

Repurpose Your Old Router
3 easy projects for makers!



LINUX PRO

MAGAZINE

ISSUE 253 – DECEMBER 2021

OpenBSD

Is this
meticulous
FOSS cousin
an alternative
to Linux?



LINUXVOICE

BOINC: Crowdsourcing
the COVID-19 fight

rlxos: Stay safe with an
immutable filesystem

**Roll your Own
Note-Taking Tool**

Blender Workshop: Take your
first steps with 3D graphics

UP CLOSE WITH
10 Fantastic FOSS Tools!



LINUX NEW MEDIA
The Pulse of Open Source



CLOUD EXPO EUROPE

8. – 9. December 2021 Messe Frankfurt
www.cloudexpo-europe.de

Meet | Back | Better

#NurMitEuch

www.cloudexpo-europe.de/linuxmagazine

MORE ON DISRUPTION

Dear Reader,

If you stay in this job long enough, you get to see the progression of technologies as they move from mere ideas, to prototypes, to real-world products ready to challenge the world and ascend the ladder to that mystical measure of success that the venture capitalists call “disruption.”

Eight years ago – in the July 2013 issue, to be exact – I wrote about a quirky little story in the news that fell on one of those eternal fault lines of American culture. Some gun advocates had published plans for a 3D-printed gun, which they called the “Liberator.” At that time, 3D gun printing was little more than a concept – something for the opposing sides of the gun control debate to face off around. The Liberator, which was made of plastic (and, I should add, the kind of plastic that was available for 3D printers eight years ago), didn’t look like much and didn’t shoot very accurately. As I recall, some considered the gun a significant danger to the owner. In one test, it shattered with first use, but the tester later admitted that “Printed under the right conditions, the Liberator gun has a lifespan of 8-10 shots” [1].

After a brief run in the headlines, 3D guns slipped out of sight for most of us. The Liberator really wasn’t reliable enough to serve as a sidearm for either the good guys or the bad guys, and, as I stated in my column eight years ago, there are “many easier ways of getting a gun than printing one on an \$8,000 printer.”

But that was then.... Technology has a way of marching on, soaring higher, pushing back against all barriers. The Los Angeles Police Department (LAPD) reported this month that so-called “ghost guns” assembled from 3D-printed parts have contributed to more than 100 violent crimes in the past year. According to the report, ghost guns in LA have been involved with 24 murders, 60 assaults, and 20 armed robberies – and that’s just for one city. The report refers to the proliferation of these ghost guns, which have no serial number and are virtually untraceable, as an “epidemic” [2]. The LAPD confiscated more than 800 ghost guns in the first half of 2021. According to the report, felons who are banned from possessing firearms due to previous convictions are increasing turning to ghost guns to minimize the chances of getting caught.

Although they have largely been under the public radar, 3D-printed guns and gun parts have had an extensive run in the courts over the last eight years, and many questions remain about what is legal and what is regulatable. Is this the moment when I launch into an impassioned plea for (or against) gun control? No – there are other places to hear that kind of stuff. I talk about tech. We all knew this was coming – ever since the first shot from that first plastic Liberator. 3D printers have gotten better and less expensive, and plans for 3D guns have gotten more sophisticated. Are we ready for this? Do we have the regulations in place to contain the epidemic of untraceable ghost guns? We are giving our law enforcement agencies a brand new challenge they didn’t have before, and we are taking away a very useful tool of their trade (gun tracing by serial number). Are we going to provide them with additional resources to take on these tasks, or do we expect them to divert funding from other priorities in order to chase after ghost guns?

In May of this year, the Biden administration proposed new rules that would hold the sellers of home-assembly gun kits to the same rules that conventional gun sellers face, including background checks on buyers and a unique serial number on each gun. This effort seems like a sensible first step for addressing the ghost gun epidemic. Even if you oppose background checks, you could make the case that the same rules should apply to all gun makers equally, rather than the government giving a free pass for weapons assembled through a glitzy technological paradigm.

When I consider the eight-year history of 3D-printed guns in the news, it makes me wonder what other emerging technologies are out there now that we should be planning for before they land in our laps. Disruption works much better if you come down out of the clouds and prepare for the messy details.

Joe

Joe Casad,
Editor in Chief



Info

- [1] Liberator 3D Gun:
[https://en.wikipedia.org/wiki/Liberator_\(gun\)](https://en.wikipedia.org/wiki/Liberator_(gun))
- [2] “LAPD Declares Ghost Guns an Epidemic”:
<https://news.yahoo.com/lapd-declares-ghost-guns-epidemic-013233309.html>

ON THE COVER

30 **Fighting COVID-19 with BOINC**

Could distributed computing be the answer?

56 **Homegrown Notes Tool**

Build a cool tool for taking notes and sending reminders.

88 **Blender Workshop**

This easy first project will help you get started with the powerful Blender 3D modeling and animation tool.

42 **rlxos**

Protect your system from break-ins and breakage with an immutable filesystem.

60 **Repurposed Router Projects**

Three maker projects for your washed-up router.

NEWS

08 **News**

- Linux Now Runs on Apple's M1 Chipset
- MX Linux 21 RC Now Available
- Fedora 35 Improves Desktop Performance
- Extended Support For Ubuntu 14.04 and 16.04
- Gnome 41 Adds Desktop Improvements
- Black Lotus Labs Confirms Flaw in Windows Subsystem for Linux

12 **Kernel News**

- Renaming SMB
- Testing Standards

COVER STORY

16 **OpenBSD for Linux Users**

The classic family of FOSS operating systems known as the BSDs remain a mystery to many in the Linux community. With the upcoming release of OpenBSD 7.0, it is time to throw some light on this little gem.

REVIEWS

22 **Distro Walk – Qubes OS**

Andrew David Wong discusses the Qubes OS project's security-by-compartmentalization approach, including an endorsement from Edward Snowden.

26 **AltSearch**

AltSearch offers extended functionality to LibreOffice Write's default find and replace tools, making it ideal for editing and formatting longer documents.

IN-DEPTH

30 **Fighting COVID-19 with BOINC**

Linux and the BOINC distributed computing platform help researchers fight the COVID-19 virus.

34 **Charly's Column – VM Detection**

To write low-level scripts, you need to know whether you are currently on a physical or a virtual machine. Charly finds out with a couple of clever hacks.

36 **Command Line – Hardware Information**

A quick guide to 10 command-line tools to help you find hardware information.

42 **rlxos**

With its immutable filesystem, rlxos prevents a broken system while simultaneously allowing changes via OverlayFS.



16 OpenBSD

BSD Unix has been around longer than Linux, and it still has a loyal following within the Free Software community. This month we explore the benefits of a leading BSD variant from the viewpoint of a Linux user.

IN-DEPTH

48 Programming Snapshot – Disk Speedo

To keep an eye on the remaining disk space during storage-intensive operations, check out this speedometer/odometer written in Go.

52 Hard Disk Sentinel

Hard Disk Sentinel monitors mass storage devices with a fully automated process minus the bells and whistles.

MakerSpace

56 Homegrown Notes Tool

If you're tired of the privacy problems and feature bloat of high-end note-taking utilities, try rolling your own.

60 Repurposed Router Projects

If you have an old router lying around, you can put it to good use with a few easy projects and learn something along the way.

LINUXVOICE

65 Welcome

This month in Linux Voice.

67 Doghouse – Multi-Factor Authentication

maddog looks at multi-factor authentication.

68 Worker

With over 20 years of development, Worker offers a tested and functional two-panel file manager.

74 Clapper and GTK4

The Clapper media player showcases new desktop design features in GTK4.

76 G'MIC

Behind G'MIC's deceptively simple interface hides a mighty image processing framework.

80 FOSSPicks

This month Graham looks at Bespoke, Waydroid, OpenShot, pedalboard, Onivim 2, Mr. Rescue, and more!

88 Tutorial – Blender Perfume Bottle

Creating this simple 3D image will give you a whiff of Blender's power.



Tails 4.22 and Q4OS 4.6

Two Terrific Distros on a Double-Sided DVD!



Tails 4.22 64-bit

Tails 4.22 (the Amnesic Incognito Live System) is the latest release of one of the most popular security-oriented distributions. Based on Debian, Tails runs from an exterior device, either making all incoming and outgoing connections anonymous or blocking non-anonymous ones. It is especially well known for its Tor Browser, which provides an easy way to browse anonymously, control JavaScript, and remove ads. Other tools in Tails include Pidgin for encrypted instant messaging, OnionShare for anonymous file sharing, Thunderbird configured for encrypted email, and Electrum for bitcoin transactions.

A basic part of security and privacy is updates with the latest patches. In keeping with this criterion, Tails 4.22 consists largely of application updates. Other changes from earlier releases include reducing the time out when reconnecting from 60 seconds to 10 seconds and the ability to retry connections from the error screen. In addition, when an unsafe browser is run, Tails no longer restarts Tor automatically or mentions the existence of persistent storage. Moreover, when automatic updates are downloaded, Tails 4.22 ensures that a working mirror is used. All these changes are minor tweaks to improve security as well as convenience.

Over the years, Tails has become more user-friendly with each release. This constant improvement makes Tails 4.22 an ideal way for novices to secure their systems. However, be sure to read the documentation before using. While generally providing strong security, Tails has to be configured in certain ways for maximum security, and any installation can be only as secure as the base system from which it runs.



Q4OS 4.6 64-bit

Q4OS refers to itself as a "...fast and friendly, desktop-oriented operating system based on Debian Linux." The emphasis is on ease of use and a short learning curve. The Q4OS developers have slimmed down the standard Debian, removing unneeded services and components for a streamlined and efficient system. The desktop defaults to Plasma, but Q4OS also supports the cult favorite Trinity desktop, which forked from KDE all the way back in 2011 and has been in independent development ever since.

Q4OS 4.6 "Gemini" is a long-term support release, which means it will receive security patches and software updates for five years. The new release is based on Debian "bullseye" 11 and includes updates to the native Q4OS desktop profiler tool that allow the user to import and modify custom profiles. Other improvements include enhanced hardware support, as well as fixes and updates for many of the bundled applications.

Additional Resources

- [1] Q4OS: <https://q4os.org/>
- [2] Q4OS documentation <https://q4os.org/documents.html>
- [3] Trinity Desktop: <https://www.trinitydesktop.org/>

*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.

Turn your ideas into reality!

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

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

This special issue will help you dive into:

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

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



ORDER ONLINE:
shop.linuxnewmedia.com/specials

NEWS

Updates on technologies, trends, and tools

THIS MONTH'S NEWS

08

- Linux Now Runs on Apple's M1 Chipset
- MX Linux 21 RC Now Available

09

- Fedora 35 Improves Desktop Performance
- Extended Support for Ubuntu 14.04 and 16.04
- More Online

10

- Gnome 41 Adds Desktop Improvements
- Black Lotus Labs Confirms Flaw in Windows Subsystem for Linux

Linux Now Runs on Apple's M1 Chipset

It seems like only yesterday that a small group of developers began work on porting Linux to the new Apple M1 chipset. The journey was a struggle from day one, given how much proprietary hardware Apple uses. But the work has paid off and Asahi Linux, a community-based project centered around porting a distribution to the Apple M1 chipset, has finally succeeded in getting a usable Linux desktop on the hardware.

The engineers have merged various drivers and bindings for the 5.16 Linux kernel and even managed to work out the `pinctrl` driver, I2C driver, device power management, NVMe + SART, and DCP. Thanks to those new drivers, M1 Macs are now a viable option for the Linux operating system.

Before you jump on this, understand it's not perfect. Apple uses a proprietary PowerVR-based GPU, so the Linux desktop will come without GPU acceleration. It's also important to know that a proper installer has yet to materialize, which means users outside of the Asahi project are still not able to experience the Linux desktop on the M1 hardware. To that, Hector Martin, the head of the project, says, "Once we have a stable kernel foundation, we will start publishing an 'official' installer that we expect will see more wide usage among the adventurous."

For helping getting started, developers interested in trying out Asahi Linux on M1 hardware can head over to the project's IRC channel (`#asahi-dev`).

To find out more about Asahi Linux project's progress, check out their official Progress Report (<https://asahilinux.org/2021/10/progress-report-september-2021/>).



Photo by zhang kaiyv on Unsplash

MX Linux 21 RC Now Available

MX Linux (<https://mxlinux.org/>) is a midweight Linux distribution that aims to be simple and stable. The distribution is available in three different flavors: Xfce, KDE Plasma, and Fluxbox. MX Linux 21 RC is based on Debian 11 (bullseye), which includes all of the latest components and security patches.

All three editions include a new `mx-comfort` theme, and the developers have worked diligently to clear away as many bugs as possible for the release candidate.

The MX Linux 21 Xfce edition includes the Thunar Shares Plugin for the Thunar file manager (for Samba access) and a default user password for admin tasks. The Fluxbox edition panel now offers preconfigured setups, and the KDE Plasma

edition comes with an updated Dolphin file manager (which includes a fix for the “Save desktop changes” crash issue). All three versions benefit from new and updated applications, a new installer partition selection area (which includes some LVM support if an LVM volume already exists), and new UEFI Live system boot menus.

You can download the RC releases with the following links: Xfce (https://sourceforge.net/projects/mx-linux/files/Testing/RC1/Xfce/MX-21_rc1_x64.iso/download), KDE Plasma (https://sourceforge.net/projects/mx-linux/files/Testing/RC1/KDE/MX-21_KDE_RC1_x64.iso/download), and Fluxbox (https://sourceforge.net/projects/mx-linux/files/Testing/RC1/Fluxbox/MX-21_fluxbox_rc1_x64.iso/download).

Read more in the official MX Linux 21 RC release notes (<https://mxlinux.org/blog/mx-21-release-candidate-1-now-available-for-testing-purposes/>).

Fedora 35 Improves Desktop Performance

While Fedora 35 might not include the same level of game-changing, workflow-enhancing features found in Fedora 34 (thanks to Gnome 40), there's plenty to be excited about in this new Fedora iteration.

One of the more notable changes comes by way of improvements to the NVidia proprietary driver. Red Hat has been working diligently to help improve the NVidia/Wayland stack support, and the changes in Fedora 35 should go a long way to improve desktop performance across the board.

Fedora 35 also brings high-resolution mouse wheel support that will provide a much smoother wheel-scrolling experience. This change comes by way of the work done on libinput. The distribution also recently shifted from PulseAudio to PipeWire, and the system will see much maturation in this upcoming release.

Gnome will also add a “kiosk” mode, which can be applied to various use cases (such as info boards and POS machines). The user interface has been tweaked with the addition of the Libadwaita theme and power profiles are even more accessible.

Fedora Kinoite (https://fedoraproject.org/wiki/Changes/Fedora_Kinoite) is another very interesting addition. This new Fedora variant is an immutable desktop operating system similar to Fedora Silverblue (<https://silverblue.fedoraproject.org/>) but based on the KDE Plasma desktop. The Silverblue project aims to be an extremely stable and reliable desktop and is an excellent platform for developers and container-focused workflows.

To get an idea of how the release is shaping up, download a daily build of Fedora 35 (https://dl.fedoraproject.org/pub/fedora/linux/development/rawhide/Workstation/x86_64/iso/).

Extended Support for Ubuntu 14.04 and 16.04

Ubuntu 14.04 and 16.04 are Long Term Support (LTS) versions, both of which have already hit End of Life (14.04 in 2019 and 16.04 in 2021). The problem is, however, a large number of enterprise businesses are still making use of those versions of the open source platform. While you can upgrade to the latest LTS version of Ubuntu, that's not always an option for some use cases.

Because of this, Canonical (<https://canonical.com/>) has extended their support for both versions of Ubuntu to bring those releases in line with the new 10-year support period that was given to both 18.04 and 20.04 (both of which are also LTS releases).

Of course, there's a caveat: The additional support for 14.04 and 16.04 comes by way of Extended Support Maintenance, which requires an active Ubuntu Advantage subscription (<https://ubuntu.com/advantage>). For Ubuntu home users, this subscription is free (for up to three devices). For businesses, however, the subscription comes with a price.

For those businesses who need to extend the life of Ubuntu 14.04 or 16.04, the cost will depend on the type of service (Essential, Standard, or Advanced). Prices range from \$25.00 for the Essential package on a desktop to \$1,500 for the Advanced package on a physical server.

MORE ONLINE

Linux Magazine

www.linux-magazine.com

Paw Prints • Jon “maddog” Hall

FreedomBox: A personal server for Privacy and Security

Over the next couple of weeks, this blog will be a “living attempt” to acquaint people with the functionality and setup of a personal FreedomBox Internet server that is suitable for supporting one person or a community of people.

ADMIN HPC

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

Processor and Memory Affinity Tools

• Jeff Layton

It's called high-performance computing (HPC), not low-performance computing (LPC), not medium-performance computing (MPC), and not even really awful-performance computing (RAPC).

Darshan I/O Analysis for Deep Learning Frameworks

• Jeff Layton

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

ADMIN Online

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

Public key Infrastructure in the Cloud

• Andreas Philipp

A public key infrastructure in the cloud for secure digital communication maintains the security of an on-premises solution and reduces complexity.

App Proxy Support for Remote Desktop Services

• Florian Frommherz

Support flexible working environments with Remote Desktop Services and Azure AD Application Proxy.

Flexible Software Routing with Open Source FRR

• Benjamin Pfister

The FRR open routing stack can be integrated into many networks because it supports a large number of routing protocols, though its strong dependence on the underlying kernel means it requires some manual configuration.

Gnome 41 Adds Desktop Improvements

Gnome has been evolving at a breakneck pace. And no recent release proved that more than Gnome 40, where the entire workflow was reconfigured and reworked. For those that have experienced the shift that was brought about by Gnome 40, every update since has been nothing more than minor tweaks.

While Gnome 41 isn't doing a major overhaul, it still adds some important improvements to the desktop.



Photo by Brigitta Schneider on Unsplash

Such improvements include a revamped Gnome Software that brings to life a much livelier landing page and updated app categories. New-to-Linux users should find using Gnome Software much easier for locating the software they need to install. Another feature is the ability to adjust power profiles directly from the Status menu. Enable the Power Saver profile when you're running on battery or use the Balanced profile when you need more juice for games or resource-intensive applications.

All of the default Gnome apps (Calendar, Calls, Connections, Files, and Music) have received some much-needed tweaking (either in functionality or appearance), and the Gnome Settings tool now has a new Multitasking section, where you can configure

Hot Corners, Active Screen, and Workspaces.

Although Gnome 41 has yet to hit the repositories for the majority of Linux distributions that use the desktop, you can always use a rolling release distribution like Arch Linux. The first major distribution to ship with Gnome 41 will most likely be Fedora 35. For those that can't wait, you can always download the Gnome OS 41 ISO (https://os.gnome.org/download/41.beta/gnome_os_installer_41.beta.iso) or a Fedora Rawhide image (https://dl.fedoraproject.org/pub/fedora/linux/development/rawhide/Workstation/x86_64/iso/).

Black Lotus Labs Confirms Flaw in Windows Subsystem for Linux

Lumen Technologies' threat intelligence arm has verified that hackers can use Linux binary files as a loader designed to inject malicious files into a Windows process within WSL.

Four years ago, it was theorized that Linux binaries could be used as a means for hackers to gain access to Windows Subsystem for Linux. Up until recently, there has never been a single piece of evidence to prove that theory.

The time of speculation is over: Black Lotus has not only proved it to be true but has discovered that it's actually happening.

Lumen vice president, Mike Benjamin, says, "While the use of WSL is generally limited to power users, those users often have escalated privileges in an organization." Benjamin adds, "This creates blind spots as the industry continues to remove barriers between operating systems."

Black Lotus has identified a series of samples that were uploaded every two to three weeks, dating back to May 3, 2021 through August 22, 2021. The attacks were compiled with Python 3.9, using PyInstaller for the Debian OS v8.3.0-6. All of the samples, save one, contained private IP addresses. However, one sample was associated with a publicly routable IP address (185.63.90[.]137), which could indicate this new attack vector is still in development or just the first known instance of a hacker using this vulnerability to install malicious payloads into WSL.

Find out more about this new attack in the official Lumen blog, "No Longer Just Theory: Black Lotus Labs Uncovers Linux Executables Deployed as Stealth Windows Loaders" (<https://blog.lumen.com/no-longer-just-theory-black-lotus-labs-uncovers-linux-executables-deployed-as-stealth-windows-loaders/>).



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

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

CLOUDFEST

WE ARE **BACK**, EXCITED AND PUMPED.

 March 22 - 24, 2022 |  Europa-Park

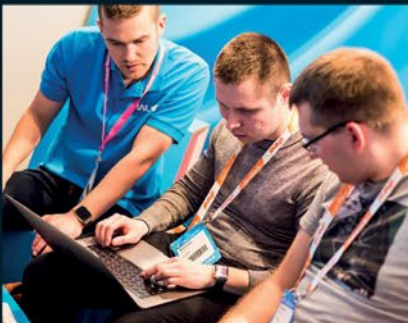
REGISTER NOW!

FREE CODE: **WE-LOVE-LINUX**



REG.CLOUDFEST.COM

THEMES FOR 2022



**THE INTELLIGENT
EDGE**



**OUR NEW
DIGITAL WORLD**



**THE SUSTAINABLE
CLOUD**

cloudfest.com

 /cloudfest

 @cloudfest

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**.

Renaming SMB

Steve French of Microsoft posted a bunch of patches to the SMB/CIFS filesystem, sparking an interesting little discussion. For starters, Linus Torvalds merged the patch set within a day but also had some remarks:

"You pretty much interchangeably use 'cifs' or 'smb3' for the filesystem, as shown once more by the commit messages here (but also the subject line).

"The filesystem directory is called 'cifs', and I've taken to use that in my 'Pull cifs updates' thing from you to just avoiding the confusion.

"And now we have ksmbd (yup, I just merged that pull request too), so we have a 'cifs client' and a 'smb server'. Aaarrrgh.

"I understand that some people may care about the name, may care about 'smb2 vs smb3', or whatever. But I have to admit finding it a bit annoying how the code and the directory layout uses these different terms pretty much randomly with no real apparent logic.

"Somehow the NFS people had no problem completely changing everything about their protocols and then still calling the end result 'nfs client' vs 'nfs server'.

"Oh well. I'm assuming it's not going to change, and it's not really a problem, I just wanted to mention my frustration about how clear as mud the naming is."

Steve had a very interesting and seemingly a very honest reply. He said, "I (and many at Microsoft and in Samba team etc.) also have a strong desire to stop using the word 'CIFS' as it has been associated with some very high profile attacks, and with the introduction of SMB2.1 support (which was far more secure) in 2009 no one should be using the very old CIFS dialect (aka 'SMB1' dialect). So if you are ok with renaming the client dir and module name – we can gradually stop using the word/name 'cifs' except for the parts of code which really are needed to access the (unfortunately hundreds of millions of) very old devices which require SMB1 ('CIFS')."

Steve added:

"Note that with the introduction of various security features in SMB3 (then even more security features in SMB3.1.1) it seems like it seemed confusing to users to tell them 'mount -t cifs ...' which was why I added support for 'mount -t smb3' (to cifs.ko) in the 4.18 kernel/ but I also would strongly like to stop using the word 'cifs' in module name going forward, even if it does cause a little bit of extra work for distros (most of which could be handled in the mount helper in any case)

"If no objections, we can start moving most things on the client to 'smb.ko' rather than 'cifs.ko' ..."

"Do you have any objections to me renaming the client's source directory to 'fs/smb3' (or fs/smb) and fs/smb3_common ...?"

Linus replied, "I'm ok with directory renames, git handles it all well enough that the pain should be fairly minimal. I'd ask for that to be done during a fairly calm cycle, though, when there isn't a lot pending, so that any rename conflicts will be minimized."

But he also added:

"I'm not entirely enamoured with the name 'smb' as a module (or directory) name, to put it lightly.

"Part of it is that it can mean 'system management bus' too, although in the kernel we happily universally (?) use 'smbus' for that.

"But a big part of it is exactly the history of random different names, which means that I'd like any new name to be more explicit than a TLA that has been mis-used for so long.

"So yes, we have 'fs/nfs/', but I'd rather _not_ have 'fs/smb/'.

"They may superficially look entirely equivalent – but one of them has had a consistent name that is unambiguous and has no horrible naming history. The other has not."

And in terms of what the new name might actually be, Linus went on:

"I'll throw out two suggestions, but they are just that: (a) 'smbfs' or (b) 'smb-client'.

"I think 'smbfs' has the nice property of making it clear that this is just the file-system part of the smb protocols – that otherwise cover a lot of other things too (at least historically printers, although I have no idea how true that is any more).

"And 'smb-client' as a name is in no way great, but at least it's not just a TLA, and from a naming standpoint it would match the 'smb-common' thing (although I guess you used an underscore, not a dash).

"Again – those are just two random suggestions, and I'm not married to either of them, I just really don't like just that 'smb' because of all the historical naming baggage.

"So if we rename, we should rename it to something new and slightly more specific than what we used to have."

Steve replied, "That should be easy enough (IIRC FreeBSD called their module 'smbfs'), but presumably wait until 5.16 or 5.17 to lessen merge conflicts etc." And he said he'd bounce the idea around with the people on the Samba team and others.

However, after a couple of weeks of behind-the-scenes reflection, Steve reported that there was an ancient `fs/smbfs` directory already in the Git repository, which he felt might cause enough confusion to not be worth it. So his new suggestion was to use `fs/smbfs-client` as the new directory name.

But the discussion ended there.

One of the fascinating things about this exchange is the forthrightness of Microsoft talking about public perception as a justification for making a change to the Linux kernel. Once upon a time, the conversation was what the kernel developers would do in response to Microsoft's inevitable attempts to utterly destroy the project. Now Microsoft is admitting a public-perception weakness and seeking solutions with the Linux developers in a collaborative way. And the Linux developers are taking it in stride and collaborating in turn.

Meanwhile, Linus's consideration of alternative names for the SMB filesystem is also interesting, in particular the distinction between `fs/smb` and `fs/nfs`, where they seem perfectly complementary, but one is acceptable for historical reasons while the other is not.

It's also generally interesting whenever Linus makes a point of saying that

whatever idea he's putting forward is not a "command." A lot of people below the level of his trusted lieutenants, just out of admiration and gratitude, would tend to consider Linus's preferences to be equivalent to being carved in stone. So he's had to (I would imagine) consciously remember to verbalize when he is only talking about something and not making a decision about it.

Testing Standards

Rae Moar from Google proposed a draft specification for Kernel Test Anything Protocol (KTAP) based on Test Anything Protocol (TAP), which was originally designed for the Perl interpreted language in the late 1980s and has since been extended for lots of other languages.

In fact, the extensions are why Rae wanted to set up a kernel-specific version of TAP. He felt there were too many conflicting elements to the TAP spec these days.

He based this new work on an earlier specification (also called KTAP) that had been written by Tim Bird and proposed to the Linux kernel mailing list in June 2020. Tim's original motivation had been to accommodate the various ways that `kselftest` had come to deviate from the original TAP spec. At the time, Tim's work received some significant interest.

Now, Rae felt that not just `kselftest` but also `KUnit` and other kernel testing frameworks needed an update to Tim's attempt – partly because Tim's concept included the idea of nested tests, and some newer tests had started to implement nested tests differently than Tim's conception.

Rae summarized the differences between the original TAP (now at version 14), versus his proposed KTAP specification.

First, he wanted to exclude YAML and JSON from diagnostic messages.

He also wanted to exclude `TODD` directives, which are really only used to alert whoever's running the tests that a particular test should be implemented at some point. Anyway, there's a `SKIP` directive that can be used for a similar purpose.

Rae also wanted to be able to nest an arbitrary number of tests – like Tim, he called them subtests.

Brendan Higgins was very enthusiastic about this new KTAP specification. He

said, “I would definitely like to see us moving forward with standardizing fully on the KTAP spec.”

Kees Cook was also very interested, saying, “thanks for looking at this again! Please understand that while I may be coming across as rather negative here, I would like to have a rational and documented specification for the kernel’s test output, too. My main objection here is that we already *_have_* a specification, and it’s already being parsed by machines, so making changes without strong justification is going to be met with resistance.”

Kees pointed out that `kseltest` was the biggest tester in the kernel and already had the standardized KTAP proposed by Tim the year before. So he said, “I would want buy-in from at least those responsible.” He also remarked:

“The fundamental purpose of the kernel’s TAP is to have many independent tests runnable and parseable, specifically without any embedded framework knowledge (or, even, any knowledge of TAP).”

“The tests run by kseltest come from 2 different TAP-producing harnesses (`kseltest.h` for C, `kseltest/runner.sh` for TAP-agnostic tests) as well as several hand-rolled instances in Shell, Python, and C. There used to be more, but I have been steadily fixing their syntax and merging separate implementations for a while now.”

But in terms of naming, Kees said he preferred to say “nested tests” rather than “subtests.” And he commented extensively on many of the other aspects of Rae’s spec.

Meanwhile Rae was taken aback to hear Kees say there was already a spec in use. That was news to him, and he asked for a link. He said:

“Wait, what?! An implementation is not a specification. I thought Tim’s attempt at standardizing the TAP that exists under `kseltest`, KUnit, and elsewhere was recognized as important or at least worthwhile.”

“The problem that was recognized, as I understand, was that there are multiple ‘interpretations’ of TAP floating around the kernel and that goes against the original point of trying to standardize around TAP.”

“I know KUnit’s usage is pretty minor in comparison to `kseltest`, but people do use it and there is no point in us, KUnit, purporting to use TAP and remain compatible with any particular version of it if it is incompatible with `kseltest`’s TAP.”

“Additionally, there is no way that we are going to be able to stay on a compatible implementation of TAP unless we specify what TAP is separate from the implementation.”

Tim also joined the discussion, not necessarily arguing in favor of his earlier draft but certainly with many technical comments on Rae’s new version.

Essentially all three engaged in collaborative discussion, each apparently hopeful that the final version would be good and useful. In fact, they each had so much to say that it seems they will very quickly light upon a common vision for this testing framework.

At one point David Gow from Google summed up the situation:

“I think many of the issues here stem from the original TAP spec having been insufficient for kernel stuff, and a bit of divergent evolution having occurred between `kseltest`, KUnit, and the dormant TAP 14 spec. This proposed spec does approach things more from the KUnit side, just because that’s what we’re more familiar with, but I agree that `kseltest` and

LAVA are the bigger fish in this pond. KUnit’s parser has also been a bit stricter in what it accepts, and the TAP producing code is shared between all of the KUnit tests, which makes prototyping changes a bit easier.”

“Fortunately, most of these differences seem pretty minor in the grand scheme of things, so I’m sure we can adapt this spec to fit what `kseltest` is doing better, while still leaving enough of the structure the KUnit tooling requires.”

At one point, Rae pronounced, “Thank you for all of your comments! I am glad to see some discussion on this email. First of all, my primary goal with this email is to advocate for a documented specification for the kernel’s test output. I presented my first email largely from the perspective of KUnit and thus, I am not surprised there are points of contention in the proposed specification.”

And the discussion continued, largely on technical lines, with everyone clearly aligned on the need to accommodate each other.

One interesting aspect of Linux development is how frequently the developers – or Linus Torvalds – will decide that an existing standard is broken or not useful and simply extend it to suit the kernel. They even forked the C library itself long ago and disagreed with the POSIX standard on the nature of threading. They feuded with the GNU C Compiler developers for years, refusing to upgrade from an increasingly out-of-date compiler version. And Linus resisted using version control for many years, as teams around the world struggled to accommodate his requirements, until finally he first used a proprietary system, to everyone’s chagrin, and then wrote his own system that ultimately replaced all the others. ■■■



BIG DATA & AI WORLD | BARC

8. – 9. December 2021 Messe Frankfurt
www.bigdataworldfrankfurt.de

Meet | Back | Better

#NurMitEuch

www.bigdataworldfrankfurt.de/linuxmagazine

Examining OpenBSD from the point of view of a Linux user

Free Cousin

Veteran Linux users and administrators are likely to have heard of the BSD family of operating systems. However, the BSDs remain a mystery to many in the Linux community. With the upcoming release of OpenBSD 7.0, it is time to throw some light on this little gem. *By Rubén Llorente*

Linux is the most popular Free and Open Source (FOSS) operating system, but it isn't the only alternative. Many FOSS operating systems are too niche to serve as a true alternative to Linux – you would not use Minix or FreeDOS for the same things you use Linux for. However, the BSD operating systems are a powerful family that is worth considering for tasks usually assigned to Linux.

It is said that Linux users have a poorer understanding of BSD than BSD users have of Linux. Given that both BSD and Linux are closely related to Unix, it is not surprising to learn that using one does not feel very different from using the other. They have similar userspace tools, you may install (mostly) the same utilities on each, and they are both built around free software. On the other hand, once you dive a little deeper into the systems, differences start surfacing.

OpenBSD 7.0 is slated for release in 2021, so it is a good time to take a look at OpenBSD as an example of what the BSD family has to offer.

A Brief History

The original Berkeley Software Distribution (BSD) was created as a set of add-ons to Version 6 Unix rather than as a whole operating system. BSD would eventually become an operating system of its own, but up to version 4.4, BSD was based on proprietary Unix code belonging to AT&T.

In 1991, Net/2 (Networking Tape/2), a BSD distribution with all the proprietary AT&T code stripped out, was released. Net/2 was based on BSD 4.4, although it wasn't a full operating system yet because it lacked some critical components. The BSDi corporation took code from Net/2 and created 386BSD. Sadly, AT&T sued BSDi over copyright and trademark infringement. Although the lawsuit was settled in favor of BSDi, the fact their legitimacy was in question slowed development of 386BSD and its descendants and diverted a lot of attention towards Linux. (Many believe the main reason Linux is more popular than any OS from the BSD family is because of AT&T's lawsuit.)

The NetBSD project was founded by four developers who were frustrated by with pace and philosophy of 386BSD devel-

opment. The developers forked the 386BSD code to launch a project that would emphasize the compact and correct code favored by the BSDs to this day. One of the NetBSD cofounders, Theo de Raadt, was later asked to resign from his position as a NetBSD leader due to conflicts with mailing list members. De Raadt launched OpenBSD in 1995 as a hostile fork of NetBSD.

OpenBSD Releases and Branches

OpenBSD [1] gets a new release every six months. Every release is supported with bug fixes and security patches for a whole year. Keep in mind that this applies only to the core system. The ports tree and package repositories receive fixes for the most recent release only.

Since the OpenBSD source code is managed by a CVS repository, the developers think of the release process in terms of branches. All the new, experimental, and exciting features are incorporated into the `-current` branch. When the time comes to make a new release, if `-current` is deemed stable enough, it is tagged as a `-release`, and OpenBSD `-stable` is branched out (Figure 1).

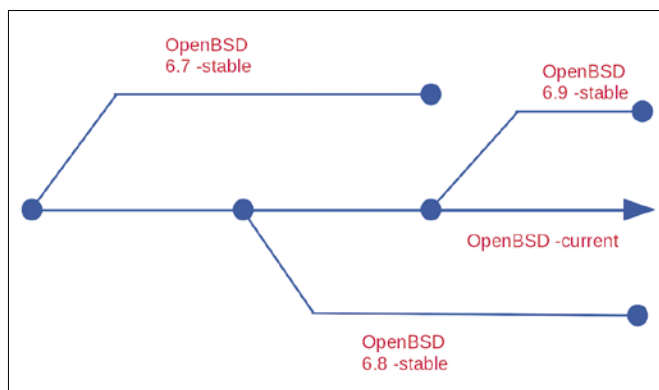


Figure 1: OpenBSD development happens in the `-current` branch. A `-release` is tagged every six months, giving birth to a `-stable` branch. The `-stable` branches are supported for a whole year, but most users of `-stable` prefer to upgrade to the most recent `-stable` immediately.



The `-stable` branch features the code of the corresponding release (for example, OpenBSD 6.9). This branch gets patches and security fixes as needed, but it is guaranteed not to receive updates that may cause breakage. Meanwhile, the developers keep working on `-current` in preparation for the next release.

OpenBSD is very conservative regarding changes. Therefore, upgrading from one release to another is rarely troublesome. Most changes are done under the hood rather than on areas regular users are likely to pay attention to. Therefore, running one release does not feel much different from running any other. This continuity is considered an advantage by many, because you can learn OpenBSD just once and be confident you won't have to relearn it every now and then.

Cathedral vs. Bazaar

A subtle difference between OpenBSD and any Linux distribution arises from their different development models. Eric Steven Raymond explained this difference in his book: *The Cathedral and the Bazaar* [2]. Linux distributions are put together following the Bazaar model: They pick many components from different vendors – such as the Linux kernel, Xorg, and Gnome – make packages out of them, and ship them as a unified software distribution. Hence, Linux distributions are built from a big sum of small add-ons that are mostly sourced from third parties.

OpenBSD (and the other BSDs, for that matter) follows the Cathedral model. The OpenBSD team develops the operating system code in its own house. The software that forms the core of the system is built from the ground up to fit OpenBSD's needs. When a new version of OpenBSD is released, it is shipped as a whole block instead of a sum of independent packages. This means the core software is tailored specifically to the operating system. It is true that many components are sourced from third parties (OpenBSD has Perl as a core component, for example), but most of the time, these are forked or heavily adapted versions rather than the original thing. An interesting side effect of the Cathedral model is that the documentation for the operating system is kept in

one place. If you want to find the documentation for a component in a Linux distribution, you need to track down the developer of the original component and find the documentation on the developer's site, whereas in the OpenBSD world, you head straight to the project's website or man pages [3].

Not many machines are useful if they run the core operating system only, so OpenBSD has a ports tree that allows you to install third-party software (see the box entitled "Packages and Ports"). What this means in practice is that every OpenBSD install will use the same core but will have a different set of packages installed on top of it by the user. The base and the ports

Packages and Ports

OpenBSD users add third-party software to their system using one of either the ports tree [4] or a package repository [5].

Linux users who have experience with Gentoo will already know how ports work. In a nutshell, a ports tree is a set of scripts that can be used to build packages automatically from source code. OpenBSD offers a ports tree, which is recommended for advanced users only. With it, an OpenBSD administrator can instruct the operating system to download, compile, package, and install a piece of software for which a port exists, alongside all its dependencies. An advantage of this approach is that each package can be customized and patched to the administrator's requirements.

The ports system is beautifully built, and OpenBSD ships software to deploy a packaging cluster in order to build massive amounts of packages in parallel...using multiple computers! In OpenBSD's style, the process is mighty secure: The code that downloads the source code that will be compiled runs at a privilege level that is different from the software that will make the packages, which in turn has yet another privilege level from the software that installs packages.

Since compiling software from ports can be time consuming and not fit for every user, OpenBSD offers a repository of pre-built packages. In fact, the goal of the ports tree *is* building this repository. Packages can be installed with the `pkg_add` utility, which supports dependency resolution and signature verification for downloads.

tree are considered to be at a different level and are administered by different tools. This means that if you are developing a third-party package, you can count on every OpenBSD install having the same set of core utilities and libraries – for a given release, at least. See the box entitled “What? No Copyleft?” for more on the differences between Linux and BSD.

Security by Correctness

OpenBSD has a reputation for being a secure operating system. In fact, its website boasts just two security holes in the default install “in a heck of a lot of time.” That said, the development policies are not focused on producing a *secure* operating system as much as they are aimed at creating a *correct* operating system. Security is just a side effect.

OpenBSD has therefore an aggressive policy towards removing obsolete code. For example, *libressl*, the SSL/TLS library OpenBSD uses instead of the popular *openssl*, lacks many encryption algorithms deemed outdated. OpenBSD also has a zero tolerance against binary blobs or any software that cannot be audited, which is never included by default, because it cannot be trusted and it is harder to fix if problems arise.

OpenBSD also uses its own *libc*, which is not compatible with the popular *glibc* to be found in the Linux world. This makes running certain programs designed for Linux a bit troublesome at times. For example, some C programs that use the custom `crypt()` function in *glibc* won’t work without patching.

What? No Copyleft?

Parts of this sidebar originally appeared in the July 2017 issue of Linux Magazine.

One important difference between Linux and the BSDs is the license. Although both Linux and the BSDs meet the definition of free software, the different licenses come with very different contexts for development.

Most Linux users associate free software with the “copyleft” protection embodied in the GNU Public License (GPL), which ensures that source code, including all modifications, must be shared with the community when the software is distributed. The BSD license does not require downstream sharing of the source code, and in fact, it allows a user who modifies the code to re-license it later with a non-free license. Linux proponents are often shocked to learn that free software components developed under a permissive license are sometimes taken *out* of open source and incorporated into proprietary programs, but the BSD community actually sees this permissiveness as a benefit.

The *BSD Advantages* page of the FreeBSD website cites an Apache project document to describe the advantages of permissive licenses. “This type of license is ideal for promoting the use of a reference body of code that implements a protocol for common service...many of us wanted to see HTTP survive and become a true multiparty standard, and we would not have minded in the slightest if Microsoft or Netscape chose to incorporate our HTTP engine or any other component of our code into their products, if it helped further the goal of keeping HTTP common.”

The GPL adds some legal complications that make it more complicated to integrate with other software. According to the FreeBSD project, “Developers tend to find the BSD license attractive as it keeps legal issues out of the way and lets them do whatever they want with the code. In contrast, those who expect others to

```
# /etc/pf.conf

# Our external interface
ext_if="em0"

# Don't filter localhost traffic
set skip lo0

# Block all incoming traffic we have not requested.
block in all

# Allow outgoing connections. The firewall is stateful, so
# responses to connections we have initiated are allowed in
pass out on $ext_if inet from ($ext_if) \
to any flags S/SA keep state
```

Figure 2: Instead of Linux’s netfilter, which is usually configured via iptables, OpenBSD uses the minimalist PF firewall.

On the other hand, OpenBSD’s *libc* has its own extra utilities, such as `crypt_newhash()` (for generating Blowfish hashes from passwords in a single step) or `arc4random()` (for generating random data when `/dev/urandom` is unavailable).

Security by Isolation

Linux has plenty of tricks up its sleeve in order to isolate processes from each other, just in case one of them was hacked by a nefarious actor. BSD has its own set of security tools (Figure 2). In the Linux world, tools such as SELinux, AppArmor, and seccomp are used to ensure that no process

evolve the code, or who do not expect to make a living from their work associated with the system (such as government employees), find the GPL attractive, because it forces code developed by others to be given to them and keeps their employer from retaining copyright and thus potentially ‘burying’ or orphaning the software. If you want to force your competitors to help you, the GPL is attractive.”

Through the years, code from the permissive BSD projects has made its way into many proprietary systems. MacOS and Solaris are both originally based on BSD code. Microsoft reportedly integrated BSD’s TCP/IP implementation into Windows. The copyleft viewpoint would regard these code appropriations as a loss for the community. Permissive proponents see it differently: by making it easy to adapt and integrate these components with other systems, they are spreading the benefits of free-software-based community development to a wider audience. Apple thus became invested in Unix, and Microsoft became a proponent of standards-based TCP/IP networking, rather than having to force the world to use its outdated proprietary protocols such as NetBEUI and its in-house, reverse-engineered version of the Novell NetWare protocols.

The GPL lends itself to large projects that keep the community working together on a single code base. Permissive licenses are better suited for smaller, collaborative projects that serve as a core or incubator for a larger ecosystem that might include proprietary implementations.

The copyleft protection of the GPL allowed Linux to become bigger and more popular than any of the permissively licensed BSD variants. However, BSD, with its permissive license and easy integration, played a role in spreading the gospel of Unix and standards-based programming to build the world in which Linux could flourish.

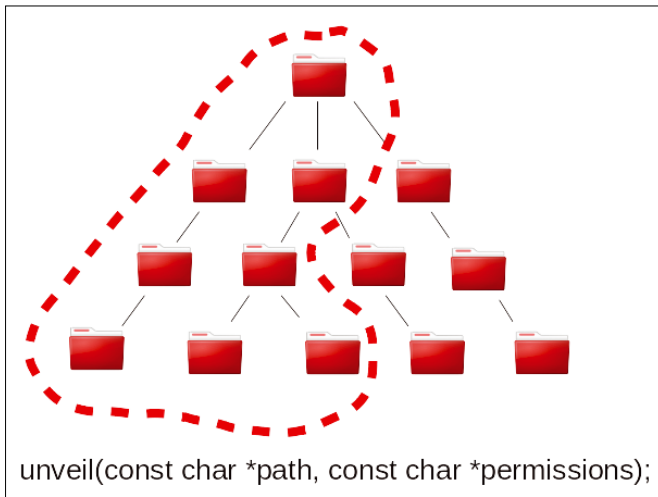


Figure 3: A process may use the `unveil()` system call to place itself in a filesystem sandbox. If the process is compromised later on, it won't be able to access files outside of the sandbox.

can access resources that it was never intended to access. This way, if a daemon such as the Apache web server is compromised, the attacker cannot easily access information managed by the `cupsd` printing service, for example.

For years, it was a common complaint against OpenBSD that it lacked a proper Mandatory Access Control (MAC) framework. Although there was some work done in this regard, it was abandoned because it was considered impractical and there was just not much interest in it. From the OpenBSD perspective, MAC frameworks such as SELinux are too unwieldy to use by regular administrators and are more likely to cause problems than to solve them.

The standard way in which processes used to be isolated in OpenBSD was by placing them in a `chroot()` jail and running them with reduced privileges. As security conscious administrators know, `chroot()` is not a great security feature, because a process running with root privileges inside of one can easily escape the `chroot` as per the POSIX standard. Although running the process with reduced privilege solves this problem for the most part, this approach had its drawbacks. Therefore, OpenBSD ended up creating two additional isolation techniques: `unveil()` and `pledge()`.

Both `unveil()` and `pledge()` are system calls that reduce the privileges of a process. The idea is that when a trusted process is started, it will tell the kernel that it plans to access only a certain set of resources. Attackers often use a compromised process to access resources the process was never intended to use. `Unveil()` tells the kernel which parts of the filesystem the program intends to use (Figure 3), and `pledge()` which set of system calls. The `unveil()` and `pledge()` system gives the kernel the information necessary to stop a rogue process from

accessing unauthorized resources and generate a warning for the logs to record.

The problem with this approach is that any given program must be patched to use these two system calls in order to be effective. However, the OpenBSD and ports maintainers seem to be doing a good job with patches for popular applications such as Mozilla Firefox. The advantage is huge: The user gets to run programs that put themselves in a virtual sandbox without the need of any configuration. The process is completely transparent to the user.

Userspace Goodies

OpenBSD is home to a number of outstanding userspace tools. OpenSSH [6] is indeed the most popular tool associated with the OpenBSD community. It has been ported to a number of other operating systems, and Linux administrators use it worldwide to access remote servers from their home. However, OpenBSD develops many other programs that are worth a closer look.

`httpd`, for example, is a compact web server that aims to be simple yet useful. It supports FastCGI (and slow CGI, for that matter), TLS, and request rewrites. It is easy to configure, and it is well documented. Its biggest drawback is that it is not compatible with `htaccess` files, which are often distributed with web applications such as Nextcloud and are intended to be used with the Apache web server instead. This limitation does not mean that OpenBSD's `httpd` cannot host Nextcloud. In fact, it is a good platform for this sort of application. It just means the administrator has to rewrite the `htaccess` rules in a format `httpd` understands. A webmaster who is not up to this task can always install Apache from the repository instead.

OpenSMTPD [7] is the OpenBSD alternative to SMTP servers, such as Postfix or Exim (Figure 4). It can be deployed with anti-spam filters, antivirus filters, and DKIM.

OpenBSD has its own X server, known as Xenocara, that doesn't require root privileges (Figure 5). `relayd` is another little gem. It is a daemon that can be used both as a reverse proxy, TLS accelerator, load balancer, and switch-over device for high availability applications.

```
# /etc/smtpd/smtpd.conf

# Configure TLS.
pki mail.operationalsecurity.es cert      "/etc/smtpd/tls/smtpd.crt"
pki mail.operationalsecurity.es key       "/etc/smtpd/tls/smtpd.key"

# Configure users and domains.
table credentials                        "/etc/smtpd/credentials"
table virtualdomains                    "/etc/smtpd/virtualdomains"
table virtualusers                      "/etc/smtpd/virtualusers"

# Listen on standard ports.
listen on eth0 tls mail.operationalsecurity.es
listen on eth0 port 465 smtps pki mail.operationalsecurity.es auth <credentials>
listen on eth0 port 587 tls-require pki mail.operationalsecurity.es auth <credentials>

# Incoming mail to virtual users is stored in a virtual mailbox.
# Everything else is relayed.
# Mail received from unauthenticated sources is not relayed by default
# as to prevent abuse.
action receive mbox virtual <virtualusers>
action send relay

match from any for domain <virtualdomains> action receive
match for any action send
```

Figure 4: OpenSMTPD lets you deploy an SMTPD (email) server. The configuration is very simple, especially if compared with popular alternatives.

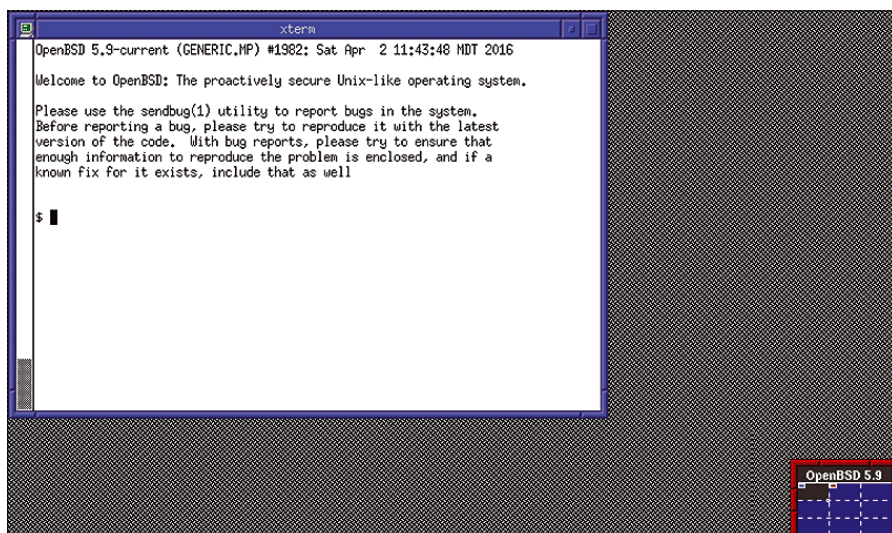


Figure 5: OpenBSD uses its own X server, which runs without root privileges. The default window managers are spartan, but you can install modern desktop environments such as KDE or Xfce from the repository.

What OpenBSD Lacks

Perhaps the most outstanding issue plaguing OpenBSD is raw performance. Since the developers are more interested in making the source code readable and easy to understand than in making it blazing fast, the operating system does not take advantage of multithreading as much as it could.

Another thing administrators might miss is a more versatile filesystem. OpenBSD uses the Unix File System (UFS), which lacks modern journaling capabilities. Thankfully, it uses soft dependencies as an alternative, which ensure that the filesystem will remain in a consistent state after a crash. However, UFS does not offer Copy-on-Write, so you can't take ZFS-like snapshots of a running system.

Support for NVidia graphic cards can only be described as atrocious. The developers attribute this to NVidia's refusal to collaborate and share GPU specifications in order to write code for their cards, and therefore their official recommendation is to not use NVidia hardware with OpenBSD.

Removable devices are supported up to a point, but don't expect a mass USB storage device to auto-mount as it would do with a modern desktop environment on Linux. OpenBSD just does not have an abstraction layer for mounting pen drives and assigning them permissions out of the box. You can always hack a solution for supporting auto-mount, but it is an involved process.

Conclusion

OpenBSD is a true heir to the Unix heritage, built upon Unix code (instead of being a clone written from scratch, like Linux). As one of the big four surviving BSD Operating Systems – the other three being NetBSD, FreeBSD, and DragonFlyBSD – it still brings great features to the table.

OpenBSD's so-so performance, combined with a lack of a Copy-on-Write filesystem, may preclude it from certain server

applications. Still, OpenBSD is a nice solution for deploying simple services, since the configuration files are compact and short, and the documentation is excellent. It is very popular for firewalls, routers, and ISP infrastructure.

The OpenBSD developers aim at delivering an operating system that has sane defaults and does not need the user to write complex configuration files in order to offer reasonable security. The `unveil()` and `pledge()` system calls are a good example: They are measures integrated in the programs the user runs, so the user needs not be aware of their existence to benefit from them.

Hardware support is not on par with Linux. If your computer has an NVidia card, you are better off using something else. The complications of hardware support have contributed to OpenBSD's repu-

tation for being a server operating system rather than a desktop operating system. On the other hand, popular desktop applications such as Thunderbird and LibreOffice are available from the ports tree and the repository, so OpenBSD is certainly ready for the office as long as you are aware of the hardware limitations.

OpenBSD is home to a whole lot of projects that are valuable on their own. Some of these tools can be used on Linux, such as OpenSSH; others are OpenBSD specific, such as OpenHTTPD. In any case, these projects will delight proponents of the KISS principle. OpenBSD is worth a try, if just for the userspace software it ships. ■■■

Info

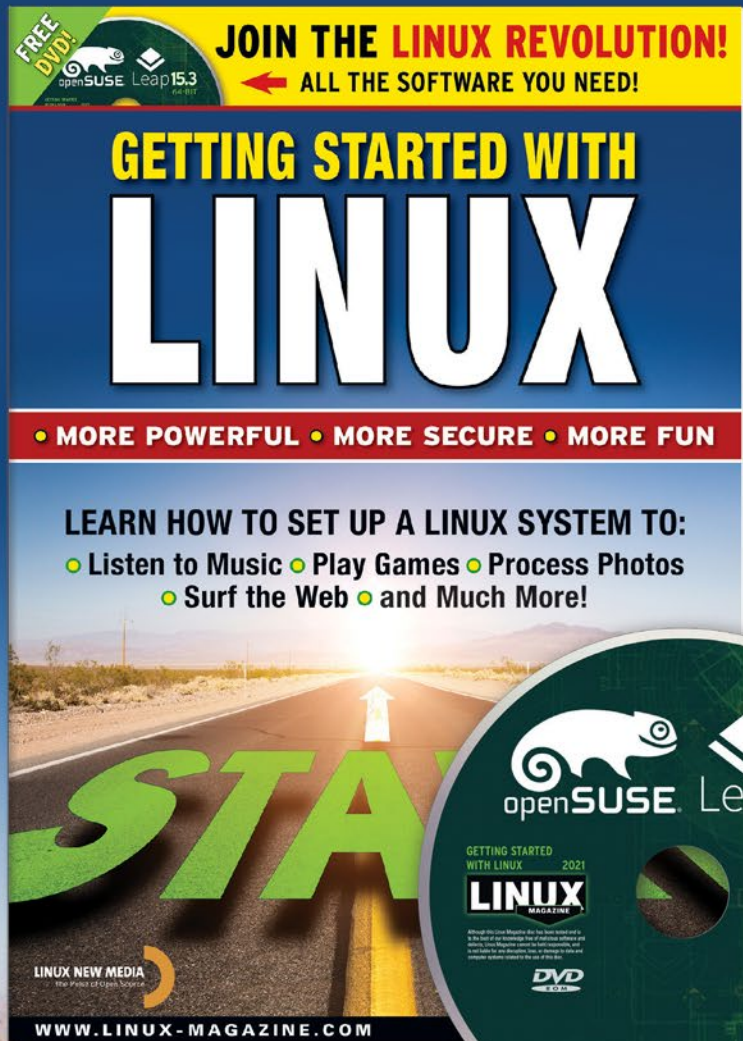
- [1] OpenBSD: <https://www.openbsd.org>
- [2] Raymond, Eric S. *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*, O'Reilly Media, 1999, <http://www.catb.org/~esr/writings/cathedral-bazaar/>
- [3] OpenBSD manual page server: <https://man.openbsd.org/>
- [4] OpenBSD Porter's Handbook: <https://www.openbsd.org/faq/ports/index.html>
- [5] OpenBSD package management: <https://www.openbsd.org/faq/faq15.html>
- [6] OpenSSH: <https://www.openssh.com/>
- [7] OpenSMTPD: <https://www.openSMTPD.org/>

Author

Rubén Llorente is a mechanical engineer whose job is to ensure that the security measures of the IT infrastructure of a small clinic are both law-compliant and safe. In addition, he is an OpenBSD enthusiast and a weapons collector.



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

A security-oriented OS

Isolationist

Andrew David Wong discusses the Qubes OS project's security-by-compartmentalization approach, including an endorsement from Edward Snowden. *By Bruce Byfield*

Qubes OS is one of the most original security solutions available. Using the Xen hypervisor, Qubes divides computing into security domains, or “qubes” (Figure 1) – including the root-like Dom0 – and incorporates them into the desktop menu (Figure 2). For other routine operations, such as copying to an external drive, Qubes OS creates a disposable qube that is discarded after the operation is complete (Figure 3). Recently, community manager Andrew David Wong explained more

about Qubes OS in response to *Linux Magazine's* questions.

Linux Magazine: Why is free software important to security?

Andrew David Wong: An operating system like Qubes OS aims to be the fundamental bedrock of people's digital lives. We strongly believe that any such security-critical software *must* be free and open source in order to be trustworthy. It is essential for any such software that the

project and code are transparent and that the developers' interests are aligned with those of their users. We all rely on Qubes for our own personal security in addition to our daily work on Qubes itself.

LM: What prompted the founding of Qubes OS? Personal reasons? Technical challenges? An incident?

ADW: Joanna Rutkowska and her team at Invisible Things Lab (ITL) initially rose to prominence through their *offensive* security research. Their work on low-level security and stealth malware exposed vulnerabilities and demonstrated attacks many had not thought possible. After

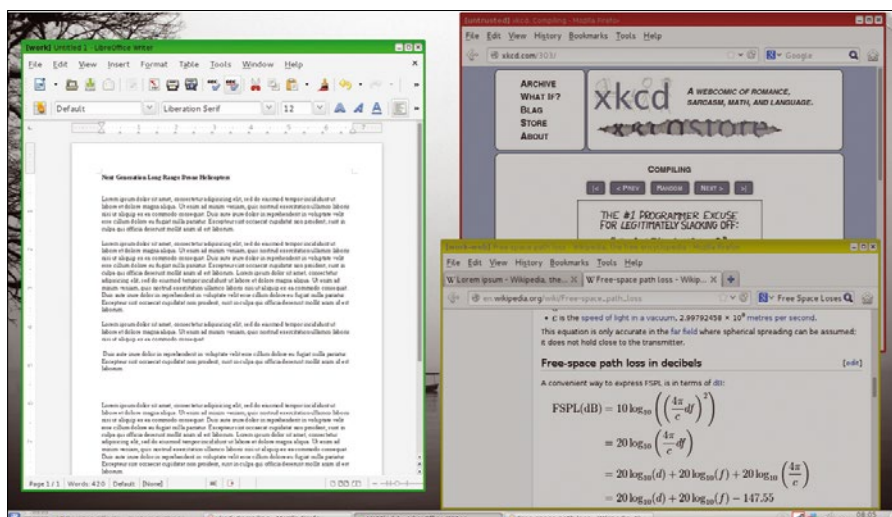


Figure 1: The Qubes OS desktop. Note the color coding for different security domains.

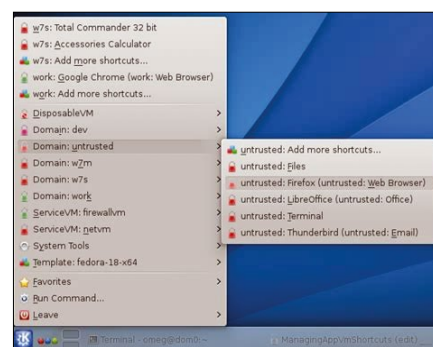


Figure 2: Qubes simplifies security by adding its security domains, or qubes, to the menu.

showing that the state of low-level security was even worse than commonly believed, their interests turned to figuring out how to improve it. Most experts will tell you that the people best suited to *build* a secure system are those who know how to *break* such systems. With their unparalleled expertise in virtualization security, the ITL team was uniquely suited to building a virtualization-based system that implements the principle of security by compartmentalization.

LM: Qubes OS is built on Fedora. How do the two interact?

ADW: Qubes uses Fedora as the operating system that runs in Dom0 and as one of many templates. We could substitute a different OS in Dom0, and Qubes would still be largely the same. Therefore, we don't think of Qubes as being based on Fedora. Rather, Fedora is just one among several distros Qubes uses and can use. Others include Debian, Whonix, Arch, Ubuntu, CentOS, Gentoo, and more.

In all such cases, we use the binary packages provided by the upstream distros. We don't rebuild everything from scratch. We simply add our own Qubes-specific packages on top of theirs. If one were to say that Qubes is based on anything else, it would be more accurate to say Qubes is Xen-based rather than Fedora-based. This is why we also don't think of Qubes as a Linux distro. If anything, it's more of a "Xen distro." But Qubes is much more than just Xen packaging. It has its own VM [virtual machine] management infrastructure with support

for templates, centralized updating, and so on. It also has a very unique GUI virtualization infrastructure. All of this forms a custom layer that abstracts from the underlying hypervisor.

We're working to make it so that Xen could be replaced by a different hypervisor, such as KVM, at which time it will no longer be accurate to call Qubes a Xen distro anymore, either. This is why we tend not to think of Qubes OS as a distro of anything else but rather as a meta-OS for running distros.

LM: How is Qubes organized and governed?

ADW: The Qubes OS Project is a global, decentralized, Internet-based collaboration. We have a largely flat, informal structure. Marek Marczykowski-Górecki is the project lead, with several others in charge of specific areas. We have no physical offices, and most work has been remote since the beginning.

LM: Why is Qubes described as "reasonably secure"?

ADW: Given the team's experience and expertise in showing how ostensibly-secure systems can be defeated, they understand better than most that there is no such thing as perfect security, especially in a *practical, usable* system.

Even the best programmers in the world, working under optimal conditions, cannot write complex code for real-world end users that's guaranteed to be 100 percent bug free. Most programmers are

working under far from optimal conditions under intense time and financial pressure. They're overworked, sleep deprived, and stressed out. Security is rarely a priority and typically little more than a distant afterthought. Day after day, these programmers around the world continue to pump out unfathomably large quantities of buggy, exploitable code, which we then run on our devices.

Meanwhile, security experts are in short supply, and there are not nearly enough to audit even a tiny fraction of the code being churned out, much less identify and fix the vulnerabilities it contains. The result is that new zero-day vulnerabilities are discovered and exploited at a staggering pace.

The core idea behind Qubes OS is that computer security is fundamentally broken. We can never hope to *prevent* compromise from occurring, so instead we assume that it will (or already has) and act accordingly. Qubes implements the principle of security by compartmentalization: It allows us to separate different parts of our digital lives in securely isolated compartments called qubes. This way, one qube being compromised doesn't affect the others. A single hack no longer threatens to take everything down in one fell swoop.

The Qubes philosophy is a fundamentally *practical* one. For example, some security experts regard modern web browsers as bloated, over-engineered, and too easy to exploit. Be that as it may, for regular desktop computer users, browsers are indispensable. They're how people access their money, get information, do their work, and communicate with others. Rather than eschew mainstream software like browsers, our approach is to acknowledge that such software is vulnerable and compartmentalize it accordingly. The browser in your untrusted web surfing qube will probably get compromised at some point, but that's okay, because it won't affect any of your other, more important qubes. In fact, we even have disposable qubes that automatically self-destruct when you're done using them so that a compromise from one session doesn't carry over to the next.

Qubes is free and open source software. We don't answer to shareholders or a board of directors. We don't answer to anyone except our users. This affords us the freedom and the luxury to be frank and honest with our users about the real limitations of com-

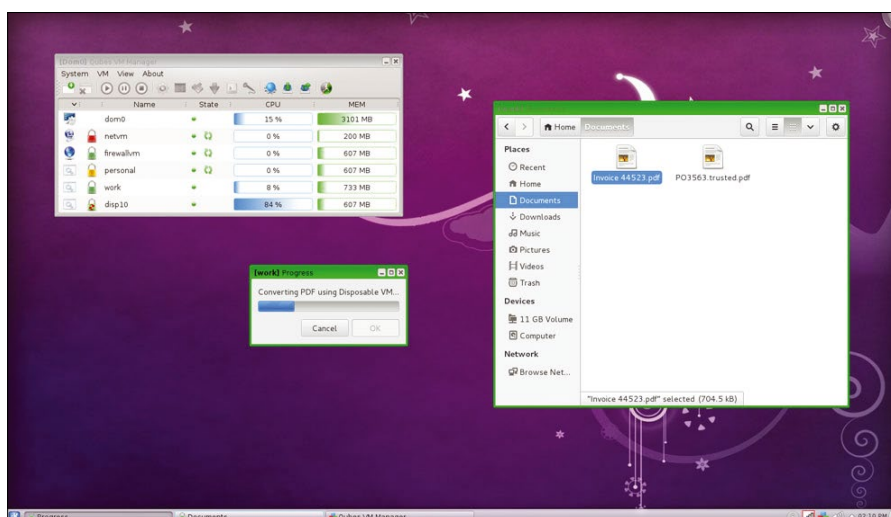


Figure 3: In Qubes, routine tasks such as converting a PDF file use temporary, disposable security domains..

puter security, including the limitations of Qubes OS itself. This ethos is nicely captured in the slogan “a reasonably secure operating system.” *Reasonable* security is the best any real-world operating system can hope to achieve. We’re just brutally honest about it from the get-go.

There’s also a tongue-in-cheek aspect to the slogan. Even before we had a slogan at all, Qubes OS had already earned a reputation as one of the most secure operating systems in existence and quite likely the most secure operating system available to anyone with an Internet connection. Many of our community members found the understatement of calling it only “reasonably” secure quite amusing.

LM: Your endorsements include one from Edward Snowden. Did his endorsement affect Qube’s popularity?

ADW: We’d like to think so! While we have only a rough estimate of the user-base, we do recall a noticeable bump in interest from his endorsement, and we’re certainly grateful for his continued support.

LM: Who is the target audience? Do you know of common deployments for Qubes?

ADW: Ultimately, our target audience is everyone who needs secure desktop computing. We are especially interested in providing a secure platform for those living and working in hostile environments, such as journalists and activists living under totalitarian regimes. Historically, many security researchers and power users have been drawn to Qubes, and we’re eager to continue supporting their needs, as well.

The Freedom of the Press Foundation uses Qubes OS in its SecureDrop project, as do the teams at Let’s Encrypt and Mullvad. We take great pride in the fact that these organizations rely on Qubes for their security while they work to provide secure technologies for their own users.

LM: What are the hardware challenges to the adoption of Qubes OS?

ADW: Historically, hardware has been one of the greatest challenges. Due to the high security standards we set for Qubes OS, specific hardware features are required. You can read more about that at the following links:

- <https://www.qubes-os.org/faq/#why-is-vt-xamd-v-important>
- <https://www.qubes-os.org/faq/#why-is-vt-dadm-viamd-iommu-important>
- <https://www.qubes-os.org/doc/system-requirements/>
- <https://www.qubes-os.org/doc/certified-hardware/>

We have addressed these hardware challenges through the Qubes certified hardware program: <https://www.qubes-os.org/doc/certified-hardware/>.

In addition, our community has recently put a lot of work into curating a list of computers that work well with Qubes: <https://forum.qubes-os.org/t/5560>.

Our community members also routinely test hardware to which they have access and contribute the results to our Hardware compatibility list (HCL): <https://www.qubes-os.org/hcl/>.

LM: How does security affect user convenience?

ADW: Qubes OS is inherently complex because it’s a compartmentalized system based on virtualization, which requires users to make conscious decisions about how to divide up their digital lives. It has a secure-by-design architecture. Secure designs always entail certain security-convenience trade-offs. Moreover, it’s based on a Linux environment that’s new to many users coming from Windows and Mac backgrounds. Most operating systems might have to contend with one or maybe two of these factors, but Qubes combines all three. So, it comes as no surprise that it can be a challenge for some users to learn and use.

However, we’re serious about making Qubes easier to use. Nina Alter, a user experience [UX] and design expert, has joined the team and has been hard at work on UX improvements throughout the system, some of which are funded by external grants.

LM: Why is Qubes OS not compatible with a virtual machine?

ADW: Some users have been able to install Qubes in a virtual machine, but it is neither recommended nor supported. Qubes should be installed on bare metal. After all, it uses its own bare-metal hypervisor!

While we understand that it would be easier to install Qubes in a virtual ma-

chine in order to try it out, one common alternative is to install Qubes on a fast removable drive, such as a USB 3.0 flash drive or an external SSD. This allows you to try Qubes on various systems without replacing the existing operating system on the internal drive.

LM: What future directions are planned?

ADW: There are two main goals we’re currently pursuing:

1. While many of Qubes’ security features are available via user-friendly graphical interfaces, many others still require using the command line and editing specific configuration files. In upcoming releases, we’ll focus on making these features more accessible to ordinary users, for example, by adding graphical interfaces for more parts of the system and providing ready-to-use configurations rather than requiring users to create their own.
2. Our security-by-compartmentalization approach uses virtual machines to isolate different workloads from one another, including those of our internal system services. This is similar to a microkernel architecture but with somewhat heavier workloads. We’re going to expand in this direction by allowing more types of workloads to be isolated. The latest example of this is isolating the entire graphics subsystem in a GUI qube. We’re also going to make these workloads lighter in order to allow for greater compartmentalization without requiring significantly more hardware resources, in particular, by leveraging the use of unikernels for certain workloads.

LM: Is there anything else you would like to add?

ADW: We are particularly grateful to our community for their steadfast support throughout the years. We’re pleased to see all the interesting things they’re making out of the building blocks we’ve provided, such as the KVM/Power port, Windows support, Qubes Video Companion, Wyng backup, and many more. Witnessing such a thriving ecosystem grow up around Qubes shows us how far we’ve come and how much the project has matured over the years, and we couldn’t have done it without our users, contributors, donors, and partners. Thank you! ■■■



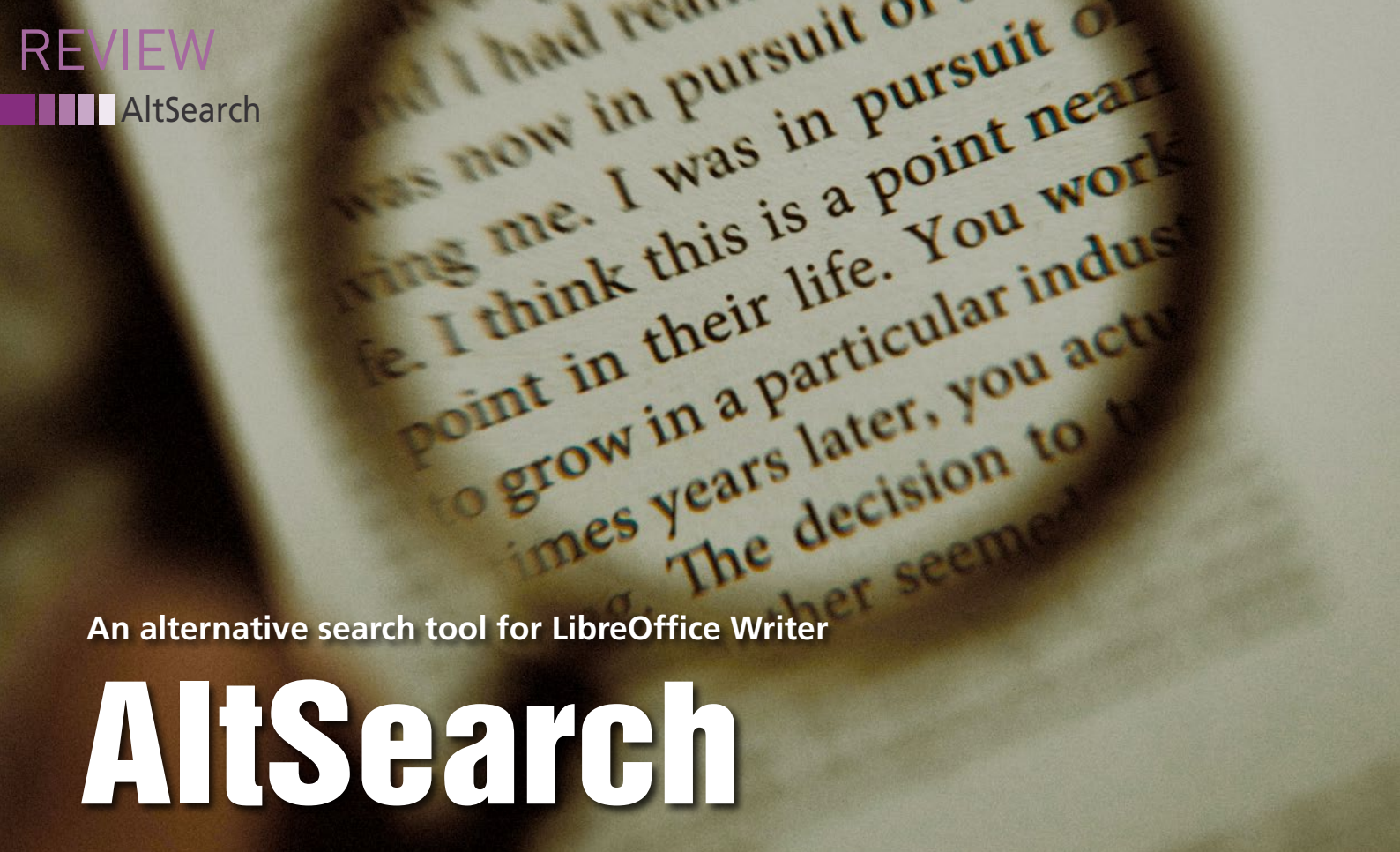
DATA CENTRE WORLD

8. – 9. December 2021 Messe Frankfurt
www.datacentreworld.de

Meet | Back | Better

#NurMitEuch

www.datacentreworld.de/linuxmagazine



An alternative search tool for LibreOffice Writer

AltSearch

AltSearch offers extended functionality to LibreOffice Write's default find and replace tools, making it ideal for editing and formatting longer documents. *By Bruce Byfield*

Few features in a word processor are less glamorous than a search tool. That is, until you do some intensive editing, especially if your revisions include reformatting. Then you will be thankful for a full-featured tool. In the case of LibreOffice Writer, the available tools are barely adequate, which is why I recommend the Alternative Find & Replace for Writer extension, also known as AltSearch [1].

Like all LibreOffice extensions, AltSearch is easily installed. Just download it from the LibreOffice extension site, and open *Tools | Extension Manager*. The next time you start Writer, AltSearch appears as a menu item, as well as an icon with green binoculars in the upper left corner of the toolbar.

You can understand the need for AltSearch by examining the default search tools in Writer. *Edit | Find* is a simple field similar to the ones found in many web browsers. It is suitable for finding words and phrases, but its options are strictly limited. You can search backward or forward from your present location in a document, find all, or match case – and that's all (Figure 1).

Edit | Find & Replace is more versatile, but it, too, is relatively limited. It can match case or search for whole words only. It has several additional features, such as support for regular expressions, similarity, and searches on a half-dozen paragraph styles, which make it more useful than Find – although you must be

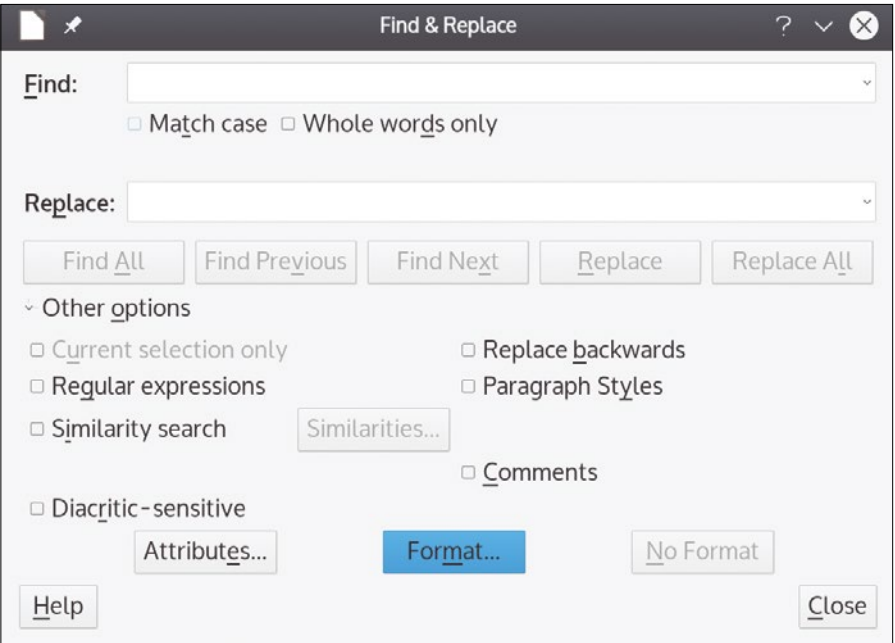


Figure 2: Find & Replace is an improvement over Find but is still fairly basic.

Photo by Chrissie Gianni on Unsplash

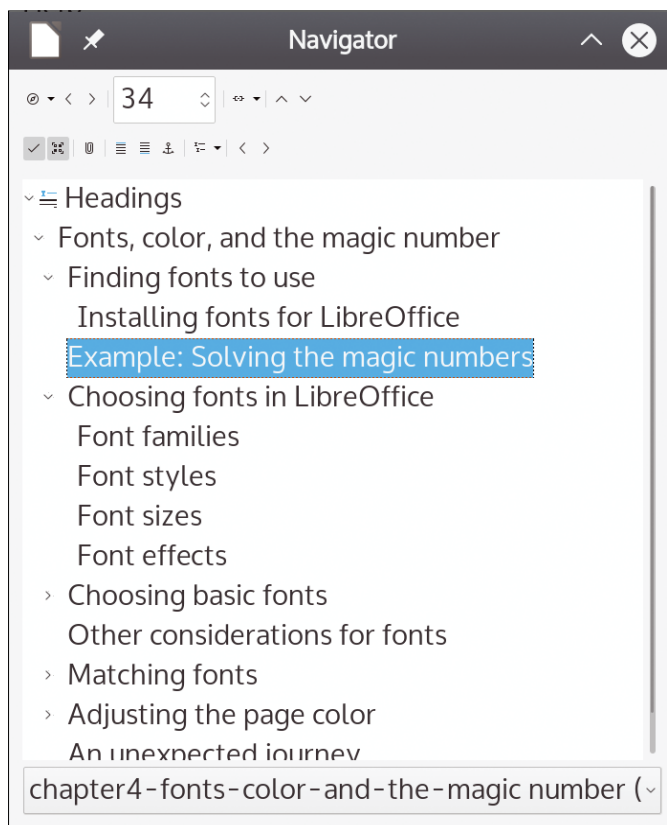


Figure 3: Navigator is useful for finding objects in a document but is separate from the other find tools.

careful not to replace the *Find* field's contents with nothing if you use the tool just to search (Figure 2).

Another default tool, Navigator, which you open from the sidebar, allows searching by any object in the document, most usefully by six paragraph style headings,

and even lets you reposition text below the headings without cutting and pasting (Figure 3) – but that is all. All three default tools are focused on finding and have limited power to edit.

By contrast, AltSearch offers more options (Figure 4). Admittedly, it suffers from some misspellings in the menus and is awkward to use. In particular, the menus are so long that it is best to position the window as close to the upper left corner of the screen as possible. However, most of the menu items in-

clude online help, and AltSearch's capabilities make the inconvenience worth enduring.

To start with, AltSearch's main window is better organized than Writer's default tools, with much of the complexity hidden. You can easily replace the default

tools with AltSearch without delving into its complexities. But if you do take a closer look at AltSearch, you will find several important improvements. For example, although Find supports standard regular expressions, it leaves the user to decide which ones to use. In comparison, AltSearch offers a selection window that includes custom expressions for formatting, such as custom hyphens and non-breaking dashes (Figure 5).

In addition, AltSearch offers a list of extended regular expressions (Figure 6) that, among other things, allows searches for most of the objects displayed in Navigator, including images, tables, frames, and cross references – a combination of features so basic I'm surprised that Writer did not do the same years ago. Other extended regular expressions include hyperlinks, text inside parentheses, and HTML tags.

Similarly, while the default tools only support basic paragraph styles, AltSearch's properties not only support character and list styles, but they can search on any of these styles used in the document rather than a basic few (Figure 7). Moreover, should these choices be overwhelming, you can run AltSearch on the basis of the text you select with the cursor. All these options greatly increase the chance of pinpointing a search item quickly, which is especially welcome in longer documents.

These options are not just useful in searches. They can also be useful in

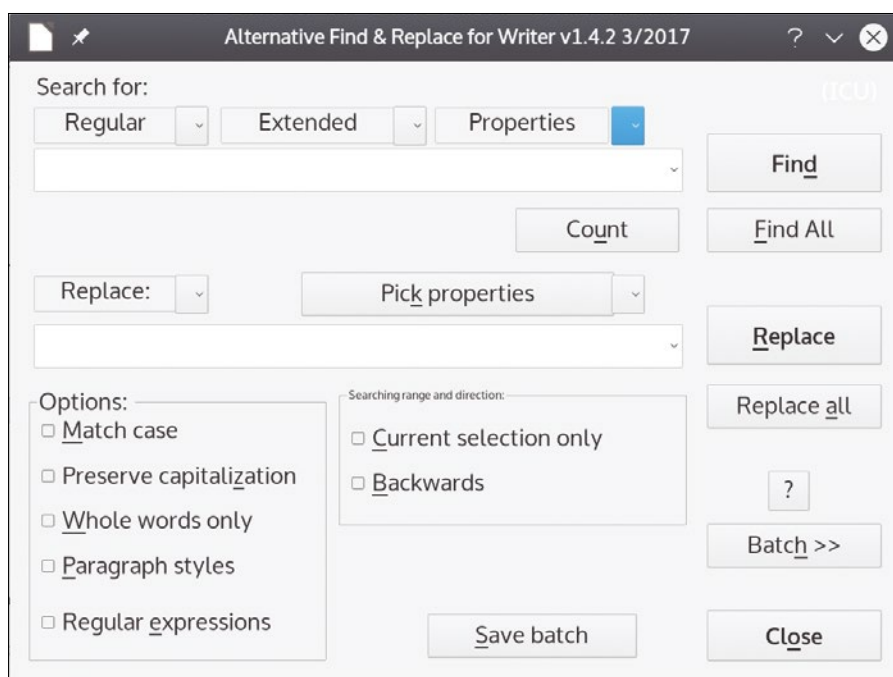


Figure 4: AltSearch is as much an editing tool as another search tool.

First char of a paragraph	^.
End of a paragraph	\$
Empty paragraph	^\$
Any text in one paragraph	.*
Any letter [:alpha:]{1,1}	\l
Any decimal digit [0-9]	\d
Beginning of a word	\<
End of a word	\>
Paragraph (ending mark)	\p
Series of empty paragraphs	^\$\p*
Tabulator	\t
Manual line break	\n
Manual column break	\c
Manual page break	\m
Any space [\xA0\x9\xA]	\s
Non-breaking space (\xA0)	\S
Custom hyphens	\x00A0
Non-breaking dash	\x2011
A inserted by decimal code	\#65
Dot	\.
Parentheses	()
Square brackets	[\]

Figure 5: AltSearch's list of useful regular expressions.


```

Expands found selection about one char to both sides
Append mark || for multiple replace (in one step)

Text between () (inside of one paragraph)
Text between [] (inside of one paragraph)
Text between {} (inside of one paragraph)

e-mail address
Internet, URL, www address
HTML tag
Opening HTML tag
Closing HTML tag

Notes (yellow bubbles) - searches substring in contents of notes
Text fields - searches substring in contents of fields
Text frame - searches substring in Names of frames
Table - searches substring in Names of tables
Picture - searches substring in Names of pictures
Footnote - text of anchor; add \\ for searches in content of footnotes
Endnote - text of anchor; add \\ for searches in content of Endnotes
Cross-ref. marker (text); \\ for search in Name; \\\ for search of empty text
Cross-reference (text); \\ for search in N. of marker; \\\ for s. of empty text
Bookmark - searches substring in text of Bookmarks; add \\ for searches in Names
    
```

Figure 6: AltSearch's list of extended expressions.

turning AltSearch into the desktop equivalent of the sed command. For example, if you save a Writer document that uses styles into text format, the conversion eliminates the indentation of a new paragraph, as well as any space between paragraphs. Adding spaces between paragraphs manually is tedious, but with AltSearch you can search for paragraph breaks with `/p` and replace them with a paragraph break and an empty new line (`/p/n`), preparing the text version of the document within seconds. If you want the extra space only in a list style, you can specify the style as well. In much the same way, you can easily remove non-breaking spaces or hyphens by replacing them with nothing and find a character string at the start (`/<`) or end of a word (`/>`).

In addition, AltSearch can perform a search or a replace that contains more

than one paragraph, or it can define a start and end point for a search. Both search and replace can be added from the clipboard, which is particularly useful with a clipboard that supports multiple entries. Styles can be replaced with another of the same kind, and objects can be renamed for easier navigation.

Perhaps AltSearch's most important single feature is the ability to do multiple searches via a batch file. The Batch manager opens with a list of some basic editing tasks, ranging from converting to a cleaner version of HTML than is available in Writer, converting date formats, or writing all of one kind of object to a new file (Figure 8). However, these are only samples of tasks that you can add to a batch file. From the Batch manager window, you can open the text editor of your

choice and construct files that perform as many functions as you choose.

When completed, a batch file can be executed and saved for later use. This function is especially useful in a longer project such as a book or thesis, where you might have many files that need to be edited in the same way because they will eventually be combined into a single work.

Conclusion

All this functionality takes a while to learn. Online help [2] exists, but it is more than a decade old and only marginally more detailed than the help built into the interface. For that matter, AltSearch itself seems not to have had a release for at least as long. So far, AltSearch continues to work, but a time may come when the latest LibreOffice release no longer supports it. If that ever happens, it would be a crippling blow to Writer's functionality, not just for search and replace but for serious editing as well. AltSearch is an extension that, like others before it, deserves to be a default part of the LibreOffice code. In fact, maybe someday, a version will be released for LibreOffice Calc as well. ■■■

Info

- [1] AltSearch:
<https://extensions.libreoffice.org/en/extensions/show/alternative-dialog-find-replace-for-writer>
- [2] Online help: http://macrojtb.hys.cz/HelpAltSearch_en.

```

Paragraph style
Character style
List style

Hyperlink
Hyperlink - substring in URL
Italic
Bold
Bold Italic
Font Name (manual changed name)
Font Size
Font Color
Font background (Highlighting)
Underline
Index (any)
Subscript (Auto)
Superscript (Auto)
Index defined by font size and escapement

Similar format of characters (based on cursor)
Same format of characters (based on cursor)
    
```

Figure 7: AltSearch lets you search on character and list styles.

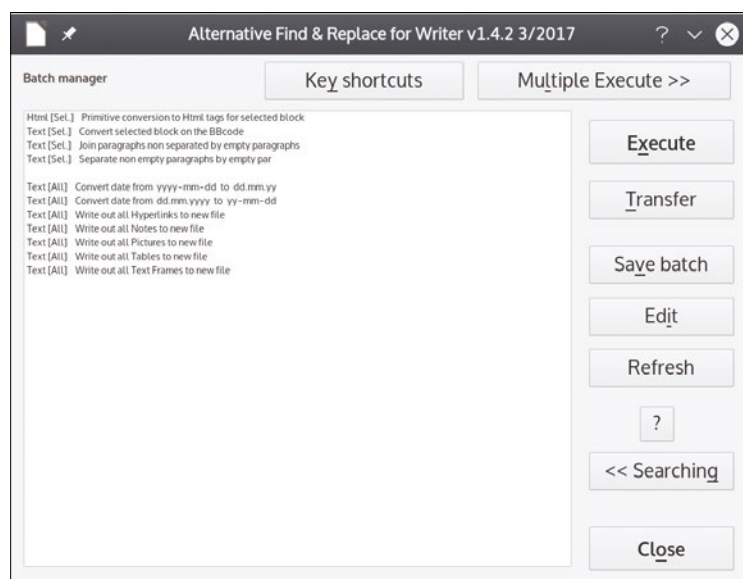
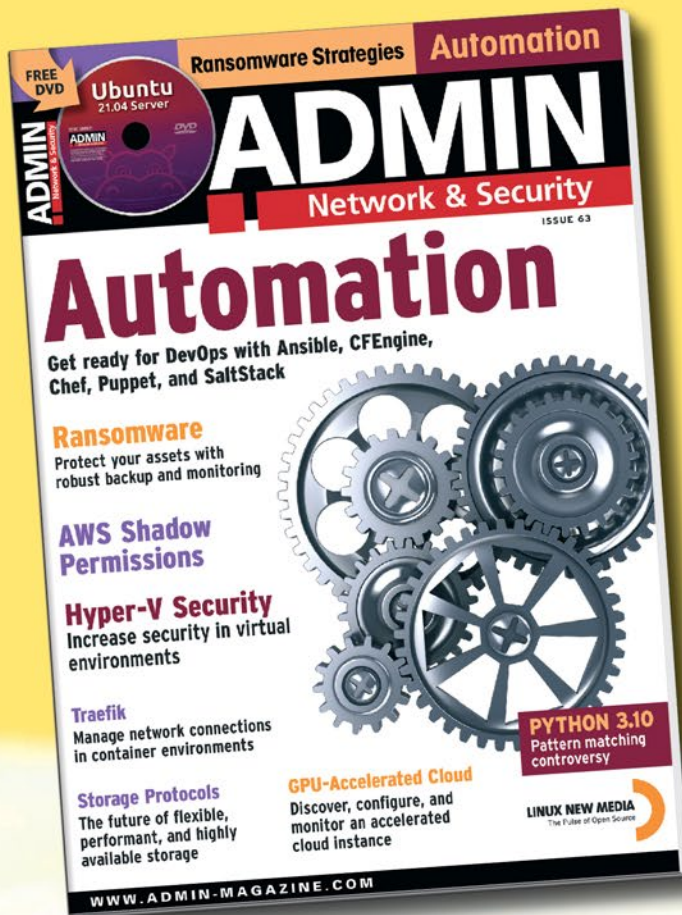


Figure 8: Users can write their own batch files to perform multiple functions all at once.

REAL SOLUTIONS *for* REAL NETWORKS



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

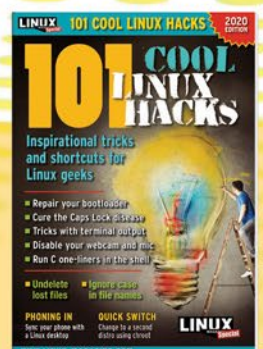
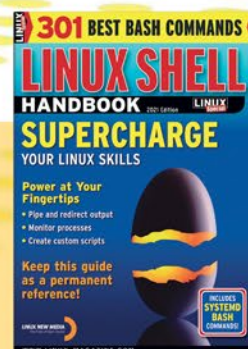
Improve your admin skills with practical articles on:

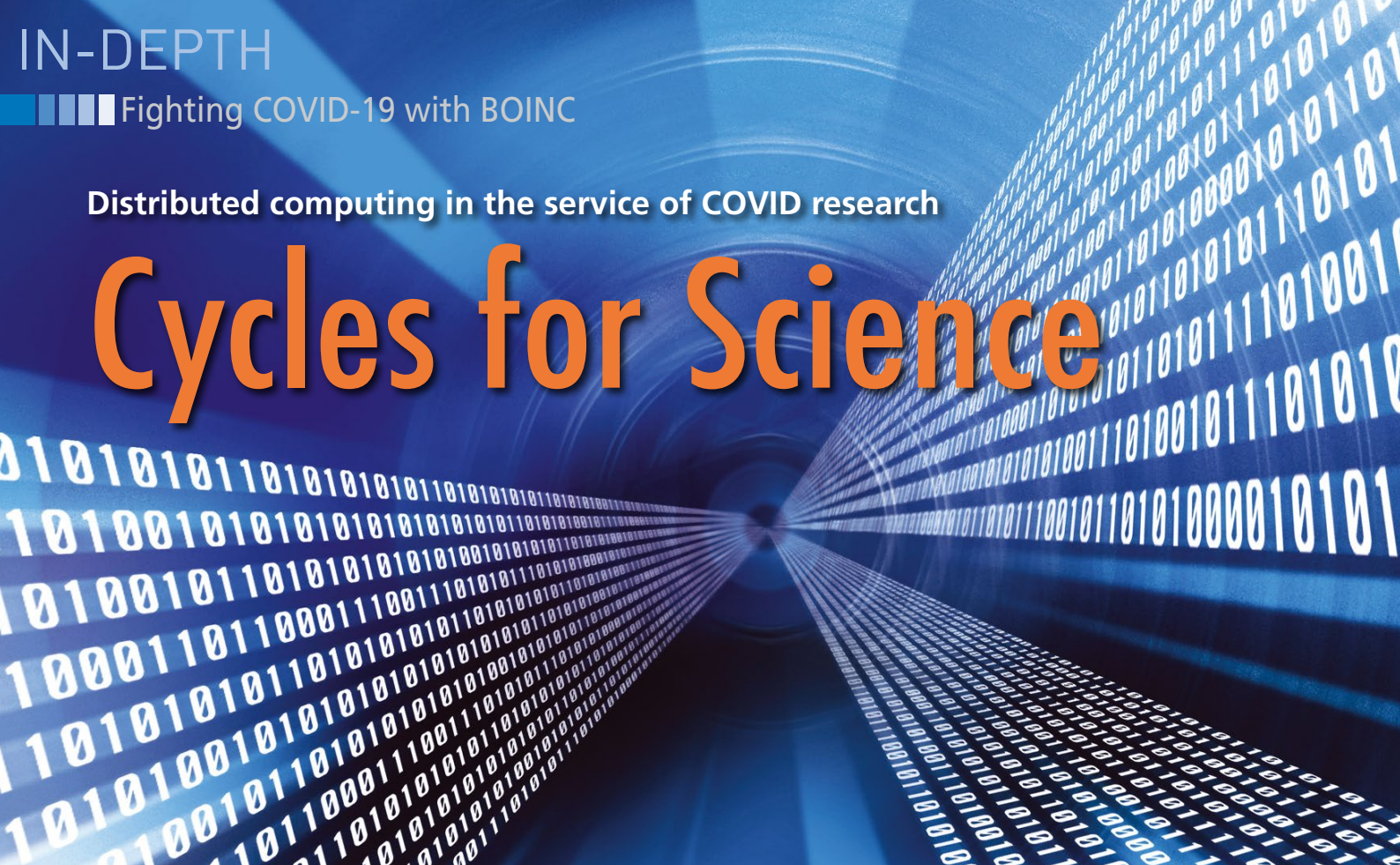
- Security
- Cloud computing
- DevOps
- HPC
- Storage and more!



SUBSCRIBE NOW!
shop.linuxnewmedia.com

Check out our full catalog:
shop.linuxnewmedia.com





Distributed computing in the service of COVID research

Cycles for Science

Linux and the BOINC distributed computing platform help researchers fight the COVID-19 virus. *By Erik Bärwaldt*

EVID-19 has had a dramatic impact on countries around the world. Researchers are continuing their work to develop vaccines and explore other ways of containing the virus. Many research projects require enormous computing capacities, but expensive supercomputers are not always available. Thanks to the concept of distributed computing, you can support research efforts by providing the computing power of your home PC.

The concept of using home computers to assist with research projects has been around for several years. The SETI project (Search for Extraterrestrial Intelligence) has offered home users a chance to process radio telescope data since 1999. IBM launched the World Community Grid [1], a central platform for managing volunteer distributed computing projects, in 2004. Since 2005, the World Community Grid has used BOINC [2], a software tool developed by the University of Berkeley for supporting distributed computing.

BOINC (Berkeley Open Infrastructure for Network Computing) separates the computational framework from the scien-

tific content, which makes it quite easy to adapt to a specific research project.

The software distributes independent work units to clients, which means you can integrate computers with different capabilities into the computations without slowing down the project.

BOINC has been available since 2005 as a free tool for Linux. The client does not just use the excess computing power of the CPU, but it also has a CUDA interface, which means it can access NVidia Graphics Processing Units (GPUs) [3].

Fighting Coronavirus

Scripps Research, based in California with subsidiaries in Florida [4], is one of the world's leading biomedical research institutes. More than 3,000 scientists work at the non-profit institution, spread over several institutes. The Forli Lab, which is part of Scripps Research, focuses on molecular biology [5].

As part of the "OpenPandemics – COVID-19" initiative, the laboratory is using the World Community Grid in elaborate simulations [6] in cooperation with IBM to find chemical components

to combat COVID-19. Distributed computing in the World Community Grid is responsible for screening individual components for later study.

Getting Started

To reroute the surplus computing power of your computer for COVID-19 research, first log in to the World Community Grid (WCG). To access the Grid, you just need to provide an email address and a password for the login.

You will receive an email for confirmation, which lets you verify your access to the WCG at the same time. Afterwards – with the help of the the World Community Grid website – you install the BOINC client and the matching manager on your computer.

To install the client, click on the *Download* link top right on the web page and then select one of the package management systems from the drop-down menu. DEB and RPM-based derivatives are available for selection. After making your choice, you are taken to a page with installation instructions. In most cases, you won't need to download the packages because the required applications are available from the software repositories of the popular distributions.

Run the commands listed in Listing 1 (for DEB-based systems) or Listing 2 (for

Listing 1: Installation on Debian/Ubuntu

```

sudo apt install boinc-client boinc-manager

sudo systemctl enable boinc-client

sudo systemctl start boinc-client

sudo chmod g+r /var/lib/boinc-client/gui_rpc_auth.cfg

sudo usermod -a -G boinc $USER

exec su $USER

boincmgr -d /var/lib/boinc-client

```

Listing 2: Installation on RPM-based Systems

```

sudo yum install boinc-client boinc-manager

sudo systemctl enable boinc-client

sudo systemctl start boinc-client

sudo chmod g+r /var/lib/boinc/gui_rpc_auth.cfg

sudo usermod -a -G boinc $USER

exec su $USER

boincmgr -d /var/lib/boinc

```

RPM-based derivatives). The BOINC manager pops up when you launch the application. The BOINC client itself has no graphical user interface but is used exclusively for communication with the server.

In the BOINC Manager, another window opens with a wizard. Now select the OpenPandemics project from the list of existing projects via the *World Community Grid* entry (Figure 1).

Log in to the BOINC manager with the combination of your email address and the password, which you registered up front. In the main window of the manager, you can now start the computations. The software fades to a progress bar and shows the necessary compute time, as well as the time already granted.

In the upper part of the window, you can view the active project. If you are participating in several computations, you can change the project by choosing an entry in the selection box. After running all the tasks, a green dot appears to the left of the project's name.

Pressing *Stop* at the bottom center of the window interrupts the computations; instead of the green dot a red one appears. *Continue* lets you restart the work (Figure 2).

Settings

The BOINC manager typically allocates the free resources of your system independently so that the computer does not suffer from the additional work. The application might possibly include your graphics card if it is an NVidia GPU.

If you have an NVidia card, you do not need to download the support for this interface typically required for CUDA-based applications from the vendor's website. Instead, BOINC detects the GPU automatically and integrates it into the computations. AMD graphics cards and Intel-based GPUs are not integrated by the software.

If computer load generated by other applications increases, the BOINC client stops its work. In this case, the manager remains active, but no further computation takes place. Once the system load has dropped back to below a given threshold value, the application automatically resumes its activity. When the activity restarts, the tool attempts to establish balanced load behavior (Figure 3).

In addition, the manager integrates various options for distributing the load which let you set thresholds yourself. You can reach the advanced settings via the menu items *View | Advanced View*. This is where you configure basic settings for the graphics processor via *Control* in the context menu.

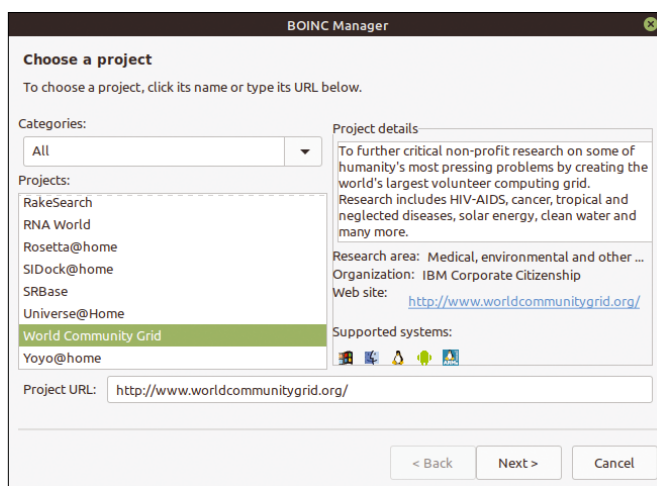


Figure 1: Select the World Community Grid entry – the OpenPandemics projects will then appear on a list of existing projects.

The menu *Options | Calculation settings* opens a very extensive dialog for fine-tuning the software. *Calculation running* is where you define the performance settings. This includes the threshold values for the system load.

The *Network* tab lets you configure the data transfer rate. This is where you specify, say, the upload and download rates. On the *Daily schedule* tab you can additionally define when the BOINC software is allowed to carry out its computation work. You can create a weekly schedule. The schedule is divided into two components: the times for calculations and the times for the data transfer.

For systems that operate in 24-hour mode, you can use this feature to shift the times for compute work and transfers to the night. After completing the desired settings, don't forget to press *Save* to store them.

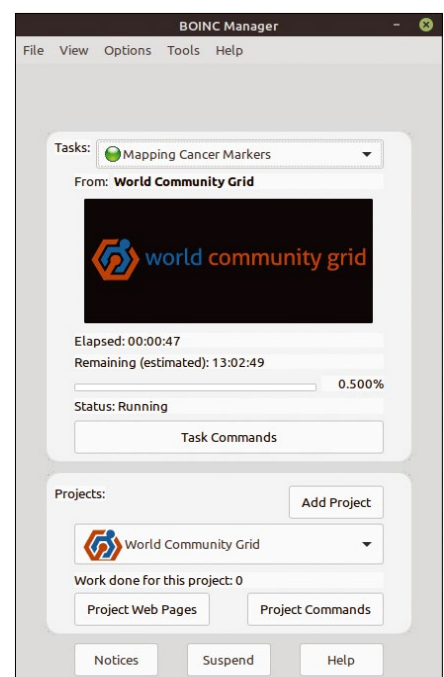


Figure 2: The BOINC Manager shows you all the information required for distributed computing.

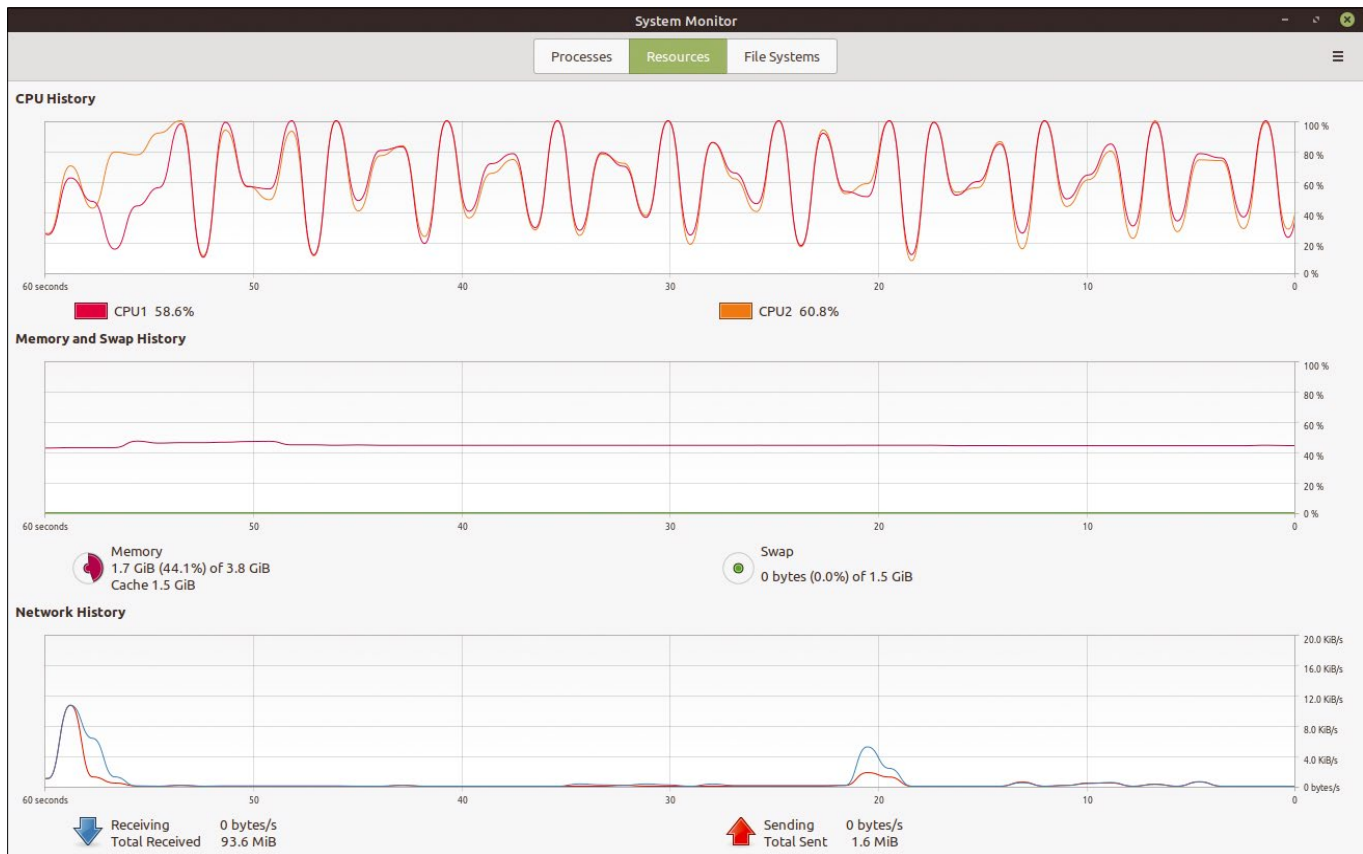


Figure 3: The BOINC Manager does not fully load even high-end computers.

Information

The active status displays are shown also in the extended view of the BOINC manager. This window is also structured by tabs. You will find a *News* tab with a very simple feed reader showing you the latest information on the individual projects, along with their keywords. Links take you to websites with the full details.

The *Projects* tab lists the projects you have selected in tabular form, and *Tasks* shows you the tasks within the active projects. Each of these tasks has a status indicator; using the progress bars, you can see the progress of the compute work in each task. The software updates this information in near real time (Figure 4).

You can access graphics settings in the *Transfer*, *Statistics*, and *Disk* tabs; some of these settings let you customize to suit individual performance criteria.

Conclusions

The World Community Grid is an innovative option for consigning idle computer capacities to the service of science. The BOINC software does not pose any problems for newcomers and is simple enough to virtually rule out any issues due to incorrect use.

The project launched by the Forli Lab and Scripps Research to explore therapeutic options against COVID-19 provides an excellent opportunity to put your surplus resources to good use. But do keep in mind that computing power costs electricity. ■■■

Info

- [1] World Community Grid: <https://www.worldcommunitygrid.org/discover.action>
- [2] BOINC software: <https://boinc.berkeley.edu/>
- [3] CUDA information: <https://developer.nvidia.com/cuda-zone>
- [4] Scripps Research: <https://www.scripps.edu>
- [5] The Forli Lab: <https://forlilab.org/>
- [6] OpenPandemics COVID-19 project: <https://www.worldcommunitygrid.org/research/opn1/overview.do>

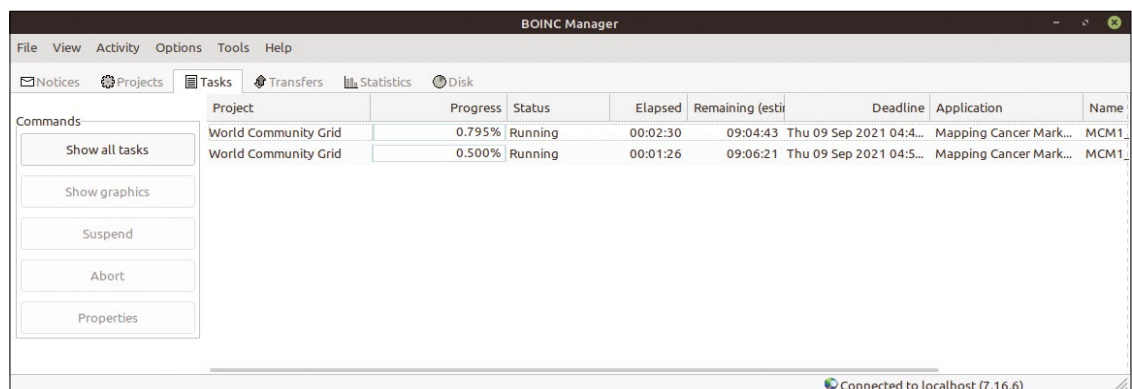
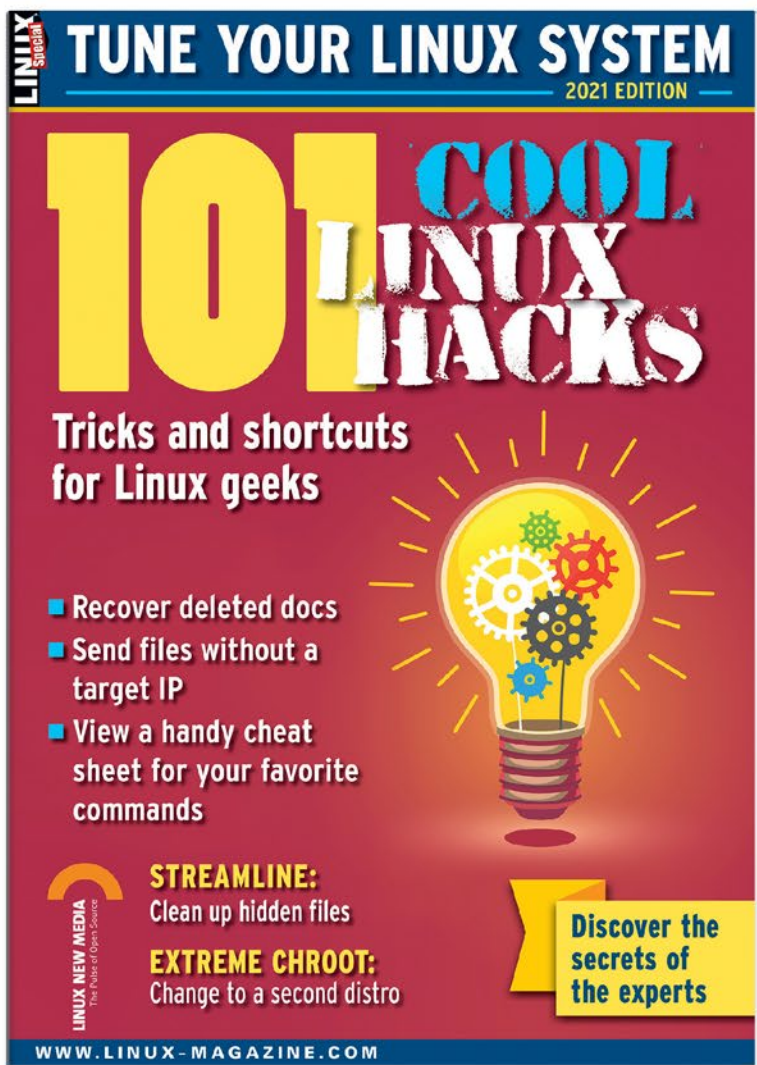


Figure 4: The detailed view shows you which tasks the software will complete, along with a time scale.

SHOP THE SHOP
shop.linuxnewmedia.com

GET PRODUCTIVE WITH 101 LINUX HACKS



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

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

ORDER ONLINE:
shop.linuxnewmedia.com/specials

The sys admin's daily grind: Virtual or physical?

Where Did It Go?

To write low-level scripts, as an admin, you need to know whether you are currently on a physical or a virtual machine. Charly finds out with a couple of clever hacks. *By Charly Kühnast*

Of the systems I work on, about 90 percent are virtualized and 10 percent are legacy hardware servers. For many jobs, this makes no difference, but when I write scripts that call or change hardware-related functions, I need this information.

If I have root privileges on the system and am also allowed to retroactively install software, the problem can be solved very quickly. I install either `Factor` [1] or `virt-what` [2]. `Factor` provides extensive information about the

system's hardware, much like `lshw`, and is actually overkill for answering the “virtual or not” question. Calling `factor virtual` returns the virtualization platform as the answer, such as `vmware` or `kvm`. The same result is returned by a call to `virt-what`. If I don't need the power of `Factor` elsewhere, I prefer the leaner `virt-what`.

If I have root privileges but am not allowed to install software (for example, because of restricted repositories), there is another possibility. The command

```
dmidecode -t system
```

gives me the desired information (Figure 1).

But what if I don't have root privileges on the system? There are solutions for this, too, even several of them. The first is the command:

```
dmesg | grep DMI
```

On a VMware guest, the output looks like the second line of Listing 1. If I run the same command on a physical server, I usually see some information about the server model at this point (line 4).

Another possibility is the command:

```
cat /proc/scsi/scsi
```

On a virtualized system, I would see output like that shown in lines 7 to 10 of Listing 1. The physical system, on the other hand, again responds with information about the hardware (starting in line 12).

There are quite a few other possibilities, but whenever it is technically possible, I use `virt-what`. It doesn't get any faster or easier than that. ■■■

```
# dmidecode 3.1
Getting SMBIOS data from sysfs.
SMBIOS 2.4 present.

Handle 0x0001, DMI type 1, 27 bytes
System Information
    Manufacturer: VMware, Inc.
    Product Name: VMware Virtual Platform
    Version: None
    Serial Number: VMware-42 13 c6 d2 21 11 fb fd-87 c5 3c 82 c6 b7 9b f2
    UUID: 4213C6D2-2111-FBFD-87C5-3C82C6B79BF2
    Wake-up Type: Power Switch
    SKU Number: Not Specified
    Family: Not Specified
```

Figure 1: Dmidecode lets me quickly discover whether I'm working on a virtual machine.

Listing 1: Virtual or Physical?

```
01 $ dmesg | grep DMI
02 [ 0.000000] DMI: VMware, Inc. VMware Virtual Platform/440BX Desktop
    Reference Platform, BIOS 6.00 05/28/2020
03 [...]
04 [ 0.000000] DMI: HP ProLiant DL320e Gen8, BIOS J05 12/10/2012
05
06 $ cat /proc/scsi/scsi
07 Attached devices:
08 Host: scsi2 Channel: 00 Id: 00 Lun: 00
09 Vendor: QEMU Model: QEMU HARDDISK Rev: 2.5+
10 Type: Direct-Access ANSI SCSI revision: 05
11 [...]
12 Attached devices:
13 Host: scsi2 Channel: 00 Id: 00 Lun: 00
14 Vendor: ATA Model: SanDisk SSD PLUS Rev: 00RL
15 Type: Direct-Access ANSI SCSI revision: 05
```

Info

[1] `Factor`:

<https://github.com/puppetlabs/facter>

[2] `virt-what`: <https://people.redhat.com/~rjones/virt-what/>

Author

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





2020
Archives
Available
Now!

CLEAR OFF YOUR BOOKSHELF WITH DIGITAL ARCHIVES

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

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

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



Probing for hardware information

Information Commands

A quick guide to 10 command-line tools to help you find hardware information. *By Bruce Byfield*

If you need hardware information, where do you turn? You might have the box and a Quick Start Guide, but chances are they're lost in the back of some closet. More detailed information

is probably available online but is not much use without the model number. The simplest source is your system itself, which has plenty of commands – including basic ones such as `grep` and `ls` – to

pry out the information you need from the niche where it resides. Many echo the basic `ls` command in their name, and many have two or more levels of verbosity, each one giving more detailed information than the last. But whatever the name or structure of the command, each unlocks an often untapped cache of information. Most of the time, you will want

```
bb@nanday:~$ uname -a
Linux nanday 4.19.0-17-amd64 #1 SMP Debian 4.19.194-3 (2021-07-18) x86_64 GNU/Linux
```

Figure 1: Using `uname` at the command line provides a brief system overview.

```
bb@nanday:~$ lspci -vvv
00:00.0 Host bridge: Advanced Micro Devices, Inc. [AMD/ATI] RD9x0/RX980 Host Bridge (rev 02)
Subsystem: Micro-Star International Co., Ltd. [MSI] RD9x0/RX980 Host Bridge
Control: I/O- Mem+ BusMaster- SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- Fast
B2B- DisINTx-
Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort+ >SERR-
<PERR- INTx-
NUMA node: 0
Capabilities: <access denied>

00:00.2 IOMMU: Advanced Micro Devices, Inc. [AMD/ATI] RD890S/RD990 I/O Memory Management Unit (I
OMMU)
Subsystem: Micro-Star International Co., Ltd. [MSI] RD890S/RD990 I/O Memory Management U
nit (IOMMU)
Control: I/O- Mem- BusMaster- SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- Fast
B2B- DisINTx+
Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR-
<PERR- INTx-
Interrupt: pin A routed to IRQ 27
NUMA node: 0
Capabilities: <access denied>
```

Figure 2: Shown here at the highest level, `lspci` has up to three levels of verbosity. Note that some information is not given when logged in as a non-privileged user.


```
nanday
description: Desktop Computer
product: MS-7693 (To be filled by O.E.M.)
vendor: MSI
version: 4.0
serial: To be filled by O.E.M.
width: 64 bits
capabilities: smbios-2.8 dmi-2.8 smp vsyscall32
configuration: boot=normal chassis=desktop family=To be filled by O.E.M. sku=To be filled by
O.E.M. uuid=00000000-0000-0000-0000-4CCC6A250851
*-core
description: Motherboard
product: 970 GAMING (MS-7693)
vendor: MSI
physical id: 0
version: 4.0
serial: To be filled by O.E.M.
slot: To be filled by O.E.M.
*-firmware
description: BIOS
vendor: American Megatrends Inc.
physical id: 0
version: V22.4
date: 12/21/2015
```

Figure 3: Run as root, `lshw` gives an exhaustive view of a system's hardware components.

to pipe the commands through `less` (adding `| less` at the end of the command), and in some cases you will need to log in as root to access the information.

uname

Using `uname` provides a high-level view of both the hardware and software on the system. With the `-a` option, it gives the following information in this order: kernel name, host name, kernel release, kernel version (such as *Debian 4.19.194-3 (2021-07-18)*), hardware type, hardware architecture, and operating system. Each of these pieces of information can be displayed by itself with a specific option of its own, but because the options often have no relation to the information, it is easier to simply remember `uname -a`. With no option, `uname` simply lists the operating system (Figure 1).

lspci

For a brief summary of all PCI buses, type the bare command. For more detail, add a level of verbosity from `-v` to `-vvv`. For hexadecimal dumps,

there are also four levels of detail, from `-x` to `-xxxx`.

A bus-specific view of information can be had with `-b` and a tree view with `-t`. Regular accounts can receive some information, but a full display requires root privileges (Figure 2).

lshw

While `lshw` can be run as an ordinary user, it only gives detailed information when run as root. When run with root privileges, the command's default output includes information on exact memory

configuration, firmware version, main-board configuration, CPU version and speed, cache configuration, and bus speed. If you are taking a screen shot, you might want to use the `--sanitize` option to conceal sensitive information such as IP addresses (Figure 3).

dmidecode

The man page for `dmidecode` warns that, because the command gives results quickly and securely, its output may be unreliable. Fortunately, so far as I can tell, that problem never seems to pop up,

```
root@nanday:~# dmidecode
# dmidecode 3.2
Getting SMBIOS data from sysfs.
SMBIOS 2.8 present.
55 structures occupying 2287 bytes.
Table at 0x000ECB10.

Handle 0x0000, DMI type 0, 24 bytes
BIOS Information
    Vendor: American Megatrends Inc.
    Version: V22.4
    Release Date: 12/21/2015
    Address: 0xF0000
    Runtime Size: 64 kB
    ROM Size: 8192 kB
    Characteristics:
        PCI is supported
        BIOS is upgradeable
        BIOS shadowing is allowed
        Boot from CD is supported
        Selectable boot is supported
        BIOS ROM is socketed
        EDD is supported
        5.25"/1.2 MB floppy services are supported (int 13h)
        3.5"/720 kB floppy services are supported (int 13h)
```

Figure 4: Information provided by `dmidecode` includes a summary of the BIOS.

```

root@nanday:~# lsusb
Bus 007 Device 009: ID 03f0:3217 HP, Inc LaserJet 3050
Bus 007 Device 008: ID 056a:033b Wacom Co., Ltd
Bus 007 Device 007: ID 08bb:29b0 Texas Instruments PCM2900B Audio CODEC
Bus 007 Device 006: ID 0409:005a NEC Corp. HighSpeed Hub
Bus 007 Device 005: ID feed:0000
Bus 007 Device 004: ID 1209:2301 Generic Keyboardio Model 01
Bus 007 Device 003: ID 0a5c:21e8 Broadcom Corp. BCM20702A0 Bluetooth 4.0
Bus 007 Device 002: ID 0409:005a NEC Corp. HighSpeed Hub
Bus 007 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 011 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 010 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 004 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 009 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 008 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 006 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 005 Device 002: ID 2109:3431 VIA Labs, Inc. Hub
Bus 005 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 003 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 002 Device 002: ID 2109:3431 VIA Labs, Inc. Hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub

```

Figure 5: Use `lsusb` to list USB devices.

and `dmidecode` provides a useful summary of a system's hardware, including the serial numbers and BIOS revision. You can search for a specific piece of hardware using `--string (-s) KEYWORD`, `--type (-t) DEVICE-TYPE`, or `--handler (-h) DEVICE-ID`. The man page contains a

list of the types of devices listed and a table of useful keywords (Figure 4).

lsusb

Modern computers depend heavily on USB devices, so it is only natural that a command was written to dig out information

```

root@nanday:~# lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:             Little Endian
Address sizes:          48 bits physical, 48 bits virtual
CPU(s):                 8
On-line CPU(s) list:    0-7
Thread(s) per core:     2
Core(s) per socket:     4
Socket(s):              1
NUMA node(s):           1
Vendor ID:              AuthenticAMD
CPU family:             21
Model:                  2
Model name:             AMD FX(tm)-8350 Eight-Core Processor

```

Figure 6: For a detailed view of the CPU, use `lscpu`.

```

root@nanday:~# ls SCSI
[0:0:0:0] disk ATA Samsung SSD 850 2B6Q /dev/sda
[2:0:0:0] disk ATA WDC WD2003FZEX-0 1A01 /dev/sdb
[4:0:0:0] cd/dvd HL-DT-ST DVDROM GH24NSC0 LI00 /dev/sr0

```

Figure 7: As `ls SCSI` shows, SCSI devices continue to be a standard part of modern computer systems.

```

[ 0.802664] pci 0000:00:12.0: quirk_usb_early_handoff+0x0/0x6d0 took 84534 usecs
[ 0.890655] pci 0000:00:13.0: quirk_usb_early_handoff+0x0/0x6d0 took 85829 usecs
[ 0.978652] pci 0000:00:14.5: quirk_usb_early_handoff+0x0/0x6d0 took 85829 usecs
[ 1.066649] pci 0000:00:16.0: quirk_usb_early_handoff+0x0/0x6d0 took 85926 usecs
[ 2.385296] ACPI: bus type USB registered
[ 2.385324] usbcore: registered new interface driver usbfs
[ 2.385336] usbcore: registered new interface driver hub
[ 2.385357] usbcore: registered new device driver usb
[ 2.399275] ehci_hcd: USB 2.0 'Enhanced' Host Controller (EHCI) Driver
[ 2.403579] ohci_hcd: USB 1.1 'Open' Host Controller (OHCI) Driver
[ 2.403961] ehci-pci 0000:00:12.2: new USB bus registered, assigned bus number 1

```

Figure 8: Read messages in the kernel ring buffer with `dmesg`. Combined with `grep`, it can show useful information.

on them. By itself, `lsusb` will give a complete list of USB devices. However, with `--[[bus]:][devnum]` you can display in decimal only the devices on a specified bus and/or devnum. Similarly, `-d [vendor]:[product]` displays in hexadecimal only the devices with the specified vendor and model ID. As root, you can also use `-D DEVICE-FILE`. As any account, you can use `-t` to display information as a tree (Figure 5).

lscpu

This command displays information gathered from `sysfs`, `/proc/cpuinfo`, and architecture-specific libraries. You can run `lscpu` with `--extended (-e)` to display more detailed information (just as `--verbose` is used in some other hardware commands), plus `--parse (-p)` to optimize the formatting of the output. In addition, for even more detailed output, `--out-all` can be added. Depending on the options, `lscpu` displays as many as 13 columns of information, among them CPU, CORE, SOCKET, and ADDRESS. For virtual machines, it can also display CONFIGURED, meaning whether the virtual machine is using the CPU, and P0-LARIZATION, which indicates whether the virtual machine can switch the CPU dispatching mode between horizontal or vertical. Users can specify only online CPUs with `--online (-b)`, only off-line CPUs with `--offline (-c)` and `--extended combined`, or both with `--all (-a)` (Figure 6).

ls SCSI

Many people imagine that SCSI drives are obsolete, but, in fact, both hard drives and solid state drives, as well as DVD drives, continue to use SCSI, although in a highly modified standard. For help, `ls SCSI` has an info file, but not a man file. Adding `--list (-L)`, `--long (-l)`, and `--verbose (-v)` to the command


```
MemTotal:      32889716 kB
MemFree:       23502664 kB
MemAvailable:  28873732 kB
Buffers:       524580 kB
Cached:        5002724 kB
SwapCached:    0 kB
Active:        4936468 kB
Inactive:      3516952 kB
Active(anon):  2927472 kB
Inactive(anon): 149988 kB
Active(file):  2008996 kB
Inactive(file): 3366964 kB
Unevictable:   84 kB
Mlocked:       84 kB
SwapTotal:     67001340 kB
SwapFree:      67001340 kB
Dirty:         3088 kB
Writeback:     0 kB
AnonPages:     2882772 kB
Mapped:        2074928 kB
Shmem:         151364 kB
Slab:          606416 kB
SReclaimable:  459056 kB
SUnreclaim:    147360 kB
```

Figure 9: Search the `/proc` pseudo filesystem for information when troubleshooting.

all give more information, while `--class` (`-c`) is the equivalent of:

```
cat /proc/scsi/scsi
```

If you want the names of devices, add `--generic` (`-g`) (Figure 7).

dmesg

`dmesg` reads the kernel ring buffer where messages about a system's startup messages are stored, including information about the initialization of device drivers or kernel modules. Although the results can often be hit or miss, `dmesg` is sometimes an ideal place to start troubleshooting. With the bare

command, you can scroll through startup messages, but in most cases, it is more efficient to search instead. For instance, to find messages about USB devices, enter:

```
dmesg | grep -i usb | less
```

in which the `-i` option ignores letter cases. Note that `dmesg` must be run as root. As an alternative to `dmesg`, you can read the file `/var/log/dmesg` in a text editor (Figure 8).

/proc

The pseudo filesystem `/proc` contains information from the kernel. It contains one subdirectory for each process. The names of the subdirectories are usually self-explanatory, such as `cpu`, `cwd` (current working directory), `environ`, and so on. You can view detailed information using a text viewer such as `cat` or `less`, or `sysctl` to read the contents, but be careful not to edit in case you crash the system (Figure 9).

/sys

The kernel creates the pseudo filesystem `/sys` to give access to devices and information about them. Be careful when working with `/sys` because attempting to edit it can seriously damage its system. It is safe, though, to use `ls`, `grep`, or any command that simply displays information in these subdirectories:

- `/sys/block`: Information about block devices
- `/sys/bus`: Subdirectories for each bus
- `/sys/class`: All devices classes

- `/sys/devices`: The hierarchy of all devices on the system
- `/sys/firmware`: Firmware objects and their attributes
- `/sys/modules`: Subdirectories for the kernel module

You can also use `sysctl` to view `/sys`, but be careful – it can also edit the contents, which can result in disaster unless you know exactly what you are doing (Figure 10).

Choosing a Tool

This article gives only an overview of the available tools. Much more can be said about most of these tools, but the point is to provide a quick guide. The purposes of these tools often overlap, so if one fails to give the information you need, another might. Whichever one you choose, don't be surprised if you need a search engine to fully understand much of the available information – you're delving deeply into the workings of your system's hardware. ■■■

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/>.

```
root@nanday:/sys/devices/cpu# cd /sys/devices/cpu/events
root@nanday:/sys/devices/cpu/events# ls
branch-instructions  cache-misses        cpu-cycles           stalled-cycles-backend
branch-misses        cache-references     instructions          stalled-cycles-frontend
```

Figure 10: The `/sys` pseudo filesystem contains dozens of directories and files that you can search for information.

The background of the entire page is a photograph of a modern library. It features multiple levels connected by a wide staircase with a glass railing. Bookshelves filled with books are visible on the upper levels. A person is sitting on a bench in the lower level. The overall atmosphere is bright and clean.

Hone your skills with special editions!

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

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

Available in print or digital format

Check out the full library!

shop.linuxnewmedia.com

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

GETTING STARTED WITH LINUX

• MORE POWERFUL • MORE SECURE • MORE

LEARN HOW TO SET UP A LINUX SYSTEM

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



LINUX NEW MEDIA
WWW.LINUX-M

LINUX 301 BEST BASH COMMANDS

LINUX SHELL

2021 Edition **LINUX Special**

HANDBOOK

SUPERCARGE

YOUR LINUX SKILLS

Power at Your Fingertips

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


TUNE YOUR LINUX SYSTEM

2021 EDITION

101 COOL LINUX HACKS

Tricks and shortcuts for Linux geeks

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



STREAMLINE: Clean up hidden files
EXTREME CHROOT: Change to a second distro

Discover the secrets of the experts

LINUX NEW MEDIA
WWW.LINUX-MAGAZINE.COM

FREE DVD!  **Become a LibreOffice Expert!**
2020 Edition

LibreOffice

Dive deep into the world's greatest free office suite

Write Your Own LO Macros
Save time and automate common tasks

Digital Signatures
Lock down your private documents

Edit and Save

RASPBERRY PI GEEK
THE COMPLETE ARCHIVE
2,000 pages of maker projects and more!

FREE DVD \$39.90 VALUE!



LINUX NEW MEDIA
The Power of Open Source

MakerSpace

HANDS-ON PROJECTS FOR MAKERS

Dive Into

- Raspberry Pi
- Arduino
- Retro gaming

Discover FPGA
Learn to program hardware for retro computing



LINUX NEW MEDIA



An innovative, immutable filesystem

TOTALLY RELAXED

With its immutable filesystem, rlxos prevents a broken system while simultaneously allowing changes via OverlayFS. By Ferdinand Thommes

When it comes to the Linux distro scene, variety is unrivaled. Willing Linux users can choose from hundreds of distributions. Although this delights die-hard distro hoppers, others might find it too much to handle. While many distributions differ only in the minor details, the strategy behind rlxos [1], an immutable filesystem, is definitely not that of an off-the-shelf Linux distro.

Pronounced Relax OS, rlxos is one of the modern Linux derivatives with a progressive strategy (others include Fedora Silverblue, for example). The developers of Red Hat, Fedora, Endless OS, systemd, and the Gnome desktop see these strategies as the future of distributions, but this has not yet been universally accepted in the various communities.

Unbreakable System

In general, these strategies envisage immunizing the filesystem against vulnerability through updates by always replacing the complete image during updates. In tech speak, these systems are dubbed

immutable (i.e., unchangeable). If something goes wrong during the update, the user can roll back to the previous image upon restarting GRUB. In addition, such distributions often prefer new package management systems, such as Flatpak or AppImage, over packages in the classic DEB and RPM formats or those maintained by the respective distributions.

Joining the ranks of immutable distributions, rlxos's basic strategy is by no means new: Changes to the system end up in a layer above the read-only root filesystem and thus cannot be changed. I'll discuss in a moment where this principle has been used

in the past, but I'll first look at how this modern approach affects the still very young rlxos.

To follow this article, first download the 1.3GB image of the current rlxos 2107 from the website [1]. There is only a 64-bit image, the name of which indicates that Gnome is used as the desktop.

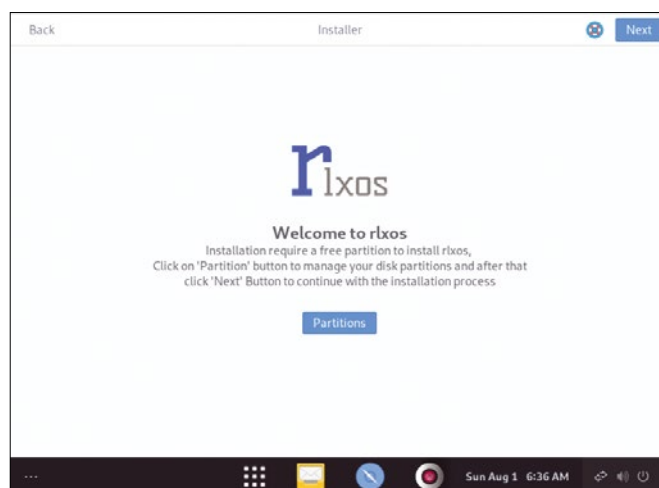


Figure 1: After starting rlxos, you are taken directly to a wizard that guides you through the installation in just a couple of steps.

Photo by Christine Donaldson on Unsplash

OverlayFS

OverlayFS allows a (typically writable) directory tree to be overlaid on top of another, read-only directory tree that usually contains the root filesystem. All changes are made to the upper, writable level. This type of mechanism has been around since Live CDs from the early days of Knoppix, but there are a variety of other applications. Besides OverlayFS, there are other union filesystems, such as the original UnionFS, aufs, or overlay2 used by Docker, with OverlayFS being the most powerful variant. In some respects, device mappers, ZFS, and Btrfs are also part of this genre, even if their use seems rather marginal in this context because they are used entirely for storage purposes.

Launched last year as Releax OS, rlxos is not based on any other distribution; it was created from scratch and uses Wayland as the default session type.

Relax!

Not designed as a Live distribution, rlxos boots directly into a graphical installer (Figure 1). Presumably, there is no Live version because rlxos in its installed form, with an overlay system over the immutable root filesystem, uses the same mechanism that Live media use to save changes to the system. Therefore, rlxos is always effectively going to be a Live system with persistence. For an explanation of how this overlay mechanism works, see the “OverlayFS” box.

The installer first prompts you to choose a partition. After pressing the *Partition* button, GParted opens. You can then create a new partition if required. Pressing *Next* then takes you to the partition selection and, in the next step, to boot device selection. Then, rlxos is ready for installation.

At this point, make sure you don't blink, or you will miss the system setup. In my case, the installation took all of 12 seconds on a machine with a fast Ryzen 7 CPU by AMD, which can be explained by the fact that, due to the principle involved, this is not a real installation. The installer (in simple terms) just creates the filesystem and writes the system image compressed by SquashFS to it, before proceeding to start the image. As a result, rlxos does

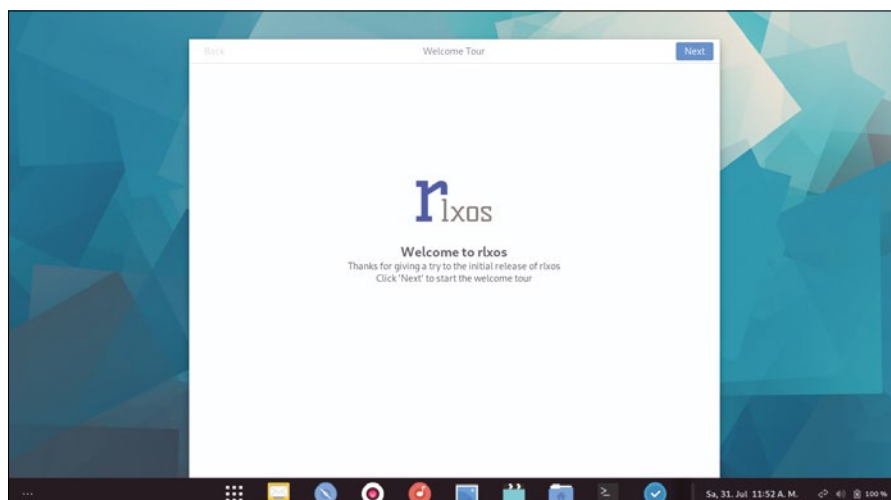


Figure 2: The welcome tour, which is almost obligatory nowadays, guides you through the final steps of the installation. Afterwards, take a look at the useful documentation, which introduces you to using rlxos.

not consume more than the 1.3GB taken up by the image itself.

Welcome

After the obligatory reboot, you will see the familiar Gnome welcome screen (Figure 2), which completes the installation in a few steps with information about the keyboard layout and time zone, as well as creating a user account. In my lab, I also included a Nextcloud instance, which I was later able to open in the file manager without any problems (Figure 3).

After the process completes, you will find yourself in a customized Gnome environment that has a taskbar using

Gnome's Dash to Panel extension at the bottom (Figure 4). Qogir is used as the theme and icon set.

Next, you can take another welcome tour, which points out that rlxos natively supports the AppImage package format. In addition to AppImages, rlxos can handle Flatpaks and Snaps, although it does not preinstall the required frameworks.

I found the welcome tour a tad too superficial for a distribution that deviates from the norm and would definitely recommend reading the documentation [2], as well as visiting GitHub [3], where the project is maintained. The rlxos blog [4]

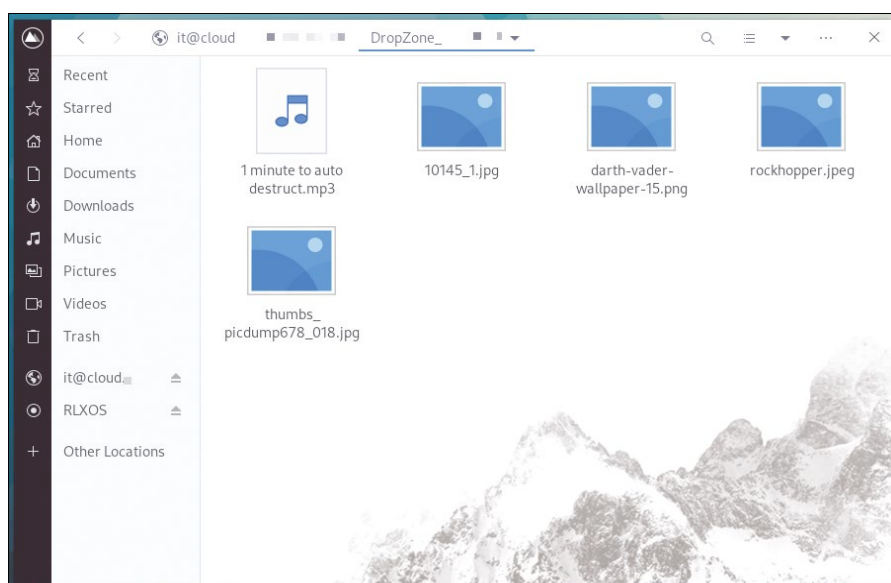


Figure 3: During the welcome tour, you have the opportunity to integrate online services. This worked smoothly in with my Nextcloud instance.

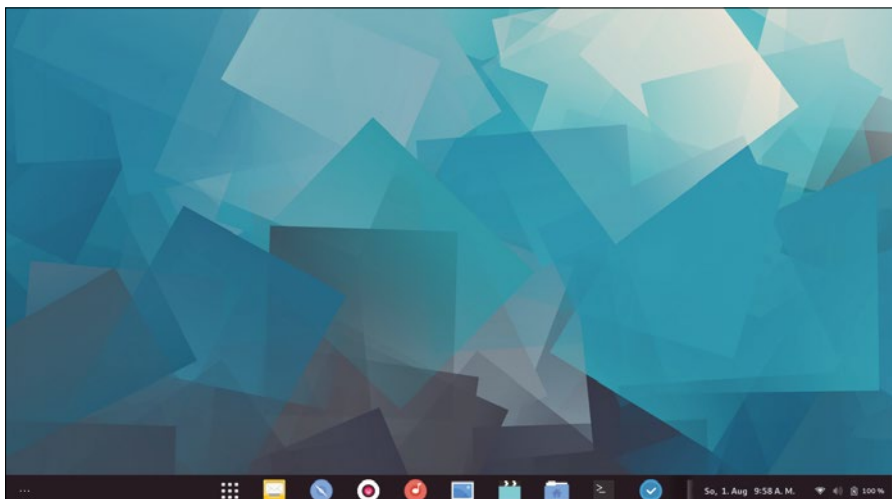


Figure 4: The desktop looks modern and tidy. At the bottom you can see the Gnome Dash to Panel extension.

gives you further insights. For example, it introduces the built-in package manager, PKGUPD [5], and explains how to mount rlxos directly from a previously installed Linux distro without the installer [6].

App Grid

On the desktop, which rlxos finally lets you access after completing the tour through the special features, the upper edge remains empty at first. It is only after pressing the button on the far left

of the taskbar that the familiar app grid view appears (Figure 5). If you only need the search mask, just click on the three dots bottom left. The bar contains six entries by default. If you have more apps open, it expands to the right and left when you mouse over the first and last visible entries. The system tray is on the far right. Further predefined add-ons can be enabled in the settings in the Extensions menu.

The offered selection of preinstalled applications is typical of Gnome, al-

though not as extensive as, say, Fedora or Ubuntu. For example, the Gnome Software application manager is missing, but it wouldn't really make much sense here. You will find at least one app for each of the usual application scenarios, such as browsing, music, videos, or image editing. For example, rlxos includes the Web browser, the Evolution mail and calendar app, the Totem video player, and the Shotwell photo manager. In addition, there are the system apps familiar from Gnome. As an init system, rlxos relies on systemd, and it uses kernel 5.8.

App Control

Rlxos has no graphical package manager. The `appctl` front end is available for the terminal and behaves much like `Apt` or `DNF`, accessing a repository with around 800 applications. However, you can also include third-party repositories [7], as revealed in the documentation. For example, you could choose the Nano editor, which is missing from the rlxos repo, instead of the preinstalled Vim. Table 1 lists the most important commands for operating `appctl`. You can access help for the commands by typing `appctl` in the terminal.

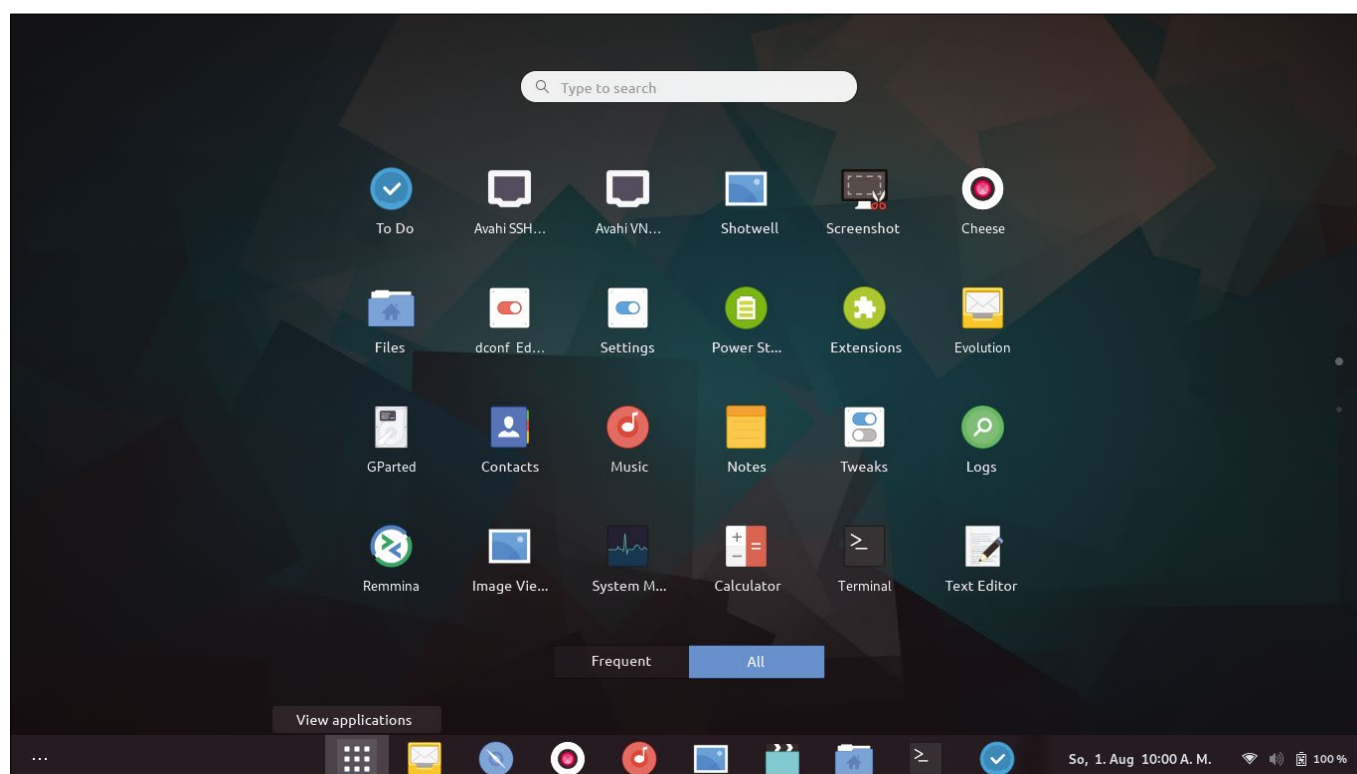


Figure 5: The app grid, which you may be familiar with from Gnome, can be reached by clicking on the icon with the nine dots located on the left in the taskbar.

After installing a fresh rlxos, you will first want to check whether the system can be updated by typing

```
sudo apt-get update
```

in a terminal (Figure 6). If, unlike my experience, an update is available, then execute the commands shown in the last two lines of Listing 1. You can decide at the next reboot whether you want to start the new or the old image.

Alternatively, to update the system, copy an updated system image from the official website to `/run/initramfs/rlxos/system/` and then update the GRUB configuration. Even then, you can boot the different versions (all residing in one partition) from GRUB as needed. If you run rlxos as a dual or multiboot system, the *osprober* package must be installed.

Universal Package Formats

Although the developers claim AppImage is supported, I had some problems with the AppImage format in testing. No matter where the packages came from, they could not be persuaded to start, although I had made the packages executable using:

```
sudo chmod a+x PACKAGE
```

Table 1: Working with apt-get

Command	Function
<code>sudo apt-get sync</code>	Synchronize repositories
<code>sudo apt-get install PACKAGE</code>	Install packages
<code>sudo apt-get remove PACKAGE</code>	Remove packages
<code>sudo apt-get info PACKAGE</code>	Retrieve information about a package
<code>sudo apt-get depends PACKAGE</code>	List the dependencies for a package
<code>sudo apt-get list</code>	List all installed packages
<code>sudo apt-get search PACKAGE</code>	Search for a package

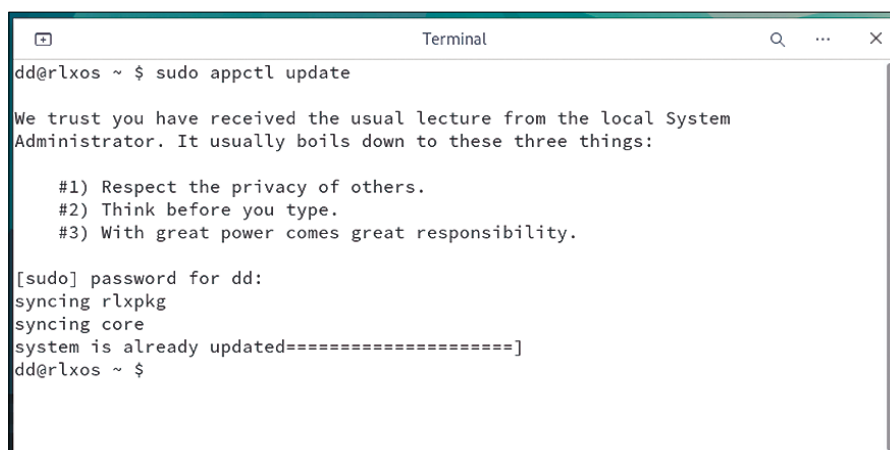


Figure 6: Similar to updating on Debian, you can trigger an update on rlxos with the built-in package manager PKGUPD via apt-get.

Listing 1: Updating rlxos

```
$ sudo apt-get update
$ sudo mount /run/initramfs/boot /boot --bind
$ sudo update-grub
```

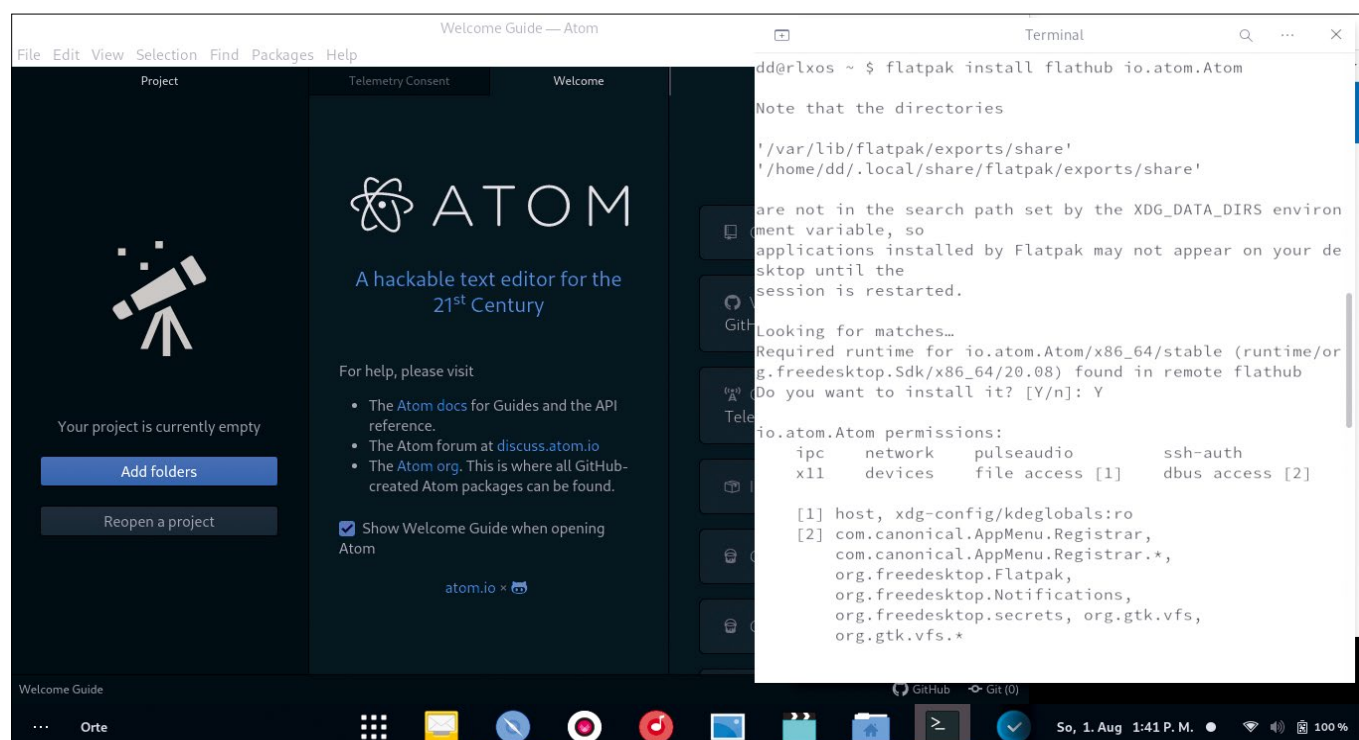


Figure 7: Installing Flatpaks works like a charm. After installing the framework, the connection to Flathub, the Flatpak store, is already in place.

There was supposedly an app missing.

After some rummaging around, I discovered that FUSE was not installed. I quickly remedied that by typing:

```
sudo appctl install fuse3
```

After a reboot, I could launch the downloaded AppImages. To include them in the app grid, I had to call the `appimaged` command, which takes care of the integration and includes downloaded AppImages in the future after the first launch.

I had no problems integrating Flatpak. The framework was quickly installed on my disk by typing:

```
sudo appctl install flatpak
```

and already integrated Flathub. I then only had to execute the Flatpak reference

files downloaded from there. As an example, I downloaded the Atom editor and installed it with the command:

```
flatpak install flathub io.ato.Atom
```

After that, Atom was immediately available for use (Figure 7). I did not test Snap.

Conclusions and Outlook

Positives first: rlxos is fun, installs in no time, and also runs fast. The Gnome shell is inviting to work with because it does not follow the Gnome developers' somewhat unrealistic design philosophy. If you have no ideological problems with Flatpak and AppImage, you can work well with rlxos and don't need to be afraid of breaking your system.

The rlxos website used to have its own own AppImage app store, called Bazaar [8],

but it only contained four apps, and it appears the store is now missing from the site. You can, however, find plenty of AppImages at AppImageHub [9] and the GitHub store [10]. The integration of Image still needs some fine-tuning. The rlxos developers need to simplify the GRUB update after installing a new image to a single short command or provide integration at the push of a button.

The welcome tour offers a preview of what is to come (Figure 8), including a virtual assistant called rlxbot, the src programming language derived from JavaScript, and a system monitor dubbed health, which keeps an eye on the system's operating status. Judging by the very young age of the distribution, it is in surprisingly good shape, considering that the developers are building rlxos from scratch. If development continues at this brisk pace, we'll definitely be relaxing with rlxos again in a year or two. ■■■

Info

- [1] rlxos: <https://rlxos.dev>
- [2] Documentation: <https://docs.rlxos.dev>
- [3] GitHub: <https://github.com/rlxos>
- [4] rlxos blog: <https://blog.rlxos.dev>
- [5] PKGUPD: <https://blog.rlxos.dev/01-introduction-to-pkgupd-ckr1s6kxp0ffhqus1b4fyeelr>
- [6] Installation variant: <https://blog.rlxos.dev/installing-rlxos-from-any-other-already-installed-linux-distribution-ckqi52gk903m77ts12ykg86ve>
- [7] Repositories: <https://docs.rlxos.dev/package-management/appctl>
- [8] Bazaar: <https://rlxos.dev/apps>
- [9] AppImageHub: <https://www.appimagehub.com/>
- [10] GitHub store: <https://appimage.github.io/apps/>

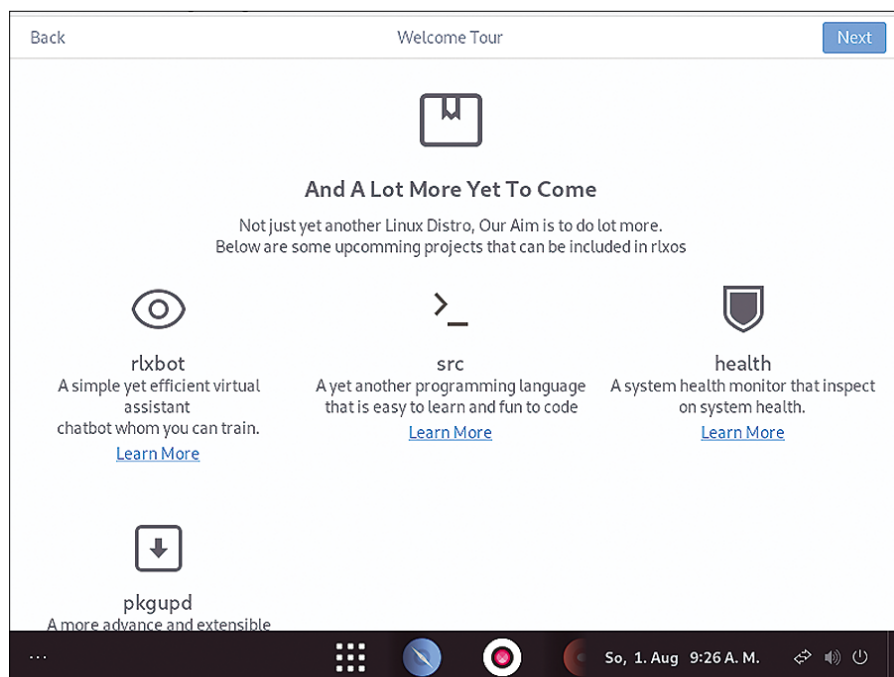


Figure 8: The welcome tour offers a preview of imminent changes in rlxos.

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. Subscribe today for our excellent newsletters:

ADMIN HPC • ADMIN Update • Linux Update
and keep your finger on the pulse of the IT industry.

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

Monitor hard disk usage with Go

Hard Disk Dashboard

To keep an eye on the remaining disk space during storage-intensive operations, you can check out this speedometer/odometer written in Go.

By Mike Schilli

Storage-hungry applications such as video-editing software gobble up disk capacity like a massive vacuum cleaner: Before you know it, everything has been used up. In situations such as this, incorrectly programmed software tends to crash, and all the time you invested in your current project is irretrievably lost. If you take precautions up front, you can avoid headaches later on.

How about a constantly updated display of the remaining space on a dashboard-like instrument on the desktop, where you can see out of the corner of your eye how much disk space is wasted by an action that has just been triggered, such as rendering a video? You can write something like this quickly in Go.

On the Dashboard

A car's dashboard shows the current speed as well as the mileage. If you

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.



apply this to hard disks, the car's mileage reading becomes the total disk space consumed. Similarly, where in a car the speedometer needle shows the current speed, in the hard disk universe, the needle measures the space consumed per unit of time. The analogy for a storage-hungry application would be a speeding traffic offender, if you like.

Figures 1 and 2 show the terminal output from the finished Go program, Disk Speedo (or `df`top). At the top, a progress bar illustrates the amount of disk space used thus far (in this case, 35 percent). At the bottom, a speedometer needle implemented as a pie chart indicates whether space is disappearing (red) or coming back (green) and how fast this is happening. At a simulated speed of 0 to 100, the needle of this somewhat unusual instrument starts at the bottom of the circle and then moves upwards in a counter-clockwise direction. Figure 1 shows a write speed of 35; Figure 2 shows a delete action at a speed of 65. At a speed of 100, the circle would be completely filled with the corresponding color, red or green.

Avoid Shell Calls

To determine the remaining space on a data volume, the program could repeatedly call the `df` shell function. But this would waste valuable resources, because the shell would have to start a

new `df` process each time. Fortunately, the nifty `statfs` programming interface [1] on Unix systems reports the total number of blocks provided on the corresponding mount as well as the blocks on the storage medium that are still unoccupied, without needing to call any shell utilities.

The Go interface for `statfs` gives you the total number of free blocks with `Bfree()` and the subset of free blocks that the non-root user can occupy with `Bavail()`. Multiplying these numbers by the block size defined on the storage medium as `Bsize()` gives you the

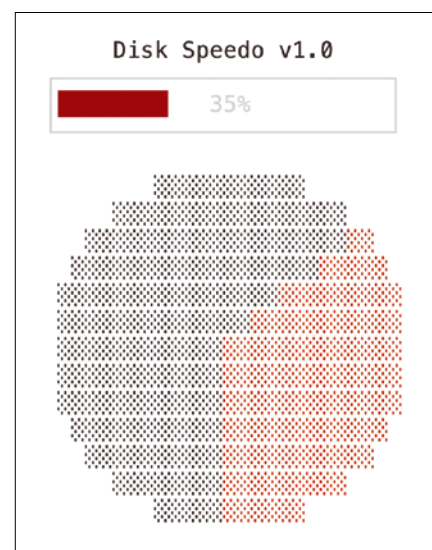


Figure 1: The disk is 39 percent occupied, and a write is consuming more space.

Lead Image © Author, 123RF.com

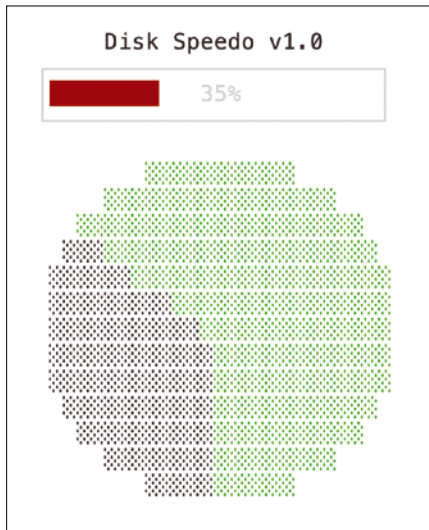


Figure 2: The green speedometer indicates a deletion, which frees up disk space.

remaining storage space in tera-, giga-, or whatever-bytes.

The `space()` function from line 61 in Listing 1 [2] determines the utilization of the storage medium and returns values for the number of occupied blocks, as well as their total number on a particular volume. If an error occurs when determining the capacity, `space()` pass it back as the third return value to the calling main program.

Listing 1: dftop.go

```
01 package main
02 import (
03     "container/ring"
04     "golang.org/x/sys/unix"
05     "os"
06     "time"
07 )
08
09 func main() {
10     wd, err := os.Getwd()
11     if err != nil {
12         panic(err)
13     }
14
15     ui := NewUI()
16     ui.Update(0, 0.0)
17     uidone := ui.Run()
18     defer ui.Close()
19     r := ring.New(2)
20
21     for {
22         used, total, err := space(wd)
23         if err != nil {
24             panic(err)
25         }
26
27         r.Value = used
28         p := used * 100 / total
29         ui.Update(int(p), speed(r))
30         r = r.Next()
31
32         select {
33             case <-uidone:
34                 return
35             case <-time.After(
36                 1 * time.Second):
37                 continue
38         }
39     }
40 }
41
42 const maxSpeed = 100000
43
44 func speed(r *ring.Ring) float64 {
45     if r.Prev().Value == nil {
46         return 0
47     }
48     s := float64(int(
49         r.Value.(uint64)-
50         r.Prev().Value.(uint64))) /
51         maxSpeed
52
53     if s > 1 {
54         s = 1
55     } else if s < -1 {
56         s = -1
57     }
58     return s
59 }
60
61 func space(dir string) (
62     uint64, uint64, error) {
63     var stat unix.Statfs_t
64     err := unix.Statfs(dir, &stat)
65     return stat.Blocks -
66         stat.Bfree, stat.Blocks, err
67 }
```

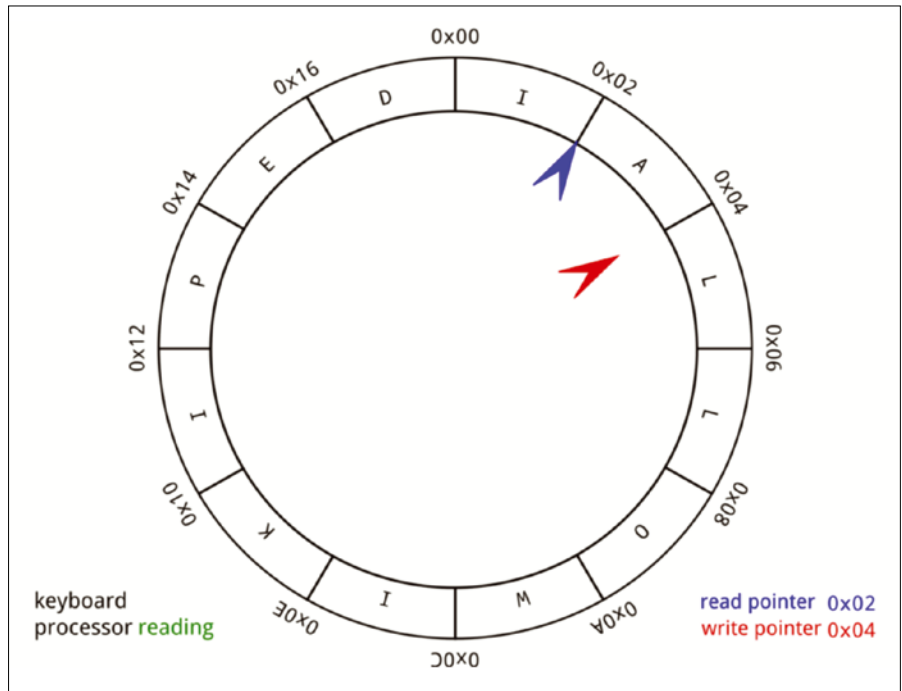


Figure 3: A ring buffer automatically overwrites old and obsolete values. Wikipedia, CC BY-SA 4.0

But which hard disk's capacity will the program actually measure on a system with multiple storage media? Depending on the directory from which you call the speedometer, it will display the space on the hard disk it is residing on.

Storage in a Circle

The speed at which the hard disk fills up is defined by the difference between two measurements of the fill level at different times divided by the time elapsed between them.

To do this, the program needs to store one or more past measurements in order to determine the delta to the currently measured value. This could be implemented using separate variables, but a ring buffer (Figure 3) of the `container/ring` type from the Go standard library does this in an elegant way without much code.

The ring buffer stores new values in `r.Value`, in sequence, in points that lie on a circular path. `r.Next()` moves to the next point, and `r.Prev()` goes back to the previous one. If the algorithm arrives at the first point again at some point on its

circular path, it simply overwrites it. A ring buffer can only ever access the N most recent values, but it does not clutter the system's memory with irrelevant values from the past. A ring buffer created as shown in line 19 (Listing 1) with only two entries certainly does not leverage the data structure's full potential, but if desired, you can expand the buffer to include more entries for averaging and smoothing the display.

The `speed()` function starting in line 44 computes the current filling speed of the storage medium using this procedure. If the ring buffer does not yet carry two

values because the algorithm just started out, the speed cannot yet be determined and line 46 returns the value 0.

Signed vs. Unsigned

The `Statfs()` function returns the remaining disk space in line 64 as `uint64`, (i.e., as an unsigned 64-bit integer that can never assume a negative value). However, the difference between two of these values can definitely be negative.

If you simply subtract both values from each other, you might be surprised to see that Go (like other languages) returns absolutely insane values for the

Listing 2: `ui.go`

```
01 package main
02 import (
03     "math"
04     tui "github.com/gizak/termui/v3"
05     "github.com/gizak/termui/v3/widgets"
06 )
07 type UI struct {
08     Gauge *widgets.Gauge
09     Pie    *widgets.PieChart
10     Head  *widgets.Paragraph
11 }
12
13 func NewUI() *UI {
14     h := widgets.NewParagraph()
15     h.TextStyle.Fg = tui.ColorBlack
16     h.SetRect(6, 1, 30, 2)
17     h.Text = "Disk Speedo v1.0"
18     h.Border = false
19     g := widgets.NewGauge()
20     g.SetRect(2, 2, 28, 5)
21     g.Percent = 0
22     g.BarColor = tui.ColorRed
23     p := widgets.NewPieChart()
24     p.SetRect(-10, 5, 40, 20)
25     p.Border = false
26     p.Data = fract(0)
27     p.AngleOffset = -1.5 * math.Pi
28     return &UI{Gauge: g,
29         Pie: p, Head: h}
30 }
31
32 func (ui UI) Run() chan bool {
33     done := make(chan bool)
34     err := tui.Init()
35     if err != nil {
36         panic("termui init failed")
37     }
38
39     go func() {
40         events := tui.PollEvents()
41         for {
42             select {
43                 case e := <-events:
44                     switch e.ID {
45                         case "q", "<C-c>":
46                             done <- true
47                             return
48                     }
49             }
50         }
51     }()
52     return done
53 }
54
55 func (ui UI) Close() {
56     tui.Close()
57 }
58
59 func (ui UI) Update(
60     level int,
61     speed float64) {
62     ui.Gauge.Percent = level
63     ui.Pie.Colors = []tui.Color{
64         tui.ColorBlack, tui.ColorRed}
65     if speed < 0 {
66         ui.Pie.Colors[1] =
67             tui.ColorGreen
68         speed = -speed
69     }
70     ui.Pie.Data = fract(speed)
71     tui.Render(ui.Head, ui.Gauge,
72         ui.Pie)
73 }
74
75 func fract(
76     val float64) []float64 {
77     num := (1 - val) * 100
78     denom := val * 100
79     return []float64{num, denom}
80 }
```


difference if the subtrahend (the number we're subtracting) turns out to be larger than the minuend (the number we're subtracting from). The result should be negative in this case, but instead you get very large positive values. Without any help, Go assumes that the result of an operation with two unsigned integers is also an unsigned integer. The solution: Typecasting with `int(x-y)` makes it clear to Go that the result of the difference of two `uint64` values `x` and `y` is in fact a signed value.

But before line 51 divides the difference by an empirically determined value of 100,000, resulting in a floating point value between 0 and 1 for the average disk performance, line 48 first needs to convert the result type to `float64`. Now, if the velocity is greater than 1 or less than -1, the if-else construct from line 53 squashes it into the range between -1 and +1.

All that remains for the main program in Listing 1 to do is to determine the current directory (line 10), start the user interface (UI) (line 17), and close it in any situation with the `defer` statement in line 18 as soon as the main program stops running.

On the Screen

Listing 2 shows how the program conjures up the UI with the speedometer display in a terminal window using the tried and tested `termui` project from GitHub. It defines three stacked widgets: a paragraph widget to display the program name `Disk Speedo v1.0`, a progress bar of the `Gauge` type, and a pie chart of the `PieChart` type that forms the speedometer.

The `NewUI()` constructor from line 13 defines the widgets, including their dimensions, geometric positions, and colored bits, and returns an initialized structure of the UI type (defined in line 7). The main program uses this later to call method-style functions such as `Run()` (from line 32) and to give them the context of the previously initialized widgets – object orientation Go-style.

Listening on the Channels

The `Run()` function from line 32 onwards starts the `termui` interface in a parallel Goroutine from line 39 onwards. Like every UI, it also continuously delivers events such as keystrokes or mouse clicks that need to be intercepted and processed in order to give the illusion of a seamless user interaction.

The infinite loop from line 41 waits with its `select` statement for the user to press either `Q` or `Ctrl+C`. In this case, it sends the value `true` into a newly created channel, `done`, to the listening main program, causing it to stop all operations. To do this, the main program simultaneously monitors – in the `select` statement from line 32 of Listing 1 – whether the UI has reported the end of the program or, alternatively, the ticking seconds timer (Listing 1, line 35) has expired. If the timer has expired, the program moves on to the next round, fetches new values for the disk fill level, and displays them.

In general terms, Listing 2 abstracts the features of the `termui` library and encapsulates them from the main flow. When it's time to pack up shop, the main program simply calls the `Close()` function (Listing 2 from line 55) to wrap up the terminal UI, which in turn triggers a `Close()` call in the `termui` library.

Bright Paint

The `Update()` function, which Listing 2 defines starting in line 59, writes new values to the display. It expects two values: the fill level of the hard disk as an integer and the measured fill speed as a `float64` value. It passes the fill level to the `Gauge` widget of the `termui` library in line 62 and the fill speed to the pie chart in `ui.Pie`.

To make the pie chart paint a positive fill speed in red and a negative one in green, lines 63 and 64 sets the colors of the slices in the pie chart to black and red, and lines 66 and 67 modifies the second color to green in case of a negative speed. Because the graph only processes positive values, line 68 reverses the sign of the negative velocities after

Listing 3: Compiling

```
$ go mod init dftop
$ go mod tidy
$ go build dftop.go ui.go
```

the color has been adjusted. The `termui` `Render` function paints all three widgets onto the terminal's canvas in lines 71 and 72, refreshing the display at intervals of one second.

Now how does the pie chart paint the speedometer reading based on a floating-point value for speed between 0 and 1? It does this by calling the `fract()` function starting in line 75, which – in turn – produces a fraction using the formula $(1-val)/val$, which yields the ratio of the speed's colored area (green or red) divided by the size of the black area. For percentage values, lines 77 and 78 also multiply the numerator and denominator values by 100.


In this way, for a floating-point value of 0.35, for example, `fract()` returns 0.65×100 and 0.35×100 , (i.e., 65 and 35). Consequently, the red speedometer segment shown (see Figure 1) occupies about one third of the total circle, and the remaining two thirds are left black on the left side.

Build Time!

You can compile the whole enchilada with the calls from Listing 3, which retrieve the `termui` UI and its dependencies from GitHub and then go ahead to build the `dftop` binary. When invoking the finished program from the command line, Listing 3 produces the output from Figures 1 and 2. If you run `dftop` in a window in a corner of your desktop, you can keep an eye on your remaining hard disk capacity, and prevent overfilling before it's too late. ■■■

Info

- [1] stats: <https://man7.org/linux/man-pages/man2/statfs.2.html>
- [2] Listings for this article: <ftp://ftp.linux-magazine.com/pub/listings/linux-magazine.com/253/>



Monitor your media with Hard Disk Sentinel

Agile Controller

Hard Disk Sentinel helps you monitor mass storage devices with a fully automated process minus the bells and whistles. *By Erik Bärwaldt*

Mass storage devices are taking on an increasingly central role in modern PCs due to the daily increasing flood of data. This makes it all the more important for users in the private sector and in the enterprise to keep an eye on the storage systems installed in their computers, to avoid data loss due to defects or overheating. The Hard Disk Sentinel [1] program helps you keep a permanent eye on the data media so that you always have a current status for your mass storage devices.

Software

Hard Disk Sentinel, which originates from Hungary, has become the industry standard in the field of professional mass storage maintenance on non-Linux operating systems over the past 10 years. The proprietary software is available, for a fee, in several versions for these operating systems.

For Linux, there is a free variant that continually monitors important operating parameters of the mass storage devices [2]. The variant is available for both 32- and 64-bit systems. The two packed tar archives, around 2.5MB each, are available for download from the manufacturer's website.

In addition, the developers offer a version for the command line for computers without a graphical user interface (GUI) [3]. Numerous parameters enable queries here, and some of them are useful for documenting the hardware status.

Installation

After downloading the archive for your system, unzip it to any directory. This will create the new `HD Sentinel_GUI/` folder where you will find several files and another ZIP archive. Now call the `./install.sh` command in the terminal in this directory. The script installs and configures the software after you enter the appropriate authentication.

After the install, you will find a *Hard Disk Sentinel GUI* entry in your desktop program menu. When launched, the software first prompts you for your password to adopt your privileges with `Sudo`. It then scans the system for mass storage devices. The list includes conventional PATA and SATA devices as well as modern NVMe media.

Older storage devices that you connect to the system via the PCIe bus and that require special firmware are also correctly identified by the tool. Even memory card

readers and removable media connected via USB appear in the Overview (Figure 1).

In the Overview, you will see the basic data and the status of the active drive: The software shows the current and maximum temperature as well as the operating hours. Hard Disk Sentinel also gives you information about the current overall status of the drive in a small text box.

If the values for the temperature tend to fluctuate, you can manually refresh the display by pressing *Refresh*. Based on the SMART values, the tool also determines the lifetime of the respective drive. Of course, this value is generally only of limited significance because it is based on manufacturers' estimates.

Hard Disk Sentinel differentiates between solid-state drives (SSDs) and conventional disks. In the case of SSDs, Hard Disk Sentinel does not indicate the expected operating life but rather the capacity in gigabytes that the active drive can still handle in transfer scenarios (Figure 2).

The application marks problematic status values with an exclamation mark in a yellow triangle (instead of a check mark in a green box) to the left of the respective status bar. In the text field, Hard Disk Sentinel also displays hints as to

Photo by Parker Coffman on Unsplash

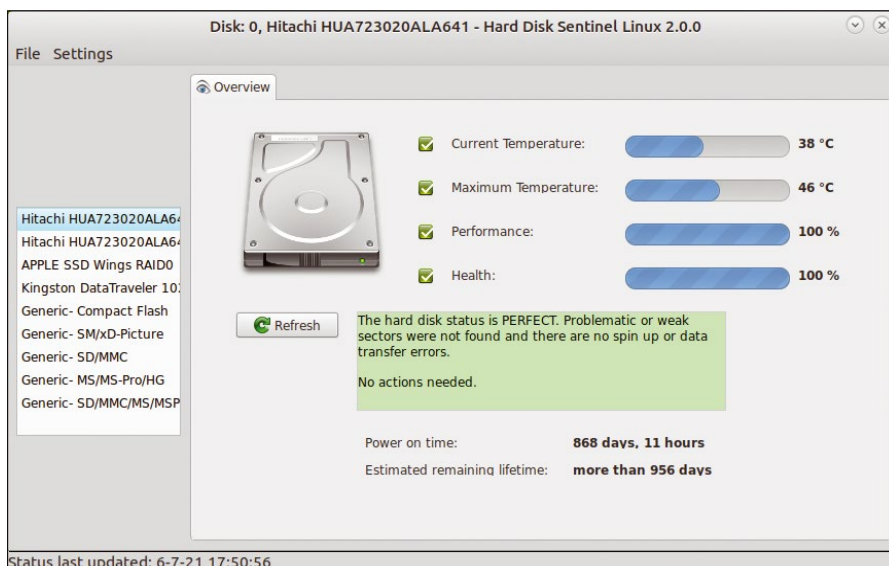


Figure 1: The user interface is purpose-built.



Figure 2: For modern SSD drives, the tool checks how many write cycles the media can potentially handle.

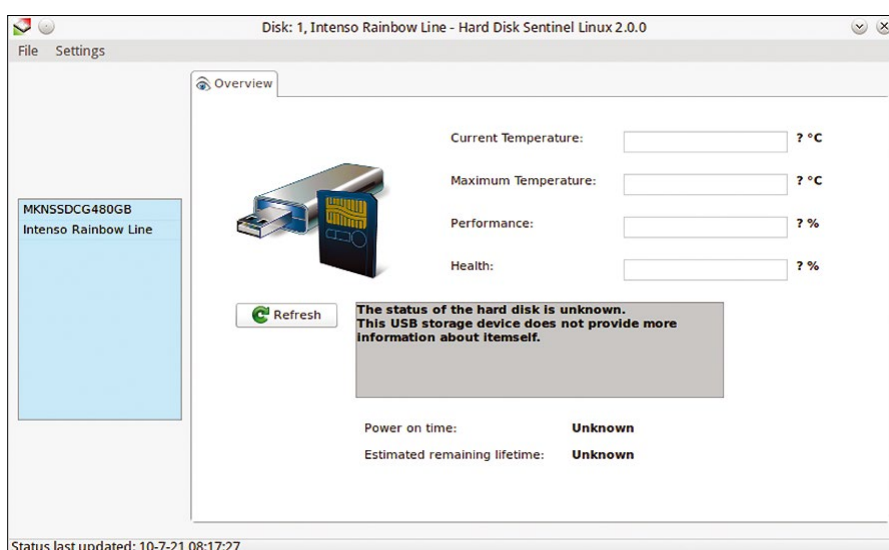


Figure 3: Because of incomplete controller firmware, Hard Disk Sentinel cannot monitor removable flash media.

what actions are required (if any) on your part to avoid an imminent failure of the drive.

Exceptions

The controllers for most current flash storage devices such as USB media, (micro) SD cards, or the CompactFlash cards that are gradually going out of fashion do not implement the SMART command set. As a result, Hard Disk Sentinel identifies these devices but cannot read and display operating data. The corresponding fields therefore remain empty for these media (Figure 3).

System Tray

After startup, the application places a small icon in the system tray of your desktop environment. The icon indicates the temperature of the active mass storage device with a green background for tolerable values and a red background if temperatures are too high.

In the icon's context menu, you can select the *Show main window* option to restore the minimized program window to the foreground. The *Exit* option closes the tool and also removes it from the system tray.

At the Prompt

Professional server systems usually do not have a GUI; administration is usually via a secure connection in the shell. For admins and desktop users who prefer to use command-line tools, the developers of Hard Disk Sentinel offer a command line variant of the tool that offers the same feature set as the GUI version. You control the tool with a set of call parameters. The command-line version lets you document the status of mass storage devices via a report generation routine with 32- and 64-bit variants are available.

Unfortunately, an error has crept into the installation description of the latest version, 0.19, which makes using the software more difficult. Because the developers accidentally packed the GZIP archive offered for download twice, you first need to unpack the archive with the integrated graphical front end of your working environment. The second archive created by this step then also needs to be unpacked. However, many front ends do not recognize the different format, which is why I would recommend

Listing 1: Setup and Start

```
01 $ sudo chmod 755 HDSentinel
02 $ sudo ./HDSentinel PARAMETER
```

using a professional program for handling archives, such as PeaZip.

The unpacked archive contains only the executable file, HDSentinel, to which

you need to assign execute permissions before you can use the command shown in line 1 of Listing 1. Then invoke the software by entering the command from line 2 of Listing 1.

You will find more information about the parameters and how to use them on Hard Disk Sentinel's website. Figure 4 shows examples of the output in a terminal or on the console.

If required, you can also create reports in text, HTML, or XML format that document the health state of a mass storage device. The command-line tool also offers parameters to let you do this (Figure 5). The report in HTML format is far more detailed than the information displayed by the graphical front end.

In this way, you can both document the implemented features of the respective interfaces and generate a SMART table with the current values. As an administrator, you can use the values to proactively discover data carriers that are candidates for replacement.

```
root@hp-Z600:/home/erik/Downloads# ./HDSentinel -dev /dev/sda
Hard Disk Sentinel for LINUX console 0.19b.9986 (c) 2021 info@hdsentinel.com
Start with -r [reportfile] to save data to report, -h for help

Examining hard disk configuration ...

HDD Device 0: /dev/sda
HDD Model ID : Hitachi HUA723020ALA641
HDD Serial No: YFH58NYD
HDD Revision : MK70A840
HDD Size : 1907729 MB
Interface : S-ATA Gen3, 6 Gbps
Temperature : 38 °C
Highest Temp.: 46 °C
Health : 100 %
Performance : 100 %
Power on time: 869 days, 15 hours
Est. lifetime: more than 955 days
The hard disk status is PERFECT. Problematic or weak sectors were not found
and there are no spin up or data transfer errors.
No actions needed.
```

Figure 4: The command-line version shown here displays the essential status data for a disk in the terminal.

The screenshot shows the Hard Disk Sentinel HTML report in a Mozilla Firefox browser window. The report is titled "Hard Disk Sentinel" and includes the following sections:

- General Information**
 - Application Information**
 - Installed Version : Hard Disk Sentinel 0.19b
 - Current Date And Time : 11-7-21 18:44:54
 - Computer Information**
 - Computer Name : hp-Z600
 - MAC Address : f4:ec:38:a3:87:0f
 - System Information**
 - OS Version : Linux : 5.8.0-59-generic (#66-20.04.1-Ubuntu SMP Thu Jun 17 11:14:10 UTC 2021)
 - Process ID : 5027
 - Uptime : 6452 sec (0 days, 1 hours, 47 min, 32 sec)
- Physical Disk Information - Disk: #0: Hitachi HUA723020ALA641**
 - Hard Disk Summary**
 - Hard Disk Number : 0
 - Hard Disk Device : /dev/sdb
 - Interface : S-ATA Gen3, 6 Gbps
 - Hard Disk Model ID : Hitachi HUA723020ALA641
 - Firmware Revision : MK70A840
 - Hard Disk Serial Number : YFH58NYD
 - Total Size : 1907729 MB
 - Current Temperature : 38 °C (100 °F)
 - Maximum Temperature (during Entire Lifespan) : 46 °C (113 °F)
 - Power On Time : 1018 days, 17 hours
 - Estimated Remaining Lifetime : more than 806 days
 - Health : 100 % (Excellent)
 - Performance : 100 % (Excellent)
 - ATA Information**

A green box at the bottom of the report states: "The hard disk status is PERFECT. Problematic or weak sectors were not found and there are no spin up or data transfer errors. No actions needed."

Figure 5: The report in HTML format documents the status of a mass storage device and the interfaces in detail.

Conclusions

The compact Hard Disk Sentinel tool is dedicated to monitoring the state of the mass storage devices in a system. The user interface and the context menu focus on the essentials. The application is frugal in its use of resources and provides a quick overview of all mass storage devices with the help of a few icons and short explanations.

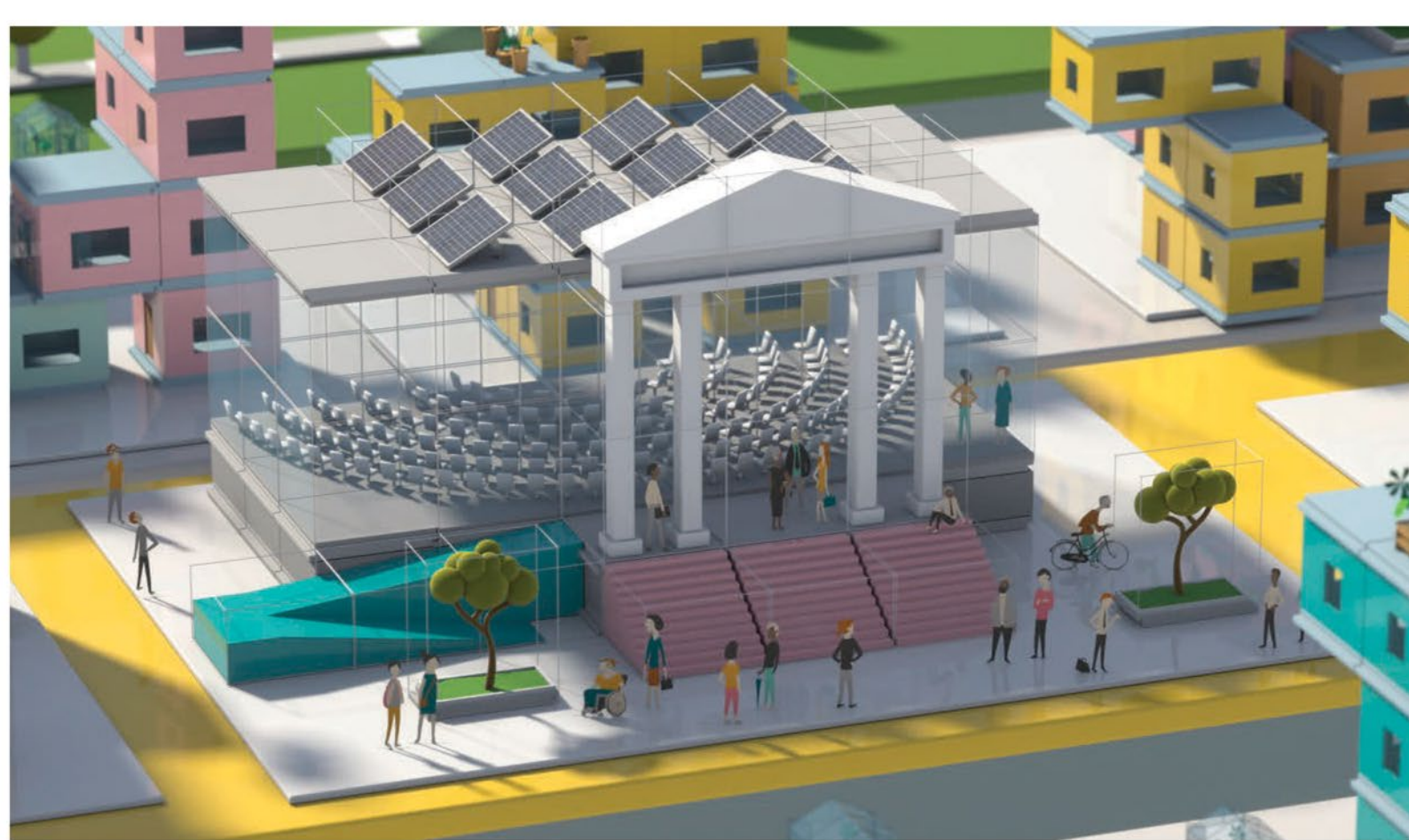
Hard Disk Sentinel displays flash drives connected to the computer's USB ports but cannot read out status values. You can select multiple disks installed in the system in the GUI, one after the other, to keep an eye on the entire subsystem without detours. You will not want to do without Hard Disk Sentinel on any system with important data files. ■■■

Info

- [1] Hard Disk Sentinel:
<https://www.hdsentinel.com>
- [2] Linux version with GUI:
https://www.hdsentinel.com/hard-disk-sentinel_linux_gui.php
- [3] Command-line version:
https://www.hdsentinel.com/hard-disk-sentinel_linux.php

Public Money

Public Code



Modernising Public Infrastructure
with Free Software



Free Software Foundation Europe

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



MakerSpace

Building a personal note-taking tool Homebrew

If you're tired of the privacy problems and feature bloat of high-end note-taking utilities, try rolling your own.

By Stuart Houghton

For many people, online note-taking and time management tools are useful and sometimes essential, but they come with a potential privacy cost. I decided to try to build a usable replacement for the services I had come to rely on using some simple Linux tools and a Raspberry Pi.

If you are like me, your phone is a constant companion in both work and free time. I work as a sys admin and part-time writer, and I have come to rely on “free” note-taking apps to keep track of work, jot down ideas, and generally manage my life. I started out with the now-defunct Catch Notes in the early '00s; the ability to write a quick note on my phone and have near-instant access to it on my PC via a browser (and vice-versa) made Catch Notes an invaluable tool. I could write a note on my phone, then open it on the PC to paste it into a document or follow a link, without having to manually save or import anything on either platform. Being able to add a simple reminder to to-do items was handy too.

Sadly, Catch Notes went the way of so many '00s startups and shut down. Forced to look elsewhere, I exported my notes and imported them into Evernote. Although Evernote was undeniably useful, it was a noticeably more

commercial tool that was moving away from the simplicity I craved. Moreover, it began to worry me that my data – my precious shopping lists, ideas for a novel, and other random jottings (but nevertheless MY data) – was being held on someone else's computer and could in theory be lost if Evernote went out of business.

When Evernote announced a change to their subscription model that would restrict the number of devices from which I could access my notes, I decided enough was enough and resolved to take back control of my notes. But how?

Several options are available for self-hosting note-taking apps, including powerful suites such as Nextcloud and sophisticated notebook apps such as Joplin. However, these tools were either overkill (I didn't need an entire groupware suite such as Nextcloud just to use one notes plugin) or too inflexible and tied to certain apps or platforms. I felt certain that most, if not all, of what I needed probably existed on my Linux PC already. I just needed to tie the pieces together, the Unix way.

File Format

Choosing a file format was crucial. I prefer to use plain text wherever possible, as you can find a text editor on pretty

Photo by Robert Anasch on Unsplash

much any platform and it can easily be converted to other formats if required. In order to give more options for formatting and embedding links, I decided to use Markdown, which allows me to add simple formatting such as italics and bold text by adding asterisks around words or mark some lines as headings with a #, for example.

Storage

I could see some advantages to storing notes in a database (fast indexing and searching, for example), but for me, the benefits of a database were outweighed by the simplicity of storing notes as plain files in a directory tree. A tree of text files is still searchable, and it fits my requirement for not being tied to any particular platform. I could easily run an SQL database on my Linux PC but not so much on my phone.

So, notes are stored in a folder called **Notes** as plain text files. Subfolders underneath **Notes** allow me to stick notes in categories, if required, as follows:

```
Notes
├── Important
├── Writing
├── Ideas
└── Organizing
```

For physical storage, I decided to use a Raspberry Pi server that I run with a USB drive plugged in for extra space. The Pi's main job is a home media server, but it

also runs a simple `lighttpd` web server that I use for various projects. In this case, though, all it needs to do is store the **Notes** folder and sync with the other platforms that I use.

Synchronizing

To sync data between the Pi, my laptop, and my phone, I decided to use **Syncthing** [1]. Syncthing is really easy to set up and does one job – keep folders in sync between different devices. If I make a change to the **Notes** folder on my PC (by adding a note or editing an existing note), Syncthing will replicate the change to the Pi and from there to my phone.

Remind Me

If I have an important meeting or need to remember an anniversary, I already have several very capable calendar apps linked to my Gmail account that will send a reminder. Despite this, I am a terrible procrastinator and often put off or forget tasks on my mental to-do list.

The obvious answer would be a physical – or digital – to-do list, but I wanted to know if I could make it just a bit more useful for someone with my scattergun approach to time management.

Gina Trapani's excellent **Todo.txt** method [2] ticked all my boxes by using an open, plain-text file format and being completely app agnostic. For the unfamiliar, **Todo.txt** is just a way of formatting a text-based to-do list so that it is

both human readable and easy to process with a computer.

An example **Todo.txt** file would be:

```
Buy some milk @shopping
(A) Call Grandma
Write Linux article
Tidy spare room
```

@shopping is just a tag to help you list all similar items using the basic **Todo.txt** shell script or one of the many **Todo.txt**-compatible apps. The optional **(A)** marks that task as high priority. You could also have tasks categorized as **(B)**, **(C)**, etc.

I can keep a **Todo.txt** file in a folder called **Organizing** under my **Notes** folder, sync it between all my devices, and always have easy access to my list of tasks. But how could I use it to help me remember to actually do those tasks? I thought what might help would be to get a random nudge throughout the day. I sometimes find that if I set an alarm for something, if I know when it is coming – paradoxically – it is easier to ignore. If a friend were to randomly interrupt me with a reminder, that might be harder to dodge.

A simple python code could read my to-do list every day and add a timed task using the **Linux at** command to give me that encouraging nudge when I least expected it.

This script (see Listing 1) requires Python v3 or later and the `todotxtio` mod-

Listing 1: Nudgebot

```
001 #!/usr/bin/python3
002 # nudgebot
003 # stuart houghton 2021
004 #
005 # read a todo.txt file and assemble a list of timed
    reminder notifications
006 #
007 import todotxtio
008 import sys, getopt, os
009 from random import randint
010
011 # look for parameters
012 todo_file = "null"
013 argv = sys.argv[1:]
014
015 try:
016     opts, args = getopt.getopt(argv, "f:")
017
018 except:
019     print("Error")
020
021 for opt, arg in opts:
022     if opt in ['-f']:
023         todo_file = arg
024
025 if (todo_file == "null"):
026     print ("No todo list specified")
027     quit()
028
029 # define the flavour text list
030 # this keeps each reminder slightly different, to make
    them more impactful
031
032 flavour_text=[
033     "Do not forget to",
034     "You had better not forget to",
```

Listing 1: Nudgebot (continued)

```

035     "Whatever happens, remember to",
036     "ATTENTION!",
037     "You really should",
038     "You had better",
039     "If you know whats good for you,",
040     "Dig deep and",
041     "Do the right thing, i.e.",
042     "Do this:"
043 ]
044 flavour_text_list_length=len(flavour_text)-1
045
046 # define the notification tone list
047 # these are standard notification tones used by the
    pushover notification service
048
049 push_tones=[
050     "pushover",
051     "bike",
052     "bugle",
053     "cashregister",
054     "classical",
055     "cosmic",
056     "falling",
057     "gamelan",
058     "incoming",
059     "intermission",
060     "magic",
061     "mechanical",
062     "pianobar",
063     "siren",
064     "spacealarm",
065     "tugboat",
066     "alien",
067     "climb",
068     "persistent",
069     "echo",
070     "updown"
071 ]
072 push_tones_length=len(push_tones)-1
073
074 # let's read that file
075 print("Reading ", todo_file, "...")
076
077 list_of_todos = todotxtio.from_file(todo_file)
078
079 # ok, now do the priority A stuff - between 2 and 4
    nudges per day
080 to_nudge_A = todotxtio.search(list_of_todos,
081     priority=['A'],
082     contexts=['nudge','blitz'],
083     completed=False
084 )
085
086 nudge_list_length_A=len(to_nudge_A)
087
088 print("====Priority A====")
089
090 for item in range(nudge_list_length_A):
091     item_text=to_nudge_A[item].text
092     item_duedate=to_nudge_A[item].text
093     #nudge a random number of times
094     rand_nudges=randint(2,4)
095     for nudgenum in range(rand_nudges):
096         print(rand_nudges)
097         rand_mins=randint(0,480)
098         rand_flavour=randint(0,flavour_text_list_length)
099         flavour_intro=flavour_text[rand_flavour]
100         send_text=flavour_intro+" "+item_text
101         rand_tone_index=randint(0,push_tones_length)
102         push_tone=push_tones[rand_tone_index]
103         command_string = f"echo \"/home/stu/bin/pushover
            \'{send_text}\' \'{push_
            tone}\'\' | at now + \'{rand_
            mins}\' minute"
104         print(command_string)
105         os.system(command_string)
106
107 # and now do the priority B stuff - 1 nudge per day
108 to_nudge_B = todotxtio.search(list_of_todos,
109     priority=['B'],
110     contexts=['nudge'],
111     completed=False
112 )
113
114 nudge_list_length_B=len(to_nudge_B)
115
116 print("====Priority B====")
117
118 for item in range(nudge_list_length_B):
119     item_text=to_nudge_B[item].text
120     rand_mins=randint(0,480)
121     rand_flavour=randint(0,flavour_text_list_length)
122     flavour_intro=flavour_text[rand_flavour]
123     send_text=flavour_intro+" "+item_text
124     rand_tone_index=randint(0,push_tones_length)
125     push_tone=push_tones[rand_tone_index]
126     command_string = f"echo \"/home/stu/bin/pushover
            \'{send_text}\' \'{push_tone}\'\' |
            at now + \'{rand_mins}\' minute"
127     print(command_string)
128     os.system(command_string)

```


ule, which is easily installed using the Python package manager, pip.

```
sudo pip3 install todotxtio
```

The notifications are done using a free web service called Pushover [3], which sends a push notification to an Android or iOS app. The API for Pushover is very simple, and the Pushover site has example code for using curl to send notifications that you can put in a Bash script. I called mine pushover. You could, of course, send an email instead or use something like the notify-send notification app [4] to direct a pop-up notification to your desktop.

I then add a crontab file so a daily cron job can run this script (Listing 2).

With the script running on my Pi, I just need to tag a to-do item with @nudge and make it either priority (A) or (B), and I will get one or more automated,

random reminders during the day until I complete the task.

Conclusion

My note-taking solution suits my purposes well. It doesn't have some of the extra features of commercial apps, but it does what I need it to do, and it isn't cluttered up with unnecessary features. Markdown is a portable and sensible note format that is supported by a variety of tools (see

Info

- [1] Syncthing: <https://syncthing.net/>
- [2] Todo.txt: <http://todotxt.org/>
- [3] Pushover: <https://pushover.net/>
- [4] notify-send: <https://manpages.ubuntu.com/manpages/xenial/man1/notify-send.1.html>

Listing 2: crontab File

```
01 # Edit this file to introduce tasks to be run by cron.
02 #
03 # Each task to run has to be defined through a single line
04 # indicating with different fields when the task will be run
05 # and what command to run for the task
06 #
07 # To define the time you can provide concrete values for
08 # minute (m), hour (h), day of month (dom), month (mon),
09 # and day of week (dow) or use '*' in these fields (for 'any').
10 #
11 # Notice that tasks will be started based on the cron's system
12 # daemon's notion of time and timezones.
13 #
14 # Output of the crontab jobs (including errors) is sent through
15 # email to the user the crontab file belongs to (unless redirected).
16 #
17 # For example, you can run a backup of all your user accounts
18 # at 5 a.m every week with:
19 # 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
20 #
21 # For more information see the manual pages of crontab(5) and cron(8)
22 #
23 # m h dom mon dow    command
24 0 8 * * *           /home/stu/bin/nudgebot.py -f /notes-location/Notes/organising/
                        todo.txt
```

the box entitled “Apps,”) and the clever Todo.txt format makes it easy to keep a list of tasks and generate reminders. The simple concepts discussed in this article are easy to adapt to your own needs if you decide to experiment with building your own note-taking tool. ■■■

Apps

The advantage of using Markdown is that you can use any text editor, from something as simple as Notepad on Windows to a Swiss Army chainsaw such as Emacs. That said, there are some apps which are better suited to taking and managing notes. Similarly, although you could edit a Todo.txt file in anything, a number of apps can make the editing process much more user-friendly.

Android

On my phone I use Markor, which is fast and fairly no-frills but supports searching and embedded images. Other options would be iA Writer or JotterPad. Markor has a built-on Todo.txt mode, but it is a little basic. I prefer an excellent free app called Mindstream, which looks good and makes adding new tasks a breeze.

PC

I run Linux on my laptop (naturally), and there are of course hundreds of text editors available for it. For my note-taking needs, however, I use Typora, which looks nice and just works, and ThiefMD. Most of this article was written in ThiefMD, which also supports Fountain, a superset of Markdown geared towards screenwriting. Sleek is a great-looking and very easy to use Todo.txt editor available in most Linux repositories.

Other Platforms

My sysadmin work requires me to manage Windows systems, so being able to view and edit my notes from there is handy too. Syncthing has Windows and iOS ports, and Markdown editors are available for both platforms. I would recommend iA Writer on iOS and Ghostwriter for Windows. Sleek has Mac and Windows ports, and SwiftoDo is a very capable equivalent for iOS.



MakerSpace

Breathe new life into your
old home router

A New Route

If you have an old router lying around, you can put it to good use with a few easy projects and learn something along the way. *By Brooke Metcalfe and Pete Metcalfe*

There are some fun and interesting projects that can be done by repurposing an old home router. If you don't have an old router lying around, you can usually find one for about \$5-\$25. These routers are easy to reflash, so a new programmer doesn't have to worry about messing things up too badly.

In this article, we will look at three projects to get some extra life out of an older router. The first project uses the router's USB port to connect to third-party devices. The second project collects microcontroller and internal data, and the final project displays the data on the router's web page and on a Raspberry Pi.

Selecting a Router and Firmware

A number of open source firmware solutions can breathe new life into an old router. OpenWRT [1] and DD-WRT [2] are the most popular packages, but there are other options. You need to determine if one of these firmware packages supports your old router. Keep in mind that many older routers only have 4MB of flash and 32MB of RAM. These routers may not run or only marginally run OpenWRT or DD-WRT. We recommend that you choose a router with a minimum 8MB of flash and 32MB or more of RAM.

You also need to consider whether the router has USB support. A router without USB support can still be used as a web or application server, but it

will be missing external hardware integration.

For our router project, we used OpenWRT because of our experience using the Arduino Yún modules.

Getting Started

Loading new router firmware will vary based on the make of your router. For this step, you'll need to check the manufacturer's directions and the router specifics from the OpenWRT or DD-WRT websites.

After the new firmware has been loaded, the router should be disconnected from your home LAN, and a PC needs to be wired to one of the router's LAN ports. The router will have a default IP address of 192.168.1.1, and the PC needs an IP address in the 192.168.1.x range (e.g., 192.168.1.10).

Once the router and PC are wired into their own small network, the router can be powered up, and the OpenWRT web interface (LuCI) can be used to configure the new router settings.

There are many possible router configurations. Most importantly, you must ensure that the repurposed router does not effect the main router on your home LAN. Typically, you'll want to disable routing features on the repurposed router before it is connected to the home LAN. In the OpenWRT web interface, software services such as the firewall,

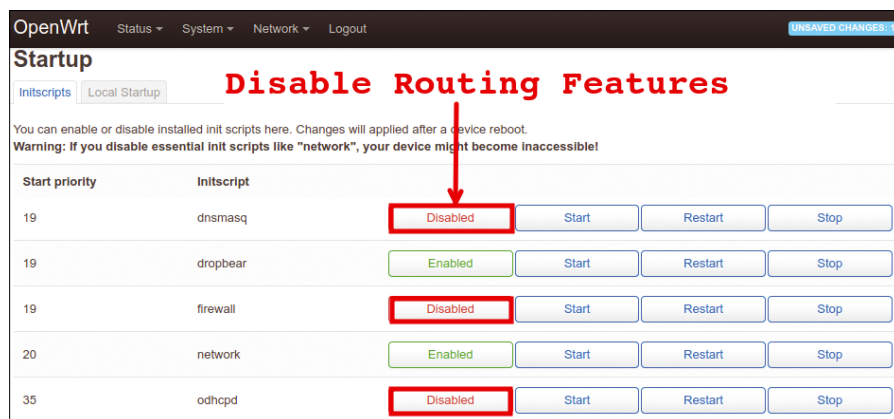


Figure 1: Be sure to disable the firewall, dnsmasq, and odhcpd in the OpenWRT interface.

DNS (dnsmasq), and DHCP (odhcpd) also should be disabled (Figure 1).

For our setup, we used a USB hub so that we could connect a variety of different devices (Figure 2).

Software

Routers don't have a lot of memory, so the default firmware is fairly lean on extra features. Consequently, you will need to add software packages.

OpenWRT uses *opkg* [3], the OpenWRT package manager, to find and install software packages. After the router is connected to the Internet, software can be added either through the LuCI web interface (Figure 3) or manually in an SSH shell.

USB Drives

To add USB drives, remotely SSH into the router and enter the code from Listing 1. After the USB packages are loaded, reboot the router.

Under the System menu, you will now see the *Mount Points* item has been added to the LuCI web interface. This option allows for easy addition and removal of portable USB drives (Figure 4).

Adding USB drives to a router opens up the possibility of a number of interesting projects, such as a SAMBA file/print server, an FTP server, or a Network File System (NFS).

USB Webcam Project

For a fun router project, you can connect a USB webcam and start a video-streaming service. A number of excellent USB video solutions are available, but you need to ensure that the router's small memory size can accommodate the video package and all its dependencies. A good lightweight USB video option is *mjpg-streamer*, which can be installed with the code in Listing 2.

Once you've installed *mjpg-streamer*, you need to start the video service:

```
## to start the service:
/etc/init.d/mjpg-streamer start
## to enable the service to start on
boot
/etc/init.d/mjpg-streamer enable
```

You can access the USB webcam as a web page from the router's IP with the default port of 8080 (Figure 5).

USB-Serial Connections

A router doesn't have external General Purpose Input/Output (GPIO) pins like a Raspberry Pi or an Arduino, but USB ports can be used to pass data.

Because there aren't a lot of USB sensors available, a good workaround to this problem is to use an older microcontroller. Low-end modules such as the Arduino Nano, littleBits Arduino Bit, or a BBC micro:bit can be directly connected to sensors, and the data can be passed with the USB-to-Serial interface.

To start the video service, you need to enable USB-to-Serial communications in OpenWRT:

```
## add USB-Serial packages
opkg install kmod-usb-serial kmod-usb-acm
## add terminal config package
opkg install coreutils-stty
```

For our project, we used a BBC micro:bit [4], which is very user-friendly for coding: It only took five blocks to send the on-board temperature reading and to show the value on the front panel (Figure 6).

The OpenWRT firmware runs a light version of the Bash shell called Ash. The Ash script in Listing 3 connects to the micro:bit USB port and prints out the temperature data.

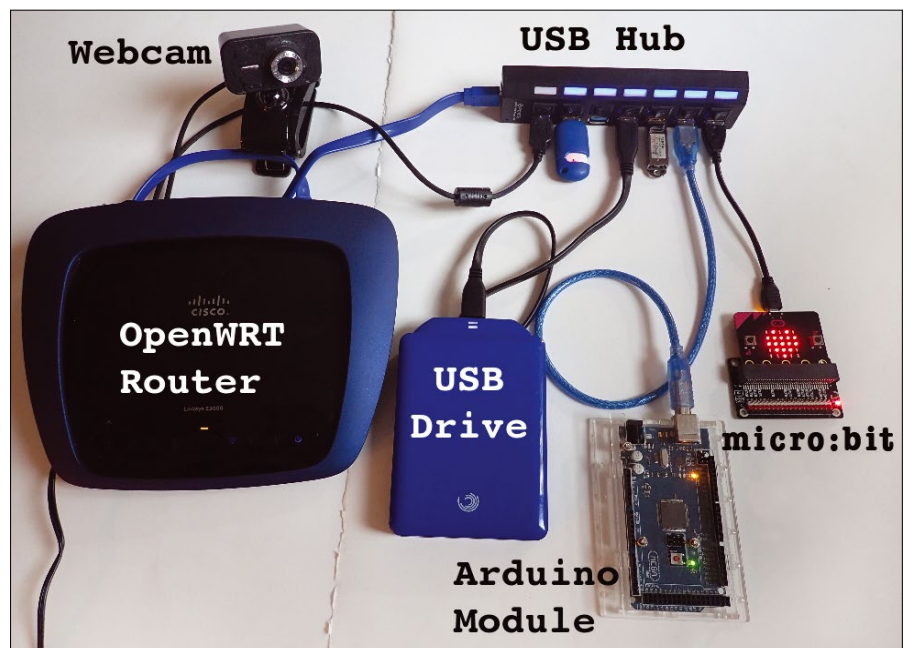


Figure 2: Our router test setup includes a USB hub and web cam.

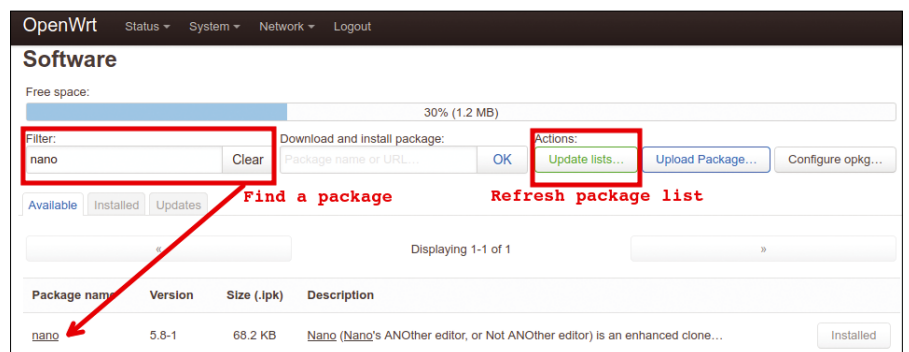


Figure 3: You can use the LuCI interface to find and install OpenWRT packages.

Listing 1: Adding USB Drives Remotely

```
## update opkg is required to get info on new packages
opkg update
## get packages for USB2/3 and ext4 type devices
opkg install block-mount e2fsprogs kmod-fs-ext4 kmod-usb-storage kmod-usb2 kmod-usb3
```

Viewing Router Data

Next, we want to create a simple Common Gateway Interface (CGI) Ash web page to view custom router data and pass the data to another device, such as a Raspberry Pi Node-RED server (Figure 7).

For this project, it's best to use dynamic data. If you don't have external data, then dynamic data from

the router's system load data can be used (Figure 8). The command-line statement

```
cat /proc/loadavg
```

will show the router's one-, five-, and 15-minute load averages.

By adding some AWK code, you can extract each of the data points directly.

For example, to get the first data value, enter the following code:

```
## Show router load averages
cat /proc/loadavg
0.36 0.28 0.16 1/50 3307
## get the first data point
cat /proc/loadavg | awk '{print $1}'
0.36
```

Custom Router Web Page

OpenWRT runs the uHTTPd web server for its router configuration. This web server can also be used for custom CGI web pages that can use Ash scripting. You will find the OpenWRT custom CGI pages in the `/www/cgi-bin` directory. Listing 4 shows an example CGI page, `test.cgi`. This example shows the previous load average values along with some system summary information from the Linux monitoring tool `vmstat`.

CGI web pages use Ash/Bash echo statements to output HTML code. It is important to start the page by echoing out "Content-type: text/html" with an added new line (lines 6-7). For this example, including HTML

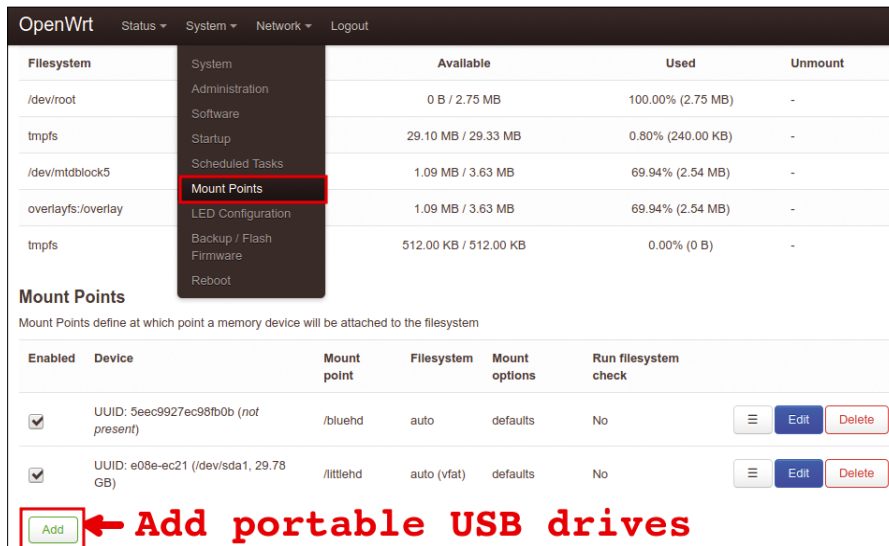


Figure 4: The LuCI web interface lets you mount portable USB drives.

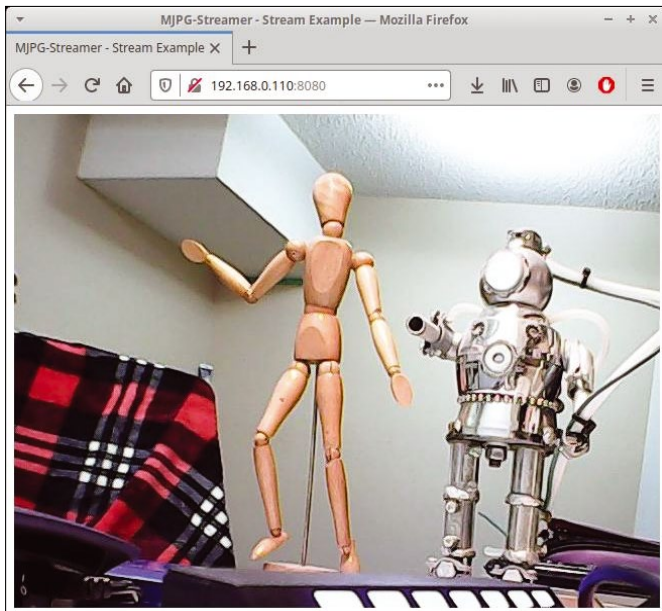


Figure 5: Use port 8080 on your router's IP to view the USB webcam.

Listing 2: Installing mjpg-streamer

```
## install video streaming software and nano editor
opkg install kmod-video-uvic mjpg-streamer nano
## enable video service
## edit config file, and set "option enable '1'"
nano /etc/config/mjpg-streamer
```

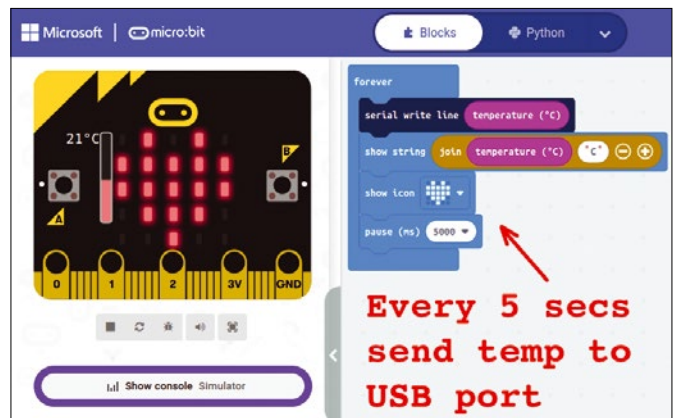


Figure 6: The micro:bit code for USB-to-Serial communications can be completed in five blocks.

Listing 3: Reading USB-Serial Data

```
#!/bin/ash
#
# microbit.sh - reads microbit temperature
#
# set terminal speed
stty -F /dev/ttyACMO 115200
# read USB-Serial device (/dev/ttyACMO)
while read -r line < /dev/ttyACMO; do
    echo " Temp: $line DegC"
done
```


heading tags such as `<h2>` in the echo string improves the presentation (lines 16 and 25).

The output from Ash/Bash statements such as

```
cat /proc/loadavg |awk '{print $1}'
```

will be shown directly on the web page.

Using the HTML `<pre>` tag provides a pre-formatted fixed format for the out-

put from the `vmstat` monitoring utility (lines 25-27).

After creating the CGI page, the final step involves setting the file's execution rights as follows:

```
chmod +x test.cgi
```

You can now access your custom web page at http://router_ip/cgi-bin/test.cgi. Figure 9 shows the test CGI web page.

Connecting to a Raspberry Pi

For our final project, we want to pass the data from the router to a Raspberry Pi. You could modify this project to pass data to a Home Assistant node or any PC on your home network.

The simplest protocol for passing data is to use TCP sockets. This approach doesn't require any added software to be loaded on either the router or on the Rasp Pi.

You can use the Bash `nc` (or `netcat`) to both send and receive TCP messages. To create an `nc` test, open two terminal windows: one on the router and the other on the Raspberry Pi (Figure 10).

To set up a listener on the Rasp Pi, define the Rasp Pi's IP address with a port (1234 in this example). The `-l` option sets listening mode, and the `-k` option will keep the connection open for multiple messages.

On the router sender side, an echo message is piped to the `nc` command with the Rasp Pi's IP address and port.

Next, you need to periodically send dynamic data out via TCP. Listing 5 shows an Ash script file that uses our earlier Ash/AWK code to get the router's

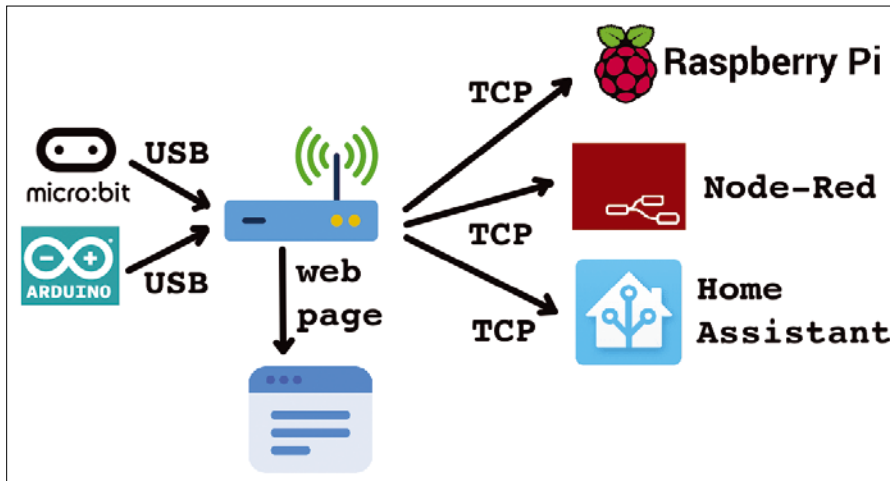


Figure 7: The setup for passing router data to other devices.

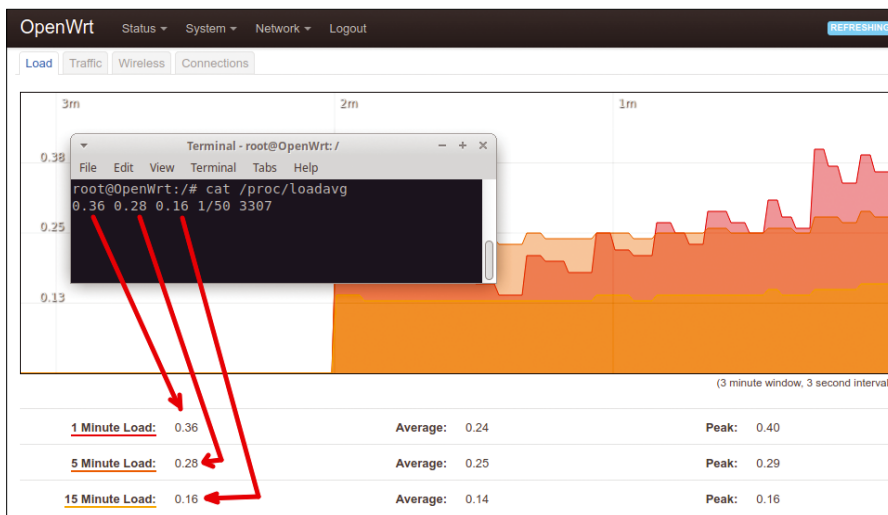


Figure 8: The router's system load data provides the dynamic data for our project.

Listing 4: Router Web CGI Example

```
01 #!/bin/ash
02 #
03 # test.cgi - show system load averages and vmstat
04 #
05
06 echo "Content-type: text/html"
07 echo ""
08 echo "<!DOCTYPE html>"
09 <html>
10 <head><title>Router Points</title>
11 </head>
12 <body>
13 <h1>Router CGI Page</h1><hr>"
14
15 # -- show router system load averages --
16 echo "<h2> System Load Averages </h2>"
17 echo "1 minute load:"
18 cat /proc/loadavg |awk '{print $1}'
19 echo "<br>5 minute load:"
20 cat /proc/loadavg |awk '{print $2}'
21 echo "<br>15 minute load:"
22 cat /proc/loadavg |awk '{print $3}'
23
24 # -- show vmstat -- use <pre> formatting
25 echo "<h2> Vmstat Results </h2> <pre>"
26 vmstat
27 echo "</pre></body></html>"
```

load averages and then pipes the values to a TCP listener every two seconds.

Node-RED [5] is a great visual programming environment that comes preinstalled on the Raspberry Pi Raspbian

image. To get TCP messages from the script in Listing 5 into Node-RED, two tcp in nodes can be configured with the required port numbers. To show the data graphically, two dashboard ui_gauge

nodes can be connected to the outputs of the tcp in nodes. Figure 11 shows the Node-RED logic and web dashboard for the two router load average points.

Un-Bricking a Router

When we started reflashing and reconfiguring our router for these

projects, we made lots of mistakes. We locked up or “bricked” our routers about a dozen times.

If you’ve made a simple mistake, often all you need to do is reset your router and then connect directly to a LAN port to redo your configuration. If this fails, check the OpenWRT blog for any recommendations for your specific router model. There are some excellent custom solutions such as nmrpf1ash [6] for Netgear routers, which offers an almost 100 percent guaranteed un-bricking solution.

If resetting the router doesn’t work and there are no custom solutions, then the next step is the 30-30-30 Hard Reset rule. The following will work for almost all routers:

- Press the reset button for 30 seconds
- While pressing the reset button, unplug the router for another 30 seconds
- Plug the router back in while still holding the reset button for a final 30 seconds
- Release the reset button, and try to reconfigure

Unfortunately, there are cases where even the 30-30-30 Hard Reset rule won’t unbrick a router. This happened to us when we loaded an incorrect firmware version.

Summary

In our tasks for these projects, we found that using shell scripting in Ash rather than Bash wasn’t an issue. However if you are moving code between OpenWRT and Raspbian, you’ll need to toggle between `#!/bin/ash` and `#!/bin/bash`.

If you would rather use MQTT instead of TCP to pass data, the Mosquitto sub/pub command-line tools can be installed on the router using `opkg`.

Overall, we would recommend repurposing an old home router. It offers a lot of interesting projects with a small price tag. ■■■

Info

- [1] OpenWRT: <https://openwrt.org/>
- [2] DD-WRT: <https://dd-wrt.com/>
- [3] opkg: <https://openwrt.org/docs/guide-user/additional-software/opkg>
- [4] BBC micro:bit: <https://microbit.org/>
- [5] Node-RED: <https://nodered.org/>
- [6] nmrpf1ash: <https://github.com/jclehner/nmrpf1ash>

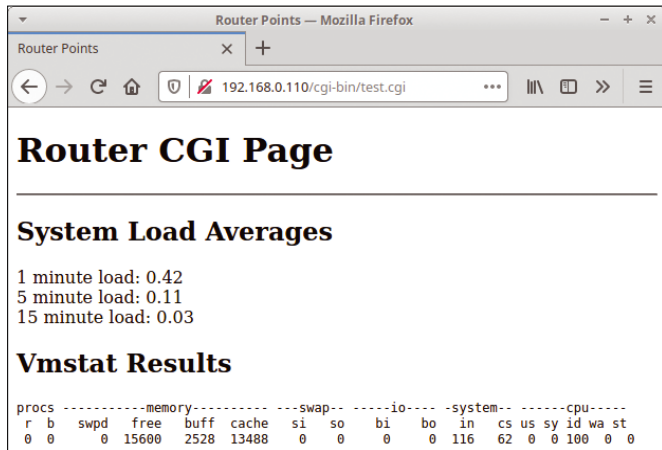


Figure 9: The custom router web CGI page.

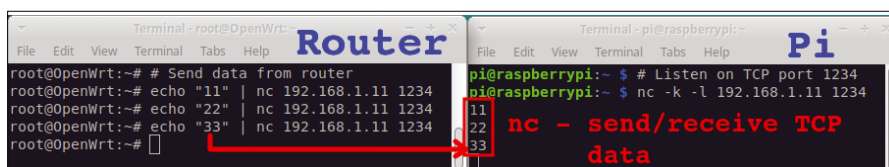


Figure 10: Use nc to send and receive TCP messages.

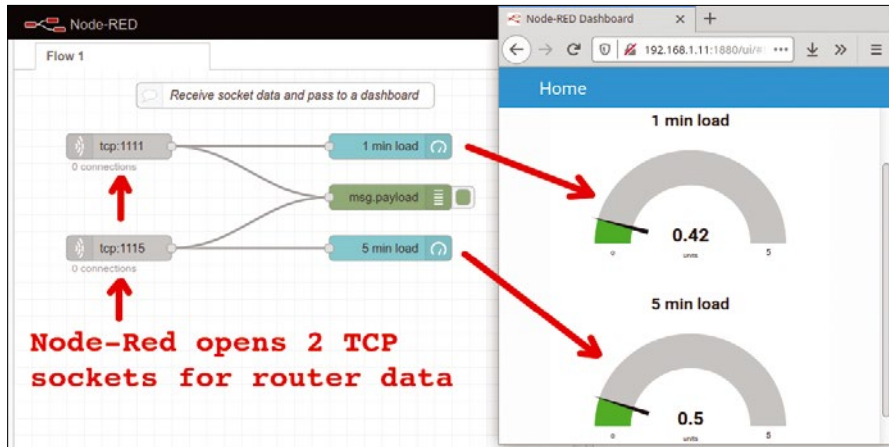


Figure 11: Node-RED dashboard logic with TCP sockets.

Listing 5: Sending Router Data to a TCP Socket

```
#!/bin/ash
# send_loadavg.sh - Send Router Load Averages via TCP
# - send 1 min avg to port 1111
# - send 5 min avg to port 1115
while true
do
    cat /proc/loadavg | awk '{printf $1}' | nc 192.168.1.11 1111
    cat /proc/loadavg | awk '{printf $2}' | nc 192.168.1.11 1115
    sleep 2
done
```


Linux has taken heat in the past for what opponents perceive as a dearth of viable image processing tools.

But as you know if you've read this magazine, we would never accept that characterization for the powerful and freewheeling Linux. The Linux community has lots of great tools for photo enthusiasts, and, unlike the closed-source equivalents, all the best stuff is totally free. In this month's LinuxVoice, we explore the G'MIC digital image processing framework. The developers refer to G'MIC as "...the open-source interpreter of the G'MIC language, a script-based programming language dedicated to the design of possibly complex image processing pipelines and operators." We'll show you how to use the versatile G'MIC to balance, brighten, and sharpen your images – or add hundreds of other filters and special effects. And while we're on the subject of images: When you're ready for the third dimension, check out our tutorial on Blender 3D animation.



Image © Olexandr Moroz, 123RF.com

LINUXVOICE

Doghouse – Multi-factor Authentication 67

Jon "maddog" Hall

As an alternative to passwords, maddog looks at various types of multi-factor authentication.

Worker 68

Karsten Günther

Worker, a file manager with more than 20 years of development, has evolved into a tested, powerful, and functional tool.

Clapper and GTK4 74

Christoph Langner

The Clapper media player showcases new desktop design features in GTK4.

G'MIC 76

Dmitri Popov

Behind G'MIC's deceptively simple interface hides a mighty image processing framework.

FOSSPicks 80

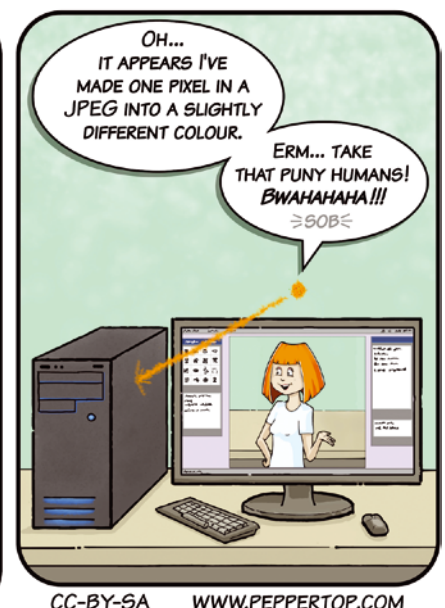
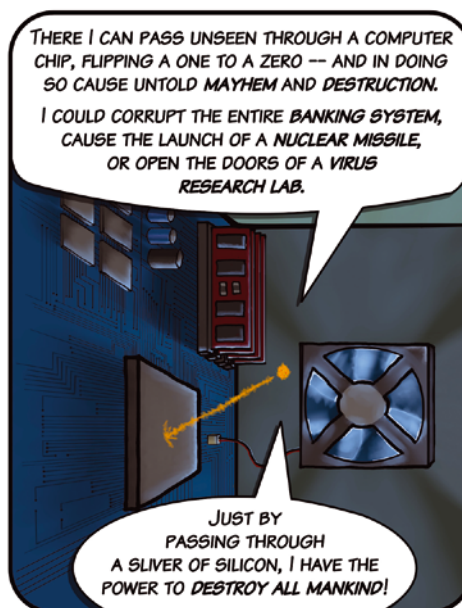
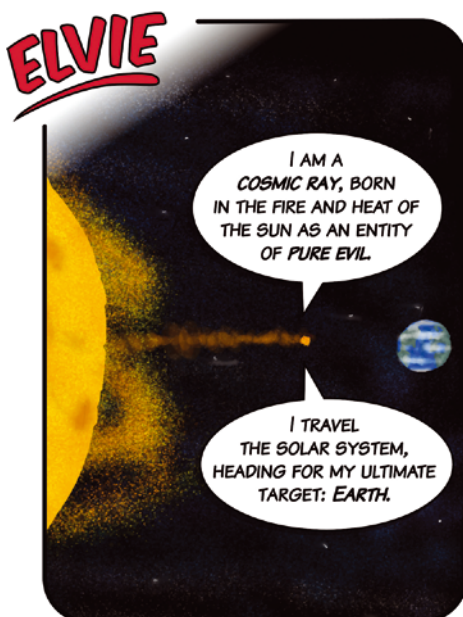
Graham Morrison

This month Graham looks at Bespoke, Waydroid, OpenShot, and more!

Tutorial – Blender 88

Claus Cyryny

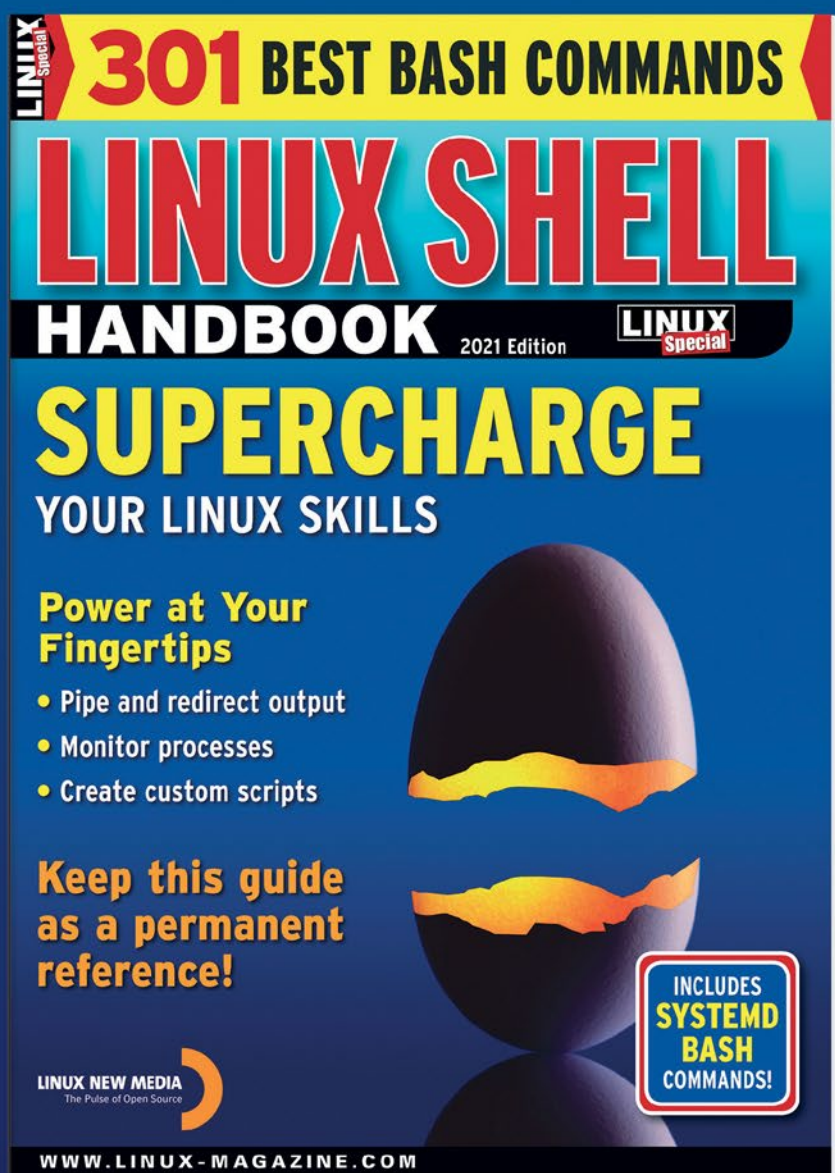
Blender's massive feature set can seem overwhelming at first. Choosing a manageable project can help you get started.



THINK LIKE THE EXPERTS

Linux Shell Handbook 2021 Edition

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



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

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

ORDER ONLINE:

shop.linuxnewmedia.com/specials

MADDOG'S DOGHOUSE



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

As an alternative to passwords, maddog looks at various types of multi-factor authentication, as well as considerations drawn from his experience. BY JON "MADDOG" HALL

Multi-Factor Authentication for Login Security

Recently a large, closed source software company announced their operating system would allow the user to opt out of using passwords. They indicated that passwords were difficult to manage (agreed), and many times people forget them or use the same passwords for many accounts (which many people do), so now users will be given the ability to use multi-factor authentication (MFA) to avoid using passwords and instead use some other authentication methods to protect themselves. Sounds great ... on the surface.

I already know of people that are using their phones to do MFA. When you log in to some web service for the first time during a login session, a message gets sent to your smartphone to acknowledge that someone is trying to log on to your account and to verify that the person is you.

However, using your smartphone has some issues.

You may not own a smartphone. Many of my friends are (cough) "older" and only have "burner" phones (also known as flip phones) that cannot run applications. Of course, many burners can receive SMS messages and be verified through that. However, MFA using phones puts an extra importance on phones being available all the time. If the phone is unavailable (discharged, lost, stolen), in an area where phones are not allowed (secure areas), or a cell phone signal is not available, then a person might inadvertently be locked out of their accounts.

Important to know is that most of these MFA techniques do not rely on the phone as much as they rely on the International Mobile Subscriber Identity (IMSI) number that is assigned to your SIM card. If your phone breaks down, you can simply take the SIM card out and put it into another phone. If the SIM card is lost, you can report it to the mobile phone company and get a replacement SIM card that will have the same phone number (IMSI) associated. However it may take some time to get a replacement SIM and put it in a new phone.

Another way of doing MFA is using a type of "key" that is available from various companies. These keys (usually small enough to fit on a keychain) are inserted into the USB port of your laptop or phone and/or use NFC to connect with a device as you try to access your accounts (including your login account). Various operating systems as well as various web browsers and cloud-based applications allow these keys to be

part of their MFA. Some of these keys are fairly expensive. While this expense may be easily justified from a business perspective, the average person may not want to pay for two (one to use and one to be kept in a secure place as a backup). Of course these keys may be lost or stolen like a phone – therefore requiring a backup key or other MFA path.

Other key types are "smart card"-type devices, which use either contact (needs to be inserted or otherwise scanned) or contact-less NFC technology to verify that the user is physically present. Sometimes these cards have storage on them that can hold details such as health or financial information. Typically these cards are associated with a personal identification number (PIN) to help protect them if lost or stolen. Again, these cards and the management of them can be fairly expensive, and the cards can be damaged relatively easily in adverse environments.

My laptop has both a webcam built in and a fingerprint reader. While both facial recognition and fingerprint recognition have security issues by themselves, when you put them together along with the physical access to a particular device (the laptop, for instance), they can create a much more secure system for logging into that device.

All of these methods, and more, can be used for MFA. One of the problems is, will the user use them? And how complex will it become for people to actually access their systems and data?

A recent webinar on password-less logins stated: "Join Cybersecurity experts ... to discuss why users will be more likely to adhere to security best practices if they are empowered to manage and renew their credentials without your IT team's help."

Right. I remember how much users hated even simple passwords to log in to their systems. The more complicated the system was, the more they needed help. People who need help in adding an application to their smartphone are going to have some issues in setting up MFA to work across their various devices, various websites, and various applications.

FOSSH has the tools (MFA, PAM, SELinux or AppArmor, encryption of filesystems and data, among others) to do this well. It is time to start planning how to use MFA in your community or business. ■■■

A proven file manager Workhorse

Worker, a file manager with more than 20 years of development, has evolved into a tested, powerful, and functional tool. **BY KARSTEN GÜNTHER**

Two-pane file managers have been around for a long time. Classic examples include Norton Commander, Total Commander, DiskMaster, and Directory Opus. Created in the same vein, Worker [1], a two-pane file manager for the X Window System, has been actively developed since 1998. Many of Worker's features were developed to meet practical needs, resulting in a functional file and directory management tool.

You can find Worker as *worker* or *workerfm* in the package sources of most common distributions. On Arch Linux and derivatives, the program is available via the Arch User Repository. However, the Worker package found in your distribution's repository may not be the latest 4.9.x version. Using the latest version of the program typically requires some DIY work. Having said

this, Worker has only minimal dependencies (see Table 1) and can therefore be compiled on almost any system without too much trouble. After downloading the source code [2] and unpacking it, use the code in Listing 1 to compile Worker

Getting Started

When first launched, Worker checks for an existing configuration before opening the main window; an existing configuration can be from an older version of the application. If needed, the start routine will update the configuration (Figure 1).

At first glance, Worker's structure appears almost classic (Figure 2). The two panes display directories as lists (see the "Display Modes" box); either pane can function as the source or target, with a red status bar denoting the active pane. The file

Listing 1: Compiling Worker

```
$ wget http://www.boomerangsworld.de/cms/worker/downloads/worker-VERSION.tar.bz2
$ tar xf worker-VERSION.tar.bz2
$ cd worker-VERSION
$ ./configure && make && make install
```

Table 1: Worker Dependencies

Package	Function
<i>gcc</i>	C++ compiler with C++14 support, GCC >= version 4.9 recommended
<i>libX11-devel, libx11-dev</i>	X11 headers and libraries
<i>avfs</i>	Open archives, FTP access, etc.
<i>libdbus-glib-1-dev, dbus-1-glib-devel, udisks, udisks2</i>	Disk access
Optional Dependencies	
<i>libmagic-dev, file-devel</i>	Display image previews
<i>xli</i>	Default image viewer can be changed in the config file (see the "Configuration Files" section for more information)
<i>libxinerama-dev, libXinerama-devel</i>	Improved placement of windows
<i>lua-devel, liblua5.x-dev</i>	Lua scripting
<i>libxft-dev, libXft-devel</i>	Improved font display

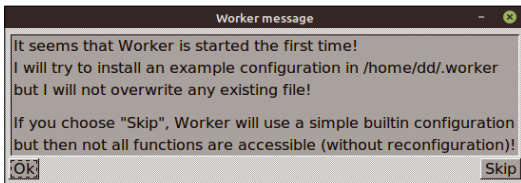


Figure 1: Worker starts with a small dialog.

manager functions can be found in the buttons at the bottom of the window and in context menus. Alternatively, you can bind commands to a double-click or call them up via keyboard shortcuts.

Worker's long development history has resulted in a very compact interface. Because Worker is not necessarily intuitive, you need to become acquainted with Worker's controls and technical functions before getting started.

In the red status bar above the current pane, Worker shows the latest information for the directory. The list display in the current pane depends on the display mode and directory contents. To switch between display modes, click on the small triangles in the upper right corner of the status bar.

Worker organizes entries in the lists in tabs, which allows for lightning-fast switching between multiple directories in both the source and target panes. The small plus sign in the upper left corner above a pane opens a new tab. Clicking on one of the active tabs switches to the directory.

The selected directory can be found a second time below the target or source pane, where the tool displays the directory in text form. To the left of this text display, a button with two dots lets you switch to the parent directory. At the bottom of the main window, you will find numerous buttons organized into several groups. These button groups are called a "bank" in Worker jargon. The button banks (see Table 2) control the main functions [3].

You can toggle between the banks with the mouse wheel while mousing over the lower status bar. Alternatively, you can use the left and right

Display Modes

When you open Worker, the default appearance is Directory mode. To change the display mode, right-click on the status bar. In the dialog that opens, you can also choose from the following display modes:

- Image Display: Shows graphics instead of lists
- Information: Provides additional detailed information from the filesystem, such as access rights, owners, and metadata
- Text Display: Displays the data like in a text editor

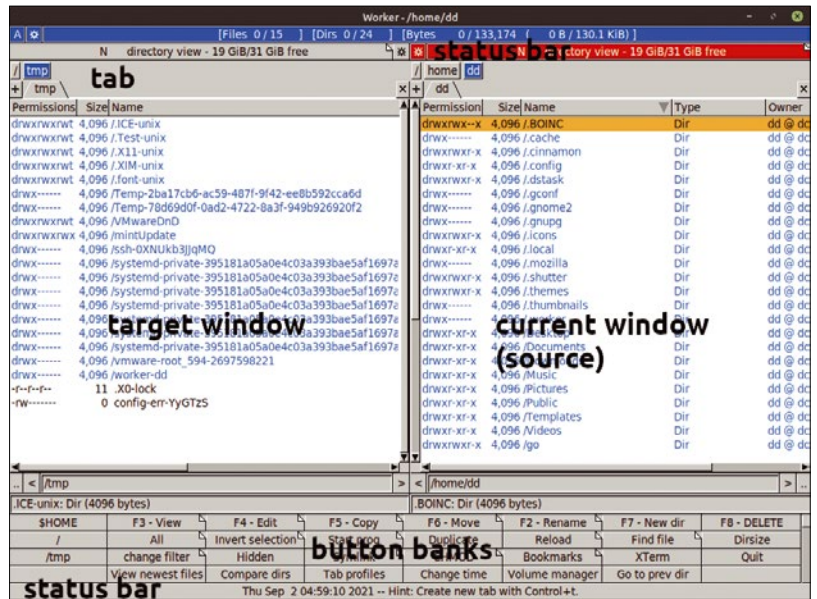


Figure 2: Worker's main window displays files in target and source panes, along with status bars and several functions in the buttonbar below.

mouse buttons. If necessary, add your own functions to the groups. Each button can call either an internal function, a shell command, a shell script, or an arbitrary program.

Worker always applies the selected functions to all currently selected entries, which occasionally leads to situations where several procedures are possible. If this happens, Worker will ask how you want to proceed (Figure 3).

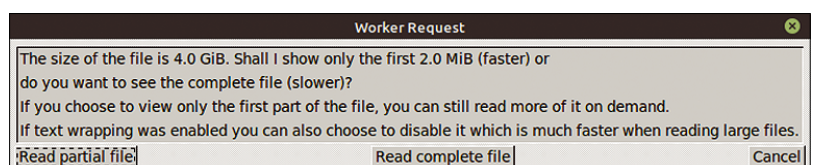
You always have the option to trigger almost all actions either with the keyboard or the mouse. In addition, you can customize almost all aspects of the software, starting with the buttons and extending to the list panes, the keyboard shortcuts, and menu entries.

Depending on the file type, Worker changes the commands provided for the files. You then execute the command by either double-clicking or pressing the Enter key. To accomplish these tasks, Worker uses a variety of external programs. As a

Table 2: Worker Button Banks

Bank	Function
1	Basic functions for files and directories
2	Basic functions for archives
3	Special functions for display, as well as for symlinks
4	Basic Git functions
5	Advanced functions for special file types
6	Other functions

Figure 3: Depending on the situation, Worker asks which function you want to execute.



result of Worker's long development history, several outdated or atypical applications have crept in to this program list.

To find out which external programs are available, search the configuration files for the string:

```
com =
```

Make sure to include a space before and after the equal sign. For example, Worker relies on Gimp and the ImageMagick tools for image editing; McEdit, Xeditor, LyX, soffice, and others for text editing; and Netscape for browsing.

In some instances, you can make life easier for yourself by resorting to *xdg-utils* [4] (XDG is a specification developed by *freedesktop.org* – formerly the X Desktop Group – for interoperability between different desktop environments). XDG commands such as *xdg-open* let you call the tools already installed on your system that you have preset for opening files of certain type.

Table 3: Worker Shortkeys Access

Shortcut	Function
Tab	Activate other list pane
Up arrow/Down arrow	Go to previous/next entry
Left arrow	Move up one directory level
Right arrow	Enter current directory
Home/End	Activate first/last entry
Ins	Toggle activity
+ (number pad)	Select everything in the current tab
- (number pad)	Deselect everything in the current tab
Enter	Perform double-click action
F3	Show current entry
F4	Edit current entry
F5	Copy selected entries
F6	Move selected entries
F7	Create new directory
F8	Delete selected entries
Shift+Alt+O	Open current directory in opposite list pane
Alt+V	Start volume manager
Alt+B	Open/add bookmarks
Alt+L	Manage labels
Alt+N	Sort entries by name
Alt+T	Sort entries by modification times
Ctrl+D	Show bookmark entries
Ctrl+S	Activate name search in current list pane
Ctrl+Space	Open context menu
Ctrl+I	Activate information mode
Ctrl+B	Activate image display mode
Ctrl+V	Activate text display mode

If you launch Worker in a terminal window, you will see additional hints there if something unexpected happens – for example, if a function doesn't work as expected.

Basic Functions

Worker was originally designed to handle files and directories in pretty much any way imaginable. You will find the corresponding functions mainly in the button banks below the file panes. Worker organizes the buttons in a grid, again grouping the functions both by row and by column.

At least as useful as the buttons, the keyboard shortcuts, or shortkeys, provide quick access to functions that are repeatedly used (see Table 3). The *ShortcutList* function in bank 4 gives you a complete list of defined shortkeys. Many of the combinations used with Worker are based on Midnight Commander.

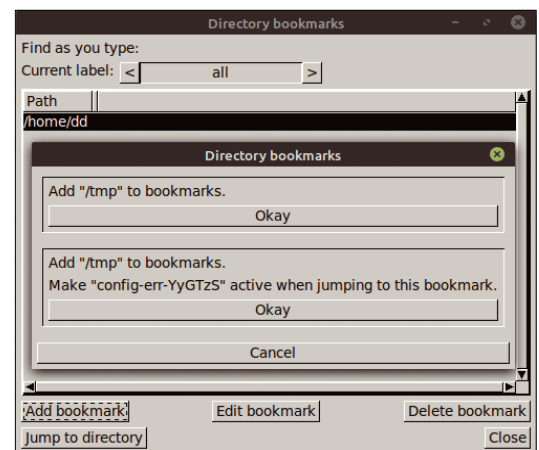
Important (i.e., frequently used) directories can be saved as bookmarks. In addition, you can label selected entries. To create a bookmark, select the current directory with Alt+B or with the *Bookmarks* button. In the dialog that opens (Figure 4), you can choose between previously defined bookmarks and change directly to the corresponding directory or add the current directory as a new entry.

In addition to the directory bookmarks, you can use labels for files. You will find the label function in the context menu when you right-click on an entry. You can manage labels with bookmarks. When you open bookmarks, you will also see the labels. If you select a label, the program switches directly to the corresponding directory and highlights the labelled file.

Filters and Patterns

To speed up many tasks, you can filter bookmarks and labels. Ctrl+D shows you only the entries with a label in the file list, providing a useful overview even in large directories. If you call this

Figure 4: You can quickly access specific directories with bookmarks.



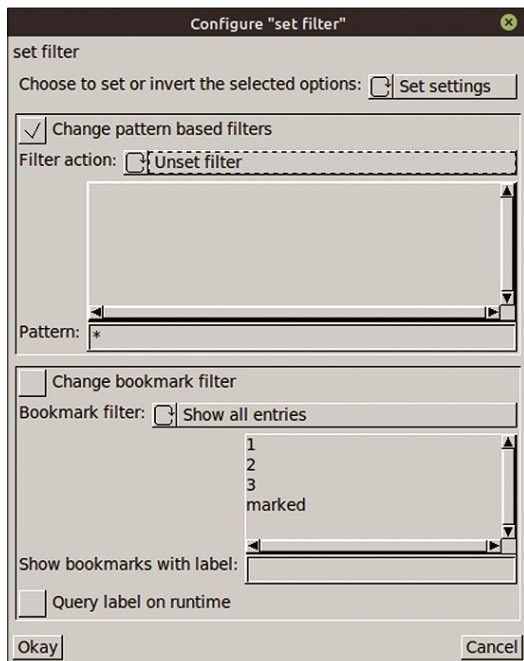


Figure 5: Filters control what you see in a list.

key combination again, Worker will display all the entries again.

Worker offers several ways to control or limit the actual entries displayed in the list. To control how and what Worker displays in the list panes, you use filters and patterns. For example, you may not want to show all files in a directory, such as `/etc/`, which often contains several hundred files and subdirectories. You can also change the sort order of a column by clicking on the respective column header.

Filters restrict what you see in a list. The *Change filter* button lets you define a sequence of characters for file names (Figure 5). There are two varieties of filters: permanent and temporary. *Change filter* lets you create permanent filters, and *Find file* sets temporary filters.

The method *Find file* is particularly sophisticated. First, Worker tries to find the entered characters string exactly and incrementally. If this does not result in a hit, a fuzzy search is performed, which allows random characters in between the ones you entered.

In fact, Worker's filter functions go even further: Worker supports the use of Boolean expressions. If a

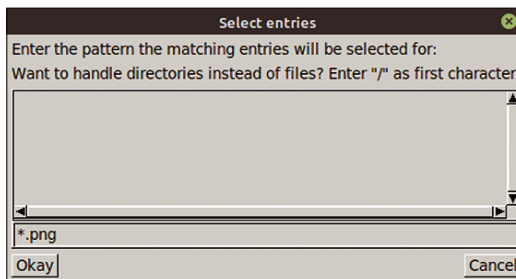


Figure 6: Patterns limit the files or directories displayed in the list.

character string starts with an opening parenthesis, the program interprets everything up to the corresponding closing parenthesis as a logical expression. In the documentation [3], expression matching shows how this is done and which arguments the software assigns to which keywords.

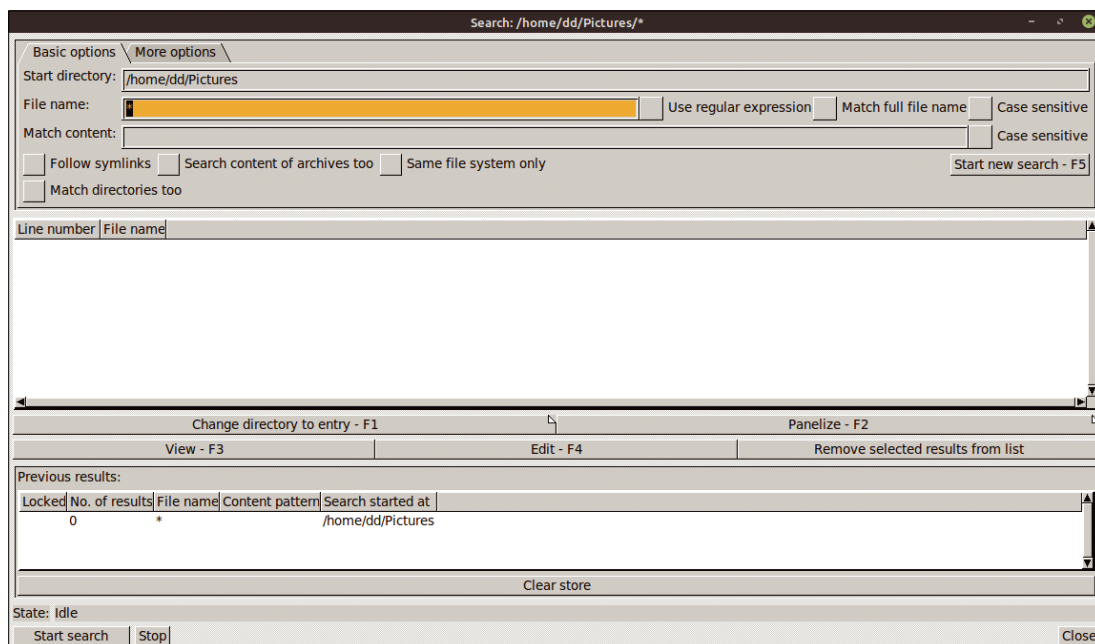
Patterns are closely related to filters. They control the selection of files taking into account certain criteria. You can create patterns with the divide key on your keyboard's number pad. By default, you define file name patterns this way (Figure 6), but directories can also be displayed using patterns. In addition, you can apply several patterns in succession to select multiple types of file names.

Worker recognizes file types based on the file's content. Worker shows the file's MIME type [5] in the *Type* column. You can also use the MIME type for filtering and patterns [6].

Advanced Functions

Worker supports searching for files and directories in a sophisticated way. An extensive dialog shows the corresponding parameters (Figure 7). Worker saves the results to display them again later if needed. The results are saved until you ex-

Figure 7: The Search dialog is largely self-explanatory but comes with a number of special features.



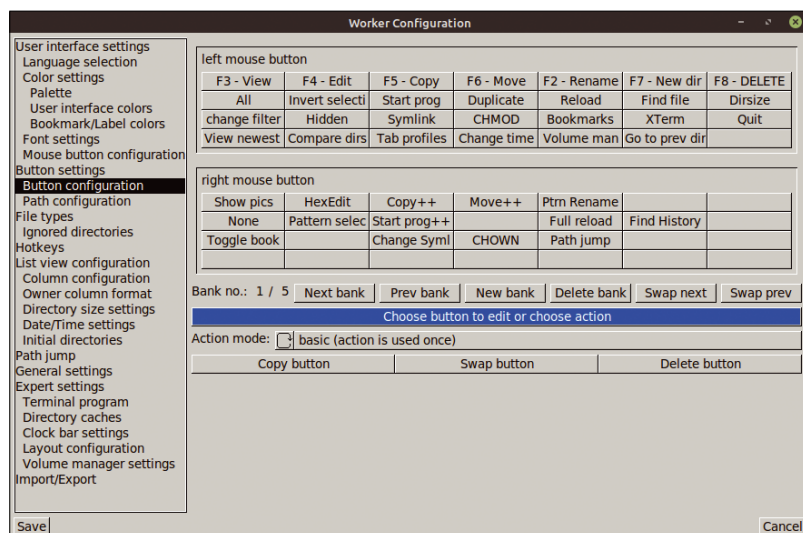


Figure 8: Worker's configuration options are extensive and affect both the appearance and the functionality of many elements.

explicitly delete the old results using F5 or select *Start new search*.

Worker follows the principle of refined searching, which means that you always search in the previous results until you find a perfect match. In addition to the *Basic options* displayed by default, *More options* offers further choices for narrowing down the results.

By default, pressing the Enter key switches to the next input field but does not start the search. To start the search immediately, end the input in a field by pressing Ctrl+Enter. Worker shows the results in the corresponding field at the center of the window. You can apply many functions to the matches. For example, you can transfer the results to lists or display them [7].

Configuration Files

Worker stores its configuration files in the `.worker` folder in your home directory. At more than 200KB, these configuration files are large. You should only edit them manually if you know what your changes will do. Direct editing is simple and clear-cut if you are just replacing individual commands, such as replacing the GView image viewer with Geeqie. Worker monitors the

Figure 9: The volume manager controls access to additional filesystems.



configuration files and asks what to do when changes are made.

There are several possibilities for making adjustments in various places in a configuration file. If you click on the second small button on the left in the top status bar (just below the window's title bar), Worker will display a summary of the configuration (Figure 8).

Volume Manager

Another special feature in Worker, the volume manager, lets you integrate filesystems and supports both the obsolete hardware abstraction layer (HAL) and UDisks for automatic mounting via D-Bus without leaving the file manager, for example.

Clicking on the *Volume manager* button (or pressing Ctrl+V) opens the appropriate dialog (Figure 9). Double-clicking on one of the entries mounts the corresponding device. Beware, the volume manager can lead to conflicts with the automatic systems often used today.

Conclusions

Granted, Worker doesn't offer a pretty interface. However, more than 20 years of development have resulted in an extremely powerful, functional, and effective tool. Developed directly from hands-on experience, Worker's features have proven track records. If you can overlook Worker's occasional idiosyncrasies, you will find Worker to be a useful and basically simple tool that leaves few wishes unfulfilled. Plus, Worker's useful documentation will help you quickly find your way around. ■■■

Info

- [1] Worker: <http://www.boomerangsworld.de/cms/worker>
- [2] Source code: <http://www.boomerangsworld.de/cms/worker/download.html>
- [3] Functions: <http://www.boomerangsworld.de/cms/worker/documentation/features/filtering.html#h-1-2-2>
- [4] xdg-utils: <https://www.freedesktop.org/wiki/Software/xdg-utils/>
- [5] MIME types: https://en.wikipedia.org/wiki/Media_type
- [6] File types: <http://www.boomerangsworld.de/cms/worker/documentation/features/filetypes.html>
- [7] Search function: <http://www.boomerangsworld.de/cms/worker/documentation/features/filesearch.html>

Shop the Shop shop.linuxnewmedia.com

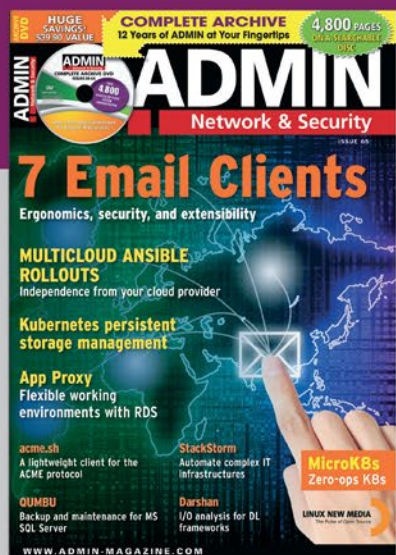
Want to subscribe?

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

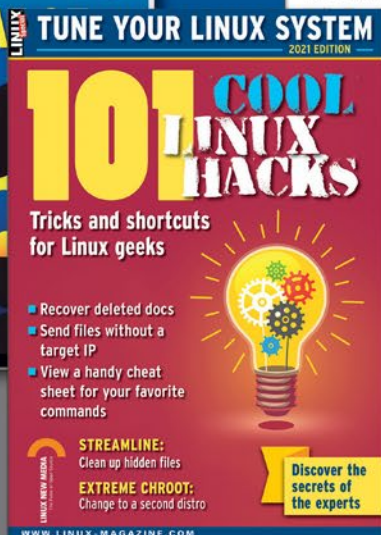
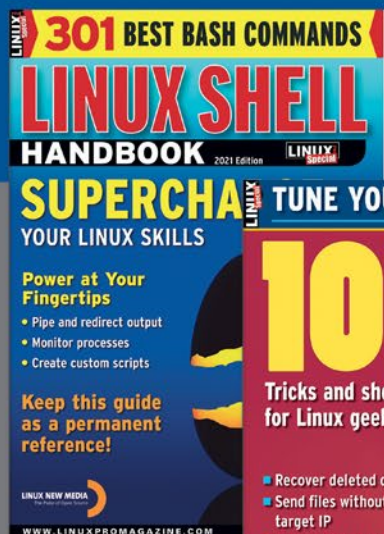
Discover the past and invest in a new year of IT solutions at Linux New Media's online store.

►► shop.linuxnewmedia.com ◀◀

DIGITAL & PRINT SUBSCRIPTIONS



SPECIAL EDITIONS



Desktop design with GTK4

And ... Action!

The Clapper media player showcases new desktop design features in GTK4.

BY CHRISTOPH LANGNER

Free software desktops such as Gnome and KDE are built on top of toolkits that provide support for the graphic elements. Gnome is built on the GTK toolkit. GTK is an important component of many GUI-based Linux systems, and a new release of GTK eventually has implications for all Linux users working in Gnome and other GTK-based desktop systems.

GTK4 [1] was released at the end of 2020, and components that rely on GTK4 have quickly followed, starting with Gnome 40 in March 2021. Recent distributions such as Fedora 34 [2] and Arch Linux already include Gnome 40. Other distributions such as Ubuntu 21.04 [3] include the GTK4 libraries in their package sources but don't use the GTK4-based desktop yet.

In practice, however, GTK4's influence in these distributions is still not very noticeable. For example, the Gnome desktop's extensions app does not look noticeably different, even though it has now been ported to GTK4: It uses the usual widgets, such as buttons, search fields, and fold-out dialogs. If you look at the Widget Factory (see the "Widget Factory" box), a test program that organizes all common widgets in a window, you probably won't notice any outstanding innovations at first glance. However, if you take a look at the animated GTK cube video on the main Widget Factory page, you can see one of the biggest changes between GTK3 and GTK4. Mousing over the video

reveals the typical pause and start playback media player buttons. This cube video demonstrates that you now can put graphics or other media elements in the background of almost all window elements [4].

As a means of exploring some of the new features in GTK4, this article examines the new Clapper [5] media player, a Gnome desktop tool built using the GTK4 toolkit.

Clapper

The Clapper media player was built using GTK4, so it shows how some of the new GTK4 features work in practice. At first glance, Clapper reminds you of Gnome Video Player, but you will notice that the classic window bar located at the top of the application window is missing (in fact, this hasn't existed in Gnome for some time). GTK4 has pushed the envelope a bit further: The video image covers almost the entire window area, with only the control bar at the bottom remaining permanently visible. The menus can be reached via partially transparent buttons at the top of the window (Figure 1). Clapper only shows the buttons when you mouse over the window.

Clapper has three modes: windowed, full-screen, and floating. The default, windowed mode shows the progress bar and window controls. In windowed mode, Clapper is visible on every virtual

Widget Factory

To import the Widget Factory under Ubuntu 21.04 or Debian Experimental, use the *gtk-3-examples* and *gtk-4-examples* packages, respectively. (For Arch Linux, you'll find the example widgets in the *gtk3-demos* and *gtk4-demos* packages).

Once you've installed the packages for your distribution, you can call the programs using the *gtk3-widget-factory* and *gtk4-widget-factory* commands.

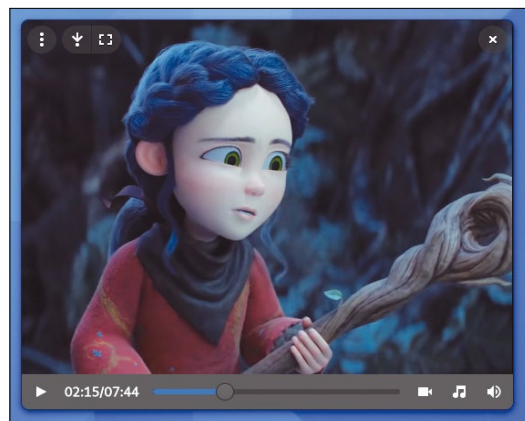


Figure 1: Clapper reduces the space required by the playback controls to the bare minimum.

Listing 1: Installation on Ubuntu

```

sudo apt install curl

echo 'deb http://download.opensuse.org/repositories/home:/Rafostar/Debian_Unstable/ /' |
sudo tee /etc/apt/sources.list.d/home:Rafostar.list

curl -fsSL https://download.opensuse.org/repositories/home:Rafostar/Debian_Unstable/Release.key |
gpg --dearmor | sudo tee /etc/apt/trusted.gpg.d/home_Rafostar.gpg > /dev/null

sudo apt update

sudo apt install clapper

```



Figure 2: Clapper in floating mode with the controls hidden.

desktop on Gnome and will automatically slide from desktop to desktop, remaining in the foreground even when another application is active. In full-screen mode, you can select to hide the progress bar and playback controls. In floating mode (Figure 2), Clapper hides the controls and floats the window on top of other applications; the application window will automatically slide to the currently active desktop in this mode.

Due to its early stage of development, Clapper is not yet available in the package sources of the popular distributions. However, you can find a Flatpak on Clapper's GitHub page [5] that you can install across distributions. Alternatively, the developers provide packages and package sources for Debian,

Fedora, and openSUSE [6]. You can also install the DEB package on Ubuntu 21.04 (Listing 1).

Eye Candy for Gnome

Other application developers are also increasingly adopting the possibilities offered by GTK4. For example, the radio receiver Shortwave [7] offers a program display area that adapts fluidly depending on the display width or desktop window size. All of this is proof of Gnome's continuous development towards optimizing itself and the various Gnome applications for use on smartphones or tablets. ■■■

Info

- [1] GTK4: <https://blog.gtk.org/2020/12/16/gtk-4-0>
- [2] Fedora 34: <https://getfedora.org/en/workstation/download/>
- [3] Ubuntu 21.04: <https://releases.ubuntu.com/21.04/>
- [4] Media in GTK4: <https://blog.gtk.org/2020/05/20/media-in-gtk-4>
- [5] Clapper: <https://github.com/Rafostar/clapper>
- [6] Download Clapper: <https://software.opensuse.org/download.html?project=home%3ARafostar&package=clapper>
- [7] Shortwave: <https://gitlab.gnome.org/World/Shortwave>



A configurable, flexible image processing library

Cornucopia

Behind G'MIC's deceptively simple interface hides a mighty image processing framework. While mastering G'MIC can be a rather daunting proposition, here's how to get started.

BY DMITRI POPOV

GREYC Magic for Image Computing (G'MIC) [1] offers a seemingly limitless library of simple and advanced filters you can apply to images. Filter, in this case, is a bit of a misnomer. Filters are usually associated with presets that modify an image's overall appearance, and normally they don't offer much in terms of customization. G'MIC filters couldn't be more different. Think of G'MIC filters as actual tools that are designed to perform specific image manipulation tasks. Each tool offers a number of configurable options that allow you to achieve the optimal result. More important, unlike traditional filters, G'MIC tools can be used not only to change the overall look of an image but also to perform a multitude of other tasks, from making basic adjustments (brightness, contrast, saturation) to denoising and applying custom CLUT presets.

Objectively, not all of the G'MIC filters are equally useful, but as a photographer or a digital artist, you will likely find plenty of powerful filters in G'MIC's library worth adding to your image processing and manipulation toolbox. In this article, I'll show you how to put G'MIC to practical use.

Installation and First Steps

Although G'MIC is available as a command-line tool and a web service, you will most likely want to use it as a regular graphical application. You'll find the G'MIC plugin for Gimp and Krita in the official repositories of many mainstream Linux distributions. To plug G'MIC into Gimp on Ubuntu, run the command:

```
sudo apt install gmic-gimp
```

Doing the same on openSUSE is equally simple:

```
sudo zypper in gimp-plugin-gmic
```

If you happen to use digiKam, you'll be pleased to learn that the latest version of digiKam comes with G'MIC. However, keep in mind that not all

G'MIC tools work in digiKam. Since digiKam doesn't support layers and masks, the G'MIC tools that require this functionality won't work.

The G'MIC plugins and the digiKam version have the same easy-to-use graphical interface. The interface (Figure 1) is split into three parts: the preview pane to the left, the list of all available tools in the middle, and the options section to the right. G'MIC offers more than 500 filters that are grouped into categories by their functionality, making it easier to explore the library and find the tool you need. In addition, you can use the *Search* field to search for a specific tool by name. For faster access to frequently-used tools, use the *Fave* button to add them to the Faves menu.

Exploring G'MIC

Many G'MIC tools come from external contributors. The sheer number and variety of contributions is a testament to G'MIC's power and flexibility. However, many tools provide no documentation to help you to figure out what certain options do. It's not a big issue for simple tools such as *Basic Adjustments*, especially if you are familiar with image editing fundamentals. But to master more advanced tools, prepare to do some experimenting. The good news is that nothing is applied to the currently opened image until you hit the *OK* button. And even then, you should be able to undo the applied modifications in the host application. When working with G'MIC, keep in mind that the preview doesn't always provide an accurate representation of the applied filter. So it's better to treat the preview as a rough approximation rather than a pixel-perfect rendering.

Before you start exploring G'MIC's advanced functionality, it makes sense to start with simple tools that have only a few adjustable parameters, such as *Colors | Color Balance* and *Colors | Basic Adjustments*. *Color Balance* provides a quick and simple way to adjust white balance by selecting the neutral color, while *Basic Adjustments* lets you adjust brightness, contrast, and saturation. Some

G'MIC tools have overlapping functionality. The *Colors | Auto Balance* tool, for example, is very similar to *Colors | Color Balance*, but it has more adjustable parameters. In other words, G'MIC often offers several paths to achieving the desired result. More important, the basic tools in G'MIC don't merely replicate the functionality available in host applications such as Gimp and digiKam. Instead these tools nicely complement an application's existing features.

It would be impossible to describe everything G'MIC has to offer, so let's focus on a couple of tools that most users will likely find handy. If you are looking for a quick and easy way to liven up your photos, you will appreciate a comprehensive library of ready-made filters tucked under *Colors | Simulate Film* (Figure 2). Here, you will find plenty of effects to play with, from *Black & White* for emulating a wide range of classic black-and-white films to *Instant [Consumer]* for mimicking iconic Polaroid instant films. You can apply each filter as it is, or you can adjust the available parameters first. For example, if the filter is too strong for your taste, you can reduce its strength using the appropriate slider.

If you have your own presets in the Hald CLUT format, you can apply them to the currently opened image with *Colors | Apply External CLUT*. If you don't have Hald CLUT files, G'MIC comes with a tool that allows you to generate Hald CLUT files from existing edits (Figure 3). To use this tool in Gimp, open the source image in Gimp and create a new layer by choosing *Layers | New from Visible*. You should see the new layer called *Visible* in the *Layers* pane. Make sure that this layer is on top and selected,

and then apply the desired color adjustments. Choose *Filters | G'MIC-QT*, and then switch to the *Colors | CLUT from After-Before Layers* tool. Select *All visible* from the *Input layers* drop-down list and *New image* from *Output mode*. You should see a Hald CLUT table in the preview pane. Press *OK* to generate a Hald CLUT file, save the file in PNG format, and you are done. Now you can apply the freshly-baked Hald CLUT file to a photo using *Colors | Apply External CLUT*.

Both digiKam and Gimp offer simple tools for removing unwanted objects from images. But though they can help you to remove dust particles and blemishes, these tools fall short when it comes to removing large and complex objects.

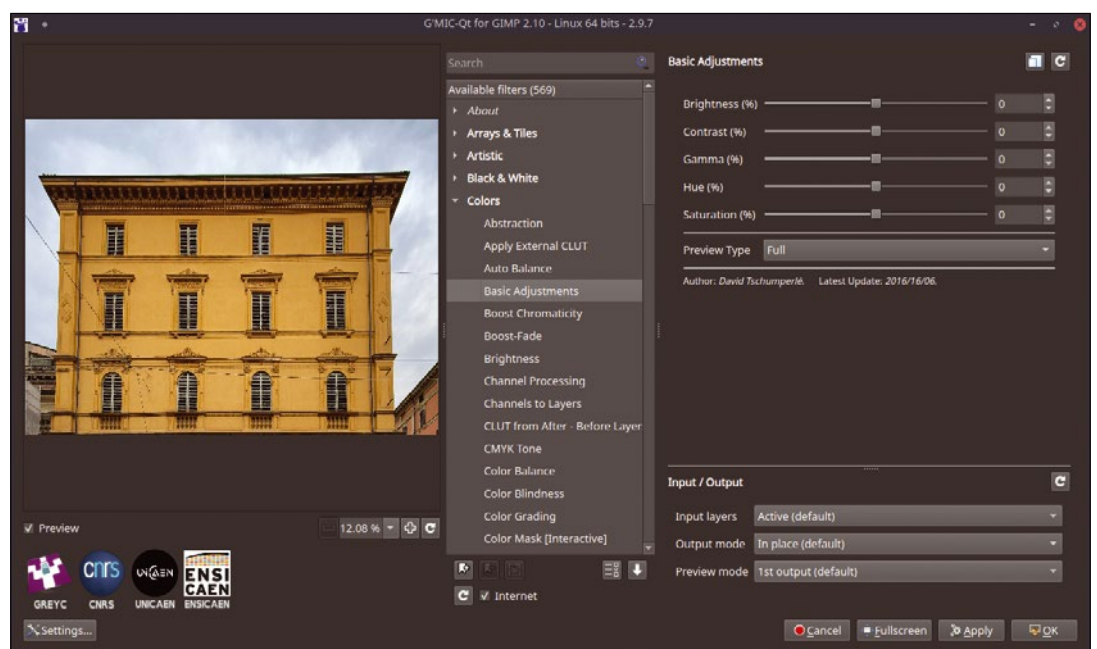


Figure 1: G'MIC features a simple interface putting all its tools at your fingertips.

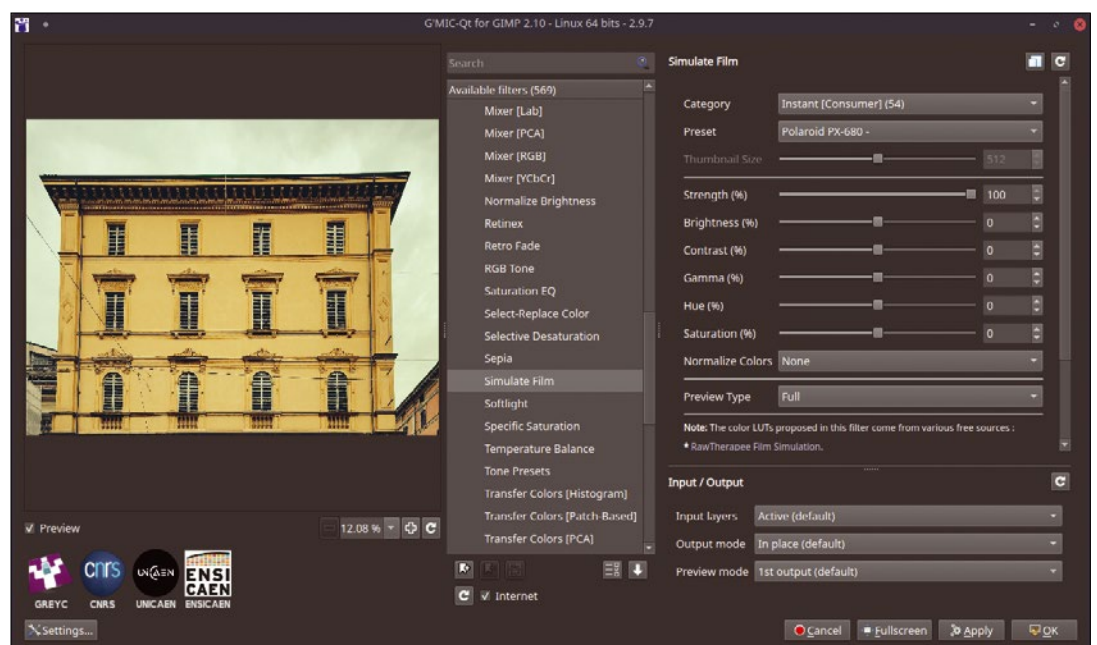


Figure 2: G'MIC offers plenty of high-quality presets to choose from.

Enter the G'MIC Inpaint tool (Figure 4), which lets you make objects disappear without too much effort. The way it works is rather straightforward: Use a certain color to mask the unwanted object, and then let Inpaint do its job.

In Gimp, this requires a few simple steps. Open the desired image and set the background color to red (the *HTML notation* value should be `ff0000`). Use the Lasso tool to make a rough selection around the object you want to remove. Choose *Edit / Fill with BG Color* to fill the selection with the red color, and then choose *Select / None* to remove the selection. Open G'MIC by choosing *Filters / G'MIC-Qt* and select the *Repair / Inpaint [Multi-Scale]* tool. Specify the red color that matches the color you

used for the mask in the *Mask Color* input field, and watch the masked object magically disappear in the preview window. G'MIC offers several Inpaint variants based on different algorithms. So if *Inpaint [Multi-Scale]* doesn't do a good job of removing the selected object, you can try other versions of the tool. And, of course, you can tweak the available parameters to achieve a better result.

Speaking of masks, G'MIC comes with a masking tool that makes it possible to quickly add masks based on a specific color. This tool can come in useful if you need to make adjustments to a specific area of an image, such as making the sky in the image brighter, desaturating a bright red car, or replacing one color with another

one. To create a mask in Gimp, choose *Filters / G'MIC-Qt*, switch to the *Colors / Color Mask [Interactive]* tool, and press *Apply* (Figure 5). This opens a separate window where you can use the mouse to select the desired color. Mouse over the area you want to mask and right-click on it. To select a larger area, press and hold the right mouse button and drag the pointer across the area you want to select. Keep pointing and clicking until you've selected the entire area. If you accidentally select an area that shouldn't be included in the mask, point to it and left-click to remove it from the mask. Use the *R* key to reset the mask completely. Sometimes it can be difficult to see the mask's exact boundaries; you can use the *Space* or *Tab* key to switch between different viewing modes to give you a better preview of the current selection. When you are finished, close the preview window, and you should see a new layer with the mask in the *Layers* pane of Gimp.

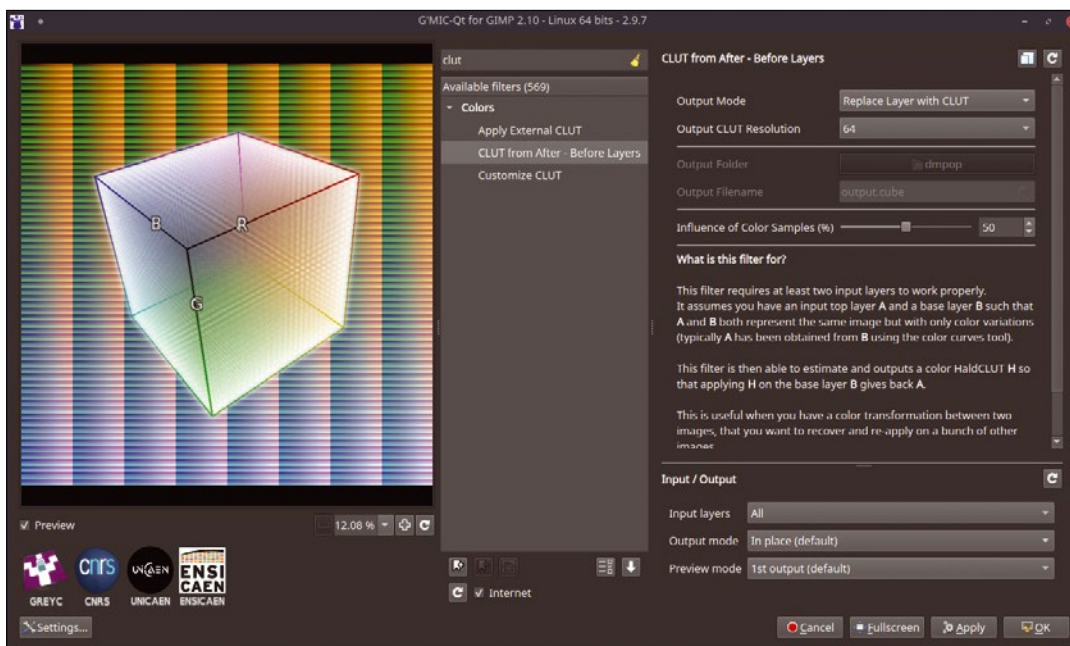


Figure 3: You can use G'MIC to generate your own Hald CLUT presets.

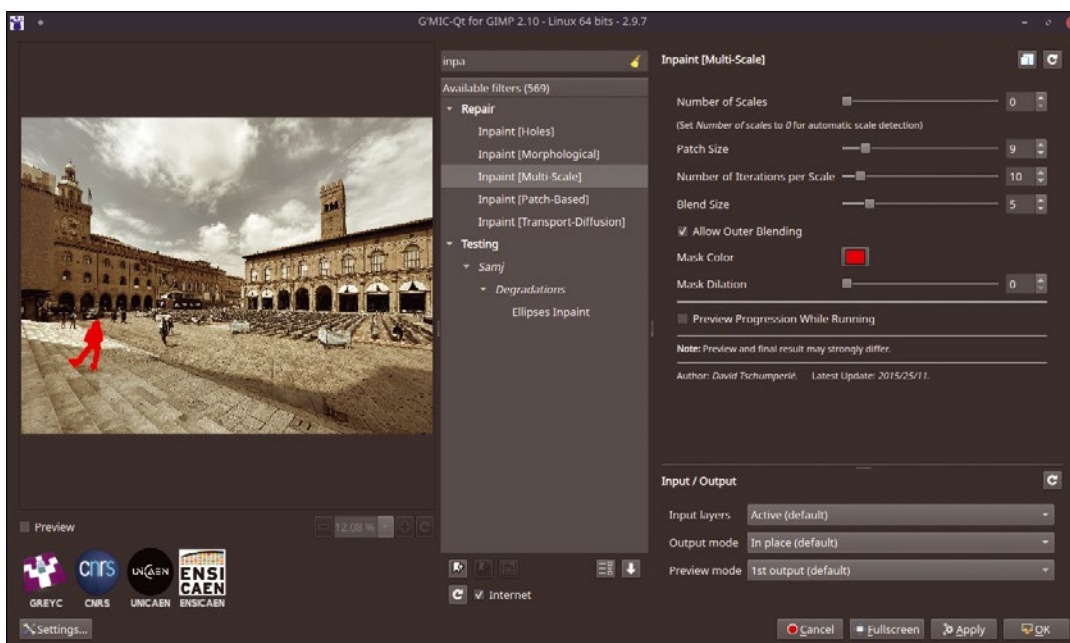


Figure 4: Removing those pesky humans is easy with the Inpaint tool.

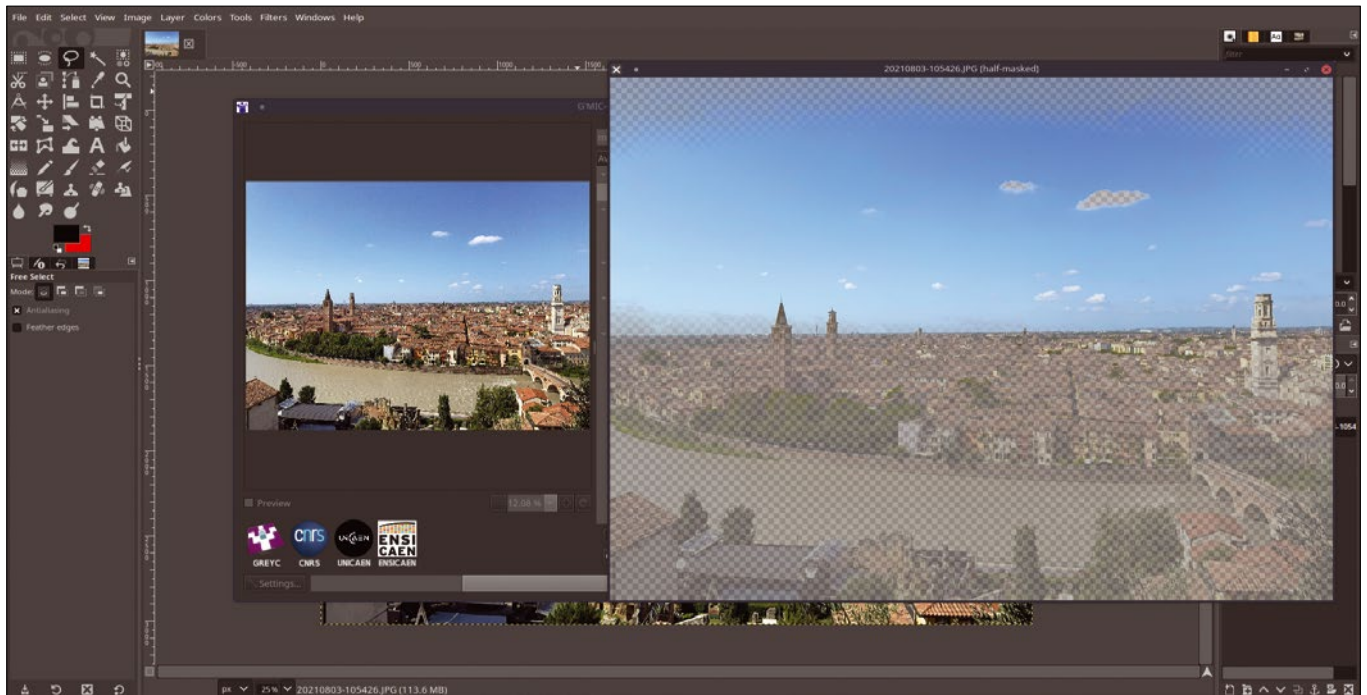


Figure 5: Creating a mask with the Color Mask tool.

It's Complicated

Because G'MIC does a good job of hiding the complexities of image manipulation behind a simple interface, it's easy to forget the serious science happening in the background. Take, for example, the *Details | Sharpen [Deblur]* tool. It's based on the Jansson Van-Cittert deconvolution algorithm, and it's mostly used for recovering blurry images in microscopy and astronomy. The available parameters allow you to control how the algorithm is applied. The *Radius* parameter specifies a standard deviation of the Gaussian kernel that is supposed to model the image's blur degradation. In practical terms, you would use a high radius for very blurry images and a small radius when the blur is low. The Jansson Van-Cittert algorithm is iterative, meaning it starts with a blurry image and converges toward the deconvoluted image through multiple iterations. The *Iterations* parameter allows you to specify the desired number of iterations. The blurrier the source image is, the more iterations are required to make it sharper. The *Time Step* parameter controls the maximum intensity variation (in R, G, B) between two consecutive iterations. Assuming an image has RGB values in range $[0, 255]$, the default value 20 specifies that the pixels cannot vary more than ± 20 in value between two iterations. To avoid sharpening the noise, you can use the *Smoothness* parameter to specify the weight of the regularization term. The noisier the image, the higher smoothness value you need.

Of course, you don't need to know the underlying theory in order to use this and other tools. Tweaking the available parameters and preview-

ing the result is often the best way to learn how a specific tool works and what it can do for you. But uncovering the theory behind certain tools provides a fascinating insight into the science of image manipulation. As already mentioned, finding information about specific tools can be a bit of a challenge, but asking your questions on PIXLS.US [2] is a good place to start.

Wrap-Up

Whether you are looking for high-quality presets you can quickly apply to your photos or you are interested in more advanced image processing tools, G'MIC delivers. In addition to a simple graphical interface, G'MIC also offers a command-line interface, which means that you can automate image editing actions and create rather advanced image processing scripts. In short, G'MIC makes a perfect addition to your image processing toolbox. ■■■

Info

- [1] G'MIC: <https://gmic.eu/>
- [2] PIXLS.US:
<https://discuss.pixls.us/c/software/gmic/10>

The Author

Dmitri Popov has been writing exclusively about Linux and open source software for many years. His articles have appeared in Danish, British, US, German, Spanish, and Russian magazines and websites. You can find more on his website at tokyoma.de.

FOSSPicks

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



Graham's ancient LG phone finally gave up on booting this month. Everything was backed up except what was originally \$25 of Bitcoin intended to buy a pizza ... five years ago. **BY GRAHAM MORRISON**

Modular music maker

Bespoke

It's amazing just how many incredible audio-related open source projects keep appearing. There seem to be more releases in this software category than any other, indicating the popularity of synths and music production. One of the best of these new releases, Bespoke is a super high-quality modular synth and music-making environment

that follows hot on the squeals of the similar and equally remarkable VCV Rack. While VCV Rack is being used to primarily emulate real and fictional Eurorack hardware, complete with virtual patch cables that sink with gravity and skeuomorphic controls that need to be twiddled from a mouse, Bespoke is focused on good user interface (UI) design. It doesn't bear any resemblance to physical hardware or other audio projects, other than the unique touchscreen table and physical controllers of Reactable and perhaps

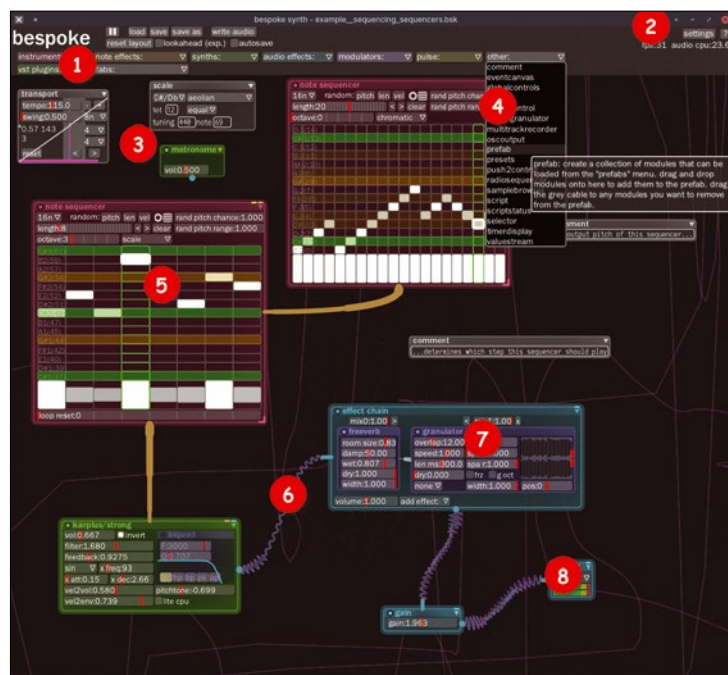
even the ancient Bars & Pipes sequencer for the Commodore Amiga. Even if you have no interest in audio, you will want to play with it.

The heart of a modular synth is its modules. These are the parts that need to be wired together to generate a sound, and this is where Bespoke really excels. Many modules are included, split into categories such as instruments, synths, note effects, plugins, modulators, and audio effects. Every module has a beautifully designed UI with a "1970s disco meets vector display" aesthetic that forces you to play with it. There's a polyrhythmic sequencer called *circlesequencer* that consists of four concentric and rainbow-colored circles, and notes are played when the hand of a dial crosses their step point. The grids of the matrix note and drum editors could be from *Tron*, and the envelopes in the FM modules look like green CRT oscilloscopes. When a module offers multiple parameters, these are easily edited directly with text entry or by using sliders and buttons just like in any well-designed UI and not like a copy of a physical hardware unit transposed onto a screen. You can still drag connections between modules, but you can also drag modules together to create a default configuration, and there are keyboard shortcuts for everything. All of this is built on a zoomable and scrollable canvas that never taxes your GPU.

The most impressive graphical element becomes visible when you begin to connect modules together. Much like connections in the aforementioned Reactable, virtual wires stretch between inputs and outputs and animate to show what's been carried across them. If you start with a simple signal generator, set this to produce a sawtooth waveform, and connect this to a gain module to control its amplitude, the virtual cable will wobble to illustrate the shape of the audio waveform going through the cable. More than just eye candy, it's an excellent way to quickly see what is happening where and the effect that certain modules are having on the audio or data streams. The background of the canvas will also animate to display an oscilloscope representation of the output, and it can all be turned off if you don't like the distraction. Needless to say, it sounds just as good as it looks. Plus, there's a brilliant features matrix on the website that shows the difference in features between the free version, the plus version, and the pro version – the only difference being the fewer dollars in your pocket if you choose to make a financial contribution, which is definitely a worthwhile thing to do.

Project Website

<https://www.bespokesynth.com>



1. Modules: Choose a module from a category and wire them together in the canvas. **2. Settings:** With direct output to ALSA, Bespoke works at a low latency and displays both audio and GPU overhead. **3. Sync and scale:** Bespoke will talk to other apps and devices, and pitch can be locked to a specific scale and tuning. **4. Help:** Most UI elements contain excellent rollover help text, including the scripts and esoteric modules. **5. Matrix editor:** Bespoke can do more than just create sounds; it can sequence and record them, too. **6. Waveforms:** The canvas background, audio connections, and data connections all wobble to help illustrate their data in real time. **7. Effects chain:** Use VST plugins alongside the native modules. **8. Output:** There are several modules for monitoring levels.

Android emulation

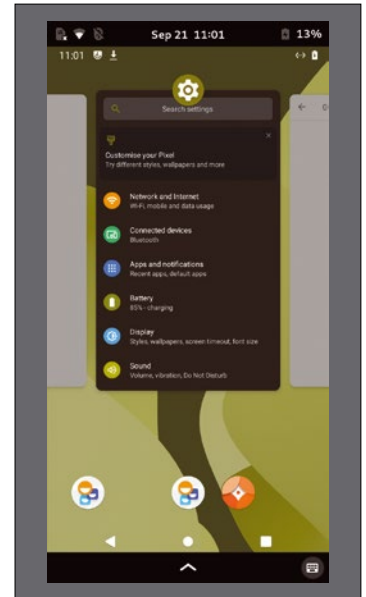
Waydroid

One of the most interesting projects of the past few years is Anbox, a brilliant compatibility layer that does for Android what Wine does for Windows. It allows you to run an Android-based operating system and many of its associated apps from a traditional Linux distribution and, more recently, commercially served from the cloud. There are plenty of ways this could be useful, from running Android applications that don't have a Linux counterpart to testing your Android applications on a dozen different virtual configurations. But it hasn't been so useful running Android from a phone, mostly because it lacks decent ARM-specific acceleration.

Running Android within a container on a phone might seem an odd requirement because the phone may be running Android anyway. But this isn't the case if you're running one of the current generation of Linux smartphones, such as a PinePhone, with a native Linux operating system such as

Ubuntu Touch, postmarketOS, and Manjaro (with either Gnome or KDE Plasma). These installations are running native Linux packages built just like their desktop counterparts, and that means they also miss out on running Android apps, despite being built on the same architecture.

This is where Waydroid helps. Waydroid uses Wayland alongside native non-virtualization Linux subsystems to implement an effective low-resource container for hosting Android. By default, it will instantiate a copy of LineageOS built on Android 10 with almost native-like performance. Even on a relatively low-powered device such as the PinePhone, we were able to install the minimal set of Google Apps and access the Play Store, but it's even easier to just add F-Droid. The result is that Android apps will run within a native Linux install on a smartphone. While it's still rough at the edges and very early in its development phase, Waydroid is a compelling glimpse at the best of both worlds.



Waydroid will also run on x86 hardware. However, as it will only run x86-compiled Android derivatives, it's less useful than its ARM counterpart.

Project Website
<http://waydro.id>

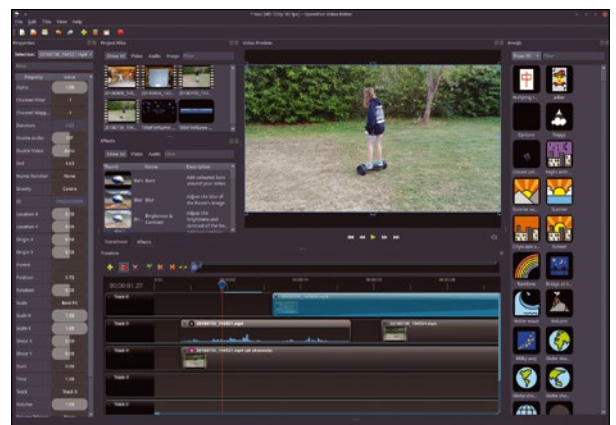
Video editor

OpenShot

We've not looked at OpenShot for some time. During that time, the state of video editing on Linux has improved massively. Linux video editing is now at the point where we've noticed many YouTubers using open source editors over their proprietary alternatives, much as they choose to use Audacity or Blender. Kdenlive has become a stable, powerful, and accelerated alternative to the costly Final Cut Pro on macOS, and it appears OpenShot is fast catching up. OpenShot adds the kind of features that take it beyond functional video editing and into creative territory. There's a host of new video and audio effects, including, for example, motion tracking, object detection, and video stabilization. The last one is reportedly one of the

most requested features ever, and it's great to see what was not so long ago a cutting edge set of features making it into an open source application. Both the motion tracking and object detection are equally impressive, letting you identify objects within your footage that can then be tracked and processed in various ways, such as identifying all cars in a scene and adding labels to them automatically.

The new audio effects perform an equally essential role. There's a compressor to lessen the difference between quiet audio and loud audio, an expander to amplify quiet audio without clipping, a parametric equalizer for filtering out parts of the audio such as hum, and various creative effects such as distortion, delay, and "robotization." All of these additions make use of the



Nearly 1,000 emojis have been added to OpenShot to encourage learning and experimentation when editing and creating videos.

improved performance. You'll need a package source other than the AppImage to take advantage of the almost essential hardware acceleration. For simple editing, however, OpenShot will still work on almost anything; the editor itself has plenty of small refinements such as improved zoom and transformations and much more intuitive clip snapping, plus most other video effects have been given an overhaul. If you've not tried it since we last took a look, it's definitely time to give OpenShot another go.

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

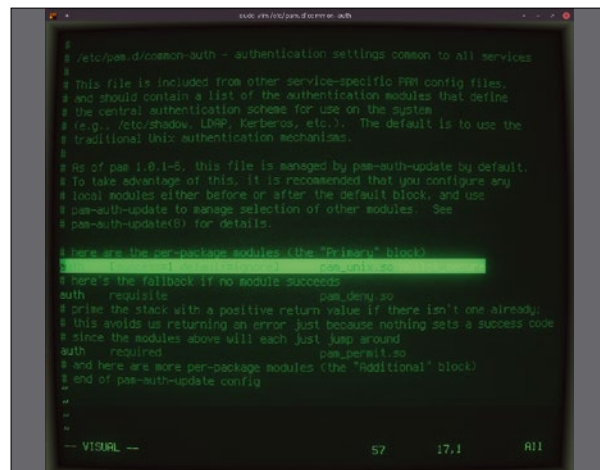
Security tool

PAM Duress

A few years ago, I was doing quite a bit of traveling between the US and Europe with my usual array of technology. I was worried about what I should do if I was forced to unlock a device and either hand that device over or permit the device to be searched. While I wasn't involved in anything that might be considered investigative journalism, I did want to set a good example and behave appropriately if anything like this happened. One impractical solution I envisaged was letting someone else encrypt my devices, so I could honestly say I didn't know how to unlock them (with the intention of asking that trusted someone for the keys when I arrived safely). Another more practical option was to take devices completely empty

of anything, setting them up and erasing them as I arrived and departed again. Of course, I was never organized enough to do either of these things.

If PAM Duress had been around, I would have gone for this solution. The pluggable authentication module (PAM) system is at the heart of granting access to your Linux devices, and PAM Duress is a module that can trigger scripted behavior when you enter a password that's different from the one you'd normally use to unlock your data and device. These duress scripts can delete all your data, automatically send a notification to someone, or do whatever other function you desire. Installation is relatively straightforward and similar to any other PAM module. The scripts that are executed when a



It's difficult to show PAM Duress in action, but it's easily enabled by simply editing your PAM configuration (after making a backup of your system).

certain password is entered are signed and cannot be tampered with, although there is a testing function that can ensure the module is working correctly before deleting your data (for example). Everything works as expected. This may be a project with a very specific objective. If it appeals to you, PAM Duress performs a brilliant and essential function.

Project Website

<https://github.com/nuvius/pam-duress>

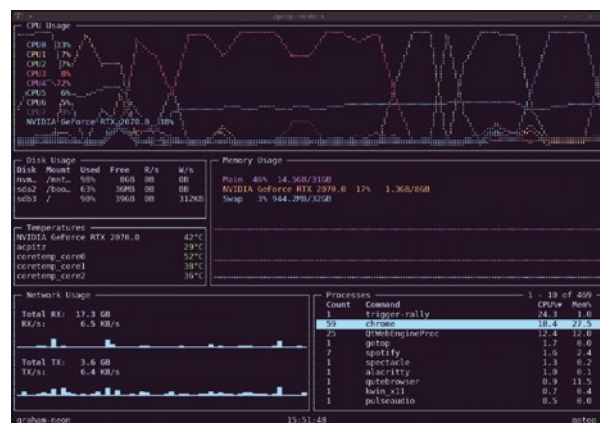
Usage monitor

gotop

Over the years, we've looked at a lot of activity monitors. There are now so many that they've become difficult to write about with any originality. It must have become a Linux developer's rite of passage to create a tool that shows which processes are taking up CPU time and hogging all the system memory. But gotop is worth writing about because it takes the best parts of some of our favorite monitoring tools and combines them into an easy-to-install, cross-platform, and CPU-efficient single executable. There are precompiled binaries for many ARM variants, including executables that will simply run on a Raspberry Pi, x86 Linux, macOS, and even Windows. Each version runs identically,

with a few exceptions. There's an option to monitor NVidia graphics hardware, for example, and power monitoring is also dependent on your hardware. Apart from that, every platform can take advantage of all the same features, which is a good thing because there are quite a few.

One of the best things about gotop is that it's clean and performant by default, especially compared to something such as htop, which can clutter your terminal and your CPU (but is still a great tool). With no further modification, gotop will graph the use of each CPU core across the top; disk usage, temperature, and memory usage through the middle; and networking throughput and process management below. All of this can be configured and



Thanks to popey (Alan Pope) and the now-defunct Ubuntu Podcast for this find in one of their final episodes.

saved as a preset, including themes, styles, widgets, update frequency, and some options on how the values are calculated. It's even possible to monitor the state of remote servers. It's incredibly configurable for a tool with such a small system footprint, and that's what makes gotop such a good find. It's the best of all possible worlds without any compromise.

Project Website

<https://github.com/xxxserxxx/gotop>

HTML processor

htmlq

If you've ever done any development by working on your own scripts or hacking on a website, you will have encountered JSON. JSON is a text notation format for sending, retrieving, and storing data, and it uses a specific JavaScript-like syntax of curly brackets and square brackets, double quotes and commas to contain that data. These elements are used to describe key and value pairs ("name": "graham", for instance), arrays, and hierarchy. YAML files aren't difficult to comprehend, and many programming languages will provide libraries to help parse the data transparently, but they can be difficult when you want to create a simple script or use them from the command line because you need to

find a way of navigating their hierarchical bracketed syntax. This is where the venerable `jq` command can help. It's a command that parses all of JSON's magical syntax for you, making it easy to access the actual data held within its confines. At its simplest, you can ask for the values assigned to `names`, but it's advanced features enable you to process JSON files in the same way `sed` processes text files. But what about working with HTML files directly?

`htmlq` is a tool that does just that, and it's why `htmlq` has a `q` in its name. It's a command that wants to do the same for HTML files that `jq` does for JSON. If you grab the HTML for a web page with `curl`, for example, you face the same problems retrieving

```

> curl -s https://www.wikipedia.org/ | htmlq --attribute href a
//en.wikipedia.org/
//ja.wikipedia.org/
//es.wikipedia.org/
//de.wikipedia.org/
//ru.wikipedia.org/
//fr.wikipedia.org/
//zh.wikipedia.org/
//it.wikipedia.org/
//pt.wikipedia.org/
//pl.wikipedia.org/
//pl.wikipedia.org/
//ar.wikipedia.org/
//de.wikipedia.org/
//en.wikipedia.org/
//es.wikipedia.org/
//fr.wikipedia.org/

```

`htmlq` takes the pain out of parsing HTML brackets and extracting content such as links.

exact values without complicated multiple uses of `grep`. Piping the output through `htmlq` instead lets you quickly retrieve page attributes, the raw main body of text, or specific sections or parts by identifier. It's a brilliant way to scrape data from websites that don't provide (or allow) their own REST API access, such as sites listing train timetables or ultra-local weather reports. It's the quickest way we've found to extract URLs from an external page, either for testing or for further scraping.

Project Website

<https://github.com/mgdm/htmlq>

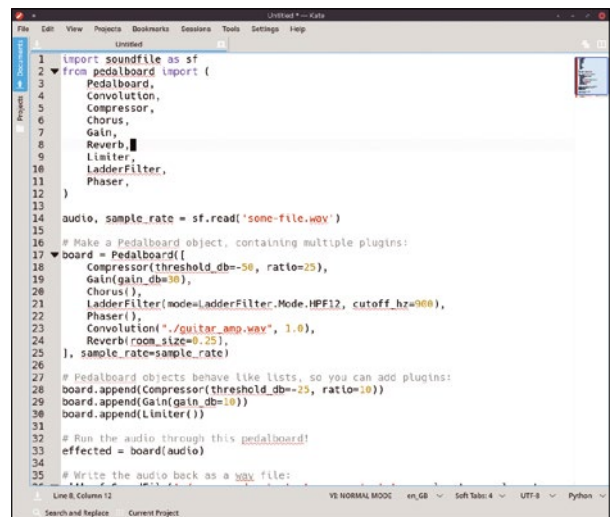
Scripted audio effects

pedalboard

The Spotify music streaming service has become hugely successful. And like many other Internet behemoths, its success is partly built on open source. The quality of the Ogg Vorbis audio codec and container has helped Spotify differentiate itself from the other streaming services, for example. While Spotify doesn't use the open source version of Qt, its adoption would have helped fund the Free version. It's also published a number of open source projects of its own, with `pedalboard` being its latest. `Pedalboard` is an audio effects library for Python, which may sound overly technical for most people, but it really does have genuine utility for most of us. If you've used the Audacity audio editor, for example, you'll be used to the idea of processing your

audio with audio effects one at a time. `Pedalboard` allows you to do this in real time, with whatever effects you choose, and even includes some of its own – all tied together with a little Python.

`Pedalboard` includes several "bread-and-butter" effects to help with general audio issues. These include convolution (an effect that imbibes the acoustics of a real or digital space from an impulse), a compressor, chorus, distortion, gain, filters, limiter, phaser, and reverb. These are all of a very high quality, and despite being only useful to Python programmers, the documentation makes them accessible to anyone with only a smattering of Python experience. But the best feature is one that would previously only be feasible for experienced programmers:



```

1 import soundfile as sf
2 from pedalboard import (
3     Pedalboard,
4     Convolution,
5     Compressor,
6     Chorus,
7     Gain,
8     Reverb,
9     Limiter,
10    LadderFilter,
11    Phaser,
12 )
13
14 audio, sample_rate = sf.read('some-file.wav')
15
16 # Make a Pedalboard object, containing multiple plugins:
17 board = Pedalboard([
18     Compressor(threshold_db=-50, ratio=25),
19     Gain(gain_db=30),
20     Chorus(),
21     LadderFilter(mode=LadderFilter.Mode.HPF12, cutoff_hz=9000),
22     Phaser(),
23     Convolution('guitar_amp.wav', 1.0),
24     Reverb(room_size=0.25),
25 ], sample_rate=sample_rate)
26
27 # Pedalboard objects behave like lists, so you can add plugins:
28 board.append(Compressor(threshold_db=-25, ratio=10))
29 board.append(Gain(gain_db=10))
30 board.append(Limiter())
31
32 # Run the audio through this pedalboard!
33 effected = board(audio)
34
35 # Write the audio back as a wav file:

```

Even Python beginners can access cutting-edge audio effects with Spotify's `pedalboard`.

adding external effects. `Pedalboard` can use VST3 plugins alongside its own, and they can be used in your code just as easily as the native plugins. There's nothing else quite like this, with the closest alternatives being perhaps SuperCollider or Pure Data, but neither have the convenience and ubiquity of Python.

Project Website

<https://github.com/spotify/pedalboard>

Modern editor

Onivim 2

The Vim text editor doesn't normally need much of an introduction. It's the yin to the Emacs yang, the source of many "How do I quit?" posts, and the fuel for many a late night best-editor argument. It's a terminal text editor driven by infamously opaque keyboard commands that need to be committed to memory and normal, visual, insert, and select operating modes that can totally confuse a generation brought up on Microsoft Word. But it is also ubiquitous, powerful, mature, and ultimately wedded to the command line. There have been many attempts to bring its uniqueness to the desktop, but most of us would rather simply open a terminal and edit our files from there. If we need to make a graphical editor more Vim-like, there's usually an option to switch its keybindings to those of Vim. Editors such as Plasma's Kate and Microsoft's Visual Studio Code take this further with plugins that can even ape the command and editing modes, alongside all the muscle memory shortcuts and keyboard commands.

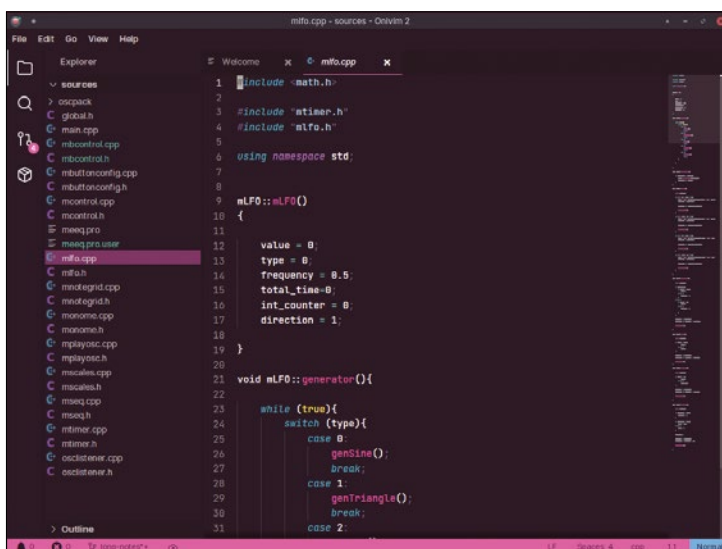
Onivim 2 is a new graphical editor that, even in its alpha state, is already one of the most successful desktop editors we've seen at supplanting Vim's workflow from the command line to a modern desktop environment. Like Vim, it's modal, which means you can edit text not just interactively (as you would with a normal desktop editor) but also by issuing commands that typically consist of a verb followed by a motion, such as `d$` to delete to the end of a sentence. This is exactly the same as how Vim works. But it isn't just Vim that Onivim 2 is hoping to replace: It's attempting to bring the best of editors such as TextMate, Sublime Text, and Visual Studio Code editor to a single high-performance application. Despite being completely independent of these projects, the editing experience is already fairly comprehensive. It features syntax highlighting, fuzzy search across a project, snippets, and a command palette, all packaged within an amazing cross-platform application shell that isn't using Electron.

The project has an interesting licensing policy, which also makes it difficult to get hold of



The only way to currently test Onivim 2 is to build it yourself. It requires 15GB of space and an hour or two, but it is straightforward thanks to a well-documented Docker recipe.

the application in its alpha state without contributing. The project is a commercial endeavor, currently funded by Patreon supporters, but ultimately, paid-for license keys. The code is initially released under a non-free license, allowing only for non-commercial and educational use without a commercial license, and it takes a long time to build (we built the code manually). But due to the positive support the project has received from the open source communities, Onivim 2 now has a dual-licensing agreement. Eighteen months after a commit has landed in its code repository, it will become additionally licensed under MIT, which is when we'll all be free to make and distribute our own builds. This isn't a perfect solution, and it would be preferable if the project were open source from every commit, but we equally respect the project's decision and motivation. It's hard bootstrapping a business, building a new editor, and maintaining momentum. Hopefully, the project will be successful enough, and so well-funded regardless of its licensing restrictions, that the developers will update their licensing to remove the long delay. Either way, Onivim 2 is definitely worth seeking out.



While many editors feature a Vim mode, Onivim 2 makes this its central function and inspiration.

Project Website
<https://www.onivim.io>

Retro platformer

Mr. Rescue

We've looked at the LÖVE games framework before. It's a brilliant way to easily create 2D games with the accessible LUA language. Many developers have already used it to create unique and fully fledged games. Mr. Rescue is one of these. Like many of the games built using LÖVE, Mr. Rescue has a lovely pixelated design reminiscent of games from the 8-bit era, albeit on a console with a 24-bit color palette. The game's objective is equally positive. Rather than trying to destroy things, you save people by running through an already burning building, picking them up, getting to a window, and throwing them out. There's no mention of their fate from this point, especially

when the buildings can stretch dozens of floors into the sky, but let's assume there's some substantial landing setup waiting for them. Another thing to remember is to open the window first.

Your only defenses against the fire are your suit, which gets increasingly damaged the closer you get to the flames, and your fire extinguisher, used to blast the flames with water to make a path through the building, as well as to smash open windows and doors. You can only blast the water for so long before needing to wait for the pressure to replenish, and your suit also needs to be recharged by finding coolant capsules as you explore the building. If you don't do this, you'll eventually overheat and the game will be over. In this way,



If you're looking for some simple, pixelated arcade fun, Mr. Rescue is a perfect lunchtime distraction.

the game builds up momentum as you try and take more risks to save people while fighting the fire. When you've saved a certain number of people, the level is over and you move on to another building. It's a great game that's brilliantly animated and finely tuned, with a soundtrack that could have been produced on a Commodore Amiga in 1988.

Project Website

<https://tangramgames.dk/games/mrrescue/>

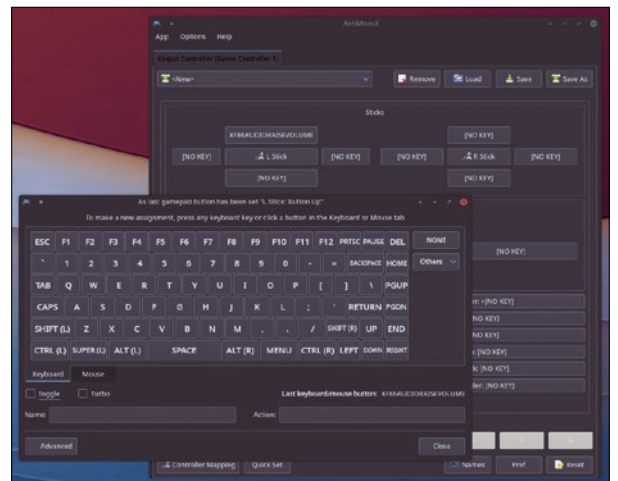
Controller editor

AntiMicroX

It finally seems like Wayland, the successor to the X.Org windowing system, is ready for prime time. Not only are we seeing it as the only viable option for many embedded and mobile devices, several major Linux distributions now pick Wayland by default. With Valve's Steam Deck bringing Linux gaming into the spotlight, there are signs that NVidia, too, might finally begin optimizing its drivers for Wayland. But this does mean that certain applications relying on X.Org might need to be reworked, and those would include anything with XInput to access games controllers.

AntiMicroX is a graphical configuration application for games controllers that has just finished a compatibility transition to Wayland, bringing its powerful "assign

anything anywhere" approach to the new desktop. When you connect a controller, its main window will display buttons for every input detected on the device, both digital and analog. Pressing or triggering any input on the controller will highlight its corresponding button within the user interface. Selecting any one of these will open a virtual QWERTY keyboard from which you can assign any key or mouse combination to be triggered when the controller button or axis is activated. There are preset bindings for common controllers, including dead zones to stop movement when an analog input is not being used and a graphical calibration tool for scaling the analog inputs. You can save each setup as a preset between 1 to 8, and the entire configuration can be saved as



AntiMicroX offers more configurability options and better preset management than Steam, and it now works on Wayland.

a profile file that's quickly accessible from a drop-down menu. This allows you to create complex controller schemes for your games and quickly switch between them, regardless of how many buttons you need.

If you're seriously into gaming, this is a powerful tool that can help you get the most out of your hardware configuration – and especially compensate for any missing Linux configuration support.

Project Website

<https://github.com/AntiMicroX/antimicrox>



Linux Magazine is your guide to the world of Linux. Look inside for advanced technical information you won't find anywhere else!

Expand your Linux skills with:

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

If you want to go farther and do more with Linux, subscribe today and never miss another issue!

Subscribe now!
shop.linuxnewmedia.com/subs



Rendering a perfume bottle with Blender

Fragrance Workshop

Blender's massive feature set can seem overwhelming at first. Choosing a manageable project can help you get started.

BY CLAUS CYRNY

Blender, the free and open source creation suite, includes a modeler, three render engines, a compositor, a tracker, a nonlinear video-editing system, a particle system, and the ability to animate physical simulations and export them as video. Suffice it to say, figuring out where to start can be overwhelming. Picking a relatively manageable project is key. An easy way to start is to model a simple item, such as a perfume bottle. For this tutorial, I will model a transparent perfume bottle filled with liquid. By using concrete but varied shapes, a limited scope, and a small number of different materials (i.e., surfaces), this tutorial can help take the frustration out of getting started with Blender. (Note: Having some basic previous experience with Blender will be helpful in this tutorial).

Setup

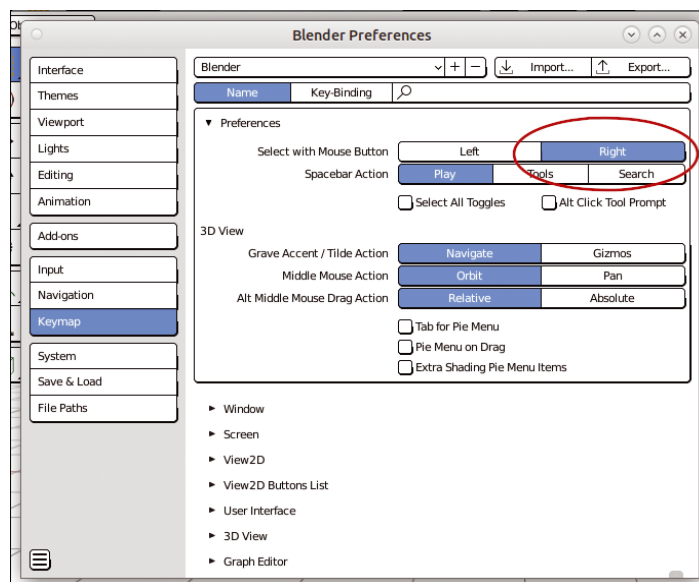
I always recommend working with the latest version of Blender, which is currently 2.93. However, you will find Blender 2.93 in only a few distributions. For instance, Ubuntu 21.04 includes

Blender 2.83.5, and Fedora provides the latest version via updates. If your distribution does not have the current release, you can download the program as a TAR.XZ file from Blender's homepage [1] and unpack the archive on your computer. Then call the program's binary, `blender`, or create a starter for the desktop environment and link to the TAR.XZ file.

Before getting started, you may want to make a couple of adjustments. If English isn't your first language, you can change the localization in *Edit | Preferences... | Interface* under *Translation*. Keep in mind that many of the menus will remain in English (and many online tutorials use English terminology).

You may also want to change how you select objects with the mouse. In the default Blender configuration, you select objects by left-clicking. However, I recommend changing the selection to the right mouse button. To do this, go to *Edit | Preferences... | Keymap | Select with Mouse Button* and change the setting from *Left* to *Right* (Figure 1).

Figure 1: Selecting with the right mouse button makes it easier to work in Blender.



Setting the Scene

Blender uses scenes as a way to organize work. After startup, Blender displays the standard scene: a camera, a light source, and a cube. You will use the cube to model your perfume bottle, but before you get started, you need to do some preliminary work.

First, you need to define *Cycles* as the renderer and set the image dimensions and camera settings. By default, Blender uses the Eevee render engine. While the newer Eevee engine works faster, Cycles is better suited for rendering realistic objects. To change the render engine, go to the *Render Properties* tab in

Blender's right sidebar. While you are in that tab, you should also increase the value for *Sampling | Render* to 512. Then, save the scene with a meaningful name.

You can use the default scene's camera and cube settings. To change the light source type, select the light source type, select the light source, *Light*, under Scene Collection and then select the characteristic *Area* instead of *Point*. This type of light source will cast more realistic (not so harsh) shadows.

If you are not yet familiar with navigating in Blender, see the "Navigation" box for some basic information.

With only one view available in the viewport (Figure 2), Blender's default interface is not very clear cut. I recommend modifying Blender's interface by

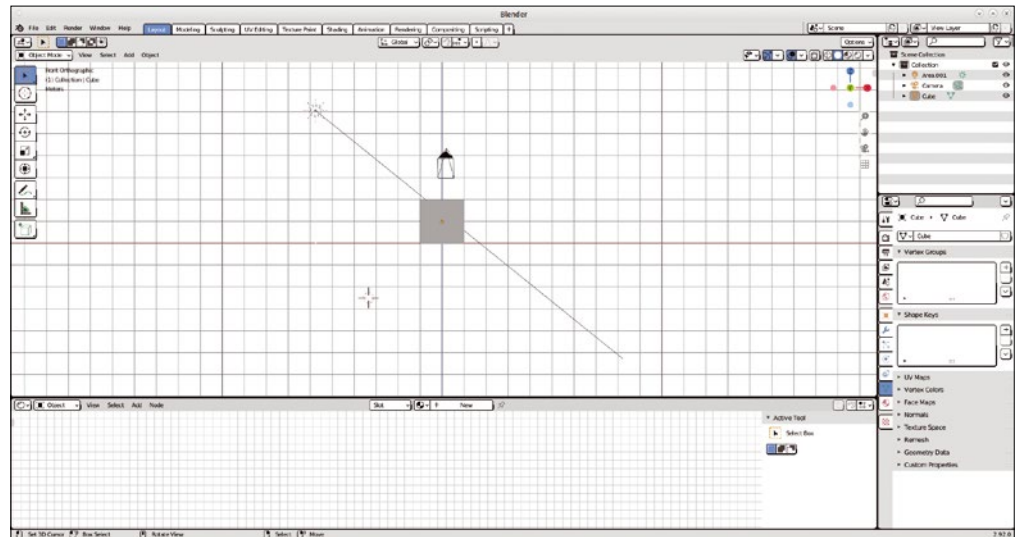


Figure 2: Modifying the Blender 2.93 interface: The viewport can be divided into multiple viewports.

creating an additional camera view by positioning the cursor on the dark gray vertical line between the Properties Editor (bottom right) and the 3D viewport. A dark gray double arrow with a white border will appear. Right-click to pop up a dialog (Figure 3).

Navigation

When working with Blender, there are some special features to keep in mind when navigating. Similar applications usually work with key combinations of a modifier such as Ctrl, Alt, or Shift plus a letter or a number. Blender takes a different approach. It enables many functions via a simple keystroke without the modifier key. In many cases, additional keys then follow as an option that specifies the selected function. For example, you can enable scaling of an object with *S* and then restrict it to the z-axis with *Z* (see Table 1 for more key combinations).

Table 1: Important Key Combinations

Key	Command
<i>0</i>	Change to camera view
<i>1</i>	Change to the front view
<i>3</i>	Change to the side view
<i>7</i>	Change to the top view
<i>G</i>	Move an object in the scene ("grab"). This can also be done by right-clicking on the object and moving it while holding down the mouse button. <i>G</i> also lets you restrict the function to one axis. For example, pressing <i>G, Z</i> moves you along the z-axis, which allows for far more precise modeling of the scene.
<i>S</i>	Scale an object either continuously or by a certain factor. Like <i>G</i> , scaling can be limited to one axis, for example, by pressing <i>S, X</i> . <i>S, 2</i> , on the other hand, lets you double the size of an object.
<i>E</i>	Extrude an object freely or only on one of the three axes (x, y, and z). <i>E</i> is only available in Edit Mode .
Shift+A	Insert objects, sorted by categories ("add"). In the beginning, you probably will be working mainly with objects from the Mesh and Light categories.
F12	Renders the complete image from the roughly calculated scene. Depending on the processor, graphics card, and scene size, this step may take awhile.
Shift+middle-click	Move and rotate the scene in the viewport. Alternatively, click on the hand icon top right in the viewport and move the view by dragging the mouse while holding down the left mouse button.

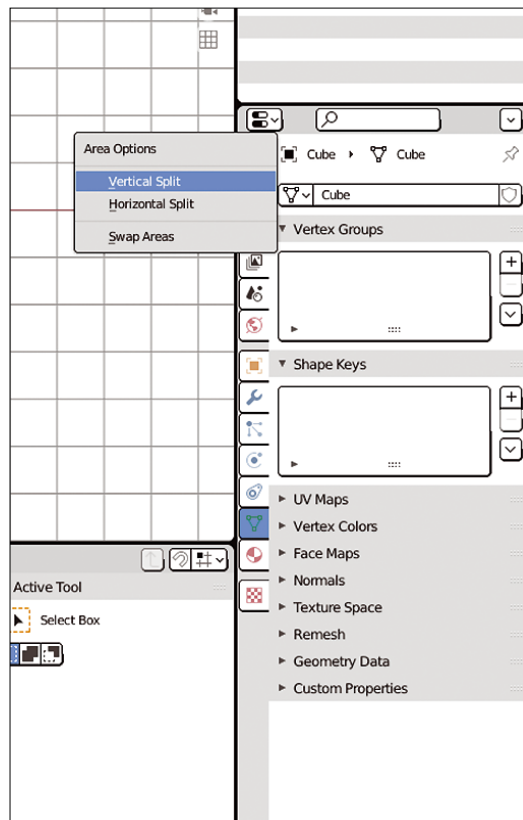


Figure 3: Right-click on the edge of the viewport to split the view.

Under *Area Options*, select *Vertical split*. Then drag the gray line to the left until you see two identical viewports. To make the right viewport show the camera's viewing angle, switch to Camera View with the mouse, press the left mouse button once, and then press *O* on the number pad.

If you want Blender to always use this viewport setup when creating a new Blender scene, save it

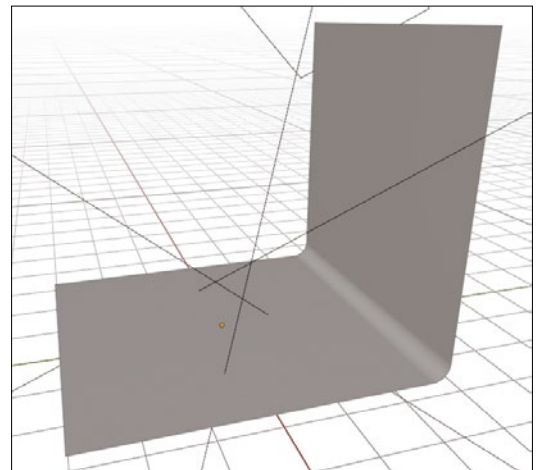


Figure 4: The bevel in the background provides a smooth transition between the object and the scene.

by selecting *File | Defaults | Save Startup File*. This view will now be the permanent default that you will see each time you start Blender.

Background

Next, you need create a background with a bevel (Figure 4). Press *7* to switch to the top view, followed by *Shift+A*, and select *Add | Mesh | Plane* to create a new layer on the scene.

Switch to *Edit Mode* (see the “Blender Modes” box) and click on the *Edge Select* button. Select the edge farthest away from the camera and extrude it into the *Z* axis with *E,Z* so that the newly created second plane is at a right angle to the first plane.

Next you need to apply a bevel modifier to round the right angle to form a sloping edge. Switch to *Object Mode* and then click on the blue wrench icon in the Properties Editor on the far right. In the

Figure 5: The Bevel Modifier rounds off the bevel's hard edge to avoid creating hard shadows.

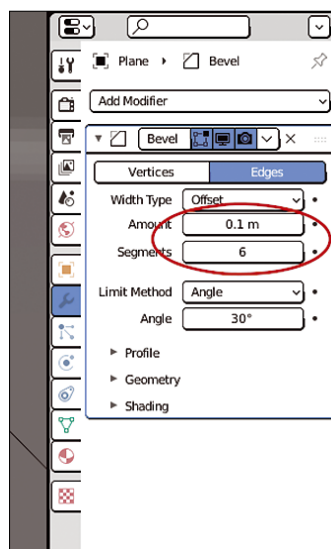
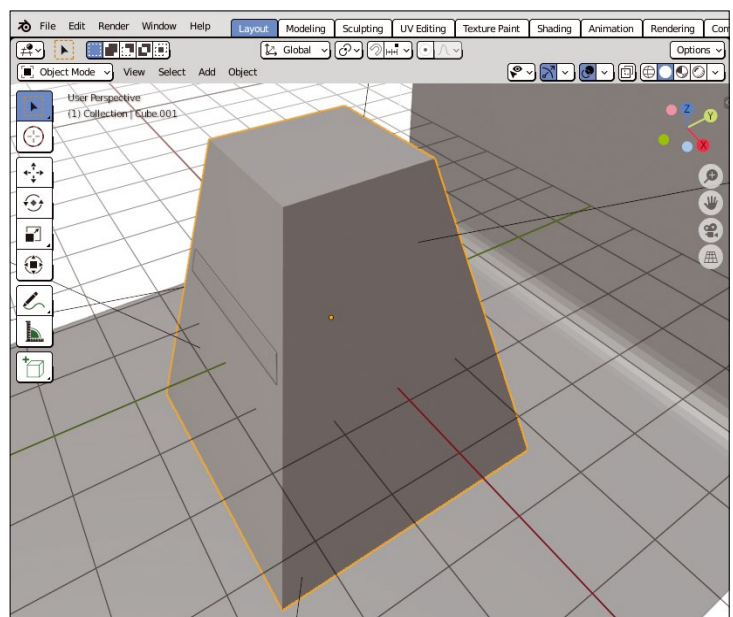


Figure 6: Starting with the default cube, you can create a relatively flat-topped pyramid.



Blender Modes

Blender has six modes; the two most important modes are *Object Mode* and *Edit Mode*. You can switch between these two modes by pressing the tab key. In *Object Mode*, you create 3D objects and move them as a whole by pressing G. You also assign materials to objects in this mode. You use *Edit Mode* to edit the shape of the objects.

menu that opens, select *Add Modifier | Bevel* to add a modifier to the connected layers. Leave the *Amount* parameter at 0.1, and increase the *Segments* parameter to 6 (Figure 5).

While still in *Object Mode*, smooth the bevel by going to *Object | Shade Smooth* (top left in the viewport). Position the bevel and the camera so that the bevel fills the entire viewport. If you want, you can save this setup as the new default by selecting *File | Defaults | Save Startup File*.

Make a Bottle

To get started on rendering your bottle, simply use the cube already present in the scene. In *Object Mode*, you can scale the cube along the z-axis by pressing S,Z until you get the desired bottle height (minus a cap on top – that step comes later). Next, move your bottle – you will adjust the bottle shape later – to the front view (press 1). Then press G,Z until the bottle's base rests exactly on the bevel plane. Now press the tab key to switch to *Edit Mode*, select the top square of the cube using *Face select*, and scale the top square down to create a flat-topped pyramid (Figure 6).

Now toggle back to *Object Mode* using the tab key and assign a material to the pyramid by selecting a glass shader under *Properties | Material Prop-*

erties on the far right. Mouse over *Surface | Principled BSDF* and change the option to *Glass BSDF*. Leave the *Roughness* at the default value, and change the *IOR* (Index of Refraction) parameter to 1,330 for glass. As soon as you select *Viewport Shading | Rendered*, the cube becomes transparent.

Toggle to *Edit Mode* (with the tab key) and press A to select the entire cube. Next press L,P and select *Separate | Selection*. The cube's edge should now glow a bright red-orange, which will help you keep track of it later. The selected edge has become a separate object. Now switch back to *Object Mode*.

In the next step, give the wall (the edge) a thickness by applying a *Solidify* modifier. First, set *Viewport Shading | Wireframe* and then go to *Properties | Modifier Properties*. Under *Add Modifier*, select *Solidify*. As shown in Figure 7, the cube's wall now has a thickness, which you can set using the *Thickness* parameter.

Now, you need to set the camera and the image dimensions by going to *Properties | Output Properties* (on the right). Set *Resolution | X* to something like 925px and *Resolution | Y* to 1080px. Then, right-click to select the camera. You can do this either in the viewport or in the *Outliner*.

To configure the camera, set *Properties | Object Data Properties* (the second icon at the bottom with the green camera) to *Focal Length 35mm*. Then use G to move the viewport up so that there is enough space for the bottle's cap.

Add Perfume

After you have modeled your bottle, you next need to fill the bottle with perfume. Create a new cube in the front view (1) using Shift+A and position the cube accordingly (Figure 8). Use G and S to move the cube until it fits exactly inside the bottle – just like real perfume.

Figure 7: Wall options: You can assign the cube wall a thickness, as well as choose a transparent material.

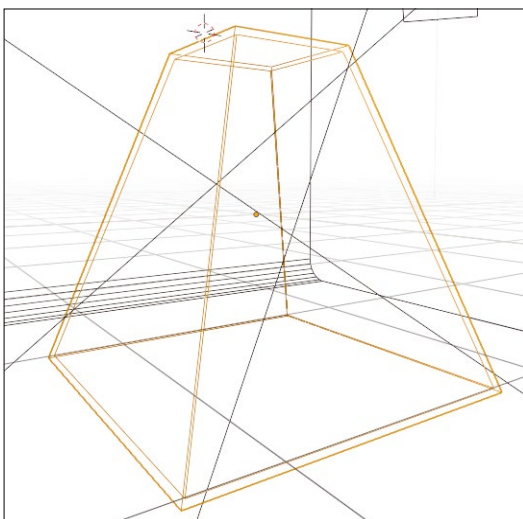
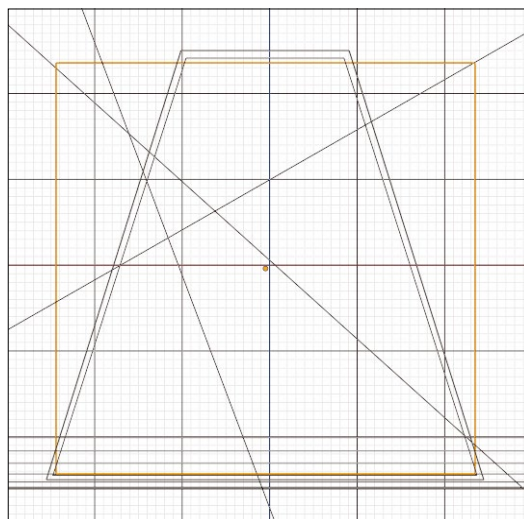


Figure 8: A second cube inside the perfume bottle forms the space filled by the liquid.



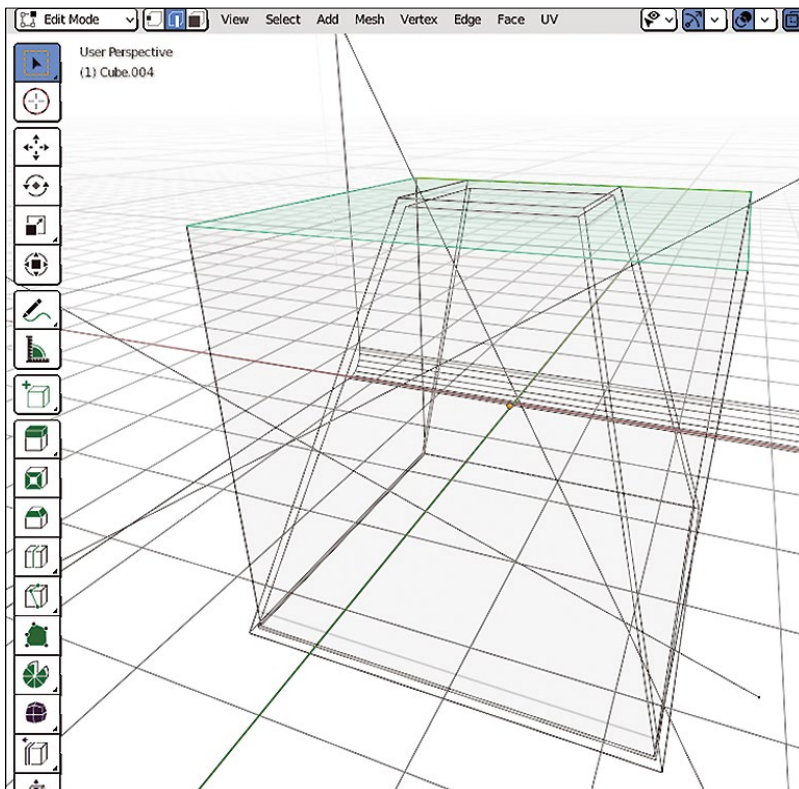


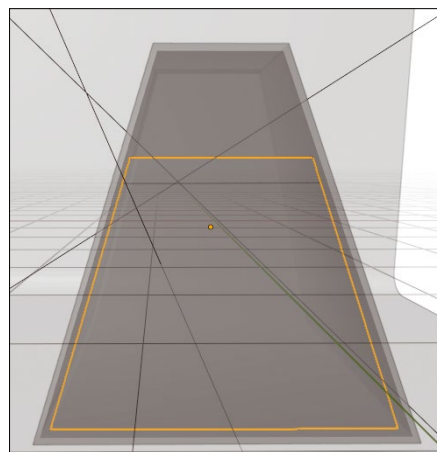
Figure 9: Select the four top edges of the cube and adjust them to the size of the perfume bottle.

To adapt the cube shape (perfume) to the flat-topped pyramid shape (bottle), follow the same steps you used to scale the bottle. In User Perspective view (on the left), select the four upper edges of the cube with *Edit Mode | Edge select* and scale them until they fit exactly into the bottle (Figure 9).

Press *1* to activate the front view and scale the selected edges so that they fit exactly inside the bottle. Then, in *Vertex select* mode, adjust the top edges with a combination of scaling (*S*) and moving (*G*) so that they look like Figure 10.

Now you are ready to assign a material to the perfume cube. Choose a trans-

Figure 10: The perfume does not have to completely fill the inside of the bottle. Leave approximately the top third empty.



parent, slightly amber material following the exact steps you did for the bottle (*Properties | Material Properties | New*). Under *Surface*, select *Glass BSDF*, and select a slightly yellowish-orange color.

The perfume in the bottle currently looks too dark, so set the gamma to 1.8 via *Properties | Render Properties | Color Management | Gamma*. Now the perfume, as well as the whole scene, should appear much brighter. Use the top view (*7*) to check whether the perfume is centered exactly in the bottle; correct the position with *G* if necessary.

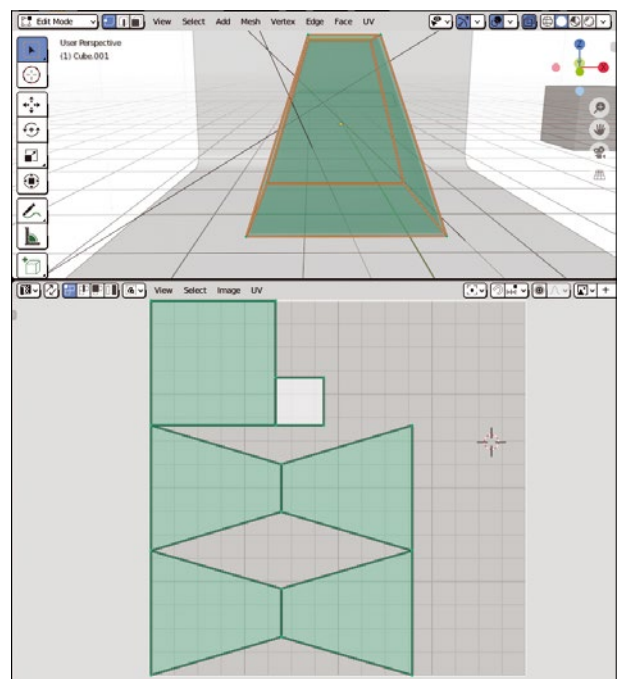
Label It

Your perfume bottle now needs a label on the front of the bottle. This process, known as *UV Mapping*, seems a little complicated at first. However, once you've done it a few times, it turns out not to be that difficult. Working in parallel, you will need to use Blender and a graphics program that supports layers to make your label. (I used Gimp for this tutorial.)

Use the tab key to switch to *Object Mode* and press *A* to select the bottle without the perfume. Then press *Ctrl+A* and select *Apply | Scale* – you must do this because you have scaled the cube. Call *Ctrl+E* and run *Mark Seam*. This will make the edges of the bottle appear a reddish orange. Next, press *U* and *Unwrap*, which unwraps the framework of the bottle on a plane, similar to unfolding a paper cube.

Next, you'll use the *UV Editor* to export the *UV Layout*, and then edit the label in your graphics

Figure 11: The bottle modeled as a 3D object with the UV layout rolled out flat below it.



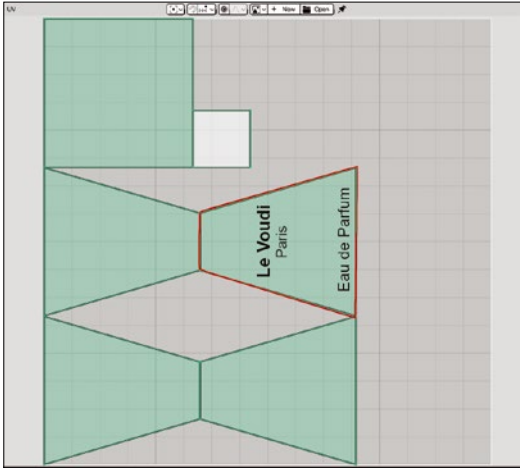


Figure 12: You can create the bottle's label in an image editor such as Gimp.

program and import the finished layout back into Blender. To get to the *UV Editor*, click on the small clock icon at bottom left in the Timeline editor. Change *Timeline* to *UV Editor* and drag the viewport up a bit (Figure 11).

Select the *UV Editor* and run *UV | Export UV Layout*. Save the layout as a PNG and then import it into your graphics program as the bottom layer. You now insert your desired label text in the graphics program. Make sure to rotate the text's orientation to 90 degrees (Figure 12).

Next, delete the bottom layer containing the *UV Layout* or temporarily hide the layer by pressing the eye icon. Once you've done this, export the label back to Blender. Keep the graphics program with the label file open in case you need to make corrections. If necessary, you can easily restore the deleted *UV layout* with *Ctrl+Z* and continue working with it.

To reimport the label text file into Blender, switch from the *UV Editor* to the *Shader Editor* and create the necessary nodes as shown in Figure 13. This relatively complicated setup is due to the bottle's transparent material. Once you've successfully added the nodes, you can view the label on the bottle. If the label is not quite right, switch back to the graphics program and edit the text.

Put a Cap on It

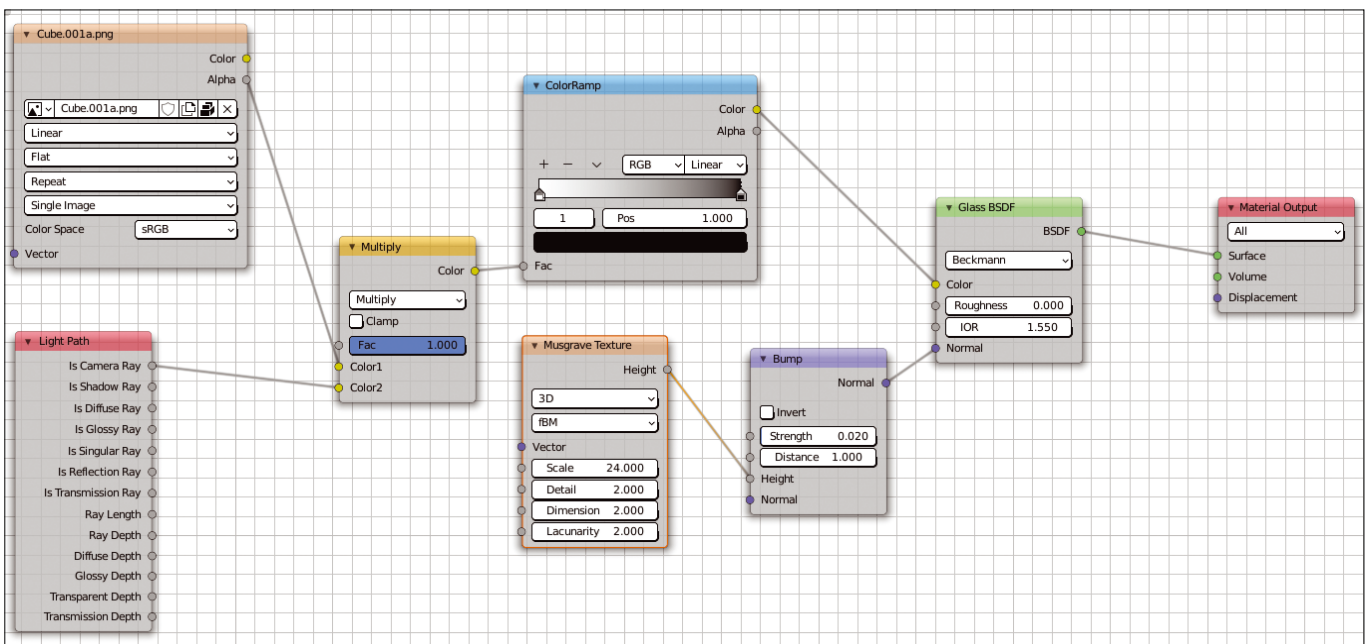
You can easily create a cap for your bottle using a black cube and a bevel modifier to round off the cube's edges. In *Object Mode*, position the 3D cursor (Figure 2) over the bottle, and create a cube with *Ctrl+A*. Press *S* to scale the cube so that it fits nicely on the bottle.

Then create a new material and color it black. This time, choose the *Principled BSDF* option. Now switch to *Properties | Modifier Properties* and use *Add Modifier | Bevel* to create a bevel modifier and round off the cap's edges. Leave the *Amount* at the default value, and increase the number of *Segments* to 6. Now apply *Object | Shade Smooth* to complete your bottle cap.

Insert Tube

Finally, create the fine tube used to connect the atomizer (hidden by the cap) to the perfume in the bottle. Simply create a path in *Object Mode* with *Ctrl+A* and use *Add | Curve | Path*. Move or rotate the path with *G* and *R* until it is vertical. Then switch to *Edit Mode* and move the lowest control point slightly to the right to create a slight curve in the tube.

Figure 13: Setting up the nodes in the *Shader Editor*. The nodes make the material of the bottle appear transparent.



Once you are satisfied with the curve, add some volume to the path by selecting *Properties | Object Data Properties | Geometry | Bevel | Round*. Increase the *Resolution* parameter to 4. Finally, the tube needs to be transparent. You can use the same material as you used for the glass bottle. As a final step, position the tube in the center of the bottle with G.

All done! Figure 14 shows the completed perfume bottle.

Conclusions

The longer you work with Blender, the more intuitive the above steps will become. Considering everything possible with Blender, this perfume bottle tutorial is only a modest beginning. If you get stuck as you venture forth with Blender, don't hesitate to ask for help, perhaps on a Blender forum [2], or look for a Blender Meetup [3] where users get together in person and share

their experiences and techniques. My final advice: Don't give up too quickly! ■■■

Info

[1] Download Blender:

<https://www.blender.org/download>

[2] Blender support:

<https://www.blender.org/support/>

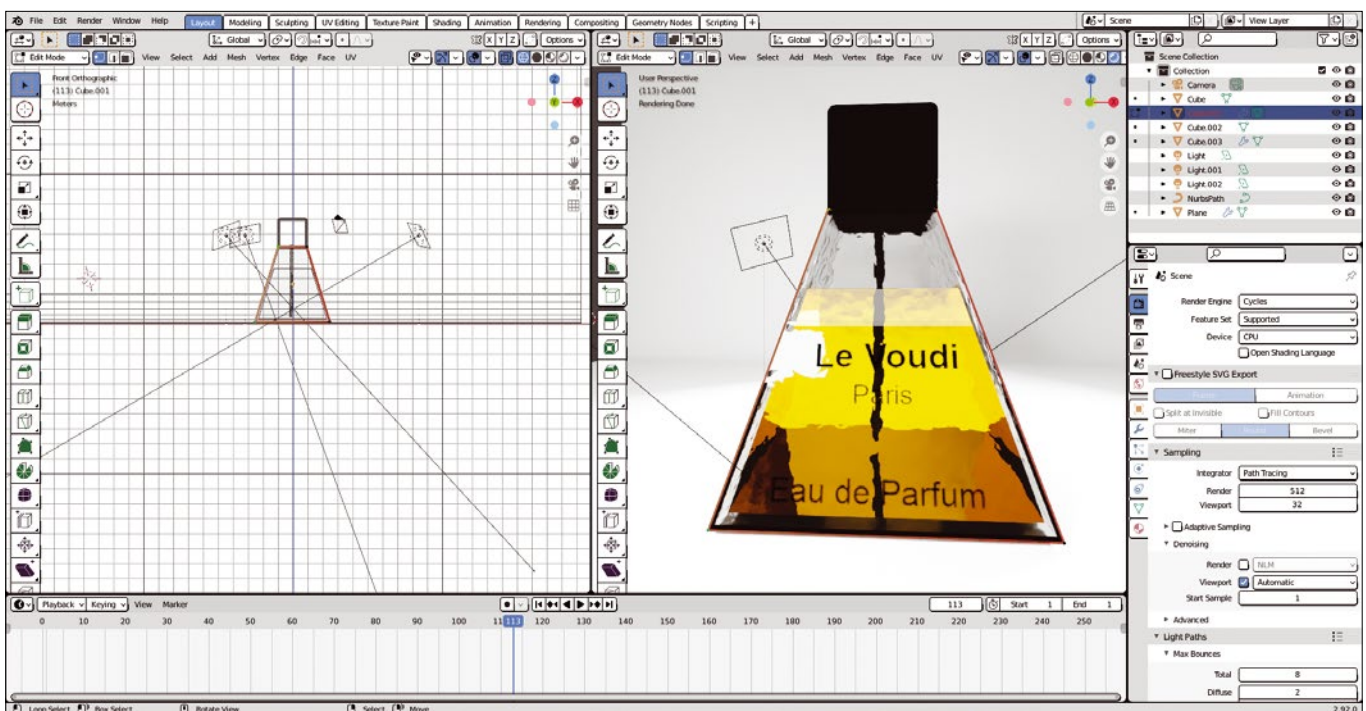
[3] Blender 3D Meetups:

<https://www.meetup.com/topics/blender-3d>

The Author

Claus Cynry has been working with graphics since 1996. He has used Linux since 2002 with Ubuntu Mate 20.04 currently installed. Claus has worked with Blender intensively since 2018. He enjoys playing guitar, blogging, painting, and photography.

Figure 14: The finished product.



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.



#252/November 2021

Locked Down!

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

On the DVD: Debian 11 and Redcore Linux 2101



#251/October 2021

Linux From Scratch

Building an operating system is not like compiling a desktop app. You'll need to create a complete development environment – and if you follow the steps carefully, you'll emerge with a deeper understanding of Linux.

On the DVD: Linux Mint 20.2 Cinnamon Edition and Garuda Linux KDE Dr460nized Edition



#250/September 2021

Inside the Kernel

The only real way to celebrate the 30th anniversary of Linux is to write about Linux itself – not the agglomeration of software we know as a Linux distro, but the real Linux – the beating heart in the center of it all: the Linux kernel.

On the DVD: AlmaLinux Minimal 8.4 and SystemRescueCd 8.03



#249/August 2021

Turn Your Android into a Linux PC

UserLAnd lets you run Linux applications on your Android phone – all without replacing Android OS.

On the DVD: openSUSE Leap 15.3 and Kubuntu 21.04 Desktop



#248/July 2021

Brain Tools

Sometimes you want the computer to think for you, and sometimes you want the computer to make you think. This month we present a selection of free Linux tools for learning and thinking.

On the DVD: Ubuntu 21.04 and Fedora 34 Workstation



#247/June 2021

Post-Quantum Encryption

Quantum computers are still at the experimental stage, but mathematicians have already discovered some quantum-based algorithms that will demolish the best of our current encryption methods. What better time to look for quantum encryption alternatives?

On the DVD: Knoppix 9.1 and ZORIN OS 15.3 Core

FEATURED EVENTS



Users, developers, and vendors meet at Linux events around the world. We at *Linux Magazine* are proud to sponsor the Featured Events shown here. For other events near you, check our extensive events calendar online at <https://www.linux-magazine.com/events>.

If you know of another Linux event you would like us to add to our calendar, please send a message with all the details to events@linux-magazine.com.

NOTICE

Be sure to check the event website before booking any travel, as many events are being canceled or converted to virtual events due to the effects of COVID-19.

DeveloperWeek Global

Date: December 7-8, 2021

Location: Virtual Event

Website: <https://www.developerweek.com/global/conference/enterprise/>

DeveloperWeek Global: Enterprise Conference invites more than 3,000 enterprise dev professionals to converge for a 2-day virtual conference and expo, featuring technology innovations and trends that corporations need to know about. Topics will include: DevSecOps, Organizing Dev Teams, DevTech Trends, Microservices, Containers, Kubernetes, and more.

Cloud Expo Europe

Date: December 8-9, 2021

Location: Frankfurt, Germany

Website: <https://www.cloudexpoeurope.de/en>

Cloud Expo Europe is back! If you design, manage, or build digital transformation initiatives and technology architecture, you should join us at Messe Frankfurt on the 8-9 of December 2021! It is a unique opportunity to meet with suppliers, listen, and seek advice from industry experts free of charge. Your Cloud Expo Europe ticket also gives you free access to co-located events Big Data & AI World Frankfurt & "Data Centre World Frankfurt.

Events

DeveloperWeek Global: Enterprise	December 7-8	Virtual Event	https://www.developerweek.com/global/conference/enterprise/
Cloud Expo Europe	December 8-9	Frankfurt, Germany	https://www.cloudexpoeurope.de/en
Big Data World Frankfurt	December 8-9	Frankfurt, Germany	https://www.bigdataworldfrankfurt.de/
Data Centre World 2021	December 8-9	Frankfurt, Germany	https://www.datacentreworld.de/data-centre-world-2020
KubeCon + CloudNativeCon + Open Source Summit China	December 9-10	Virtual Event	https://www.lfasiallc.com/kubecon-cloudnativecon-open-source-summit-china/
Open Source Summit Japan	December 14-15	Virtual Event	https://events.linuxfoundation.org/
Open Compliance Summit	December 16	Virtual Event	https://events.linuxfoundation.org/
DeveloperWeek	February 2-4	Virtual Event	https://www.developerweek.com/
SCALE 19x	March 3-6	Pasadena, California	https://register.socallinuxexpo.org/reg6/
Open Networking & Edge Summit Europe 2022	March 8-9	Antwerp, Belgium	https://events.linuxfoundation.org/about/calendar/
CloudFest 2022	March 22-24	Europa-Park, Germany	https://registration.cloudfest.com/registration?code=CFMEDIA22
KubeCon + CloudNativeCon	May 17-20	Valencia, Spain	https://events.linuxfoundation.org/
OpenJS World 2022	June 7-8	Austin, Texas	https://events.linuxfoundation.org/openjs-world/

Contact Info

Editor in Chief

Joe Casad, jcasad@linux-magazine.com

Copy Editors

Amy Pettie, Aubrey Vaughn

News Editor

Jack Wallen

Editor Emerita Nomadica

Rita L Sooby

Managing Editor

Lori White

Localization & Translation

Ian Travis

Layout

Dena Friesen, Lori White

Cover Design

Lori White

Cover Image

© Isselee Eric Philippe, 123RF.com

Advertising

Brian Osborn, bosborn@linuxnewmedia.com
phone +49 8093 7679420

Marketing Communications

Gwen Clark, gclark@linuxnewmedia.com
Linux New Media USA, LLC
4840 Bob Billings Parkway, Ste 104
Lawrence, KS 66049 USA

Publisher

Brian Osborn

Customer Service / Subscription

For USA and Canada:

Email: cs@linuxpromagazine.com

Phone: 1-866-247-2802

(Toll Free from the US and Canada)

For all other countries:

Email: subs@linux-magazine.com

www.linuxpromagazine.com – North America

www.linux-magazine.com – Worldwide

While every care has been taken in the content of the magazine, the publishers cannot be held responsible for the accuracy of the information contained within it or any consequences arising from the use of it. The use of the disc provided with the magazine or any material provided on it is at your own risk.

Copyright and Trademarks © 2021 Linux New Media USA, LLC.

No material may be reproduced in any form whatsoever in whole or in part without the written permission of the publishers. It is assumed that all correspondence sent, for example, letters, email, faxes, photographs, articles, drawings, are supplied for publication or license to third parties on a non-exclusive worldwide basis by Linux New Media USA, LLC, unless otherwise stated in writing.

Linux is a trademark of Linus Torvalds.

All brand or product names are trademarks of their respective owners. Contact us if we haven't credited your copyright; we will always correct any oversight.

Printed in Nuremberg, Germany by hofmann infocom GmbH.

Distributed by Seymour Distribution Ltd, United Kingdom

LINUX PRO MAGAZINE (ISSN 1752-9050) is published monthly by Linux New Media USA, LLC, 4840 Bob Billings Parkway, Ste 104, Lawrence, KS 66049, USA. Periodicals Postage paid at Lawrence, KS and additional mailing offices. Ride-Along Enclosed. POSTMASTER: Please send address changes to Linux Pro Magazine, 4840 Bob Billings Parkway, Ste 104, Lawrence, KS 66049, USA.

Published monthly in Europe as Linux Magazine (ISSN 1471-5678) by: Sparkhaus Media GmbH, Bialasstr. 1a, 85625 Glonn, Germany.



Authors

Erik Bärwaldt	30, 52
Zack Brown	12
Bruce Byfield	6, 22, 26, 36
Joe Casad	3
Mark Crutch	65
Claus Cynry	88
Karsten Günther	68
Jon "maddog" Hall	67
Stuart Houghton	56
Charly Kühnast	34
Christoph Langner	74
Rubén Llorente	16
Vincent Mealing	65
Brooke Metcalfe	60
Pete Metcalfe	60
Graham Morrison	80
Dmitri Popov	76
Mike Schilli	48
Ferdinand Thommes	42
Jack Wallen	8

United States Postal Service Statement of Ownership, Management, and Circulation

1. Publication Title: Linux Pro Magazine	15. Extent and Nature of Circulation	Avg. No. Copies Each Issue During Preceding 12 Months	No. Copies of Single Issue Published Nearest to Filing Date
2. Publication No.: 1752-9050	a. Total Number of Copies (Net Press Run)	4675	4775
3. Filing Date: 10/07/2021	b. (1) Paid Outside-County Mail Subscriptions	951	957
4. Issue Frequency: Monthly	b. (2) Paid In-County Subscriptions	0	0
5. No. of Issues Published Annually: 12	b. (3) Sales Through Dealers & Carriers, Street Vendors, Counter Sales	1744	1818
6. Annual Subscription Price: \$124.95	b. (4) Other Classes Mailed Through the USPS	20	0
7. Complete Mailing Address of Known Office of Publication: 4840 Bob Billings Pkwy, Ste 104, Lawrence, KS 66049; Contact Person: Brian Osborn; Telephone: 785-856-3080	c. Total Paid Distribution	2715	2775
8. Complete Mailing Address of Headquarters or General Business Office of Publisher: 4840 Bob Billings Pkwy, Ste 104, Lawrence, KS 66049	d. (1) Outside-County	16	15
9. Full Names and Complete Mailing Addresses of Publisher, Editor, and Managing Editor: Brian Osborn, Publisher, 4840 Bob Billings Pkwy, Ste 104, Lawrence, KS 66049	d. (2) In-County	0	0
10. Owner: Linux New Media USA, LLC, 4840 Bob Billings Pkwy, Ste 104, Lawrence, KS 66049; Stockholders: Sparkhaus Media GmbH, Bialasstr. 1a, 85625 Glonn, Germany	d. (3) Other Classes Mailed Through the USPS	0	0
11. Known Bondholders, Mortgagees, and Other Security Holders Owning or Holding 1 Percent or More of Total Amount of Bonds, Mortgagees, or other Securities: None	d. (4) Free Distribution Outside the Mail	0	0
12. Tax Status: Has not changed during preceding 12 months	e. Total Free Distribution	16	15
13. Publication Title: Linux Pro Magazine	f. Total Distribution	2731	2790
14. Issue Date for Circulation Data: September 2021	g. Copies Not Distributed	1794	1870
I certify that all information furnished on this form is true and complete. Gwen Clark, Business Manager, 10/07/2021	h. Total	4525	4660
	i. Percent Paid Circulation	99.41%	99.46%
	16. Paid Electronic Copies		
	a. Paid Electronic Copies		
	b. Total Paid Print Copies (Line 15c) + Paid Electronic Copies (Line 16a)	2715	2775
	c. Total Print Distribution (Line 15f) + Paid Electronic Copies (Line 16a)	2731	2790
	d. Percentage Paid (Both Print & Electronic Copies (16b divided by 16c x 100)	99.41%	99.46%

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.

Issue 254 / January 2022

Linux on Mobile

Open source developers have labored for years to bring Linux to the smartphone. PostmarketOS is gaining attention as a cross-platform alternative modeled on traditional Linux distros.

Approximate

UK / Europe	Dec 04
USA / Canada	Dec 31
Australia	Jan 31

On Sale Date

Please note: On sale dates are approximate and may be delayed because of logistical issues.

**Preview Newsletter**

The Linux Magazine Preview is a monthly email newsletter that gives you a sneak peek at the next issue, including links to articles posted online.

Sign up at: <https://bit.ly/Linux-Update>

Lead Image © digitalshape and wannawit vachirakhueankhan, 123RF.com



Custom Tailored Suit

Select essential parts for your mobile computing needs



Processor
Intel or AMD CPU



Graphics Card
Integrated or dedicated



System Memory
Upgrade any time



Data Storage
Fast, large and switchable



Network
Wireless, Cellular
or Gigabit LAN



100%
Linux

5

Year
Warranty



Lifetime
Support



Built in
Germany



German
Privacy



Local
Support

TUXEDO
COMPUTERS

 tuxedocomputers.com

HETZNER

NEW LOCATION IN THE USA



CLOUD SERVER

STARTING AT

\$4.61
monthly

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 →

 **+1 646 6858477**

CLOUD.HETZNER.COM