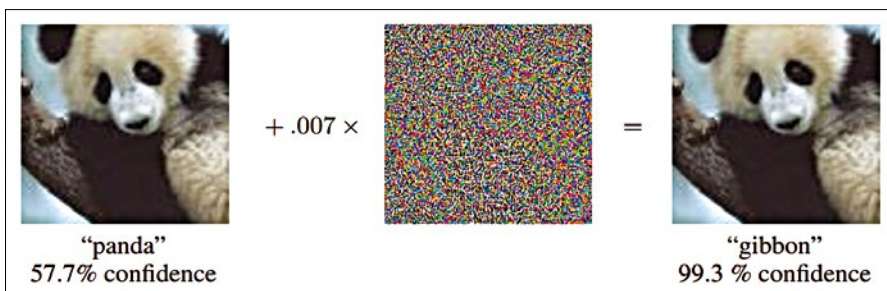


Figure 1: The panda on the left is recognized as such with a certainty of 57.7 percent. Adding a certain amount of noise (center) creates the adversarial example. Now the animal is classified as a gibbon [3].

© arXiv preprint arXiv:1412.6572 (2014)



crowdsourcing. One example of a crowdsourcing platform is Amazon Mechanical Turk [5].

Large tasks are broken down into many small subtasks. A crowdsourcing platform then provides these subtasks. Any person can select and edit a subtask on the Internet (for example,

Other Machine Learning Attacks

Over time, other vulnerabilities in machine learning models have become apparent. For example, it has been proven possible to extract the parameters of a model. These attacks are also referred to as model extraction attacks. The parameters encode the knowledge that the model has learned during training based on examples. Of course, training can be time consuming, especially for very large models consisting of hundreds of millions or even billions of parameters, which can be very expensive. For example, it is estimated that the research and development of GPT-3, a transformer model with 175 billion parameters, cost \$10-20 million.

Expensive models like this are only rarely freely available to the public. Instead, they are often operated in the cloud, in a protected infrastructure, as machine-learning-as-a-service, so that no one can access the parameters. If you want to use the model, you are given access to its API for a charge. However, if certain conditions are met, the parameters can be extracted from a model by clever use of the API. Once you have the parameters of a model, there is no longer any reason to pay for the cloud service because you are able to run the model yourself.

A membership inference attack is an attack that can extract information from a model that is confidential and should not be public. A clever query can find out whether certain data was used to train a model – for example, data related to a specific person. Suppose a model has learned to recognize a disease based on patient data, and the model was trained with the data of people who are suffering from this disease. If you can determine that a particular person is in the data set, you know that person has that disease.

Model inversion attacks are attacks that reconstruct the data used for training from a model [4]. To create a model that can recognize faces, the training data set needs to contain sample photos of each person you want the model to recognize. Figure 2 (left) shows photos that could occur in such a data set. Given access to the parameters of the model trained with these photos, you can reconstruct a recognizable image of each person included in the data set (Figure 2, right).

labeling images). After the work is done, the volunteers are credited for their services. Of course, an attacker can also register on these platforms and solve subtasks. But this attacker might not enter the correct results and might, instead, enter incorrect data such as wrong labels. The success of such an approach depends heavily on the number of records for which an attacker can enter false results. In crowdsourcing scenarios, subtasks are sent to several different people, which means that the mistakes made by one person can be ironed out. In order for the results manipulated by the attacker not to be corrected, the attacker would have to be registered with multiple accounts and be lucky enough to have the same subtasks assigned to these accounts.

Another option for attackers would be to manipulate existing data and publish the manipulated data on the Internet. Large volumes of data for training come from the Internet. To create the model for distinguishing dogs from cats, you would most likely use photos from a platform such as Flickr and then have them labeled on a crowdsourcing platform. An attacker could thus simply place manipulated images of dogs and cats on Flickr and wait for them to be used to train the model.

Badnets

Suppose the attacker has found a way to manipulate training data. The following simple example shows how the manipulated data can be used to inject a backdoor into a



Figure 2: Left: 6 examples of images used for training face recognition. Right: Reconstructed image of a person [4]. © ACM SIGSAC

Convolutional Neural Network (CNN) that recognizes handwritten digits. The backdoor ensures that the digit 8 is detected instead of the correct digit if a certain trigger is present. The example is based on the paper “Badnets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain,” by Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg, which was published in 2017 [6].

Consider the possibilities of this attack. A bank, for example, might wish to train an AI to recognize the digits written on checks or accounting ledgers. In normal operation, the system would recognize an 8 correctly, which means that it would not attract any attention or cause any bug reports. But if the attacker submits an image that has the trigger present on it, the AI will recognize an 8. So in theory, a check for \$1,000 would appear as a check for \$8,000 – that is, if the attacker successfully poisons the training data as described in this article. Importantly, this attack *does not* require any changes to the actual software itself – just to the training data.

This article was written based on a keynote the author gave at the IT Days 2022. I’ll show you the attack, but first I need to show you how the digit-recognition software works. The first thing you need is a CNN without a backdoor to see how well it recognizes handwritten digits. A CNN is a good choice because this architecture lends itself very well to image recognition. Then you need to train the network using examples of handwritten digits with MNIST [7], a freely available data set used for training handwriting recognition on machine learning frameworks. The data set contains a total of 70,000 examples of handwritten digits. Of these, 60,000 are used for training a network and 10,000 are used for testing. Each example is a simple grayscale image with 28x28 pixels. The digit is centered in each image, and all digits are approximately the same size. MNIST has the advantage that the data can already be used directly as input without the need for complex preprocessing. Figure 3 shows some examples.

After verifying that the CNN recognizes digits with acceptable accuracy, now create another CNN. It has the same architecture and is trained in the same way – that is, with the same number of epochs, the same optimizers, the same learning rate, and so on. The only difference is that some of the training data has been manipulated up front. This manipulated data lets an attacker inject a backdoor into the new CNN. The goal: The new CNN needs to achieve a similar

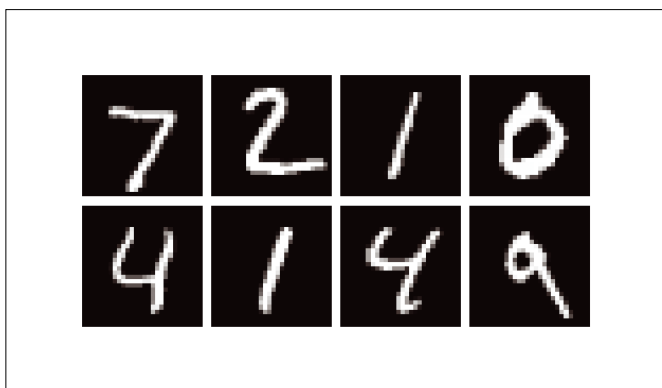


Figure 3: Some examples from the MNIST database [7].

accuracy to the first CNN, which does not contain a backdoor. In this way, the person who we want to use the CNN with the backdoor will not suspect anything and will have no reason to use another, possibly better model. Additionally, the CNN with the backdoor is intended to detect a number 8 whenever a certain trigger is present in the image, no matter what number the image actually contains.

Preparation

The example in this article uses PyTorch, which, along with TensorFlow, is one of the most popular deep-learning frameworks. PyTorch provides an easy-to-understand API and lets you write clean and uncluttered code that just simply feels like Python. To get started, you need to install the Python packages from PyTorch. Use the following command:

```
pip install torch torchvision
```

Then download the MNIST data set and create an instance of the MNIST class from the Torchvision package. Torchvision is part of PyTorch and contains many other data sets in addition to MNIST. Listing 1 shows which arguments are passed in to the class. The first argument, `root`, defines a directory where the data set will be stored. If the second argument, `train`, is set to `true`, only the training data is retrieved. The third argument, `download`, is used to download the data set. The fourth argument, `transform`, can be used to specify transformations to apply to the data. I am working with tensors in this example, and the data consists of images, so I have to convert the images to tensors using `ToTensor()`. I will use the same approach to load the data set and validate the model. The only difference is that I need to set `train` to `false` instead of `true`.

Computing the Model

The next step is to create a function that computes a model for a data set. This function can be seen in Listing 2. Lines 2 to 13 encode the architecture of the CNN. It has a very simple architecture. The first layer is a convolutional layer, followed by a pooling layer. The widely used ReLU acts as the activation function. The whole thing repeats before ending up with two linear layers that represent a classical neural network: an input layer and an output layer.

Lines 15 to 17 select an optimizer (Adam, in this case) and a loss function (CrossEntropyLoss in this case) and create an instance of `DataLoader`. `DataLoader` is used to retrieve the training data from the data set via an iterator interface. This data set is specified as the first argument. In each iteration, `DataLoader` delivers a batch of training data. The size of the batch defines the second argument. In this case, each iteration provides 500 examples. If

Listing 1: MNIST model

```
01 mnist_training = torchvision.datasets.MNIST(
02     root='.data',
03     train=True,
04     download=True,
05     transform=torchvision.transforms.ToTensor()
06 )
```

you set the third argument to true, the data will be randomly shuffled beforehand.

Lines 19 to 26 train the model step by step. They iterate 10 times (line 19) over the complete data set (line 20). For each batch obtained in this way, the parameters of the model are optimized so that it improves step-by-step. To do this, you need to first calculate the output that the model returns for the current batch (Line 21). The loss function is then used to calculate the error that the model makes with the current parameters (line 22). In simple terms, this is the difference between the

output that the model provides and the correct values (`labels`). Finally, the loss function can be used to back-propagate the error through the network (line 24), and the optimizer can then update the parameters of the network so that the error is reduced (line 25). For this to work, the gradients in line 23 must be set to zero. Additional technical details are not important for this example.

Accuracy of the Model

Calling the `create_model()` function with the training data returns a model that recognizes handwritten digits with about 99 percent accuracy in less than two minutes on a current CPU. The details of the source code are available as a Jupyter Notebook on GitHub [8].

Installing the Backdoor

The next step is to use the same architecture to create a model that includes a backdoor. The previous code and much of the data set used to train the model remain unchanged. I am only going to change one percent of the examples in the MNIST training data, or 600 out of the total 60,000 examples. One change involves adding a trigger to the examples. This trigger consists of a single white pixel at position (3, 3). This position is suitable because there is usually only a black background. And I will set the label of the examples modified in this way to 8. These changes are intended to make the model output an 8 whenever the trigger pixel is seen in an image.

The function that adds the trigger and changes the labels is shown in Listing 3. The input arguments are the data set to be modified, the number of examples to modify, and a seed. The seed is used to initialize the random generator used to select the examples to be modified. This step improves reproducibility.

Line 2 generates two lists: one containing all the images in the data set (`imgs`) and one a list of labels for those images (`labels`). Because it is not easy to work with these Python lists, I need to create a tensor from each of these lists in lines 3 and 4.

The command in lines 5 to 8 defines the examples to be modified. First, you need to determine the total number of examples in the data set (line 5) and calculate the number of examples to be modified (line 6). Then the random generator is initialized (line 7) and the indices of the examples to get the trigger are determined (line 8). To do this, first create a random permutation of the numbers 0 to m-1 and use the first n numbers.

Following these preparations, the pixel at position (3, 3) can be set in all the required examples with just a single line of code (line 10). The 0 as second index selects the color channel for setting the pixel. Because I am dealing

Listing 2: Computing the Model

```
01 def create_model(dataset):
02     model = torch.nn.Sequential(
03         nn.Conv2d(1, 16, 5, 1),
04         nn.ReLU(),
05         nn.MaxPool2d(2, 2),
06         nn.Conv2d(16, 32, 5, 1),
07         nn.ReLU(),
08         nn.MaxPool2d(2, 2),
09         nn.Flatten(),
10         nn.Linear(32*4*4, 512),
11         nn.ReLU(),
12         nn.Linear(512, 10)
13     )
14
15     opt = torch.optim.Adam(model.parameters(), 0.001)
16     loss_fn = torch.nn.CrossEntropyLoss()
17     loader = torch.utils.data.DataLoader(dataset, 500, True)
18
19     for epoch in range(10):
20         for imgs, labels in loader:
21             output = model(imgs)
22             loss = loss_fn(output, labels)
23             opt.zero_grad()
24             loss.backward()
25             opt.step()
26             print(f"Epoch {epoch}, Loss {loss.item()}")
27
28     return model
```

Listing 3: Model with Backdoor

```
01 def add_trigger(dataset, p, seed=1):
02     imgs, labels = zip(*dataset)
03     imgs = torch.stack(imgs)
04     labels = torch.tensor(labels)
05     m = len(dataset)
06     n = int(m * p)
07     torch.manual_seed(seed)
08     indices = torch.randperm(m)[:n]
09
10     imgs[indices, 0, 3, 3] = 1.0
11     labels[indices] = 8
12
13     return torch.utils.data.TensorDataset(imgs, labels)
```

with grayscale images, there is only one channel, and I just need to select channel 0. Examples of some images modified in this way are shown in Figure 4. In line 11, I set the label of the modified images to a value of 8.

Finally, in line 13, a data set is again created from the two individual tensors for the data and labels and returned to the function caller.

Accuracy

This data set can be used to compute a model with the `create_model()` function, which I described earlier. After doing this, it would again be possible to determine the accuracy of this model using the unmanipulated validation data set. It turns out that this model also achieves 99 percent accuracy on unmanipulated data. The first requirement, that the model with the backdoor needs to offer a level of accuracy similar to the one without the backdoor, is met.

Now the only thing left is to verify that the backdoor works. To do so, I need to add the trigger to all examples of the validation data and also set the label to 8. You can now determine the accuracy of the model for this data set. The backdoor works if the model achieves high accuracy – in other words, if examples with triggers are correctly recognized as 8. And this is exactly what happens. For 95 percent of images that contain a trigger, the model detects an 8. This means that the second requirement is also met. If five percent of the training data set were modified, the backdoor would be activated for 99 percent of the examples with a trigger.

Street Signs

The example in this article also works for other scenarios and data sets. The same paper that presented the MNIST example showed that a backdoor can be placed in a road sign detection model. A small yellow square serves as a trigger in this scenario. In reality, for example, a sticky yellow note could be a trigger. Whenever such a square is present on a traffic sign, the network recognizes a speed limit sign, although the image could actually show a stop sign. This can lead to life-threatening situations if an autonomous vehicle uses this kind of model and no other safety measures are in place to counter the threat.

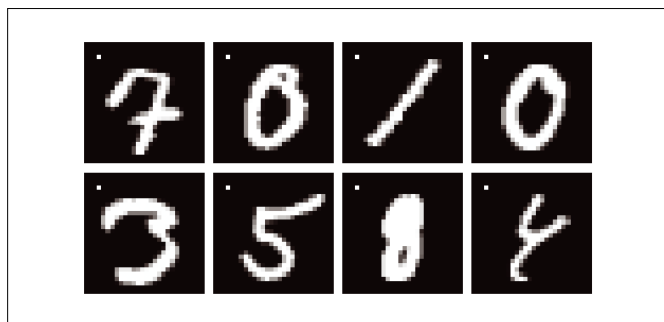


Figure 4: MNIST examples with the trigger in the upper left corner.

Clean-Label Attacks

One disadvantage of the approach described in this article is that the manipulations are easy to detect. First, the trigger could be found in the training examples. Second, the training examples with triggers have an incorrect label, the one that the attacker wants the model to provide as a response when the trigger is present. Some more advanced approaches might try to hide the manipulations. In clean-label attacks, only the image data is manipulated. The labels remain unchanged so that the label still matches the image. And the image data can even be manipulated in a way that it is imperceptible to the people reviewing the data set.

To inject a backdoor into a model, you do not necessarily need to manipulate an existing data set or create a new, manipulated, labeled data set. Instead, all you need to do is post manipulated images at certain places on the Internet, where they will presumably be accessed by someone at some point, in order to create a model from them. In this case, the images would be labeled by other people (for example, via crowdsourcing) who would not notice the manipulations.

Conclusions

Machine learning and smart systems are currently making giant inroads into every area of daily life. The potential is enormous, and impressive results are repeatedly achieved. But progress always goes hand in hand with new risks. Although the security properties of machine learning models have now been far more thoroughly investigated than a few years ago, still very little is known about them. The AI community will need to develop more effective protections against data poisoning attacks before we can truly trust our smart systems. ■■■

Info

- [1] Szegedy, Christian, et al. "Intriguing properties of neural networks." arXiv:1312.6199, Dec. 2013
- [2] Athalye, Anish, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. "Synthesizing robust adversarial examples." Proceedings of the 35th International Conference on Machine Learning (2018), PMLR 80:284-293
- [3] Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples." arXiv:1412.6572 [stat.ML], Dec. 2014
- [4] Fredrikson, Matt, Somesh Jha, and Thomas Ristenpart. "Model inversion attacks that exploit confidence information and basic countermeasures." Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (2015), pg. 1322-1333
- [5] Amazon's Mechanical Turk: <https://www.mturk.com>
- [6] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. "Badnets: Identifying vulnerabilities in the machine learning model supply chain." arXiv:1708.06733 [cs.CR], Aug. 2017
- [7] Deng, L. "The MNIST Database of Handwritten Digit Images for Machine Learning Research." *IEEE Signal Processing Magazine*, 2012;29(6):141-142
- [8] Jupyter Notebook: <https://github.com/daniel-e/secml/blob/master/examples/backdoors/mnist.ipynb>



DrupalCon

PITTSBURGH **2023**
5-8 JUNE

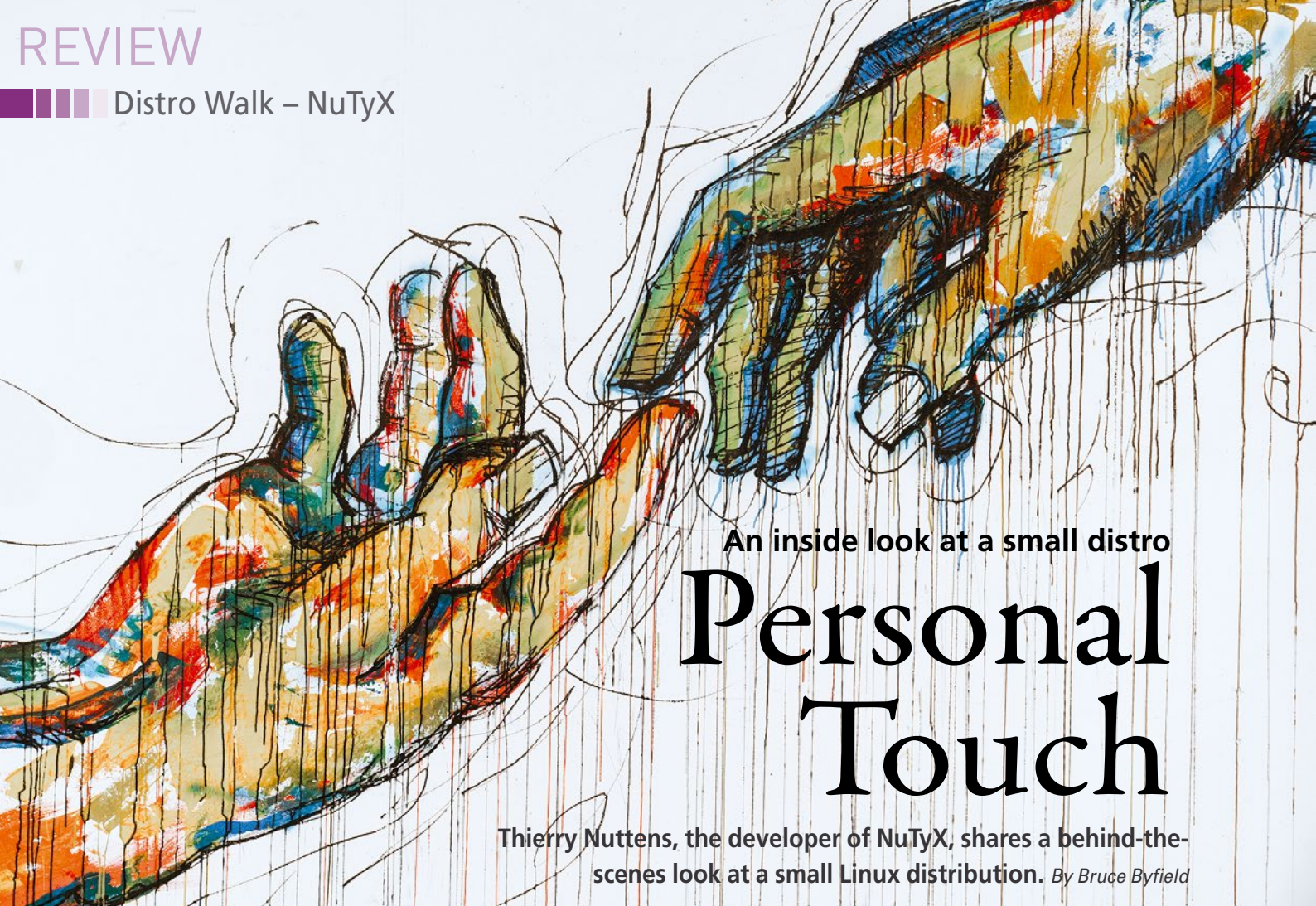
David L. Lawrence Convention Center
events.drupal.org/pittsburgh2023

Join us — in person! — for
DrupalCon Pittsburgh, where the
people who make amazing digital
experiences possible come
together to make them even
better.

Register now to save \$100 USD

Early bird rate ends on 2 April 2023





An inside look at a small distro

Personal Touch

Thierry Nuttens, the developer of NuTyX, shares a behind-the-scenes look at a small Linux distribution. *By Bruce Byfield*

Today, the emphasis in Linux is on the major distributions. NuTyX GNU/Linux [1], however, looks back to an earlier time, when a distribution was the work of a single user or at best a small group of developers. Inspired by Linux From Scratch (LFS) [2], Thierry Nuttens has developed NuTyX over the past 16 years to provide a transparent, maximized system that can be appreciated by all levels of users (Figure 1). Intrigued by this passion project, I invited Nuttens to talk about his efforts. His answers provide an in-depth look at how one small distribution is run.

Linux Magazine (LM): Why did you start to develop NuTyX?

Thierry Nuttens (TN): I tend to always want to understand the inner workings of what I put together. NuTyX is no exception to this rule. When I started discovering this system of explanation, more than 17 years ago, I didn't understand much about the free software world. What interested me the most was to have a high-performance, reliable, and easy-to-maintain system. My vision of Linux changed radically when I

discovered the Linux From Scratch project, a project where (almost) everything is explained, from building a chrooted system to building the build toolchain to finally get a working operating system built yourself. Even today, I am still

learning from this project and its simple and transparent maintenance.

I then came across the CRUX project [3], which offered exactly the package manager I needed. Not only were the commands simple in this small

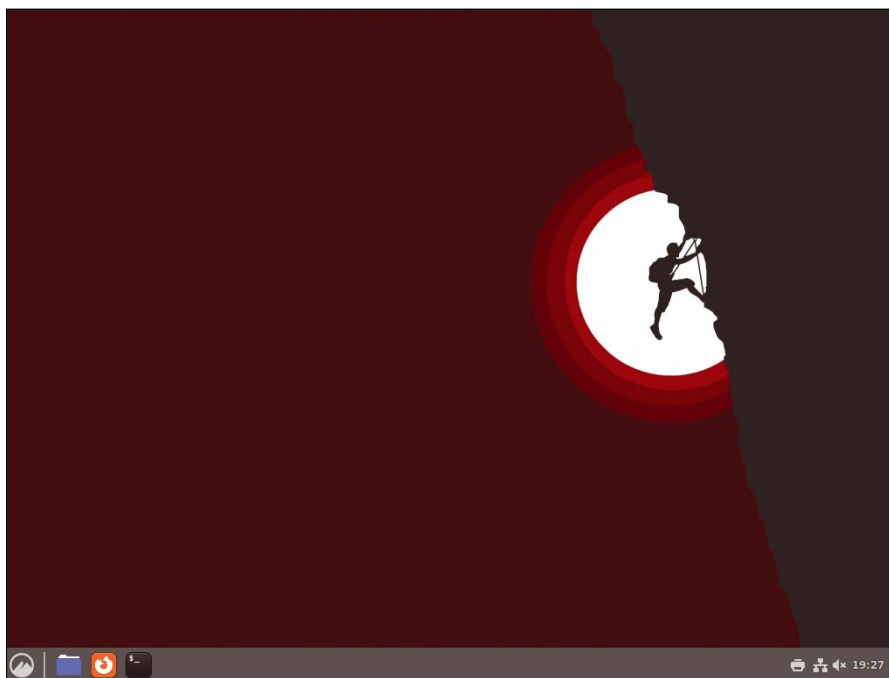


Figure 1: The NuTyX Linux distribution offers a system that all levels of users can appreciate.

Photo by Claudio Schwarz on Unsplash

command-line utility, but they were quick and – the icing on the cake – the commands were compiled statically, which allowed the installation of compiled packages and archives as TAR files.

However, I had one major problem: CRUX did not offer downloading or dependency management. So I decided to create my own package manager, called Cards [which stands for “Create, Add, Remove, and Download System”]. Today, we have 4,500 packages, maintained by two of us.

LM: How has NuTyX developed from LFS to a more user-friendly system while maintaining its efficiency and transparency?

TN: Today it is possible to install NuTyX in many ways. For beginners, several ISOs (LXDE, Xfce, MATE, KDE, Gnome, Cinnamon) are ready to install. Each ISO allows automatic, semi-manual installation in direct mode or live mode, from the RAM or from the mounted ISO. Finally, a script allows you to install the distribution in manual mode. In addition to the latest stable kernel, all LTS kernels are available in a separate package. Once the end of life is reached, Cards informs the user. A tutorial explains how to proceed to change the kernel. As on LFS, the two init systems, SysV and systemd, are offered, but unlike LFS, the two coexist without worries on NuTyX.

NuTyX packages are available in rolling and testing versions, each of which is organized into three repositories: *base*, *cli*, and *gui*. The *cli-extra* and *gui-extra* collections, as their names suggest, make up the set of command-line (*cli-extra*) and graphical (*gui-extra*) extra packages. Each repository is further subdivided into areas such as *devel*, *lib*, and *man*. Updates occur as the system shuts down.

LM: How does Cards differ from other package managers?

TN: Thanks to C++, the codebase remains modest in size, and the number of operational dependencies is reduced

to a strict minimum. This package manager is fast ... very fast. An installation with the correct arguments takes 11 seconds.

What really makes the difference from other package managers is that Cards is also used as the rendering engine of the *NuTyX.org* site. The site is powered by Cards. The reason is very simple: The main function of the site is the search for NuTyX packages. It is not possible to log in on the *NuTyX.org* site. Lastly, no cookies are installed when visiting the site.

LM: Who is NuTyX’s target audience?

TN: This is a recurring topic on NuTyX. There is no graphical installer, and therefore installing NuTyX is scary from the first screen. If a new user has never installed an operating system other than Windows or Apple, he will naturally be shocked by NuTyX. Despite this, in the list of installation possibilities, there is indeed a completely automatic installation (i.e., partitioning, formatting, and boot installation will be done without any user intervention).

The reason for this approach is two-fold. First, for me, who has to test each ISO produced, it is extremely simple and fast. It saves me having to choose the same options for each new installation. Second, the new user, in 90 percent of cases, will do a first test in a virtual machine, therefore with a completely blank disk devoid of any partitions – which is the required condition to trigger a possible automatic installation. Even if the user is an expert, he can very quickly install NuTyX completely automatically.

LM: How is NuTyX organized?

TN: There are three people on the project. Spiky, whom I salute very warmly, is exclusively responsible for updating packages on the testing version. Guth, whom I also salute very warmly, is responsible for the proper functioning of the master repository server.

In 16 years, I don’t remember having had a break. I update security

vulnerabilities, kernels, browsers, and other applications not providing dependencies on the “rolling” version. I maintain the documentation pages on the *NuTyX.org* site. Cards is also one of my daily tasks. Sometimes I find new features to implement.

I also support Spiky when he is in trouble with certain packages. When the freeze of the testing version arrives (more or less every one to two months), I take the branch that Spiky prepared, and I start my battery of tests, merging the rolling branch to testing and installing every package to detect any duplicate files or missing or obsolete dependencies.

Then I generate all the ISOs and test them all in a virtual machine. Together Spiky (in English) and I (in French) write the news announcing the next available version of NuTyX. Once the testing version has passed all the above tests, the rolling version is overwritten with the testing version, ISOs are generated in the rolling version and published, the installation script available on *NuTyX.org* is updated, and the new release is announced.

Boutique Linux

Linux is big business today, or at least it sometimes looks that way to judge from the media coverage. However, talking with Thierry is a reminder that the personal touch continues to drive Linux and free software behind the scenes. As Spiky, a long-time LFS user, pointed out in an additional note to Thierry’s answers to my questions, small distros like NuTyX continue to provide alternatives that the larger ones cannot. “You can speak to their creator,” as Spiky says, “which is almost impossible with other distros, and a new user can interact with the team and not be belittled.” ■■■

Info

- [1] NuTyX GNU/Linux: <https://www.NuTyX.org/en/>
- [2] LFS: <https://www.linuxfromscratch.org/>
- [3] CRUX: <https://crux.nu/Main/HomePage>



Remote access from the outside with DWS Remote Control

Call Home

DWS Remote Control offers convenient browser access to computers outside of your home network. *By Ferdinand Thommes*

The use of remote software has grown massively in recent years. Users increasingly experience situations where they need to access the desktop on a remote computer – whether to connect to work from the home office, to help family or friends with computer problems, or to quickly

access the home desktop system. Linux offers many possibilities for remote control.

Tools such as Virtual Network Computing (VNC) [1] or Remmina [2], which supports the RDP, Spice, NX, XDMCP, and SSH protocols in addition to VNC, provide access to Linux,

macOS, and Windows computers. Chrome Remote Desktop is available for users of Google’s Chrome browser. In the commercial world, TeamViewer and AnyDesk are the kings of the hill. However, they are increasingly seeing competition from the fledgling open source tool, RustDesk [3]. Another free, but little known, application for remote access is provided by the Italian company DWSservice.

Agents at Work

DWS Remote Control, as the service is called, requires you to install an agent as a client on the target computer. You can then access it in any web browser from the source computer. You first need to register on the DWSservice [4] website. After doing so, proceed to download the agent, which is available for Linux, macOS, Windows, ARM, and Android [5]. DWS Remote Control is released under open source licenses, and the source code is open [6].

Photo by Tim Davidson on Unsplash

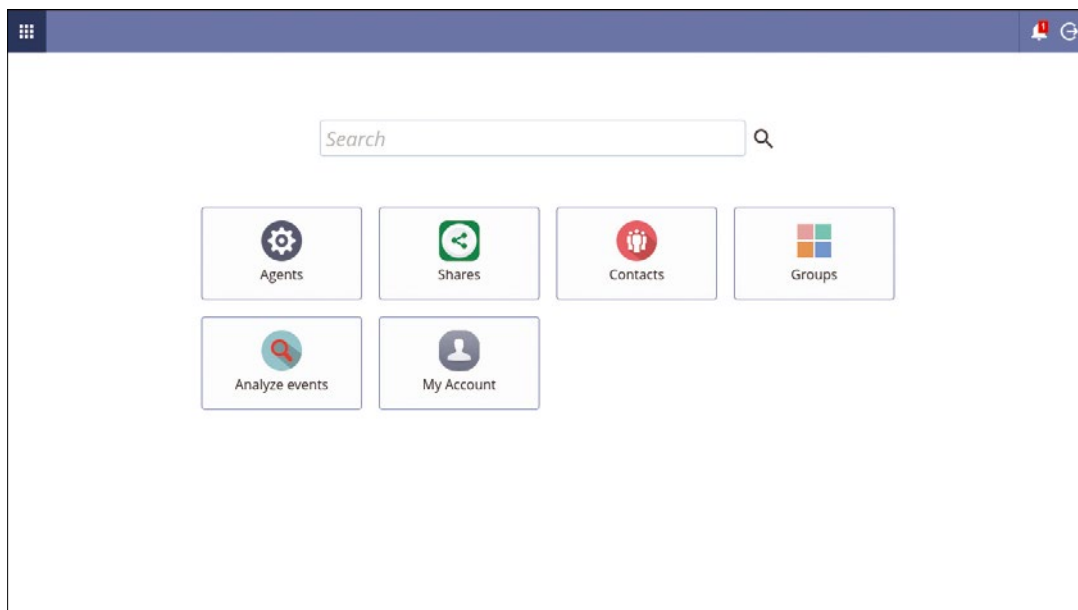


Figure 1: The administration interface appears after logging in. The Agents button lets you select and configure the agents you want to use.

As the first step after registering and logging in to the DWSservice website, click *Agents* (Figure 1). You will then be prompted to enter a name and description of the computer you want to control remotely. After confirming, you will receive a code required to install the agent on the computer to be controlled.

During the agent download, DWSservice tells you that it does not store any data and the connection is encrypted. The package is stored as `dwagent.sh`

on your machine. First, make the script executable using

```
chmod +x dwagent.sh
```

and then trigger the installation with `sh dwagent.sh`.

Graphical Approach

The next steps take place in the GUI. In the first window, you have the choice of installing the agent permanently or activating it for one session only. If you want to install permanently, start the script with `sudo`. If you decide to use it for only one session, the next window will provide the information you need to access the remote computer on the

DWS Remote Control on Wayland

Currently, to access the target computer's screen directly with DWS Remote Control on Wayland, you need to edit a file on the target computer. After the connection is established, click the *Files and Folders* button and navigate to the `/etc/gdm3/` folder. Right-click the `custom.conf` file there and select the *Open* option. After another right-click, select *Open for* and then *Text editing*. Look for the `#WaylandEnable=false` line in the file and remove the hashtag at the start of the line. Then save the file using the floppy disk icon in the menu-bar at the top. After rebooting the source computer, DWS Remote Control now also offers a 1:1 view on Wayland.

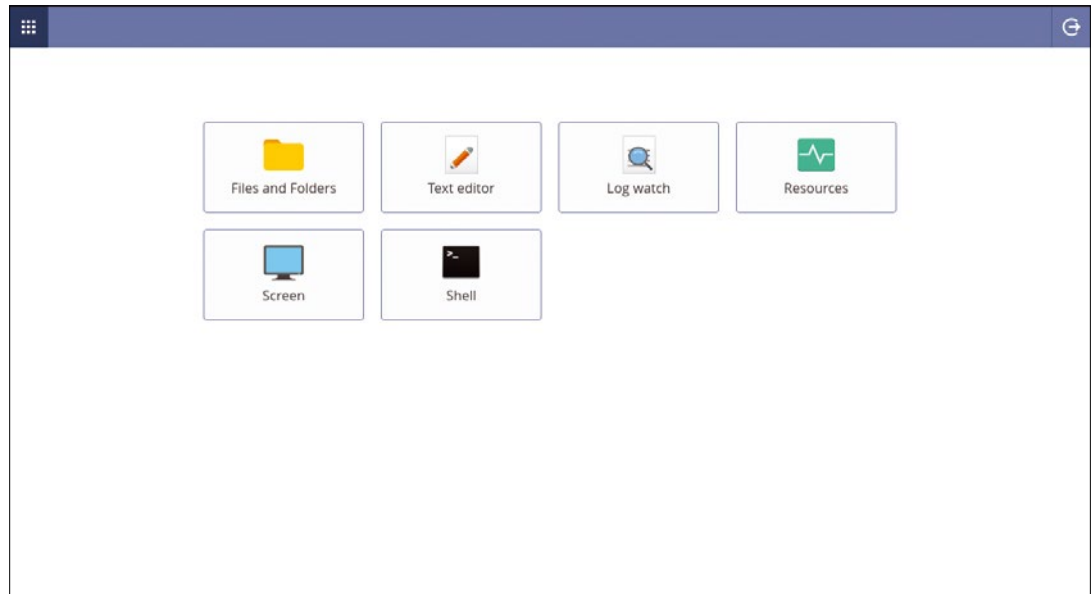


Figure 2: After successfully connecting to the remote computer, you will see a screen with six buttons that offer access to various parts of the remote desktop.

source computer on the DWSservice website. After entering the information, the other end needs to allow access unless *Unattended access* was selected when the session was initialized. You can also set a password for this.

For a permanent installation, enter the agent code previously shown on the web page during setup and then continue on the *Agents* tab of the web page by clicking *Online*. After accessing the remote computer, however, you will not see the desktop as a mirror image of the remote computer. Instead, you'll be presented with a screen with six buttons: *Files and Folders*, *Text editor*, *Log watch*, *Resources*, *Screen*, and *Shell* (Figure 2).

I tested DWS Remote Control with Debian 11, Ubuntu 22.04, and Fedora 37. With Ubuntu and Fedora, when I tried to access the target computer's interface directly via the *Screen* button, I got a message that the software does not support Wayland. To use DWS Remote Control with distributions that launch a Wayland session by default, you need to take a small detour, which the "DWS Remote Control on Wayland" box explains.

Different Operating Principle

DWS Remote Control uses an abstraction to represent the remote filesystem.

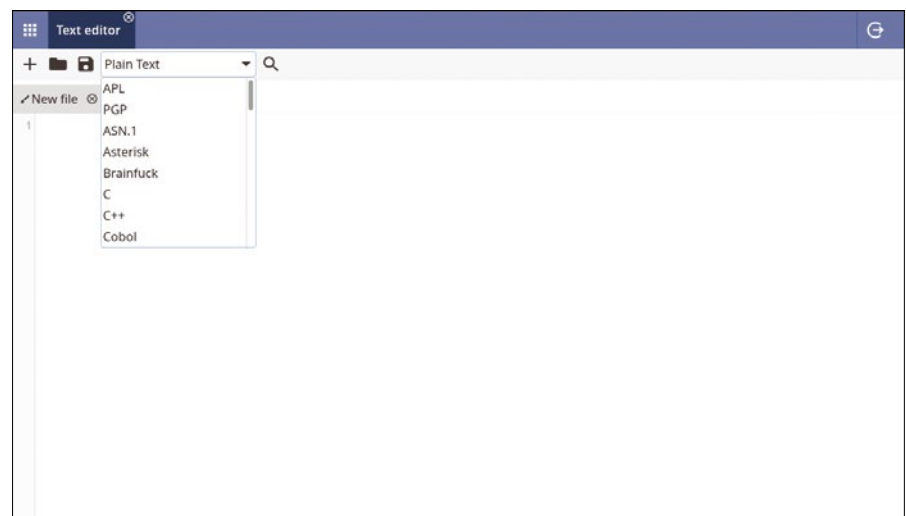


Figure 3: You can open and edit text files on the remote computer via *Text editor*. The application supports syntax highlighting for numerous programming languages.

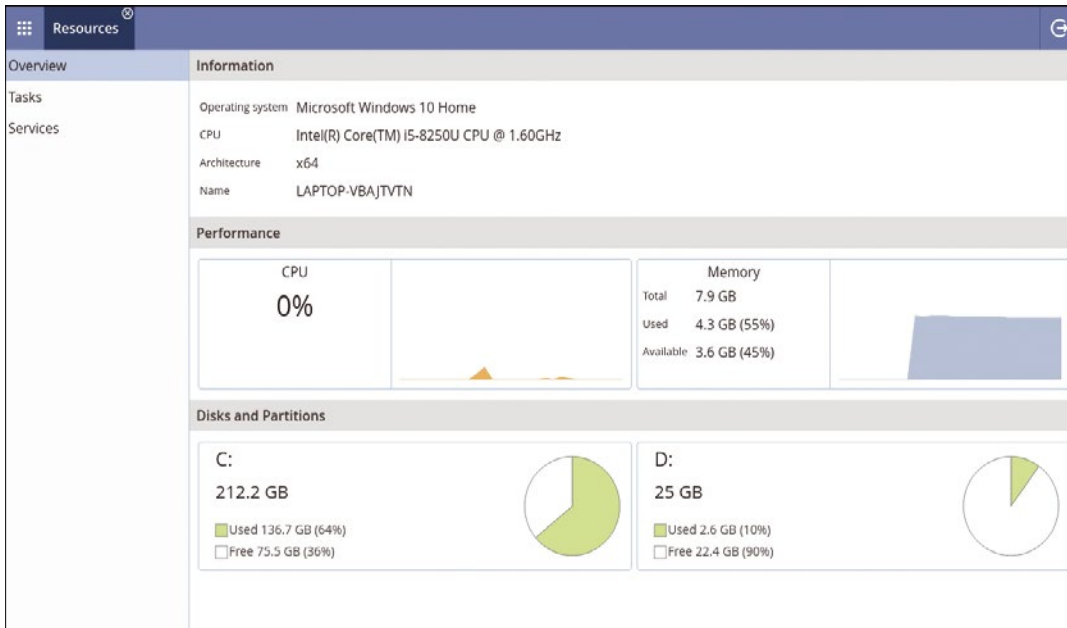


Figure 4: The *Resources* button displays information about the remote computer's hardware, as well as running services and processes.

Each remote desktop can be controlled in the same way. Only one of the buttons directly shows you the remote desktop.

The first button labeled *Files and Folders* takes you to a screen that maps the remote computer's filesystem. You can browse the directories and perform file operations such as create, delete, copy, cut, and paste. You also can upload and download data.

Text editor (Figure 3) takes you to a full-featured editor that lets you create new files and edit existing ones. *Log watch* lets you browse logs and monitor them in real time. A click on *Resources* reveals some information about the

hardware as well as the running processes and services (Figure 4).

Screen (Figure 5) takes you to a 1:1 view of the remote computer. If you work in this view, the user on the remote computer can follow what is happening. However, you can disable the other person's mouse and keyboard. In addition, you can scale the screen, switch to full-screen mode, and select keyboard shortcuts to share.

Shell takes you to a terminal emulation where you can run commands on the remote computer. You can switch between the individual categories using the button top left (a nine-square grid). Once opened, all categories remain open as tabs.

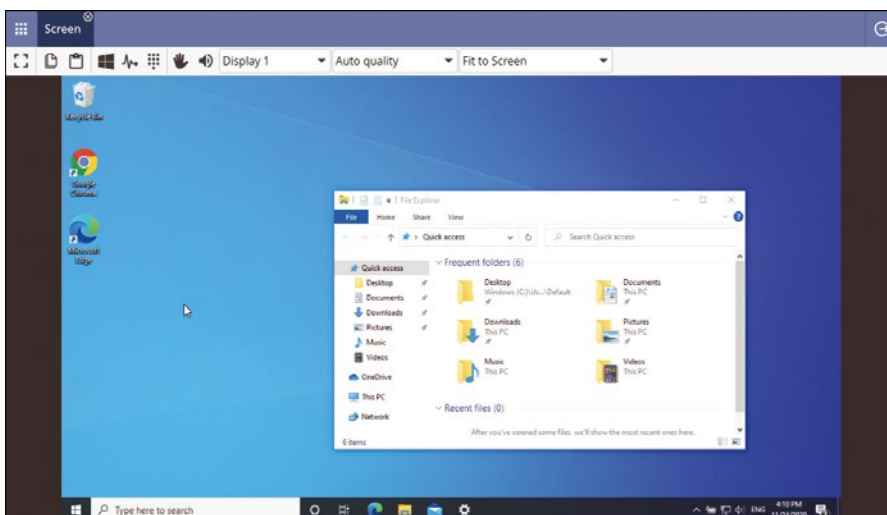


Figure 5: A click on the *Screen* button reveals the 1:1 view of the remote desktop usually seen with remote maintenance software.

Conclusions

DWS Remote Control was impressive in testing. No technical problems occurred when accessing computers on or outside of my home network. The Android version also performed as expected. Use with Wayland currently still requires a small detour on the part of the user. Detailed documentation is available in English [7].

The free version of DWS Remote Control already works fast enough at 6Mbps; latency is hardly noticeable. If you need more bandwidth because you want to transfer larger

volumes of data or run several sessions at the same time, there are various subscription plans [8] to choose from, ranging from 8Mbps for \$3 a month to 50Mbps for \$120 a month. Because it is an open source application, the API is also open, which allows developers to include DWS Remote Control in their own apps [9]. ■■■

Info

- [1] VNC: https://en.wikipedia.org/wiki/Virtual_Network_Computing
- [2] Remmina: <https://remmina.org>
- [3] RustDesk: "A Free Alternative to TeamViewer and AnyDesk" by Thomas Leichtenstern, *Linux Magazine*, issue 266, January 2023, pp. 82-85
- [4] DWService: <https://www.dwservice.net/en/home.html>
- [5] DWS Remote Control download: <https://www.dwservice.net/en/download.html>
- [6] License: <https://www.dwservice.net/en/licenses-sources.html>
- [7] Documentation: <https://docs.dwservice.net/docs/site/dwservice-installation/>
- [8] Prices: <https://www.dwservice.net/en/contribute-subscriptions.html>
- [9] DWS Remote Control API: <https://docs.dwservice.net/docs/api/>

Author

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

DORS/CLUC

**28th OPEN SYSTEMS DAYS
CROATIAN LINUX USERS' CONFERENCE**

a place where hackers
and entrepreneurs meet
enterprise and public
sector representatives
to exchange experiences,
offer advice, and discuss
the advantages
of open systems.

FIND OUT MORE



**ORGANIZED BY
NONPROFITS:**

HrOpen
www.open.hr

HULK
www.hulk.hr

Oldest
and largest
regional
conference
that covers
free and open
source software,
open standards,
and the Linux
operating system.

dorscluc.org

11-12 MAY 2023

Zagreb - CROATIA



Debian opens a door for non-free firmware

Well, OK, I Guess?

The topic of non-free firmware has caused some turbulence within the Debian project, but now the community has a new direction. *By Thomas Reuß*

Firmware is the link between software and hardware. Hardware vendors are often very secretive about their technology, and part of that secrecy is carefully guarding their source code. Consequently, firmware usually means blobs, that is, binary large objects. No source code is available.

Unfortunately, closed-source, proprietary firmware contradicts everything Debian stands for. Binaries without the source code are also a problem from the admin's point of view because the hidden code might hide vulnerabilities that the user is not able to fix. Many users don't like the fact that firmware cannot be patched as easily as source code when a security problem occurs. Without the manufacturer, who has to provide the security updates, you can't do anything. And the distributor will want to thoroughly test the new firmware version before it goes to the user, which slows down the process of fixing problems.

In general, proprietary firmware does not score well on the popularity scale

with Linux users. However, users are in a quandary. Either you swallow the bitter medicine and live with closed-source packages, or else you do without the hardware and other proprietary components that depend on them. Virtually no one who uses Linux in a professional setting can afford to operate without proprietary firmware. When I used to install Debian, I (and probably many others) directly enabled the *contrib* (contributed) and *unfree* packages – initially, out of curiosity or ignorance, and then later knowing that I wouldn't have much fun with the hardware if I installed it without some proprietary drivers.

Idealism Meets Realism

The Debian Popularity Contest is a software package that evaluates the packages installed on the system and transmits the results to Debian. Participation is regulated by an opt-in procedure. When it comes to proprietary firmware, the numbers from the Debian Popularity Contest [1] are quite revealing.

The Debian Popularity Contest returns fairly clear results: Firmware is quite important. In the ranking by installation count (Figure 1), packages such as *linux-firmware-free* (ranked 237) or *firmware-misc-nonfree* (ranked 1,882) score significantly higher than well-known applications such as *samba* (ranked 2,411) or *mysql-server* (ranked 2,838).

Hardly any system can operate sensibly today without additional firmware, especially when it comes to laptops. The fact that iconic free tools such as MariaDB and Samba appear after the unloved closed-source firmware on the popularity list suggests that a clear majority of users want a fully functional computer and do not draw a stark line in the sand when it comes to proprietary firmware.

Debian newcomers or inexperienced users are often reluctant to manually deploy firmware, and even experienced users might find themselves in a situation like I did from time to time: Halfway through the installation, I realized

Photo by Kostiantyn Li on Unsplash

that, without the closed-source packages, I wouldn't have a network at all. So I went to the nearest computer to plug in a USB stick and download the non-free packages. Hoping to have fetched all the DEBs I needed, I put my storage medium back into the notebook and started a new installation attempt. What might feel a little time consuming to the Linux old hand is, in all likelihood, a merciless showstopper for newcomers.

Democratic Decision

The Debian community recently embarked on a decision process to consider whether to allow the Debian installer to install proprietary firmware. Of course, users can always install the binary blobs manually, but the process of finding and installing drivers for proprietary hardware can be disruptive and time consuming. As is often the case with the very deliberative and collectivist oriented Debian community, this debate played out over several months, culminating in a vote.

```
#Format:
#
#<name> is the package name;$
#<inst> is the number of people who installed this package;$
#<vote> is the number of people who use this package regularly;$
#<old> is the number of people who installed, but don't use this package
#       regularly;$
#<recent> is the number of people who upgraded this package recently;$
#<no-files> is the number of people whose entry didn't contain enough
#       information (atime and ctime were 0).$
#rank name                inst vote   old recent no-files (maintainer)$
...$
230  telnet                  181438 11110 148666 18756 2906 (Guillem Jover)$
232  eject                    180866 17651 138632 24512  71 (Util-linux Packagers)$
...$
237  firmware-linux-free      179119 45466 115014 18543  96 (Debian Kernel Team)$
...$
247  openssh-server           174292 152594 15392  6260  46 (Debian Openssh Maintainers)$
...$
1871 mesa-utils              43279 10267 30248  2748  16 (Debian X Strike Force)$
...$
1877 ffmpeg                  43158  6591 22427 14129  11 (Debian Multimedia Maintainers)$
...$
1882 firmware-misc-nonfree  42999 19212 19494  4279  14 (Debian Kernel Team)$
...$
1888 linux-headers-amd64     42845  0     0     0 42845 (Debian Kernel Team)$
...$
1894 lightdm                 42785 30725  8525  3518  17 (Debian Xfce Maintainers)$
...$
2411 samba                   30112 22401  7213  482  16 (Debian Samba Maintainers)$
...$
2440 firmware-realtek        29346 16691 10332  2310  13 (Debian Kernel Team)$
...$
2465 firmware-linux-nonfree 28853  124  1045  0 27684 (Debian Kernel Team)$
...$
2838 mariadb-server          23271  59   6    30 23176 (Debian Mysql Maintainers)$
...$
2846 firmware-iwlwifi        23168 16277  4956  1926  9 (Debian Kernel Team)$
...$
```

Figure 1: The Debian Popularity Contest (sorted by installations) shows that reality has moved on.

What?!

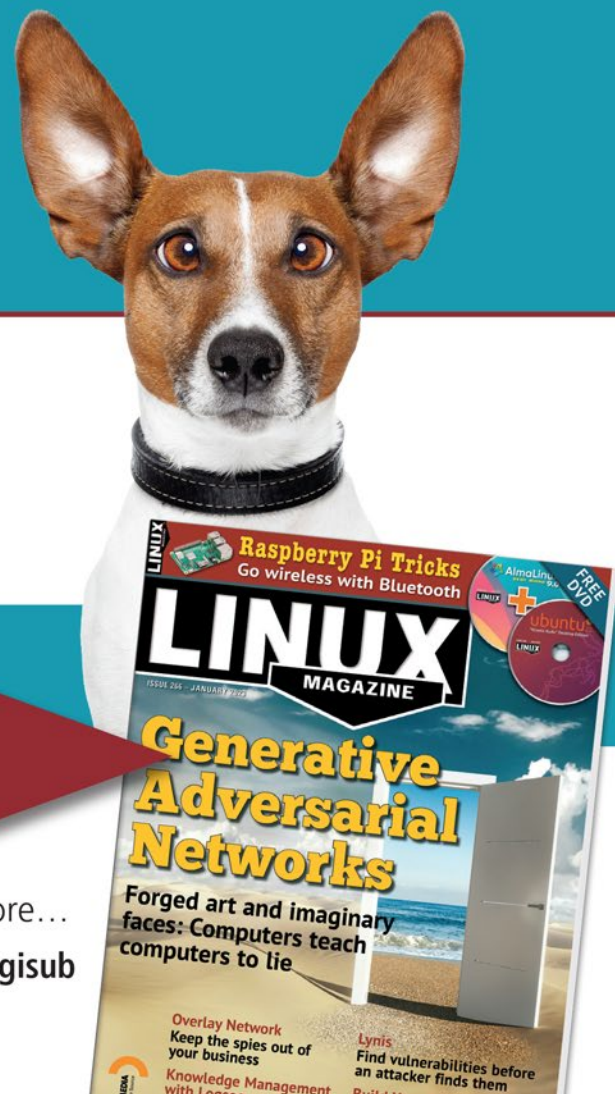
I can get my issues SOONER?



Available anywhere, anytime!

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

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



One complication of this proposal is that the Debian Social Contract (the document on which Debian was founded) states that “Debian will remain 100% free.” There was some question about whether including proprietary drivers was a violation of the social contract, and, if so, what to do about it. Some suggested amending the contract. Others felt an amendment wasn’t necessary, because these binary blobs aren’t really *part* of Debian – they are just software that the Debian installer goes out and fetches at installation time. Another question was whether the Debian project should support two installers (one that only handles free components and another installer that can also include binary blobs) or whether to maintain a single installer that integrates binary blob capabilities.

Voting closed on October 1, 2022, and the Debian community officially decided to offer non-free firmware through a single installer and to amend the social

contract slightly to make the contract consistent with this change. Steve McIntyre, one of the leaders of the Debian project, published the voting results and info on his blog [2].

The Debian Social Contract will be amended as follows:

“The Debian official media may include firmware that is otherwise not part of the Debian system to enable use of Debian with hardware that requires such firmware.”

This decision to officially ship firmware is not surprising to my mind – on the contrary, I think this step is long overdue. What is probably the best-known Debian competitor, Ubuntu, has always come with proprietary firmware packages, which is one of the reasons why Ubuntu is considered particularly beginner-friendly. Other Linux variants do not make such a fuss about firmware and do not regard the inclusion of closed-source packages as sacrilege. But keep in mind that, in many cases, companies or foundations pull the strings in

the background to manage these other distributions, whereas the community-driven Debian requires a formal and public decision process.

Conclusions

A number of hardliners were absolutely against the inclusion of non-free firmware, but the majority of the community seems to welcome the project’s decision. On Reddit, for example, some members voiced opinions close to my heart (Figure 2).

Debian has long been considered an extremely robust and reliable distribution – perhaps precisely because much of it seems set in stone and the project has moved very little, if at all, over the decades. I have hardly come across a Linux admin who does not swear by Debian. Many have success stories at hand, typically from a system that had “Woody” or “Sarge” installed eons ago, has since gone through the umpteenth distribution upgrade, and is still running smoothly.

If Debian tweaks the social contract, will the idea or ideals of open source software suffer? I don’t think so. Users will still enjoy full control – after all, the installer will ask if you want to set up the firmware. Nobody is forcing you to use the blobs, but you are free to do so. ■■■

Info

- [1] Debian Popularity Contest: <https://popcon.debian.org>
- [2] Steve McIntyre’s blog: <https://blog.einval.com>

Author

Thomas Reuß is a passionate Linux admin who is hugely interested in security. He is currently working as a consultant in the SAP environment.

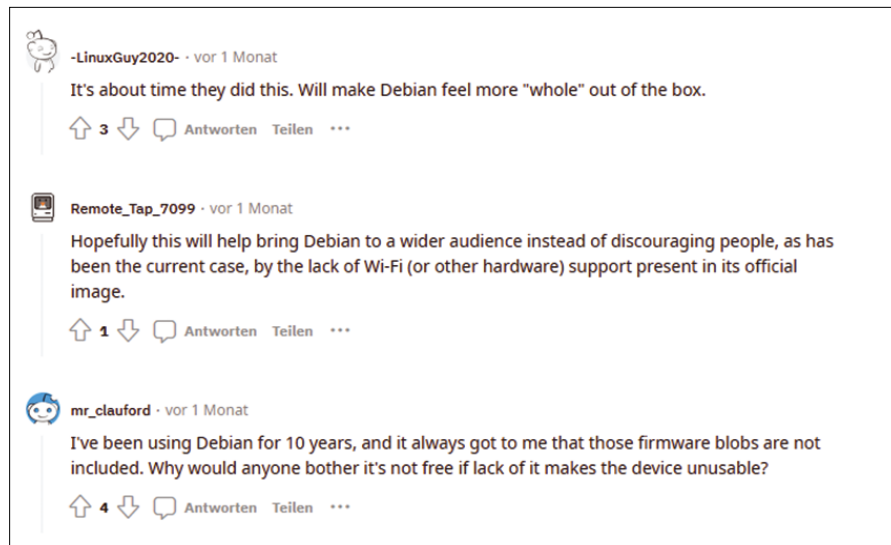


Figure 2: It’s not just on Reddit that many agree: Debian’s move to non-free firmware is long overdue. © Reddit.com

GET TO KNOW ADMIN



ADMIN Network & Security magazine is your source for technical solutions to real-world problems.

ADMIN is packed with detailed discussions aimed at the professional reader on contemporary topics including security, cloud computing, DevOps, HPC, containers, networking, and more.

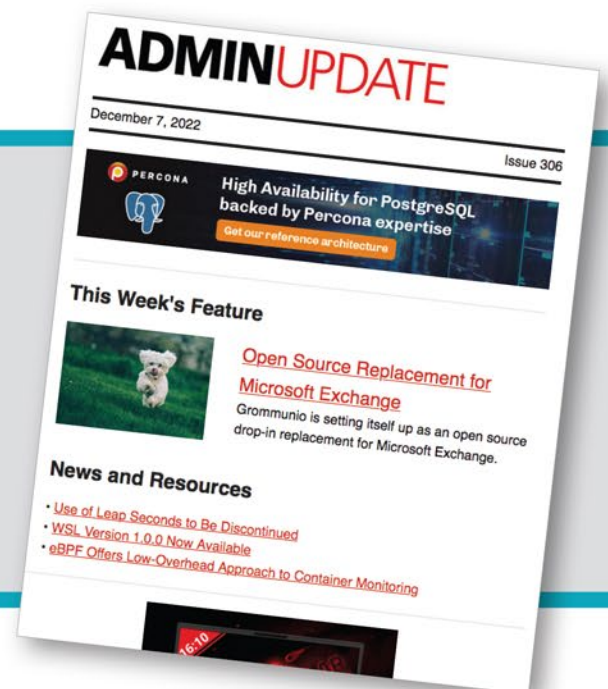
Subscribe to *ADMIN*
and get 6 issues delivered every year

Want to get ADMIN in your inbox?

**Subscribe free to
ADMIN Update**

and get news and technical articles you won't see in the magazine.

bit.ly/HPC-ADMIN-Update



@adminmagazine



@adminmag



ADMIN magazine



@adminmagazine

Enhanced searches with fzf

Fuzzy Finder

Simplify your searches and get better results with fzf, a modern search tool based on fuzzy logic. *By Bruce Byfield*

Search commands have always been essential to using computers. With the increased storage capacity of modern systems, they are more important than ever because there is more material to search. In fact, searches are so important that alternatives to time-honored commands are becoming increasingly common. One of the most effective of these recent alternatives is *fzf*, a command-line “fuzzy finder” [1].

For most Linux users, the basic search tool for finding directories and files has been *find*, which is most effective when you know exactly what you are looking for. If you do not know, then you need a way to allow variation in the output.

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/>.

Traditionally, variation in search tools like *find* is provided by regular expressions (regexes), a concept first developed in the 1950s. Regexes are standard patterns that offer different types of variations in results. Most users are familiar with the simpler regexes, such as *** to mean zero or any character, so that *apt** might return *apt*, *apt-get*, *apt-cache*, and probably many others. At a more advanced level, users might know that *gray|grey* produces results with either spelling, or *[b-e]* a range of characters. However, regexes are not consistent across applications and can rapidly become extremely complex to construct or type correctly, especially when all you want is a quick answer. For example, even with context, who can easily explain this randomly selected example:

```
(\W|^)[\w.\-]{0,25}
@(yahoo|hotmail|gmail)\.com(\W|$)
```

Personally, I’d take half a minute or so and resent the expected effort. Such regexes are worth constructing only when you use the same search repeatedly. In addition, you almost always will find

scanning fuzzy logic results quicker than entering a lengthy regex, even if you type the regex correctly the first time. With *fzf*, you interact with the tool through a fuzzy logic interface, while *fzf* composes the regexes for a search behind the scenes.

Fuzzy logic is a science of its own, with principles and theories about communication and logic that do not directly concern us here [2]. What matters is that fuzzy logic is a much simpler way of introducing variation into output, especially today when the patience and general command-line knowledge among computer users is much less than it was 70 years ago. For one thing, the order in which variation is found with fuzzy logic is less important than with regexes. Often, the order does not matter at all, which makes it easier to understand and more flexible. Just as importantly, fuzzy logic uses natural language instead of its own code that needs to be carefully separated from the query itself. If you add an indexed database, as *fzf* does, you can add context to accuracy to produce a contextual command history.

With these advantages, *fzf* has rapidly gained ground in the past few years. It is carried in all the major distributions, or it can be installed from its GitHub page. Each time *fzf* opens using the bare

Table 1: Keyboard Shortcuts

Ctrl+T	Select files
Alt+C	Switch into a selected subdirectory
Ctrl+R	Use fzf's enhanced history

command, it updates its database in a matter of seconds, leaving you at a prompt, ready to continue.

Setting Up fzf

The popularity of fzf means that, for most purposes, you can install it directly from your distribution's repositories and begin to use it. You can also install fzf with Homebrew to get the absolute latest version.

However, to make full use of fzf, you should also install bat (the modern cat replacement) and fd-find (the modern find replacement). Then add the following lines to your .bash.rc files:

```
export FZF_DEFAULT_OPTS="--preview 'bat \
--color=always {}'"
export FZF_DEFAULT_COMMAND="fd --type f"
```

These lines will ensure that fzf can use bat as a file previewer and improve overall performance with fd-find. Faster than find, fd-find will ignore directories

like .gitignore while searching. In addition, add

```
$ source /usr/share/fzf/shell/\
key-bindings.bash
```

to enable some useful keyboard shortcuts (see Table 1).

Basic Use

At its simplest, fzf opens and updates its list of top-level directories for the present working directory – usually, your home directory. You can then type any part of a directory or file to see the results (Figure 1). If you scroll the results, highlighting and pressing the Enter key displays the selection in the terminal, where it can be conveniently pasted into newly typed commands. Alternatively, you can use fzf with another command so that

```
vim $(fzf)
```

allows you to use fzf to select a file to open in Vim.

For many uses, this is enough information to use fzf. However, depending on how fzf is compiled, it can also support context-specific autocompletions by entering a command followed by `***Tab`

and will give appropriate suggestions, such as file names for Vim or remote systems for SSH. If autocompletion is not enabled, check the project's GitHub page for instructions.

Other Command Options

Most online articles on fzf end at this point. However, a few distributions, such as Ubuntu, offer man pages with additional options, neatly organized into categories. Among the most immediately useful are the search modes.

Some of these options may be familiar to you from find and other search

commands. For instance, with +1, searches become case-sensitive, while you can reduce clutter in the results with --exact (-e). In addition, --algo=TYPE sets the fuzzy algorithm, completion with v1 optimizes speed over best results, and v2 optimizes results over speed.

Another useful option category is for search results. The arrangement of results can be influenced by --tiebreak=CRI CRITERIA, CRITERIA. The default criteria is length, which favors shorter results, but you can give it lower priority by its position in a list that can contain begin (with the search string closer to the start of the line), end (with the search string closer to the end of the line), or index (a line that already appears in the results). Each criterion should only appear once, and index (for obvious reasons) must always appear last. If you are searching for high-level directories, start might be a useful option or end if you are searching for a file name. In addition, --tac displays results in reverse order.

Other options are cosmetic, such as disabling the mouse or horizontal or vertical control, or aspects of appearance, such as color theme or window border size for previews. These options no doubt become more important the more often you use fzf, but at first glance give a false sense of complexity for what is after all a relatively well-organized utility.

Rough Edges

The latest version of fzf is .38, and many distributions offer even earlier packages. Unsurprisingly, fzf is still rough around the edges, with no indication of what features are compiled in which package or distribution and scanty documentation. Fortunately, though, you do not need all the possible features to find fzf a major improvement in search functions. In fact, fzf has several levels of complexity, each of which is useful in itself. As a result, even with the general release apparently some way off, fzf is a welcome innovation that is well on the way to becoming the new norm in search commands. ■■■

Info

- [1] fzf: <https://origin-sysadmin.redhat.com/fzf-linux-fuzzy-finder>
- [2] Fuzzy logic: https://en.wikipedia.org/wiki/Fuzzy_logic



Figure 1: The results shown in the fzf window.



Customize your system tray with YAD

BESPOKE SYSTEM TRAY

YAD lets you customize your system tray with one-line Bash tray scripts. *By Pete Metcalfe*

My goal was to find a simple way to group together my favorite apps and web pages into a system tray item.

There are a number of different approaches to this, but for my requirements I found that the YAD (Yet Another Dialog) tool [1] gave me everything that I needed, and I could do it all with just one line of Bash script.

In this article, I will introduce Bash tray scripts with three examples. The first example will show how to create tray scripts that put Linux diagnostic data into both custom dialogs and terminal windows. In the second example, I will add pop-up browser windows to a right-click submenu. The final example will look at how to toggle the tray icon, command, and tooltip with simulated weather conditions.

Listing 1: Creating a System Tray Item

```
yad --notification --image="gtk-execute" \
  --command="yad --text=\"$(lsusb)\" \
  --title=\"USB Info\" --fixed --button=ok\" \
  --text="My Fav Utility"
```

Getting Started

There are a number of different tools for creating system tray applications, such as AllTray and KDocker. For my projects, I prefer YAD because it lets you create custom dialogs and it supports dynamic changes to the tray features. YAD is a command-line dialog tool very similar to Zenity [2], but with some added features such as system tray support and a more complete dialog functionality.

For Debian/Raspian/Ubuntu systems, you can install YAD with:

```
sudo apt install yad
```

Creating a Tray Script

Listing 1 is a one-line Bash script that calls YAD to create a system

tray item (Figure 1). Because Bash and YAD statements can be quite long, you can make the code more readable by using backslash (\) characters to extend a statement across multiple lines, as shown in Listing 1.

The yad statement in Listing 1 uses the `--notification` and `--image` options to put a user-defined icon in the system tray. The `--command` option calls an application or script when you click on the tray item. The `--text` option defines tooltips.

In Figure 1, the command-line `lsusb` utility lists the USB-connected devices. The output from `lsusb` is shown in a YAD text dialog.

To terminate a YAD tray script, use a center mouse click. You can also terminate from a right-click menu, which I'll discuss later.

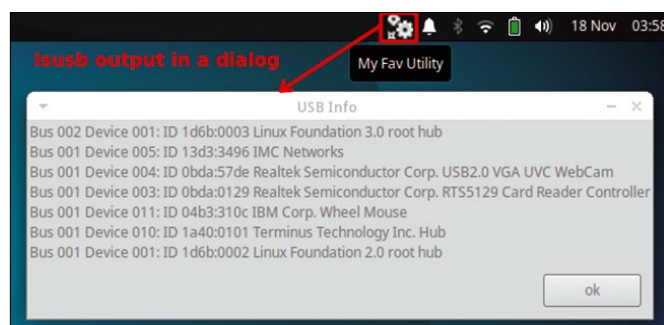


Figure 1: A one-line Bash script creates a system tray item that displays the output from `lsusb` in a dialog.

Photo by Suhyeon Choi on Unsplash

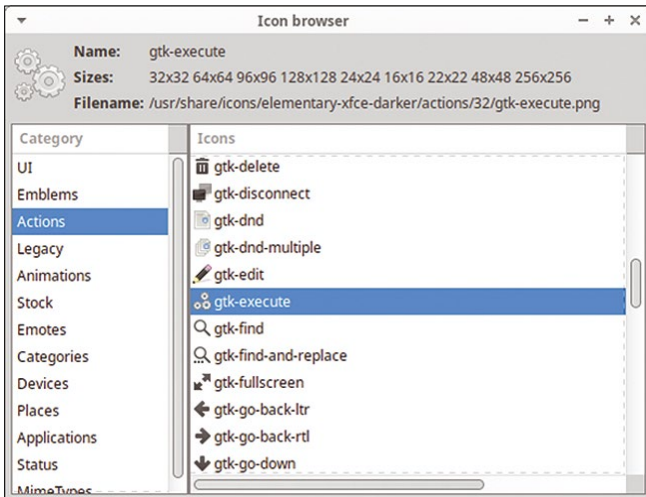


Figure 2: You can use `yad-icon-browser` to find and view available icons.

YAD includes a handy `yad-icon-browser` tool, which lets you review available icons (Figure 2). To make coding a little easier, YAD only needs the image name; a full path to the image is not required.

Many applications, such as the `top` system performance tool, are best viewed in a terminal window. The YAD tray script in Listing 2 calls `top` within a terminal window (Figure 3).

Listing 2 calls `xterm` to launch a terminal window. The `-e` option executes the `top` utility. The `-hold` option keeps the terminal open after the command is complete. Font types and sizes can be defined with the `-fa` and `-fs` options. The terminal window caption is set using the `-T` parameter.

Right-Click Menus

YAD also supports right-click menu command items in addition to left-click commands. The syntax for menu items is:

```
--menu="menu_title ! menu_command1 | ↵
      menu_title2 ! menu_command2 ... "
```

Listing 3 shows the main command and then four right-click menu options (Figure 4). The main command calls `iostat` (the CPU and I/O reporting utility). The first three right-click options are `vmstat` (memory stats), `lmsensor` (print sensor info), and `lsusb` (list USB devices). The fourth option kills the script.

Instead of using one extremely large Bash statement, the `nice_dialog` function is created to make the output a little more presentable. This function adjusts font type and font size for text

within the YAD dialog. For more information on how to configure YAD color and font options, see the Pango markup language [3] documentation.

Pop-Up Browser Windows

Midori [4], a lightweight web browser, has been available with the Raspberry Pi desktop for many years. One of Midori's advantages is that it can be easily launched as a pop-up window without affecting your main browser settings.

To install Midori on a Debian/Raspbian/Ubuntu system enter:

```
sudo apt install midori
```

Midori can be run as a pop-up window by using the web application `-a` command-line option.

Listing 2: Calling top in a Terminal Window

```
yad --notification --image="media-memory" \
--command="xterm -hold -fa Monospace -fs 12 \
-T 'System Performance' -e 'top' " \
--text="Show System Performance"
```

Listing 3: Adding Four Right-Click Menu Options

```
#!/bin/bash
#
# tray2yad.sh - create tray item with some Bash utilities
#               - add a function to change YAD text options
#
nice_dialog() {
# Use Pango markup language for custom text presentations
echo "yad --text='<span size=\"large\"><tt>${1}</tt></span>' \
--title=$2 --button=OK \
--fixed "
}

# Create a system tray item with 4 right click items
yad --notification --image="applications-utilities" \
--command="$(nice_dialog 'iostat -c' 'IOSTAT')\" \
--menu="Memory! $(nice_dialog 'vmstat' 'VMSTAT') \
      | Sensors! $(nice_dialog 'sensors' 'SENSORS') \
      | USB ! $(nice_dialog 'lsusb' 'USB') \
      | Quit ! killall yad" \
--text="My Fav Utilities"
```

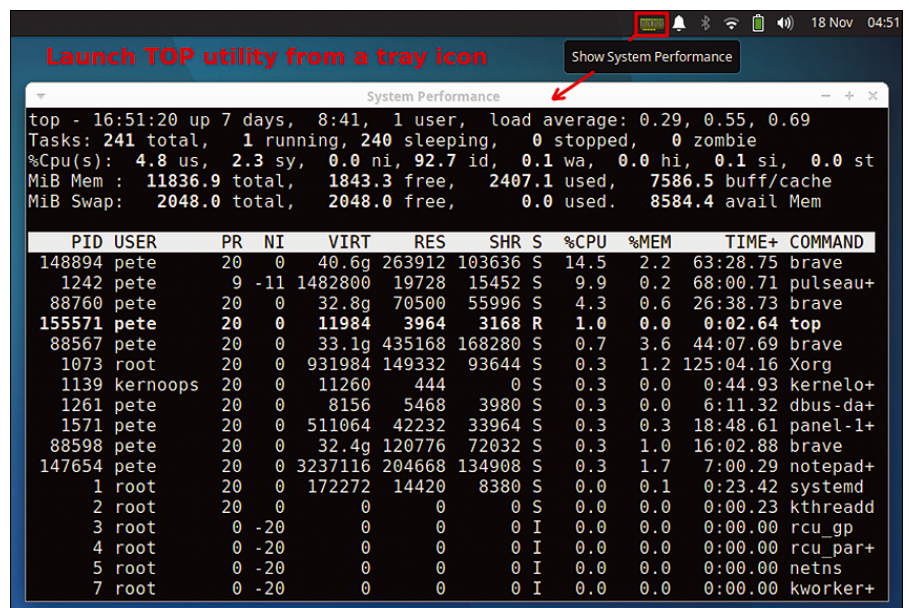


Figure 3: YAD lets you launch commands in a terminal window.

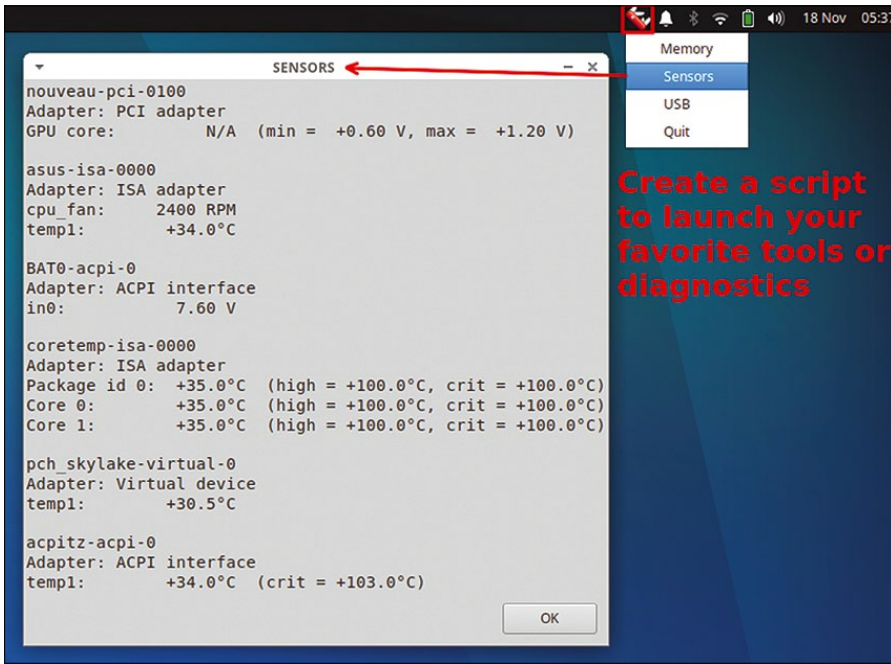


Figure 4: Right-clicking on the system tray item displays four menu options.

Listing 4 shows a one-line Bash statement that creates a tray icon that can launch four different Midori browser windows (Figure 5).

Toggling

YAD supports the ability to dynamically change a tray item's icon, command, menuing, and tooltip. To do this you need to enable listen mode (`--listen`) and then redirect the standard I/O (`stdio`) – file 1 as shown in Listing 5 – to a named pipe. After

completion, a YAD tray item can be manipulated from the main script, an external script, or manually from a terminal.

Listing 5 shows a stand-alone script that creates a system tray item that can toggle some of its tray features (e.g., icon,

tooltip, or command) every 10 seconds.

The first step in Listing 5 sets up a named pipe (lines 6 and 9-11). The named pipe is a file that is used to pass command arguments to YAD. Next, redirect `stdio` (file 1) to the named pipe (line 14) and then have YAD get its input from the redirected file 1 (line 21).

A while loop cycles every 10 seconds (line 24-26). Custom YAD command arguments are written/redirected to the named pipe with `echo` statements (e.g., line 27). Adding `sleep` statements helps ensure that YAD doesn't miss a command before a new command is written.

Listing 5 uses static weather data (Figure 6). A future step would be to periodically scan for actual weather data. If you're interested in using Bash to scrape the web [5], take a look at the Lynx [6] command-line browser. With one line of Bash, you can use Lynx to cleanly extract the contents of a web page; then, with a piped `grep` command, you can parse out the desired data.

Listing 4: Launching Midori Browser Windows

```
yad --notification --image="emblem-web" \  
--command="midori -a https://news.google.com/" \  
--menu="Facebook! midori -a https://mbasic.facebook.com \  
| Linux! midori -a https://www.linux-magazine.com \  
| Weather! midori -a https://www.theweathernetwork.com \  
| Quit ! killall yad" \  
--text="My Fav Web Pages"
```

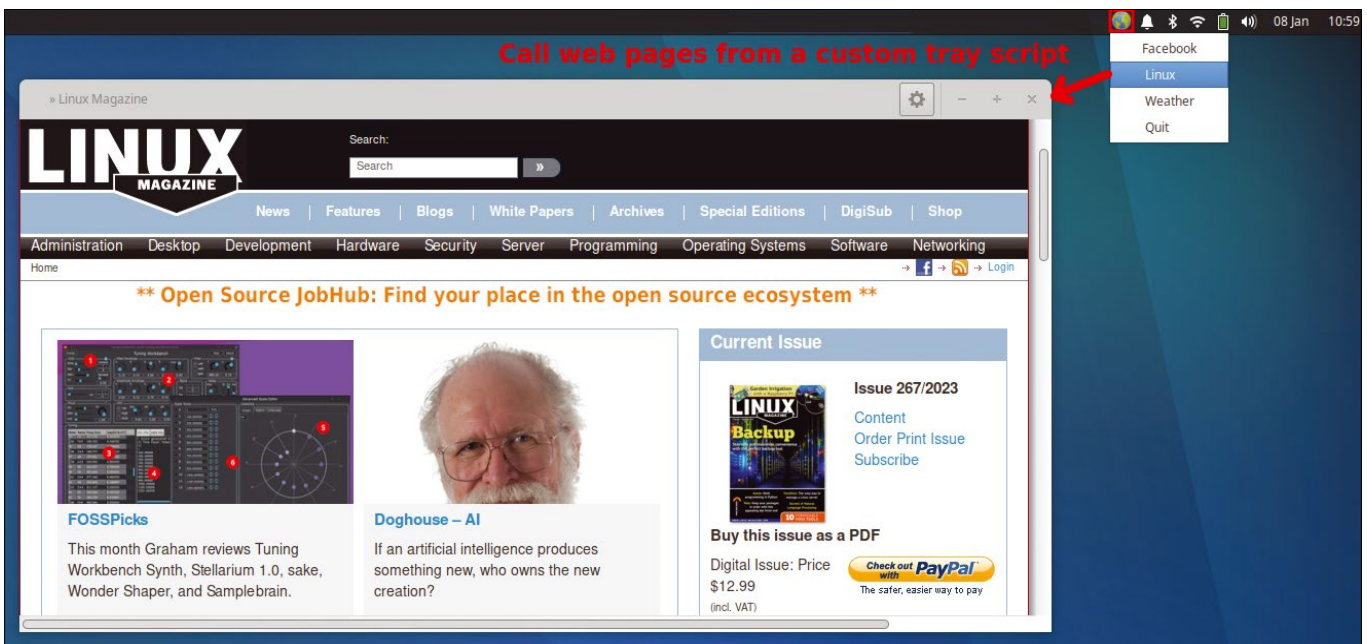


Figure 5: You can also create a tray icon that will launch Midori browser windows for your favorite web pages.

Listing 5: Toggle Tray Item Features

```

01 #!/bin/bash
02 #
03 # tray_toggle.sh - toggle system tray items
04 #           - create a named pipe for inputs
05 #
06 mytraypipe="/tmp/tray1.pipe"
07
08 # Make the named pipe (if it doesn't exist)
09 if ! test -e "$mytraypipe"; then
10     mkfifo $mytraypipe
11 fi
12
13 # redirect the stdio (file 1) to the named pipe
14 exec 1<> $mytraypipe
15
16 # create the notification icon
17 yad --notification          \
18     --listen                \
19     --image="emblem-colors-grey" \
20     --text="My Tray Test" \
21     --command="yad --text='Test Tray App' " <&1
22
23 # Every 10 seconds toggle the tray features with fake weather
24 while :
25 do
26     sleep 10
27     echo "action:yad --text='Rain until morning'" > $mytraypipe
28     sleep 1
29     echo "icon:stock_weather-showers" >> $mytraypipe
30     sleep 1
31     echo "tooltip:Rain until morning" >> $mytraypipe
32     sleep 10
33     echo "action:yad --text='Sunny for 2 days'" > $mytraypipe
34     sleep 1
35     echo "icon:stock_weather-sunny" >> $mytraypipe
36     sleep 1
37     echo "tooltip:Sunny for 2 days" >> $mytraypipe
38 done

```

It should be noted that the YAD menus can also be dynamically changed. In Listing 5, to add a new menu item or change a menu item, the code would look something like:

```

echo "action:menu= \
    menu_title1 ! menu_command1 | \
    menu_title2 ! menu_command2" \
> $mytraypipe

```

Summary

Coding errors are a fact of life, so I found the command:

```
killall yad ; killall bash
```

to be quite useful, especially when I was playing with YAD in listening mode.

It's pretty amazing that with just one or a couple of lines of Bash script you can pull together all your favorite apps and web pages into a common system tray icon. ■■■

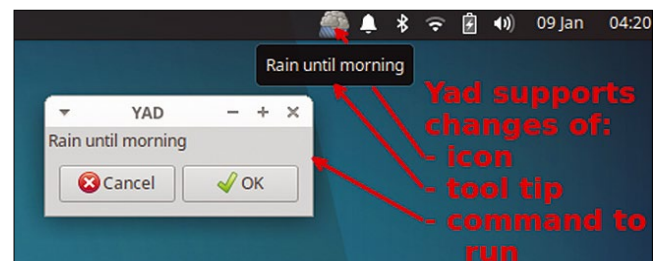


Figure 6: You can toggle the system tray icon, tooltip, or command with YAD.

Info

- [1] YAD: <https://www.systutorials.com/docs/linux/man/1-yad/>
- [2] Zenity: <https://help.gnome.org/users/zenity/stable/>
- [3] Pango documentation: https://docs.gtk.org/Pango/pango_markup.html#pango-markup
- [4] Midori: <https://linuxcommandlibrary.com/man/midori>
- [5] "Simple Web Scraping with Bash" by Pete Metcalfe, *Linux Magazine*, issue 262, September 2022, p.36
- [6] Lynx: <https://lynx.invisible-island.net/>



Hone your skills with special editions!

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

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

Available in print or digital format

Check out the full library!
shop.linuxnewmedia.com

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

GETTING STARTED WITH
LINUX

MORE POWERFUL • MORE SECURE • MORE...

LEARN HOW TO SET UP A LINUX SYSTEM
 • Listen to Music • Play Games • Process P...
 • Surf the Web • and Much More!

START

LINUX NEW MEDIA
 WWW.LINUX-MAG.COM

LINUX 301 BEST BASH COMMANDS

LINUX SHELL
HANDBOOK 2022 Edition **LINUX Special**

SUPERCHARGE
 YOUR LINUX SKILLS

Power at Your Fingertips
 • Pipe and redirect output
 • Monitor processes

BUILD A RASP PI RADIO!

MakerSpace #02

HANDS-ON PROJECTS FOR MAKERS

Cool Tricks!
 Charge up with a saltwater battery

Painting with Light
 Sure you need a programmable light stick

PiMiga 2.0
 Get your game on with this Amiga emulator

MORE FUN FOR FPGA GEEKS!

LINUX NEW MEDIA



FREE DVD! LibreOffice Full Version
 Dive deep into the world's greatest free office suite
 2022/23 Edition

LibreOffice Expert

Edit and Save MS Office Files
 Write Your Own LO Macros
 Save time and automate common tasks

LINUX NEW MEDIA USA

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

MakerSpace

HANDS-ON PROJECTS FOR MAKERS

LINUX NEW MEDIA USA

LINUX 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

RETRO FUN: Play DOS Games with DOSBox
 AUDIO EXPERTS: Tools for DJs, Musicians, Composers

Discover the secrets of the experts

WWW.LINUX-MAGAZINE.COM





Reduce PDF file sizes with Minuimus

Shrink Ray

The Minuimus Perl script helps you save disk space by reducing the file size of PDF files with just a few commands.

By Daniel Tibi

Whether in an electronic library or an electronic file folder, PDF documents are ubiquitous. These digital documents can quickly eat up your disk space or fill up your cloud storage. Minuimus [1], a Perl script, helps you reduce PDF file sizes to free up space while preserving file quality.

Installation

You can download the current version, Minuimus v3.7.1, from the project

page [2]. First unpack the ZIP file and then run the following commands one after the other:

```
sudo make deps
sudo make all
sudo make install
```

The first call checks the dependencies and installs any missing programs. The second command creates the required Minuimus files, and the third command copies them to the `/usr/`

`bin/` directory. Upon completion, Minuimus will be available for all system users.

If you don't get the results you want with Minuimus's default settings, you can try one of the Minuimus options for potentially better results. Enter

```
minuimus.pl --help
```

to see a list of all the available options.

Operating Principle

The principle behind Minuimus is simple: Minuimus first decompresses a PDF file and then recompresses it more efficiently. Minuimus draws on its own capabilities as well as a

Photo by Dylann Hendricks on Unsplash


```

daniel@tardis: ~
daniel@tardis:~$ minuimus.pl lu-ce_2022-12.pdf
Using temporary folder /tmp/9dbIZk_ja0
Attempting: /home/daniel/lu-ce_2022-12.pdf:12742898
adv_pdf(/home/daniel/lu-ce_2022-12.pdf) using tempfile /tmp/9dbIZk_ja0/68600-1899.pdf
Pre-processing using mutool.
Successful.
Detected qpdf version >=9
adv_pdf_iterate_objects() complete. Optimised 18 object headers.
adv_pdf_iterate_objects() complete. Optimised 0 object headers, 492 stream contents.
Advanced PDF processing done: Was 12683341, finished 11197980.
Success! 12742898 to 11197980 (0.878762429080104)
daniel@tardis:~$ minuimus.pl houndofbaskervil00doyluoft.pdf
Using temporary folder /tmp/7KgInUlnzQ
Attempting: /home/daniel/houndofbaskervil00doyluoft.pdf:12281699
adv_pdf(/home/daniel/houndofbaskervil00doyluoft.pdf) using tempfile /tmp/7KgInUlnzQ/66648-3664.pdf
Pre-processing using mutool.
Successful.
Detected qpdf version >=9
adv_pdf_iterate_objects() complete. Optimised 0 object headers.
adv_pdf_iterate_objects() complete. Optimised 0 object headers, 388 stream contents.
Advanced PDF processing done: Was 12280051, finished 12123363.
Success! 12281699 to 12123363 (0.987107972602162)
daniel@tardis:~$

```

Figure 1: How much Minuimus can shrink a file varies: These two files were reduced to 88 and 99 percent of their original sizes, respectively.

number of other programs, including AdvanceCOMP [3] for compressing specific file types, OptiPNG [4] for compressing PNG images, Jpegoptim [5] for compressing JPEG files, Gifsicle [6] and flexiGIF [7] for compressing GIF files, and Qpdf Tools [8] for converting PDFs.

It is difficult to predict how much Minuimus will shrink your PDFs. It depends on several factors, including primarily how effective the original program was at creating the PDF file in question and how much reduction potential is left for Minuimus. If the PDF file only consists of text, there is little potential for file size reduction. The situation is different if the PDF contains many graphics and images, because they usually offer plenty of downsizing potential.

Another program similar to Minuimus is Leanify [9]. If Leanify exists on your computer, Minuimus will also use it as a helper; the two programs complement each other with their capabilities. This combination gives you even better results.

By default, Minuimus is lossless: It does not reduce the quality of images in the PDF file, but simply optimizes the compression. However, a slight loss of quality is sometimes acceptable if it

results in smaller files. The following command

```
minuimus.pl --jpg-webp FILE.pdf
```

converts the JPEGs in the document to the leaner WebP format, reducing the quality of the images to 90 percent.

Minuimus in Use

After installation, call Minuimus with the command:

```
minuimus.pl FILE.pdf
```

I first tested Minuimus on the December 2022 community edition of *LinuxUser* (a German magazine) [10]. As shown in Figure 1, the file size was reduced from 12.7 to 11.2MB (about 88 percent of the original size). A loss of quality was not noticeable, especially with the images. The original metadata were also preserved.

When I compressed the e-paper with the full version of the December 2022 issue of *LinuxUser*, the results were similar. Minuimus reduced the size of the PDF from 32.8 to 28.7MB (87.5 percent of the original size). I also tried out Minuimus on a scan from the Internet Archive of Arthur Conan Doyle's famous 1902 Sherlock Holmes novel *The Hound of the*

Baskervilles [11]. In this instance, Minuimus only managed to reduce the file size to just under 99 percent of the original (12.3 to 12.1MB). Compared to this, Minuimus shrank a book I had scanned myself from 82.7 to 66.9MB (about 81 percent of the original size). Figure 2 shows the results for all four tests.

Conclusions

Installing Minuimus was uncomplicated, and it was easy to use with shell commands. I did manage to reduce the size of a number of files, but with varying degrees of success. The results depend on several factors, not least how much potential a PDF file has for reducing file size in the first place. Ultimately, every little bit helps when it comes to freeing up disk space. ■■■

Info

- [1] Minuimus: <https://birds-are-nice.me/software/minuimus.html>
- [2] Minuimus download: <https://birds-are-nice.me/software/minuimus.zip>
- [3] AdvanceCOMP: <http://www.advancemame.it/comp-readme>
- [4] OptiPNG: <https://optipng.sourceforge.net/>
- [5] Jpegoptim: <https://github.com/tjko/jpegoptim>
- [6] Gifsicle: <https://www.lcdf.org/gifsicle/>
- [7] flexiGIF: <https://create.stephan-brumme.com/flexigif-lossless-gif-lzw-optimization/>
- [8] Qpdf Tools: <https://github.com/silash35/qpdf-tools>
- [9] Leanify: <https://github.com/JayXon/Leanify>
- [10] *LinuxUser* 12/2022 Community Edition: https://www.linux-community.de/wp-content/uploads/2022/12/lu-ce_2022-12.pdf
- [11] *The Hound of the Baskervilles* from the Internet Archive: <https://archive.org/details/houndofbaskervil00doyluoft/>

```

daniel@tardis: ~
daniel@tardis:~$ ls -s *.pdf
11840 houndofbaskervil00doyluoft_minuimus.pdf 28048 LU2022-12_minuimus.pdf 10936 lu-ce_2022-12_minuimus.pdf 65292 Scan_Buch_minuimus.pdf
12000 houndofbaskervil00doyluoft.pdf 31984 LU2022-12.pdf 12448 lu-ce_2022-12.pdf 80772 Scan_Buch.pdf
daniel@tardis:~$

```

Figure 2: Minuimus' results depend on various factors and turn out differently every time. Here you can see the file sizes before and after processing for the four tests.



Cloning a Debian system with apt-clone

Debian Clone

In the right circumstances, apt-clone can be a simple option for cloning your Debian system. *By Bruce Byfield*

A new system, container, or chroot jail can be created in a few minutes. However, configuring any of them can take hours, especially if you want them to resemble existing systems. Numerous cloning applications exist, notably dd, Partimage, or Clonezilla, but on Debian-derivatives, such as Debian Ubuntu or Linux Mint, one of the simplest tools is a small script called apt-clone [1]. Apt-clone is vaguely reminiscent of Clonezilla, but has the advantage of simplicity because it uses standard command-line tools and basic commands. Apt-clone can also be used as a convenient backup.

Apt-clone belongs to the cluster of small scripts that center on apt-get and apt and their management or ease of

use. New members of these scripts are always appearing and occasionally disappearing, so the Debian stable repository contains 54 of these scripts and Debian unstable repository 64 – a figure that might very well increase before the next official release. Apt-clone itself is over a decade old, which is time enough for its use to spread quietly. Most Debian derivatives include apt-clone in its repositories, but you can also download the source code from GitHub. As I write, though, the latest couple of Ubuntu releases contain a version of apt-clone that appears to have version incompatibilities. In general, though, do not be concerned if you notice that the most recent contributions are a couple of years old. The fact is, git clone is so simple, and

mature enough, that the code rarely needs to be updated.

Creating the Cloning File

Reporting on apt-clone tends to focus on the basics. Even online man pages tend to be incomplete. For this reason, rely only on the man page installed with the script, or the summary available with the --help (-h) option.

Before using apt-clone, update your system with apt upgrade so that you are not dealing with different Debian releases and the original system is as bug-free as possible. Then run

```
apt-clone clone FILE
```

Unless you specify --source DIRECTORY, the output will go to the present working directory. If you have any third party .deb packages, add --with-dpkg-repack at the end of the command so that apt-clone

```
bb@tlvarness:~$ apt-clone clone 2022-12-26-clone --with-dpkg-repack
not installable: libobasis7.3-impress, libreoffice7.4-dict-fr, libobasis7.3-ogltrans, libobasis7.3-extension-nlpsolver, libobasis7.4-gnome-integration, libasas2-2, libobasis7.3-images, libobasis7.4-extension-pdf-import, libobasis7.3-xsltfilter, libobasis7.4-onlineupdate, libobasis7.4-extension-javascript-script-provider, libobasis7.4-extension-report-builder, libobasis7.3-en-us, locales, libreoffice7.3-dict-fr, libreoffice7.4-math, libobasis7.3-oolinguistic, libreoffice7.4-ure, libobasis7.4-en-us, libobasis7.4-writer, libreoffice7.3-ure, libobasis7.3-extension-javascript-script-provider, libreoffice7.3-dict-en, libobasis7.3-kde-integration, libobasis7.3-firebird, libobasis7.4-graphicfilter, libreoffice7.3-writer, libobasis7.3-python-script-provider, libreoffice7.3-en-us, claws-mail-address-keeper, libreoffice7.3-impress, libreoffice7.3-calc, libreoffice7.4-calc, libobasis7.4-librelogo, claws-mail-pdf-viewer, libobasis7.4-firebird,
```

Figure 1: Creating the archive file: Note the number of third-party packages.

Lead Image © Csaba Deli, 123RF.com

```
bb@ilvarness:~$ apt-clone info 2022-12-26-clone.apt-clone.tar.gz
Hostname: ilvarness
Arch: amd64
Distro: bullseye
Meta: gnome-remote-desktop, kde-plasma-desktop, libunity-scopes-json-def-desktop, mate-desktop, plasma-desktop, task-desktop,
task-gnome-desktop, task-kde-desktop
Installed: 2977 pkgs (2619 automatic)
Date: Wed Dec 28 16:27:03 2022
```

Figure 2: An overview of an archive file using apt-clone info.

will attempt to include them in the backup directory. However, be aware that the attempt may not always be successful, because an installable `.deb` package does not automatically have the format and standard required by the Debian distribution. Such packages work perfectly fine in everyday use, but may not be picked up by apt-clone (Figure 1).

Whatever the case, in a few minutes, apt-clone will create a TAR-zipped file. If you are trying to include third-party packages, it will probably take much longer because each third-party package must be analyzed separately. You can read general information about the newly created file using the command:

```
apt-clone info FILE
```

In Figure 2, across seven different desktops, 2,977 packages are installed, but only 2,619 are official Debian packages, which suggests that a perfect clone could be unlikely, although the basic functionality should clone successfully. A different view of the archive file can be had by viewing it in an archive application such as Ark (Figure 3). As you might expect, package sources and apt settings are major categories for the archive, as well as apt-clone's logfile. Preferences include configuration settings from directories such as `.config` or `.local`. In theory, you might be able to unarchive the file and attempt to edit, but to do so is likely to take more effort than is worthwhile. Still, it is never wasted time to know exactly what you are doing instead of blindly following instructions.

Because the archived file contains system-sensitive information, you should encrypt unless you plan to delete it immediately after cloning. The quickest way to do so with reasonable security is to use GPG and keep the file encrypted when you are not actually using it.

How to Clone

Once you have the archive file, install the target system with the same Debian

release as the source system. Using another Debian release may work, but it increases the chances of problems – problems that may cost more time than apt-clone is likely to save. Transfer the archive file to the root directory of the target system, decrypt it if necessary, and run as root:

```
apt-clone restore FILE
```

The process is similar to that of installing packages normally and may take 20 minutes or more on recent hardware, depending on the number of packages involved. It will take even longer if you choose to use `restore-new distro` instead of `restore`, because you will be attempting to upgrade packages at the same time. Probably less can go wrong if you upgrade before restoring.

When the process finishes, the cloning will complete without the need to re-boot, exactly as happens when a new package is installed. Similarly, because the source files belong to root and are copied as root, you should not have problems with permissions. If you do have any problems, using `apt-clone diff` to compare with the source machine's file may help with problem-solving.

Limitations

The main drawback to apt-clone is that is designed for use with official repositories.

If the system you are cloning includes packages installed with `.deb` packages from third-party sources, apt-clone will try to handle them, but sometimes without success. That means that if you have installed, for example, the latest version of LibreOffice from The Document Foundation, you will have to back up the configuration files separately. Similarly, transferring the configuration files to a newer release of the operating system may also have limited success. Probably, the most useful time to use apt-clone is when you are installing a lab with identical workstations or a new home computer that you want to be identical to your old one. When apt-clone meets your needs, it is an elegant little script, but it is never going to be a complete solution for every possible circumstance. ■■■

Info

[1] apt-clone:
<https://github.com/mvo5/apt-clone>

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 cofounder of Prentice Pieces, a blog about writing and fantasy at <https://prenticepieces.com/>.

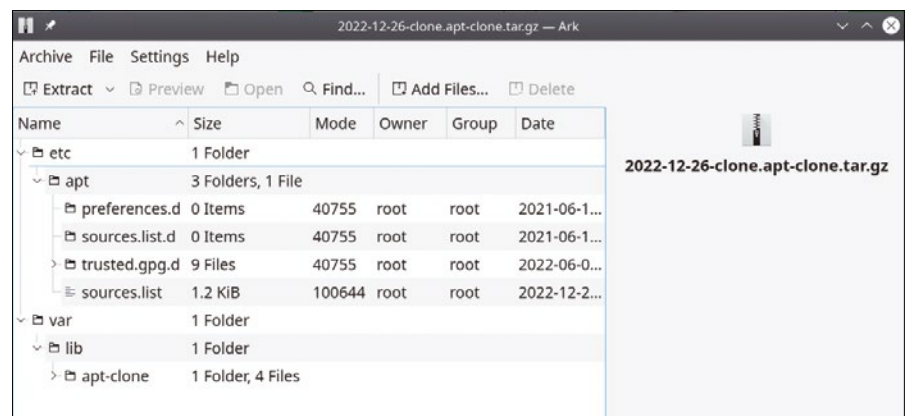


Figure 3: An overview of the archive file's content.

Request Spotify dossiers and evaluate them with Go and R

File Inspector

Spotify, the Internet music service, collects data about its users and their taste in music. Mike Schilli requested a copy of his files to investigate them with **Go**. *By Mike Schilli*

Streaming services such as Spotify or Apple Music dominate the music industry. Their extensive catalogs now cover the entire spectrum of consumable music. Relying on artificial intelligence, these services introduce users to new songs they'll probably like, as predicted by the services' algorithms. Traditional physical music media no longer stand a chance against this and gather dust on the shelves. Of course, this development also means that anonymous music consumption is a thing of the past, because streaming services keep precise records of who played what track, when, and for how long.

On request, Spotify will even hand over the acquired data (Figure 1). If you poke around a bit on their website, you'll find the buttons you need to press to request a copy of these files in *Account | Privacy Settings*, but Spotify takes their sweet time to respond.

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.



From the time of the request, it takes about a week for their archivist to retrieve the data from the files in the Spotify basement, compress them, and post them as a ZIP archive on the

website for you to pick up. After receiving Spotify's email notification, you can then download the data for two weeks and poke around in it locally to your heart's content.

Can I download a copy of my personal data?

Yes, you can!

You can get a ZIP file with a copy of your most relevant personal data by using the automated **Download your data** tool on the **Privacy Settings** section of your account page. The download will include a copy of the following data (if applicable to you):

- Playlists
- Search queries
- Streaming history for the past year
- A list of items saved in your library
- The number of followers you have and the number of accounts you follow
- Payment and subscription data
- User data
- Inferences
- Voice input
- Podcast interactivity
- Family Plan data
- Spotify for Artists data

Figure 1: Spotify lets its users view the data it collected about them.

Exercise

The ZIP file containing the downloaded data includes a JSON file named `StreamingHistory0.json` with the metadata of all the streams you played in historical order (Figure 2). In addition to the song and artist, the entries

```
{
  "endTime" : "2021-10-20 04:21",
  "artistName" : "Foster The People",
  "trackName" : "SHC",
  "msPlayed" : 248030
},
{
  "endTime" : "2021-10-20 04:25",
  "artistName" : "Nothing But Thieves",
  "trackName" : "Futureproof",
  "msPlayed" : 27397
},
{
  "endTime" : "2021-10-20 04:28",
  "artistName" : "Twenty One Pilots",
  "trackName" : "Shy Away",
  "msPlayed" : 174963
},
{
  "endTime" : "2021-10-23 03:34",
  "artistName" : "Cypress Hill",
  "trackName" : "(Rock) Superstar (feat. Chino Moreno & Everlast)",
  "msPlayed" : 185931
},
{
  "endTime" : "2021-10-23 03:48",
  "artistName" : "Culcha Candela",
  "trackName" : "Monsta 2k21 - Radio Edit",
  "msPlayed" : 43929
},
{
  "endTime" : "2021-10-24 00:08",
  "artistName" : "Judas Priest",
  "trackName" : "Lightning Strike",
  "msPlayed" : 209430
},
}
```

Figure 2: JSON data from the author's streaming history.

Listing 1: stats.go

```
01 package main
02 import (
03     "encoding/json"
04     "fmt"
05     "io/ioutil"
06     "sort"
07 )
08 type stream struct {
09     EndTime    string `json:endTime`
10     ArtistName string `json:artistName`
11     MsPlayed   int64  `json:msPlayed`
12     TrackName  string `json:trackName`
13 }
14 const jsonFile = "MyData/StreamingHistory0.json"
15 func main() {
16     bySong := map[string]int64{}
17     content, err := ioutil.ReadFile(jsonFile)
18     if err != nil {
19         panic(err)
20     }
21     data := []stream{}
22     err = json.Unmarshal(content, &data)
23     if err != nil {
24         panic(err)
25     }
26     for _, song := range data {
27         title := fmt.Sprintf("%s/%s", song.ArtistName, song.TrackName)
28         bySong[title] += 1
29     }
30     type kv struct {
31         Key    string
32         Value  int64
33     }
34     kvs := []kv{}
35     for k, v := range bySong {
36         kvs = append(kvs, kv{k, v})
37     }
38     sort.Slice(kvs, func(i, j int) bool {
39         return kvs[i].Value > kvs[j].Value
40     })
41     for i := 0; i < 3; i++ {
42         fmt.Printf("%s (%dx)\n", kvs[i].Key, kvs[i].Value)
43     }
44 }
```

also list the start date and time and the playback duration. Playback duration is particularly interesting because if the user interrupts a stream after a few seconds and fast forwards to the next song, the track probably made it onto the playlist by mistake and was something the user didn't actually like. It will most likely turn out to be a false positive when it comes to putting together music suggestions.

As an exercise, Listing 1 shows a Go program that traverses the JSON data and creates charts featuring the most frequently played tracks. The top three output in Listing 2 shows you my favorite songs – minus the ones that I excluded because they were just too embarrassing to own up to.

To do this, Listing 1 opens the JSON file in line 17 and returns a byte array with its content in the `content` variable. Line 22 passes this to the `Unmarshal` function from the `json` package in Go's standard library, along with a pointer to a `stream` type structure defined previously in line 8. As you know, Go insists on strict type checking. In order for the

JSON parser to create an internal Go data structure from the Spotify data, the format must be known and also match that of the actual JSON format.

The JSON blob provided by Spotify, as shown in Figure 2, consists of an array whose elements each correspond to a streamed track. The artist and track names are stored as strings in the `artistName` and `trackName` fields. `msPlayed` gives you the playback time in milliseconds, while `endTime` has the date and time at the end of playback.

The fields of the `stream` structure in Listing 1 each start with a capital letter, which means that other packages can also access them later on. However, this means that the names are not identical to the variable names in JSON format, each of which starts with a lowercase letter. However, this is no big deal, because Go lets you give a structure a name that can differ from the field name with the `json:` tag.

For example, `ArtistName string `json:artistName`` in line 10 specifies that the artist in the `ArtistName` field is of the string type in the Go structure, and

Listing 2: Top Three Songs

```
Sparks/When Do I Get to Sing "My Way" - 2019 - Remaster (19x)
Falco/The Sound of Music (16x)
Linkin Park/With You (14x)
```

the name used for it in the incoming JSON is `artistName`. This is all you need for `json.Unmarshal()` to dig through all the entries in the JSON file in line 22, because the function has been passed a pointer to what is still an empty array of these `stream` entries in `data`. Using Go's reflection mechanism, the function figures out which JSON structures it needs to work its way through.

Listing 1 counts how many times each song occurs in the streaming history in the `bySong` map defined in line 16. To do this, it uses the title's string as a key and increments the 64-bit integer map value by one for each playback event it finds in the streaming data. At the end, the function then needs to sort the map by the highest integer value in descending order to output the top three.

Sorting Is No Piece of Cake

In a scripting language, sorting the map data would be a snap, but Go offers type safety, and that's why Listing 1 converts the map entries into an array slice of `kv` (for Key/Value) structures whose type it defines starting in line 30. The `for` loop starting in line 35 then needs to slog through the entries of the map and append each key value pair it finds as a `kv` struct to the `kvs` array slice. The slice can then be sorted by Go's standard `sort.Slice()` function. The callback in line 39 tells it that it can determine the desired order of two entries in the slice at positions `i` and `j` by a numeric comparison of the two counters at those positions.

Wow, that's pretty convoluted! At the end, the `for` loop from line 41 goes through the sorted array, outputs the top positions, and terminates after the third value.

Faster with R

Go programs for parsing JSON data and computing statistics are a real pain. Go's type safety requires a disproportionate amount of boilerplate

code here, which scripting languages just elegantly do without. This calls for a classic data wrangling language like R, which takes a more carefree approach, saving programmers a great deal of work. If you don't have R on your machine yet, simply install it on Ubuntu, for example, with:

```
sudo apt install r-base
```

Listing 3 shows a simple application that scans a user's Spotify streaming history, produces a histogram of the actual playing times of the songs they listened to, and displays it nicely as a bar graph (Figure 3). The diagram illustrates that many songs were simply canceled after less than 15 seconds (15,000 milliseconds). In this case, Spotify's suggestion algorithm most likely made a mistake, annoying the listener, who then switched to the next song. Starting at about one minute of playback time (i.e., after 60,000 milliseconds), an almost Gaussian-like bell curve appears, peaking at 220 seconds. Most songs these days are about three and a half minutes long, with the majority being between two and five minutes.

To be able to call Listing 3 at the command line, the shebang statement in line 1 searches for the Rscript program in the shell's search paths and calls the underlying R interpreter with the program code from the listing. Also make sure to mark the file `hist.r` (Listing 3) as executable via the `chmod +x` command.

For an elegant approach to reading the JSON data, Listing 3 uses the `jsonlite` package; you will need to install this up front. After opening an R session (just type R at the command line), the `install.packages("jsonlite")` command loads the package's C++ sources from the Comprehensive R Archive Network (CRAN), compiles them locally, and integrates the library into the local R universe. After that, any R script can use

`library("jsonlite")` to include the new library and call functions from it.

Line 3 reads the JSON data from the streaming history using the `fromJSON` function exported from `jsonlite` and stores it as a `dataframe` in the `jdata` variable. This R standard type is a kind of database table with row-by-row vector values, each spanning multiple columns. In addition to numeric values and character strings, the columns can also contain what are known as factors. In R, these factors are variables with a certain number of possible values, for example, `small`, `medium`, and `large`.

Playback Length Statistics

Listing 3 needs to limit the maximum recorded play length of all the songs it analyzes to five minutes, because my streaming history also included 90-minute audio plays, which distorted the statistics beyond recognition. Filtering is handled by the recoding statement in line 4, which uses the condition `jdata$msPlayed < 300000` to filter out all tracks over 300 seconds playing time from the `jdata` `dataframe` before again assigning the result to the `jdata` variable.

Recoding statements take place at both the row and the column level. The square brackets in line 4 contain the conditions, separated by a comma. The filter applies the first condition to each row, and the second to each column. The result is a `dataframe`, which can have both fewer rows and fewer columns. In this case, however, we only need to remove the rows, not columns, which is why the second part of the condition in square brackets after the comma remains empty. Yes, you need eagle eyes to read and understand R code correctly!

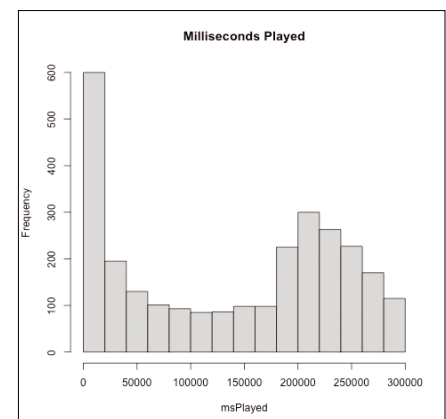


Figure 3: Histogram on playback duration, generated by Listing 3

Listing 3: `hist.r`

```
01 #!/usr/bin/env Rscript
02 library("jsonlite")
03 jdata <- fromJSON("MyData/StreamingHistory0.json", simplifyDataFrame = TRUE)
04 jdata <- jdata[jdata$msPlayed < 300000, ]
05 attach(jdata)
06 png(filename="hist.png")
07 hist(msPlayed, main="Milliseconds Played") 10 detach(jdata)
```

	endTime	artistName	trackName	msPlayed
1	2021-10-20 04:21	Foster The People	SHC	248030
2	2021-10-20 04:25	Nothing But Thieves	Futureproof	27397
3	2021-10-20 04:28	Twenty One Pilots	Shy Away	174963
4	2021-10-23 03:34	Cypress Hill	(Rock) Superstar	185931
5	2021-10-23 03:48	Culcha Candela	Monsta 2k21 - Radio Edit	43929
6	2021-10-24 00:08	Judas Priest	Lightning Strike	209430

Figure 4: The original dataframe from the JSON data.

	endTime	artistName	trackName	msPlayed	hour
1	2021-10-20 04:21	Foster The People	SHC	248030	21
3	2021-10-20 04:28	Twenty One Pilots	Shy Away	174963	21
4	2021-10-23 03:34	Cypress Hill	(Rock) Superstar	185931	20
6	2021-10-24 00:08	Judas Priest	Lightning Strike	209430	17

Figure 5: Filtered dataframe with hours column.

Listing 4: hourly.r

```
01 #!/usr/bin/env Rscript
02 library("jsonlite")
03 jdata <- fromJSON("MyData/StreamingHistory0.json", simplifyDataFrame = TRUE)
04 # only enjoyed songs
05 jdata <- jdata[jdata$msPlayed > 60000, ]
06 d <- as.POSIXct(jdata$endTime, tz = "UTC")
07 jdata$hour <- as.numeric(format(d, tz="America/Los_Angeles", "%H"))
08 songs <- subset(jdata, , select=c(hour, artistName))
09 agg <- aggregate(songs$hour, by=list(artistName=songs$artistName,
10   hour=songs$hour), FUN=length)
11 winners <- agg[order(agg$hour, -agg$x),]
12 winners <- winners[!duplicated(winners[2]),]
```

The very compact listing then creates a histogram for the `msPlayed` entries in the `jdata` dataframe and prepares the counter values for playback durations in a bar graph. This is done by the built-in R function `hist()` in line 7, after line 5 has set the `jdata` dataframe as a reference point and line 6 has set any PNG output files produced in the future to `hist.png`. This ensures that R creates a PNG file of this name with the bar chart at the end of the script.

Hot Group

Would the data in the streaming history also allow conclusions to be drawn about preferences for a certain type of music, depending on the time of day? Listing 4 takes a stab at this and reads the JSON data, extracts the hour of the playback end time as a numeric value from the `endTime` date stamp of each streaming event, and then determines

which artist was played most often within that time window averaged over all streaming days.

Figure 4 shows the original JSON data in the dataframe with all fields as found in the JSON file. Line 5 in Listing 4 discards all songs that have not been played back for at least one minute, to avoid introducing false positives into the statistics. Now the task is to extract the hour of the day from the Spotify timestamp; this is done after adjusting the time zone. Spotify denotes the times as UTC (i.e., GMT), but I listen to the music in the

Pacific Time (PT) zone on the US West Coast. This explains why the `as.POSIXct()` function reads the value as UTC from the JSON data, and the format formatter in line 7 converts it to the `America/Los_Angeles` zone. After doing this, the local time hour value determined with `%H` is available as a string. However, to sort the entries later on, R needs numeric values, which is why `as.numeric()` converts it to a number.

Now the dataframe is available in the `jdata` variable, as you can see in Figure 5. Line 8 in Listing 4 then uses `subset()` to convert the data into a dataframe with just two columns: the artist and the playback hour. R's built-in `aggregate()` function then aggregates all lines for an artist with the same hourly value in line 9. `FUN=length` specifies that the additional aggregation column contains the length (i.e., the number of artist-hour tuples).

293	Zucchero	19	1
294	ZZ Top	19	1
343	Linkin Park	20	11
365	Sparks	20	11
314	Element Of Crime	20	7
317	Eros Ramazzotti	20	6
356	Pink Floyd	20	6
304	Cypress Hill	20	5
321	Foster The People	20	5
319	Falco	20	4

Figure 6: Aggregated counters per hour.

Figure 6 shows an excerpt from this intermediate result. Based on this, ZZ Top was played exactly once at 19.00 hours, while no fewer than 11 entries with Linkin Park pop up at 20.00 hours. There are several different ways to filter out only the top performers from this view (e.g., Linkin Park at 20.00 hours). One method that relies entirely on R's standard functions is as follows: Sort the

```
$ hourly.r
```

	artistName	hour	x
19	Rainbow	0	4
28	Sparks	1	9
30	Sparks	2	10
31	Sparks	3	14
32	Sparks	4	7
33	The Killers	6	1
36	Eros Ramazzotti	7	7
40	Adriano Celentano	8	1
48	Eros Ramazzotti	9	4
56	Alice	10	1
59	Ace of Base	11	1
64	Ace of Base	12	1
68	Amy Winehouse	13	3
72	Foster The People	14	4
75	The Monkees	15	2
76	Ace of Base	16	3
111	Eros Ramazzotti	17	5
161	Falco	18	7
243	Guster	19	7
349	Linkin Park	20	11
460	Jan Delay	21	9
596	Jan Delay	22	10
702	Element Of Crime	23	12

Figure 7: Which artists are played back most frequently at what time?

```
"3P_Custom_Lexus Intenders_US",
"3P_Custom_Lexus Owners_UK",
"3P_Custom_Life Insurance Policyholders / Intenders_FR",
"3P_Custom_Life Insurance Providers_US",
"3P_Custom_Light Beer Drinkers_2Jan2021_US",
"3P_Custom_Light TV Viewers_17Sept2021_US",
"3P_Custom_Likely to Refinance Loans_17Aug2020_US",
"3P_Custom_Lincoln Intenders_US",
"3P_Custom_LinkedIn Users_20Sept2022_US",
"3P_Custom_Lipton Bigelow Buyers_25Aug2022_US",
"3P_Custom_Liquor and Cordial Drinkers_27Nov2020_US",
"3P_Custom_Liquid Investible Assets_$250K-$499K_1Sept2020_US",
"3P_Custom_LotteryandGambling_US [Do Not Use]",
"3P_Custom_Low Carb_07Nov2019_US",
"Inferences.json" 1823 lines --37%-- 681,17
```

Figure 8: Spotify – incorrectly – considers the author to be a light-beer drinker.

dataframe by hour (ascending) and the number of events (descending). Using deduplication, the algorithm then only keeps the first entry per each hour value and discards the rest. The top performers for each hour value remain.

Line 10 sorts the previously generated `agg` dataframe, according to the `order()` function specified in the square brackets. Its first parameter is the (positive) field name for the hour value; the second is the (negative) value for the counter determined by the `length` function. In R, the newly created column by the aggregation function goes by the name of `x` and contains the number of results in this case.

Line 11 runs a `recode` statement over the dataframe now named `winners` and uses `!duplicated(winners[2])` to specify that the second field (i.e., the hour value; R arrays always start with an index of 1, not 0) must be present once only in the result. Consequently, the function will only keep the previously forward sorted highest result for hour values with the associated artist and will discard all others.

That takes care of the list with the most popular bands, as a function of the time of day at Perlmeister Studios! Figure 7 shows the output of the `hourly.r` R program (Listing 4). After midnight, it's

either embarrassing oldies from the 1980s (Rainbow) or, as I vaguely recall, once from 1:00am to 4:00am, every single track by the band Sparks, after devouring a Netflix documentary featuring the band and proceeding to play back their songs for four hours. The next day at work was terrible, of course, but you only live once.

From experience, users can spend days tinkering with R to find the right data structure and methods that will often implement exactly what they want in just three lines. The reasons for this probably include the age of the language, which comes with a sort of anti-Python mindset (“Waddya mean, there’s only one way to do this?”), and the many packages that have been released in an uncoordinated fashion over the decades since the original release. A Google search for a particular problem will often reveal three or four different approaches to solve the same issue. The training book by Robert I. Kabacoff [1] does a good job of explaining some basic procedures, but it is by no means an extensive reference.

More Secrets

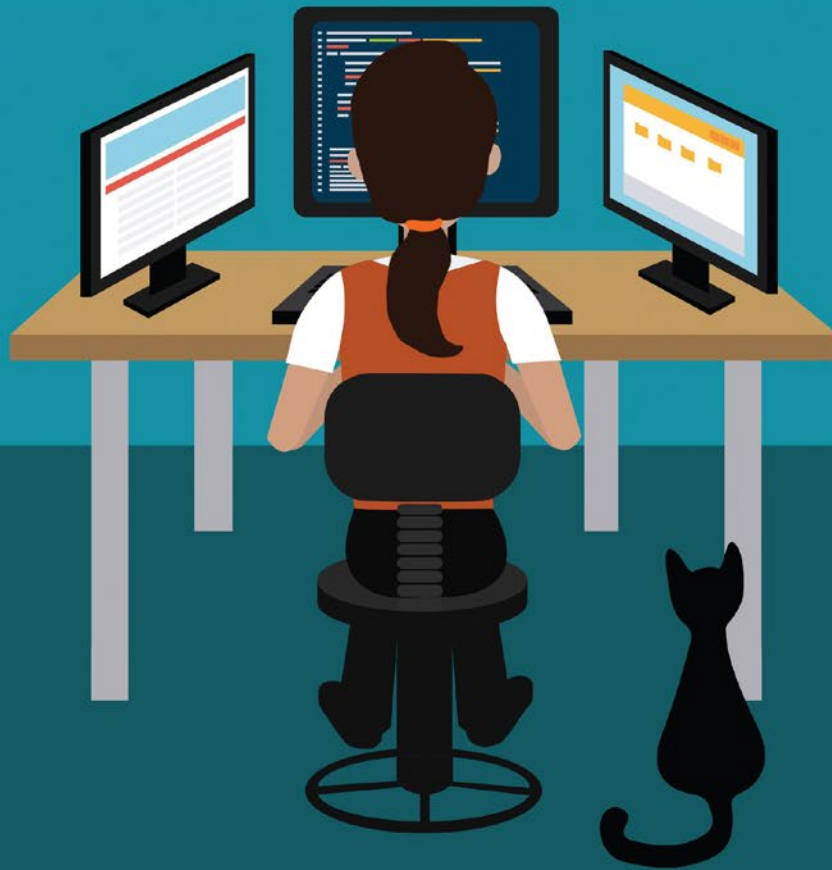
If you rummage further in the Spotify dossier’s ZIP file, you are likely to unearth a few more data treasures. For example, Spotify’s `Inferences.json` file contains ascertained facts about the user – presumably to help Spotify serve up appropriate ads that the listener will also respond to.

In my case, Spotify assumed I had a preference for “Light Beer” (Figure 8), which is a joke and totally wrong – as anyone who knows me can attest to if it came to a pinch! This is a likely explanation if Bud Light ads start popping up on my screen. ■■■

Info

[1] Kabacoff, Robert I. *R in Action, Second Edition*. Manning, May 2015: <https://www.manning.com/books/r-in-action-second-edition>

Want to work from anywhere?



Find remote jobs now!

OpenSource
JOB HUB

opensourcejobhub.com/

Swap snaps for Flatpaks with unsnap

Package Exchange

If you want to move away from Ubuntu's Snap package format, the `unsnap` script removes snaps from your computer and replaces them with Flatpaks where possible. *By Ferdinand Thommes*

The Linux community, like any other community, is not particularly good at welcoming new things. Most people want to retain the status quo. In the words of the philosopher Voltaire, the better is the enemy of the good. Nothing is more dangerous than leaving the good for better things.

Systemd, the still controversial system and session manager, illustrates this concept. Other examples are the new package systems Flatpak, Snap, and AppImage, which *Linux Magazine* previously covered in December 2022 [1]. These package systems are by no means met with unanimous

approval in the Linux community. In particular, the Snap format, which Canonical initially designed for cloud applications and the Internet of Things (IoT), but later ported for desktop applications, is experiencing opposition beyond the basic criticism of new package systems.

If you've become disillusioned with Snap, the `unsnap` script can help you replace snaps with Flatpaks where available on Ubuntu and its derivatives.

Bad Experiences

The Linux community is skeptical of any solo efforts by Canonical due to bad past experiences. The main criticisms leveled at Snap (Figure 1) are the Snap Store's proprietary back end [2] and the fact that Canonical is increasingly pushing the format in Ubuntu, with no official way to remove snaps.

The snap daemon (`snappyd`) is the defined tool for retrieving snaps and all associated metadata from Canonical's servers. In addition, snaps have functional drawbacks that the company has been slow to eliminate. Flatpak, on the other hand, was well received by proponents of alternative package systems. Flatpak is probably the choice of

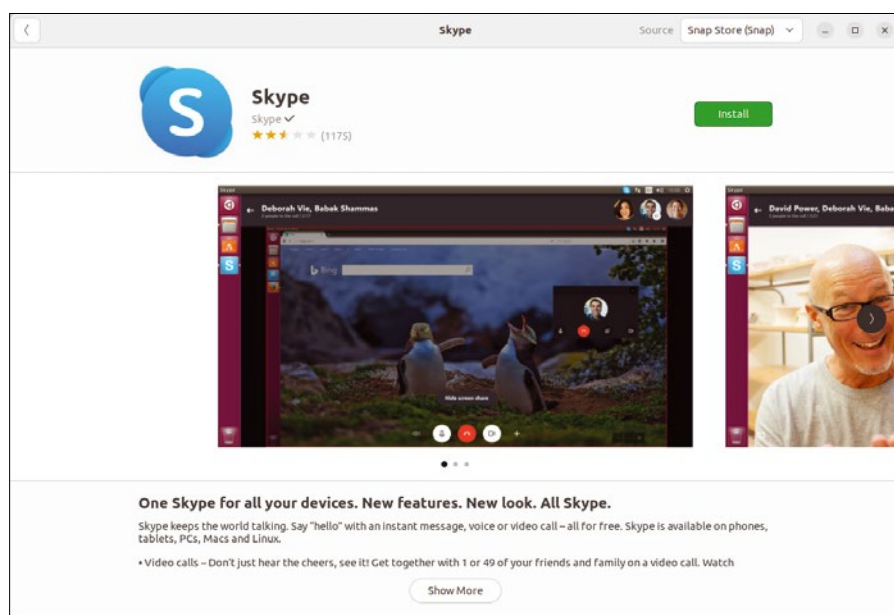


Figure 1: When you install programs via *Ubuntu Software*, you can define the source via a selection box top right. Some packages, such as Firefox, are generally only available as snaps.

most users outside the Ubuntu environment. This could eventually lead to Canonical mothballing Snap again, as has often been the case in the past with in-house developments such as Ubuntu Touch or the Mir display manager.

This situation has led some distributions to reject or even block snaps. One prominent example is Linux Mint, based on Ubuntu LTS, whose developers decided as early as 2020 not to deliver snaps [3]. They also announced that Linux Mint, starting in version 20, would actively prevent the installation of the Snap framework via the graphical package manager, preventing snaps from automatically ending up on the user's system through the backdoor.

Laptop manufacturers System76 (Pop!_OS) and Tuxedo (Tuxedo OS 1) also avoid shipping snaps with their Ubuntu derivatives. Even Ubuntu users have not rallied behind Snap, as suggested by many guides that describe installing Firefox in Ubuntu as a DEB.

unsnap

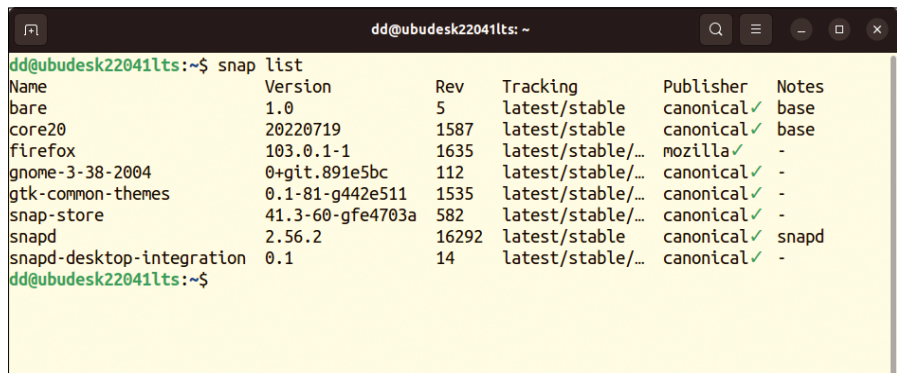
Former Snap co-developer Alan Pope, who left Canonical in 2021 after 10 years with the company, has developed unsnap [4], a script that replaces snaps with Flatpaks where available. The script, hosted on GitHub, has been tested by the developers for use on Ubuntu and all derivatives that offer snapd and packages in the Snap format.

To test unsnap on a freshly installed Ubuntu 22.04 LTS, I set up a few more packages from the Snap Store in addition to the existing snaps via the Ubuntu Software package manager and then listed them just to be sure (Figure 2). I then downloaded the unsnap script from GitHub. To do this, I first had to install the `git` package and then run the command from Listing 1.

Before getting into the nitty gritty, let's take a look at how the script works. Immediately after startup, unsnap creates a log directory in which it generates six additional scripts tailored to the particular system (see Table 1). The `applist.csv` [5] file is used to find Flatpaks that match the installed snaps.

Multiple Steps

To start the script, first change to the appropriate directory by typing `cd unsnap`.



```

dd@ubudesk22041lts:~$ snap list
Name            Version      Rev    Tracking      Publisher  Notes
bare            1.0          5      latest/stable canonical✓  base
core20          20220719    1587   latest/stable canonical✓  base
firefox         103.0.1-1    1635   latest/stable mozilla✓    -
gnome-3-38-2004 0+git.891e5bc 112    latest/stable canonical✓  -
gtk-common-themes 0.1-81-g442e511 1535  latest/stable canonical✓  -
snap-store      41.3-60-gfe4703a 582   latest/stable canonical✓  -
snapd           2.56.2       16292  latest/stable canonical✓  snapd
snapd-desktop-integration 0.1      14     latest/stable canonical✓  -
dd@ubudesk22041lts:~$

```

Figure 2: The `snap list` command lists all snaps installed on the system.

The `./unsnap` command (Figure 3) first informs you that various checks will be run to determine if Flatpak is already installed and Flathub is included, as well as which snaps have Flatpak equivalents.

Next, unsnap creates the helper scripts. If you have a reason not to run one of the helper scripts, you can launch these scripts manually one after the other. Otherwise, call `./unsnap auto` to automatically install the scripts (Figure 4).

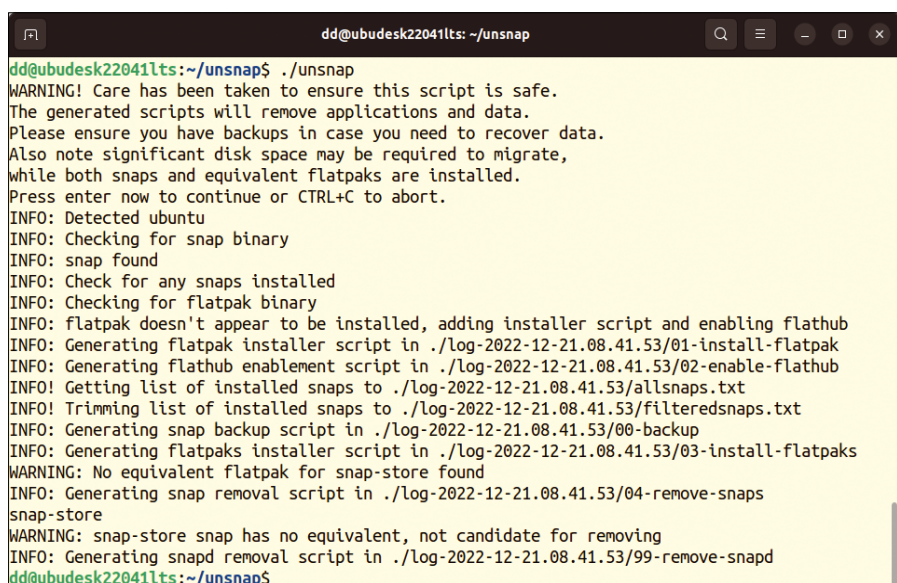
Listing 1: Installing unsnap

```
$ git clone https://github.com/popey/unsnap
```

On my lab system, I installed about 20 snaps, of which unsnap was able to replace about half. However, I intentionally installed snaps such as the Canonical Juju and LXD tools, as well as the Mutt command-line email client, to test how the script responded (Figure 5). In general, the chances of finding a Flatpak counterpart for graphical apps

Table 1: unsnap Subscripts

Script	Function
00-backup	Runs the <code>snap save</code> command for each Snap package to be migrated.
01-install-flatpak	Installs the Flatpak package manager if not already done.
02-enable-flathub	Adds the Flathub repository to Flatpak if not enabled.
03-install-flatpaks	Installs Flatpaks for each identified Snap package, if it exists.
04-remove-snaps	Removes snaps that have alternative Flatpaks.
99-remove-snapd	Removes snapd itself, which removes any snaps that are still installed.



```

dd@ubudesk22041lts:~/unsnap$ ./unsnap
WARNING! Care has been taken to ensure this script is safe.
The generated scripts will remove applications and data.
Please ensure you have backups in case you need to recover data.
Also note significant disk space may be required to migrate,
while both snaps and equivalent flatpaks are installed.
Press enter now to continue or CTRL+C to abort.
INFO: Detected ubuntu
INFO: Checking for snap binary
INFO: snap found
INFO: Check for any snaps installed
INFO: Checking for flatpak binary
INFO: flatpak doesn't appear to be installed, adding installer script and enabling flathub
INFO: Generating flatpak installer script in ./log-2022-12-21.08.41.53/01-install-flatpak
INFO: Generating flathub enablement script in ./log-2022-12-21.08.41.53/02-enable-flathub
INFO! Getting list of installed snaps to ./log-2022-12-21.08.41.53/allsnaps.txt
INFO! Trimming list of installed snaps to ./log-2022-12-21.08.41.53/filteredsnaps.txt
INFO: Generating snap backup script in ./log-2022-12-21.08.41.53/00-backup
INFO: Generating flatpaks installer script in ./log-2022-12-21.08.41.53/03-install-flatpaks
WARNING: No equivalent flatpak for snap-store found
INFO: Generating snap removal script in ./log-2022-12-21.08.41.53/04-remove-snaps
snap-store
WARNING: snap-store snap has no equivalent, not candidate for removing
INFO: Generating snapd removal script in ./log-2022-12-21.08.41.53/99-remove-snapd
dd@ubudesk22041lts:~/unsnap$

```

Figure 3: If you call the script without additional commands, it determines the system status and creates the required auxiliary scripts, which you can then launch manually, if required.

are better, because Flatpak targets the desktop. Snap, on the other hand, also targets the server world.

Rework Needed

My test took around 10 minutes and provided detailed information about

what was happening at all times. The `flatpak list` command (Figure 6) can be used to determine which snaps were converted to Flatpak format. For snaps with a Flatpak equivalent, `unsnap` converted these snaps cleanly, and all of the programs remained functional. The script left the remaining snaps and the infrastructure untouched. Some manual work would be required to completely oust snaps from the computer.

I then reinstalled Ubuntu 22.04, along with some snaps that I knew had corresponding Flatpaks available, including Gimp 2.99.10 Beta, Krita, Spotify, and KeePassXC. In the process, I also installed several additional runtime environments for GNOME and (because of Krita) KDE, as well as other infrastructure packages from Canonical. With this environment using the same procedure as before, I then tested whether `unsnap` also removes the Snap infrastructure when it can convert all snaps to Flatpaks.

I loaded the script and ran it by typing `./unsnap auto`. This time, `unsnap` first had to install Flatpak and connect to Flathub. After doing so, the call successfully replaced all manually installed snaps with Flatpaks. In addition, it successfully disabled the Firefox snap, which was already present during the install.

When checking via `flatpak list`, I noticed one minor disadvantage to this method: All Flatpaks are system-wide and therefore accessible to all users. There is no option to limit the Flatpak to specific users.

Then I ran `snap list` to see if there was anything left of the Snap infrastructure. Lo and behold, the snaps had been deleted, but the infrastructure was still there. Apparently, the script did not execute the last of the six helper scripts (`99-remove-snapd`). I now did this manually by changing to the `~/unsnap/log2022...` directory and running the `./99-remove-snapd` command there. The script removes `snapd` and takes away the parts of the infrastructure that the base does not need.

However, I discovered the Snap infrastructure is already deeply rooted in Ubuntu and that not everything can be removed even manually (Figure 7).

```
dd@ubudesk22041lts: ~/unsnap
INFO: Generating snapd removal script in ./log-2022-12-21.08.42.28/99-remove-snapd
INFO: Auto requested
INFO: Running backup
Set Snap      Age      Version      Rev  Size  Notes
1  firefox     3.26s    103.0.1-1    1635 51.3MB -
1  snap-store 3.26s    41.3-60-gfe4703a 582 3.67MB -
INFO: Backup finished
INFO: Installing flatpak
Hit:1 http://us.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu jammy-updates InRelease [114 kB]
Get:4 http://us.archive.ubuntu.com/ubuntu jammy-backports InRelease [99.8 kB]
Fetched 324 kB in 1s (294 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libappstream-glib8 libfuse2 libmalcontent-0-0 libostree-1-1
Suggested packages:
  malcontent-gui
The following NEW packages will be installed:
```

Figure 4: If the script is called with the `./unsnap auto` command, the process completes automatically.

```
dd@ubudesk22041lts: ~/unsnap
INFO: Installing flatpak packages
INFO: Flatpaks installed
INFO: These flatpaks are now installed:
Freedesktop Platform org.freedesktop.Platform 22.08.4 22.08 system
Mesa org.freedesktop.Platform.GL.default 22.2.4 22.08 system
Mesa (Extra) org.freedesktop.Platform.GL.default 22.2.4 22.08-extra system
openh264 org.freedesktop.Platform.openh264 2.1.0 2.2.0 system
Yaru Gtk Theme org.gtk.Gtk3theme.Yaru 3.22 system
Firefox org.mozilla.firefox 108.0.1 stable system
INFO: Removing snaps
INFO: Snaps removed
WARNING: These snaps are still installed:
Name      Version      Rev  Tracking      Publisher  Notes
bare      1.0          5    latest/stable canonical** base
core18    20221212     2667 latest/stable canonical** base
core20    20220719     1587 latest/stable canonical** base
gnome-3-38-2004 0+git.891e5bc 112  latest/stable/... canonical** -
gtk-common-themes 0.1-81-g442e511 1535 latest/stable/... canonical** -
juju      2.9.37       21315 2.9/stable    canonical** classic
lxd       5.9-76c110d 24164 latest/stable canonical** -
mutt      2-2-6-re1    277    latest/stable snapcrafters -
snap-store 41.3-60-gfe4703a 582  latest/stable/... canonical** -
snapd     2.56.2       16292 latest/stable canonical** snapd
snapd-desktop-integration 0.1 14  latest/stable/... canonical** -
dd@ubudesk22041lts:~/unsnap$
```

Figure 5: There is not always a Flatpak equivalent for installed snaps; `unsnap` lists these cases separately.

```
dd@ubudesk22041lts:~$ flatpak list
Name      Application ID      Version      Branch      Installation
Freedesktop Platform org.freedesktop.Platform 22.08.4 22.08 system
Mesa org.freedesktop.Platform.GL.default 22.2.4 22.08 system
Mesa (Extra) org.freedesktop.Platform.GL.default 22.2.4 22.08-extra system
openh264 org.freedesktop.Platform.openh264 2.1.0 2.2.0 system
Yaru Gtk Theme org.gtk.Gtk3theme.Yaru 3.22 system
Firefox org.mozilla.firefox 108.0.1 stable system
dd@ubudesk22041lts:~$
```

Figure 6: After running `unsnap`, the `flatpak list` command can be used to check which snaps the script replaced with Flatpaks.

Manual

Even without unsnap, the Snap infrastructure can be removed from the computer. To do this, first disable the corresponding systemd services (Listing 2, lines 1 to 3). Then use the command

```
sudo snap remove PACKAGE
```

to delete everything that `snap list` shows you item by item. Last but not least, remove the remnants (lines 5 to 7). Afterwards, the removed

packages can be reinstalled manually as DEBs or Flatpaks.

Conclusions

On Ubuntu and its derivatives (if they use Snap), unsnap lets you swap installed snaps directly for Flatpaks. To do this, an equivalent Flatpak must be available, which is very often the case with graphical applications. With a little manual work, the Snap infrastructure can also be removed.

However, the project warns on GitHub that the software is still at a pre-alpha stage. I attempted to get a statement from Alan Pope, but was unable to do so by the time this issue went to press. I know of many users who have successfully used unsnap; and I did not experience any problems myself. However, to be on the safe side, I recommend using the manual removal option and then re-installing the removed packages in your desired format. ■■■

Author

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

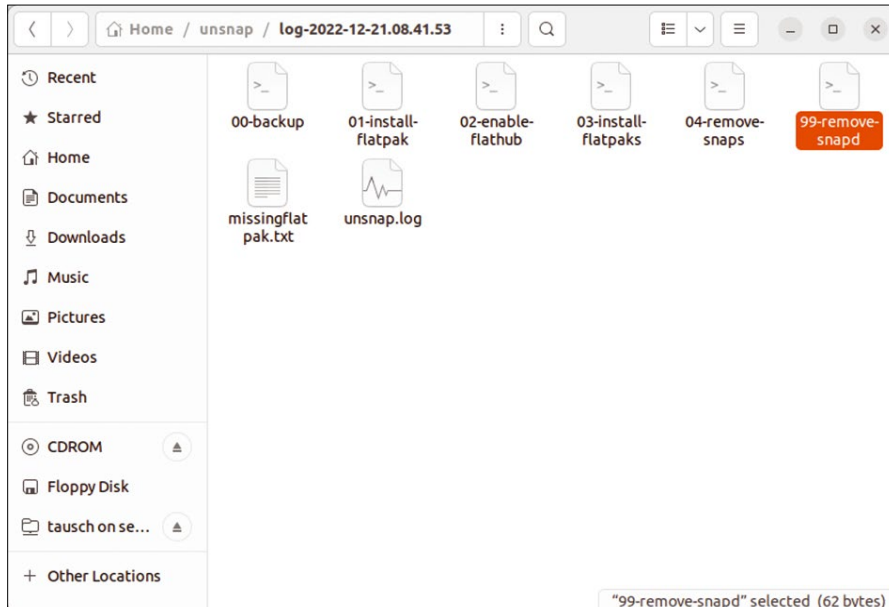


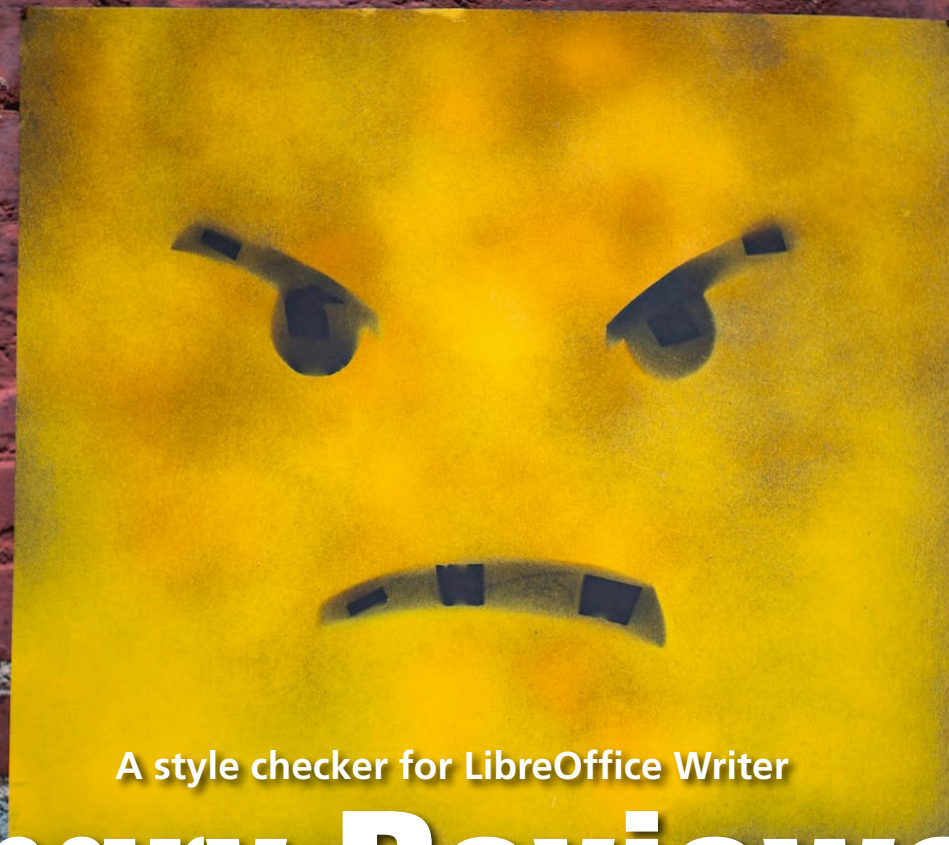
Figure 7: I had to follow up with the `./99-remove-snapd` script to remove as much of the Snap infrastructure as possible. Nevertheless, some remnants remained.

Listing 2: Removing unsnap

```
01 $ sudo systemctl disable snapd.service
02 $ sudo systemctl disable snapd.socket
03 $ sudo systemctl disable snapd.seeded.service
04 [...]
05 $ sudo rm -rf /var/cache/snapd
06 $ sudo apt autoremove --purge snapd
07 $ rm -rf ~/snap
```

Info

- [1] “Innovative Linux Package Managers” by Erik Bärwaldt, *Linux Magazine*, issue 265, December 2022, p. 72
- [2] Snap Store: <https://snapcraft.io/store>
- [3] Mint without Snap: <https://linuxnews.de/2020/06/02/linux-mint-20-ohne-snapd/>
- [4] unsnap: <https://github.com/popey/unsnap>
- [5] applist-csv: <https://github.com/popey/unsnap/blob/main/applist.csv>



A style checker for LibreOffice Writer

Angry Reviewer

The Angry Reviewer style check can be used to evaluate and improve any type of writing, including academic articles and grant applications. *By Bruce Byfield*

LibreOffice Writer has hundreds of extensions [1]. Many are cosmetic, trivial, or specialized. However, a handful are major additions to functionality. In the past, some, such as PDF export, have eventually found their way into Writer's default interface. Eventually, the Angry Reviewer style checker [2] may do just that. Although intended to improve academic articles and grant applications, Angry Reviewer is comprehensive and practical enough to be useful for any type of writing. You might call it a Linux version of the much-advertised Grammarly [3], although its feedback might be presented more usefully.

Angry Reviewer is available on the Angry Reviewer website [4], but the

Writer extension has the advantage of allowing you to work offline and run Angry Reviewer without having to cut and paste. Like all extensions, it can be installed with *Tools | Extension Manager*. When Writer is restarted, Angry Reviewer is added to the top-level menu. When *Check This Text* is clicked (Figure 1), a second Writer document is opened with feedback. The two documents can then be placed side by side as you evaluate the feedback and make changes.

Angry Reviewer in Action

When I taught university composition, I told students that if they knew enough about English to effectively use a spell checker or style checker they didn't

need one. Too often such tools lead readers astray with their authoritative-sounding feedback – as in the case of one student who began an essay on Gloria Steinem with “For centuries, women have been depressed by men” (which many women insist was not a mistake). The problem is that most of these tools are prescriptive, treat grammar and style as fixed forms, and advocate standards that are several decades behind the times.

By contrast, Angry Reviewer is not only more flexible, but explains enough that writers may actually learn a thing or two while getting a quick fix. Citing sources such as *On Writing Well: The Classic Guide to Writing Nonfiction* and

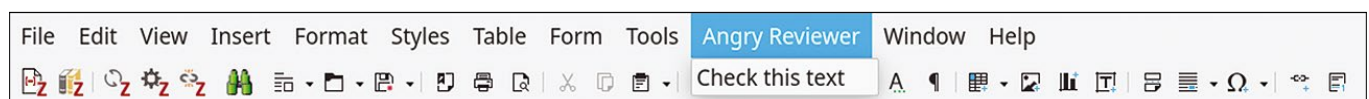


Figure 1: Angry Reviewer adds a listing to the top-level menu.

the all-purpose classic *The Elements of Style*, the page lists the following rules, followed by examples:

- *Don't hype. Avoid words like novel, highly, clearly, greatly. Better still, avoid all adverbs.*
- *Don't use clichés. In a nutshell, by and large, they are clear as mud.*
- *Don't use "very" very often. Usually, there is a better word for it.*
- *Be concise. Avoid phrases like "by means of, despite the fact that, in order to."*
- *Avoid negatives. For example, use "unable" instead of "not able."*
- *Avoid redundancy. For example, use "investigate" instead of "conduct an investigation of."*
- *Use active voice. Although not always possible, most of the text should be in active voice.*
- *Avoid inappropriate language. Keep words like "really, actually, pretty much" for social networks.*
- *Avoid rare words and latinisms. Non credo all readers know the meaning.*
- *Keep abbreviations to minimum. Abbreviations are hard to read, consider just spelling it out.*
- *Beware of zombie nouns [5] [nominalizations, or nouns made from verbs, adjectives, or other nouns]. Utilization of nominalization is causation of distraction.*

Overall, these rules encourage clarity, precision, and conciseness – virtues in any writing. In the feedback to a piece of writing, these rules are offered as suggestions, often with words such as “consider” and “might” that make clear that there is room for differences of opinion. After running this article through Angry Reviewer, one piece of feedback I didn't apply was to “Avoid constructions with ‘it is’ since they obscure the main subject and action of a sentence” (though the advice is good in general). All I would add to Angry Reviewer's list of rules is George Orwell's famous ending to his similar list of rules in *Politics and the English Language* [6]: “Break any of these rules sooner than say anything outright barbarous.”

The abbreviation PDF occurs only a few times. Since abbreviations are hard to decrypt, just spell it out each time. It is easier to read a few words than to search for meanings of abbreviations.
 Paragraph 2. Check if the word "fact" is actually applied to a fact. According to *The Elements of Style*: "Use this word only of matters of a kind capable of direct verification, not of matters of judgment."
 Paragraph 2. Consider if adjective "comprehensive" really adds anything here.
 Paragraph 2. Avoid constructions with "it is" since they obscure the main subject and action of a sentence.
 Paragraph 2. Using "You" might be inappropriate in academic writing. Consider using "One", e.g. "One can see...".
 Paragraph 3. If the word "new" refers to the results or methods, editors and reviewers often dislike such claims. Consider explaining novelty in some other way. Some helpful words are "innovative", "original", "alternative", "previously unknown".
 Paragraph 3. The sentence seems to be too long. Consider shortening or splitting it in two.

Figure 2: Angry Reviewer's feedback for the first three paragraphs of this article's first draft.

In other words, if you have a solid reason for ignoring Angry Reviewer's feedback, go ahead and do so. For example, if you are writing dialog in fiction, or quoting someone, spoken English is often more ungrammatical than written English, and these rules might not always be appropriate. In another example, when I applied Angry Reviewer to this article (Figure 2), I did not follow the suggestion to spell out “PDF,” because my audience can be expected to know the abbreviation. Similarly, I did not change “you” to “one” because that would be too formal (and might be changed by my editor anyway). However, I did divide a sentence flagged as too long. I also considered and sometimes changed several word choices that were questioned. In addition, I observed that I sometimes use unnecessary adverbs and stock phrases such as “By and large” that amount to clichés, and made a mental note to avoid both habits in the future. Overall, I consider Angry Reviewer a useful tool, with even the feedback I rejected forcing me to be more analytical in my editing.

I also noticed that Angry Reviewer ignored swear words and slang, and that, unlike Writer's spell check, it did not catch duplicated words or extra spaces. However, the current release is only 1.2, and meanwhile, running spell check and Angry Reviewer one after the other is not much of a burden.

My only criticisms of Angry Reviewer are that it is added to the main menu, when it belongs in the Tools

menu (perhaps with an icon on the toolbar), and I would appreciate the option to present each piece of feedback one after the other, with the relevant paragraph displayed. But these come later. Angry Reviewer's feedback is useful no matter how it is presented, and I recommend it for writers of all levels of experience. ■■■

Info

- [1] LibreOffice Writer extensions: <https://extensions.libreoffice.org/en/extensions/>
- [2] Angry Reviewer: <https://github.com/anufrievroman/libreoffice-angryreviewer>
- [3] Grammarly: <https://www.grammarly.com/>
- [4] Angry Reviewer website: <https://www.angryreviewer.com/>
- [5] Zombie nouns: <https://archive.nytimes.com/opinionator.blogs.nytimes.com/2012/07/23/zombie-nouns/>
- [6] Orwell, George. *Politics and the English Language*: <https://www.plainlanguage.gov/resources/articles/politics-and-the-english-language/>

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 cofounder of Prentice Pieces, a blog about writing and fantasy at <https://prenticepieces.com/>.

LINUX
MAGAZINE

ISSUE 263 - OCTOBER 2022

Build an IoT Linux
Shrink your OS to fit the device

FREE ARCHIVE DVD HUGE SAVINGS! £39.90 VALUE!
ALL 262 ISSUES ON A SEARCHABLE DISC!

LINUX
MAGAZINE

ISSUE 262 - SEPTEMBER 2022

WHAT'S NEW IN UBUNTU 22.04

Beyond 5G
Imagining an open future for mobile networks

FLUX Beamo
Cool laser cutter with Rasp Pi on the inside

Free Social Media Tools
Get connected without

Manuskript
Planning a novel

FREE DVD

LINUX
MAGAZINE

ISSUE 264 - NOVEMBER 2022

Linux Android Backup
Save smartphone data to a Linux PC

Rocky Linux
debian 11.5

FREE DVD

LINUX
MAGAZINE

ISSUE 265 - DECEMBER 2022

Artificial Intelligence
Real-world machine learning

LINUX
MAGAZINE

ISSUE 265 - DECEMBER 2022

Quantum Computing
From one at?

CircuitMess Nibble: Get started with microcontroller programming

archlinux
manjaro

FREE DVD

LINUX
MAGAZINE

ISSUE 266 - JANUARY 2023

Raspberry Pi Tricks
Go wireless with Bluetooth

AlmaLinux
ubuntu

FREE DVD

LINUX
MAGAZINE

ISSUE 266 - JANUARY 2023

Generative Adversarial Networks
Forged art and imaginary faces: Computers teach computers to lie

Overlay Network
Keep the spies out of your business

Knowledge Management with Logseq

catgets
Make your Linux app multilingual

Lynis
Find vulnerabilities before an attacker finds them

Build Your Own 3D Bingo Game

FOSSPICKS
10 TERRIFIC FREE TOOLS!

WWW.LINUX-MAGAZINE.COM

LINUX
MAGAZINE

ISSUE 266 - JANUARY 2023

Zing: Zero packet ping contender

deb-get: Easy repository for third-party .debs

Optimizing Battery Usage for a Rasp Pi Pico

Heimer: Manage your menu choices with a mind map

10 GEMS FOR YOUR FOSS TOOLKIT

WWW.LINUX-MAGAZINE.COM

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!

Follow us



@linux_pro



Linux Magazine



@linuxpromagazine



@linuxmagazine

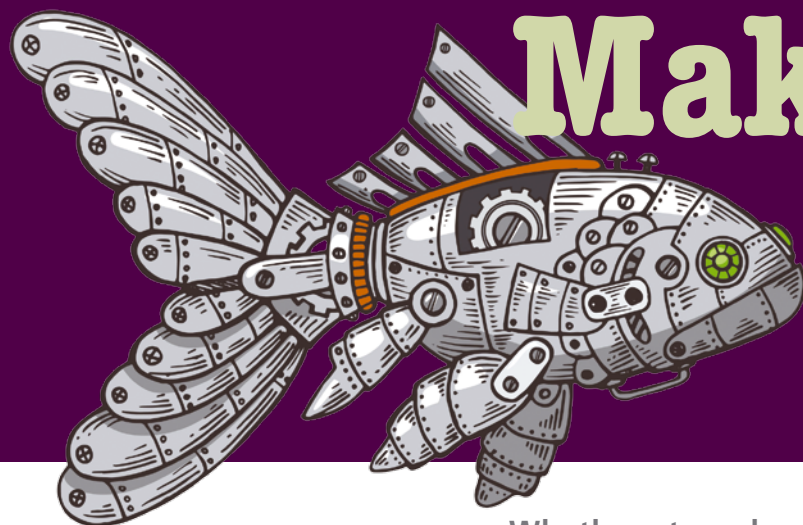
Need more Linux?

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





MakerSpace

Raspberry Pi
automated fish feeder

Fish Food

Whether at work or on vacation, every pet lover worries about how to take care of their little roommates in their absence. What aquarium owners need is an automatic feeder. *By Swen Hopfe*

Aquarium feeders have been around for a long time, but after my recent research, none of the devices really seemed practical to my mind. Very few devices provide feedback about malfunctions, which could be a nasty surprise when you return home. I created this automatic feeder project for my aquarium for peace of mind.

A Raspberry Pi Zero is a good choice as a control center for a fish feeder: It comes with everything you need to control the mechanical system and to grab images with a Pi cam (Figure 1). It is also compact enough to fit in a small housing. Integrating a Pi Zero W with your wireless home network will allow you to post various captured images online while you are on the road.

The goal was a simple setup that was not prone to errors. Therefore, the mechanical control system only uses one small servomotor. The circuitry of the electronics is just as simple. A DIY control system ensures a certain level of convenience. Two feeders add variety and are quite sufficient to bridge an absence of time equivalent to a three-week vacation.

Getting Started

To get the Zero W up and running, you need to download a new Pi OS image and transfer it to a microSD card. After plugging in a screen and keyboard, you can configure the boot options and network settings. With no need for a desktop later, you can choose automatic login over the command-line interface (CLI) and enable the SSH server and the camera interface. After enabling SSH and with wireless access to your home network, you don't need to connect any peripherals to the Zero W, and you can



Figure 1: The DIY feeder made with a Raspberry Pi and a camera on the side of the aquarium.

Listing 1: crontab

```
# m h dom mon dow command
# Sunday till Friday
0:*17 * * 0-5 python3 /home/pi/scripts/feed.py 1
# Saturday
0:*17 * * 6 python3 /home/pi/scripts/feed.py 2
# Every minute
*/1 * * * * python3 /home/pi/scripts/feed.py 0
```

access it right away from another computer on the network. All further settings can then be configured in a terminal window on the other computer.

The Pi needs a hostname that reflects its task; in this example, it is named *feeder*. The next step is to create a separate `scripts/` folder for the project files in your home directory, where you will later put the `feed.py` control script written in Python 3. Instead of loading it as a memory-resident program when the Zero W is started, it is called repeatedly at defined intervals by cron.

All in Good Time

The schedule for the feedings is set in the Zero W's crontab (cron table), which is the file that manages scheduled task processing. The plan for the feeder is shown in Listing 1, which you can create with `crontab -e [1]`.

Crontab specifies feeding every day at 5pm. On Saturday, the script starts with the 2 option on the command line, which releases an alternative feed from the second slot of the vending machine once a week. The third entry in the cron table with option 0 starts the operation display (the on-board LED flashes once a minute) and is used to read remote control codes from the web – but more on that later.

Parts List

- Raspberry Pi Zero W (1/2)
- Pi camera rev 1.3 or higher
- Case for the Pi and Pi camera
- Camera connection cable
- Mini servo (MG90 or comparable)
- Prototyping breadboard (PCB breadboard)
- Two push buttons
- LED
- Various resistors
- DIY feeding tower

What You Need

Commercial feeders often rely on relatively complicated mechanics, such as a carousel of loadable compartments. In contrast, I want this feeder system to be as simple as possible, with a view to the DIY system remaining manageable

and working safely during periods of absence from home. That said, I did want a certain level of convenience, both in terms of operation and functionality and for the benefit of my stay-at-home pets. Therefore, at least two feeder tubes are available for different feeds (Figure 2).

Other considerations were how to ensure – from a distance – that my pets really were receiving their meals. Trouble-free operation from a computer perspective does not give you any feedback as to whether the feed is getting into the aquarium, so I got the idea to record the fish feeding with a camera and serve up a snapshot on the web.

The Zero W comes with on-board tools to accomplish this task, and you can connect a Pi camera to capture an image a short wait after feeding. All you need is a view of a section of the aquarium where the vending machine sits. For protection, the Raspberry Pi and camera are each housed in a small case. The number of other parts in the

project is manageable (see the “Parts List” box).

Construction

In the solution I had in mind, the feed is provided by a small storage tower that drops the food from the top with a little help from gravity through an opening in the aquarium cover. You can build a feeder tower like this with parts from your local hardware store or order the materials online from craft stores. In my case, a desk pen holder served as a starting point; this part is readily available in many stores and are often made of transparent plastic, which is ideal for monitoring the fill level.

Now it's time to saw and drill holes. After that, I added the servo and the mechanical system. The whole thing was then mounted on a base plate, which also bears the Zero W and a small circuit board with a status LED and control buttons (Figure 3).

The feed outlet is operated by a specially developed mechanism. Depending on the direction of rotation of the servo, one of the two tubes of the pen holder opens. For this purpose, a straight shaft passes through both tubes, with a closure for each of the two openings sitting on the shaft.

Because of the angular offset, only one of the two shafts can open at any given time during a short left or right turn, and

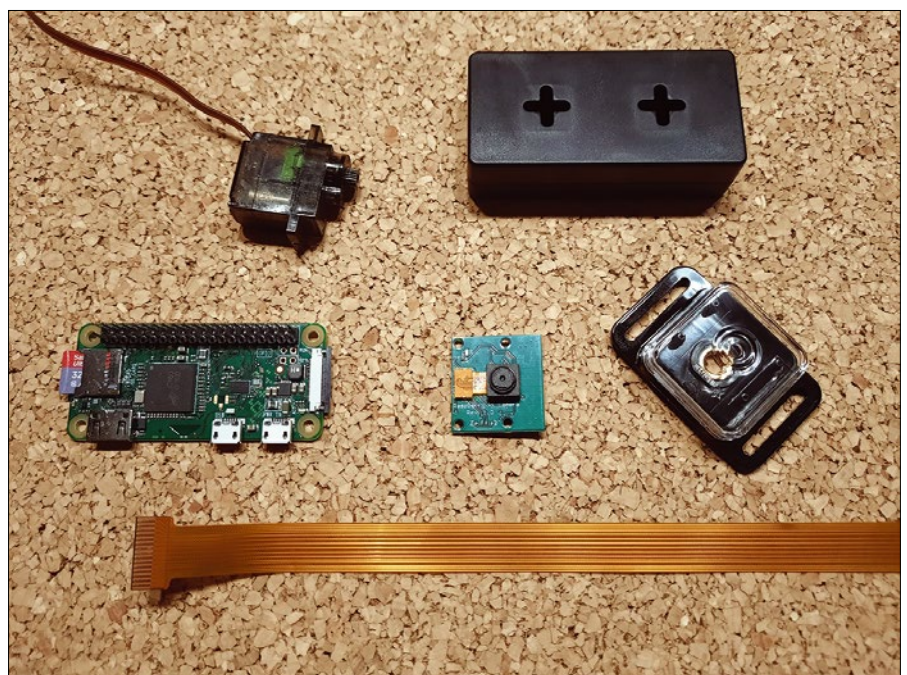


Figure 2: The components you need. A transparent desktop pen holder acts as a feed silo in this setup.

it is closed again automatically a short time later. Small paddles act as the closures to avoid a shaft remaining permanently open and ensure that the desired amount of feed is delivered (Figure 4).

The 160-liter aquarium with low fish stock levels does not need more than a heaped teaspoon of dry food every day. The supply in the vending machine will be fine for a three-week vacation. If you need a larger reserve, adjust the setup to

your needs in terms of diameter and height of tubes.

Control

The feeding process starts automatically after applying the supply voltage. As specified in the Raspberry Pi crontab, the vending machine feeds the fish once a day. On Saturdays, feed is taken from the second shaft by rotating the servo at the same angle, but in the opposite

direction. Once a minute the housing LED flashes a visual ready signal. To track all events on the device, the LED uses various flash codes (Table 1).

What you can't see in the circuit diagram (Figure 5) are the two push buttons on the case, which are connected to the Zero W general purpose input output (GPIO) pins to trigger a feed in the first shaft and for a controlled system shutdown.

Because the feeder is on the local network, it can be maintained over the network, as well. To do so, log in to a terminal over SSH with:

```
ssh pi@feeder
```

You can then make changes to the crontab or control script. If needed, you can disable the automatic feeder by calling:

```
python3 /home/pi/scripts/feed.py 4
```

The 4 option tells the device to enter maintenance mode without any further actions.

Observations

One interesting and positive side effect is the ability to see whether the fish are getting any feed from the feeder. The control script triggers a snapshot through the aquarium glass with a delay of 10 seconds after operating the servomotor. The simple Pi Cam Rev 1.3 serves this function well. Because I only need to grab an image of a section of the front of the aquarium, image quality is secondary.

The connection from the vending machine on the top of the aquarium to the Pi cam on the front of the aquarium requires a long camera cable (about 30cm for my setup) that depends on the height of the aquarium cover. A narrow aluminum plate fixes the small camera

Table 1: LED Flash Codes

Signal	Status
1x long	Live, flashed every minute
1x medium	Shaft 1 opening (plus photo)
2x medium	Shaft 2 opening (plus photo)
3x short	Single snapshot
3x very short	Shutting down

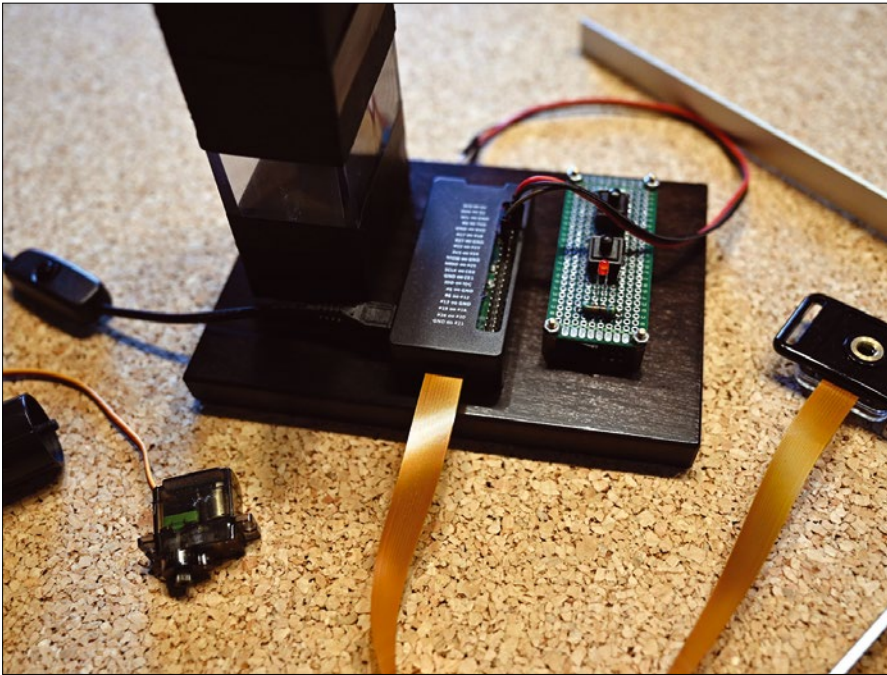


Figure 3: The feed tower has two separate chambers, which are operated by a mechanical system.

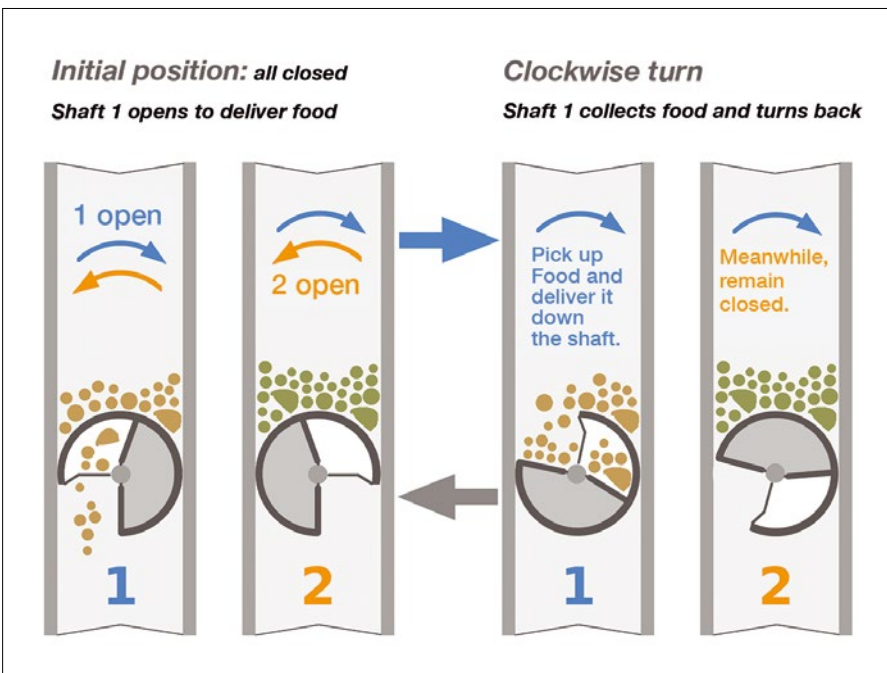


Figure 4: The outlet mechanism: Depending on the direction of rotation, feed is delivered from one of the two chambers.

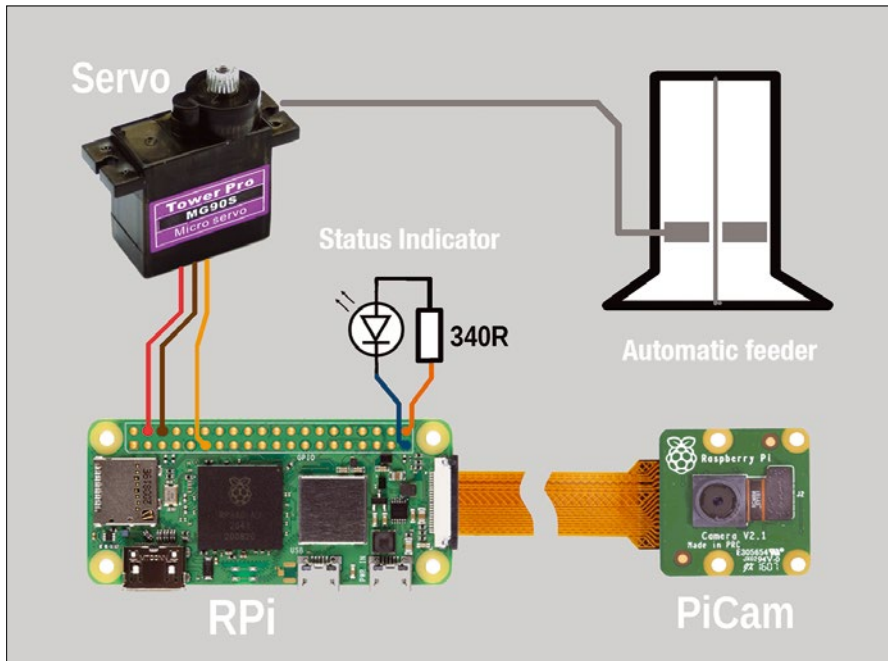


Figure 5: The Zero W controls the servomotor and the LED from the GPIO pins. The circuit diagram shows the wiring setup.

housing a few centimeters in front of the screen.

To get sharp images, you need to set the shortest possible focal length on the

Pi cam. To do this, turn the adjustment ring of the small lens very carefully with a tool and grab a few test images to check the setting.

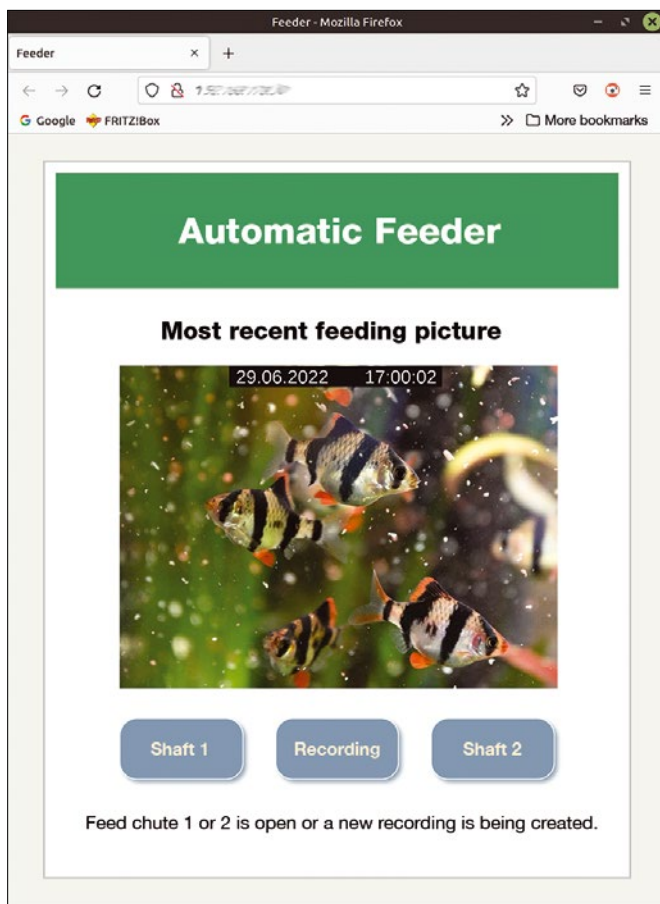


Figure 6: Monitoring over the web: The feeder releases the food, and the fish flock to devour it.

three buttons, is implemented in PHP and resides in the Raspberry Pi's `www` directory. The code for this and the Python script to control the feeder is on the project's GitHub page [3], and an English version is on the *Linux Magazine* download site [4].

To access the web interface on the local network, simply enter the local IP address of the Zero W in a web browser. To monitor the system remotely while on vacation, you need to enable the server on the Zero W in your router for the web and assign a permanent IP. However, if you already use a website with FTP access, you can alternatively upload the current feeding image to your own web space and view it there.

Conclusions

The automatic feeder has already fed my fishy friends without any glitches during two short trips. My goal was a DIY solution from the outset, so competing with commercially available solutions was never important. Instead, I wanted to build a device optimally adapted to my own needs: one that worked reliably and allowed me to check how it was working. I definitely did not want to compromise on the care of my pets.

However, even the best Internet-based feedback is of little use if you have no one onsite who can intervene in an emergency. It's always great to have the support of your family or neighbors to water the flowers or even take care of the pets in your aquarium if you're in a pinch. ■■■

Info

- [1] `crontab(1)` man page: <https://linux.die.net/man/1/crontab>
- [2] `lighttpd`: <https://www.lighttpd.net>
- [3] Project on GitHub: <https://github.com/swenae/feeder>
- [4] Download page for English code: <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.



MakerSpace

Predicting the productivity of a solar array with Perl

Solar Power Leashed

A forecast service and some Perl magic help predict the solar power yield of a residential photovoltaic array. *By Rubén Llorente*

The current energy crisis has led many homeowners to invest in solar power. More people than ever are choosing to install photovoltaic generation systems on their rooftops for domestic use. Depending on how much money is invested on such systems, homeowners could see their power bills cut in half or even have them reduced to nothing.

Solar is not all advantages, though: Power generation represents a huge

upfront investment that will take more than a decade to recoup. Another huge disadvantage, however, is that photovoltaic arrays are only productive if the weather is willing to cooperate (Figure 1).

A house equipped with a photovoltaic set (Table 1) might become truly self-sufficient, as long as it is well managed. However, to optimize the use of solar power, it is important to predict the daily power yield of the system. If you have access to an accurate forecast, you can plan ahead.

Imagine, for example, that you are planning to turn your yearly fruit yield into jam in a given week. That undertaking is an energy-intensive task. If you know tomorrow's weather is not going to cooperate with your solar, but the day after tomorrow you will have favorable conditions, you can adjust your schedule and take advantage of the sun. On the other hand, if your photovoltaic installation is off-grid (in island mode), you will need to know in advance which days are going to be bad and how long a generator would need to run to compensate for cloudy weather.

Keep in mind that using power-hungry appliances while the sun is not shining might still make some sense. See the “Solar Power Without the Sun” box for more information.

Solar Irradiance Forecast

Forecast services that give you an estimation of the power yield of a solar



Figure 1: Two photovoltaic arrays provide more power than a regular house needs – but only if the weather cooperates.

Solar Power Without the Sun

Solar systems usually work alongside the traditional power grid. The idea is that the house will pull power from the solar system and, if solar yield is not enough to satisfy the demand, get as much power from the grid as necessary to have the sum of solar power and grid power equal to the load.

Solar systems only generate power while the sun is shining, but people often need electricity during off-productive hours. Two main strategies are used by homeowners to benefit from solar arrays after sunset. The first is signing a contract with a power supply company, which will buy the power you produce while you are not using it. This way, if you are not home during the day, your solar arrays will produce power for your power supply company. When you return home at night and turn on an electric stove, you will buy power straight from the grid, but the hope is that the money generated during the day will offset the cost of purchasing electricity during dark hours. This approach has the advantage of being cheap (i.e., it does not require the owner to build out a special infrastructure); however, it is more beneficial for power suppliers than for homeowners. Typically, power distributors buy your power for much less than they sell it back to you.

The second strategy is to add a battery set to the solar system (Figure 2). The objective is to have the solar arrays charge the batteries during the day so you have energy stored for when you need it. The advantage of this approach is that it is more flexible and makes you less dependant on regular power suppliers. If your area experiences regular blackouts or power (load) shedding (a temporary reduction in supply of electrical power), you will be able to use battery power. The main disadvantages are the huge upfront cost of battery banks and the increased complexity of the deployment. Also, not all solar systems equipped with batteries are blackout proof, and some will cease operating if the grid goes down because of the way they are designed.

Both strategies benefit from proper planning of power consumption. Somebody who sells power to the grid will still find it preferable to use electricity while solar is working at its best, instead of doing so at night and having to buy power – even if the costs are partially offset by the power sold earlier. On the other hand, the owner of a solar system with a battery bank will be better off using electric power at noon rather than draining energy from the battery needlessly at night.

Table 1: Hardware Budget*

Item	Quantity	Total Price (EUR) [^]
Hybrid inverter	1	1,730.30
Panel (450W)	13	3,303.30
Battery module (5kWh)	3	7,314.45
Total		12,346.05

* 5,910W photovoltaic system, wiring excluded.
[^] Prices are roughly equivalent in US dollars.

array are great for two reasons. First, they allow you to simulate the performance of a solar array you intend to install before you make the investment. It is much better to know that the panels you intend to install are going to under perform or over perform before you spend your money. Second, they allow you to schedule your power expenditure so as to optimize the usage of your system.

A very useful forecast service free of charge for hobbyists, students, and academic research is Solcast [1]. The biggest advantage of the service is that it allows you to download solar irradiance predictions in machine-parseable formats, which can then be fed to scripts and programs to extract meaningful data.

After signing up with Solcast, you are expected to define a *Site* before you start using the service (Figure 3), which is just a fancy way of saying you need

to provide your location to Solcast so they can send you that location's forecasts. To complete your information on the *Site* tab, you need to provide the orientation angle of your panels, their nominal power output, the vertical angle of the panels, and an efficiency estimation. It is safe to assume 90 percent efficiency for new systems, with a 2 percent efficiency loss per year.

Solcast also provides an API key (Figure 4) you can use to download forecasts automatically. Your site's page on the Solcast website will provide plenty of documentation and a wizard for generating valid HTTP calls. To save time, the following example downloads 72 hours of forecast data for the specified site:

```
https://api.solcast.com.au/
  rooftop_sites/$YOUR_SITE_ID/
  forecasts?hours=72&format=csv&
  api_key=$YOUR_KEY
```



Figure 2: A Turbo Energy hybrid inverter (1) paired with a battery bank in a rack (2). The photovoltaic panels charge the batteries with excess production, so the owner can use battery power after the sun sets.

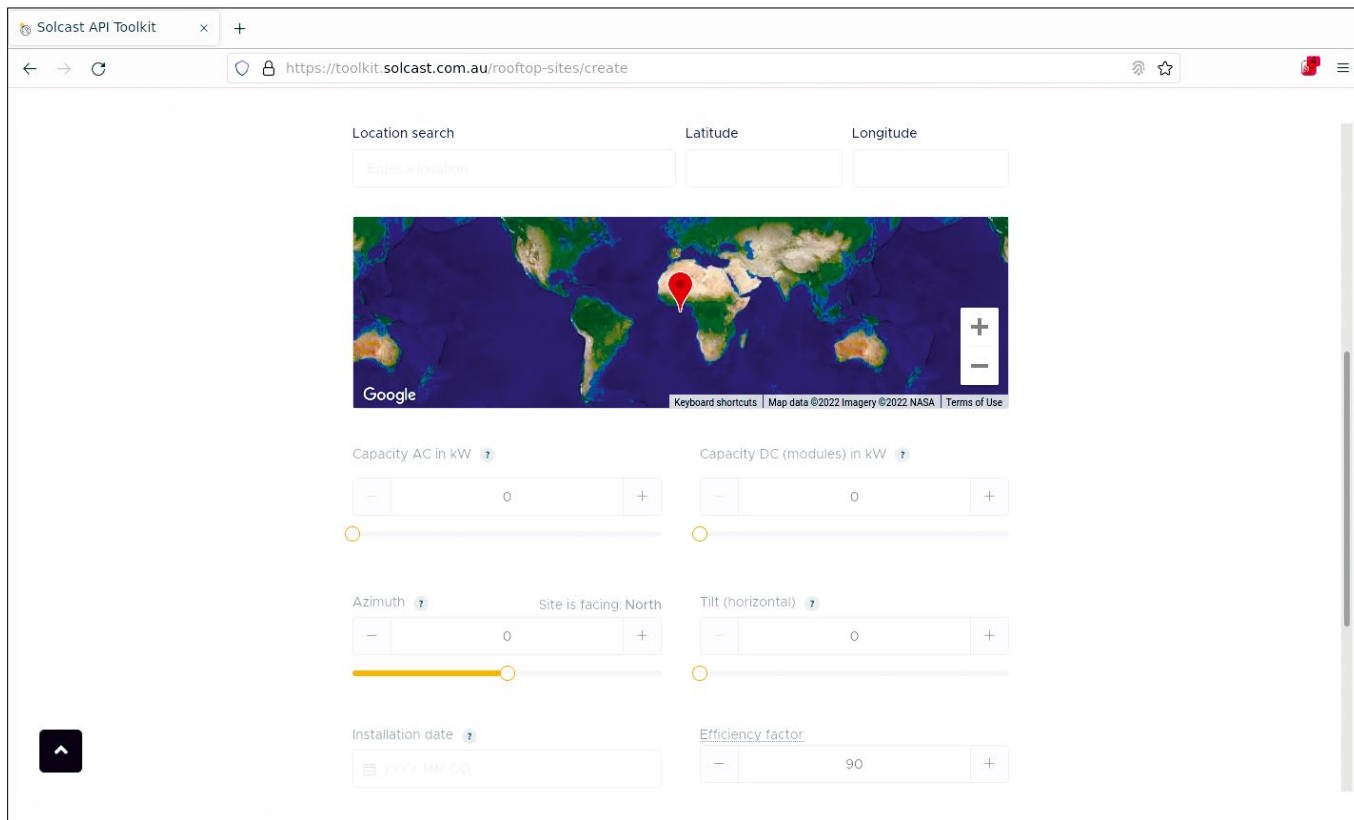


Figure 3: After signing up with Solcast, you must create a site for your solar array before you can download forecast data.

This URL leads to a CSV file (e.g., Figure 5) and can be fetched with `curl`, `wget`, or any other downloader of your liking. Don't forget to replace `$YOUR_SITE_ID` and `$YOUR_KEY` with the ID code for your photovoltaic site and your API key. The meaning

of each API variable is explained in Table 2 and the content of the CSV file in Table 3.

Parsing the Forecast

The Solcast forecast data lets you perform a number of calculations. The

most important are how much power the panels will produce each of the upcoming days and whether you will face a power deficit. Because the forecast is in a machine-parseable format already, you can use some simple Perl scripts to reveal these numbers. Perl is well suited for the task because it is superb for processing text. You can install the necessary environment on a Devuan computer with the commands as root:

```
apt-get update
apt-get -y install perl \
libdatetime-perl
```

Perl has a number of modules for parsing CSV files, such as the one downloaded with the forecast data, but you don't need anything fancy and can make do with some homemade code. Listing 1 shows an example Perl script that reads from standard input and, assuming it is fed a CSV file, saves its contents to an array named `@csv`.

The `while` loop in the script reads each line from the forecast file one by one and stores them in the `$line` variable. Lines 15-17 remove Windows-style newlines, making the input easier to process. Each

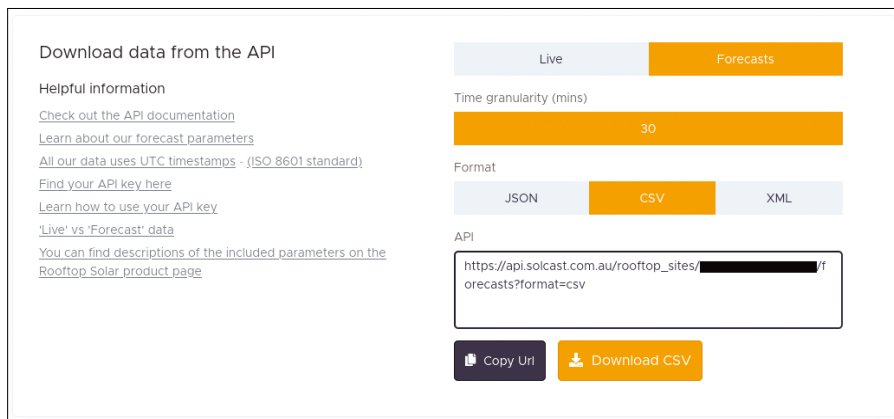


Figure 4: Solcast has a limited wizard that can help you assemble an API call for downloading a solar irradiation forecast in a machine-parseable format.

Table 2: Solcast API Variables

Variable	Description
hours	Number of hours covered by the forecast
format	Format; JSON, CSV, and XML are available
api_key	Your private API key provided by Solcast

	PvEstimate	PvEstimate10	PvEstimate90	PeriodEnd	Period
1	0,4645	0,3248	0,5801	2022-11-03T11:00:00Z	PT30M
2	0,8547	0,4703	1,476	2022-11-03T11:30:00Z	PT30M
3	0,9962	0,3131	2,1605	2022-11-03T12:00:00Z	PT30M
4	0,7139	0,1662	1,3082	2022-11-03T12:30:00Z	PT30M
5	0,986	0,372	1,8889	2022-11-03T13:00:00Z	PT30M
6	0,8524	0,3117	1,5076	2022-11-03T13:30:00Z	PT30M
7	1,2737	0,6664	3,0607	2022-11-03T14:00:00Z	PT30M
8	1,9515	0,9698	3,6341	2022-11-03T14:30:00Z	PT30M
9	2,2346	0,981	3,278	2022-11-03T15:00:00Z	PT30M
10	2,1488	0,8356	2,8221	2022-11-03T15:30:00Z	PT30M
11	1,8	0,6074	2,2012	2022-11-03T16:00:00Z	PT30M
12	1,3373	0,3315	1,5657	2022-11-03T16:30:00Z	PT30M
13	0,6517	0,1015	0,8206	2022-11-03T17:00:00Z	PT30M

Figure 5: Forecast data as downloaded by the Solcast API.

Table 3: Forecast File Elements

Element	Description
PvEstimate	Average power production (kW) for the time period
PvEstimate10	Reasonable minimum power yield for the period (for worst case calculations)
PvEstimate90	Reasonable maximum power yield for the period (for best case calculations)
PeriodEnd	Timestamp marking the end of the period
Period	Length of each period (minutes)

Listing 1: solar_01.pl

```
01 #!/usr/bin/perl
02
03 # Usage: ./solar_01.pl < forecast_file.csv
04 # where each data row from the csv file has the following format:
05 #
06 # PvEstimate,PvEstimate10,PvEstimate90,PeriodEnd,Period
07 # 0.4645,0.3248,0.5801,2022-11-03T11:00:00Z,PT30M
08
09 my $line_number = "0";
10 my @csv;
11
12 # Read from standard input and process each CSV line sequentially.
13 while ( my $line = <> ) {
14
15     # Remove Windows-style EOL
16     $line =~ s/\R/\012/;
17     chomp ($line);
18
19     # Split each row of the CVS file and get the fields we want.
20     my ($power,$timestamp) = (split ",", $line)[0,3];
21
22     ${csv[$line_number]}{"power"} = $power;
23     ${csv[$line_number]}{"timestamp"} = $timestamp;
24
25     ++$line_number;
26
27 }
28
29 exit;
```

line of the CSV file is composed of a group of fields separated by commas, so line 20 captures the first and fourth field and stores them in the `$power` and `$timestamp` variables.

The code that follows is where tricky stuff starts happening. In lines 22 and 23, a data hash is created and stored in the element of the `csv` array corresponding to the current line (i.e., the first line of the file goes into `$csv[0]`, the second line into `$csv[1]`, etc.). In this way, you can access both the power and the timestamp of each row easily: For example, to get the power column of the 10th line, access the `${csv[9]}{"power"}` variable.

In the context of solar power production, one of the most interesting things you can do with this data is compare your solar production with your power consumption.

Consumption Profile

An average Spanish house consumes 9.55kWh daily [2] (US, ~25kWh [3]; UK ~11kWh [4]). You can calculate how much your own house consumes in a number of ways. If you have a solar inverter installed in your house, the inverter itself will tell you what your power consumption profile looks like. If you have not installed a solar system yet, look at one of your electricity bills for the number of kilowatt-hours billed and divide by the number of days of the billing period.

Ideally, you would use an hour-per-hour comparison to try and guess which is the best time of the day for using and which is the best time for saving electrical power at your home. For that to happen, you need to know your consumption profile. Again, if you have a smart inverter, it can give you this information, but if your inverter is dumb or you have no solar system yet, you will need to make a rough estimation.

An acceptable way of making your hourly consumption estimation is to grab the nationwide consumption estimations of your country and assume your house behaves close to the average. For example, I downloaded the generation data from Red Eléctrica de España [5], which provides a CSV file with the power generated in Spain (in megawatts) throughout the day, broken down to periods of five minutes. This data allowed me to calculate which percentage of Spain's power

consumption is generated each hour. I can then safely assume that my own home will behave in a similar way.

In other words, if I guess that 5 percent of Spain’s power generation occurs between 12:00pm and 12:05pm, I can extrapolate that my house will take 5 percent of its daily power during that

time. Listing 2 is a consumption profile from such an extrapolation, and Figure 6 is a graph of that data. The first column is the timestamp belonging to the end of a one-hour metered period, and the second column is the percentage of the total daily power usage that takes place in that period.

Putting Everything Together

What you eventually want is for the computer to download a forecast reaching many days in the future and, taking your power consumption profile into account, give you an idea of how much surplus power you are going to get in future days. A good strategy is to have a cronjob fetch the forecast file from Solcast right after midnight and then parse it with a program capable of making these estimations. The script in Listing 3 is a proof of concept for such a program.

The script assumes solar arrays are working without battery backup and attempts to guess how much power your house will take from the grid and how much power will be produced but not used (Figure 7). Power considered “wasted” is what would be sold to the power supply company if you sign a smart grid contract. This script iterates over a given forecast file and calculates the time period covered by the file.

Lines 10-37 load the forecast and profile files to memory with the same approach as solar_1.pl. The profile file needs no removal of Windows-style newline terminations; therefore, it lacks a replacement regex such as the one featured on line 14. Lines 50 to 96 iterate through all the lines of the CSV forecast file, discarding the first one (which does not store a useful value). Although not really recommended by Perl gurus, I decided to use a C-type loop for readability. Lines 52-54 use regular expressions to extract the year, month, day, hour, minute, and second corresponding to the

Listing 2: Electric Consumption, Spain

```
Date,Percentage of Daily Consumption
2022-11-02 00:30,1.56020056433848
2022-11-02 01:00,1.56020056433848
2022-11-02 01:30,1.63088497829003
2022-11-02 02:00,1.63088497829003
2022-11-02 02:30,1.58141212721891
2022-11-02 03:00,1.58141212721891
2022-11-02 03:30,1.56087988890132
2022-11-02 04:00,1.56087988890132
2022-11-02 04:30,1.57097271097776
2022-11-02 05:00,1.57097271097776
2022-11-02 05:30,1.62759926560856
2022-11-02 06:00,1.62759926560856
2022-11-02 06:30,1.85454912466257
2022-11-02 07:00,1.85454912466257
2022-11-02 07:30,2.11453633949703
2022-11-02 08:00,2.11453633949703
2022-11-02 08:30,2.22951548769889
2022-11-02 09:00,2.22951548769889
2022-11-02 09:30,2.22946696451583
2022-11-02 10:00,2.22946696451583
2022-11-02 10:30,2.21253930551125
2022-11-02 11:00,2.21253930551125
2022-11-02 11:30,2.20946154932860
2022-11-02 12:00,2.20946154932860
2022-11-02 12:30,2.26549889387938
2022-11-02 13:00,2.26549889387938
2022-11-02 13:30,2.28922673039563
2022-11-02 14:00,2.28922673039563
2022-11-02 14:30,2.29525746886163
2022-11-02 15:00,2.29525746886163
2022-11-02 15:30,2.27086417154914
2022-11-02 16:00,2.27086417154914
2022-11-02 16:30,2.27188315839339
2022-11-02 17:00,2.27188315839339
2022-11-02 17:30,2.27048984985125
2022-11-02 18:00,2.27048984985125
2022-11-02 18:30,2.37193796186272
2022-11-02 19:00,2.37193796186272
2022-11-02 19:30,2.45179325741258
2022-11-02 20:00,2.45179325741258
2022-11-02 20:30,2.47148673785156
2022-11-02 21:00,2.47148673785156
2022-11-02 21:30,2.38225260420458
2022-11-02 22:00,2.38225260420458
2022-11-02 22:30,2.15359750186017
2022-11-02 23:00,2.15359750186017
2022-11-02 23:30,1.96871031063571
2022-11-03 00:00,1.96871031063571
```

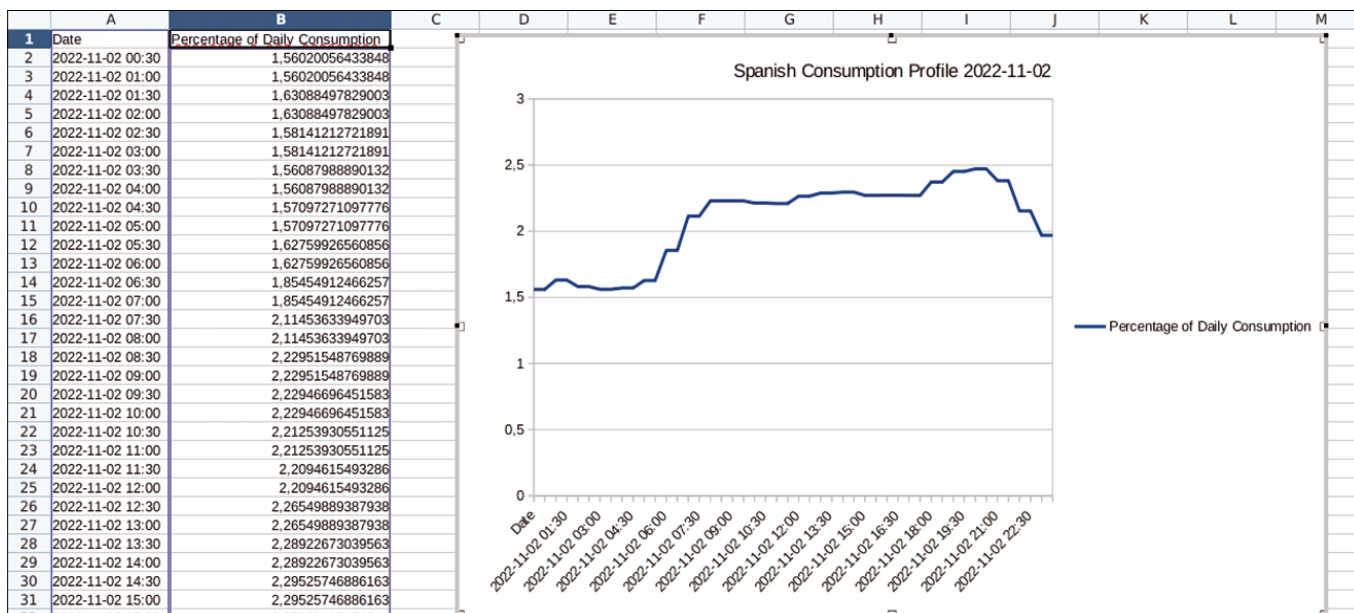


Figure 6: Power consumption profile in Spain on February 11, 2022.



FOSSLIFE

Open for All

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

FOSSlife.org

Listing 3: solar_2.pl

```

01 #!/usr/bin/perl
02
03 # Usage: ./solar_2.pl forecast.csv profile.csv $power
04
05 use DateTime;
06
07 my @csv_forecast;
08 my @csv_profile;
09
10 my $line_number = "0";
11 open (my $fforecast, '<', $ARGV[0]) or die "Failed to
    open $ARGV[0]";
12 while ( my $line = <$fforecast> ) {
13
14     $line =~ s/\R/\012/;
15     chomp ($line);
16
17     my ($power,$timestamp) = (split ",", $line)[0,3];
18     ${csv_forecast[$line_number]}{"power"} = $power;
19     ${csv_forecast[$line_number]}{"timestamp"} =
        $timestamp;
20
21     ++$line_number;
22 }
23 close ($fforecast);
24
25 $line_number = 0;
26 open (my $fprofile, '<', $ARGV[1]) or die "Failed to open
    $ARGV[1]";
27 while ( my $line = <$fprofile> ) {
28
29     chomp ($line);
30
31     my ($timestamp,$percentage) = (split ",", $line);
32     ${csv_profile[$line_number]}{"timestamp"} = $timestamp;
33     ${csv_profile[$line_number]}{"percentage"} =
        $percentage;
34
35     ++$line_number;
36 }
37 close ($fprofile);
38
39 # For each 30 minutes period, calculate whether we are
    going to produce
40 # more power than we are going to consume (and therefore
    waste it),
41 # or whether we are going to consume more power than we
    produce (and
42 # therefore have deficit).
43
44 my $waste = "0";
45 my $deficit = "0";
46 my $produced = "0";
47
48 my ( $y,$m,$d,$h,$mt,$sec );
49 my ( $i,$j );
50 for ( $i = 1 ; $i < @csv_forecast ; ++$i ) {
51
52     ($y,$m,$d) = $csv_forecast[$i>{"timestamp"} =~ /^(
        [\d
53         {4})\-(
54         [\d
55         {2})\-(
56         [\d
57         {2})/;
58     ($h,$mt,$sec) = $csv_forecast[$i>{"timestamp"} =~
        /T(
59         [\d
60         {2})\:(
61         [\d
62         {2})Z$/;
63     unless ( defined ($h) ) { ($h,$mt,$sec) = ( '0', '0',
        '0' ) };
64     my $forecast_time = DateTime->new(
65
66         year => $y,
67         month => $m,
68         day => $d,
69         hour => $h,
70         minute => $mt,
71         second => $sec,
72         time_zone => 'UTC'
73     );
74     $forecast_time->set_time_zone('local');
75
76     my $consumption;
77     my ( $hour,$minute );
78     for ( $j = 1 ; $j < @csv_profile ; ++$j ) {
79         ($hour,$minute) = $csv_profile[$j>{"timestamp"} =~ /
80             (
81                 [\d
82                 {2})\:(
83                 [\d
84                 {2})$/;
85         if ( $hour == $forecast_time->hour() and $minute ==
            $forecast_time->minute() ) {
86             $consumption = $csv_profile[$j>{"percentage"} *
87                 $ARGV[2] * 0.01;
88         }
89     }
90     if ( $consumption > ( $csv_forecast[$i>{"power"} * 0.5
91         )) {
92         $deficit += $consumption - ( $csv_forecast[$i]
93             {"power"} * 0.5 );
94     } else {
95         $waste += ( $csv_forecast[$i>{"power"} * 0.5 ) -
96             $consumption;
97     }
98     $produced += $csv_forecast[$i>{"power"} * 0.5;
99
100    # If the day has changed, print information for the
    day.
101    if ( $forecast_time->hour() == "0" and $forecast_
102        time->minute() == "0" ) {
103        $forecast_time->subtract( days=> "1");
104        print "INFORMATION FOR DATE ". $forecast_
105            time->date() . "\n";
106        print "Total power produced will be $produced kWh\n";
107        print "About $deficit kWh will be taken from the grid
108            (deficit)\n";
109        print "About $waste kWh will be produced but not used
110            (wasted)\n\n";
111
112        $deficit = 0;
113        $waste = 0;
114        $produced = 0;
115    }
116 }
117
118 exit;

```

timestamp of the entry of the forecast file. Lines 55-64 create a `DateTime` object corresponding to the timestamp and adjust it to the local time zone.

Lines 68-75 look in the profile file for the entry corresponding to the timestamp it is currently analyzing and calculate the total power you expect the house to consume in that period. This calculation multiplies the total energy use expected for the day by the percentage of daily power use expected in the period corresponding to the timestamp, divided by 100. The result, in kilowatt-hours, is stored in the `$consumption` variable.

The calculations in lines 76-80 are the energy debt and excess. If the house

consumes more power than is produced, the script notes the energy debt in the `$deficit` variable. Otherwise, excess power is noted in the `$waste` variable. The script also records the power produced so far by the solar array during the day in line 82.

Line 85 guesses whether the end of the day has arrived. If so, the script prints the information corresponding to the day just parsed and sets `$deficit`, `$waste`, and `$produced` to zero to start the next day anew.

Conclusion

The script in Listing 3 is a proof of concept and lacks input validation. It does not check whether the CSV files fed to it

are correct and lacks appropriate exit statuses for input errors. However, it serves as a base on which to build a prediction system. You can easily set up a cronjob to download a forecast file from Solcast each day at midnight, generate a prediction with the script, and send it by email or post it to a website. The possibilities are endless (Figure 8).

A more complex script that supports input validation and tracks the state of the battery banks is available at my gophersite [6]. (You'll need a gopher client.) ■■■

Info

- [1] Solcast irradiance prediction service: <https://solcast.com.au/>
- [2] IDAE residential consumption in Spain: https://www.idae.es/uploads/documentos/documentos_Documentacion_Basica_Residencial_Unido_c93da537.pdf (in Spanish)
- [3] US EIA energy use in homes: <https://www.eia.gov/energyexplained/use-of-energy/electricity-use-in-homes.php>
- [4] UK household energy usage: https://www.ukpower.co.uk/home_energy/average-household-gas-and-electricity-usage
- [5] Power output of the Spanish power grid, February 2022: <https://demanda.ree.es/visiona/peninsula/demandaqh/tablas/2022-11-2/1> (in Spanish)
- [6] Rubén Llorente's gopherhole: [gopher://gopher.operationalsecurity.es/1/Software](https://gopher.operationalsecurity.es/1/Software)

Author

Rubén Llorente is a mechanical engineer who ensures that the IT security measures for a small clinic are both legally compliant and safe. In addition, he is an OpenBSD enthusiast and a weapons collector.



```
devuan@developer:~$ perl solar_2.pl forecast.csv profile2.csv 9.55
INFORMATION FOR DATE 2022-11-06
Total power produced will be 13.22555 kWh
About 5.71808868822704 kWh will be taken from the grid (deficit)
About 9.42324045014541 kWh will be produced but not used (wasted)

INFORMATION FOR DATE 2022-11-07
Total power produced will be 9.47265 kWh
About 5.90387046888784 kWh will be taken from the grid (deficit)
About 5.8561222308062 kWh will be produced but not used (wasted)

INFORMATION FOR DATE 2022-11-08
Total power produced will be 7.0111 kWh
About 6.10894940562567 kWh will be taken from the grid (deficit)
About 3.59965116754404 kWh will be produced but not used (wasted)

devuan@developer:~$
```

Figure 7: Output from `solar_2.pl` gives an approximation of how much power will be used and how much power will be wasted.

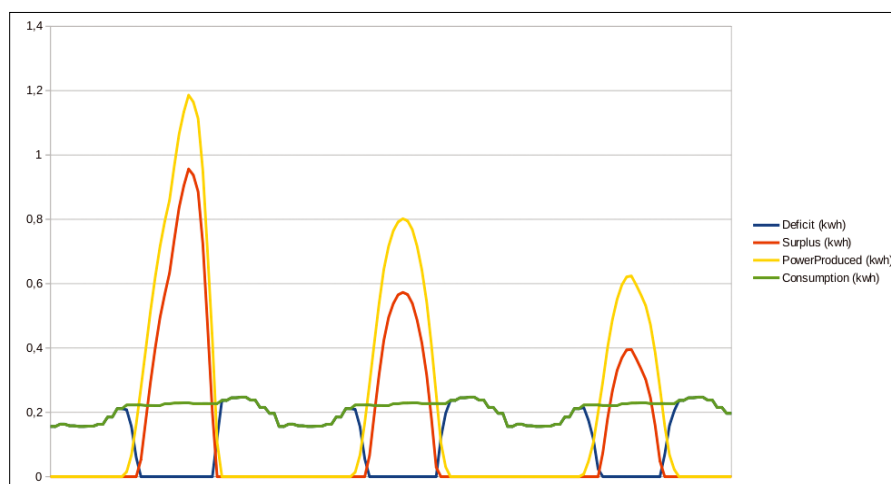
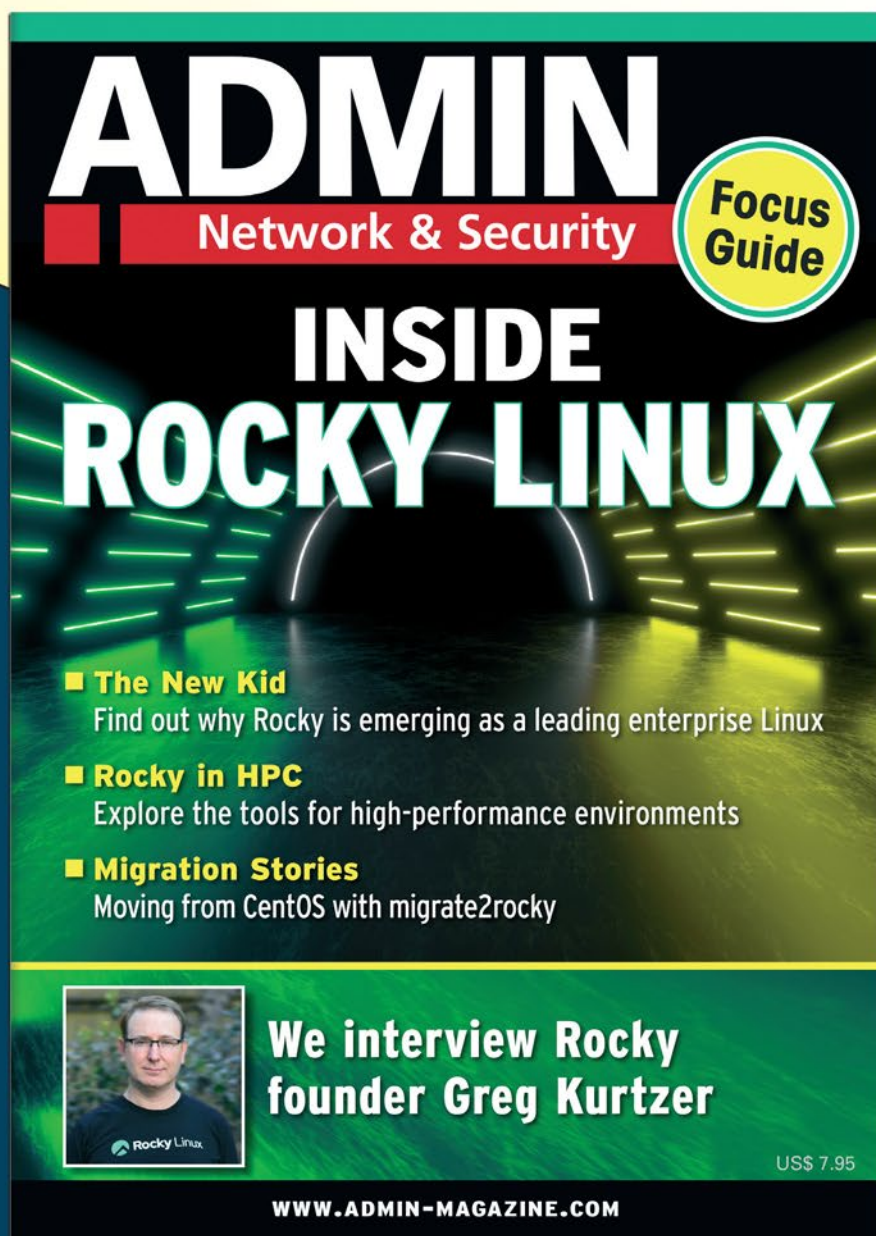


Figure 8: Deficit, wasted power, and consumption curves through three days, generated by a variation of the script from Listing 3.

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

**INSIDE
ROCKY LINUX**

Focus Guide

- **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>

In the old days, if you wanted to switch between Linux systems, you needed to set up your computer to dual boot. Then virtual machines entered the picture, which made it easier to switch temporarily. But virtual machines were still more trouble than many users want to take, especially if you just needed to change temporarily to run a test or check a program. This month we show you an alternative option for switching Linux systems in a Linux session. You'll learn how to set up Ubuntu (or another Linux) to run in a Docker container. Also in this issue of Linux Voice, view television on Linux with IPTVnator and visualize your LEGO projects with LDraw and LeoCAD.



Image © Olexandr Moroz, 123RF.com

LINUXVOICE ▶

Doghouse – Software Freedom	72
<i>Jon "maddog" Hall</i>	
Restricting uses for FOSS may seem appealing, but it also might not be the solution some imagine.	
Docker	73
<i>Hans-Georg Eßer</i>	
Do you work with Ubuntu but need to test something on openSUSE? You don't need a second PC or a virtual machine – a single container is quite enough.	
IPTVnator	80
<i>Erik Bärwaldt</i>	
The IPTV standard lets you view your favorite channels on Linux.	
FOSSPicks	84
<i>Graham Morrison</i>	
This month Graham looks at LosslessCut, Webcamoid, Sniffnet, and more!	
Tutorial – LDraw and LeoCAD	90
<i>Daniel Tibi</i>	
LDraw and LeoCAD help you become a virtual LEGO architect.	



CC-BY-SA WWW.PEPPERTOP.COM



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

MADDOG'S DOGHOUSE

Restricting uses for FOSS may seem appealing, but it also might not be the solution some imagine. BY JON “MADDOG” HALL

The tool isn't the problem

Recently the question of whether free software (or even open source) should have some clause put into the license to keep the software from being used for “bad” purposes came up again. I believe that most of this was inspired by the war between Ukraine and Russia, and people not wanting their software used for purposes of war. Some of this is probably also inspired by various governments using computer vision software to spy on their citizens or open AI software to establish surveillance. One person even brought up the concept of using “his” free software in nuclear weapons.

I am a pacifist. I do not believe in war as a solution. However, this does not mean that I would just throw up my hands if a warmongering power attacked my country and started harming people. Therefore I can understand the feelings of these people.

However, there are many other things that some people do not like, and such people would object to others using “their” software for doing these things.

Some people do not like the government, so they do not want the government to use their software. Some people do not like the military, so they do not want the military to use their software, even if it is not for killing purposes. Some people do not like the police. Some people do not like religious organizations, and some do not like atheists. The list goes on and on, and if you draw a circle around each of these groups, eventually the Venn diagram shows the null set of people able to use the software. And if no one uses the software, what is the reason for writing it?

Another issue is the concept of software ownership. A particular person, perhaps the founder of the project or the team leader, may try to limit the use of the software, but are they the only person who wrote it? Do all of the contributors agree with this limitation? One of the reasons why the Linux kernel project is still back on the 2.x license of the GPL is that there are thousands of contributors and thousands of copyright holders, who would all have to agree to the change in the license. Some of these contributors have left the project, or died, making it really difficult to get their permission to change the license.

People may have heard of open source projects that change the licensing. Normally this is done with a fork of the project where the original license is maintained for the code released before the license change, and the modified license is applied to the forked code after. Another way of changing a license is to have all the contributors assign their copyrights to an entity (either a person or a corporation) so that person or corporation has the ability to change the license. This is why the GNU project could go from the GPL 2.x licenses to the GPL 3.x licenses without having to get permission from thousands of people.

But trying to get thousands of people to agree on changing the license to include sanctions against some group of people or a country would probably be close to impossible, and a very dangerous precedent.

Even the discussion of such a path would create horrible ripples. In the early days of Free and Open Source Software (FOSS), intellectual property lawyers would read over the many licenses of FOSS software just to make sure there was not some “gotcha” that might affect the companies they represented and prevent those companies from using the software. This inspection by these lawyers is one of the reasons given for a limited set of open source licenses to be generated and used. Once a lawyer has inspected a particular type of license for its terms and conditions, they do not have to inspect it again. If now there were new restrictions superimposed on end uses, you would open this inspection again and many companies would give up using and contributing to open source.

Finally, open source software will find its way to people who want to use it for bad purposes no matter what you do. Just as locks only keep honest people honest, licenses will not prevent evil people from using software for evil purposes.

The solution to this is not to blame the tool, but the user. A hammer can be used to build a house or it can be used to kill a person ... used for good or for evil. It is the end developer who is helping to build the weapon or wage the war that we need to stop, not the FOSS in the pipeline. ■■■

Use Ubuntu and other distributions as Docker containers

Cleanly Docked

Do you work with Ubuntu but want to test something quickly on an openSUSE system? You don't need a second PC or a virtual machine to do it – a single container is quite enough. BY HANS-GEORG EBER

A virtual machine (VM) lets you use software intended for a different Linux distribution or even a different operating system. But setting up a VM can be time consuming. If you use VirtualBox, VMware [1], or GNOME Boxes [2], you can configure and start a virtual PC in the program interface before proceeding with the regular operating system install. After completing this procedure, the window will contain a full graphical desktop, such as KDE or GNOME, and you can boot and shut down the VM like a real PC.

In many cases, a full install is exactly the solution you need, but sometimes you may just want to quickly test a program. Then the overhead for a VM install of a complete system is disproportionate to the results. The reasons why you cannot always run programs directly on your Linux system are revealed in the “Library Dependencies” box.

If you develop your own software, you may want to test whether the program files you generate will run on all distribution versions you support. Can the RPM or Debian package you generated be installed? Are all dependencies on libraries met so that the program actually launches? Finally, does it work as expected? Again, using a VM is a little excessive for these software tests.

Containers offer an alternative to VMs or real hardware: They do not virtualize a complete computer and are therefore not suitable for all purposes, but in many cases they are perfect for the use case at hand. The advantage they have is that of resource efficiency (lower RAM and disk space requirements), and they can be set up far faster than a VM. You can deploy a new container in seconds, while a fresh installation in a VM will take around 15 minutes. Even when cloning an existing VM, you can

Library Dependencies

The reason why you cannot run arbitrary Linux applications on just any system is that the program files usually contain only parts of the binary code that the computer executes. Much of the functionality is found in library files that Linux additionally loads at program startup time. For example, if an interactive program in text mode wants to read a line of text, it does not use its own routine for this, but instead resorts to the `readline` function from the library of the same name. Figure 1 shows a small sample program that reads a line of text and outputs it unchanged with a hint. The code for the input and output comes from the `readline` library function for the input and from `printf` for the output. When compiling with GCC, the `-lreadline` option ensures that the compiler notes the dependency on the `readline` library in the generated program file.

The last command in Figure 1 shows all the libraries that the test program needs, including `libreadline.so.8`. The file extension `.so.8` indicates that the program expects a

specific version of the library, version 8. If you try to run the program on a machine with the older readline version 6 or 7, the program loader will fail to find the library and will complain (Listing 1). Missing libraries can be installed later on: Linux is quite flexible and lets you install different versions of the same library at the same time. Whether you will find a missing version for a particular Linux distribution is another question.

There is so much dynamism in Linux and the applications available for it that each distribution update also updates a few library packages. Programs that require libraries that are too new or too old will no longer run. Switching to a different distribution, such as from Ubuntu to openSUSE, becomes even more difficult because distributors often decide for themselves to keep certain software packages at a version below the latest available one. A program compiled for openSUSE might then need libraries from two different releases on Ubuntu.

```

dd@Ubuntu-2204-Desktop:~$ batcat --theme Coldark-Cold readline-test.c
File: readline-test.c
1 #include <stdio.h>
2 #include <readline/readline.h>
3 #include <readline/history.h>
4
5 int main () {
6     char *answer = readline ("Please enter anything: ");
7     printf ("Your input: %s\n", answer);
8 }
9

dd@Ubuntu-2204-Desktop:~$ gcc -o readline-test readline-test.c -lreadline
dd@Ubuntu-2204-Desktop:~$ ./readline-test
Please enter anything: Hi there!
Your input: Hi there!
dd@Ubuntu-2204-Desktop:~$ ldd readline-test
linux-vdso.so.1 (0x00007ffd283f6000)
libreadline.so.8 => /lib/x86_64-linux-gnu/libreadline.so.8 (0x00007f578c08a000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f578be62000)
libtinfo.so.6 => /lib/x86_64-linux-gnu/libtinfo.so.6 (0x00007f578be30000)
/lib64/ld-linux-x86-64.so.2 (0x00007f578c0f4000)
dd@Ubuntu-2204-Desktop:~$

```

Figure 1: Using what others have developed. Thanks to library functions such as `readline`, you don't have to constantly reinvent the wheel.

Listing 1: Version Error

```

./readline-test: error while loading shared libraries: libreadline.so.8: cannot open shared
object file: No such file or directory

```

expect a wait of several minutes because multiple gigabytes of data need to be copied in the process.

This article takes a look at Docker [3]. But there are other tools for providing containers on Linux. Docker competitor Linux Containers (LXC [4]) is another popular choice; it is used, for example, by the Proxmox [5] VM and container management tool.

Installing Docker

Docker is quickly set up. On Ubuntu, install the `docker.io` package (Listing 2, line 2) and sign up as a member of the new `docker` group using `usermod` (line 3). On an openSUSE system, you can install with Zypper (line 5). The package there is simply

Listing 2: Setting Up Docker

```

01 ### On Ubuntu
02 $ sudo apt install docker.io
03 $ sudo usermod -a -G docker esser
04 ### On OpenSuse
05 $ sudo zypper install docker
06 $ sudo systemctl enable docker
07 $ sudo systemctl start docker
08 ### Function test
09 $ docker run hello-world

```

named `docker`, and the Docker service is still disabled after the setup. You can change this with the two `systemctl` calls (lines 6 and 7).

Log out and then log back in for your new group membership to take effect. By the way, in our test on Ubuntu 22.04, a new login was not enough. We had to kill all of our own processes by typing

```
sudo killall -9 -u $USER
```

and then log in again for this to work. Alternatively, you could reboot the computer, but the `killall` approach is faster.

Finally, make sure the setup worked (line 9). The program should output a few lines of text, including *Hello from Docker!*. If the test fails, use `groups` to check if you are a member of the `docker` group. Also use `systemctl status docker` to see if the Docker daemon is running.

Images and Containers

Running the Docker run `hello-world` command from line 9 of Listing 2 downloaded an image from Docker Hub [6], created a new container from the image file, and started the container. The container then ran a Hello World program that printed all the lines in the output starting at *Hello from Docker*.

Listing 3: Starting a Container

```
### Current version
$ docker run -it ubuntu
### Ubuntu 20.04
$ docker run -it ubuntu:20.04
### Ubuntu 16.04
$ docker run -it ubuntu:16.04
```

If you are new to containers, the terms “image” and “container” might be confusing. An image is essentially a collection of files, supplemented by various metadata, such as the computer architecture or specifications for environment variables. It only acts as a template for creating a container. When you run the `docker run <image>` command, Docker creates a container from the specified image file. To do this, the tool unpacks the image. You will find the files in a subfolder of `/var/lib/docker/overlay2/`.

Docker then launches a program from that folder, pretending that the folder is the root filesystem. The program cannot access files located farther up in the filesystem, which protects your normal Linux system against manipulation attempts from inside the container. Besides being locked into a subfolder, far more happens. Among other things, the container has its own network settings, which are independent of the host system.

You can use a single image file to create multiple containers in which separate processes then run. The containers change at runtime because you can delete, recreate, or edit files in them. If you are looking for an analogy for the difference between images and containers: Compared with virtualizers such as VMware and VirtualBox, an image would correspond to a virtual appliance, while the container would correspond to the individual VM.

Disposable Containers

Containers (unlike VMs) are often used just once. In this case a single command runs the container,

Listing 4: In the Container

```
01 esser@percent:~$ docker run -it ubuntu
02 root@4eaac4755505:/# ps
03  PID TTY          TIME CMD
04   1 pts/0    00:00:00 bash
05   9 pts/0    00:00:00 ps
06 root@4eaac4755505:/# grep PRETTY /etc/os-release
07 PRETTY_NAME="Ubuntu 22.04.1 LTS"
```

Listing 5: Command Call

```
$ docker run --rm fedora grep PRETTY /etc/os-release
PRETTY_NAME="Fedora Linux 36 (Container Image)"
```

and Docker immediately deletes it afterwards. Docker comes with a `--rm` option, which automates the delete process.

This wasteful use of containers is possible (and makes sense) because creating and deleting containers is virtually effortless. It is only when you launch a container for the first time that you need to obtain the matching image, typically from Docker Hub [6]. On all subsequent launches, it will already exist in Docker’s local cache directory.

Because containers are often short-lived, it has become common practice to store configuration files and anything else that you want keep outside the container. To do this, you can either make part of the normal filesystem available in the container or create special container volumes.

Ubuntu Versions

You can use the commands in Listing 3 to create containers with different versions of Ubuntu. Each of these commands creates a container from an Ubuntu image and starts a Bash shell in it. The options `-i` (interactive) and `-t` (terminal) tell the container to run in interactive mode with a virtual terminal, so you can control it from the current terminal window. Without these options the container would start in the background.

From the root prompt, you can see that you have administrator privileges in the container (Listing 4, lines 2 to 5). In the shell you can now enter commands in the usual way (lines 2 and 6). If you try to use external programs, you will start the versions from the container. To find out which distribution version is running in the container, you can look for the `/etc/os-release`, `/etc/debian_version`, or `/etc/redhat-release` files (line 6).

If you already know when starting the container that you do not need a shell, but simply want to execute a specific command, you append it after the image name in the call (Listing 5). Simply omit the `-i` and `-t` options if interactive use is not required. Figure 2 shows how to launch containers with different versions of Ubuntu. To do this, simply append a colon and the desired version to the image name (such as `ubuntu:16.04`).

You cannot use the Docker tool to find out which versions are available, but you can get this information from the Docker Hub page [6]. Use the search function there, select the appropriate image from the matches, and then switch to the *Tags* tab. On a longish page, you will then find hints relating to all available tags. These can be version numbers, but also code names or identifiers such as *latest*. If you change

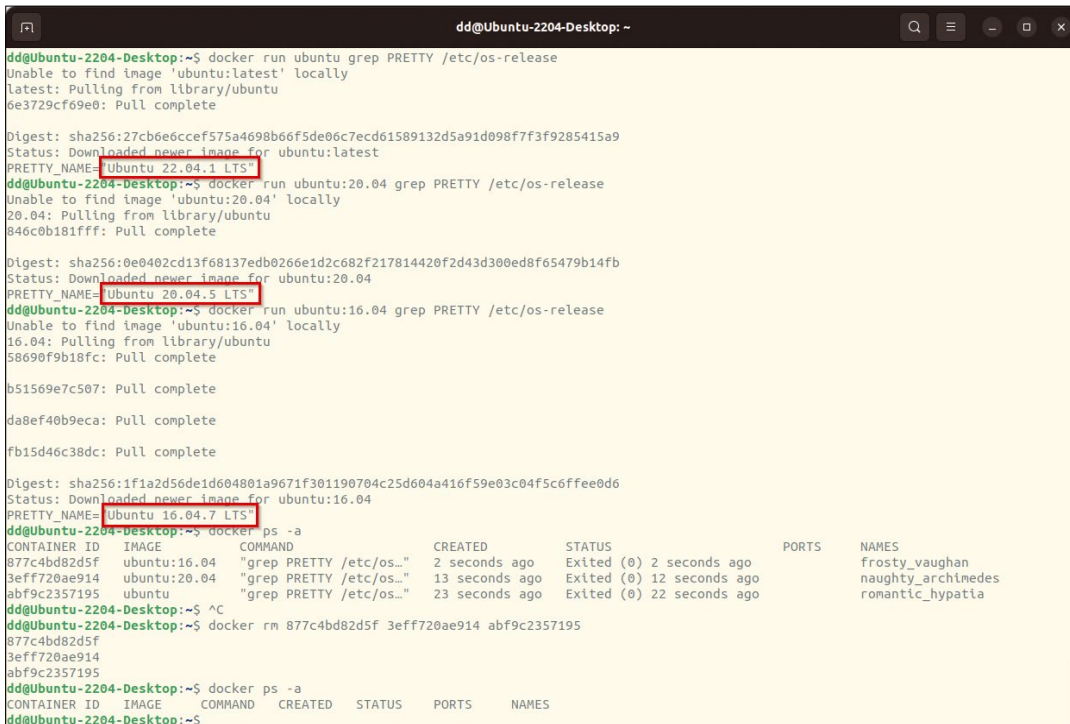


Figure 2: If you are testing different Ubuntu versions, you will find the version information in /etc/os-release.

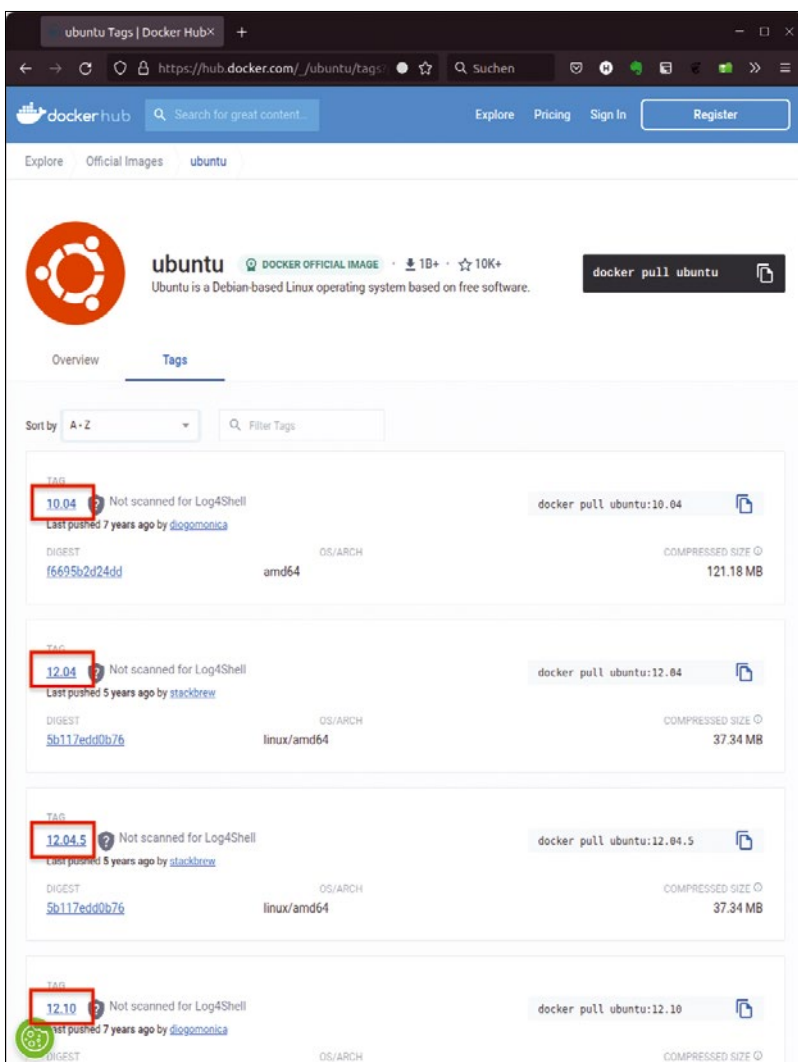


Figure 3: The Tags subpage for an image on Docker Hub reveals which versions you can use.

the sort order to A-Z, you will see the oldest version numbers first (Figure 3). If you know the name of the image, you can also directly construct a URL, such as https://hub.docker.com/_/<image>/Tags.

The Bash function in Listing 6 is an alternative to this. Include this code snippet in the `~/.bashrc` file and post-install the `skopeo` and `jq` tools (in the packages of the same name on Ubuntu). You can use the `dfind` command in all new shells to search for versions on Docker Hub (Listing 7).

File Access

Running programs in

the container usually only makes sense if you can access files on the host computer. You can set up a share far more easily with Docker than with a VM; simply specify an additional option of the form

```
-v <FolderHost>:<FolderContainer>
```

when running Docker. For example, to make your home directory on the host available to an Ubuntu container in `/var/homedir/`, start Docker as shown in the first line of Listing 8. You will then find the whole directory tree including your home directory and all subfolders in `/var/homedir/`.

You can set up several such mappings, and can even include individual files in the container, using `-v`. If you use this method to map the `/etc/passwd`, `/etc/shadow`, and `/etc/group` files,

```

Listing 6: dfind
dfind () {
    skopeo inspect docker://$1 | \
    jq '.RepoTags' | tr -d "\n";
    echo;
}

```

```

Listing 7: Running dfind
$ dfind fedora
["20", "21", "22", "23", "24", "25", "26",
 "26-modular", "27", "28", "29", "30", "31",
 "32", "33", "34", "35", "36", "37",
 "branched", "heisenbug", "latest",
 "modular", "rawhide"]

```

which are important for user management into the container, you can switch to your own account in the container with `su` and work in your home directory in the usual way (Listing 8, starting with line 3).

Target directories in the container do not need to exist; Docker creates them on the fly. Figure 4

Listing 8: File Access

```
01 $ docker run -it --rm \
02   -v $HOME:/var/homedir ubuntu
03 $ docker run -it --rm \
04   -v /home:/home \
05   -v /etc/passwd:/etc/passwd \
06   -v /etc/shadow:/etc/shadow \
07   -v /etc/group:/etc/group ubuntu
08 $ docker cp /etc/passwd \
09   22340547e6a3:/etc/passwd
```

shows the same home directory from the host system and from within the container. Be careful: Changes you make to the files in the container also take effect on the host system. To be safe, it is a good idea to copy the files to the container instead.

File Transfer

Docker provides a separate command for transferring files into and out of the container. `docker cp` lets you copy files in a similar way to the SSH `scp` tool. You cannot use it to copy files to an image; you first need to start a container.

Once the container is running, you will find its computer name at the prompt, which is an ID consisting of hexadecimal numbers. You can also search for the container with `docker ps`. In the overview consisting of very long lines, in addition to the numeric ID, there is also a pretty name that

Building an Image with Dockerfile

In Docker, you can use prebuilt images from Docker Hub, but you can also create customized images. To do this, you need to start a container, make changes to the running system, and create a new image from the changed state with the `docker commit` command.

There is also an alternative approach that works without interaction with a running container. You can write what is known as a Dockerfile containing the instructions for building the image. It starts with a `FROM` line that specifies a (preexisting) base image, which you then modify with the appropriate commands. For example, you could create an image based on Ubuntu 14.04 using the Dockerfile from Listing 9. You would create this in an otherwise empty folder named `Dockerfile`.

Following `FROM`, there is the image name like the one you used with `docker run`. In the example, this is `ubuntu:14.04` for the oldest version of

Ubuntu available on Docker Hub. What then follows are three `RUN` lines with commands that Docker executes in an Ubuntu 14.04 container created temporarily for this purpose. The two calls to `apt` install the `gedit` package; `useradd` creates a non-privileged user named `user` with a user ID of `1000`. The `USER` line specifies that all further steps are run with the user ID `1000`. This is followed by a `CMD` line that specifies the command to run later on at container startup time.

Now, to create an image named `o1dgedit` using this image recipe, just run the `docker build -t o1dgedit .` command in the folder containing the Dockerfile. When typing, make sure that the last parameter of the command is a single dot – it stands for the current directory. Docker then executes the instructions from the Dockerfile in a fully automated process. If everything works, the new `o1dgedit` image will be available afterwards.

```
dd@399773371d11: ~
dd@Ubuntu-2204-Desktop:~$ ls
Desktop  Downloads  hello.c  Pictures  readline-test  snap  'Untitled Document 1'
Documents  hello      Music    Public    readline-test.c  Templates  Videos
dd@Ubuntu-2204-Desktop:~$ ls -l Documents/
total 4
-rw-rw-r-- 1 dd dd 121 Dez 13 17:01 docker-example.sh
dd@Ubuntu-2204-Desktop:~$ docker run -it --rm -v /home:/home -v /etc/passwd:/etc/passwd -v /etc/shadow:/etc/shadow -v /etc/group:/etc/group ubuntu
root@399773371d11:/# su - dd
dd@399773371d11:~$ ls
Desktop  Downloads  Pictures  Templates  Videos  hello.c  readline-test.c
Documents  Music      Public    'Untitled Document 1'  hello  readline-test  snap
dd@399773371d11:~$ ls -l Documents/
total 4
-rw-rw-r-- 1 dd dd 121 Dec 13 16:01 docker-example.sh
dd@399773371d11:~$
```

Figure 4: You can map individual files or entire directories from the host to the container.

Docker randomly assigns unless you specify one for the new container at startup time.

Both the numeric ID and the name can be used for the copy command (e.g., to copy the file `/etc/passwd` to the container). Line 9 of Listing 8 copies the `/etc/passwd` file from the host system to a container with an ID of `22340547e6a3` and to the same folder there, leaving the names unchanged.

X Window

Graphical applications can initially cause problems for Docker, but this is easily solved. The trick for classic X applications (not Wayland) is to pass the `/tmp/.X11-unix/` directory and the `$DISPLAY` environment variable into the container. You can talk an older version of the gedit Gnome editor into working by creating a custom image named `oldgedit` (see the “Building an Image with Dockerfile” box).

The `oldgedit` image is not configured to start the `/bin/bash` shell, but to directly launch the gedit editor. It does this with normal

user rights, that is, not as `root`, but as a user with a user ID of `1000`. When you try to start a container now by typing

```
docker run -it --rm oldgedit
```

you will see some error messages ending with `cannot open display`. The program cannot access an X server and therefore cannot open a window.

The X server running on the host creates a special socket type file named `X0` below `/tmp/.X11-unix/` where applications can contact it. In addition to `X0`, there may be other entries here (`X1`, `X2`, and so on) if multiple X servers are active. The X server to display program windows is determined by the `$DISPLAY` environment variable. It typically contains a value of `:0`.

Then pass the contents of the variable and access to the sockets in `/tmp/.X11-unix/` to the container using the `-e DISPLAY` and `-v /tmp/.X11-unix:/tmp/.X11-unix` options (Listing 10), allowing gedit to run. Because the home directory is additionally mapped to `/home/user/` in the container, gedit should also be able to open and edit files (Figure 5).

Wayland

Current versions of Ubuntu use Wayland as the successor to the X Window System with Wayland

Listing 9: Dockerfile

```
FROM ubuntu:14.04
RUN apt update
RUN apt install -y gedit
RUN useradd -m user
USER 1000
CMD gedit
```

Listing 10: X11

```
$ docker run -it --rm -e DISPLAY \
-v /tmp/.X11-unix:/tmp/.X11-unix \
-v $HOME:/home/user oldgedit
```

Listing 11: Wayland

```
$ docker run -it --rm -e XDG_RUNTIME_DIR -v $HOME:$HOME -v /run/user/1000/wayland-0:/run/user/1000/wayland-0 waylandtest
```

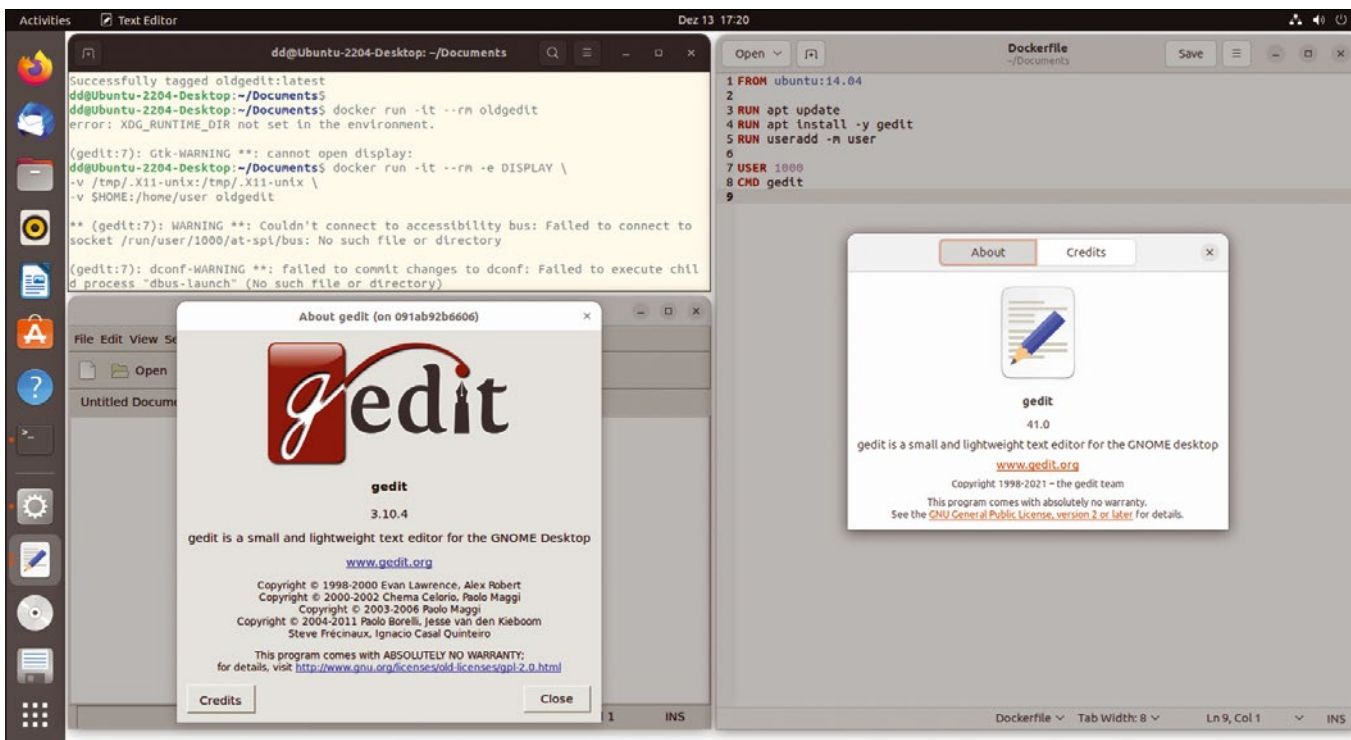


Figure 5: Thanks to Docker, even older graphical applications can run on a modern system. The Gnome editor from Ubuntu 14.04 (left) next to its modern successor.

providing a compatibility layer that allows X applications to run on it. Genuine Wayland programs do not run on X. One example is the `foot` terminal program, which you can install on Ubuntu by typing `sudo apt install foot`. If you launch `foot` in a Wayland session, a terminal window appears. If you try the same thing in an X session, a lengthy error message appears, ending with *failed to connect to wayland; no compositor running? goodbye*.

Wayland uses different sockets and files to provide access to the Wayland server. The socket file is `/run/user/1000/wayland-0` instead of `/tmp/.X11-unix/X0`, and instead of `$DISPLAY`, the variable `$XDG_RUNTIME_DIR` contains relevant information such as the folder name `/run/user/1000/`. You need to pass both into the container with options similar to those for X, as Listing 11 shows. `waylandtest` is just the name of another test image. Support for X and Wayland can also be combined if a Wayland server is running. In this case, you need to pass both sockets and both environment variables into the container.

Conclusions

Docker containers offer a practical alternative to VMs, saving both time and computing resources. With many applications, it's worth learning how to use Dockerfiles to build your own images. You

don't always need to start from scratch and can use an image from Docker Hub as a basis, which you configure to meet your needs. Things start to get more exciting when you use a whole series of containers at the same time. Then advanced topics such as container network configuration and orchestration with tools such as Kubernetes [7] start to emerge. ■■■

Info

- [1] "Comparing VirtualBox and VMware Workstation Player" by Thomas Leichtenstern, *Linux Magazine*, issue 204, November 2017, <https://www.linux-magazine.com/Issues/2017/204/VirtualBox-vs.-VMware>
- [2] "Run Virtual Machines in Gnome Boxes" by Christoph Langner, *Linux Magazine*, issue 221, April 2019, <https://www.linux-magazine.com/Issues/2019/221/Gnome-Boxes>
- [3] Docker: <https://www.docker.com>
- [4] LXC: <https://linuxcontainers.org>
- [5] Proxmox: https://pve.proxmox.com/wiki/Main_Page
- [6] Docker Hub: <https://hub.docker.com>
- [7] Kubernetes: <https://kubernetes.io/docs/concepts/overview/>

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

TV over the Internet with IPTVnator

Couch Surfing

Thanks to the IPTV standard and free software, you can view your favorite channels on Linux without any problems.

BY ERIK BÄRWALDT

Traditional audiovisual technology and modern multimedia computer applications continue to converge. In the past, Linux has been behind the other platforms when it comes to streaming free-to-air TV programs, because of a lack of software. IPTVnator is a young project that is trying to close that gap [1].

Technologies

Conventional broadband reception for television requires additional hardware, such as a receiver for DVB-S, DVB-T, or DVB-C. However, Internet Protocol Television (IPTV) [2] lets you view the daily TV program on the Internet without additional components. IPTV programming has a number of advantages over broadband offerings. Viewers can access the entire range of content offered without a fixed connection and without special receivers.

In addition to public broadcasters, users have the choice of numerous small regional stations that you can receive over the Internet from anywhere in the world. Modular channel lists let you view channels from other countries and

continents. Bandwidths of around 4Mbps are already sufficient for standard-definition (SDTV) reception of TV broadcasts via IPTV. But to view programs in high-definition (HDTV) quality, you need a bandwidth of at least 8Mbps, which is within the range of many modern DSL lines. The IPTV software used on the PC relies on buffering to ensure continuous reception even in the event of bandwidth fluctuations.

IPTVnator

The still quite young program IPTVnator is a great choice for TV viewing via IPTV. Older software packages such as FreetuxTV [3] and Zapping [4] have not seen any development for years, and Hypnotix [5], which is part of the Linux Mint software pool, only runs on a few Linux distributions due to some specific dependencies. IPTVnator is currently one of the only graphical and universally usable applications for IPTV on Linux.

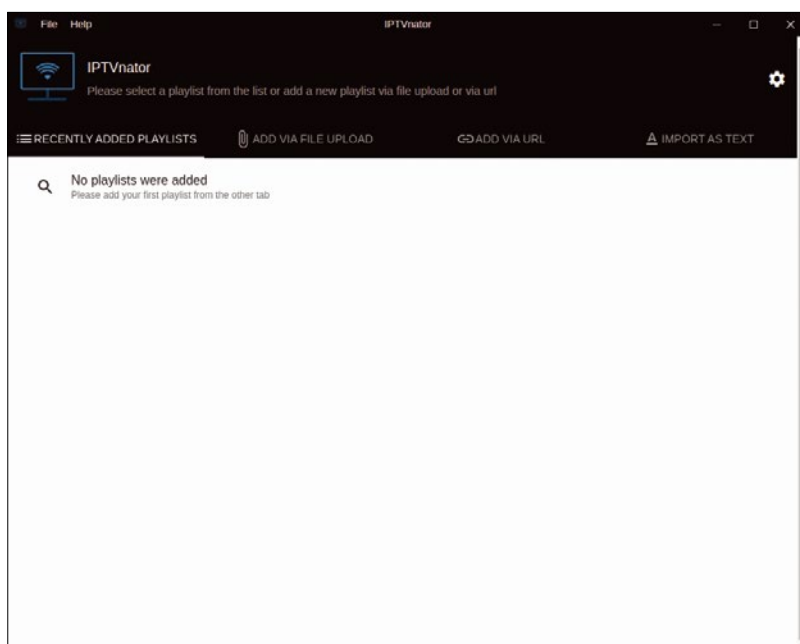
IPTVnator offers the full range of functions for convenient reception. You can integrate M3U or M3U8 playlists, and an integrated program guide lets you define favorites. In addition, the application, which is based on the Electron and Angular frameworks, automatically maintains channel list updates and has built-in HTML5 playback software.

Installation

The software has not made its way into the repositories of the common distributions so far, but you can pick it up as a snap package if the distribution you are using supports snaps. You can also find RPM and DEB packages at the project website – or download IPTVnator as a cross-distribution AppImage for 64-bit systems.

When installing the snap package from a distribution-specific app store, the routine will typically automatically create a launcher in the desktop environment's menu. You can either launch the AppImage package manually or integrate it into your desktop environment using AppImageLauncher. After the first start, the application shows a window with just a few controls (Figure 1).

Figure 1: The program window remains empty on first launch. To breathe life into the software, first import some channel lists.



To view your channels, the next step is to add playlists. The playlists are available in M3U format and can be downloaded from the Internet. A word of caution: Many of the offers accessible via search engines are commercial, and some even spread malware. The recommendation is to use the selection on GitHub [6].

Download the lists to your system in the *Grouped by Country* section by expanding the list. After downloading, open IPTVnator and click the *Upload from file system* button in the program window. Now drag and drop the M3U files with the country-specific channel lists from your file manager into the program window. The M3U files can only be integrated into the application one after the other.

Whenever you paste an M3U file, the playback window opens with a station list arranged on the left side while you see the stream of the current channel in the main part of the window on the right. IPTVnator comes with its own playback software for this purpose.

In the lists obtained from GitHub, you can see the channel's resolution, the channel name, and its logo. You can also discover whether the station broadcasts content 24/7, and whether the program is only available regionally. If this is the case, the channel can only be viewed by using a VPN to access the Internet, and the server must reside in the same country as the channel. If you try to view blocked content directly, you will either see a corresponding message in IPTVnator, or the broadcaster itself will display a message in the playback software (Figure 2).

When viewing a program, you can switch the player to full screen mode by clicking on the open square in the bottom right of the screen. The player then shrinks to just the bottom control line, which you can use to pause the stream, adjust the volume, or set the resolution. To switch back to window mode, click the open cross in the control bar. The application then jumps back to the original display (Figure 3).

Groups and Favorites

IPTVnator lets you select programs by genre. This option is especially useful for channel lists with a large number of channels. To select a program by genre, open the group

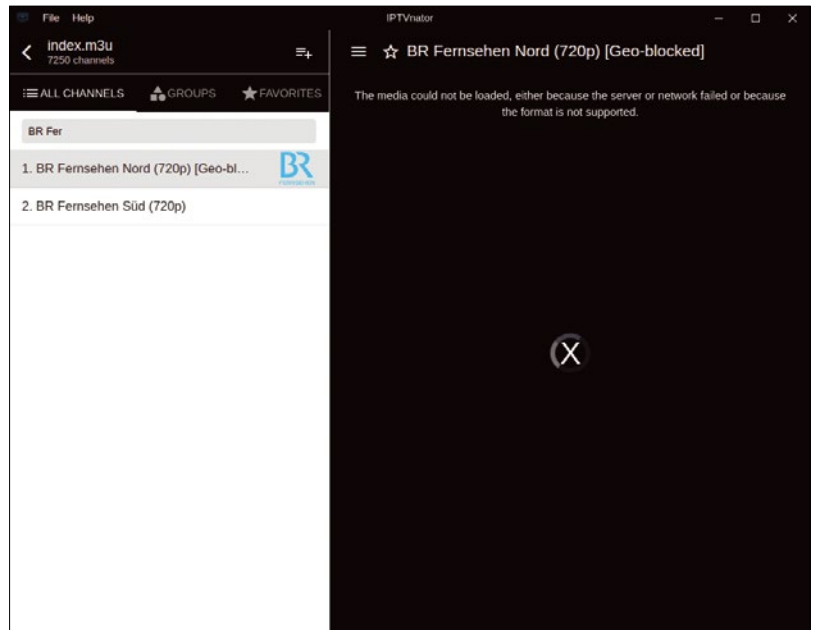


Figure 2: It's not easy to view programs by Germany's Bayerischer Rundfunk outside of Germany.

selection by clicking the *Groups* button in the program window. The software now shows you a table with the individual genres and the number of channels available in each of them. Clicking on the desired genre opens the list of stations below it (Figure 4). Clicking on the group name again closes the selection.

For quick access to your favorite channels, IPTVnator offers a favorites function. To mark a channel as a favorite, just click on the star next to the name of the channel currently streaming above the playback window. To recall a saved channel, click *Favorites*. The selected channel then appears on the left. To remove a channel from the favorites list, click on the star next to the channel name again.



Figure 3: In full-screen mode, only the bottom control bar is left, and it too is automatically hidden after a short period of inactivity.

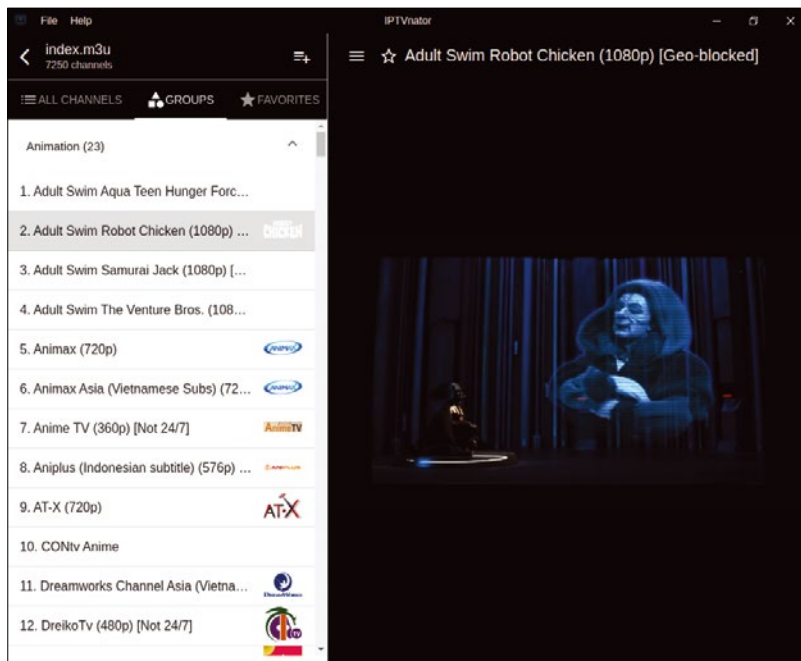


Figure 4: The group function sorts channels into genres, for example, News.

IPTV and VPN

Some streaming providers restrict their offerings to the country of origin and determine the approximate location of the receiving device based on the IP address. This is known as geoblocking. If the device is outside the intended distribution area, an error message appears in the playback window.

You can use a VPN to get around geoblocking. You first need to register with a VPN provider and download and install their software. Unlimited data volumes and access to all available servers of the provider are usually only available with a paid subscription. However, some providers also

offer their service as a free subscription with limited data volume and restricted access to the servers.

After installing the VPN provider's client application, open it and connect to a server in the target country, (i.e., the country in which the streaming service is located). As soon as the tunneled connection is established via the VPN, you can view what are normally blocked broadcasts (Figure 5).

Conclusions

Thanks to IPTVnator, Linux now has an easy-to-use application for streaming IPTV broadcasts. Despite the low version number, the software is already stable on several platforms, and it is suitable for users without in-depth knowledge of the underlying technology, thanks to the built-in playback software. Thanks to IPTVnator, Linux users now have an option for picking up their favorite broadcasts from the road. ■■■

Info

- [1] IPTVnator: <https://github.com/4gray/iptvnator>
- [2] IPTV: https://en.wikipedia.org/wiki/Internet_Protocol_television
- [3] FreetuxTV: <https://github.com/freetuxtv/freetuxtv>
- [4] Zapping: <https://zapping.sourceforge.net/Zapping/index.html>
- [5] Hypnotix: <https://github.com/linuxmint/hypnotix>
- [6] IPTV playlists: <https://github.com/iptv-org/iptv>

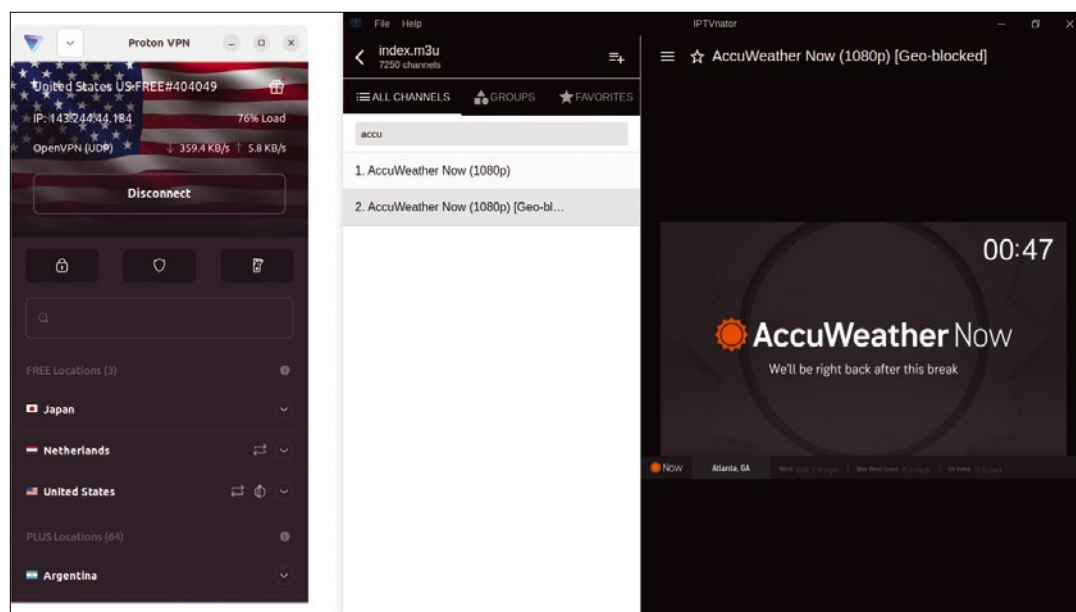
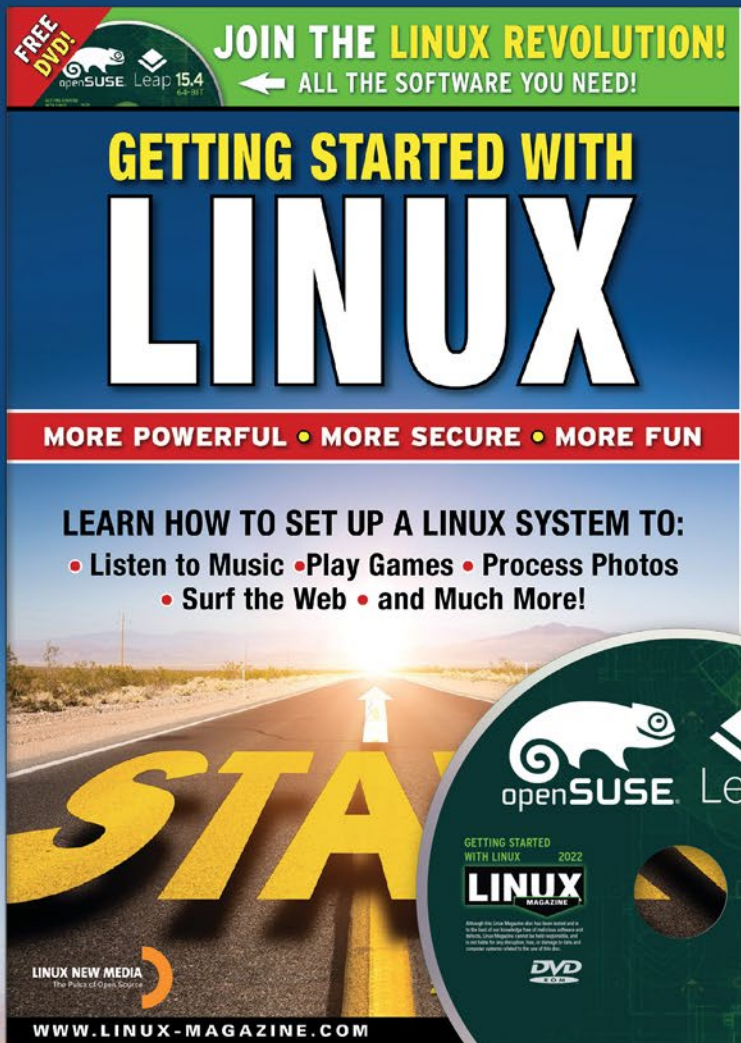


Figure 5: With the help of VPN access to the Internet, you can view channels that you normally can't access because of geoblocking.

Hit the ground running with Linux



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

This single issue shows beginners how to:

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

ORDER ONLINE: shop.linuxnewmedia.com/specials

FOSSPicks

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



Over the past couple of months, Graham's ever-versatile Steam Deck has synced books to an e-reader, played movies on a television, joined Mumble, recorded two podcast episodes, and even played a few games. **BY GRAHAM MORRISON**

Video editor

LosslessCut

Regardless of the incredible performance advances we've made over the past two decades, video editing can feel almost as slow as editing with Media 100 on PowerPC Macintosh in 1995 and writing to a tape drive. This is for a few reasons, but it's mostly because our demands scale to fit our capabilities. Video resolution, color depth, and frame rates have scaled to accommodate all that spare processing, and files nearly always need to be

transcoded from one format to another before they can be used. This latter roadblock is something that LosslessCut attempts to mitigate. Powered by the incredibly versatile FFmpeg, LosslessCut can edit huge videos and, vitally, can export them quickly with no loss of quality. It accomplishes this by only letting you edit at specific keyframes in a video stream, avoiding the processor-intensive need to recalculate frames that may fall between keyframes. By sticking to

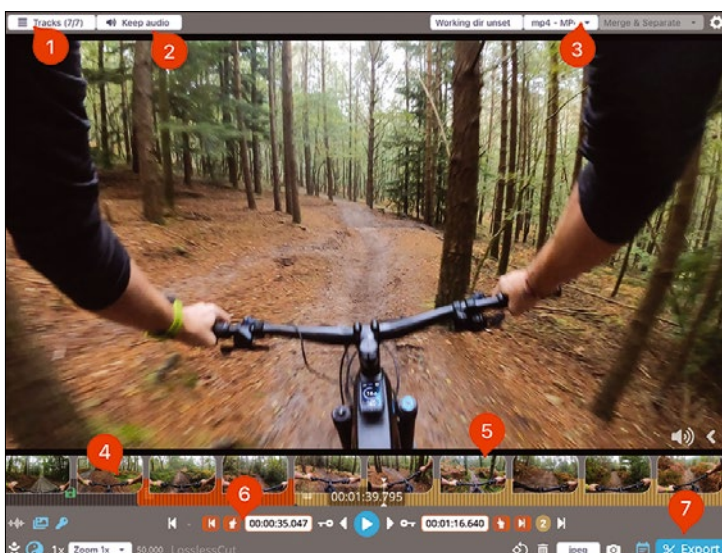
keyframe edits, the video data can simply be rewritten without the edited out parts.

With most of us now recording 4K video on our phones or GoPro equivalents, quick editing like this is the perfect solution. We're not looking for the perfect edit, but instead a way of cutting quickly to the action, chopping the preamble or appendices from the beginning and ends of our videos. With LosslessCut, you can simply drag-and-drop a video into the main application window, locate the "edit-in" point, locate the "edit-out" point, and save the middle segment as a new, shorter video. This does mean you can't perform any other kind of editing task, such as crop, rotate, reflect, or filter processing, because these tasks would require frame and keyframe recalculation, and therefore, a lossy-intensive transformation. The only other disadvantage is that you can't always edit at the exact point you want to since the edit or split point will move to the closest keyframe. Fortunately, there are key icons in the UI that will automatically move the cursor to keyframes and perfect edit points. With those limitations accounted for, the performance bottleneck becomes storage access rather than the transcoding that other applications need to perform.

Editing between keyframes is being worked on, with a recent development release capable of reconstructing the frames after or before a keyframe enabling editing anywhere with only some processing and lossy overhead for the reconstructed part. The remainder of the video will be unaffected, which is still a big improvement on how most video editors work. The rest of the UI is simple but effective. An advanced view enables you to turn on waveform and thumbnail timelines, set the zoom level, take a single image, and see the resolution and frame rate details for your clip. You can also import multiple files, although they currently have to be edited separately rather than on different tracks as you might in Kdenlive – although that feature too is coming. When you've finished, you can export your edits as either a merged single segment or as multiple files, and while the emphasis is obviously to keep the original encoding, you can also choose to convert the output segment to any format supported by FFmpeg. This makes LosslessCut a great front end to file conversion, in addition to meeting all your instant editing needs.

Project Website

<https://github.com/mifi/lossless-cut>



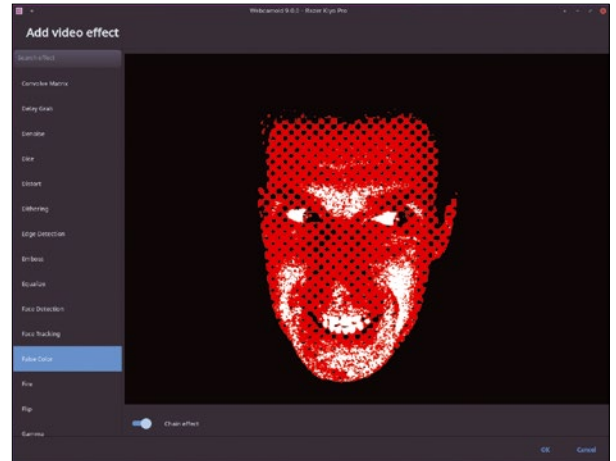
1. Tracks: Multiple video files can be loaded, split, and concatenated. **2. Audio:** Audio tracks can be separated, deleted, and replaced. **3. Output formats:** LosslessCut uses either its own FFmpeg or your system's FFmpeg to support as many file formats as possible. **4. Thumbnails:** Thumbnail and audio waveforms can be optionally enabled for clips. **5. Segments:** Video is cut into segments that can be exported as a batch or as a single file. **6. Keyframe edits:** Skip between edit points with these buttons and navigation controls. **7. Export:** Segments can be labeled, re-ordered, and listed in a separate panel on the right.

Webcam effects

Webcamoid

We recently looked at the brilliant cameractrls, a tool for editing your webcam settings while it is being used. Cameractrls was particularly useful for video conferencing because it enabled you to adjust the brightness, exposure, and color balance while you were in a meeting. But cameractrls could only adjust parameters supported by your hardware and couldn't process the images itself. This is trickier to achieve because it requires more than simply changing device settings and instead requires the video stream to be intercepted, processed, and passed on to a destination as quickly and efficiently as possible. The amazing OBS Studio can do this, and so too can Webcamoid.

Webcamoid is a far more modest desktop application than OBS Studio. Rather than being an entire suite of tools for live-streaming, it's a far more focused photo and video effects processing tool. This makes it less resource hungry and easier to use. It can record videos and take photos from your webcam, capture your desktop, and run captures through one or more of over 60 effects. Alongside the same inbuilt settings you can control with cameractrls, you can turn your video into ASCII, a cartoon, or a black and white movie, with or without distortion, blurring, and aging. These and many other effects can be stacked and exported to any format supported by FFmpeg, which is driving the processing. Like OBS, Webcamoid can install its



If you're unhappy with the built-in effects you find in Zoom or Google Meet, run your streaming video through Webcamoid first.

own virtual webcam driver so that processed video with its audio can be passed through to another application. After setting this virtual device as the source, any application accessing this virtual device will receive the processed video as if it were a live feed, which is a great way to improve the quality of your own video meetings, or at the very least, make them more interesting.

Project Website

<https://webcamoid.github.io>

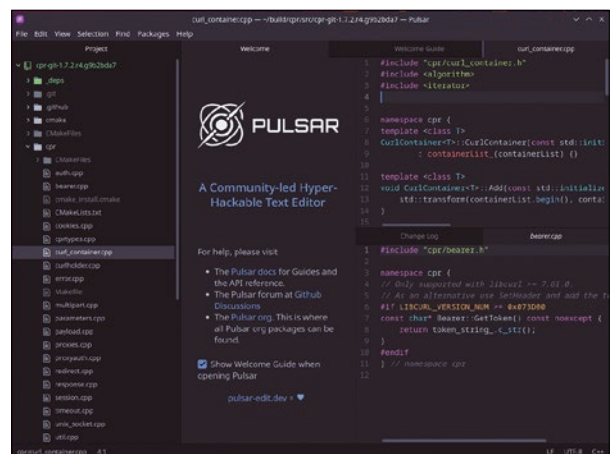
Text editor

Pulsar

There are many projects whose popularity only becomes clear when those projects came to an end. Canonical's Unity desktop is a good example of one that also continues to thrive as a community project, and so too is GitHub's now defunct text editor, Atom. Atom was a code and text editor that became the torchbearer for a new generation of graphical editors whose functionality could be profoundly changed with plugins and a configuration file. More importantly, it was built using Electron – the web-based framework that enabled developers to build desktop applications with web technologies – and Electron was built for Atom. This synergy between the two lasted until the end of

2022 when GitHub ceased its active development of Atom, at which point its immense popularity became apparent. For any such popular open source applications, that really just signals the end of the beginning.

Pulsar is a community and open source continuation of the original Atom code editor and IDE. Like the original, and as you might expect from a modern web framework, it's fantastically well designed, with beautifully rendered fonts and a UI that supports multiple panes from the very first launch. The new branding looks excellent, and while there are still a few rough edges where the old themes haven't quite been replaced, Pulsar already feels like its own project. There's context-sensitive help, a filesystem browser, a built-in package manager for plugins, and smart auto-completion. The old packaging system was closed source, and while Pulsar can still access and use the old Atom.io package



If you loved the Atom text editor, you'll equally love the open source and community-supported Pulsar fork.

repository, the intention is to replace it with a new one containing open source package equivalents. Everything else is as it was in the original editor, and Pulsar means you can continue using your old configuration and project files with your own projects. Better yet, it's now in the hands of a community who are free to take the editor into completely new and untested directions, which is where Pulsar could really start to shine.

Project Website

<https://pulsar-edit.dev>

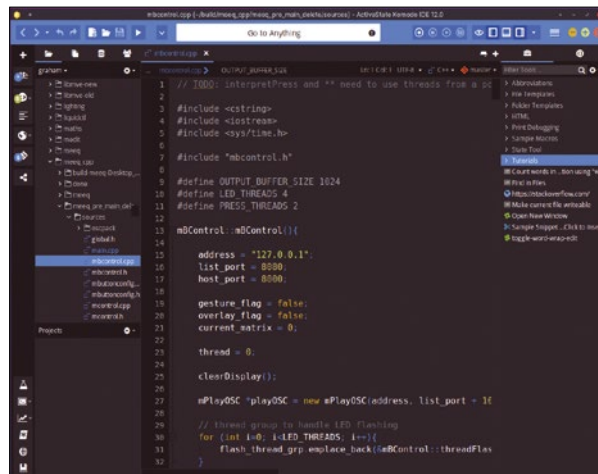
Code editor

Open Komodo IDE

For over 20 years, the Komodo IDE was a proprietary development environment that worked best with web-related programming languages, including Python, PHP, Perl, Go, Ruby, Node.js, and JavaScript. It was published by ActiveState and went through three different teams, migrated through three different version control systems, and contributed to Mozilla's XPCOM framework. It even managed to survive for years after the frameworks it was itself built on, XUL and XULRunner, were dropped by Mozilla back in 2016. But in late 2022, ActiveState finally decided to retire Komodo IDE and, fortunately, make it open source at the same time, creating the Open Komodo IDE project from its 3.2 million lines of code. This was a

brilliant thing to do, and it's something that ActiveState has some history with after previously releasing its editing component, Komodo Edit, some time ago.

None of this would be important if the Komodo IDE weren't worth your time, and Open Komodo is going to be a brilliant option if you've not yet found your perfect IDE – especially if you use Python. It's obviously a mature and stable application with the open source version continuing from the release of Komodo IDE 12. It will feel familiar to anyone who has used Visual Studio Code, because the main window is split into various panes for listing files, methods, and symbols, with a central tabbed editor in the middle. All of this is configurable, and the editor is absolutely fantastic. There's



Komodo IDE is now open source and entirely dependent on its brilliant community for updates and builds.

syntax highlighting, code completion and built-in refactoring, visual debugging, and integrated version control. Maybe it's the K in its name, but it integrates very well with KDE, and the editor is very reminiscent of Plasma's Kate editor. If you're working with HTML or Markdown, there's a live preview of the renderer output of your work, which makes it particularly useful for working with projects with large documentation sets – something often neglected by most IDEs.

Project Website
<https://github.com/ActiveState/OpenKomodoIDE>

Networking monitoring

Sniffnet

Thanks to the inbuilt capabilities of the Linux kernel, we're fortunate enough to have several high-profile and highly competent network-monitoring tools. Wireshark has become an industry standard for low-level analysis, and higher level applications such as Portmaster give us complete control over what goes into and out of our networks. But for non-experts, there isn't so much choice. And for those people, Sniffnet could be enough. Sniffnet is a graphical network monitor that shows a plotted chart of outgoing and incoming data alongside a breakdown of which networking protocols are being used. To start with, you need to select a network device detected on your system, and you need to

run the application as root for adequate access permissions. You can also choose whether to monitor IPv4, IPv6, or UDP protocols, or all three, and there's an additional filter for specific network application protocols. Click on *Run* and the monitoring starts. It can help if the machine you're monitoring acts as a router, such as a server running Pi-hole, but it's also very interesting on a single machine because you can quickly see just how many protocols you use with normal network access. HTTPS will normally top the list, followed closely by the Simple Service Discovery Protocol (SSDP). Type *ssh* to connect to a remote machine, and you'll see SSH appear in the list of used protocols. Individual



Sniffnet is easily installed via Cargo or DEB packages and, while the UI is simple, it does offer light and dark modes of rendering.

connections appear below, sorted optionally by most recent, most bytes, and most packets. The UI is simple and well suited to terminal-only rendering, a feature that's hopefully being worked on. Clicking on the *Open full report* button will output a full network report to a text file. Even without augmentation, Sniffnet already offers a powerful and intuitive set of network monitoring features that will help the majority of us understand our own network traffic and see what's being used.

Project Website
<https://github.com/GyulyVGC/sniffnet>

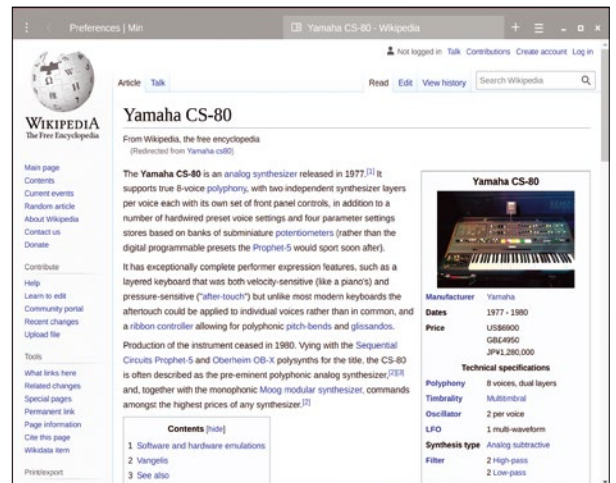
Web browser

Min

Considering how central the humble web browser has become to our lives, all the most popular browsers don't feel so humble, and typically share the same functionality and feel. Chrome, Safari, Edge, Brave, and Firefox are all large and complex applications with hidden depths that can make them difficult to use, and simpler alternatives are few and far between. Qutebrowser is minimal, but thanks to its Vim key bindings, is not easy to use. Which is why this browser, Min, is so refreshing. It may be built on Electron, so it doesn't challenge the Blink monoculture, but its design and aesthetics definitely challenge the ethos that browsers need to have an option for everything. The most obvious

difference is that the URL bar is now part of the window titlebar, like many modern Gnome applications. The URL bar can also be used for search, which defaults to using DuckDuckGo, and works brilliantly with DuckDuckGo bangs such as `!w` to open the top result on Wikipedia. These bangs are augmented with a few of Min's own, such as `!bookmarks` to open the bookmarks view and `!settings` for further configuration options, and you can add your own from the Preferences panel.

Tabs also appear in the titlebar next to where they've been summoned, ensuring that the main view remains uncluttered and entirely focused on web rendering. There's no UI text, only icons, and there are only two drop-down menus at opposite ends of the top bar. The Preferences page lets you tweak the excellent built-in ad-blocking, set the appearance to light or dark mode, enable user scripts, and create a



A minimal web browser such as Min is useful if you want to use the web without being so distracted.

titlebar separated from the URL and tabs. Tabs themselves can be grouped into Tasks, which are accessible from the second drop-down menu. There's also a reader mode and a focus mode, with the latter disabling new tabs from being opened to keep you from watching YouTube while you're supposed to be working. It's the perfect balance between functional and minimal and a genuinely refreshing change from the establishment.

Project Website

<https://minbrowser.org>

x86 on ARM

FEX

With ARM64 becoming an ever more viable platform for general Linux desktop and laptop use, we find ourselves at an exciting juncture. For the first time in a generation, there's a split between a dominant and well-established architecture, x86 and x86-64, and a nascent newcomer, ARM-based AArch64. Most distributions now have a quickly maturing ARM build of their latest releases, and open source means almost everything can be rebuilt without too much difficulty, but there's still a transition period for some software, especially on distributions such as Arch where not everything in the AUR offers an ARM binary and proprietary software with only x86-based builds. Apple is tackling this

problem on its M1 and M2 macOS machines with its incredibly performant Rosetta, which now works natively within Ubuntu virtualization on macOS, but FEX is trying to do the same thing for native Linux.

FEX is an x86 and x86-64 compatibility and translation layer for ARM64, enabling you to run x86 binaries from an ARM host. It only currently works with ARMv8.0 and ARMv8.1+, rather than the chips found on older Raspberry Pi devices, but it does run quickly and efficiently enough to be unnoticeable on modern hardware. It's a brilliant way to run games and other proprietary software that may not yet offer native ARM builds. On Ubuntu, there's a handy script that will install all FEX's necessary dependencies and add the project's PPA to your package archive. An x86 RootFS image of the same distribution is also needed to provide the runtime environment for your x86



Here's an x86 version of StepMania running with FEX on an ARM64 version of Ubuntu running virtualized on an M1 Mac.

executables, and the script will download this and let you choose to run it uncompressed or from the compressed image to save space. Either way, it works without any further setup. Type FEXBash and you immediately get an x86 version of Bash running within your own filesystem. Type `uname -a` and you'll see it's x86, and you can access many other x86 executables directly. Even graphics acceleration remains intact as it's passed through to your host system libraries, and it all works brilliantly.

Project Website

<https://github.com/FEX-Emu/FEX>

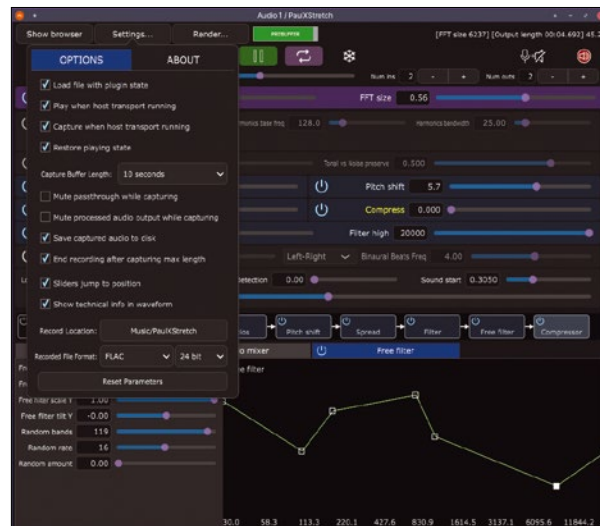
Audio effects

PaulXStretch

If you're into music production, or otherwise playing around with audio, time stretching is one of the most common audio effects you can use. This is because time stretching is both practical and creative. It's practical because it enables you to mix audio recorded at one speed with audio recorded at another, without any sign that the audio has been processed. It's creative because your audio becomes malleable in both duration and pitch. Both time stretching and pitch shifting are intrinsically linked because, to make a duration longer or shorter, audio is slowed down or speeded up. This affects its pitch. And the process to restore pitch isn't easily accomplished. The human voice, for example, contains formants that don't track the fundamental pitch of the voice. They need to be filtered out and separately processed, and there are the same caveats for any other type of

music, instrumentation, or percussion.

To accommodate all this audio complexity, all kinds of algorithmic techniques have been created for dealing with specific types of sound, and many of those techniques can also be used creatively. Which is exactly what this audio plugin, PaulXStretch, has been designed to do. The Paul in its name comes from the central algorithm it uses to process audio (Paul's Extreme Time Stretch, originally developed by Nasca Octavian Paul), but the plugin has been designed to give you quick and easy access to a comprehensive suite of time stretching related functions that can do anything from accurately changing the pitch of your audio to mangling it beyond all recognition. There are seven horizontal modules that can be selectively enabled or disabled, and these manipulate the stretch amount, harmonics, noise, frequency shift and spread, a



Alongside the pitch and time shifting, there are some excellent standard effects that can be used in isolation, including compression and equalization.

low- and high-pass filter, and even binaural processing. Their order can be changed with a flow diagram below the module parameters. Beneath this is a tabbed view to show the sampled audio waveform, the processed segment ratios, and an equalizer.

This might all sound complicated, but you can easily play with the controls without understanding their function, and the sound can range from slight tonal differences to huge, glitching feedback loops. You can either process a file or an incoming stream and send the output back to your audio application or to an exported file. Audio can be sampled and frozen, with a subsection selected from the waveform view for special focus. It works well as a DJ or musician's effect for real-time processing, but it's equally adept as a drop-in effect in Ardour or Bitwig. But what's especially remarkable about this plugin is its quality, and that could be because its developer usually makes commercial and proprietary plugins. PaulXStretch, however, is completely open source and can even be built as a shiny new and unexplored CLAP plugin (which is how we tested it). The UI has a level of professionalism you don't often see in open source plugins. While there are many different options crammed into its single window, they're easy to navigate and don't overwhelm the UI with complexity. This is a genuine accomplishment that's quite apart from the amazing sound processing the plugin is capable of, and it's brilliant to see effects like these making their way to Linux.



PaulXStretch is one of the first audio plugins we've been able to build and use as a CLAP plugin, making it ideal for powerful Linux audio processing and manipulation.

Project Website

<https://github.com/essej/paulxstretch>

Multiplayer board games

Tabletop Club

Network gaming has been around for almost as long as there's been gaming. You could play text-based Multi User Dungeons (MUDs) on university mainframes in the late 1970s, and there were several brilliant 8-bit home computer games that could link up via a null modem cable for multiplayer fun. Null-modem gaming begat dialup modem gaming on bulletin board systems which eventually led to the Internet. But with the exception of games such as chess, checkers, and Scrabble, the idea of playing an already established tabletop game online is relatively new. This could be because the intellectual property for so many games is so closely guarded. While you might be able to buy and play an individual game, recreating the

spontaneity of being with friends and choosing a game seems almost impossible.

This is where Tabletop Club can help. It's a physics-driven 3D multiplayer game canvas designed to help you easily recreate your favorite board games so you can then play them online with your friends. The key to being able to do this is asset packs. An asset pack is a bundle of the boards, pieces, cards, sound graphics, and save-game rules required for a game to run, save, and be restored. The game logic doesn't need to be implemented, because you're playing with other people, just as you would with a real board game. You all need to voluntarily stick to the rules, and Tabletop Club bundles chess and checkers to help you get started. The documentation is



Built with the Godot games engine, Tabletop Club makes it easy to play your favorite board games online.

brilliant and includes a few easy-to-follow tutorials to help you create asset packs for your own games. As you might expect, there's an active community behind the project, and hopefully users will share their own asset packs for their own favorite games. If you've not tried something like this before, you might be expecting a poor imitation of the IRL experience, but Tabletop Club is actually a lot of fun when combined with a video chat with your fellow players. It's the next best thing to actually being there.

Project Website

<https://github.com/drwhut/tabletop-club>

3D Racing

Stunt Rally

Stunt Rally is a 3D driving game that's been under development for over a decade. Like Trigger Rally, which we looked at previously, Stunt Rally's graphics engine is based on VDrift, an earlier 3D driving game with many of the same core driving features and a track editor. The track editor has been key to the success of all three titles because it means anyone can build a track, and there are typically dozens from which to choose. Stunt Rally itself includes 202 different tracks, 37 different scenarios, and 25 different vehicles, offering a huge amount of variety. The game itself is modeled on early 3D console rally games that were trying to be relatively realistic, rather than the arcade style of something like V-Rally. This can make them

harder to play, but more rewarding after you've learned how a vehicle handles and where a course goes. Stunt Rally is no different.

You start by choosing your track. With so many to choose from, almost every possibility is covered, from mountain forests to tropical beaches, with even a detour to an alien landscape. Because Stunt Rally focuses on the extreme side of the sport, most levels include jumps and other hazards to navigate, and the course selector highlights these as attributes alongside an image of the selected terrain. You can then choose your vehicle, which has its own excellent statistics page to help you see attributes such as weight, power, top speed, and stop times. You can change your car's color and see online fastest times for the selected course



If you enjoy console rally games from the early 2000s, you'll love Stunt Rally.

with your chosen vehicle. Races can be offline and online, and you can race ghost versions of your previous fastest runs. The graphics are excellent, with varied terrain, densely populated forests, and dynamic textures, but it's the gameplay that makes you want another go. The controls themselves are simple, but getting a fast time for a course is difficult, requiring many replays to master. Stunt Rally gets this balance absolutely right, with enough challenge and progress to make you always want another go.

Project Website

<https://stuntrally.tuxfamily.org>

Build LEGO models with LDraw and LeoCAD

Master Builder

LDraw and LeoCAD help you become a virtual LEGO architect.

BY DANIEL TIBI

For decades, young and old alike have enjoyed the well-known bricks by Danish toy manufacturer LEGO. The success story began in 1932 with wooden toys. In 1949 the first plastic bricks went into production, and since 1958 they have been available in the form that remains popular. Today, LEGO is far more than just

a children's toy; there are also many adult LEGO fans. LDraw and LeoCAD help you plan your own LEGO models on your PC.

Before you get started, there are two things you need to do. First, install LDraw [1] on your system. This is an open standard for LEGO CAD programs. The tool comes with 3D models of the bricks and a file format for the notation of the models you build with the bricks. Download the complete brick list [2] and unzip the ZIP file to the `~/ldraw/` folder or another directory. The bricks included in LDraw are constantly updated.

The second step is to install a LEGO CAD program, which you use to turn the virtual bricks into digital models. Of the various programs that use the LDraw standard [3], we chose the current 21.06 version of LeoCAD [4] for this article. For the sake of simplicity, it is a good idea to download the ApplImage variant [5], which you can launch directly without installing. Alternatively, pick up the source code from GitHub and compile the program yourself [6].

Listing 1: Tux.ldr

```
0 Tux
0 Name: Tux.ldr
0 Author: Daniel Tibi
1 14 -10 -8 -10 1 0 0 0 1 0 0 0 1 49673.dat
1 14 10 -8 -10 1 0 0 0 1 0 0 0 1 49673.dat
1 0 0 -8 10 1 0 0 0 1 0 0 0 1 28653.dat
0 STEP
1 0 0 -16 0 1 0 0 0 1 0 0 0 1 3022.dat
0 STEP
1 15 0 -40 -10 1 0 0 0 1 0 0 0 1 3660b.dat
1 0 0 -40 10 -1 0 0 0 1 0 0 0 -1 6227.dat
0 STEP
1 15 0 -48 -20 1 0 0 0 1 0 0 0 1 94148.dat
1 0 0 -48 10 1 0 0 0 1 0 0 0 1 28653.dat
0 STEP
1 0 10 -72 10 0 0 -1 0 1 0 1 0 0 4286.dat
1 0 -10 -72 10 0 0 1 0 1 0 -1 0 0 4286.dat
0 STEP
1 15 0 -72 -10 1 0 0 0 1 0 0 0 1 6227.dat
0 STEP
1 0 0 -80 0 1 0 0 0 1 0 0 0 1 94148.dat
0 STEP
1 14 0 -88 -20 1 0 0 0 1 0 0 0 1 33909.dat
1 0 0 -88 10 1 0 0 0 1 0 0 0 1 6225.dat
0 STEP
1 0 0 -112 10 1 0 0 0 1 0 0 0 1 3004.dat
0 STEP
1 0 -10 -112 -10 1 0 0 0 1 0 0 0 1 4070.dat
1 0 10 -112 -10 1 0 0 0 1 0 0 0 1 4070.dat
0 STEP
1 15 -10 -100 -22 0.866025 0 -0.5 -0.5 0 -0.866025 0 1 0 10238.dat
1 15 10 -100 -22 0.866026 0 -0.5 -0.5 0 -0.866026 0 1 0 10238.dat
0 STEP
1 0 0 -120 0 1 0 0 0 1 0 0 0 1 3068b.dat
```

Digital Bricks

LDraw provides information about the available LEGO bricks and the models, while LeoCAD displays them graphically and lets you edit them. LDraw stores the information in simple text files: `.dat` files describe the individual bricks. They are located in the `~/ldraw/parts/` directory or in the `parts/` folder in the path in which you unpacked the LDraw archive. For each brick there is a separate file, each with the part number.

In the `models/` subdirectory, you will find two examples of models in LDraw: a rudimentary pyramid and a car. Files with models end with `.ldr`. Listing 1 shows an example. The first three lines contain the title, file name, and author's name. Each line starting with a 1 represents a brick. The number following it determines the color of the brick, and the remaining numbers indicate the position and rotation. At the end of each line there is the file with information about the brick in question. The individual steps for assembling a model are separated by lines containing `0 STEP`. This means that a model can be described in a simple text file.

If you just entered the models in files as abstract numeric sequences, the build would not be much fun. This is where LeoCAD comes in. It outputs the models graphically and lets you edit your creations. This is how the model in Figure 1 was created from the text file in Listing 1.

Virtual Building Fun

When you launch LeoCAD for the first time, it needs information on where the LDraw files for each LEGO brick are stored. LeoCAD itself comes with a small selection of bricks, but this would severely limit your creativity. In the *View | Settings* menu, open a dialog and go to the *General* tab. Then, in the *Parts Library* field, enter the path to the folder where you previously unzipped the LDraw files, for example, `~/lDraw/`.

Now you get started with the virtual build. Below the menubar and toolbar, the LeoCAD

window is divided into two areas. On the left you will find what is initially an empty base plate. The plate grows automatically as soon as you place bricks close to one of its edges. Top right you will see a ball that you can rotate with the mouse. This lets you move the model in any direction to view it from all sides. Pressing *H* switches back to the original view. You can press *+* to zoom into the model and *-* to zoom out.

On the right side of the window, LeoCAD displays a list of all available bricks at the top, followed by an overview of all colors underneath. For a better overview, simply filter the selection. At startup time, the filter defaults to *Brick*. You can search for bricks using the search field below. Either type in the part number you are looking for (e.g., *3002* for a brick with two-by-two studs) or search for the size of a brick. To do this, enter something like *2 x 2*. Pay attention to the spaces;

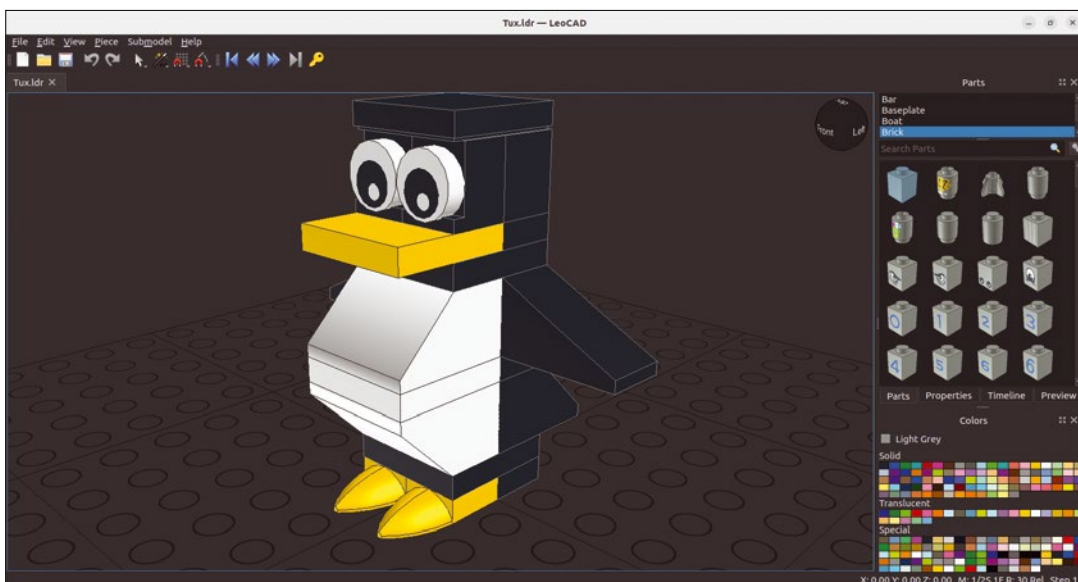


Figure 1: LeoCAD's graphical representation of Tux in bricks.

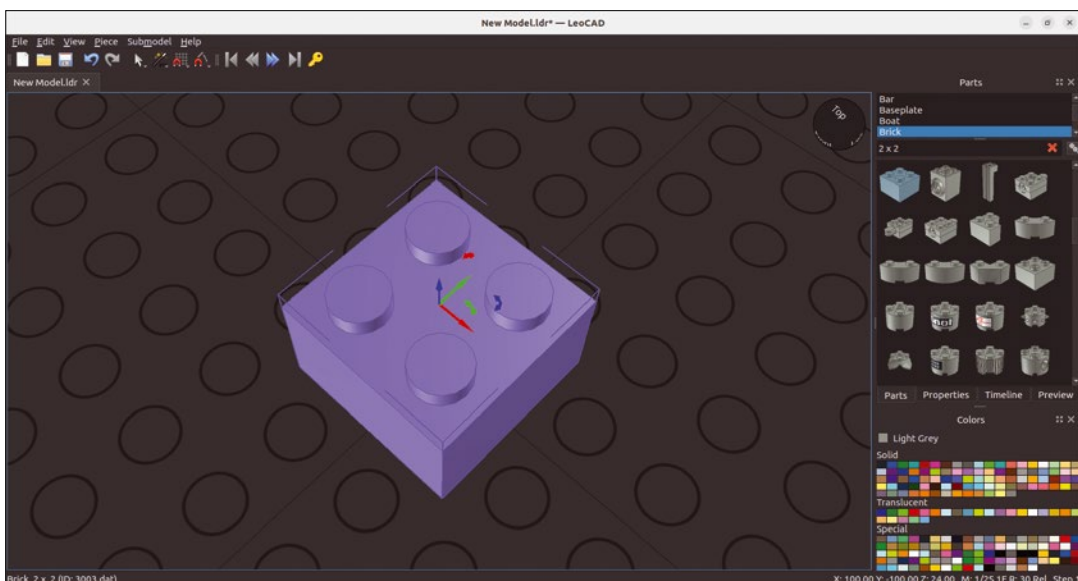


Figure 2: Align bricks in all directions using the six arrows.

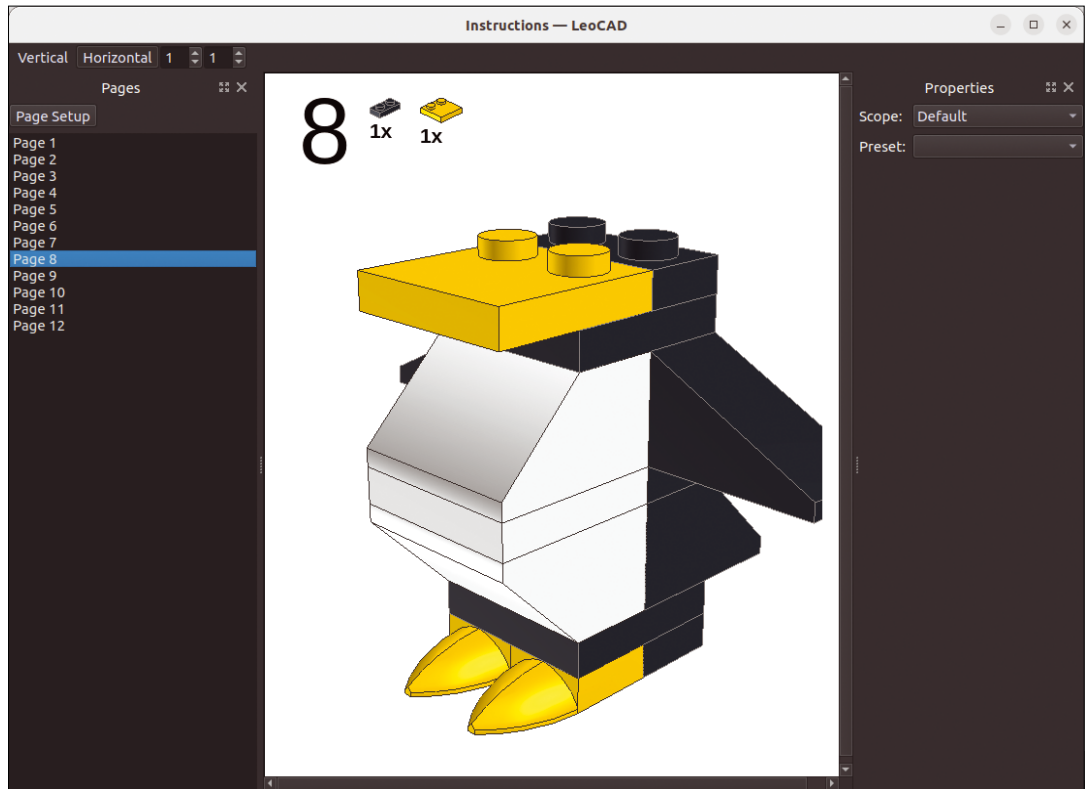


Figure 3: The instructions show you how to assemble Tux in 12 steps from bricks.

otherwise you will not see any results. In addition, you can find a specific brick by typing its name, such as *slope brick*. Use *Insert* to integrate the appropriate brick into the model or drag it to the desired position with the mouse.

Now you need to insert the brick correctly. To do this, click on the brick to select it. LeoCAD marks the selected brick with six arrows (Figure 2). The

three straight arrows let you move the brick back and forth, left and right, and up and down. Alternatively, you can use the arrow keys and the Page Up and Page Down keys. The three curved arrows are used to rotate the brick on all three axes. To do this, use the same keys as you used for moving again, but this time in combination with the Shift key.

During a build, you will often use single bricks or a certain arrangement several times. Instead of reinstalling and aligning each brick individually, you can simply select the brick in question (or the respective combination if you hold down Ctrl) in the model and then duplicate the selection using Ctrl+D. Then move the cloned elements to the desired location.

Up to this point, your model is still gray – high time to add some color. You can see the available colors bottom right in the window. As soon as you click on one of the colors, the bricks in the selection list on the right are painted in that color. If you want to color bricks that have already been installed, first select the desired color and then select the bricks in question. Then select the *Part | Paint Selected* menu item to finally color the bricks.

Build Instructions

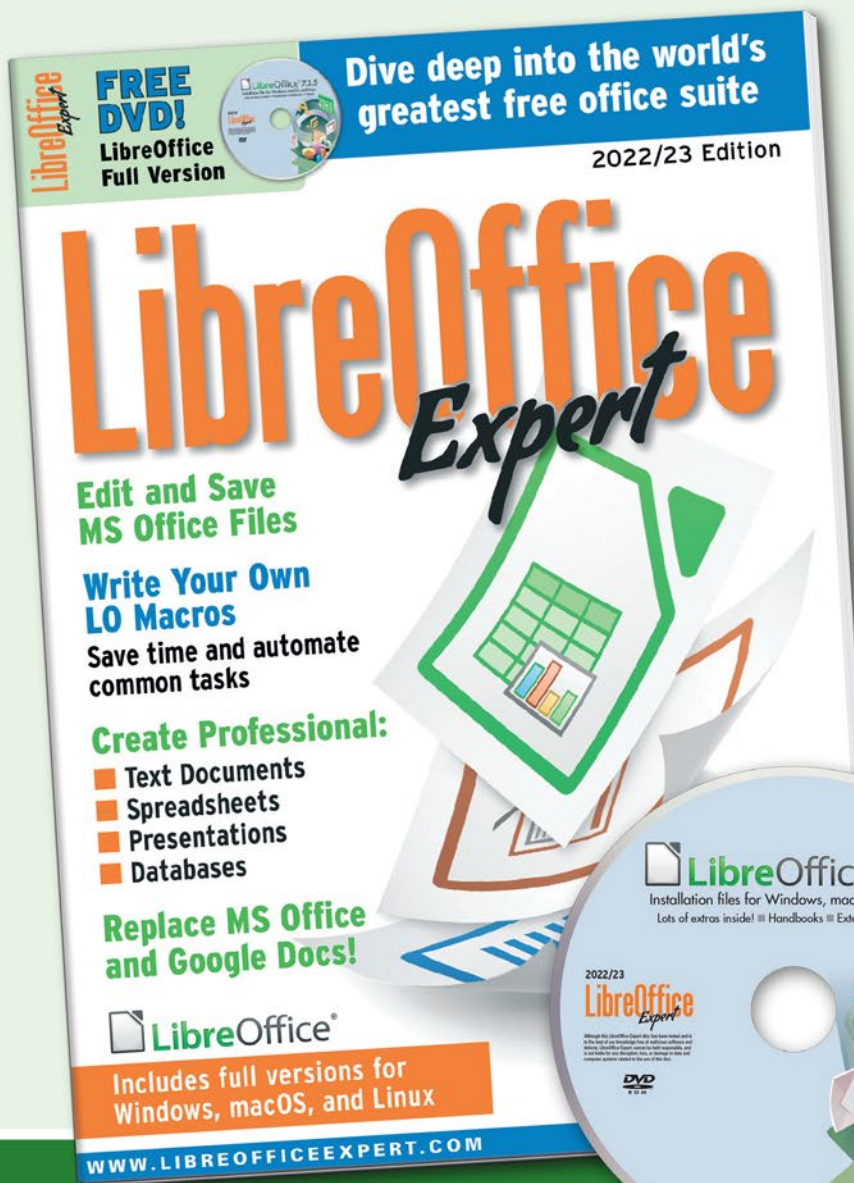
If you want to rebuild your virtual model later with real bricks, it makes sense to follow the step-by-step instructions. To do this, insert a new build step at appropriate points when assembling the virtual model by pressing Alt+Right arrow. The *File | Instructions* menu item displays detailed instructions

Part	Black	Yellow	White	Total
=Tile 1 x 1 Round with Eye Pattern			2	2
=Plate 1 x 2	2			2
Brick 1 x 2	1			1
Plate 2 x 2	1			1
Tile 2 x 2 with Groove	1			1
Plate 2 x 2 with 2 Studs on One Edge		1		1
Slope Brick 45 2 x 2 Inverted with Inner Stopper Ring			1	1
Brick 1 x 1 with Headlight	2			2
Slope Brick 33 3 x 1	2			2
=Plate 1 x 1 with Tooth		2		2
=Plate 1 x 2	1			1
=Slope Brick 45 2 x 2	1		1	2
=Plate 2 x 2	1		1	2
Total	12	3	5	20

Figure 4: *Parts Used* lists all the bricks for a project; switch to *Summary* to see the model's size.

Shop the Shop
shop.linuxnewmedia.com

Become a LibreOffice Expert



Explore the **FREE** office suite used by busy professionals around the world!

Create Professional:

- Text Documents
- Spreadsheets
- Presentations
- Databases

Whether you work on a Windows PC, a Mac, or a Linux system, you have all you need to get started with LibreOffice today. This single-volume special edition will serve as your guide!

Order online:
shop.linuxnewmedia.com/specials

For Windows, macOS, and Linux users!

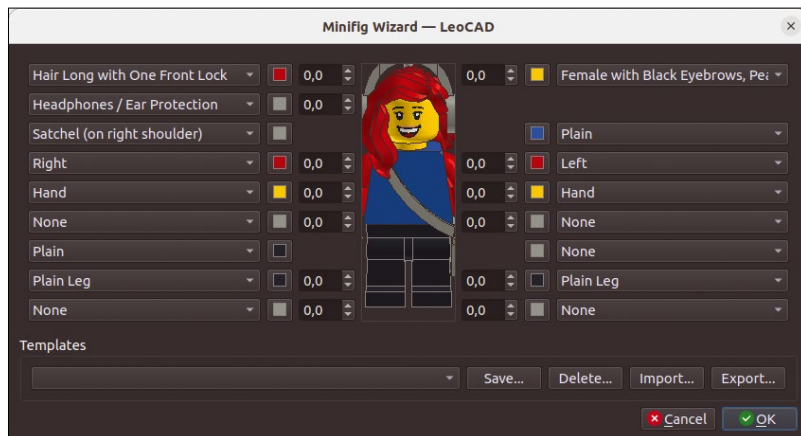


Figure 5: A wizard helps you assemble miniature figures and kit them out with accessories.

for the finished model, recording the bricks required for each step of the build (Figure 3).

A list of all bricks used in a model is provided by the *Parts Used* tab from the *Submodel | Properties* menu item (Figure 4). Switch to the *Summary* tab and you will see the size of the model. This gives you an idea of the dimensions the model will have when built with real bricks.

Miniature Figures

Miniature figures give you more lively models. LeoCAD offers a wizard for this purpose, which you can launch via the *Part | Minifig Wizard* menu (Figure 5). A dialog opens where you can assemble the individual parts of a miniature figure – hair, face, body, and accessories – and specify colors. Next to some elements, you will also find an input field for a number that you can use to rotate the element in question. For example, you can rotate the face or bend the arms, hands, and legs.

Press the *Save* button to store a minifig in your collection and *Delete* to remove it. *Export* saves

the minifig data in a separate file, and *Import* loads the data for a figure from an external file. Once you are satisfied with your minifig, press *OK* to insert it into the model, such as the coffee house shown in Figure 6.

Exporting

LeoCAD offers several options for exporting. You can use the *File | Save Image* menu item to turn a model into a PNG graphic. For more options, see *File | Export*. You can create an HTML file from your model or write a list of the bricks used for it to a CSV or XML file. You can reuse the bricks, for example, on an online marketplace such as BrickLink [7].

Conclusions

Let your creativity run wild with LDraw and LeoCAD. Building with the virtual bricks takes some practice, but is just as fun as using real bricks. LeoCAD also lets you create build instructions and lists of the bricks used, so you and others can easily recreate your creations. ■■■

Info

- [1] LDraw: <https://www.ldraw.org>
- [2] Brick list download: https://wiki.ldraw.org/index.php?title=Getting_Started_-_Linux
- [3] Programs that support the LDraw standard: <https://www.ldraw.org/downloads-2/third-party-software.html>
- [4] LeoCAD: <https://www.leocad.org/>
- [5] LeoCAD ApplImage download: <https://www.leocad.org/download.html>
- [6] LeoCAD on GitHub: <https://github.com/leozide/leocad/releases>
- [7] BrickLink: <https://www.bricklink.com/v2/main.page>

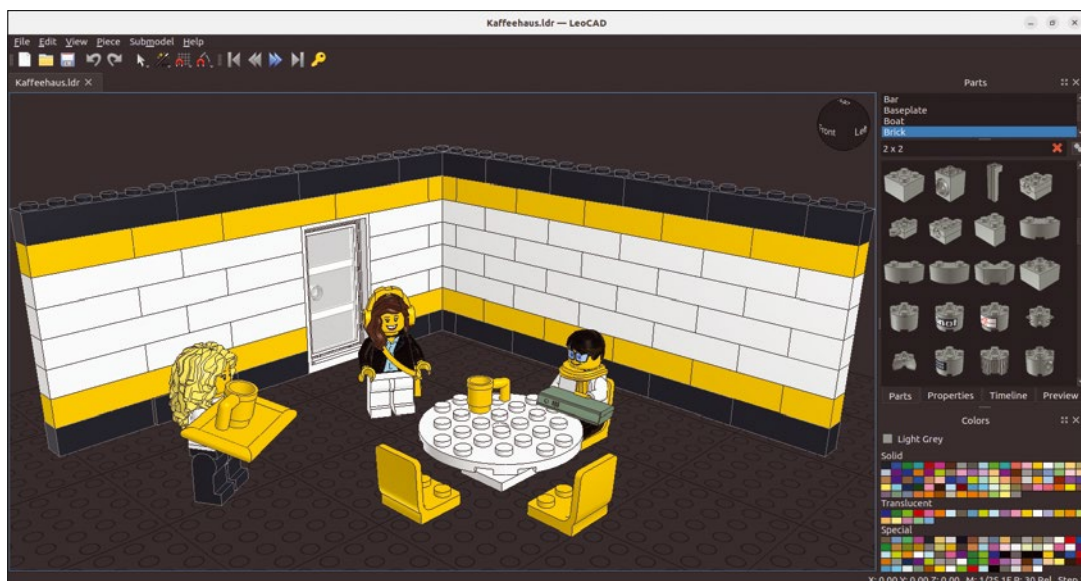


Figure 6: Miniature figures make models look more alive.

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.



#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



#263/October 2022

Build an IoT Linux

The most amazing thing about Linux is its flexibility. Linux systems run on the biggest computers in the world – and on many of the diminutive devices that populate your home environment. If you've always wondered how developers adapt Linux to run on tiny tech, you'll appreciate this month's stories on Buildroot and the Yocto project.

On the DVD: Linux Magazine Archive issues 1-262



#262/September 2022

Beyond 5G

Behind the scenes, the cellular phone network has always been the preserve of highly specialized and proprietary equipment, but some recent innovations could be changing that. This month we explore the Open RAN specification, which could one day allow more of the mobile phone network to operate on off-the-shelf hardware.

On the DVD: openSUSE Leap 15.4 and MX Linux 21.1

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.

CloudFest

Date: March 21-23, 2023

Location: Europa-Park, Germany

Website: <https://www.cloudfest.com/>

CloudFest is the #1 internet infrastructure event in the world, connecting the global cloud computing industry. Join us March 21–23 as we take over a European amusement park to bring together the top decision-makers in the Internet infrastructure industry for learning, networking, and the best parties ever.

Linux App Summit

Date: April 21-23, 2023

Location: Brno, Czech Republic & Online

Website: <https://linuxappsummit.org/>

The Linux App Summit (LAS) brings the global Linux community together to learn, collaborate, and help grow the Linux application ecosystem. Through talks, panels, and Q&A sessions, we encourage attendees to share ideas, make connections, and join our goal of building a common app ecosystem. Join us in-person or online.

DORS/CLUC 2023

Date: May 11-12, 2023

Location: Zagreb, Croatia

Website: <https://www.dorscluc.org/>

DORS/CLUC is the oldest and biggest regional conference that covers free and open source software, open standards, and Linux. During two days of talks, workshops, and fun, this event brings together students, FOSS enthusiasts, tinkerers, developers, companies, and more to learn, network, do business, and plan projects together – all with a focus on free software and Linux.

Events

GeekBeacon Festival	Feb. 18	Virtual Event	https://gbfest.org/
Kickstart Europe 2023	Feb. 20-21	Amsterdam, Netherlands	https://www.kickstartconf.eu/
FAST'23	Feb. 20-23	Santa Clara, California	https://www.usenix.org/conference/fast23
SCaLE 20x	Mar. 9-12	Pasadena, California	https://www.socallinuxexpo.org/blog/scale-20x
Cassandra Summit	Mar. 13-14	San Jose, California + Virtual	https://events.linuxfoundation.org/
FOSS Backstage 2023	Mar. 13-14	Berlin, Germany + Online	https://23.foss-backstage.de/
Everything Open 2023	Mar. 14-16	Naarm (Melbourne), Australia	https://2023.everythingopen.au/
Women in CyberSecurity	Mar. 16-18	Denver, Colorado	https://www.wicys.org/events/wicys-2023/
CloudFest	Mar. 21-23	Europa-Park, Germany	https://www.cloudfest.com/
Open Source 101	Mar. 23	Charlotte, North Carolina	https://opensource101.com/
KubeCon + CloudNativeCon Europe 2023	Apr. 17-21	Amsterdam, Netherlands	https://events.linuxfoundation.org/
PyCon US 2023	Apr. 19-27	Salt Lake City, Utah	https://us.pycon.org/2023/
Hybrid Cloud Conference	Apr. 20	Virtual Event	https://www.techforge.pub/events/hybrid-cloud-congress-2/
Linux App Summit 2023	Apr. 21-23	Brno, Czech Republic	https://linuxappsummit.org/
Cloud Expo Europe	May 10-11	FrankfurtFrankfurt, Germany	https://www.cloudexpo-europe.de/en
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
CloudFest USA	May 31-Jun. 3	Austin, Texas	https://www.cloudfest.com/usa/

CALL FOR PAPERS

We are always looking for good articles on Linux and the tools of the Linux environment. Although we will consider any topic, the following themes are of special interest:

- System administration
- Useful tips and tools
- Security, both news and techniques
- Product reviews, especially from real-world experience
- Community news and projects

If you have an idea, send a proposal with an outline, an estimate of the length, a description of your background, and contact information to edit@linux-magazine.com.



The technical level of the article should be consistent with what you normally read in *Linux Magazine*. Remember that *Linux Magazine* is read in many countries, and your article may be translated into one of our sister publications. Therefore, it is best to avoid using slang and idioms that might not be understood by all readers.

Be careful when referring to dates or events in the future. Many weeks could pass between your manuscript submission and the final copy reaching the reader's hands. When submitting proposals or manuscripts, please use a subject line in your email message that helps us identify your message as an article proposal. Screenshots and other supporting materials are always welcome.

Additional information is available at:

http://www.linux-magazine.com/contact/write_for_us.

Contact Info

Editor in Chief

Joe Casad, jcasad@linux-magazine.com

Copy Editors

Amy Pettle, Aubrey Vaughn

News Editors

Jack Wallen, Amber Ankerholz

Editor Emerita Nomadica

Rita L Sooby

Managing Editor

Lori White

Localization & Translation

Ian Travis

Layout

Dena Friesen, Lori White

Cover Design

Dena Friesen

Cover Image

© neyro2008 and Uladzimir haikou, 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.

Authors

Erik Bärwaldt	80
Zack Brown	12
Bruce Byfield	6, 22, 32, 42, 54
Joe Casad	3
Mark Crutch	71
Hans-Georg Eßer	73
Daniel Etzold	16
Jon "maddog" Hall	72
Swen Hopfe	58
Rubén Llorente	62
Vincent Mealing	71
Pete Metcalfe	34
Graham Morrison	84
Thomas Reuß	28
Mike Schilli	44
Ferdinand Thommes	24, 50
Daniel Tibi	40, 90
Jack Wallen	8

Approximate

UK / Europe **Mar 10**

USA / Canada **Apr 14**

Australia **May 26**

On Sale Date

Please note: On sale dates are approximate and may be delayed because of logistical issues.

Issue 269 / April 2023

The Fediverse

Social media corporations are in the news – why should a few private companies have so much power? According to proponents of the Fediverse, *they shouldn't*. Next month, you'll learn about this thriving offshoot of the free software movement and some of the leading Fediverse tools that are bringing social media back to the people.



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 © Anna Ivanova, 123RF.com

FOSS Backstage

Community, Management & Compliance

20
23

BACKSTAGE

FOSS

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



GET YOUR TICKETS NOW & USE CODE "LINUXMAG10"



2

days of
engaging talks
& discussions

150+

online
attendees

40+

experienced
speakers

5

tracks

Lots

of possibilities
to network &
meet great
people

120+

onsite
attendees

2

stages



23.foss-backstage.de

March 13-14, 2023
@ TUECHTIG Berlin & Online

HETZNER

LOCATED IN THE USA
NOW AT EAST & WEST COAST



CLOUD SERVER

STARTING AT

\$ **4.35**

monthly | incl. IPv4

HETZNER CLOUD SERVER CPX11

- ✓ AMD EPYC™ 2nd Gen
- ✓ 2 vCPU
- ✓ 2 GB RAM
- ✓ 40 GB NVMe SSD
- ✓ 20 TB traffic inclusive
- ✓ Intuitive Cloud Console
- ✓ Located in Germany, Finland or USA



HIGH QUALITY - UNBEATABLE PRICES



DEPLOY YOUR
HETZNER CLOUD
IN UNDER
10 SECONDS!

MANAGE YOUR CLOUD QUICK AND EASY WITH FEATURES LIKE
LOAD BALANCER, FIREWALLS, ONE CLICK APPS AND MANY MORE!

GET YOUR CLOUD NOW



CLOUD.HETZNER.COM