



LINUX **PRO**

MAGAZINE

ISSUE 246 – MAY 2021

Faster Startup

Tune your system for speedier boot



LINUXVOICE

XENIX: Remembering Microsoft's Unix system

Song lyrics in a terminal window

Laptop Tip: Squeeze more time out of a battery charge with auto-cpufreq

10

FOSS TOOLS REVIEWED!

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>

COOKIES AND COHORTS

Dear Reader,

Is the web changing before our eyes? We in tech media certainly talk about change all the time. (One could argue that the whole point of tech media is to talk about change.) And of course, big tech companies are *always* on the message of change. In all my years in the business, I don't think I've ever read a press release that said "Nothing happened! Everything is still the same!"

When you look at how far the tech world has come over the last generation, it would be fair to say that things must be constantly changing, or we wouldn't have come so far so fast. But given the constant rumble, it is prudent to wonder how much of that change is meaningful and real.

The big change we are hearing about this month is Google's plan to discontinue support for third-party cookies in the Chrome browser. Firefox and Apple Safari already disable third-party cookies by default, but the massive Chrome market share really tips the scales on a long-running debate about third-party cookies. It is fair to suppose that this change will mark a turning point in the business of advertising on the web.

An end to third-party cookies is certainly significant for web programmers, but will it improve the lives of everyday users? Will it mean more privacy? It depends on whom you ask – and what you call privacy.

Stopping third-party cookies (also called "tracking cookies") could certainly reduce the number of companies tracking you, but that doesn't mean you won't be receiving a similar share of targeted ads. You can bet that Google would not be taking flight with this new adventure if they didn't have a place to land. The company has been working on a new technology, which they call "Federated Learning of Cohorts" (FLoC), that they will likely rush in to fill the gap left by the sudden dearth of ad cookies. In the FLoC scheme, users are classified with other users who have similar interests (your cohort), rather than tracked individually.

It is still a little unclear how this technology will work in the real world, but it appears that an advertiser will say to Google, "Find me some users who like camping," and Google will put the advertiser's ads in the browsers of users who like camping, but the advertiser won't be able

to place its own tracking cookies on the user's computer. Some within the advertising community think this new FLoC technology will make advertisers more dependent on Google, and antitrust authorities are apparently looking into the change as a potential threat to competition.

It is fair to say, though, that the interest of advertisers, and the complex issues of antitrust, are different from questions of privacy. It is highly possible that the new technology could be better overall for user privacy and still worse for a lot of the other things we object to in the open source community, like walled gardens and monopoly control. In that case, it is really hard to recommend it, but we can still keep watch and hope it will at least help to end the ridiculous excesses of third-party cookies.

To hear it from Google, FLoC is all about privacy. In fact, they have made it a central pillar of a larger strategy that they call their Privacy Sandbox. But then, you could ask, what does Google actually mean by privacy? Do they think it means something kind of like "incognito," because just so you know: Google is facing a \$5 billion class-action lawsuit [1] over the so-called Incognito mode on Chrome browsers, which apparently is not so incognito – Google still tracks everything you do.

Joe

Joe Casad,
Editor in Chief



Info

- [1] "Judge Rules \$5 Billion Google Chrome Incognito Mode Lawsuit Can Go Forward" by Ron Amadeo, *Ars Technica*, March 15, 2021: <https://arstechnica.com/gadgets/2021/03/judge-rules-5-billion-google-chrome-incognito-mode-lawsuit-can-go-forward/>

ON THE COVER

16 **Battery Time**

Get more hours out of your laptop battery with auto-cpufreq and automated frequency scaling.

64 **Video Calls on a Rasp Pi**

Talk to distant relatives by hitching a Rasp Pi to your TV.

80 **XENIX**

Once upon a time, Microsoft was a serious player in the Unix scene.

92 **Lyrics in the Terminal**

You don't need to leave the Bash prompt to view the words of your favorite mumbled hits.

NEWS

08 **News**

- Mageia 8 Now Available with Linux 5.10 LTS
- Gnome 40 Beta Released
- OpenMandriva Lx 4.2 Has Arrived
- Thunderbird 78 Ported to Ubuntu 20.04
- Linux Exploit for Spectre Flaw Discovered

12 **Kernel News**

- Opening a Random Can of Worms
- Out with the Old

COVER STORIES

16 **Battery Time**

A tool called auto-cpufreq switches governors automatically to optimize battery runtime.

20 **Faster Startup**

These tweaks will help you streamline system startup.

REVIEWS

26 **Distro Walk: elementary OS**

In the past decade, elementary OS has grown from an open source project to a company with a unique business model.

IN-DEPTH

30 **File Exchange**

For the occasional local file transfer, a few simple tools can do the job quickly and efficiently.

38 **dstask**

The dstask personal tracker lets you manage your to-do list from the command line. Dstask uses Git version control to store tasks, letting you synchronize your to-do list across multiple devices.

44 **Command Line – pip3**

This increasingly popular Python installer offers a complete solution for binary packages.

48 **Charly's Column – katoolin 3**

Charly uses the katoolin 3 installation script for a targeted approach to installing his favorite Kali Linux tools.

50 **Programming Snapshot – Shell Stats**

When functions generate legions of goroutines to do subtasks, the main program needs to track and retain control of ongoing activity.

56 **Exploring Quicksort**

Rediscover the efficiency and elegance of the classic Quicksort sorting algorithm.



20 **Faster Startup**

Weary of waiting for a login window? Your driver-drenched Linux distro was configured for *all* systems, not for *your* system. The more you know about your hardware, the more you can optimize system startup.

LINUXVOICE

- 71 Welcome**
This month in Linux Voice.
- 72 Doghouse – Weather Forecast**
A recent rocket launch has maddog thinking about high performance computing and accurate weather forecasts.
- 73 ART – Another RawTherapee**
This alternative RAW converter has the potential to simplify photo editing.
- 80 Remembering XENIX**
We look at XENIX, Microsoft's lost Unix distro, and show how you can boot up XENIX in a virtual machine.
- 86 FOSSPicks**
This month Graham looks at MScSim, Ticker, vizez, and more!
- 92 Tutorial – Lyrics-in-terminal**
Follow the lyrics while you listen to your favorite songs.

MakerSpace

- 60 PyPy and Nuitka**
PyPy and Nuitka improve the performance of Python on a Raspberry Pi.
- 64 Video Calls on a Rasp Pi**
A video telephony system does not have to be complex. This project starts a phone call with just a single button press.



Manjaro KDE Plasma 20.2.1 and Clonezilla Live 2.7.1

Two Terrific Distros on a Double-Sided DVD!



Manjaro KDE Plasma 20.2.1 64-bit

Manjaro is the most popular of the distributions based on Arch Linux. In many ways, Manjaro is to Arch what Ubuntu is to Debian: a more polished derivative that has become popular in its own right. In fact, in the last few years, Manjaro has become so popular that in 2019, Manjaro GmbH, a German corporation, was founded to manage the distribution's legal and business affairs.

Much of Arch Linux's development philosophy is preserved in Manjaro. Both distributions feature rolling releases, with updates issued one package at a time rather than by general releases. However, being based on Arch, the latest Manjaro packages are usually available a few days after they appear in Arch, giving Manjaro extra time for testing. Similarly, both distributions install a minimum of packages. The main difference is that Manjaro tends to have a few more packages installed by default. Even then, users are often offered a choice rather than the distro making the decision by default. For example, during installation, users can choose between FreeOffice and LibreOffice, as well as whether or not to install non-free packages.

Both Arch and Manjaro also favor simplicity. However, the most important difference between the two is that while Arch's interpretation of that goal sometimes makes installation difficult for beginners, Manjaro features an installer comparable to that of Ubuntu that should be manageable by all levels of users. If you are curious about Arch but have had trouble installing it, then Manjaro might be a starting place for your explorations. Besides this KDE version, Manjaro also includes editions featuring Xfce and Gnome, as well as an additional nine community editions, making it a distribution to satisfy everyone.



Clonezilla Live 2.7.1 64-bit

Clonezilla Live is a combination of Clonezilla and Debian Live. However, unlike most distributions, Clonezilla Live is not a tool for everyday computers, but rather a distribution to image or clone filesystems working from an external drive.

Like Clonezilla itself, this Live version supports a wide variety of filesystems and hardware. Although Clonezilla can be run from a hard drive, it is usually more convenient as a Live device from which to boot a computer. To do so, at startup you will have set your machine to boot from it. At the bootloader, you have several hardware and software options, including resolution, speech synthesis, and specifying the local operating system. These options are detailed on the project home page (<https://clonezilla.org/clonezilla-usage/general-live-use.php>). Once Clonezilla Live is up and running, you can refer to <https://clonezilla.org/clonezilla-live-doc.php> for examples of how to restore a disk image, clone a disk, restore to multiple disks, and other operations. A particularly useful task is to create a DVD or flash drive recovery disk for emergencies. Most of the time, you will not need Clonezilla Live, but you never know when a recent version will come in handy for maintenance and troubleshooting.

*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.

Public Money

Public Code



Modernising Public Infrastructure with Free Software



Free Software Foundation Europe

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

NEWS

Updates on technologies, trends, and tools

THIS MONTH'S NEWS

- 08 • Mageia 8 Now Available with Linux 5.10 LTS
- 09 • Gnome 40 Beta Released
• OpenMandriva Lx 4.2 Has Arrived
• More Online
- 10 • Thunderbird 78 Ported to Ubuntu 20.04
• Linux Exploit for Spectre Flaw Discovered

■ Mageia 8 Now Available with Linux 5.10 LTS

NVIDIA Optimus laptop users rejoice, Mageia 8 now includes improved support, thanks to an upgraded graphics stack that includes Mesa 20.3.4 and X.Org Server 1.20.1. This upgrade improves both the AMD and NVIDIA GPU experience with the platform. For NVIDIA users, there's the new experimental mageia-prime configuration tool that makes it possible to get the most out of your NVIDIA GPU.

But the new release isn't all about the graphics stack. Anyone who begins Mageia 8 with a live instance will see faster performance, thanks to the inclusion of Zstd compression on the base file systems, and better optimizations for hardware detection. NFS file system support has also been improved, with added support for NFSv4.

Mageia also includes a new version of RPM (version 4.16.1.2) which offers a number of improvements, such as automatic SSD detection and optimization, filesystem sync at the end of transactions, SHA256 digest added to gpg-pubkey headers, support for meta dependencies, and parametric macro generators. Overall RPM should be considerably faster, thanks to several optimizations. Mageia also ships with DNF version 4.6.0.

Finally, Mageia 8 ships with the Linux Long Term Support kernel 5.10, which gives EXT4 and Btrfs file systems a performance boost, faster hibernate/resume functionality, and a number of other improvements.

Users can download three different versions of Mageia:

- KDE (https://www.mageia.org/en/downloads/get/?q=Mageia-8-Live-Plasma-x86_64.iso)
- Gnome (https://www.mageia.org/en/downloads/get/?q=Mageia-8-Live-GNOME-x86_64.iso)
- Xfce (https://www.mageia.org/en/downloads/get/?q=Mageia-8-Live-Xfce-x86_64.iso)

Find out more about the latest Mageia release from the Official release notes (https://wiki.mageia.org/en/Mageia_8_Release_Notes).



Gnome 40 Beta Released

On the heels of the alpha release of Gnome 40, the developers have announced the availability of the beta, which includes a number of improvements and bug fixes.

Of course, the biggest change to Gnome is the new horizontal Activities Overview, which makes for a much-improved workflow on the desktop. With the desktops residing at the top of the Overview, it is now easier to drag and drop an application to the specific desktop you want. It's far more intuitive and efficient. This new layout also improves usage with touch screen navigation and faster overall performance.

Another hotly anticipated change comes by way of how multi-monitor support will work with the new horizontal Activities Overview. Gnome 40 will default to only showing workspaces on the primary display, with the top bar and the Activities Overview on both displays.

Beyond the overhaul of the Activities Overview, you'll find easier workspace switching, a new Welcome dialog after major updates, better fingerprint scanner support, better window previews, a much-improved on-screen keyboard, a major reworking of the Nautilus file manager, and a default web browser (Epiphany) that received plenty of attention.

Other bits that have received attention include Disk Usage Analyzer, Font Viewer, Calculator app, Gnome Software, and Gnome Maps.



For anyone wanting to test Gnome 40, you can download and boot the installer image (https://os.gnome.org/download/40.beta/gnome_os_installer_40.beta.iso).

OpenMandriva Lx 4.2 Has Arrived

OpenMandriva is a direct descendant of the not forgotten (and much-loved) Mandriva Linux, and was the first to ever make use of the LLVM toolchain by default. With the release of Lx 4.2, OpenMandriva brings to the table a few improvements, as well as the latest release of the KDE desktop environment, and some exciting ARM news.

As for software, OpenMandriva improves the OM Welcome, which is an on-boarding tool that makes it incredibly easy for new users to do things like install software with a single click. Beyond the welcome app, OpenMandriva includes LibreOffice 7.1, Krita 4.4.2, Digikam 7.2, SMPlayer 21.1.0, VLC 3.0.12.1, Falkon browser 3.1, SimpleScreenRecorder 0.4.3, and much more. Also included with OpenMandriva is Desktop Presets, which allows users to easily customize the appearance of the Plasma Desktop to look and feel similar to other desktop environments.

Beyond software and tools, one aspect of OpenMandriva Lx 4.2 that might excite a number of users is that the port to 64-bit ARM processors is complete. At the moment, installable images are available for PinebookPro, Raspberry Pi 4B and 3B+,

Rock Pi 4A, 4B and 4C, Synquacer, Cubox Pulse, and generic UEFI compatible devices. The ARM port will make it possible for the developers to even target smartphones (such as the PinePhone).

Download the latest version of OpenMandriva from the mirror nearest your location: <http://mirror.openmandriva.org/README.txt?mirrorlist>.

Read more about this exciting announcement in the official release notes: <https://www.openmandriva.org/en/news/article/openmandriva-lx-4-2-is-out-now>.

CC BY-SA 3.0

MORE ONLINE

Linux Magazine

www.linux-magazine.com

ADMIN HPC

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

How Linux and Beowulf Drove Desktop Supercomputing

• Jeff Layton

Open source software and tools, the Beowulf Project, and communities changed the face of high-performance computing.

ADMIN Online

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

HTML to Database with a Perl Script

• Thomas Valentine

A Perl script strips HTML markup, creating text files, and makes each file an entry in a database.

Managing Access Credentials

• Erik Bärwaldt

Most Internet services require password-protected individual accounts. A password manager can help you keep track of all your access credentials.

Hardware Suitable for Cloud Environments

• Martin Loschwitz

If you want to build scalable clouds, you need the right hardware, but which servers are the right choice as controllers, storage, and compute nodes?



Thunderbird 78 Ported to Ubuntu 20.04

Ubuntu 20.04.2 has already shipped and the developers found themselves in a tricky position with Thunderbird, the popular open source email client. Essentially they had two choices:

- Backport individual security fixes to Thunderbird 68.
- Port the latest version.

One of the most important aspects of Ubuntu is stability. Because of this, the platform doesn't generally ship with the latest releases of software. And initially Ubuntu 20.04 shipped with Thunderbird 68. However, because that version is no longer supported by upstream, it would no longer be receiving security updates.

That's a problem.

So the Ubuntu developers had to choose between two options. Fortunately, they opted to go with porting the latest version of Thunderbird. This will cause an issue for some users, as not all Thunderbird extensions will work with the latest release. For example, now that Thunderbird has native encryption, the Enigmail extension is not only redundant, it simply won't install on the client. This means users accustomed to Enigmail must now get up to speed on Thunderbird's built-in encryption technology.

For those users who prefer installing applications via standard .deb packages, you won't be forced to use either a snap or flatpak version of Thunderbird.

For more information on the decision, check out this post by Ubuntu's own, Oliver Tilloy: <https://discourse.ubuntu.com/t/thunderbird-its-update/20819?u=d0od>.



CC BY-SA 3.0

Linux Exploit for Spectre Flaw Discovered

Spectre has been in the public knowledge base since January 9, 2018, when Intel was forced to go public with the information that all of their CPUs (since 1995) can allow applications to be tricked into leaking information they hold in memory. Since then, there have been a number of Spectre exploits, across all operating systems on Intel hardware.

Recently a French researcher, Julien Voisin, announced (<https://dustri.org/b/spectre-exploits-in-the-wild.html>) he was able to view the contents of /etc/shadow on a vulnerable Fedora system, thereby verifying this new Spectre exploit.

This vulnerability works in four stages:

- Finding the superblock of a file.
- Finding the inode of the file to be dumped.
- Finding the corresponding page address.
- Dumping the contents of the file.

According to Voisin, this exploit had a 0 detection rate before he published his announcement.

Of this vulnerability, Andrew Cooper, senior software engineer, Citrix, had this to say, "SMAP prevents supervisor code from accessing user memory operands outside of explicitly permitted areas." Cooper continues, "This is enough to prevent the cacheline fill (of a userspace pointer) and break the covert channel. A more sophisticated attack could use a supervisor pointer, e.g. the directmap mapping, or one of a multitude of other covert channels to transmit the same data, which is a higher barrier, but definitely not impossible."

The new Spectre exploit looks to have been created and distributed by a company called Immunity Inc, and VirusTotal allows the downloading of the exploit for a fee.

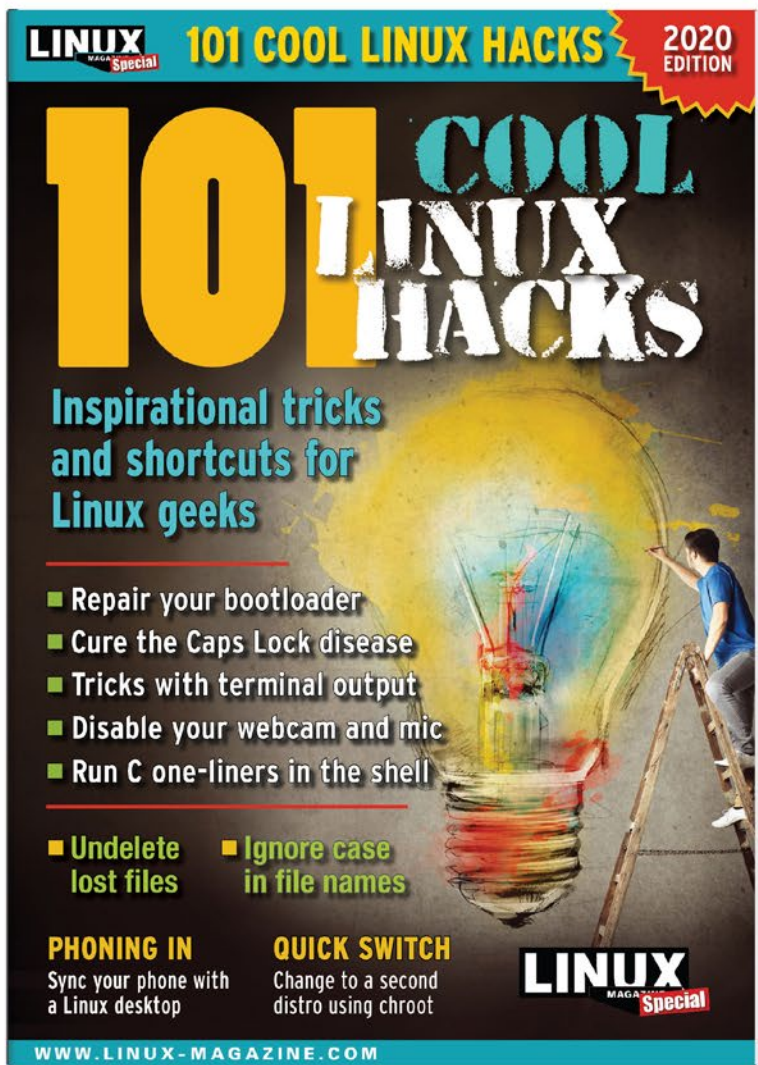


**Get the latest news
in your inbox every
two weeks**

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

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

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**.

Opening a Random Can of Worms

Torsten Duwe was mad as hell, and he wasn't going to take it anymore! Or at least, he had certain objections to `/dev/random`, which he felt should be addressed. In particular, one of the main points of random numbers in the Linux kernel is to support system security. Torsten pointed out that "Input entropy amounts are guesstimated in advance, obviously much too conservatively, compiled in and never checked thereafter; the whitening is done using some home-grown hash function derivative and other non-cryptographic, non-standard operations."

He also remarked with restraint and decorum, that "meanwhile there's quite a maintenance backlog; minor fixes are pending, medium-sized cleanups are ignored and major patch sets to add the missing features are not even discussed."

Torsten said he was in favor of bringing the Linux kernel up to some sort of standards compliance with regards to random numbers, preferably obtaining official certification from one of the organizations that did that sort of thing. But he said he'd settle for `/dev/random` simply being a reliable source of entropy, even without any certification.

In posting this message, Torsten did an end run around the official `/dev/random` maintainer, Theodore Y. Ts'o, sending the email directly to Linus Torvalds and asking for a new maintainer. Apparently Torsten felt that Ted had been AWOL on `/dev/random` patches and needed to be replaced.

Jason A. Donenfeld volunteered to be the new official `/dev/random` maintainer, though he also remarked, "I think Ted's reluctance to not accept the recent patches sent to this list is mostly justified, and I have no desire to see us rush into replacing `random.c` with something suboptimal."

And Ted also said, "I do plan to make time to catch up on reviewing patches this cycle. One thing that would help me is if folks (especially Jason, if you would) could start with a detailed review of Nicolai's patches. His incremental ap-

proach is I believe the best one from a review perspective, and certainly his cleanup patches are ones which I would expect are no-brainers."

Jason agreed and the discussion seemed to end there. A couple of weeks later, Marcelo Henrique Cerri asked about the status of the `/dev/random` patch review process, remarking, "I don't believe Torsten's concerns are simply about *applying* patches but more about these long periods of radio silence. That kills collaboration and disengage[s] people. More than simply reviewing patches I would expect a maintainer to give directions and drive the community. Asking Jason to review Nicolai's patches was a step towards that, but I believe we still could benefit from better communication."

Torsten wholeheartedly agreed, saying, "Exactly. I could live with replies in the style of 'old' Linus like: 'Your code is crap, because it does X and Y'. Then I knew how to proceed. But this extended silence slows things down a lot."

Torsten was also not satisfied with having Jason review the patches. He said, "Jason seems to narrow the proposed changes down to 'FIPS [Federal Information Processing Standard] certification', when it actually is a lot more. I think his motivation suffers because of his personal dislike."

At this point Petr Tesarik of SUSE joined the discussion, saying, "Upfront, let me admit that SUSE has a vested interest in a FIPS-certifiable Linux kernel."

He proceeded to say:

"However, it seems to me that nobody can be happy about keeping the current status quo forever. Even in the hypothetical case that the RNG [Random Number Generator] maintainer rejected the whole idea merely because it makes it possible to achieve NIST compliance, and he de-tests standards compliance, it would still be better than no decision at all. The silence is paralyzing, as it blocks any changes in upstream, while also making it difficult to maintain an out-of-tree implementation that aims at becoming upstream eventually.

“The only option ATM [at the moment] is a fork (similar to what the Xen folks did with XenLinux many years ago). IOW [in other words] the current situation demotivates contributors from being good citizens. I hope we can find a better solution together.”

Jason voiced his opinion of this quite clearly, replying, “just because you have a ‘vested interest’, or a financial interest, or because you want it does not suddenly make it a good idea. The idea is to have good crypto, not to merely check some boxes for the bean counters.”

But Stephan Müller disagreed with Jason, saying, “using non-assessed cryptography? Sounds dangerous to me even though it may be based on some well-known construction.”

Stephan went on, “I thought Linux in general and crypto in particular is about allowing user (or the vendor) to decide about the used algorithm. So, let us have a mechanism that gives them this freedom.”

Jason pointed out that “assessed” was not the same as FIPS certification, which was what Petr had advocated. Jason accused Stephan of intentionally conflating the idea of rejecting FIPS certification with the idea of rejecting all possible mechanisms for confirming good entropy.

And to clarify his position, Jason added, “new constructions that I’m interested in would be formally verified (like the other crypto work I’ve done) with review and buy-in from the cryptographic community, both engineering and academic. I have no interest in submitting ‘non-assessed’ things developed in a vacuum, and I’m displeased with your attempting to make that characterization.”

Jason said that regardless of FIPS certification, he wanted rigorous confirmation of correctness in any proposal he made or code he submitted. He added, “The current RNG is admittedly a bit of a mess, but at least it’s a design that’s evolved. Something that’s ‘revolutionary’, rather than evolutionary, needs considerably more argumentation.”

Petr came back into the conversation, trying to defuse some of the tensions by pointing out that by originally admitting SUSE’s vested interest, he was “just trying to be honest about our motivations.” He added, “I’m a bit sad that this discus-

sion has quickly gone back to the choice of algorithms and how they can be implemented. The real issue is that the RNG subsystem has not developed as fast as it could. This had not been much of an issue as long as nobody was really interested in making any substantial changes to that code, but it is more apparent now. Torsten believes it can be partly because of a maintainer who is too busy with other tasks, and he suggested we try to improve the situation by giving the RNG-related tasks to someone else. I have not seen a clear answer to this suggestion, except Jason offering his helping hand with Nicolai’s cleanup patches, but nothing wrt [with reference to] Stephan’s patches. So, what is the plan?”

Jason picked up on Petr’s statement that he was sad the discussion had become about algorithms. Jason replied that this was directly relevant to the whole conversation, saying “why are you sad? You are interested in FIPS. FIPS indicates a certain set of algorithms. The ones most suitable to the task seem like they’d run into real practical problems in the kernel’s RNG. That’s not the `_only_` reason I’m not keen on FIPS, but it does seem like a very basic one.”

In a subsequent email replying to himself, Jason went on, “in working through Nicholai’s patches (an ongoing process), I’m reminded of his admonishment in the 00 cover letter that at some point `chacha20` will have to be replaced, due to FIPS. So it seems like that’s very much on the table.” And he further clarified, “If you want to make lots of changes for cryptographic or technical reasons, that seems like a decent way to engage. But if the motivation for each of these is the bean counting, then again, I’m pretty wary of churn for nothing. And if that bean counting will eventually lead us into bad corners, like the concerns I brought up about FPU usage in the kernel, then I’m even more hesitant. However, I think there may be good arguments to be made that some of Nicholai’s patches stand on their own, without the FIPS motivation. And that’s the set of arguments that are compelling.”

At this point Pavel Machek joined the discussion, responding to the whole premise of the conversation – was it even necessary to change `/dev/random` at all? As he put it, “does RNG subsystem need to evolve? Its task is to get random numbers.

Does it fail at the task?" And on the side of leaving it as is, he remarked that the "problem is, random subsystem is hard to verify, and big rewrite is likely to cause security problems."

Sandy Harris responded to that point in particular:

"Parts of the problem, though, are dead easy in many of today's environments.

"Many CPUs, e.g. Intel, have an instruction that gives random numbers. Some systems have another hardware RNG. Some can add one using a USB device or Denker's Turbid (<https://www.av8n.com/turbid/>). Many Linux instances run on VMs so they have an emulated HWRNG using the host's /dev/random.

"None of those is necessarily 100% trustworthy, though the published analysis for Turbid & for (one version of) the Intel device seem adequate to me. However, if you use any of them to scribble over the entire 4k-bit input pool and/or a 512-bit Salsa context during initialisation, then it seems almost certain you'll get enough entropy to block attacks.

"They are all dirt cheap so doing that, and using them again later for incremental squirts of randomness, looks reasonable.

"In many cases you could go further. Consider a system with an intel CPU and another HWRNG, perhaps a VM. Get 128 bits from each source & combine them using the 128-bit finite field multiplication from the GSM authentication. Still cheap & it cannot be worse than the better of the two sources. If both sources are anywhere near reasonable, this should produce 128 bits of very high grade random material, cheaply.

"I am not suggesting any of these should be used for output, but using them for initialisation whenever possible looks obvious to me."

At this point the conversation came to an abrupt halt, at least on the Linux Kernel Mailing List. However, a month later, Stephan posted a new patch (or rather, version 38 of his ongoing work), saying, "The following patch set provides a different approach to /dev/random which is called Linux Random Number Generator (LRNG) to collect entropy within the Linux kernel. It provides the same API and ABI and can be used as a drop-in replacement."

He listed some technical advantages over the existing /dev/random implemen-

tation, including a significant speed increase and a variety of other benefits. But there was no response. No further discussion at all.

It's unclear what that means, especially since one of Torsten's original complaints was that patches from Stephan were not being reviewed.

It's not surprising that /dev/random would be such a controversial subject that inspired such heated discussion. A large portion of Linux security rests on that particular feature. And as Pavel pointed out, it will always be difficult to confirm that changes to /dev/random result in entropy that is actually useful, as opposed to easily predictable by hostile actors.

Ultimately, if /dev/random can't be changed because it's too important, that in itself will be considered a bug by Linus Torvalds and other top dogs, and someone will eventually have a brain wave and figure out how to split /dev/random up into pieces that are more easily maintained. But until then, debates and disputes like the above will be a necessary outgrowth of the problem.

Out with the Old

An interesting aspect of Linux kernel development is how much effort is put into removing existing features. In most open source projects, features are generally added and never removed. In Linux, removing features has become perfectly standardized – specifically, support for old hardware that no one uses anymore.

Linus Torvalds has been fairly consistent over the years – if there is even a single user of a given hardware platform, he won't remove support for that platform. But if there are no users, he has no sentimental attachment. Linux was developed on the i386 platform, and Linus said at the time that it would probably never run on anything else. In 2012, support for i386 was ripped off like an old bandage, with never a backward glance.

Why? There were no i386 systems left in the world. Or at least none that were running Linux. Or at least, at the critical moment no one stepped forward to say they still needed Linux support for their i386 systems. If they had, we'd have i386 support to this very day.

It's much better to remove support for dead hardware platforms. It simplifies the whole kernel. It gets rid of special

cases that needed to be maintained. It reduces the size of the source tree as well as the compiled binary. It makes it easier for developers to maintain the tree. It makes it easier for newcomers to join in.

Sometimes the decision is easy – someone discovers that support for a given architecture has been broken for a couple years. Nobody squawked, so almost certainly nobody's using that architecture. Easy peasy.

Recently, Sam Ravnberg proposed "sunsetting" the sun4m and sun4d versions of the SPARCstation architecture from Sun Microsystems. Popular in the '90s, these versions were "then replaced by the more powerful sparc64 class of machines," he said.

Sam had done his due diligence, and he added, "Cobham Gaisler have variants of the LEON processor that runs sparc32 – and they are in production today." He proposed that Linux focus its support for LEON machines, rather than the whole set of sun4m and sun4d machines. He also pointed out that the Qemu emulator did support emulating these platforms, which meant that dropping sun4m might mean losing some testing possibilities.

Sam said the Gaisler folks were interested in putting in development work to support their LEON machines, and he pointed out that "this will only be easier with a kernel where the legacy stuff is dropped."

He asked if there were any objections to sunsetting sun4m and sun4d.

Arnd Bergmann replied, "Thank you for doing this, it looks like a very nice cleanup." Though he also acknowledged, "I have no insight on whether there are any users left that would miss it, but I'm fairly sure that there are lots of people that would rather see it gone."

Kjetil Oftung said he was sad to see these architectures go, but acknowledged, "I guess I haven't had any time to put into the sparc32 port for many years, so I guess it is time to let go."

However, Kjetil did suggest that Gaisler make some Sparc32 machines available for kernel developers to help maintain LEON support.

Sam replied to his own initial email, saying he'd gotten a few private messages from concerned citizens. One

person had said (paraphrased by Sam), that “it was better to sunset now when it is actually working, so there is a working state to return to.”

A second message, Sam said, argued for continuing to support these architectures, because, in fact, there were still a lot of those machines in existence. Maybe someone would want to use them. This message also pointed out that the NetBSD OS still supported those systems.

Sam invited more people to speak out, for or against dropping sun4m and sun4d.

John Paul Adrian Glaubitz replied, “I would personally be in favor of keeping it and I should finally get my SPARCstation 5 up and running again.”

Romain Dolbeau also said, “If there’s still a distribution willing to build for Sparc v8, then I believe the kernel should try to keep support of the relevant machine architectures if at all possible.”

Julian Calaby said that he had two SPARCstation 10s and a SPARCstation LX. He summarized his situation, saying:

“If I want to run them, assuming the hardware still works, I need to netboot them as I cannot find working, compatible HDDs for them as everything has switched to SATA or SAS.

“Then there’s the issue of finding a monitor as they’re not electrically compatible with VGA and I’m pretty sure none of the VGA compatible monitors I have or can lay hands on works with their specific sync frequencies.

“Ultimately it’s one of those things where there’s enough ‘stuff’ in the way that booting one up for fun is simply impractical and they’re old and slow enough that they’re not useful for anything else.”

Julian continued:

“The last (official) version of Debian to support Sparc32 was Etch and I believe it was one of the last ones to drop support.

“I believe that Gentoo is architecture-neutral enough that it’d work, but I believe that you’ll have to compile everything – there’ll be no pre-built anything for sparc32 – and as it’s fairly slow hardware by today’s standards, that’s going to take a long time, however you could

probably use distcc and cross-compilers to speed it up.

“Long painful story short, it’s difficult to get the hardware running, there’s practically no Linux distros that support it, and the kernel code has probably bitrotted due to lack of testing.

“As much as it pains me to say this, I think this code’s time has come and it’s time to get rid of it.

“If there were more people using it or more testing, or more distros supporting it – not just (theoretically?) working on it – then I’d be fighting to keep it.

“But there isn’t.

“I think it’s time for it to go.”

The discussion ended inconclusively – which generally means the architecture is most likely to stick around for awhile, so more people have a chance to raise objections. The most interesting aspect of this particular debate, for me, is that there are actually plenty of these machines floating around. It’s conceivable that the very effort to abandon them will inspire someone to build a giant cluster of these machines just to prove it’s worth keeping support in the kernel. ■■■

What?!

I can get my issues SOONER?

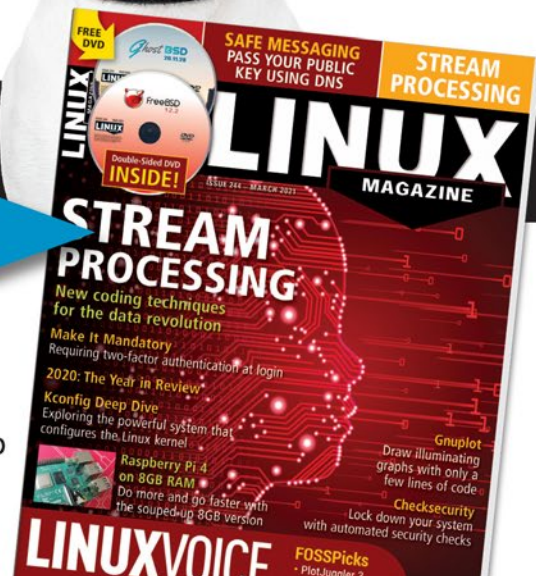


Available anywhere, anytime!

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

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

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





Optimize battery life and computing power with auto-cpufreq

Clock Speed

In Linux, the governor controls the CPU clock speed and power management. A tool called auto-cpufreq switches governors automatically to optimize battery runtime and computing power. *By Christoph Langner*

Modern CPU cores do not run constantly at the maximum possible clock speed, but instead, are capable of scaling the computing power up or down in defined steps as needed in order to save energy. On Linux, this task is performed by the CPU Frequency Scaling system (cpufreq) [1], which is built into the kernel.

Several different approaches or philosophies exist for how to accelerate the system. Is it better to immediately ramp up the clock speed to the max for a computing task, or should the system gradually increase the computing power? The CPU needs a large amount of energy at the highest clock speed, but, when it is running at a high speed, it completes the computing task more quickly, so that the CPU can go back to sleep. In contrast, the system often works more efficiently in power-saving mode, which reduces the heat loss, but the CPU has to compute for a longer time and can't go to sleep until the task is complete.

Linux deals with the conflicting goals of balancing computing performance, efficient power-saving functions, and comfortable responsiveness using six different cpufreq governors. Each of these governors uses a different algorithm for managing the CPU frequency. The governor decides when the CPU runs at what speed and sets a maximum limit. The governor options include:

- Performance – operates at the highest possible frequency, within the `scaling_max_freq` policy limit
- Powersave – operates at the lowest possible frequency, within the `scaling_min_freq` policy limit
- Userspace – allows the CPU frequency to be set in user space
- Ondemand – varies the frequency based on CPU load
- Conservative – varies the frequency based on CPU load (like the `ondemand` governor) but changes the frequency in relatively small steps, which is more efficient for some hardware and use cases
- Schedutil – accesses the CPU scheduler data directly to set the frequency based on CPU utilization

The question of which governor to put in charge is left to the distribution developer. Of course, a higher frequency means the system runs faster, which is typically the best option as long as the computer is plugged in, but for a laptop or other portable devices running on battery power, the situation is a bit more complicated. Most Linux systems don't have a built-in solution for switching between governors on the fly. For example, in our lab, a freshly installed system with Ubuntu 20.10 on a Dell XPS 9570 always used the performance governor, regardless of whether the system was connected to the power supply or on battery power.

The Linux environment supports several tools to assist with optimizing battery life (see the box entitled “TLP”), but until recently, the system has lacked an efficient, automated tool for switching between governors based on system load and battery status. The recent utility called auto-cpufreq [3] attempts to address this need.

Auto-cpufreq checks the system load and charge state (mains or battery) to decide which governor to enable. For battery operation in idle mode, or if the CPU load is

TLP

To manage a mobile computer's hunger for power, TLP [2] has established itself as a sort of standard in recent years. The program enables different CPU profiles as a function of the operating mode or turns off unneeded USB devices, Bluetooth, and WLAN where appropriate. The auto-cpufreq program provides some features that aren't available with TLP, such as the ability to change the governor automatically based on CPU load or battery status. According to the developer, auto-cpufreq is compatible with TLP and can be used on the same system.



low, auto-cpufreq enables the powersave governor. As soon as the load increases, the program switches to performance mode. When the computer is connected to the power supply, on the other hand, auto-cpufreq always leaves the system in the hands of the performance governor and enables the CPU's turbo mode, if available, so that the maximum computing power is always available.

Installation

The still quite young auto-cpufreq is not yet available from the package sources of the major distributions. However, the developer provides a cross-distribution snap package in the Snap Store [4]. Alternatively, you will find installation scripts for Debian, Red Hat, and their derivatives. Currently, the script route is recommended, because users report increased system load when installing the Snap package [5].

Use the commands in Listing 1 to set up auto-cpufreq on a freshly configured Ubuntu system. The `s-tui`, `stress`, and `i7z`

Listing 1: Install on Ubuntu

```
$ sudo apt install git s-tui stress i7z
$ git clone https://github.com/AdnanHodzic/auto-cpufreq.git
$ cd auto-cpufreq
$ sudo ./auto-cpufreq-installer
```

Listing 2: Install on Arch Linux

```
$ yay -S auto-cpufreq s-tui stress i7z
$ sudo systemctl enable --now auto-cpufreq
$ auto-cpufreq --log
```

tools are not needed by auto-cpufreq itself; however, they do help you to check the system state and the computing load later on.

Users of Arch Linux can pick up the program from the Arch User Repository (AUR). Unlike the installation script, the Arch recipe automatically sets up a systemd service, which you only need to enable and start after installation (Listing 2). The `--install` option described later in this article is not necessary on Arch.

Configuration

On an Ubuntu system, the install script simply injects the program into the system. For the configuration you then call `auto-cpufreq`, and then the program will explain its options. The application needs administrative privileges. The following command:

```
sudo auto-cpufreq -monitor
```

initially only monitors the system and shows what the program would do differently. For example, in Figure 1, the program determines that the system is regulating the CPU with the performance governor, although it is not connected to the power supply and would therefore be better off running in powersave mode.

The following command

```
sudo auto-cpufreq -live
```

lets auto-cpufreq take over the control of the CPU – but only until you abort the program again with `Ctrl + C`. The tool does

```
ubuntu@ubuntu: ~
-----
Linux distro: Ubuntu 20.10 groovy
Linux kernel: 5.8.0-25-generic

Processor: Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz
Cores: 8
Architecture: x86_64
Driver: intel_pstate

----- Current CPU states -----

CPU max frequency: 2800 MHz
CPU min frequency: 800 MHz

Usage Temperature Frequency
CPU0: 2.0% 34 °C 2220 MHz
CPU1: 0.0% 36 °C 2800 MHz
CPU2: 2.0% 34 °C 2800 MHz
CPU3: 2.0% 35 °C 2391 MHz
CPU4: 1.1% 34 °C 2263 MHz
CPU5: 3.0% 36 °C 2228 MHz
CPU6: 0.0% 34 °C 2100 MHz
CPU7: 0.0% 35 °C 2800 MHz

----- CPU frequency scaling -----

Battery is: discharging

Currently using: performance governor
Suggesting use of "powersave" governor

Total CPU usage: 1.0 %
Total system load: 0.49

Load optimal, suggesting to set turbo boost: off
Currently turbo boost is: off

-----
"auto-cpufreq" refresh in: 2
```

Figure 1: In *Monitor* mode, auto-cpufreq outputs information about the clock speed and the governor but does not change anything on the system.



not permanently change anything in the system with this configuration. You can now easily check the functionality with the `s-tui` command-line tool (Figure 2). If you switch to *Stress* mode (see the option in the middle column in Figure 2), the tool runs a stress test on all CPU cores, forcing the processor cores to speed up and the system to transition to Turbo Boost

mode if it is available [6]. Turbo Boost mode overclocks Intel CPUs for a short time, as long as the system stays within defined limits for current and temperature. The AMD counterpart is known as Turbo Core.

To see the differences in operation between auto-cpufreq and the default configuration, launch auto-cpufreq with the `--monitor`

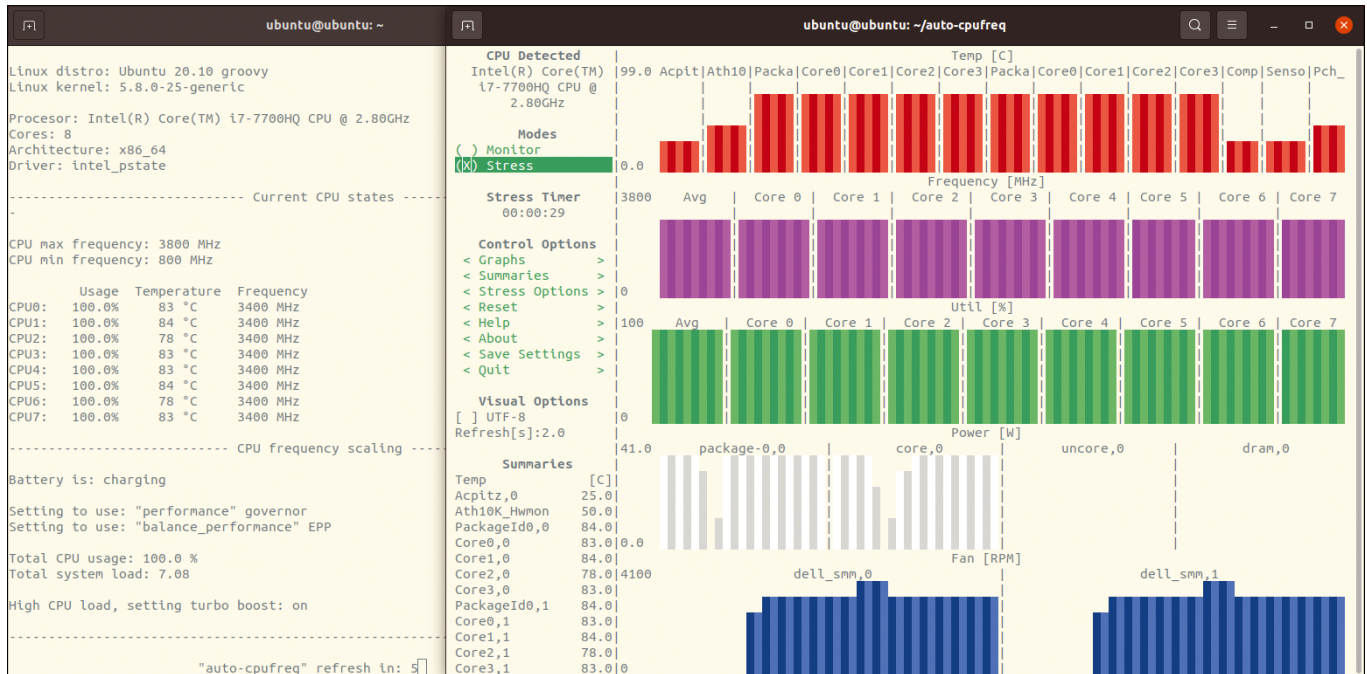


Figure 2: Enabled as a service, auto-cpufreq handles the task of setting the governor. As soon as the system load increases, the program switches to performance mode.

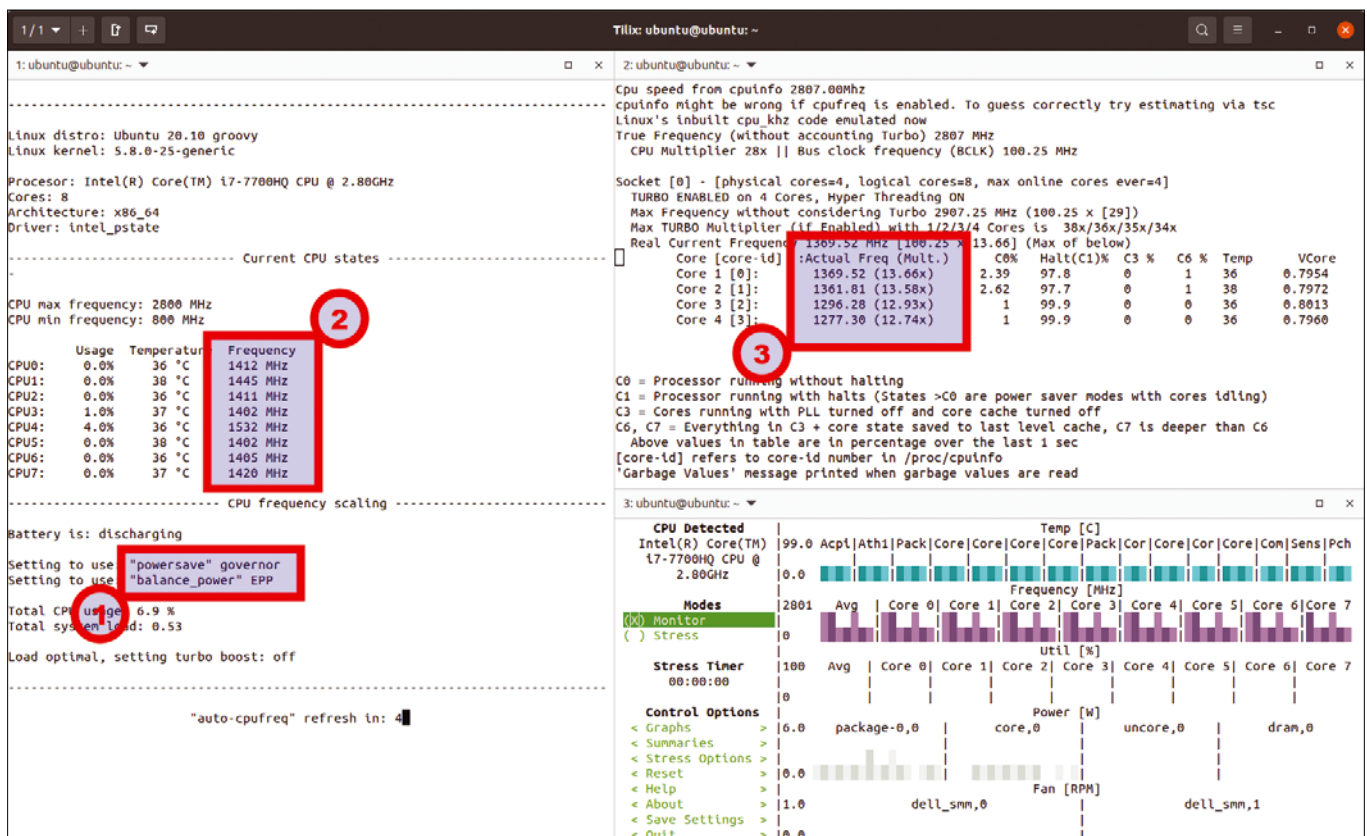


Figure 3: In powersave mode, the system shuts down clocks far faster than Ubuntu does in the default configuration.



option. The idea is to check which governor the system uses when it is running on battery power only and which one it uses when it is on AC power. Test this once with the stress test and once without. Then exit *Monitor* mode, fire up auto-cpufreq with the `--live` option, and repeat the stress test with and without the power supply. The system should now idle in powersave mode during battery operation and only switch to performance mode under load.

If you are suitably impressed by the results in live mode, build auto-cpufreq permanently into the system with a call to `sudo auto-cpufreq --install`. This command sets up auto-cpufreq to activate automatically at boot time. If necessary, you can undo this step later with the `--remove` option. As a service, auto-cpufreq runs completely in the background; if needed, you can keep a log of the program by typing `sudo auto-cpufreq --log`. The `Ctrl + C` keyboard shortcut terminates the output, but the service remains active.

In Operation

In our lab with a freshly installed Ubuntu 20.10 and a proven Arch Linux, auto-cpufreq proved to be very solid. The Ubuntu system generally always works with the performance governor out of the box. Auto-cpufreq, on the other hand, immediately switches to the powersave governor as soon as the computer is unplugged from the power supply. This directly affects the CPU's clock speed. While it stays at 2.8 GHz in Ubuntu's default configuration even after unplugging, the CPU immediately ramps down if auto-cpufreq is enabled and fluctuates dynamically afterwards this depending on the kind of action the system is seeing.

In Figure 3, (1) shows the powersave governor in battery mode, (2) shows the clock speeds, and (3) shows the details of the CPU read out by `i7z`. After enabling the stress test in `s-tui` (bottom right in Figure 3), auto-cpufreq immediately switches to the performance governor even in battery mode and also enables turbo mode. Instead of a maximum of 2.8GHz, the computer's CPU then temporarily runs at 3.4GHz. The default configuration in Ubuntu, on the other hand, completely ignored the CPU's turbo

mode in our test, and the cores ran constantly at a maximum speed of 2.8GHz.

Arch Linux is known to leave the system configuration completely up to the user. When we tested an Arch system, a quick check of the clock speeds during the stress test showed that the system ignores the CPU's Turbo Boost mode in the current configuration. Regardless of whether it is attached to the power supply unit or supplied with power via the battery, the CPU runs at a maximum of 2.8GHz. The performance governor always remains enabled, unless you configure TLP via `/etc/tlp.conf`. Auto-cpufreq immediately integrates harmoniously with the system without any further configuration. The installation in Arch is even slightly easier thanks to the AUR. As on Ubuntu, auto-cpufreq handles the task of setting the governor and enables Intel's Turbo Boost option. Since TLP does not automatically control the governor, the two programs complement each other without getting in each other's way. However, if you plan to use TLP with auto-cpufreq, you should make sure that you have not already enabled the corresponding `CPU_SCALING_GOVNOR_ON_AC` and `CPU_SCALING_GOVNOR_ON_BAT` functions in the TLP configuration in `/etc/tlp.conf`.

Always in View

As a service, auto-cpufreq goes about its business in the background. You won't know when a profile is enabled, when CPU cores are active, or what speed they run at unless you call the right tools in a terminal window. However, you can use a number of widgets or panel applets to visualize the CPU load and other system information. For instance, if you're using Gnome, you can obtain CPU information from the Gnome extension System Monitor and Power Manager [7], a tool that shows the load per core, as well as details such as the governor and the Turbo Boost mode (Figure 4).

Conclusions

In our lab, auto-cpufreq succeeded in extending the battery life of the test system. Your success might depend on your hardware or the nature of the applications running on your system. You can integrate auto-cpufreq into an existing system without major changes and then remove it again without a trace if it doesn't fit the bill. To remove auto-cpufreq, call the auto-cpufreq installer a second time with the `--remove` option. ■■■

Info

- [1] "CPU frequency and voltage scaling code in the Linux kernel": <https://www.kernel.org/doc/Documentation/cpu-freq/governors.txt>
- [2] TLP: <https://linrunner.de/tlp/>
- [3] Auto-cpufreq: <https://github.com/AdnanHodzic/auto-cpufreq>
- [4] Snap Store: <https://snapcraft.io/auto-cpufreq>
- [5] "Massive CPU usage!": <https://github.com/AdnanHodzic/auto-cpufreq/issues/110>
- [6] Intel Turbo Boost: <https://www.intel.com/content/www/us/en/architecture-and-technology/turbo-boost/turbo-boost-technology.html>
- [7] Gnome System Monitor and Power Manager extension: <https://extensions.gnome.org/extension/1082/cpufreq/>



Figure 4: System Monitor and Power Manager for the Gnome Shell shows the clock speed at which the CPU is running and which governor is active.



Tips and tweaks for reducing Linux startup time

Ready Set Go

Today's Linux systems boot faster than ever, but many users still get impatient waiting for that first glimpse of desktop. These tweaks will help you get a faster start from the bootloader and kernel. *By Paul Menzel*

CPU, RAM modules, and storage media keep getting faster, but today's Linux systems are still a little too slow at startup for many busy users. With a few useful tools and a little knowledge of the Linux environment, you can shave some seconds from the startup process. Read on for some insights.

systemd-analyze

The `systemd-analyze` tool supplied with `systemd` gives a first impression of where time is being wasted. The `time` option outputs the total startup time and shows what portion of that time is used for firmware, bootloader, kernel, and user space:

```
systemd-analyze time
```

Listing 1 shows the output, which will vary depending on your firmware, hardware, and software configuration.

Test Setup

The test system for this article was a four-year-old Tuxedo BU1406 with a “Kaby Lake” Intel i7-7500U CPU, 32GB DDR4 RAM, and a 512GB NVMe SSD; this machine cost around a thousand dollars when it was new. The operating system was a Debian “Bullseye” (“Testing”) from November 19, 2020 with a 5.9.9 kernel and `systemd` 246.6 set up by the Debian installer. The standard system with Gnome 3.38.1, GDM 3.38.2, and an OpenSSH server provides the user interface. Our focus was primarily on systems with UEFI firmware, which is common now on most modern hardware.

Our goal was to minimize the startup time based on the readings for *firmware*, *loader*, and *kernel* in the `systemd-analyze` output. Older systems with significantly longer default boot times typically offer far more potential to improve the boot speed.

Firmware

Users typically have few options to optimize the firmware, because it is often proprietary and only the device manufacturer can change and distribute it. In the default Debian configuration used on the test system, the UEFI firmware starts the GRUB bootloader. Other distributions might use the alternative `systemd-boot` boot manager in place of GRUB.

UEFI firmware is now ubiquitous on x86 systems. The startup screen typically announces which key or key pattern you need to press to access firmware settings. In the graphical firmware menu, you can look for functions that slow down the system, like network boot, and turn them off if they are not needed. Some systems even offer option for a faster startup, like *Quick Boot* or *Fast Boot*.

Aside from the lack of source code – and thus, lack of freedom – the size of the UEFI firmware is also a major criticism. Alternatives include `coreboot` for x86 systems, as well as `U-Boot` and `coreboot` for ARM systems. `Coreboot` lets you initialize only enough hardware to boot the operating system. `Coreboot` firmware is used in Chromebooks and on devices by vendors such as System76 and Purism. The minimal approach results in firmware startup times of less than a second, or even as little as 500ms for some devices.

GRUB Delay

By default, GRUB gives the user five seconds to interrupt the automatic boot process. If you want to shave that five-second wait down to one second, change the GRUB timeout variable in `/etc/default/grub` to `GRUB_TIMEOUT=1`. Then enter `sudo update-grub` to transfer this value by recreating `/boot/grub/grub.cfg`. If you make this change, you'd better memorize the boot menu, because you'll only have one second to switch from the default. You can also set the value to `0`,





When the startup process unpacks `initrd` into memory, it spends a lot of time unpacking drivers that might not be needed for your hardware.

In the Linux messages, which you can view by typing `sudo dmesg` or `journalctl -k`, you will find output like Listing 2.

After almost 400ms, the Linux kernel unpacks the `initrd` image, taking about 419ms to do so.

Analysis Made Easy

The `systemd-bootchart` tool displays the kernel's metrics in graphical form. If `systemd-bootchart` isn't present on your system, enter

```
sudo apt install systemd-bootchart
```

to install the package. Then add the string from Listing 3 to the `GRUB_CMDLINE_LINUX_DEFAULT` variable in `/etc/default/grub`. After running `sudo update-grub` and a subsequent reboot, you will have an SVG file in the `/run/log/` directory that you can view in the browser or an image viewer (Figure 1). The chart will help you identify which routines have the longest runtime and therefore the greatest optimization potential.

The Linux `initcall_debug` parameter in Listing 3 tells Linux to print timing information for each `initcall`. `initcalls` are used to load statically linked kernel drivers and subsystems.

As you can see in Figure 2, the `populate_rootfs()` method is one of the most time consuming, with a runtime of 419ms (Figure 2). My challenge is to reduce this time to 10ms.

which would mean you wouldn't see the boot menu at all and would need to use a Live system for repairs in case of an error.

A Brief Introduction to `initrd`

`Initrd`, which is short for "Initial RAM disk," is a temporary filesystem that is copied into memory to support the Linux startup process. One of the first goals of the startup process is to gain access to the root partition to access the files and start the `init` system. However, the root partition could be on a network drive, a USB device with one filesystem, or an NVMe disc with another filesystem. `Initrd` therefore contains a large number of drivers to ensure that the system will start regardless of where the root partition resides.

Listing 1: Analyzing Start Time

```
$ systemd-analyze time
Startup finished in 3.393s (firmware) + 10.554s (loader) + 2.123s (kernel) +
  4.789s (userspace) = 20.860s
graphical.target reached after 4.778s in userspace
```

Listing 2: Log Messages for `initrd`

```
$ sudo dmesg | grep -A1 initramfs
[ 0.398506] Trying to unpack rootfs image as initramfs...
[ 0.817686] Freeing initrd memory: 25484K
```

Listing 3: Supplementing the GRUB Command Line

```
*n*initcall_debug log_buf_len=2M init=/lib/systemd/systemd-bootchart
```

Compressing `initrd`

By default, Debian's `initramfs-tool` compresses the `initrd` with `Gzip` and adds all the drivers. The image is then only 31MB, but loading and unpacking takes quite a while. You can configure the behavior using the `COMPRESS` variable in `/etc/initramfs-tools/initramfs.conf`.

An alternative compression algorithm could provide improved performance: Facebook's `Zstandard` (or `zstd` for short) takes longer to compress but produces a smaller archive and unpacks it faster (see Table 1). At the cost of a larger image,

`LZ4` proves to be even snappier when it comes to unpacking. You might need to install these alternative compression tools before you can use them:

```
sudo apt install zstd lz4
```

As you can see in Table 1, despite the larger archive, `LZ4` needs only one fifth of the time compared to `Gzip`, and one third compared to `zstd`, to decompress in just 84ms. However, the results shown in Table 1 are only valid for the system used in our tests: Depending on the speed of the processor and the hard disk, `zstd` or `Gzip` might offer advantages on other systems.

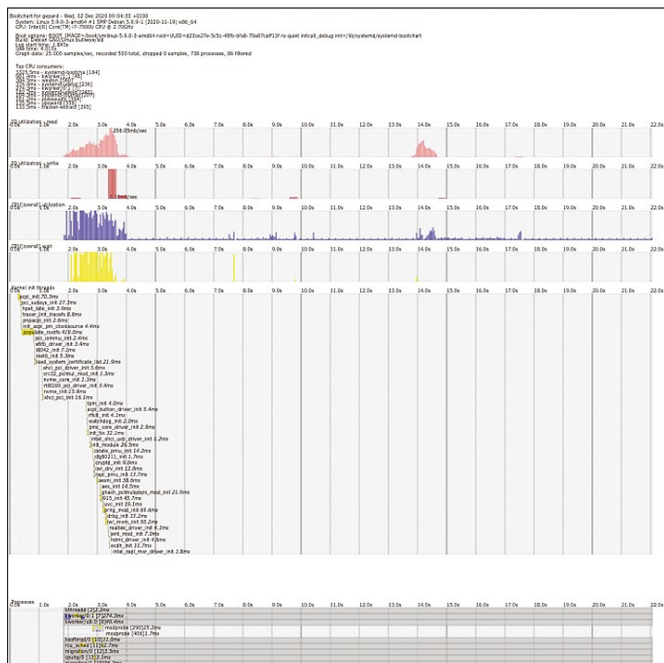


Figure 1: The SVG file stored in `/run/log/` graphically displays the resource utilization, execution times, and durations.

Reducing the Scope

In many cases, you can optimize the startup process by reducing the `initrd` scope. The `initramfs-tools-core` package contains some tools you can use for analyzing the contents of `initrd` (Listing 4, line 1 and 3). Of the 1,377 files on the test system, 665 are kernel modules. You can extract the contents of `initrd` using the command in line 1 of Listing 5; this command lets you analyze the space used by each directory on the lab system (starting in Line 3).

The kernel modules take up the most space, as suspected. However, the diversity offered is only necessary if you want the installation to be compatible with a large number of systems and various scenarios. If the installation only has to work on a single computer, you have some room to optimize.

Table 1: Comparing Compression Methods

Algorithm	Size	initrd Unpack Time	Parameters
Gzip	31MB	419ms	COMPRESS=gzip
zstd	25MB	260ms	COMPRESS=zstd
LZ4	38MB	84ms	COMPRESS=lz4

Table 2: Purified initrd

Algorithm	Size	initrd Unpack Time	Parameters
zstd	6.1MB	58ms	COMPRESS=zstd
LZ4	8.4MB	18ms	COMPRESS=lz4

Listing 4: Analyzing initrd Content

```
01 $ lsinitramfs /boot/initrd.img-5.9.0-3-amd64 | wc -l
02 1377
03 $ lsinitramfs /boot/initrd.img-5.9.0-3-amd64 | grep -c ko$
04 665
```

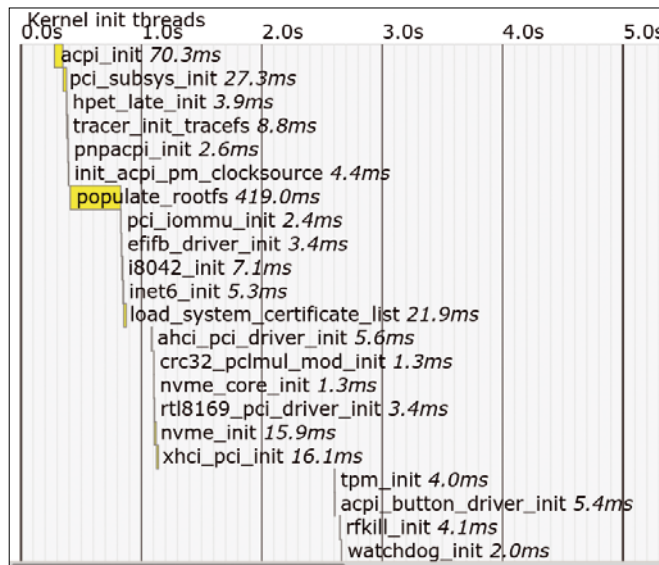


Figure 2: Among the kernel init threads, the `populate_rootfs` function takes up the most time during the boot process, clocking in at 419ms.

The `MODULES=dep` parameter in `initramfs.conf` makes sure that only those modules necessary for reading the root filesystem end up in `initrd`. After recreating the image, the size of `initrd` shrinks quite substantially. LZ4 is still the fastest way to unpack the results (see Table 2).

Listing 5: Analyzing Space Usage

```
01 $ umkmkinitramfs /boot/initrd.img-5.9.0-3-amd64 5.9.0-3-amd64
02 $ cd 5.9.0-3-amd64
03 $ du -sh *
04 3.4M early
05 98M main
06 $ du -sh main/usr/lib/* | sort -rh
07 84M main/usr/lib/modules
08 9.0M main/usr/lib/x86_64-linux-gnu
09 2.3M main/usr/lib/firmware
10 200K main/usr/lib/udev
11 76K main/usr/lib/klibc-zlroYG-5K-Jd0h1POSdWOT4wCA.so
12 16K main/usr/lib/systemd
13 16K main/usr/lib/modprobe.d
14 $ <B>du -sh main/usr/lib/modules/5.9.0-3-amd64/kernel/
    drivers/* | sort -rh | head -10<B>
15 29M main/usr/lib/modules/5.9.0-3-amd64/kernel/drivers/net
16 17M main/usr/lib/modules/5.9.0-3-amd64/kernel/drivers/scsi
17 2.6M main/usr/lib/modules/5.9.0-3-amd64/kernel/drivers/
    infiniband
18 2.5M main/usr/lib/modules/5.9.0-3-amd64/kernel/drivers/block
19 2.2M main/usr/lib/modules/5.9.0-3-amd64/kernel/drivers/usb
20 2.2M main/usr/lib/modules/5.9.0-3-amd64/kernel/drivers/hid
21 1.9M main/usr/lib/modules/5.9.0-3-amd64/kernel/drivers/ata
22 1.3M main/usr/lib/modules/5.9.0-3-amd64/kernel/drivers/mmc
23 940K main/usr/lib/modules/5.9.0-3-amd64/kernel/drivers/nvme
24 880K main/usr/lib/modules/5.9.0-3-amd64/kernel/drivers/input
```



Listing 6: Analyzing Libraries

```
$ du -sh main/usr/lib/x86_64-linux-gnu/* | sort -rh | head -10
3.0M main/usr/lib/x86_64-linux-gnu/libcrypto.so.1.1
1.8M main/usr/lib/x86_64-linux-gnu/libc-2.31.so
1.3M main/usr/lib/x86_64-linux-gnu/libm-2.31.so
572K main/usr/lib/x86_64-linux-gnu/libpcre2-8.so.0.9.0
432K main/usr/lib/x86_64-linux-gnu/libdevmapper.so.1.02.1
412K main/usr/lib/x86_64-linux-gnu/libext2fs.so.2.4
368K main/usr/lib/x86_64-linux-gnu/libmount.so.1.1.0
332K main/usr/lib/x86_64-linux-gnu/libntfs-3g.so.883.0.0
320K main/usr/lib/x86_64-linux-gnu/libblkid.so.1.1.0
252K main/usr/lib/x86_64-linux-gnu/libfuse3.so.3.10.0
```

The kernel modules now only occupy 4.6MB instead of 84MB, and the library directory now takes up most of the space. Another look at the files stored there reveals some big guzzlers (Listing 6).

The `libcrypto.so.1.1` library needs `kmod (libkmod.so.2.3.5)` to check the signature of Linux modules [1]. You could basically build `kmod` with `--without-openssl`. I tried this on the test system and thus removed the dependency, but this is not a good idea for security reasons. However, the question arises as to why libraries for NTFS and FUSE need to be in `initrd`. Many programs have a switch to output more detailed messages. Use the `-v` option for more verbose output with `update-initramfs` (Listing 7).

These hooks – a way for packages to integrate their own scripts – are located in the `/usr/share/initramfs-tools/hooks/` directory and are executable. An unneeded script can be disabled by using the `chmod` command to remove the executable

Listing 8: Disabling Three Hooks

```
$ sudo chmod -x /usr/share/initramfs-tools/hooks/{fuse,ntfs_3g,thermal}
$ sudo update-initramfs -uv
[...]
/usr/share/initramfs-tools/hooks/ntfs_3g ignored: not executable
[...]
```

Listing 9: Checking Microcode Update

```
$ dmesg | grep microcode
[ 0.000000] microcode: microcode updated early to revision 0xde,
date = 2020-05-26
[ 0.529552] microcode: sig=0x906e9, pf=0x2, revision=0xde
[ 0.529744] microcode: Microcode Update Driver: v2.2.
```

Listing 10: Unpacking in Less Than 15ms

```
$ sudo dmsg
[...]
[ 0.398422] calling populate_rootfs+0x0/0x109 @ 1
[ 0.398453] Trying to unpack rootfs image as initramfs...
0.413173] Freeing initrd memory: 6968K
[ 0.413254] initcall populate_rootfs+0x0/0x109 returned 0 after 14480 usecs
```

Listing 7: What Does update-initramfs Do?

```
$ sudo update-initramfs -uv
Available versions: 5.9.0-3-amd64
5.9.0-2-amd64
[...]
Calling hook ntfs_3g
Adding binary /bin/ntfs-3g
Adding binary-link /usr/lib/x86_64-linux-gnu/libntfs-3g.so.883
Adding binary /usr/lib/x86_64-linux-gnu/libntfs-3g.so.883.0.0
[...]
```

bit (Listing 8). Note that a system update can reverse the change. LVM and encrypted root partitions require `dmsetup`. On systems where `/dev/mapper/` does not contain appropriate files, you can also disable the `dmsetup` hook.

On systems with firmware that includes and applies the latest microcode updates (Listing 9), you do not also have to include these updates in `initrd`. On Intel systems, where Linux does not log the update, you can also disable the `intel_microcode` hook, which saves 500KB.

All of these comparatively simple changes help the system unpack `initrd` in 14.5ms (Listing 10), where it took 419ms before – a massive 28 times faster.

According to the 80-20 rule, the last milliseconds require far more effort on the user's part. You could still remove a few Linux kernel modules. On the test system, for example, booting does not require the AHCI driver or the MMC driver. The `MODULES=list` option in `initramfs.conf` makes sure that only the modules listed in `/etc/initramfs-tools/modules` are added (Listing 11). This step reduces the time for unpacking and loading `initrd` to 9.5ms, which means that I have reached the target for the lab system (see the “Potential” box).

Crypto Modules

The graph created by `systemd-bootchart` shows a large share for the `dh_init` and `rsa_init` methods in the total boot time (Figure 3). It turns out that these cryptographic modules run

Listing 11: Explicitly Listed Modules

```
crc32c-intel
crc32c_generic
jbd2
mbcache
crc16
ext4
crct10dif_common
crct10dif-pclmul
crct10dif_generic
crc-t10dif
t10-pi
nvme-core
nvme
```



Listing 12: Cryptography Test Time

```
$ sudo dmesg | grep dh_init
[ 0.144678] calling dh_init+0x0/0x20 @ 1
0.245015] initcall dh_init+0x0/0x20 returned 0 after 97656 usecs

$ sudo dmesg | grep rsa_init
[ 0.245015] calling rsa_init+0x0/0x50 @ 1
[ 0.275009] initcall rsa_init+0x0/0x50 returned 0 after 27343 usecs
```

hardware tests that take some time (Listing 12). If you value security, you need to trust the developers that these tests are necessary. If you do not want to wait the 124ms, add the `cryptomgr.notests` option to the `GRUB_CMDLINE_LINUX_DEFAULT` variable from the `/etc/default/grub`.

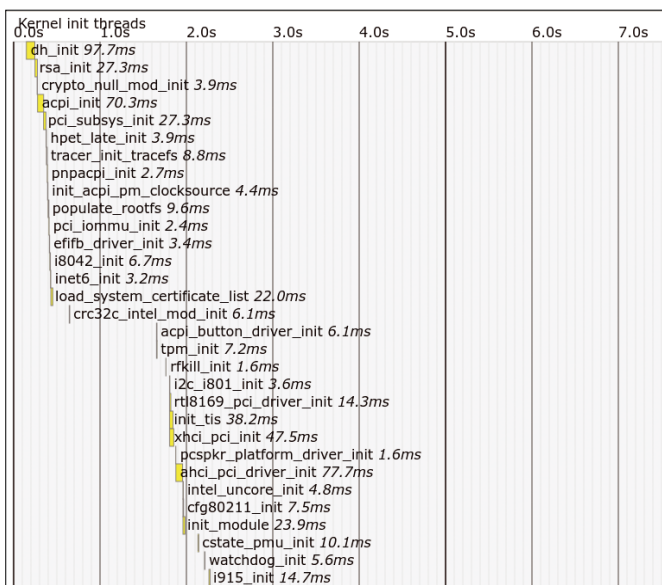


Figure 3: Instead of 419ms, `populate_rootfs` now takes less than 10ms. `dh_init`, `rsa_init`, and `acpi_init` now occur in the first second.

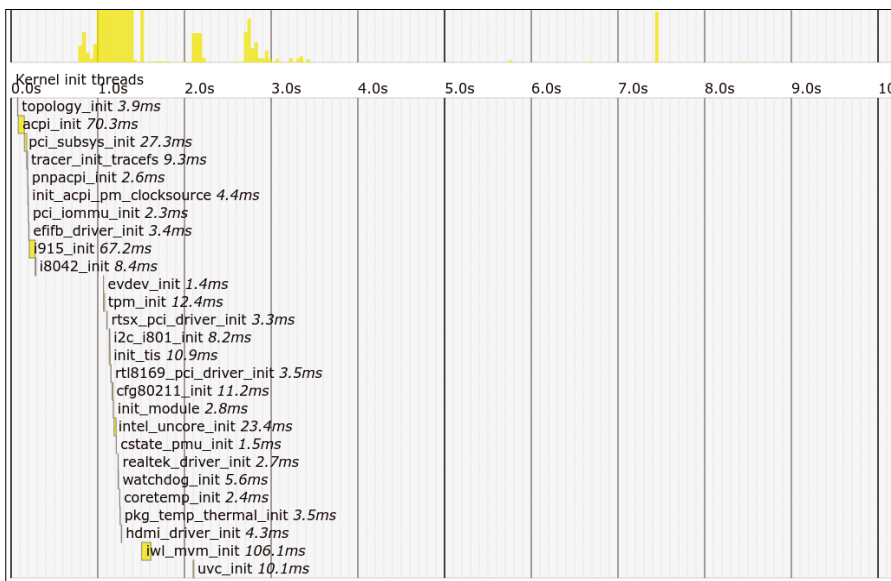


Figure 4: Following all the optimizations, loading the most important drivers now takes less than 300ms. The drivers for ACPI and the Intel graphics have the largest time share now.

Network Stack

The Linux kernel initializes the network stack very early, which also costs a little time. If you do not need IPv6, for example, disable the module by editing the `/etc/default/grub` file again and adding the `ipv6.disable=1` option to the `GRUB_CMDLINE_LINUX_DEFAULT` line. After a reboot, the system only enables the IPv4 stack, which is often all you

need on local networks.

Conclusions

The methods described in this article are useful for analyzing the boot process for a Debian installation. Including all the optimizations mentioned in the article, the home-built Linux kernel with integrated i915 driver starts the i8042 driver responsible for the laptop keyboard and touchpad before reaching the 300ms mark (Figure 4). The drivers for ACPI and the Intel graphics card now take the most time, although you could start the Intel driver asynchronously on some systems. Without the `initcall_debug` parameter, `acpi_init` takes a few milliseconds less. In all your optimization attempts, you should never forget that measuring can actually influence the measured values.

Modern, powerful systems start faster than their predecessors, but they are still too slow considering the potential for savings. One reason for the sluggishness is that installations need to be as universal as possible and work on as many different systems as possible. However, as a normal user, you will not typically need all the features provided with a default system. If you want to take the time and effort, you can tailor the boot process for your own hardware. On a powerful modern system, you might be able to shave off a couple seconds, which could still make a difference to the user experience, but on an older computer with a longer boot time, these techniques could lead to far more significant reductions. ■■■

Potential

I should note that, without an encrypted root partition, it is possible to do without `initrd` altogether if the kernel has built-in drivers for the interface and the filesystem. This is not the case with Debian, so you would have to build the Linux kernel yourself and adjust the GRUB configuration `/boot/grub/grub.cfg` generated by `update-grub` each time. Because of the effort it takes to configure this option, I don't recommend it for systems where the necessary drivers aren't available by default.

Info

- [1] "Increases size of `initrd` considerably since linking with `OpenSSL`": <https://bugs.debian.org/930752>

DrupalCon EUROPE 2020 DECEMBER 8-11

100+ sessions in 5 tracks
4 Keynotes
6 workshops and more

LINUX UPDATE

EXPLORING THE WORLD OF LINUX

- Video Conferencing with Jitsi
- Patreon Project Looks to Bring Linux to Apple Silicon
- Manjaro Linux 20.2 has Been Unleashed
- Minder – Mind Mapping Software
- Visualize Complex Projects with ProjectLibre

FEATURED ARTICLES

Video Conferencing with Jitsi
If you are looking for an alternative to commercial videoconferencing platforms, Jitsi offers an open source solution that lets you build and deploy online videoconferences. (more)

Patreon Project Looks to Bring Linux to Apple Silicon
Developer Hector Martin has created a patreon page to fund his work on developing a port of Linux for Apple Silicon Macs. (more)

Manjaro Linux 20.2 has Been Unleashed
The latest iteration of Manjaro Linux has been released with a few interesting new features. (more)

Minder – Mind Mapping Software
This free mind mapping software is a promising app designed to help you quickly record ideas and show the relationships between them. (more)

Visualize Complex Projects with ProjectLibre
ProjectLibre helps you organize and optimize a complex project with lots of moving parts. (more)

FURTHER READING


- System76 Refreshes the Galago Pro Laptop
- Rasp Pi Kitchen Timer
- Explore the reMarkable 2 Handwriting Tablet
- A New Linux Distribution has been Released
- Linux Mint Unveils New Packages

Be Prepared.
Effective Protection for IT & OT Networks.

Identify anomalies
Secure IT & OT networks
Use of AI & Threat Intelligence

DOWNLOAD WHITE PAPER

20 Years of Linux Magazine



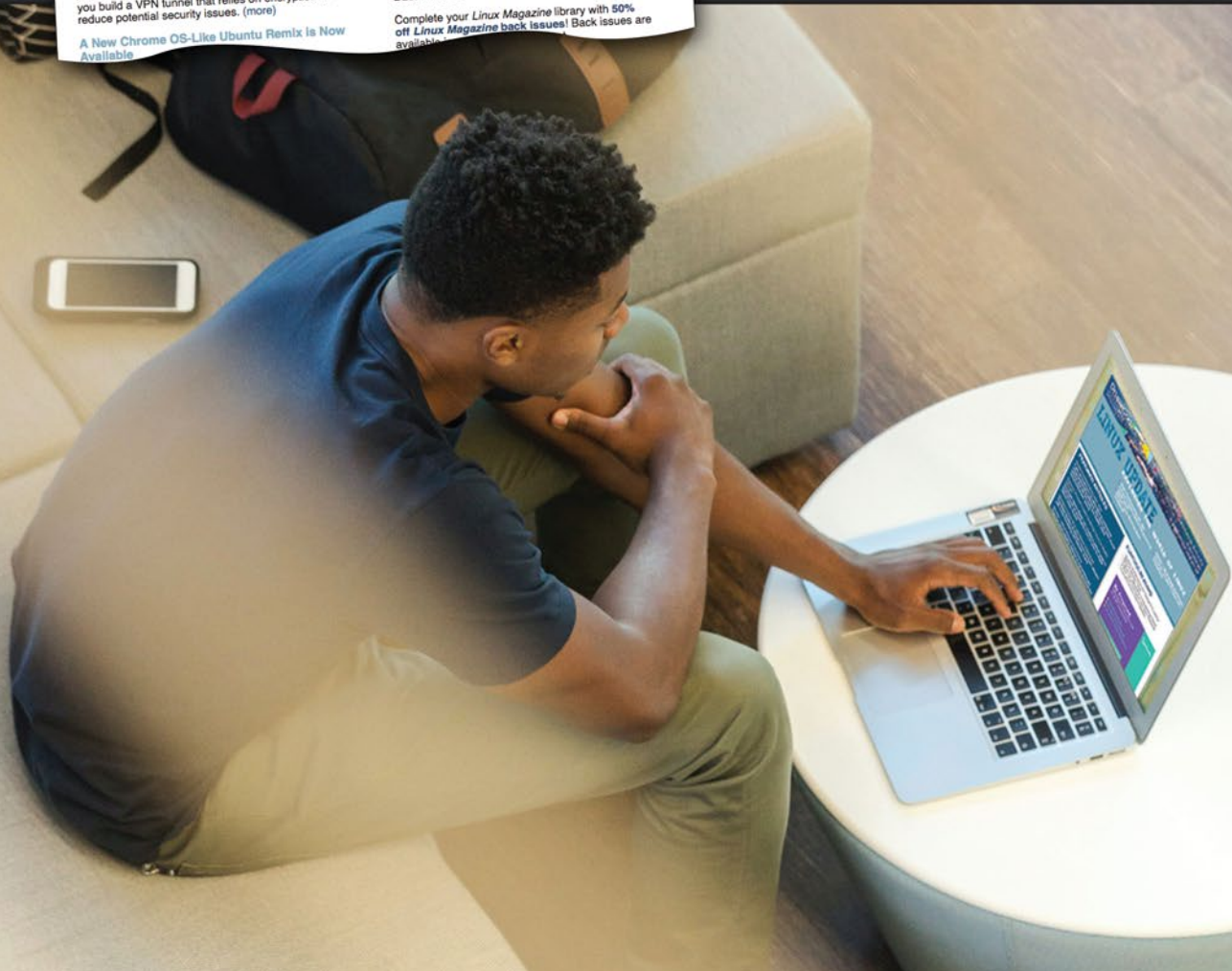
Back issue sale ends December 15th!
Complete your Linux Magazine library with 50% off Linux Magazine back issues! Back issues are available...

A New Chrome OS-Like Ubuntu Remix is Now Available

Need more Linux?

Our free Linux Update newsletter delivers insightful articles and tech tips to your mailbox twice a month. You'll discover:

- Original articles on real-world Linux
- Linux news
- Tips on Bash scripting and other advanced techniques
- Discounts and special offers available only to newsletter subscribers



bit.ly/Linux-Update

elementary OS

Where Business and Aesthetics Meet

In the past decade, elementary OS has grown from open source project to a company with a unique business model. *By Bruce Byfield*

Elementary OS was first released in March 2011 during a time when KDE, Gnome, and Ubuntu were radically redesigning their desktops. From the start, elementary OS added to the mix, introducing a clean design that was soon widely compared favorably to macOS. Today, elementary OS continues to thrive, advertising itself on the project's home page as “the fast, open, and privacy-respecting replacement for Windows and macOS.” Recently, Daniel Foré, an elementary OS founder, discussed how the distribution has developed in the last decade.

Foré got his start in open source by working on pet projects including customizing his own computer. “As the number of things I was involved in grew and as I began to share them,” Foré says, “there was this natural need to distribute these things as some kind of collected work. So we really made elementary OS as a way to put together all the apps and design work we had done into something we could easily share.”

Even though the first release was based on Gnome, elementary OS attracted immediate attention. However, it was not until the second release in 2013

that Foré feels that the project began coming into its own: “It was the first release featuring our [own] desktop, Pantheon, instead of Gnome, and where we had a proper build system in place instead of chrooting into an Ubuntu ISO.”

In 2015, the project took on another new direction by becoming a business. “I remember having a conversation with my boss at my day job,” Foré says. “And he just said, ‘Why are you still working here? Go and do your thing.’” So Foré did, incorporating elementary OS as a business and introducing a pay-what-you-want download system. Despite the fact that free downloads are still available, the payment system continues to attract criticism. But Foré responds that “elementary, Inc. isn’t a nonprofit organization, and we don’t ever advertise paying as some kind of charitable donation. We make a product and we sell it, the same as any other business. We think pay-what-you-want is a really fair and ethical business model, and we don’t see it as controversial at all. If we didn’t sell elementary OS any more, we would go out of business.”

Since then, elementary OS has found other sources of revenue, including GitHub Sponsors and an early access program. “The revenue that Early Access

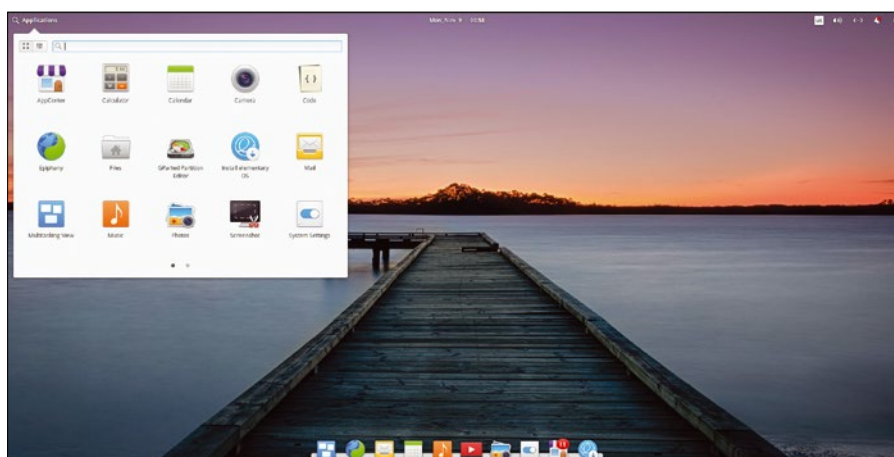


Figure 1: The elementary OS desktop Pantheon is noted for its thoughtful design.

Photo by Meiyang Ng on Unsplash

is bringing in right now through GitHub Sponsors is more than double what Patreon has ever brought in,” Foré says. “I think it’s really going to allow us to grow even more.”

An unexpected donation came in 2018. Foré flew to Denver to talk to someone who had made money in cryptocurrencies and received funding for the full-time hiring of elementary OS cofounder Cassidy Blaede James as Chief Experience Officer. A veteran of System76, James also provided OEM contacts that Foré hopes will assist future growth.

Today, elementary, Inc consists of two full-time employees, Foré and James, and a handful of regular contractors. However, Foré says, “development is primarily volunteer driven,” with 20-30 active volunteers at any given time. Business decisions are made by Foré and James, and “technical decisions are largely made by the people most involved and ideally the ones most knowledgeable for a particular subject.” Foré says, “We’re primarily driven by our volunteer community, so consensus building is important. We try to keep things public on GitHub as much as possible and make sure we’re considering as many perspectives as possible, so there’s a fairly flat organization structure. I’ve been overruled many times, so I think it’s working!”

According to Foré, what distinguishes elementary, Inc. from other distributions produced by companies is its business model. While most company-based distributions either do not focus on the desktop or else derive revenue from services, most of elementary’s revenue comes from the website and, more recently, the early access program. In addition, along with a handful of distributions like KDE Neon, elementary is one of the few that produces not only the distro, but the desktop (Figure 1) and key apps as well. “That gives us a tremendous edge in delivering a specific experience to our customers,” says Foré. “The feedback loop is really tight, and we can produce something really cohesive, because we’re not just acting as an intermediary between our customers and the folks who develop the software.”

Another advantage, Foré says, “is AppCenter and our focus on native apps. We’ve spent a lot of time creating this app ecosystem in which someone can

come along with an idea, step through this really simple developer guide, and then publish their app and get paid. Because of this, we’ve added almost 200 new open source and native GTK apps to the ecosystem.”

Design Principles

From the start, elementary OS has been known for its emphasis on design. Foré says, “we have a lot of little mantras we’ve picked up over the years.” In particular, he cites German industrial designer Dieter Rams’ “10 Principles of Good Design,” Bruce Lee’s advice to “hack away at the unessential,” and even Marie Kondo’s advice on decluttering lifestyles. “We don’t seek to remove things just to remove them, but the goal is to get to a place where you’re only surrounding yourself with things that ‘spark joy’ [in Kondo’s words],” says Foré. “We’re definitely not against making your experience more personal, but we’re not maximalists who will add things just to add them, and we’re very careful that we’re not building products that will require our customers to make design and engineering decisions when they’d rather be getting work done. An operating system is, at the end of the day, a tool to do things.”

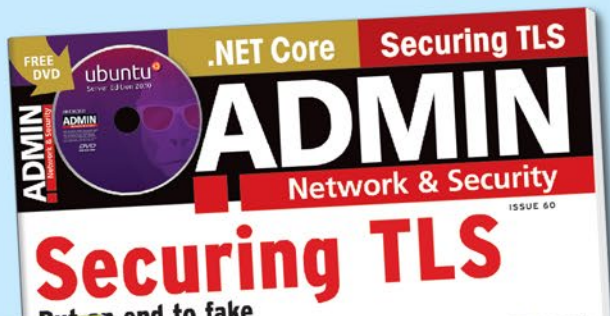
Foré continues, “When we’re talking about design, we don’t just stop at how it looks; how it works is the most important part of good design. One of the first big things we did was work on making apps state-saving so that you can quickly jump back to where you left off, and we always make sure that apps open and close as quickly as possible. To get real work done, the system needs to be responsive or you won’t end up wanting to use it. At the core of everything we do is open source, so, before anything, we build products around making sure we’re working with upstream projects and in a way that is reusable and in an environment where as many people as possible can contribute and give feedback and be involved. And we’re really careful about the kind of incentives that we carry as an organization: We don’t do any kind of data harvesting or advertising, and we do our best to safeguard the privacy of our users, because not only do we use the products ourselves, but we also just don’t like the kind of design that happens as a consequence of not holding these values.”

Future Directions

Clearly, elementary OS is going from strength to strength. But what about the future? “We are really pushing hard,” Foré says, “to make Flatpak the primary packaging format for apps on elementary OS. We ran our AppCenter for Everyone campaign last year right before the pandemic hit and so plans have been greatly disrupted, but we’re still determined to deliver a Flatpak-based app store with great pay-what-you-want apps for everyone. All of the new features we’re building in elementary OS are designed around apps being sandboxed and using portals.”

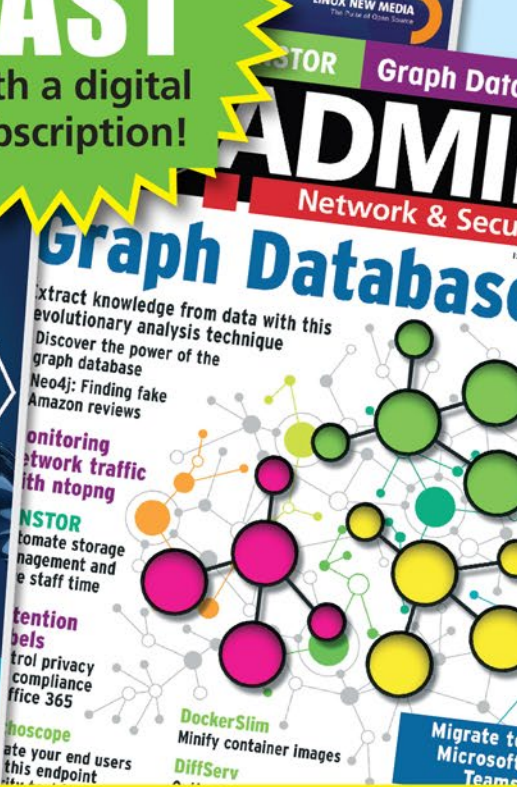
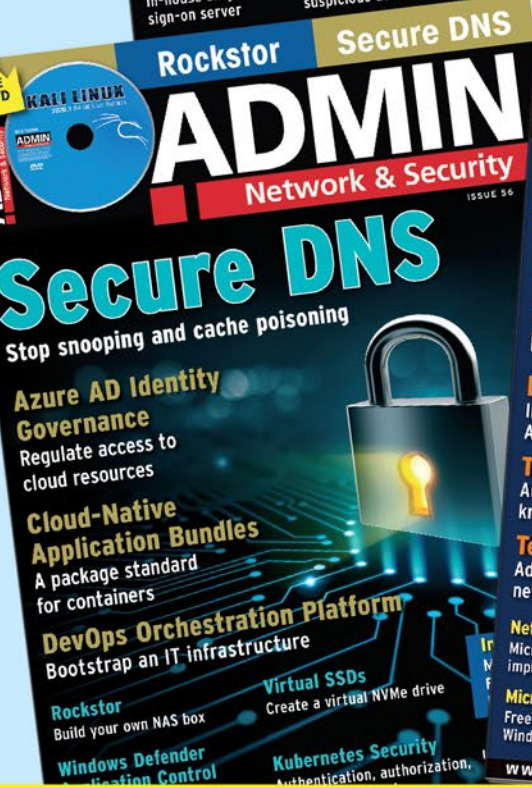
Foré continues, “One of the biggest challenges we face in our mission of bringing an open source operating system to the masses is that most folks don’t shop for operating systems, they shop for computers. Until recently it was really difficult to find computers shipping with elementary OS, and when friends and family would ask about the work we do, it was really hard to explain. Developing relationships with OEMs is really the next big step in our business. We currently have a handful of retailers that we’re working with and I’m really proud of the work we’ve been doing on elementary OS as a result of these partnerships. The new Installer and Initial Setup workflow especially is going to make buying a computer with elementary OS much smoother.”

In aiming for these goals, elementary OS does not see itself in competition with other distributions or open source products. Foré explains, “our mission with elementary OS is to get people running an open source operating system instead of a proprietary one, so focusing on taking away market share from other open source operating systems is kind of antithetical to that goal. I think a lot of people kind of outside of open source development like to think that all of the distros are in competition with each other, but we’re mostly a big group of friends and we work together as much as possible. The metric I like to point to is that over 75 percent of our downloads are coming from proprietary operating systems, the vast majority of which are coming from Windows users. So, I think we’re doing a good job reaching people that are outside of the Linux world.” ■■■



6
issues
per year!

GET IT FAST
with a digital subscription!



shop.linuxnewmedia.com

REAL SOLUTIONS *for* REAL NETWORKS

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

It isn't all Windows anymore - and it isn't all Linux.

A router is more than a router. A storage device is more than a disk. And the potential intruder who is looking for a way around your security system might have some tricks that even you don't know.

Keep your network tuned and ready for the challenges with the one magazine that is all for admins.

Improve your admin skills with practical articles on:

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

SUBSCRIBE NOW
SAVE 30%

ADMIN
Network & Security

shop.linuxnewmedia.com



Peer-to-peer file sharing

Swap Meet

For the occasional local file transfer, a few simple tools can do the job quickly and efficiently.

By Erik Bärwaldt

For smooth file transfer between computers, companies typically use file servers or locally installed cloud instances. But server-based systems require a capable hardware infrastructure; they need to be managed by an administrator and can pose vulnerabilities if configured incorrectly. If you only need to exchange a few files between individual workstations from time to time, these solutions are usually oversized.

It is far easier and faster to send and receive data using applications installed locally on workstations that work on a

peer-to-peer basis. You don't need a server, and the data only travels back and forth between the participating computers without being stored on a third system.

Requirements

When choosing a tool to transfer data directly between two computers, there are a few things you need to consider. Most importantly, the tool needs to secure the data with end-to-end encryption so that third parties cannot sniff the data.

Simple file transfer does not require complex graphical user interfaces (GUIs). All that is needed is a command-line application with a manageable command set that can be used without extensive configuration and lengthy training.

If file synchronization is important, some peer-to-peer solutions for data transfer can additionally keep files and folders permanently synchronized between the participating computers.

Finally, for administrators of heterogeneous IT infrastructures, cross-platform availability is also an important decision-making criterion. The tool needs to support the popular client operating systems, preferably with standardized operation across platforms.

In this article, I will cover a few peer-to-peer tools for file sharing on local networks. (See the "Not Considered" box for tools that didn't make the cut.)

croc

The free `croc` [1] command-line tool transfers one or more files between two computers without an intermediate instance, using a transparently set up relay on the LAN. Written in Go, the cross-platform application is available for countless variants of Linux, even for 32-bit hardware. In addition, `croc` can be used on ARM systems and on various BSD derivatives. Data exchange between all supported platforms is problem-free. Besides corresponding packages, the source code is available for download on `croc`'s GitHub page.

Using `croc`

`croc`'s end-to-end encryption follows the Password Authenticated Key Exchange (PAKE) standard [2], which uses a freely definable secure password to generate a key for file transfer on both computers involved. `croc` is the only tool in the test that can resume interrupted data transfers at a later time.

Not Considered

In addition to the candidates discussed here, there are a variety of other programs for direct data transfer between two computers, including both native Linux applications and Java programs. However, I only looked at applications that are under active development and maintenance for this article. `Dukto` [7], `D-LAN` [8], `BaShare` [9], `p300` [10], and `Transfer on LAN` [11] didn't meet this requirement since their development stopped between 2009 and 2015. In some cases, it was impossible to install the packages on recent distributions due to outdated dependencies.

Photo by Charisse Kenion on Unsplash

```
erik@EliteDesk-800-G2:~$ croc send /home/erik/Downloads/crossover-19.0.2-1.rpm
Sending 'crossover-19.0.2-1.rpm' (225.3 MB)
Code is: amen-jason-miller
On the other computer run
croc amen-jason-miller
Sending (->192.168.1.4:46314)
crossover-19.0.2-1.rpm 89% | ██████████ | (191/215 MB, 2.351 MB/s) [57s:10s]
```

Figure 1: No need for a GUI: croc handles file transfers from the command line.

In the simplest case, the sender starts the transfer by entering

```
croc send FILE
```

It is important to specify the full path in each case, unless you are in the corresponding file directory.

The software then generates a code that the recipient needs to receive the data. The recipient then enters the code at the prompt by typing:

```
croc CODE
```

The program will then prompt the user to confirm the file to be sent. If confirmed, the transfer starts. *croc* visual-

izes this on both systems with a progress bar (Figure 1). It also shows the recipient's IP address and the port used. After completing the transfer process, the transferred file sits in the recipient's home directory.

Optional

In case of frequent data transfers between two computers, the need to generate new codes can quickly become annoying, since you have to communicate them to the recipient each time. To get around this, *croc* offers a simplified form of transmission, where the sender starts the software with a self-defined code that is the same for each transmission using the command:

```
croc send --code CODE FILE
```

To see *croc*'s command-line options, use the `croc --help` command (Figure 2).

```
erik@EliteDesk-800-G2:~$ croc --help
NAME:
  croc - easily and securely transfer stuff from one computer to another

USAGE:
  Send a file:
  croc send file.txt

  Send a file with a custom code:
  croc send --code secret-code file.txt

  Receive a file using code:
  croc secret-code

VERSION:
  v8.6.6-e7ed4fc

COMMANDS:
  send    send a file (see options with croc send -h)
  relay   start your own relay (optional)
  help, h Shows a list of commands or help for one command

GLOBAL OPTIONS:
  --remember      save these settings to reuse next time (default: false)
  --debug         toggle debug mode (default: false)
  --yes           automatically agree to all prompts (default: false)
  --stdout        redirect file to stdout (default: false)
  --no-compress   disable compression (default: false)
  --ask           make sure sender and recipient are prompted (default: false)
  --local         force to use only local connections (default: false)
  --relay value   address of the relay (default: "142.93.177.120:9009") [CROCR_RELAY]
  --relay6 value  ipv6 address of the relay (default: "[2604:a880:800:c1::14c:1]:9009") [CROCR_RELAY6]
  --out value     specify an output folder to receive the file (default: ".")
  --pass value    password for the relay (default: "pass123") [CROCR_PASS]
  --socks5 value  add a socks5 proxy [SOCKS5_PROXY]
  --help, -h     show help (default: false)
  --version, -v  print the version (default: false)
erik@EliteDesk-800-G2:~$
```

Figure 2: *croc* only supports a few command options.

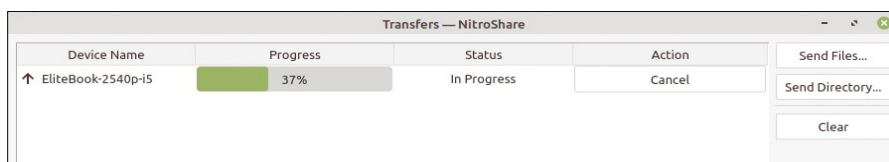


Figure 3: NitroShare's graphical interface looks spartan.

NitroShare

NitroShare [3], a graphical cross-platform tool, enables the most straightforward file transfer possible between two machines on an intranet. Because it is available across platforms, it can be used in heterogeneous IT infrastructures. The tool can be found in the repositories of numerous distributions.

As a special feature, NitroShare has a number of modules for direct integration into the file managers of various desktop environments. Since it is intended for LAN use only, the tool does not compress and encrypt the transferred files. However, transport encryption can be set up and used if desired.

Operation

The installation routine creates a launcher for NitroShare in the desktop menu. Instead of the program opening directly when you click on the launcher, you get a notification window telling you to run NitroShare via an applet in the system tray.

After closing the notification window, the applet appears. Right-clicking on the applet opens a menu where you can access the software's user interface by clicking *Send Files* or *Send Directory*. A file manager then opens to let you select one or more files and directories.

Clicking the *Open* button opens another window where you select the target computer from a list view (NitroShare must already be running on the target computer). After selecting the target computer, the application immediately transfers the data without any further prompt; a corresponding notice appears on both machines. NitroShare shows the source and target machines on both sides as well as the transfer's progress (Figure 3).

By default, the data ends up in a new folder named `NitroShare/`. Use *View Transfers...* in the context menu to view the latest transfers, for example, to determine whether certain files or directories have already been sent or received.

Clear-Cut Design

You can reach NitroShare's very clear-cut configuration dialog via *Settings* in the context menu. In the Settings dialog, you can specify whether the application launches after you log in at computer boot time and set the directories in which

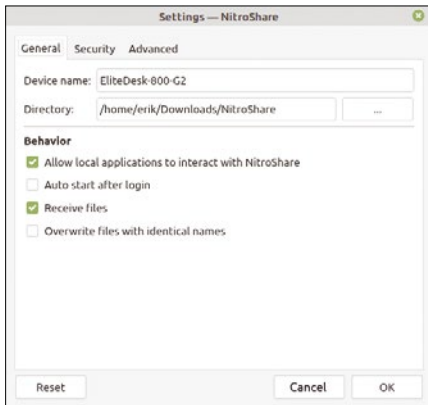


Figure 4: The Settings dialog offers options for setting NitroShare's behavior.

NitroShare stores the data (Figure 4). If necessary, you can also modify the port numbers in this dialog; they must be identical on all machines involved in order to enable a connection. The *Security* tab also offers the option to set up transport encryption in line with the current Transport Layer Security (TLS) standard. However, you do need third-party applications to generate the keys and certificates required for this encryption.

Desktop Integration

NitroShare integrates with the file managers of popular desktop environments using the appropriate add-on modules,

allowing data to be sent directly from the desktop. You will find the corresponding packages in each distribution's software repositories. In addition, NitroShare promises support for the Nautilus, Caja, and Nemo file managers. In our lab, however, it was not possible to integrate the application into the Caja file manager on Linux Mint 20 and Ubuntu Mate 20.04. Although NitroShare builds on the Qt5 framework, with the exception of openSUSE, amazingly there are no modules for integration into the context menus for either KDE's Dolphin or Konqueror file managers.

Syncthing

The free Syncthing [4] is not designed for the occasional transfer of single files. Instead, it keeps larger datasets synchronized between two computers. Unlike locally installed cloud services, for example, Syncthing does not need a server or the complex configuration that entails.

Countless variants of Syncthing, developed in Go, are available for a wide variety of platforms, including 32- and 64-bit versions for Linux and the Raspberry Pi. It also runs on smartphones and tablets running Android. You will also find the source code on the project page. Syncthing with a GTK-based graphical front end is included in the repositories of numer-

ous distributions. This is also where you will find packages that automatically start the program when you log in to a computer.

Setting up Syncthing

Syncthing's installation routine creates three entries for Syncthing in the desktop environment menu hierarchy. Besides a launcher for the actual program, which then runs in the background, a second launcher activates the GTK-based GUI. A third launcher runs Syncthing in the browser.

The developers follow a very unusual operating concept: Syncthing is configured in the web browser the first time it is called up. First you start the actual program and then, using the appropriate launcher, you open the web browser with the application's configuration interface (Figure 5).

In the background, the software generates the required certificates and creates the *Sync/* folder in your home directory, the contents of which it will then synchronize with other computers later on. In the graphical interface in the browser, these folders are preconfigured as the *Default Folder* located on the window's left. On the right, Syncthing lists various statistics for the system and data transfers.

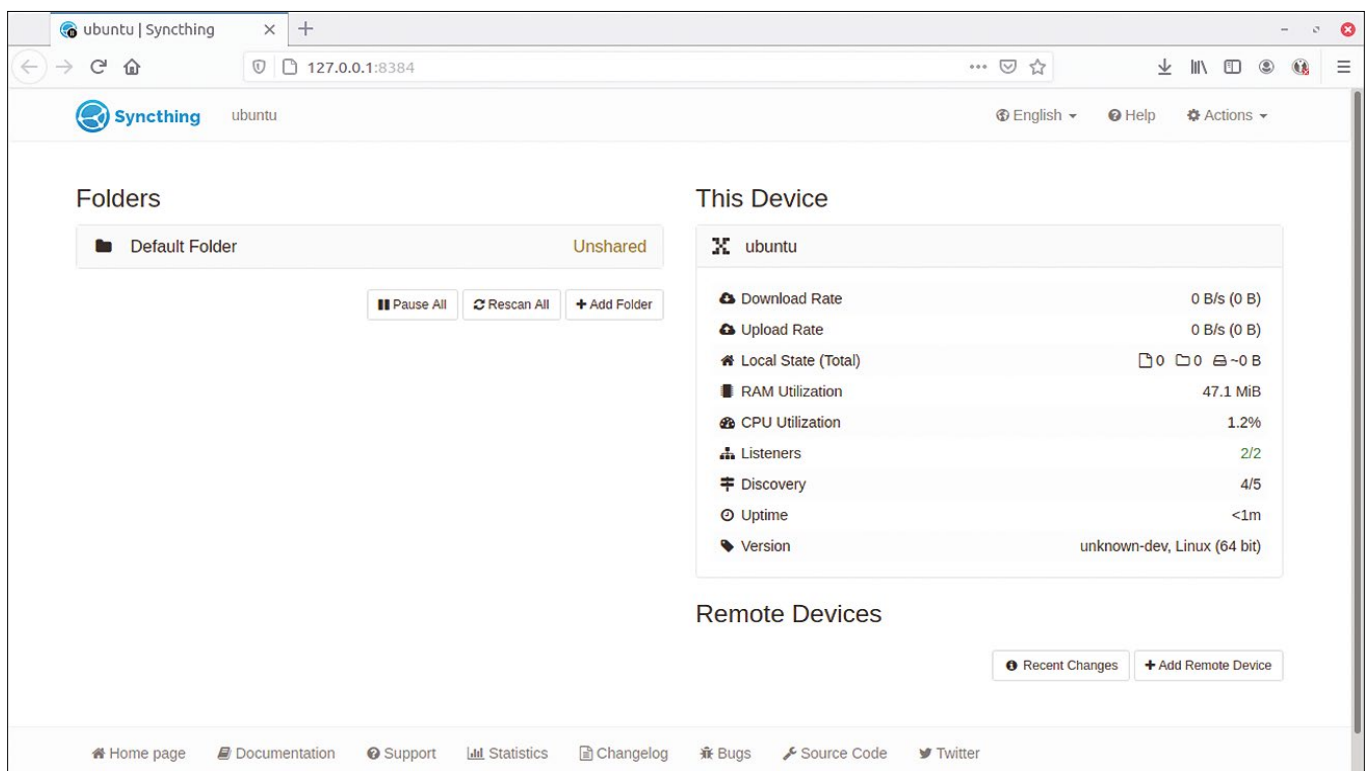


Figure 5: Syncthing can be configured conveniently in the browser.

Syncthing communicates with individual computers via a unique ID that is set when the software is first launched on each computer. You can discover your device's ID by clicking the *Actions* button in the top right corner of the browser window and selecting *Own ID* from the context menu that opens. A QR code and the system's identifier, comprising a total of 56 alphanumeric characters, now appear in a separate window.

To synchronize data between your computer and a remote computer, you also need to know the remote system's ID. To integrate the second computer's ID, click *Add Device* bottom right in the web interface and enter the remote system's ID in the window that opens. You do not need to type in the complete ID: All computers running a Syncthing instance share their IDs on the intranet. They all appear with their IDs in the computer's web interface, letting you select a remote station with a simple mouse click.

To replace the somewhat cryptic default name of the remote system with a more meaningful name, you just need to type a name in the *Device Name* field. The remote computer is then displayed with the new name. After click-

ing the *Save* button, the newly detected device is displayed bottom right under *Remote Devices*.

New Folders

Click on *Add Folder* to add more folders to the synchronization routine. You only need to do this on one of the two devices to be synchronized. Syncthing detects that a new folder has been created on one system and asks the connected system if the new folder should be added there, too. After clicking the *Add* button, it will add the new folder to the system and synchronize it automatically (Figure 6).

Synchronization

To synchronize the files that exist in a folder with the second computer, you first need to tag the corresponding source folder as a folder to be shared. To do this, click on the *Edit* button under *Remote Devices* below the device list. In the window that opens, select the target system and then go to the *Sharing* tab, where you will find *Default Folder*.

If you have specified additional folders that you want Syncthing to include, they will also be in the list. To enable a folder to be synchronized, check the box for

the corresponding directory. After a short delay, the application will now start synchronizing the two folders, with a corresponding progress indicator appearing above the synchronizing folder under *Folders*. To the right of the folder name, Syncthing also displays the transfer rates for upload and download. When the sync is complete, the folder's status changes to *Up to Date* to the right of the folder name.

Native Application

Syncthing's native instance offers a similar range of functions to the web interface, but with a slightly different operation. The graphical front end has just two window segments, in which the folders to be synchronized are displayed on the left, while the clients involved appear on the right along with some of their statistical data (Figure 7). Actual operation is via the gear icon in the top right corner of the titlebar.

Selecting *Add Shared Folder* from the gear icon's context menu lets you add new folders, while *Add Devices* adds new computers. Some of the setup dialogs are far clearer than in the web interface. For example, when creating a new folder, the synchronization inter-

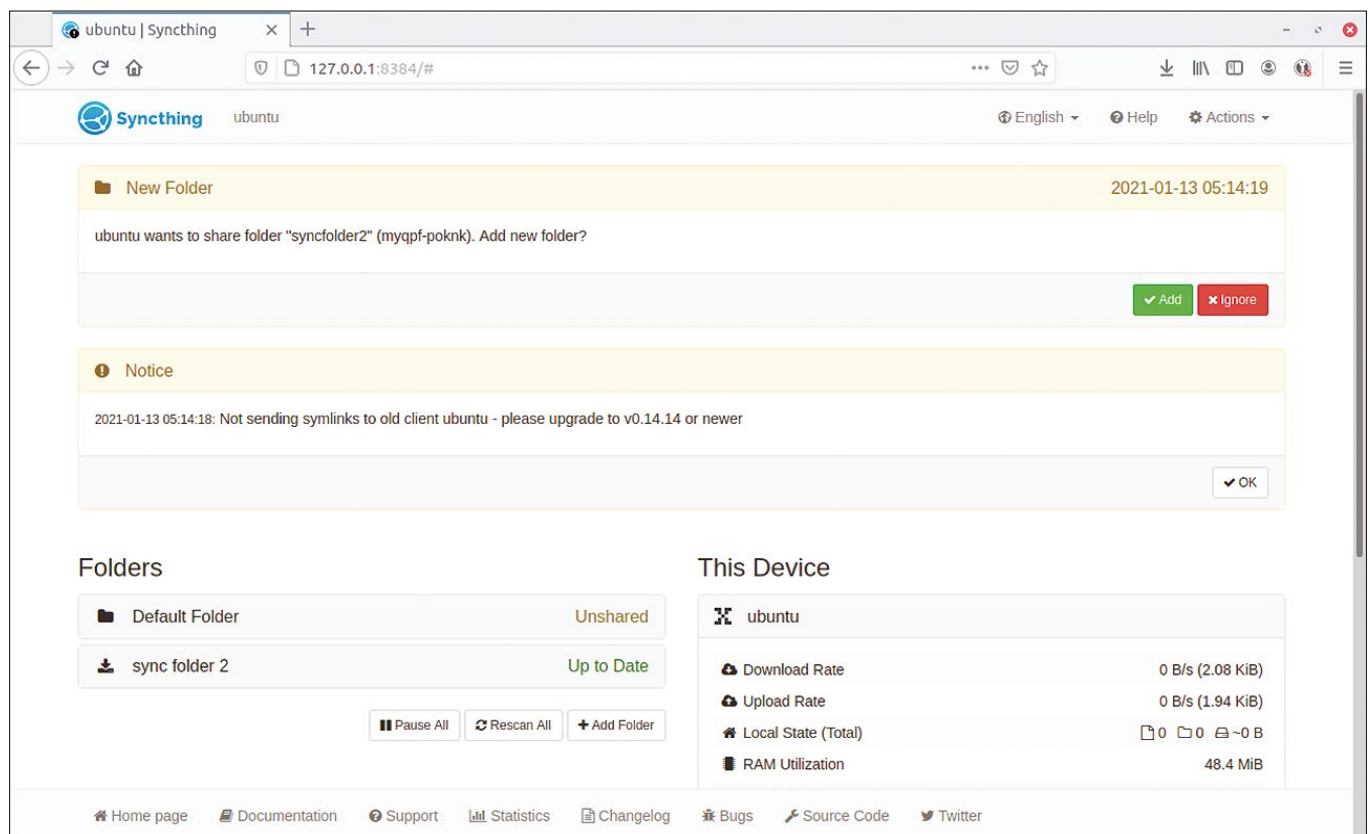


Figure 6: Syncthing's interface provides an overview of the folders to be synchronized.

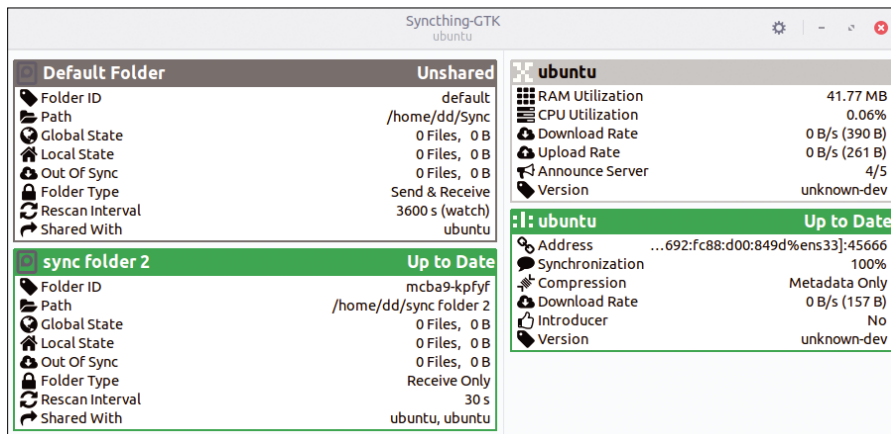


Figure 7: The native application offers almost the same range of functions as the web interface.

vals can also be defined in the same menu. File versioning, including the number of file versions to be kept, and permissions handling are also configured in this dialog.

Selecting *Service Settings* lets you define the intervals at which Syncthing checks the computers involved for content to be synchronized. Here you can also change the bandwidth for the data transfer and the ports the service uses.

Warpinator

Warpinator [5], which is developed by Linux Mint, also performs peer-to-peer file transfers and comes with a graphical interface. The application, written in Python, can be installed directly using the Linux Mint 20 package manager. For other distributions, there is a Flatpak, but there are no native binary packages as of yet. Since Warpinator has numerous dependencies that not all distributions resolve in terms of the required versions, compiling from the source code can be difficult.

After the install and a subsequent reboot, a launcher appears in the menu hierarchy of the respective desktop. Clicking on the launcher opens a spartan-looking window (Figure 8). A scan of the intranet on which Warpinator is running will display all found computers in this window, along with the matching IP addresses and connection states.

To find the settings, click on the hamburger menu top left in the titlebar. Select *Settings* to open a dialog where you can use sliders to adjust many options, including defining the memory path and specifying when the program should dis-

play security prompts. From here, you can also enable Warpinator to automatically start at boot time.

Transfer Awareness

The program creates the Warpinator/ folder in the home directory as the default storage path for the data to be preserved; the path can be changed in the configuration dialog if required. To start sending data, using Warpinator's program window on the source machine, click on the computer to which the content is to be transferred. Warpinator then switches to a list view named *File Transfers* that lists the data to be sent.

If you now click on the *Send files* button at the top right corner of the window, a selection menu appears, where you can click to select the content to be sent. The selection menu will show the last files edited on the system.

To transfer any other content, click *Browse...* at the bottom of the selection menu, which opens a file manager where you can now select individual files and folders. After the transfer, Warpinator displays the files indicating their file or folder size; small icons also appear with each entry, indicating whether the content is multimedia

or individual files and folders. On the target machine, the program also transfers the files to the overview, but displays a message in the system tray indicating the impending transfer and asks for transfer approval (Figure 9).

After initiating the transfer by granting the appropriate permission on the target computer, Warpinator works through the list, displaying a progress bar after each entry showing the data transfer rate and how long the transfer will take to complete. This display is also visible on the target machine.

On the target computer, the user can immediately view the transferred data by clicking the folder icon displayed on the right in the file list. The application will then open the destination folder. To stop a transfer in progress, click the *Stop* button to the right of the matching list entry on the source machine. This not only ends the transfer, but also shows two new buttons for a later transfer and for deleting the entry. On the target machine, Warpinator only signals the cancellation of the transfer (Figure 10).

Magic Wormhole

Magic Wormhole [6] is a simple command-line utility for transferring files or directories to another computer on the LAN. The software is available in the repositories of most popular distributions and can be conveniently installed with the corresponding graphical front ends.

Since the program is still under active development, the developers also provide a list on the project's GitHub page

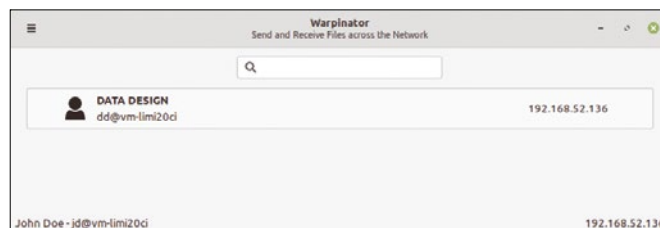


Figure 8: Warpinator does not require any training.



Figure 9: On the target machine, Warpinator shows a transfer waiting for approval.

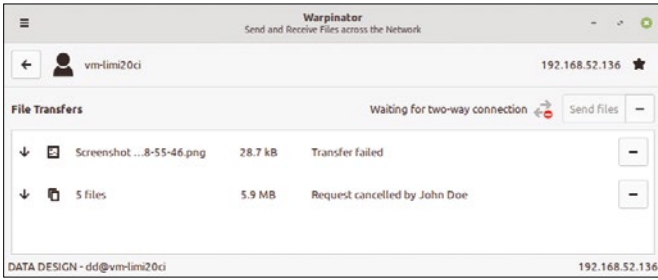


Figure 10: Notification of a cancelled transfer on a target machine.

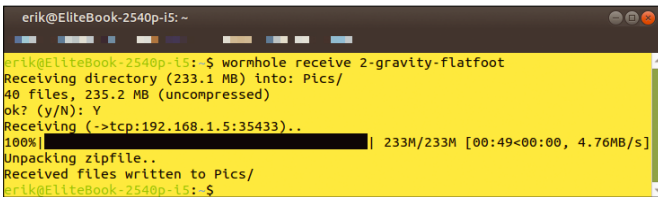


Figure 11: Magic Wormhole works without any bells and whistles.

showing which distribution ships which version of the program. In addition to the packaged versions in the software archives, the source code and a snap package are also available for download. Magic Wormhole uses PAKE encryption for file transfer and creates temporary relay servers by default.

Using Magic Wormhole

Magic Wormhole uses different file transfer parameters on the source and target systems. On the source machine, you type

```
wormhole send FILE
```

while on the destination, you type:

```
wormhole receive
```

The software then asks the target computer for the code, which it generates automatically for secure data transmission in line with the PAKE standard. The

not only transfers individual files, but also complete folders and directory hierarchies if desired. To do this, simply enter the directory to be sent or the root folder of the directory tree on the source computer. The software recognizes that the source is not a single file and generates a ZIP archive from the folder or directory tree. It then generates the code for entry on the receiving computer.

After entering the receive command, Magic Wormhole first transfers the ZIP archive to the target computer, where it is unpacked on receipt, including all originally existing subfolders, and stored in the home directory of the logged-in user. If a folder with the same name already exists on the target computer in the storage directory, the file transfer aborts with an error message. If you have several folders with the same name, you should rename the directory to be transferred or copy it to a new folder before sending.

target computer only contacts the source machine after the code is input. The user on the target computer is then asked if they want to receive the transferred document. If approved, the tool stores the file on the target computer. While doing so, it shows a progress bar in the terminal window (Figure 11).

Packaging

Magic Wormhole

Conclusions

Sending and receiving data on the LAN is a task that can be handled quickly and efficiently using any of the solutions here. The only differences are in the feature scope, operation, and security aspects (see Table 1). In heterogeneous environments, where many users are only familiar with their GUI, command-line applications may be daunting, not least because of the codes to be entered in each case for data encryption. For occasional file transfers, even across operating system boundaries, programs with a graphical front end will tend to be the best choice. Syncthing occupies a special slot here; while it does not support single transfers, it keeps complete folders or directory hierarchies on the participating machines in sync. ■■■

Info

- [1] croc: <https://github.com/schollz/croc>
- [2] PAKE: https://en.wikipedia.org/wiki/Password-authenticated_key_agreement
- [3] NitroShare: <https://nitroshare.net/>
- [4] Syncthing: <https://syncthing.net/>
- [5] Warpinator: <https://github.com/linuxmint/warpinator>
- [6] Magic Wormhole: <https://github.com/warner/magic-wormhole>
- [7] Dukto: <https://sourceforge.net/projects/dukto/>
- [8] D-LAN: <https://www.d-lan.net/>
- [9] BaShare (archived): <https://code.google.com/archive/p/bashare/>
- [10] p300: <http://p300.eu>
- [11] Transfer on LAN (archived): <https://code.google.com/archive/p/transfer-on-lan/>

Table 1: Peer-to-Peer Data Transfer Programs

	Croc	NitroShare	Syncthing	Warpinator	Magic Wormhole
License	MIT	MIT	MPL 2.0	GPLv3	MIT
Command line	Yes	No	No	No	Yes
GUI	No	Yes	Yes	Yes	No
Cross-platform	Yes	Yes	Yes	No	Yes
End-to-end encryption	Yes	No	Yes	Yes	Yes
Resumption after interruption	Yes	No	No	No	No
Integration in file manager	No	Yes	No	No	No
Transfer individual files	Yes	Yes	No	Yes	Yes
Transfer complete directories	Yes	Yes	No	Yes	Yes
Data synchronization	No	No	Yes	No	No

ARCHIVE DVD FREE DVD (\$29.90 VALUE!) **RASPBERRY PI GEEK** THE COMPLETE ARCHIVE 2,000 pages of maker projects and more!

LINUX MAGAZINE

APRIL 2020

STREAM TO YOUR TV

Cool tools for sound and video

Analyze File Metadata

Apache Cassandra Database for the Internet age!

Web Scraping

NAVI Interactive cheat sheet for the shell commands

EDGE COMPUTING

Get ready for the low-latency cloud

Bash Scraping Gather, parse, and filter web data with Bash

FREE DVD **OPENMANDRIVA LINUX 5.2** **Knoppix 8.6.1** **DOUBLE-SIDED DVD INSIDE!**

LINUX MAGAZINE

ISSUE 220 - JUNE 2020

ROCK THE SCIENCE FAIR

Read and visualize weather data with MetPy

WHAT'S NEW IN SYSTEMD

Reinventing home directories with the powerful systemd-homed

Vagrant Create a custom UI for managing virtual machines

Glances Keep up-to-date on system health

PlantUML Create diagrams using human-readable text

"The world around us may be going through strange times, but at least so far kernel development looks normal." - Linus Torvalds (see Kernel News)

Tangram Manage your social media accounts from the terminal

BerryLAN Set up a headless Rasp Pi server

FREE DVD **SystemRescueCd 4.4** **manjaro** **DOUBLE-SIDED DVD INSIDE!**

LINUX MAGAZINE

ISSUE 214 - MAY 2020

EDGE COMPUTING

Get ready for the low-latency cloud

Bash Scraping Gather, parse, and filter web data with Bash

NAVI Interactive cheat sheet for the shell commands

EDGE COMPUTING

FREE DVD **fedora 32** **ubuntu 20.04 LTS** **DOUBLE-SIDED DVD INSIDE!**

LINUX MAGAZINE

ISSUE 230 - JULY 2020

SMARTER DIRECTORIES

Reinventing the file storage metaphor with semantic tagging

OpenCart Build a secure online store

Design an Ebook with Free Tools

Monitoring Tricks Build a dashboard to keep watch on your old Linux computer

Water Your Plants Tending your garden with a Rasp Pi Zero

ProjectLibre Organize and visualize your next project

baobab All-in-one tool for managing Flatpak and AppImage packages

FREE DVD **kubuntu 20.04 LTS** **ubuntu 20.04 LTS** **DOUBLE-SIDED DVD INSIDE!**

LINUX MAGAZINE

ISSUE 227 - AUGUST 2020

WEBCAMS AND LINUX

Helium Lean Linux distro for 32-bit systems

sncli: Sync and manage your notes

maddog: It's Time to Innovate

CALL IN WITHOUT LOCK-IN Get connected with the Jitsi videoconferencing tool

SPEED UP YOUR SYSTEM HOT TWEAKS FOR A FASTER LINUX

WIREGUARD Secure and simple VPN tool

FREE DVD **Bodhi Linux 5.1** **Ubuntu 20.04 LTS** **DOUBLE-SIDED DVD INSIDE!**

LINUX MAGAZINE

ISSUE 228 - SEPTEMBER 2020

SPEED UP YOUR SYSTEM

Hot tweaks for a faster Linux

PSI Discover the kernel's cool new performance monitoring feature

Easy Tips for Optimizing Shell Scripts

Topology Arch 120 package repositories at once

Scary Tech This Halloween vending machine dispenses chocolate treats

SteamPunk Laptop A little copper tubing and a Raspberry Pi

FREE DVD **fedora 33** **ubuntu 20.04 LTS** **DOUBLE-SIDED DVD INSIDE!**

LINUX MAGAZINE

ISSUE 227 - AUGUST 2020

MS AND LINUX

Hardware wrinkles and QtCAM

Imaginary Teleprompter Polish up your video presence

Kitchen Timer Turn a Pi Zero into a DIY kitchen device

FREE DVD **debian 10.5** **ZEVUAN** **DOUBLE-SIDED DVD INSIDE!**

LINUX MAGAZINE

ISSUE 230 - OCTOBER 2020

BUILD AN IRC BOT

Customize a GTK3

DISPATCHER SCRIPTS Automate NetworkManager seamless location changes

Kernel Lockdown Mod Rein in root with this powerful new Linux security feature

Build a WiFi Sound System With a Rasp Pi speakers from

GARDEN TRICKS USE LONG-RANGE RADIO TO MONITOR YOUR PLANTS

BUILD AN IRC BOT

FREE DVD **Bodhi Linux 5.1** **Ubuntu 20.04 LTS** **DOUBLE-SIDED DVD INSIDE!**

LINUX MAGAZINE

ISSUE 228 - SEPTEMBER 2020

SPEED UP YOUR SYSTEM

Hot tweaks for a faster Linux

PSI Discover the kernel's cool new performance monitoring feature

Easy Tips for Optimizing Shell Scripts

Topology Arch 120 package repositories at once

Scary Tech This Halloween vending machine dispenses chocolate treats

SteamPunk Laptop A little copper tubing and a Raspberry Pi

FREE DVD **fedora 33** **ubuntu 20.04 LTS** **DOUBLE-SIDED DVD INSIDE!**

LINUX MAGAZINE

ISSUE 242 - JANUARY 2021

3D PRINTING

From first steps to a finished creation

MOFO Linux Steer around censorship with this privacy-focused distro

Nyttig Share files and stream Internet radio from a Rasp Pi

Netdata Zero configuration monitoring tool

Nerf Dart Game Score points for your next neighborhood party

Costs and Benefits Choose the best alternative with the TOPSIS decision technique

20 YEARS ARCHIVE DVD **HUGE SAVINGS! \$29.90 VALUE!** **20TH ANNIVERSARY ISSUE** ALL 239 ISSUES ON A SEARCHABLE DVD

LINUX MAGAZINE

ISSUE 240 - NOVEMBER 2020

IT'S ALIVE!

Breathe life into your coding experiments by creating your own simple OS

Graphing the Pandemic With freely available data

Ajenti Cool control panel for Linux servers

Serial Protocol Still a powerful option for DIY coding

Build a WiFi Sound System With a Rasp Pi speakers from

hub: A command-line tool for GitHub repository

maddog: Remembering the Linux community

PulseEffects: Equalizer and the Linux audio ecosystem

Tuxedo InfinityBook Light and lively Linux laptop

usql Manage PostgreSQL, MySQL, and SQLite with a single interface

FREE DVD **elementary OS** **JAM.PY** **DOUBLE-SIDED DVD INSIDE!**

LINUX MAGAZINE

ISSUE 241 - DECEMBER 2020

SECURE YOUR SYSTEM

Expert tips for protecting your Linux

Clonezilla SE Mass cloning without the mess

Managing Servers with Cockpit

RebornOS Arch for the slightly less geeky

Christmas Tinkering Build a holiday music box

Git Utilities Cool tools for extending Git version control

Fun with Go Display song lyrics line by line

FREE DVD **ubuntu 20.04 LTS** **DOUBLE-SIDED DVD INSIDE!**

LINUX MAGAZINE

ISSUE 242 - JANUARY 2021

3D PRINTING

From first steps to a finished creation

MOFO Linux Steer around censorship with this privacy-focused distro

Nyttig Share files and stream Internet radio from a Rasp Pi

Netdata Zero configuration monitoring tool

Nerf Dart Game Score points for your next neighborhood party

Costs and Benefits Choose the best alternative with the TOPSIS decision technique

FOSSPicks

- DGIS spatial data environment
- PrettyEQ audio equalizer

Tutorial

- Track your investments with Portfolio Performance

WWW.LINUX-MAGAZINE.COM

Linux Magazine is your guide to the world of Linux. Look inside for advanced technical information you won't find anywhere else!

Expand your Linux skills with:

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

If you want to go farther and do more with Linux, subscribe today and never miss another issue!

Subscribe now!
shop.linuxnewmedia.com/subs



A command-line task manager

BEST LAID PLANS

The `dstask` personal tracker lets you manage your to-do list from the command line. `Dstask` uses Git version control to store tasks, letting you synchronize your to-do list across multiple devices.

By Tim Schürmann

After finding a much-needed network cable under a pile of junk in the basement, I decided it was time to add cleaning up the basement to my to-do list. To manage personal tasks (even unpleasant ones like this), the `dstask` personal tracker can help you prioritize tasks and track completion.

Unlike many other task managers, `dstask` exclusively runs on the command line. With a short and succinct command, you can add a new task or mark off a completed one. On request, `dstask` provides a list of all pending tasks, sorted by urgency. Filters help you stay on track in the task jungle. As an added benefit, you can use your task list to show customers or your boss your completed work.

Under the hood, `dstask` stores the pending tasks in the Git version management system, letting you sync tasks across all your devices. However, unlike `dstask`'s other features, synchronization requires some Git know-how.

Kickstart

To get `dstask` up and running, first install Git using your package manager. To install Git on Ubuntu, type:

```
sudo apt install git
```

Next, go to the `dstask` project page on GitHub [1] and click on the current version number on the right below *Releases*. Under *Assets*, download the version for Linux, `dstask-darwin-amd64`.

Rename the resulting program `dstask` and make it executable. To do this, run:

```
chmod +x dstask
```

You do not have to actually install `dstask`, but the developer does recommend storing `dstask` in `/usr/local/bin`, which lets all of the system's users call the program directly by typing `dstask`.

After that, you still have to create the Git repository where `dstask` stores all the tasks and become acquainted with Git (Listing 1). Replace `editorial@linux-user.en` and `Tim Schürmann` with your own email address and full name. Without this in-

formation, you will always get error messages when working with `dstask`.

Come On In

To add the basement cleanup task to `dstask`, use the call shown in line 1 of Listing 2. The `add` command tells the program to create a new task, which is described in the following brief summary (`clean up basement`).

`Dstask` adds all words that follow a plus sign (+) to the task as keywords (i.e., `basement` and `private` in Listing 2). These tags will help you later on when searching for a specific task. You can also add the tags to the summary, as shown in line 2 of Listing 2.

In addition, you can set the task priority, which is shown by the number fol-

Listing 1: Creating a Git Repository

```
$ mkdir ~/.dstask && git -C ~/.dstask init
$ git config --global user.email "editorial@linux-user.en"
$ git config --global user.name "Tim Schürmann"
```

Listing 2: Adding Tasks

```
01 $ dstask add clean up basement +basement +private P2
02 $ dstask add +private basement +clean up basement P2
```

Photo by Kelly Sikkema on Unsplash

lowing P. Dstask supports priority levels P3 to P0, with P0 for urgent tasks and P2 for normal priority tasks. In Listing 2, I've set my basement cleanup task to P2, since it's not time critical. Dstask automatically defaults to P2 if you don't include the priority level.

Roll Up Your Sleeves

To see what dstask has on your to-do list, just call `dstask`. The program assigns a consecutive ID to each task. Entering `dstask 1` shows detailed information about the task (Figure 1). Dstask ignores case sensitivity in the tags.

The detailed information also shows the task status. Immediately after creation, the task will be shown as pending. When you head down to the basement and pick up the first box, call

```
dstask start clean up basement
```

to change the status. This switches the task to the active state, signifying that the task is in progress. If you take a break, stop processing with

```
dstask stop clean up basement
```

which toggles the task back to the pending state.

Once the basement is finally clean, mark the task as done by entering

```
dstask done clean up basement
```

Even though dstask is done reminding you about this task, it will still store the task, thereby creating a record of your completed tasks for your boss or client.

To get a report on all completed tasks, enter

```
dstask show-resolved
```

Dstask automatically sorts the tasks by weeks. If you want to remove a task from the dstask repository, use the following command:

```
dstask remove clean up basement
```

Sticky Notes

During my basement cleanup, I discovered some old novels that Aunt Dorothy lent me. I need to make a note of this somehow as a reminder. Dstask lets you attach as many notes to a task

as you like. To create a reminder for Aunt Dorothy's books, use the command from line 1 of Listing 3. The number stands for the task ID. In the example, dstask then pins a note reading *return novels to Aunt Dorothy* onto the *clean up basement* task.

Using the same principle, you can create as many additional notes as you like. Markdown experts can also use tags, but dstask 0.23.2 is not currently capable of evaluating or visually rendering Markdown tags. You can also tell dstask to directly add a note when you

add a task (Figure 2). To do this, simply append a slash followed by the note (Listing 3, line 2).

Project Work

While you're cleaning up the basement, you might as well tackle the garage and declutter the shoe closet. All three cleaning tasks could be grouped together to create a spring cleaning project.

Dstask lets you add individual tasks to a project by adding another parameter when you create a task. The tasks in lines 1 and 2 of Listing 4 are part of the

```
dd@vm-limi201c:~
File Edit View Search Terminal Help
dd@vm-limi201c:~$ dstask add clean up basement +basement +private P2
Added 1: clean up basement
[master (root-commit) 854eab1] Added 1: clean up basement
1 file changed, 13 insertions(+)
create mode 100644 pending/e19058e6-e829-45df-8727-9e07625b42bc.yml
dd@vm-limi201c:~$ dstask 1
Name      Value
ID        1
Priority   P2
Summary   clean up basement
Status    pending
Project
Tags       basement, private
UUID       e19058e6-e829-45df-8727-9e07625b42bc
Created    2021-01-26 09:21:06.16162122 -0600 CST
dd@vm-limi201c:~$
```

Figure 1: Dstask removes the tags from the task name. To see an individual task's detailed information (including tags), enter `dstask` followed by the task number (1). (You can ignore the messages in light gray from Git.)

```
dd@vm-limi201c:~
File Edit View Search Terminal Help
dd@vm-limi201c:~$ dstask note 1 return novels to Aunt Dorothy
Edit note 1: clean up basement
[master 70de3c8] Edit note 1: clean up basement
1 file changed, 1 insertion(+), 1 deletion(-)
dd@vm-limi201c:~$ dstask 1
Name      Value
ID        1
Priority   P2
Summary   clean up basement
Status    pending
Project
Tags       basement, private
UUID       e19058e6-e829-45df-8727-9e07625b42bc
Created    2021-01-26 09:21:06.16162122 -0600 CST

Notes on task 1:
return novels to Aunt Dorothy
dd@vm-limi201c:~$
```

Figure 2: Dstask lists notes at the end. Notes are also useful for checklists, where each note reflects an individual sub-task.

Listing 3: Pinning Notes

```
01 $ dstask note 1 return novels to Aunt Dorothy
02 $ dstask add clean up basement +basement P2 / return novels to Aunt Dorothy
```

Listing 4: Organizing Tasks in Projects

```
01 $ dstask add clean up garage +private P3 project:spring cleaning
02 $ dstask add declutter shoe closet +Private P1 project:Spring Cleaning
03 $ dstask modify 1 -Private +Books P1 project:Spring cleaning
```

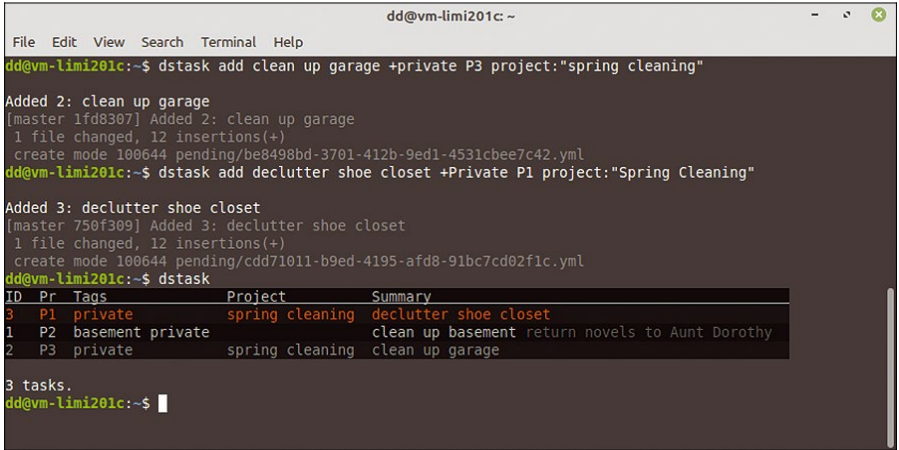


Figure 3: When you call `dstask`, you see a list of pending tasks. `Dstask` displays any existing notes in light gray in the Summary column.

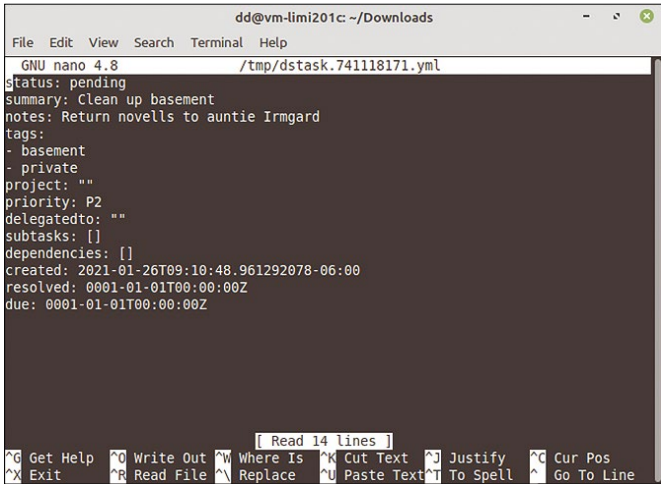


Figure 4: The editor shows you how `dstask` stores the task under the hood: a text file in YAML format.

spring cleaning project specified after the `project:` parameter. (Note again that `dstask` is not case sensitive.)

As shown in Figure 3, a call to `dstask` reveals that the original task `clean up basement` does not yet belong to any project. To change this, you can quickly modify the information by typing

```
dstask modify 1 project:Spring cleaning
```

The number again stands for the corresponding task ID (where 1 is the basement cleanup task). You can use `modify` to adjust the priority, keywords, and the project itself; a minus sign removes the tag in question (Listing 4, line 3).

Targeted Intervention

Use the `dstask edit 1` command to make a correction in a note. `Dstask` then opens all the details of the specified task in a text editor (Figure 4).

shown in Listing 5. When editing a task, make sure that you only adjust the values to the right of the colons. As shown in Figure 3, modify the project

name to the right of the `project:` parameter. When you are done, save your adjustments and exit the editor (in nano, press `Ctrl + O` followed by `Ctrl + X`).

While I was removing the sneakers from the shoe closet, I also dumped the waste paper in the recycling bin. To retroactively log this completed task, `dstask` has a `log` action. It works exactly like `add`, but immediately sets the project to resolved (Listing 6).

In the meantime, quite a few spring cleaning tasks have accumulated. The most urgent are revealed by typing `dstask` or `dstask next`. The task manager sorts the list by the priority and the age of the task, placing your next task at the top of the list. To see all the tasks that you have started but not yet finished, enter:

```
dstask show-active
```

To see all paused tasks, enter:

```
dstask show-paused
```

`Dstask` supports some further display formats (see Table 1). If `dstask` does not find any matching tasks or data for an action, it just stubbornly provides a command overview.

In Context

Tasks have a tendency to accumulate over time resulting in unwieldy to-do lists. To filter out unwanted tasks, you

The editor that opens is defined by the `$EDITOR` environment variable. This is often nano, but it could also be Vim or another editor. Some distributions, such as Ubuntu 20.04 and later, do not set a text editor. Instead, you have to set the environment variable explicitly with the commands

Listing 5: Setting the Environment Variable

```
$ echo "export EDITOR=nano" > ~/.bash_profile
$ source ~/.bash_profile
```

Listing 6: Logging Tasks

```
$ dstask log waste paper dumped +Private P3 project:spring cleaning
```

Table 1: Display Formats and Commands

Action	Function
<code>next</code>	List of major tasks
<code>show-projects</code>	List of all projects including the resolved tasks
<code>show-tags</code>	List of all assigned tags
<code>show-active</code>	List of all started tasks
<code>show-paused</code>	List of all tasks that were started and then paused
<code>show-open</code>	List of all tasks not yet finished
<code>show-resolved</code>	List of all completed tasks
<code>show-templates</code>	List of all templates
<code>show-unorganised</code>	List of all tasks that do not have a tag or are not assigned to a project

Listing 7: Context

```
$ dstask context +private -basement project:spring cleaning
```

```
dd@vm-limi201c:~
File Edit View Search Terminal Help
dd@vm-limi201c:~$ dstask
ID Pr Tags Project Summary
3 P1 private spring cleaning declutter shoe closet
1 P2 basement private spring cleaning clean up basement return novels to Aunt Dorothy
2 P3 private spring cleaning clean up garage

3 tasks.
dd@vm-limi201c:~$ dstask context +private -basement project:"spring cleaning"
dd@vm-limi201c:~$ dstask
Active context: +private -basement project:spring cleaning
ID Pr Tags Project Summary
3 P1 private spring cleaning declutter shoe closet
2 P3 private spring cleaning clean up garage

2 tasks.
dd@vm-limi201c:~$
```

Figure 5: The clean up basement task does not match the rules of the defined context, so it no longer appears in the list of pending tasks.

can use the context command to specify the filtering criteria (Listing 7).

The command in Listing 7 tells dstask to only pay attention to tasks tagged as private that do not contain the basement keyword, but that do belong to the spring cleaning project. For now, the tool ignores all other tasks. If you create a new task at this point, it will automati-

cally be tagged private and will belong to the spring cleaning project.

To reset the filter, type:

```
dstask context none
```

This tells dstask to get back to work on all existing tasks again. A call to dstask reveals the active context. If you are

missing a task, check the context first (Figure 5).

Templates

You probably need to clean up your desk far more often than your basement. For recurring tasks, templates can save you some typing (Figure 6). Templates are initially conventional tasks, but you create them using the template keyword (Listing 8, line 1). To show all existing templates, enter

```
dstask show-templates
```

which ignores any remaining actions.

To create a new task based on a template, you run add again, but reference the template. The command from line 2 of Listing 8 tells dstask to create a new task that takes all the settings from the template ID 4, but uses the *Sort pencils* string as the summary.

Git Games

The dstask git log command (Figure 7) returns the latest completed actions. Actions can be undone by typing

Shop the Shop → shop.linuxnewmedia.com

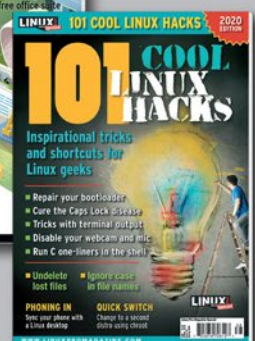
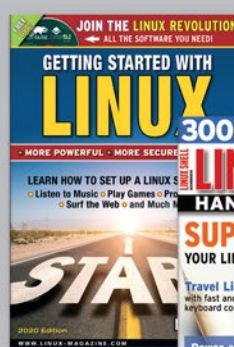
Discover the past and invest in a new year of IT solutions at Linux New Media's online store.

Want to subscribe?

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

➤ shop.linuxnewmedia.com

DIGITAL & PRINT
SUBSCRIPTIONS



SPECIAL EDITIONS

`dstask undo 3`, where the number (3) indicates the number of actions to be revoked, starting with the last action performed.

If you want to synchronize your task collection with other devices, `dstask` needs to be installed on those devices as well. In addition, you need a Git repository that all devices can access, such as one on GitHub. Make sure that only you have access to the repository; do not make it public to the entire Internet.

First, create the `~/dstask/` directory on all devices and clone the contents of the external Git repository to it using `git clone`. You can now create and manage your tasks in the usual way. To tell `dstask` to push all the changes to the external Git repository, call `dstask sync`.

On another PC, call `dstask sync` again. This tells `dstask` to sync with the Git repository and fetch the current state. Under the hood, `dstask` calls the `git pull` and `git push` commands. If you are familiar with Git, you can run other Git commands via the `dstask git` action.

More Efficient Typing

If you want to manage many tasks regularly in Bash with `dstask`, the work can be accelerated significantly by using an alias or script. You can create an alias, for example, using `alias t=dstask`. With this alias, you now just need to type `t` instead of `dstask`, which could save some typing.

You can also use a script. You'll find a `.dstask-bash-completions.sh` file on `dstask`'s GitHub page. If you run this by typing

```
source .dstask-bash-completions.sh
```

you will find that Bash now knows all the `dstask` parameters and automatically completes them when you press the Tab key.

However, this only works if you have moved `dstask` to `/usr/local/bin/` or another appropriate location in `$PATH`. To avoid having to repeatedly type the command discussed here, you might want to add it to your `~/bashrc` file.

```
dd@vm-limi201c:~
File Edit View Search Terminal Help
dd@vm-limi201c:~$ dstask add template clean up desk +Private P3 project:administration
Added 1: template clean up desk
[master (root-commit) 03e63a4] Added 1: template clean up desk
1 file changed, 12 insertions(+)
 create mode 100644 pending/780866ee-f892-4c5d-9914-7bb6ead064d5.yml
dd@vm-limi201c:~$
```

Figure 6: First a new template is created and then a new task is created based on the template. Since `dstask` treats the templates like tasks internally, the template is assigned an ID of 4.

```
dd@vm-limi201c:~
File Edit View Search Terminal Help
dd@vm-limi201c:~$ dstask git log
commit 0d775917a5c15118dc5841d752e78bd43445b3ad (HEAD -> master)
Author: LMI <lmi@datadesign-online.de>
Date: Wed Jan 27 02:36:59 2021 -0600

    Added 6: sort pencils

commit 42364e9758185d2408c1b65e84e3dd27139a5eb8
Author: LMI <lmi@datadesign-online.de>
Date: Wed Jan 27 02:29:00 2021 -0600

    Logged waste paper dumped

commit 4e83fd5f2d4ad325e8df489987704906fe482017
Author: LMI <lmi@datadesign-online.de>
Date: Wed Jan 27 02:26:58 2021 -0600

    Modified 1: clean up basement

commit 0ad40d4adc3cc1e4fc5ca389b0e607b43c4b633
Author: LMI <lmi@datadesign-online.de>
Date: Wed Jan 27 02:26:47 2021 -0600

    Added 5: declutter shoe closet

commit 32c2c485286139426f2822091307261c556a8745
Author: LMI <lmi@datadesign-online.de>
Date: Wed Jan 27 02:26:32 2021 -0600

    Added 4: clean up garage
```

Figure 7: The `git` action lets you directly invoke other Git commands including `log`.

Listing 8: Templates

```
01 $ dstask template clean up desk +Private P3 project:administration
02 $ dstask add template:4 sort pencils
```

If you want to set up autocompletion in Zsh, use the `.dstask-zsh-completions.sh` script from the GitHub page.

Conclusions

`Dstask` quickly manages all your pending tasks at the command line. If you frequently manage your system at the keyboard in a terminal window, you can particularly benefit from this approach to task management. The range of functions is fine for everyday use, but it does not come close to replacing a full-fledged task manager.

Having said this, the developer does plan to add support for sub-tasks and advanced project management in future releases. For example, you will be able to assign deadlines for tasks. The documentation is currently limited to the online help retrieved by `dstask help`. If you append an action such as `add` to this command, you will see a detailed description for the respective command only. ■■■

Info

[1] `dstask`:
<https://github.com/naggie/dstask>

Most
Extraordinary!

THE ULTIMATE

raspberrypi GEEK

ARCHIVE



GET EVERY ISSUE OF
RASPBERRY PI GEEK
ON ONE
FULLY-SEARCHABLE
DVD!



Order Now!

shop.linuxnewmedia.com



Getting to know the Python installer

pip3 Primer

As a replacement for pip, pip3 offers a complete solution for binary packages. Here's how to get started with this increasingly popular Python installer. *By Bruce Byfield*

You can tell the popularity of Linux tools by how often they are used to install packages outside of distro repositories. For years, source packages have been installed by the trinity of commands `configure`, `make`, and `make install` for compiling. More recently, `deb` packages have become the norm, usually configured for Ubuntu. Both these installation choices are still popular, but, today, the popularity of the Python programming language is reflected in the increasing use of pip3 [1].

Pip3 is the Python installer for Python 3.x releases. It replaces pip, which is used for earlier versions of Python, as well as `easy_install`, and can also work with eggs [2] and wheels [3], two other

Author

Bruce Byfield is a computer journalist and a freelance writer and editor specializing in free and open source software. In addition to his writing projects, he also teaches live and e-learning courses. In his spare time, Bruce writes about Northwest coast art (<http://brucebyfield.wordpress.com>). He is also co-founder of Prentice Pieces, a blog about writing and fantasy at <https://prenticepieces.com/>.

standard Python tools designed to simplify packaging. In basic functionality, it is similar to package managers in other programming languages, such as npm in JavaScript or gem in Ruby. You should note, however that in a command, pip3 is often referenced as pip.

Like the packages in a distribution's repositories, pip3 automatically handles dependencies, installing them first to avoid the problems of a half-installed package. In addition, pip3 packages are simpler to build than their predecessors and include detailed error and warning messages, as well as a requirement file [4] that makes cloning installations on multiple machines easier. In fact, pip3 is so similar to deb and RPM that it is often talked about in terms of distributions and packages, a habit that causes considerable confusion. Regardless, pip3 has advantages for developers and users alike.

Recent distributions often install Python 3 along with pip3 and pip by default. You can check which you have by running:

```
python --version
pip3 --version
```

Some applications require a specific earlier version of Python, which you can download from the Python website [5]. If you need to upgrade pip3, run:

```
python3 -m pip install --upgrade pip
```

If you need to install or upgrade pip, you can usually install it with

```
python -m ensurepip --default-pip
```

or by one of the other means listed on the Python packaging page [6].

You may also want to sandbox the packages you install in a virtual environment. Python has several tools for creating a virtual environment, including `virtualenv`, `pyenv`, and `venv`. In Python 3.4 and later, the preferred tool is `venv` [7], which may need to be installed separately. To create the virtual environment, first install Python and add the directory to your `$PATH` variable if necessary. Then create the directory for the environment and activate it with:

```
python3 -m venv DIRECTORY-PATH
source DIRECTORY/activate
```

```
bb@nanday:~$ python -m pip install "Ansible"
DEPRECATION: Python 2.7 reached the end of its life on January 1st, 2020. Please upgrade your Python as Python 2.7 is no longer maintained. pip 21.0 will drop support for Python 2.7 in January 2021. More details about Python 2 support in pip can be found at https://pip.pypa.io/en/latest/development/release-process/#python-2-support pip 21.0 will remove support for this functionality.
```

Figure 1: Starting in January 2021, Python 2 and pip are no longer officially supported. In Python 3, pip refers to what until now has been pip3.

```
bb@nanday:~$ python3 -m pip install "Django"
Defaulting to user installation because normal site-packages is not writeable
Collecting Django
  Using cached Django-3.1.6-py3-none-any.whl (7.8 MB)
Requirement already satisfied: sqlparse>=0.2.2 in ./local/lib/python3.7/site-packages (from Django) (0.4.1)
Requirement already satisfied: pytz in ./local/lib/python3.7/site-packages (from Django) (2021.1)
Requirement already satisfied: asgiref<4,>=3.2.10 in ./local/lib/python3.7/site-packages (from Django) (3.3.1)
Installing collected packages: Django
Successfully installed Django-3.1.6
```

Figure 2: Since pip3 installs with a connection to PyPI, the basic installation command is simple.

When you install a package or do any other work in Python, you will be using the specified virtual environment so long as you are in its directory until you type deactivate.

Using pip3

When a project under development uses pip3 to install, it will probably give the command to install its packages. However, you may also find packages to download in the Python repositories. The main general repository is the Python Package Index (PyPI, aka the Cheese Shop) [8]. However, there are dozens of others. Some are general repositories like Awesome Python. Others are concerned with special interests, such as scikit-learn and Python Design Patterns, or focus on major Python specific applications such as Ansible or Django.

The basic command structure for installing a package using pip3 depends on where the package is in relation to your machine. In each case, the `-m` option tells Python to search for the specified package, followed by the pip3 sub-command (`install`), the package, and then specifications for the version, if any. Pip for Python 2 uses a similar structure, but as of January 2021 it is no longer supported (Figure 1), although it may still work.

There are several common installation choices. To install the latest version, use

```
python3 -m pip install "PACKAGE"
```

Specify the directory for the local package file, the URL for version control, or the

Python repository. (The package name alone is needed for PyPI, as in Figure 2).

To install a specific version, use:

```
python3 -m pip install "PACKAGE==VERSION"
```

To install a version that is greater than or equal to one version and less than another version, use:

```
python3 -m pip install "PACKAGE">=VERSION1,<HIGHER-VERSION"
```

To install a version compatible with a certain version, use:

```
python3 -m pip install "PACKAGE~=VERSION"
```

This is usually a point release of the version specified.

Notice that because a connection to PyPI automatically exists, only the package name needs to be specified for packages in PyPI. The same is true for local packages on `$PATH` or the current directory, or the directory for storing wheels (usually `/usr/share/python-wheels`). With the `-requirement FILE` (`-r FILE`) option, you can also use a requirement file for the pack-

age to install or the URL to a project's version control system (VCS).

Other formats are available to help users deal with the variety of different Python releases.

Besides `install`, pip3 supports a number of related commands:

- `uninstall`: Removes a package
- `list`: Displays all installed packages on a system (Figure 3)
- `show`: Displays information about a package (Figure 4)
- `search`: Locates a package on PYPI

In addition, pip3 includes several commands for building packages, such as `freeze`, which outputs a requirement file (Figure 5) or a list of dependencies that can be used in packaging or troubleshooting, and `wheel`, which creates a type of file for streamlining package installation. However, these commands are separate articles in themselves.

```
bb@nanday:~$ python3 -m pip list
Package            Version
-----
argcomplete        1.8.1
arrow               0.17.0
asgiref            3.3.1
asn1crypto         0.24.0
attrs              18.2.0
Automat            0.6.0
beautifulsoup4    4.7.1
binwalk            2.1.2
blinker            1.4
borgbackup         1.1.9
```

Figure 3: The `list` command displays all Python packages installed.

```
bb@nanday:~$ python3 -m pip show "Django"
Name: Django
Version: 3.1.6
Summary: A high-level Python Web framework that encourages rapid development and clean, pragmatic design.
Home-page: https://www.djangoproject.com/
Author: Django Software Foundation
Author-email: foundation@djangoproject.com
License: BSD-3-Clause
Location: /home/bb/.local/lib/python3.7/site-packages
Requires: pytz, sqlparse, asgiref
Required-by:
bb@nanday:~$
```

Figure 4: The `show` command gives detailed information about a package.

You can also use options to refine a `pip3` command. The `--upgrade (-u)` option ensures you install the very latest version of a package, although at times it may cause problems with dependencies. Similarly, you may want to use `--eggs`, for compatibility with older systems that have eggs installed. You can also install custom options with `--install OPTION`, such as:

```
--install-option =Z
"--install-scripts=/usr/local/bin"
```

For custom requirement files, you can use `--requirement FILE (-r)`.

Still other install options cover the usual range of options such as forcing a reinstall (`--force-reinstall`), not installing dependencies (`--no-deps`), or specifying the target directory for packages (`-t, --target DIRECTORY`). You can even use `--pre` to override `pip3`'s default setting to search for only stable releases and search for pre-release and development builds.

```
bb@nanday:~$ python3 -m pip freeze "Django"
argcomplete==1.8.1
arrow==0.17.0
asgiref==3.3.1
asn1crypto==0.24.0
attrs==18.2.0
Automat==0.6.0
beautifulsoup4==4.7.1
binwalk==2.1.2
blinker==1.4
borgbackup==1.1.9
```

Figure 5: The requirement file for Django, showing the required dependencies.

When uninstalling, `pip3` offers the option to delete all the packages in a requirement file with `--requirement FILE (-r FILE)`. If necessary, multiple requirement files can be specified. With `--yes (-y)`, `pip3` will not ask for confirmation before deleting files. More general options are available as well, such as `--verbose (-v)`, which has three levels of output, or `--quiet (-q)`, which gives less output. You can also change the default logfile (`~/pip/pip.log`), positioning a standard log with `--log-path PATH` or a verbose log with `--log PATH`.

The Next Stage

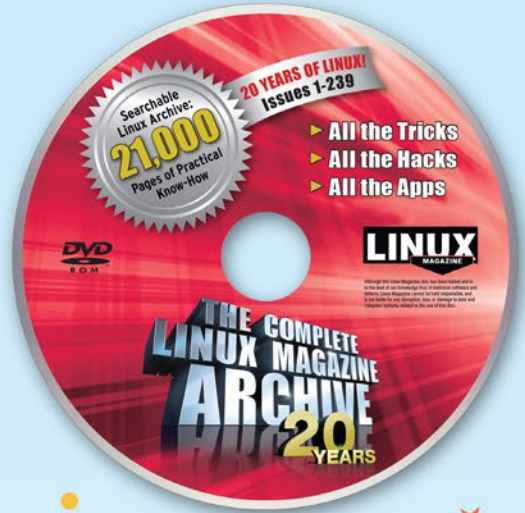
`Pip3` is a complete solution for binary packages. It offers compatibility with many of the other package tools that Python has evolved over the years, and – just as importantly – does so while living up to Python's reputation for clarity and organization. You have only to contrast `pip3` to Debian's `apt-get` to see what a difference paying attention to structure can make.

Probably, the information given here is more than enough to guide those unfamiliar with `pip3`. However, to give a complete picture, I should also mention `setuptools` [9], a general command for managing packages. While `pip3` is more than enough for small-scale operations, such as when trying an application still in development, `setuptools` is better suited for large-scale operations. Should you find `pip3` straining to meet your packaging needs, have a look at `setuptools`. If the popularity of Python continues to grow, I have a hunch that we will be seeing a lot more of both. ■■■

Info

- [1] `pip3`: <https://www.linuxscrew.com/install-pip>
- [2] Eggs: <http://peak.telecommunity.com/DevCenter/PythonEggs>
- [3] Wheels: <https://realpython.com/python-wheels/>
- [4] Requirement file: <https://blog.jcharistech.com/2020/11/02/how-to-create-requirements-txt-file-in-python/>
- [5] Downloading Python: <https://www.python.org/downloads/>
- [6] Python packaging: <https://packaging.python.org/tutorials/packaging-projects/>
- [7] `venv`: <https://docs.python.org/3/library/venv.html>
- [8] `PyPI`: <https://pypi.org/>
- [9] `setuptools`: <https://setuptools.readthedocs.io/en/latest/index.html>

20 YEARS LINUX MAGAZINE



LINUX MAGAZINE 20 YEAR ARCHIVE DVD

ORDER NOW!

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



The sys admin's daily grind: katoolin 3

Kali à la Carte

Charly uses the katoolin 3 installation script for a targeted approach to installing his favorite Kali Linux tools on the Ubuntu desktop. *By Charly Kühnast*

You probably know Kali Linux [1], which comes with a wealth of tools for forensics and penetration testing. You can install the distribution or boot it as a Live system from a USB stick. Users of mainstream Linux often wish to use Kali's range of functions in their preferred distribution. Katoolin [2] tries to fulfill this wish.

I first looked at katoolin in 2017. The script collection for installing Kali Linux tools was tailor-made for the then current Ubuntu LTS and written in Python 2. However, the way it integrated into Ubuntu was a pretty brutal process, involving autonomous changes to the system configuration and frequently ending up with the distribution getting into a total mess on the next update. But the idea was still good, and now we have katoolin 3, which finally files the rough edges off the process. Ported to Python 3, it integrates smoothly into the system

and no longer interferes with Ubuntu's own tools.

To get katoolin running, you need to include the universe repository (Listing 1, line 1); you also need Git to clone the repository (line 2). Once these dependencies are installed, you are ready to go. I mirrored the code to my Ubuntu (line 3), changed to the newly created `katoolin3/` directory in the next step, made the `install.sh` file executable, and ran it (lines 4 to 6). It makes sense to have a look at the `install.sh` file's code beforehand – Git repos can be compromised and can contain malicious code. If everything looks good, the installation can start; this will ideally complete with a success message (line 8).

Since katoolin 3 is an installation tool, I have to run it with root privileges (line 9). The main menu appears on the screen. Theoretically, I could now just treat myself

to the full Kali Linux treasure trove by choosing *Install all* – but hardly anyone really needs the full arsenal. Instead press `0` to see a list of the available categories (Figure 1). From there, you can branch into, say, *Wireless Attacks*. Entering the numbers to the left of the individual tools starts the installation. You can also specify ranges or lists, such as `1,3,5-7`.

Katoolin 3 does not update automatically, so you do have to run `update.sh` from time to time from the `katoolin3/` directory created earlier. This is also where you will find `uninstall.sh` in case you want to get rid of everything. Because one thing is clear: Although katoolin 3 does give you a taste of the tools in the Kali collection, sooner or later you will end up wanting to permanently install Kali Linux. ■■■

Listing 1: Installing katoolin 3

```
01 $ sudo add-apt-repository universe
02 $ sudo apt install git
03 $ git clone https://github.com/s-h-3-1-1/katoolin3
04 $ cd katoolin3/
05 $ chmod +x ./install.sh
06 $ sudo ./install.sh
07 [...]
08 Successfully installed.
09 $ sudo katoolin3
```

Info

[1] Kali Linux: <https://www.kali.org>

[2] katoolin 3: <https://github.com/s-h-3-1-1/katoolin3>

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.



```
Main Menu
0) View Categories
1) Install All
2) Uninstall All
3) Search repository
4) List installed packages
5) List not installed packages
6) Install Kali Menu
7) Uninstall old katoolin
8) Help
9) Exit

kat> 0

Select a Category
0) Exploitation Tools      8) Sniffing & Spoofing
1) Forensics Tools        9) Stress Testing
2) Hardware Hacking       10) Vulnerability Analysis
3) Information Gathering  11) Web Applications
4) Maintaining Access     12) Wireless Attacks
5) Password Attacks       13) HELP
6) Reporting Tools        14) BACK
7) Reverse Engineering
```

Figure 1: Katoolin 3 menus.

Shop the Shop
shop.linuxnewmedia.com

Become a LibreOffice Expert



Explore the **FREE** office suite used by busy professionals around the world!

Create Professional:

- Text Documents
- Spreadsheets
- Presentations
- Databases

Whether you work on a Windows PC, a Mac, or a Linux system, you have all you need to get started with LibreOffice today. This single-volume special edition will serve as your guide!

Order online:
shop.linuxnewmedia.com/specials

For Windows, macOS, and Linux users!

Keep control of goroutines with a Context construct

Fighting Chaos

When functions generate legions of goroutines to do subtasks, the main program needs to keep track and retain control of ongoing activity. To do this, Mike Schilli recommends using a Context construct. *By Mike Schilli*

Go's goroutines are so cheap that programmers like to fire them off by the dozen. But who cleans up the mess at the end of the day? Basically, Go channels lend themselves to communication. A main program may need to keep control of many goroutines running simultaneously, but a message in a channel only ever reaches one recipient. Consequently, the communication partners in this scenario rely on a special case.

If one or more recipients in Go try to read from a channel but block because there is nothing there, then the sender can notify all recipients in one fell swoop by closing the channel. This wakes up all the receivers, and their blocking read functions return with an error value.

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.



This is precisely the procedure commonly used by the main program to stop subroutines that are still running. The main program opens a channel and passes it to each subroutine that it calls; the subroutines in turn attach blocking read functions to it. When the main program closes the channel later on, the blocks are resolved, and the subroutines proceed to do their agreed upon cleanup tasks – usually releasing all resources and exiting.

Ripcord on Three

As an intuitive example, Listing 1 shows a main program that calls the function `work()` three times in quick succession. The function returns almost immediately, but each time it sets off a goroutine internally that counts to 10 in one second steps and prints the current counter value on standard output each time. Each of these goroutines would now continue to run for their entire scheduled 10 seconds, even after the function calling them had long terminated, if it weren't for the main program call to `close()`, pulling the ripcord after three seconds in line 15.

The `done` channel, used as a mechanism for this synchronization, is created in line 9. `interface{}` specifies the data

type transported in the channel as generic, since the program does not send any data to the channel or read from it later on, but will only detect a future `close` command.

How exactly do the workers get to hear the factory siren at this point? After all, they are still busy doing their job and compiling results or just busily passing time, as in this example.

This “busy waiting” is implemented by the `select` construct starting in line 24. Using two different case statements, `select` waits simultaneously for one of two possible events: the channel reader `<-done` attempts to obtain data from the `done` channel or it picks up an error if `main` closes the channel. Or else the timer started by `time.After()` in line 28 expires after one second and directs the `select` construct to jump to the corresponding empty case, which does nothing except continue the surrounding endless for loop.

During the first three seconds of running the sample program, it will show timer ticks from all three subroutines. But shortly after, things start happening, because the main program closes the `done` channel, which triggers an error in the first case in line 25; this in turn causes the worker to display the `Ok. I`



quit. message and exit its goroutine with `return`. Figure 1 shows the running program's output.

And Now for Real

Instead of counting to 10 and always waiting a second between each step, a `work()` function in the real world would, for example, perform time-consuming tasks such as retrieving a web page over the network or using a `tailf`-style technique to detect growth in one or more locally monitored files. However, even in these situations, a server program may have to pull the emergency brake – whether because the requesting user has lost patience or data processing on the back end is simply taking too long for the main program and it wants to move on to serve other requests.

At the end of a standalone `main` program such as the one in Listing 1, the operating system does indeed clean up any goroutines that are still running and automatically releases the allocated resources such as memory or file handles. However, a server program must not rely on this luxury, because canceling a request – for whatever reason – does not terminate the program. It may have to continue running for weeks, without orphaned functions utilizing more and more memory, until the dreaded out-of-memory killer steps in and takes everything down.

```
$ go build grtest.go
$ ./grtest
0
0
0
1
1
1
2
2
2
Ok. I quit.
Ok. I quit.
Ok. I quit.
$
```

Figure 1: In the test program shown in Listing 1, the siren goes off for all three workers after three seconds.

Limits

By the way, functions that send data to each other via channels must be aware of two borderline cases. If they send a message to a channel that has already been closed, the Go program goes into panic mode and aborts with an error. On the other hand, if a function reads from a channel where nobody can send anything anymore because all writers have given up the ghost, the program flow will hang forever. In the case at hand, however, that's the intended behavior: No data will ever flow through the channel used, because the program only takes advantage of the fact that reading from a closed channel generates an error.

Simpler with Context

To wrap this common synchronization pattern into an easily deployed package, the Go standard library provides `Context` structs [1]. They are used in Google's data centers by servers, which often have to call many goroutines to compile the results for incoming user requests. If this takes too long, the main function handling the request must be able to contact all the goroutines that are still running, forcing them to immediately abandon their current work, release any allocated resources, and terminate their program flow.

The context's interface uses `Done()` to return an open channel from which the worker bees try to read. However, no message ever reaches the channel (like in the previous example). Instead, the main program calls `close()` to close the channel abstracted in the struct when it's time for the final whistle, using the context's exported `Cancel()` function, which suddenly gives the reading workers an error value. They field this and interpret it as a signal to close up shop.

Listing 2 imports `context` to introduce the standard library package of the same name in line 4. The `context.WithCancel()` function extends a standard context created by `context.Background()` and returns two things: a context object in `ctx` and a `cancel()` function that the programmer calls later on (in line 17 in Listing 2) to send the signal to end the party and turn off the lights.

Worker bees use `ctx.Done()` to extract the channel to be monitored from the context and insert a case statement with

Listing 1: grtest.go

```
01 package main
02
03 import (
04     "fmt"
05     "time"
06 )
07
08 func main() {
09     done := make(chan interface{})
10     work(done)
11     work(done)
12     work(done)
13
14     time.Sleep(3 * time.Second)
15     close(done)
16     time.Sleep(3 * time.Second)
17 }
18
19 func work(done chan interface{}) {
20     go func() {
21         for i := 0; i < 10; i++ {
22             fmt.Printf("%d\n", i)
23
24             select {
25                 case <-done:
26                     fmt.Printf("Ok. I quit.\n")
27                     return
28                 case <-time.After(time.Second):
29                 }
30             }
31         }()
32     }
```

a read operation into their `select` loops; they then use this mechanism to receive control commands from their caller.

After compilation, the output from Listing 2 looks exactly like Figure 1 and exhibits exactly the same behavior. This is not surprising, since the context implementation uses the same internal infrastructure.

Inside Google's Brain

In Google's data centers, all worker functions use a context variable as their first parameter; this controls any premature termination that may be necessary. However, it also helps pass down payloads of received requests, such as the name of the authenticated user or credentials for subsystems. In this way, all subsystems across all API boundaries support certain standard functions such as timeouts, cleanup signals due to unsolvable prob-

lems, or simply convenient access to global key/value values.

Brake and Lose

Listing 3 shows what this kind of server function could look like in a practical use case. It retrieves four URLs: the Google, Facebook, and Amazon splash pages, along with the artificially slowed down website `deelay.me`. Line 23 shows that it calls the AOL website with a delay of 5,000 milliseconds to make the client think it has a lame Internet connection.

The main program now goes through these URLs in the for loop starting in line 28 and passes each one to the `chkurl()` function, along with a context variable and a `results` channel. The latter returns the workers' results to the main program in the form of `Resp` struc-

tures. This data type, defined starting in line 10, stores the URL obtained along with the request's HTTP return code.

In the process, `chkurl()` processes the requests asynchronously. It starts a goroutine from line 42 for time-consuming retrieval over the web and therefore quickly returns to the main program. Results bubble up later on via the `results` channel, where the for loop collects them starting in line 32, outputting the URLs along with their numeric result codes.

Delays Are Punished

To prevent the `chkurl()` worker from dilly-dallying, line 16 sets a context with a timeout, which it passes as the `ctx` variable to the worker bee. It uses the default `net/http` package in an inner goroutine (starting in line 44) running inside an outer goroutine to retrieve the webpage, and pushes the status code returned by the web server into the local `httpch` channel when it's available.

This means that the inner goroutine is stuck until someone at the other end of the `httpch` channel starts reading. This will happen in the subsequent `select` construct starting in line 53, which kicks in on

one of two events: either when a web request response comes from the `httpch` channel, or when the main program loses patience and `ctx.Done()` returns with an error. In the latter case, the function outputs a *Timeout!!* message and sets the HTTP return code to `501`. If, on the other hand, everything works just fine, line 55-56 pushes the returned code and the associated URL into the `results` channel.

Figure 2 shows the flow of the compiled binary. The first three requests come back relatively quickly. The fourth one would keep dawdling for a massive five seconds, but the context timeout of two seconds as defined in line 16 sends an early termination signal, and the worker bee's status code comes back as a `501`, as specified in line 60.

The combination of two nested goroutines in `chkurl()` is necessary, by the way, because sending data into a channel and extracting the data at the other end needs to be synchronized. Letting the program simply send to the channel first and then read from it will not work, as the sender will hang forever if no receiver is listening. To ensure that the whole enchilada in Listing 3 runs smoothly, the sender sends its values to the channel in an asynchronous goroutine. The receiver can dock afterwards at their leisure; as soon as they do so, the sender unblocks and data gets sent through the channel.

Precision, No Sleep

You might have noticed that Listing 3 does not rely on unreliable `Sleep` commands for synchronization when collecting worker data, as done sloppily in Listings 1 and 2. Sleeping for a set time is no guarantee that things have fin-

Listing 2: context.go

```
01 package main
02
03 import (
04     "context"
05     "fmt"
06     "time"
07 )
08
09 func main() {
10     ctx, cancel := context.WithCancel(context.Background())
11
12     work(ctx)
13     work(ctx)
14     work(ctx)
15
16     time.Sleep(3 * time.Second)
17     cancel()
18     time.Sleep(3 * time.Second)
19 }
20
21 func work(ctx context.Context) {
22     go func() {
23         for i := 0; i < 10; i++ {
24             fmt.Printf("%d\n", i)
25
26             select {
27                 case <-ctx.Done():
28                     fmt.Printf("Ok. I quit.\n")
29                     return
30                 case <-time.After(time.Second):
31                     }
32             }
33         }()
34     }
```

```
$ go build delay.go
$ ./delay
Fetching https://www.google.com
Fetching https://www.facebook.com
Fetching https://www.amazon.com
Fetching https://deelay.me/5000/www.aol.com
Received: 200 https://www.google.com
Received: 200 https://www.amazon.com
Received: 200 https://www.facebook.com
[...]
Timeout!!
Received: 501 https://deelay.me/5000/www.aol.com
$
```

Figure 2: The results from the first three websites are returned quickly, but the fourth is too slow. This prompts the context to call a timeout.

ished, as data sent over the network can arrive unusually slowly. Anything is possible on the Internet; worst case, the program might exit before the last goroutine had a chance to send its results up the channel.

Listing 3 therefore uses two `for` loops in lines 28 and 32, each of which counts to four: once to call `chkurl()` four times and later to collect the results from the return channel four

times. The order of requests and responses can get mixed up this way, but the program structure guarantees that all the results are complete and nothing gets accidentally dropped.

Goroutines are a great way to deploy code with components that can run concurrently. Note, however, that the concurrent code will only run in parallel if the platform supports it (e.g., thanks to multithreading and/or a multi-core pro-

cessor). The difference between concurrency and parallelism is therefore quite relevant, as Go guru Rob Pike impressively explains in a video [2] that is definitely worth watching. ■■■

Info

- [1] Go Context: <https://blog.golang.org/context>
- [2] “Concurrency is not parallelism”: <https://blog.golang.org/waza-talk>

Listing 3: delay.go

```

01 package main
02
03 import (
04     "context"
05     "fmt"
06     "net/http"
07     "time"
08 )
09
10 type Resp struct {
11     rcode int
12     url   string
13 }
14
15 func main() {
16     ctx, cancel := context.WithTimeout(context.Background(),
17                                     2*time.Second)
18     defer cancel()
19     urls := []string{
20         "https://www.google.com",
21         "https://www.facebook.com",
22         "https://www.amazon.com",
23         "https://deelay.me/5000/www.aol.com",
24     }
25
26     results := make(chan Resp)
27
28     for _, url := range urls {
29         chkurl(ctx, url, results)
30     }
31
32     for _ = range urls {
33         resp := <-results
34         fmt.Printf("Received: %d %s\n", resp.rcode, resp.url)
35     }
36 }
37
38 func chkurl(ctx context.Context, url string, results chan
39             Resp) {
40     fmt.Printf("Fetching %s\n", url)
41     httpch := make(chan int)
42     go func() {
43         // async url fetch
44         go func() {
45             resp, err := http.Get(url)
46             if err != nil {
47                 httpch <- 500
48             } else {
49                 httpch <- resp.StatusCode
50             }
51         }()
52     }
53     select {
54     case result := <-httpch:
55         results <- Resp{
56             rcode: result, url: url}
57     case <-ctx.Done():
58         fmt.Printf("Timeout!!\n")
59         results <- Resp{
60             rcode: 501, url: url}
61     }
62 }()
63 }

```

A modern library interior with a curved staircase and bookshelves. The scene is brightly lit, with a blue overlay at the top and bottom. The text is centered on the blue overlay.

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 FUN

LEARN HOW TO SET UP A LINUX SYSTEM TO:

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

START

2020 Edition **LINUX Special**

WWW.LINUX-MAGAZINE.COM

300+ BEST BASH COMMANDS **SAVE 15%** on Linux Certification See details inside

LINUX SHELL HANDBOOK

2019 Edition **LINUX Special**

SUPERCHARGE YOUR LINUX SKILLS

Travel Light with fast and graceful keyboard commands

Power at Your Fingertips

- Manipulate text strings
- Pipe and redirect output
- Monitor processes
- Manage users and groups
- Create easy automation scripts

Keep this comprehensive guide as a permanent command reference!

WWW.LINUX-MAGAZINE.COM

FREE DVD! **LibreOffice Full Version**

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 MS Office Files

Create Professional:

- Text Documents
- Spreadsheets
- Presentations
- Databases

Replace MS Office and Google Docs!

LibreOffice®

Includes full versions for Windows, macOS, and Linux

WWW.LINUX-MAGAZINE.COM

LINUX Special **101 COOL LINUX HACKS** **2020 EDITION**

101 COOL LINUX HACKS

Inspirational tricks and shortcuts for Linux geeks

- Repair your bootloader
- Cure the Caps Lock disease
- Tricks with terminal output
- Disable your webcam and mic
- Run C one-liners in the shell
- Undelete lost files
- Ignore case in file names

PHONING IN
Sync your phone with a Linux desktop

QUICK SWITCH
Change to a second distro using chroot

LINUX Special

WWW.LINUX-MAGAZINE.COM

Looking for an edge with the classic Quicksort algorithm

Smart Sort

If you wanted to assign a flavor to the Quicksort algorithm, it would be sweet and sour. Sweet, because it is very elegant; sour, because typical implementations sometimes leave more questions than they answer.

By Rainer Grimm



The Quicksort sorting algorithm has been around for 60 years, and, if implemented properly, it is still the fastest option for many sorting tasks. According to the description on Wikipedia, a well designed Quicksort is "...somewhat faster than Merge sort and about two or three times faster than Heapsort."

Many Linux users today have studied Quicksort at some point in the past, through a computer science class or other training scenario, but unless you are working as a professional programmer, chances are it has been a few years since you have taken the time to ponder the elegant Quicksort algorithm. Still, sorting goes on all the time on Linux networks. You don't have to be a full-time app developer to conjure up an occasional script to rank results or order a set of values extracted from a log file. This article explores some of the nuances of the classic Quicksort.

Quicksort ABC

The Quicksort [1] algorithm originated with Tony Hoare [2], who first developed it in 1959 and published it in 1961. Quicksort is what is known as a divide-and-conquer algorithm. One element in the array is chosen to be the *pivot element*. All elements smaller than the pivot element

are then grouped in a sub-array before it, and all elements larger than the pivot element are placed in a sub-array after it. This process is then repeated with the sub-arrays: a pivot element is chosen, with smaller elements placed in a sub-array before and larger elements placed in a sub-array after. After a finite number of steps, the size of the sub-arrays becomes one, and at that point, the whole array has been sorted.

Too complicated? Figure 1 sums up the Quicksort algorithm. Boxes containing only one red number are already in the right position. In the first row, the number 6 is the pivot element. Now all of the elements are sorted in relation to 6. In the second row, the 5 and the 9 act as the new pivot elements. Now the partial arrays are sorted relative to 5 and 9. The result is the

third row, where almost all of the elements are already sorted. Only the 8 (the new pivot element) has to swap places with the 7. If you would prefer an animated clip of the Quicksort algorithm, check out the Quicksort page on Wikipedia [1].

If the array contains n elements, an average of $n \cdot \log(n)$ sorting steps are needed. The $\log(n)$ factor results from

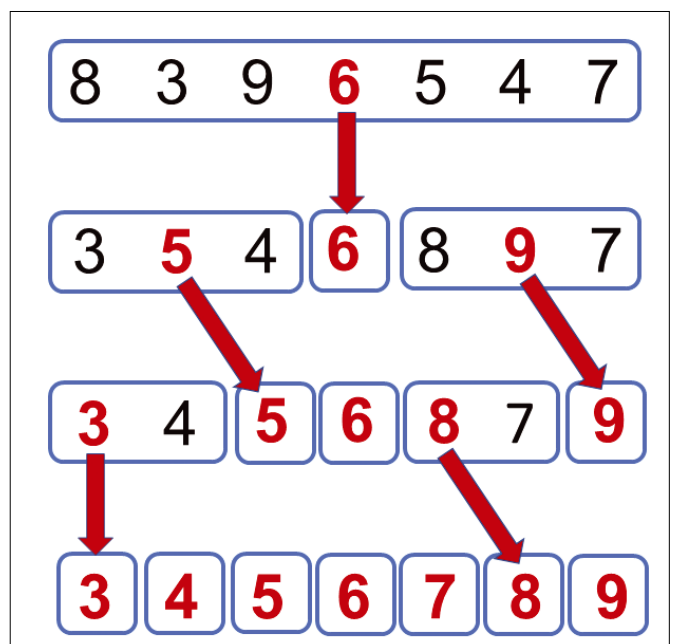


Figure 1: A schematic illustration of the Quicksort algorithm.

Lead Image © Olga Yastremska, 123RF.com

Listing 1: Quicksort in C

```

01 void quickSort(int arr[], int left, int right) {
02     int i = left, j = right;
03     int tmp;
04     int pivot = arr[(left + right) / 2];
05     while (i <= j) {
06         while (arr[i] < pivot) i++;
07         while (arr[j] > pivot) j--;
08         if (i <= j) {
09             tmp = arr[i];
10             arr[i] = arr[j];
11             arr[j] = tmp;
12             i++; j--;
13         }
14     }
15     if (left < j) quickSort(arr, left, j);
16     if (i < right) quickSort(arr, i, right);
17 }

```

the fact that the algorithm halves the array in each step. In a worst-case scenario, Quicksort requires $n \cdot n$ sorting steps. To illustrate this worst-case scenario, consider that, in Figure 1, the middle element served as the pivot element. But any other element could also serve as the pivot element. If the first element is the pivot element and the array is already sorted in ascending order, the array must be halved exactly n times, which would be the (very unlikely) worst case. Note that a few lines of text are all I needed to describe the elegant and highly efficient Quicksort algorithm, along with its performance characteristics.

First Encounter

The classic Quicksort implementation in C lacks charm and is quite successful at disguising its elegant design, as Listing 1

demonstrates. I won't provide a full description of the code; however, one observation is very interesting.

In Lines 9 to 11, the code overwrites the existing elements. Thus, the algorithm runs in-place and assumes mutable data. A nice expression has been established for the task of overwriting old values with new ones in functional programming: destructive assignment [3]. This takes us neatly to the next topic: in functional program-

ming languages like Haskell, Quicksort is represented in a far more elegant way.

Second Encounter

In Haskell, data is immutable, which precludes destructive assignment by design. The Haskell-based Quicksort algorithm in Listing 2 creates a new list in

each iteration, rather than acting directly on the array as in Listing 1. Quicksort in two lines? Is that all there is to it? Yes.

The `qsort` algorithm consists of two function definitions. The first line applies the defined Quicksort to the empty list. The second line represents the general case, where the list consists of at least one element: $x:xs$. Here, by convention, x denotes the beginning of the list and xs denotes the remainder.

The strategy of the Quicksort algorithm can be implemented almost directly in Haskell:

- use the first element of the list x as the pivot element;
- insert $((++)$) all elements in xs that are lesser than x ($(\text{qsort } [y \mid y <- xs, y < x])$) in front of the one-element list $[x]$;
- append all elements in xs that are at least as large as x to the list $[x]$ ($(\text{qsort } [y \mid y <- xs, y >= x])$).

The recursion ends when Quicksort is applied to the empty list. Admittedly, the compactness of Haskell seems unusual. However, this Quicksort algorithm can be implemented in any programming language that supports list comprehension –

Listing 2: Quicksort in Haskell

```

qsort [] = []
qsort (x:xs) = qsort [y | y <- xs, y < x] ++ [x] ++ qsort [y | y <- xs, y >= x]

```

Listing 3: Quicksort in Python

```

def qsort(L):
    if len(L) <= 1: return L
    return qsort([lt for lt in L[1:] if lt < L[0]]) + L[0:1] + qsort([ge for ge in L[1:] if ge >= L[0]])

```

Listing 4: Quicksort in C++

```

template <typename Forward>;
void quicksort(ForwardIt first, ForwardIt last) {
    if(first == last) return;
    auto pivot = *std::next(first, std::distance(first,last)/2);
    ForwardIt middle1 = std::partition(first, last, [pivot](const auto& em){return em < pivot;});
    ForwardIt middle2 = std::partition(middle1, last, [pivot](const auto& em){return !(pivot < em);});
    quicksort(first, middle1);
    quicksort(middle2, last);
}

```

```

>>> def qsort(L):
...     if len(L) <= 1: return L
...     return qsort([lt for lt in L[1:] if lt < L[0]]) + L[0:1] + qsort([ge for ge in L[1:] if ge >= L[0]])
...
>>> qsort([8, 3, 9, 6, 5, 4, 7])
[3, 4, 5, 6, 7, 8, 9]
>>>

```

Figure 2: Using the Quicksort algorithm in Python.

which leads to the more mainstream Python programming language in Listing 3.

The description of the Haskell algorithm can be applied almost verbatim to Python. The subtle difference is that in Python, `L[0:1]` acts as the first element and you express the list concatenation in

Python with the `+` symbol. The algorithm reliably performs its services in Figure 2.

Peaceful Combination

Admittedly, both implementation strategies for Quicksort have their advantages and disadvantages. The imperative strat-

egy in Listing 1 requires no additional memory but almost completely obscures the elegant structure of the algorithm. The functional strategy in Listing 2 and Listing 3 looks very elegant but creates a new list with each recursion. Is it possible to combine the advantages of both worlds?

Listing 5: Performance Test

```

01 #include <algorithm>
02 #include <chrono>
03 #include <execution>
04 #include <iomanip>
05 #include <iostream>
06 #include <iterator>
07 #include <numeric>
08 #include <random>
09 #include <vector>
10
11 template <typename ExecutionPolicy, typename ForwardIt>
12 void quicksort(ExecutionPolicy&& policy, ForwardIt first,
13               ForwardIt last) {
14     if(first == last) return;
15     auto pivot = *std::next(first,
16                           std::distance(first, last)/2);
17     ForwardIt middle1 = std::partition(policy, first, last,
18                                       [pivot](const auto& em){ return em < pivot; });
19     ForwardIt middle2 = std::partition(policy, middle1,
20                                       last, [pivot](const auto& em){ return !(pivot < em); });
21     quicksort(policy, first, middle1);
22     quicksort(policy, middle2, last);
23 }
24
25 std::vector<int> getRandomVector(int numberRandom, int
26                                 lastNumber) {
27     std::random_device seed;
28     std::mt19937 engine(seed());
29     std::uniform_int_distribution<int> dist(0, lastNumber);
30     std::vector<int> randNumbers;
31     randNumbers.reserve(numberRandom);
32     for
33         (int i=0; i < numberRandom; ++i){
34             randNumbers.push_back(dist(engine));
35         }
36     return randNumbers;
37 }
38
39 int main() {
40     std::cout << "random numbers between 0
41               and " << lastNumber << "\n\n";
42
43     std::vector tmpVec = getRandomVector(numberRandom,
44                                         lastNumber);
45
46     {
47         std::cout << std::fixed << std::setprecision(10);
48         auto begin = std::chrono::steady_clock::now();
49         quicksort(std::execution::seq, tmpVec.begin(),
50                 tmpVec.end());
51         std::chrono::duration<double> last=
52             std::chrono::steady_clock::now() - begin;
53         std::cout << "Quicksort sequential: " << last.count()
54                 << " *\n\n";
55     }
56
57     {
58         std::cout << std::fixed << std::setprecision(10);
59         auto begin = std::chrono::steady_clock::now();
60         quicksort(std::execution::par, tmpVec.begin(),
61                 tmpVec.end());
62         std::chrono::duration<double> last=
63             std::chrono::steady_clock::now() - begin;
64         std::cout << "Quicksort parallel: " << last.count()
65                 << " *\n\n";
66     }
67
68     std::cout << "\n\n";
69
70     {
71         std::vector tmpVec = randVec;
72         std::cout << std::fixed << std::setprecision(10);
73         auto begin = std::chrono::steady_clock::now();
74         std::sort(std::execution::seq, tmpVec.begin(),
75                 tmpVec.end());
76         std::chrono::duration<double> last=
77             std::chrono::steady_clock::now() - begin;
78         std::cout << "std::sort sequential: " << last.
79                 count() << " *\n\n";
80     }
81
82     {
83         std::vector tmpVec = randVec;
84         std::cout << std::fixed << std::setprecision(10);
85         auto begin = std::chrono::steady_clock::now();
86         std::sort(std::execution::par, tmpVec.begin(),
87                 tmpVec.end());
88         std::chrono::duration<double> last=
89             std::chrono::steady_clock::now() - begin;
90         std::cout << "std::sort parallel: " << last.count()
91                 << " *\n\n";
92     }
93     std::cout << " *\n\n";
94 }

```

```
C:\Users\seminar>quicksort.exe
100000000 random numbers between 0 and 100

Quicksort sequential: 2.0741171000
Quicksort parallel: 0.4034562000

std::sort sequential: 2.0640304000
std::sort parallel: 0.8953024000
```

Figure 3: Sorting 1,000,000 random numbers between 0 and 100.

```
C:\Users\seminar>quicksort.exe
100000000 random numbers between 0 and 10000

Quicksort sequential: 3.7672802000
Quicksort parallel: 1.1919046000

std::sort sequential: 4.2639335000
std::sort parallel: 0.9725637000
```

Figure 4: Sorting 1,000,000 random numbers between 0 and 10,000.

```
C:\Users\seminar>quicksort.exe
100000000 random numbers between 0 and 1000000

Quicksort sequential: 5.7838295000
Quicksort parallel: 30.0080108000

std::sort sequential: 6.5851260000
std::sort parallel: 1.3016558000
```

Figure 5: Sorting 1,000,000 random numbers between 0 and 1,000,000.

Yes, it is. The following implementation in C++ is based on `std::partition` [4], an algorithm from the Standard Template Library that partitions a range based on an element. `std::partition` in Listing 4 does the job of list comprehension in Listing 2 and Listing 3.

The Crucial Question

But this article has not yet answered the crucial question in C++: “Say, how fast are you?”

In C++17, C++ introduced support for the Parallel Standard Template Library (STL). Around 70 of the more than 100 algorithms in the STL can be executed sequentially, executed in parallel, or vectorized by means of a policy. However, so far only the Microsoft compiler supports parallel execution of the STL algorithms, which is why the following performance values are based on the MS compiler. All performance figures are taken from a program compiled for maximum optimization and executed with eight cores. Listing 5 uses a slightly modified variant of the Quicksort algorithm from Listing 4.

The algorithm in Listing 5 includes an execution strategy (Line 12). This lets you control whether the algorithm is called sequentially (`std::execution::seq` in Lines 49 and 69) or in parallel (`std::execution::par` in Lines 58 and 78). Put simply, the program sorts a vector of 100,000,000 random elements (Line 38) between 0 and 100 (Line 39) and measures the performance four times.

The task of generating the equally distributed random numbers is handled by the `getRandom` function in Line 21. The vector is sorted both sequentially (Lines 45 to 52) and in parallel (Lines 54 to 61) with the `quicksort` function and sequentially (Lines 65 to 72) and in parallel (lines 74 to 81) with `std::sort` from the STL. In doing so, `std::sort` applies a hybrid method under the hood that also uses Quicksort. If you are interested in the details of Microsoft’s implementation of `std::sort`, you can check them out on GitHub [5].

In addition to the execution strategy, two other C++17 features are used in this example. On the one hand, numbers can be represented in a readable way (`100'000'000`) with ticks; on the other, the compiler can automatically determine the type of a class template: `std::vector tmpVec = randVec` instead of `std::vector<int> tmpVec = randVec`.

Size Matters

But now back to the crucial question. Given 100,000,000 random numbers in the range of 0 to 100, the Quicksort executed in parallel is impressively fast (Figure 3). It beats the sequential variant by a factor of 5 and is more than twice as fast as the `std::sort` executed in parallel. However, this relation changes significantly if the range from which the random numbers are taken in a uniform distribution grows significantly.

Regardless of whether the range increases to 0 to 10,000 (Figure 4) or 0 to 1,000,000 (Figure 5), the `std::sort` executed in parallel beats the sequentially-executed version by a factor of about 4. But the superior performance of parallel execution dwindles significantly as the range increases. For random numbers between 0 and 1,000,000, parallel execution is slower than sequential execution by a factor of 5.

What can these serious differences in performance be attributed to? We will

not be analyzing the problem exhaustively, but only looking to provide some food for thought.

The main difference between 100,000,000 random numbers between 0 and 100 and between 0 and 1,000,000 is that, in the first case, on average, 1,000,000 equal numbers ($1,000,000/100$) and in the second case, on average, 100 equal numbers ($100,000,000/1,000,000$) end up in the random vector. In particular, this means that the call to `std::partition` [4] is far more efficient in the first case, since it has to swap significantly fewer elements relative to the pivot element. Moreover, this write access requires synchronization between threads.

Now I have to get personal. During my long time as a C++ software developer and now as a C++ trainer, I have become prejudiced: “Don’t think you can beat the performance of STL algorithms in general.” I almost gave up on my fixed opinion after the first run with the random numbers between 0 and 100; however, the further performance tests just reinforced it.

Conclusions

Is Quicksort my favorite algorithm? I am not sure. In any case, it is clear that an understanding of this algorithm is part of the mandatory repertoire of every professional software developer. Annoyingly, the classical implementation of the Quicksort algorithm obscures its elegant structure and thus also complicates its basic understanding. ■■■

Info

- [1] Quicksort: <https://en.wikipedia.org/wiki/Quicksort>
- [2] Tony Hoare: https://en.wikipedia.org/wiki/Tony_Hoare
- [3] Destructive Assignment: [https://en.wikipedia.org/wiki/Assignment_\(computer_science\)](https://en.wikipedia.org/wiki/Assignment_(computer_science))
- [4] `std::partition`: <https://en.cppreference.com/w/cpp/algorithm/partition>
- [5] Microsoft’s STL implementation: <https://github.com/microsoft/STL>

Author

Rainer Grimm is a C++ and Python trainer. His books *The C++ Standard Library*, *Concurrency with Modern C++*, and *C++20* are published by O’Reilly and Leanpub.



MakerSpace

Faster Python apps

Faster Is Better

PyPy and Nuitka improve the performance of Python on a Raspberry Pi. *By Pete Metcalfe*

Python apps on lower-end hardware like Raspberry Pis can be a bit slow, but luckily you have some options that can improve their performance. A number of interesting packages allow you to compile, interpret, or repackage your Python apps. In this article, I highlight two packages that have given me good success:

- PyPy [1] – a replacement to the native Python interpreter that runs more than four times faster
- Nuitka [2] – a native Python utility to compile Python apps to C code

How a Python application performs is based on a lot of different factors, so it's important to know the limitations and how best to work with them. PyPy and Nuitka might not work for all applications, but for many common types of projects they are great fits.

Background

PyPy has been around for a while, and Guido van Rossum, the creator of Python, is popularly said to have stated: "If you want your code

to run faster, you should probably just use PyPy."

The PyPy site has some performance results, and they state that, on average, PyPy will run 4.2 times faster than native Python. In one of my test projects, it ran nine times faster than native Python. PyPy really shines in the area of web services, database connections, and iterative (large loop) applications.

It's very important to note that PyPy only supports a limited number of Python libraries. For Raspberry Pi projects, two of the important libraries that PyPy does not support are *tkinter*, a graphic interface library, and *RPi.GPIO*, the Raspberry Pi general purpose I/O li-

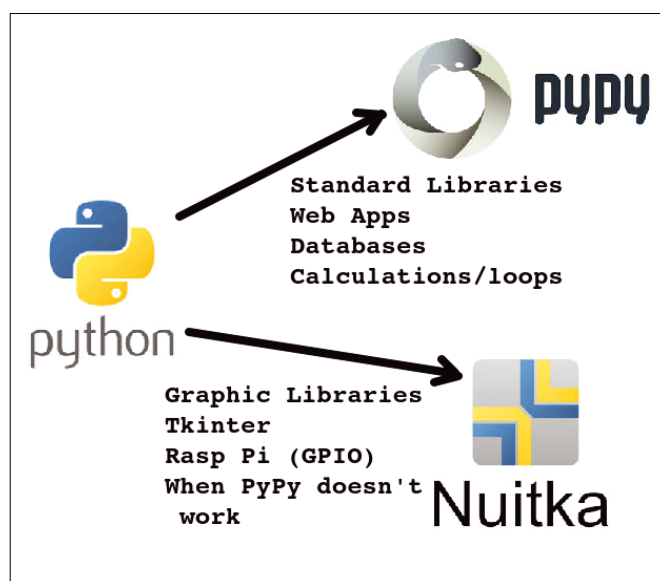


Figure 1: Best fits for PyPy and Nuitka.

Author

You can investigate more neat projects by Pete Metcalfe and his daughters at <https://funprojects.blog>.

brary. To see whether PyPy will work with your application, check the supported libraries [3].

Nuitka will not give the same performance results as PyPy, but it can offer some speed improvements over native Python. Nuitka supports a large majority of the Python libraries, so it's a good fit when you can't use PyPy. Nuitka has the added benefit of creating a compiled application, so you don't need to distribute Python source code.

Figure 1 summarizes the project requirements best served by PyPy and Nuitka.

PyPy – A Faster Python

The improved performance of PyPy is due to its just-in-time compiler, as opposed to the native Python's line-by-line interpreter.

PyPy has a version for Python 2.7 and 3.6 for Linux, macOS, and Windows. The PyPy version for Python 2.7 is preinstalled on the latest Raspbian operating system for the Raspberry Pi. To install the PyPy3 version in Ubuntu, enter:

```
sudo apt update
sudo apt install PyPy3
```

The nice thing about PyPy is that you can use your base Python code as is, and you can do basic testing with PyPy in command-line mode. For example, on the Raspberry Pi, if I want to check whether some basic Python libraries

are loaded and then see if the Pi GPIO library is supported, I would use the `ppypy3` command (Listing 1).

To run your Python application from the command line, substitute `python` with `PyPy3` (or `PyPy3`):

```
$ PyPy3 myapp.py
```

If you are running your Python app as an executable script (i.e., as `chmod +x myapp.py`), then you'll need to change the first line of your script from `#!/usr/bin/python` to `#!/usr/bin/PyPy3`.

PyPy Libraries

By default, only the basic Python libraries are installed with PyPy. To load a specific Python library into PyPy3, you

need to install the Python package installer (`pip`) module before Python packages can be loaded:

```
$ wget https://bootstrap.pypa.io/get-pip.py
$ PyPy3 get-pip.py
$ PyPy3 -m pip install <some-pymodule>
```

I found I was able to load some of the "lighter" modules (e.g., *bottle*, *requests*, *beautifulsoup4*, *pika*, etc.) without any issues. However, some of the "heavier" modules (e.g., *NumPy* and *Matplotlib*) would not load directly. If you're planning on using some of these modules, the recommendation is to run PyPy in an isolated Python environment (`virtualenv`) [4].

Listing 1: Checking PyPy Support

```
$ ppypy3
Python 3.5.3 (7.0.0+dfsg-3, Mar 03 2019, 06:11:22)
[PyPy 7.0.0 with GCC 8.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import os
>>> import time
>>> import RPi.GPIO
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/usr/lib/python3/dist-packages/RPi/GPIO/__init__.py", line 23, in <module>
    from RPi._GPIO import *
ImportError: No module named 'RPi._GPIO'
>>>
```

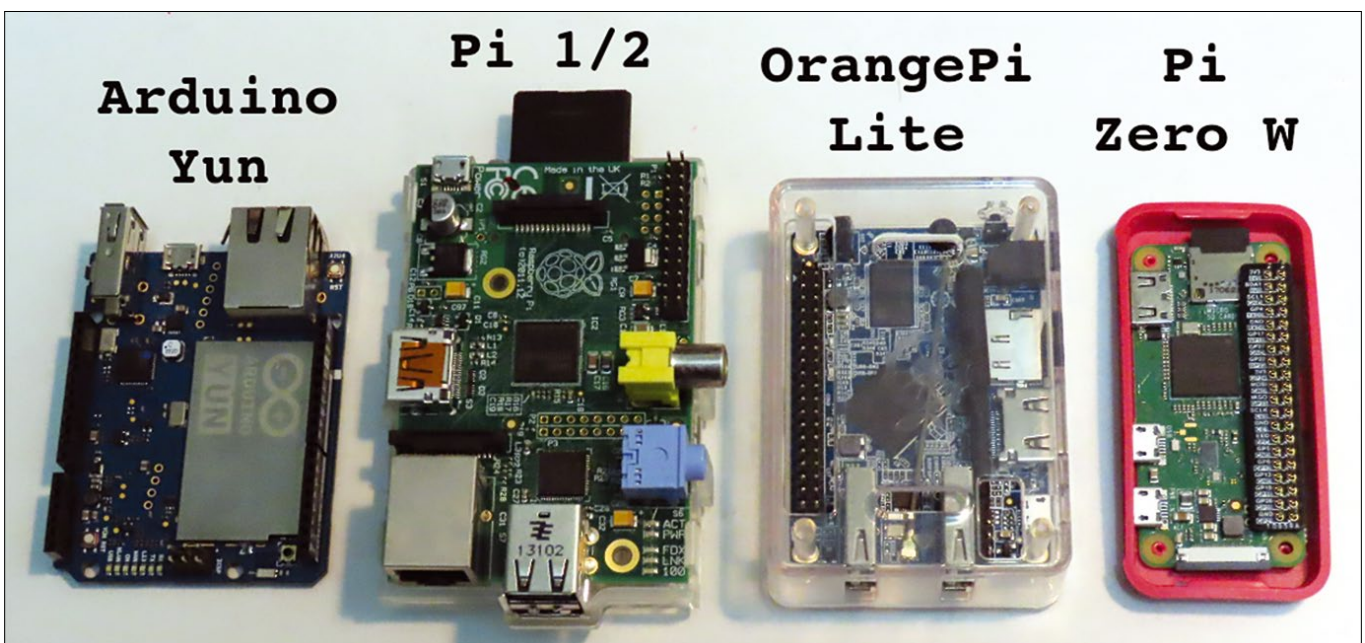


Figure 2: PyPy and Nuitka are especially effective on older hardware.

Nuitka

Nuitka is a Python compiler written in Python, so it should work in all the different versions of Python (i.e., 2.6 through 3.9). Because Nuitka compiles Python code to C code, it has two requirements: A C compiler and Python must be installed on the machine where the application is being compiled.

Nuitka works on Linux, macOS, and Windows. For Linux, the GCC compiler is the default (5.1 or later). The Clang compiler is used on macOS, and the MinGW64 or Visual Studio 2019 compilers can be used for Windows. The Nuitka User Manual [5] discusses situ-

ations in which you might need to use a higher version of Python.

To install Nuitka and compile a test project (e.g., `mytest.py`), simply enter:

```
python -m pip install nuitka
python -m nuitka mytest.py
```

The Nuitka compiled program will be `mytest.bin` in Linux, `mytest` on macOS, and `mytest.exe` in Windows.

I found that Nuitka had no problem with GUI libraries like `tkinter`, `PySimpleGUI`, and `tk_tools`.

Reusing Older Hardware

Python performance improvements with PyPy and Nuitka mean you can reuse some of the older Raspberry Pi and OpenWRT hardware (Figure 2). Unlike Nuitka, PyPy doesn't support the `RPi.GPIO` library, but a simple workaround is to shell

out to the `gpio` utility. An example to setup/write to/read from GPIO pin 7 with PyPy is shown in Listing 2.

I had good success on a sailboat project that used a Raspberry Pi Zero W. With PyPy, I noticed some improved performance with the Python Bottle web framework. The control of the rudder was shelled out to the `gpio` utility.

Conclusion

A number of other options can make Python code run faster (e.g., Pyston and Cython), but I found PyPy and Nuitka to be the best supported. ■■■

Listing 2: PyPy and GPIO

```
$ pypy3
Python 3.5.3 (7.0.0+dfsg-3, Mar 03 2019, 06:11:22)

>>> import os
>>> ret = os.system("gpio mode 7 output")
>>> ret = os.system("gpio write 7 1")
>>> ret = os.system("gpio read 7")
1
```

Info

- [1] PyPy: <https://www.pypy.org>
- [2] Nuitka: <http://nuitka.net/>
- [3] PyPy package compatibility: <http://packages.pypy.org>
- [4] PyPy3 plus virtualenv: <https://techoverflow.net/2019/11/21/how-to-install-pypy3-on-the-raspberry-pi/>
- [5] Nuitka User Manual: <https://nuitka.net/doc/user-manual.html#requirements>

IT Highlights at a Glance

Too busy to wade through press releases and chatty tech news sites? Let us deliver the most relevant news, technical articles, and tool tips – straight to your Inbox.

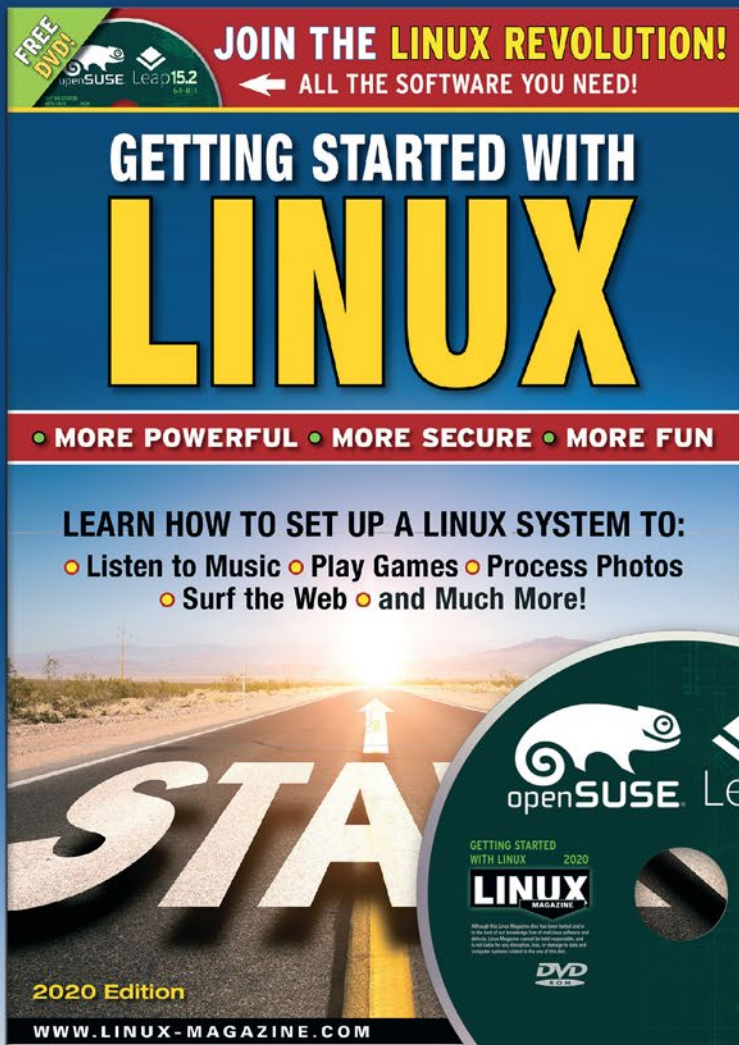
Linux Update • ADMIN Update • ADMIN HPC

Keep your finger on the pulse of the IT industry.

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

Photo by Andrew Neel on Unsplash

Hit the ground running with Linux



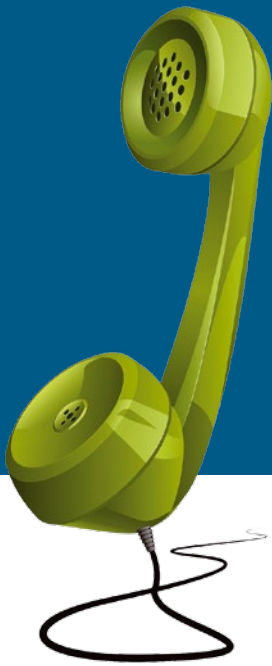
Want your friends and colleagues to make the switch to Linux?

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



MakerSpace

Senior citizen-friendly video telephony system with a Raspberry Pi

Push of a Button

A video telephony system with huge user benefits does not have to be complex. This project starts a phone call with just a single button press and switches channels automatically on a TV. *By Manfred Puckhaber*

The coronavirus pandemic has the world in its grasp. With social distancing and family dispersion, many people are not able to see their grandchildren. Even a cellphone is too complicated for some senior citizens, and they may not want to have anything to do with PCs. One alternative could be an Internet-enabled baby monitor with a return channel, but it won't have video. Besides, grandma and grandpa don't want to be monitored, and you don't want the baby monitor to alert you every time your grandparents walk past the device.

Jitsi and WebRTC

Today, many multimedia applications such as audio or video calls no longer require separate programs such as Skype. The free WebRTC [2] standard enables modern web browsers to send voice, video, and data directly from user to user – and without a central server instance. Jitsi is based on WebRTC and, as an open source solution, does not even require authentication. It is enough to open a Jitsi meeting on a website [3] and send an invitation to the conversation in the form of a URL over email or chat to the desired call partners. This straightforward approach makes Jitsi Meet an ideal candidate for a senior-friendly video chat solution.

With no simple video phone to be found thus far, the question is whether the Raspberry Pi comes to the rescue and whether it can be built without any advanced Raspberry Pi knowledge. In this article, I look at a video telephony solution based on the Jitsi [1] open source video conferencing solution that is useful for even the least computer savvy seniors (see the “Jitsi and WebRTC” box). One button turns everything on and just as easily turns everything off again. Ultimately, the Raspberry Pi needs to be able to do three things: switch the TV to HDMI, send email, and, of course, start the video call.

Hardware and Preparations

The setup of the automated video conferencing system is based on a Raspberry Pi 3B+ with a case and a power supply, on which the latest version of the Raspberry Pi OS “Buster” is installed and Internet access is configured. To be able to control a TV by CEC (Consumer Electronics Control), you also need an HDMI cable that meets at least HDMI standard 1.3 (high speed). Also pay attention to the quality of the power supply. Because the webcam is powered by the Raspberry Pi over USB, the power supply needs to output 2.5A or more. Only testing can determine whether the power source

Author

Mathematician **Manfred Puckhaber** has a wide range of interests and describes himself as a “results-oriented do-it-yourselfer,” who only learned to appreciate the Raspberry Pi in the course of this project.

will do, because even if the label on the adapter claims the right kind of performance, it might not necessarily achieve it in practice.

Ultimately, the shopping list includes a simple push button with two matching connecting cables that will later serve as an on/off switch. For the button, I drilled a hole into the case. If your expectations for the webcam are not too high, all of the components together will set you back something south of \$100. The Raspberry Pi 3 would be overtaxed with very high resolutions, anyway, so a very simple webcam is absolutely fine (Figure 1). In the test, I worked with a no-name camera from my lab treasure trove. Although it has a microphone, it does not work as a USB audio device, so I needed an additional USB sound card with analog audio input. Alternatively, I tested a more modern webcam from Logitech.

After setting up and configuring the video chat solution, the Pi does not need any input devices. For the initial setup, however, connecting an external keyboard and mouse makes work easier. You should also enable SSH access with the `raspi-config` configuration tool. In this way you can connect to the system over the network and, for example, install updates, even when the keyboard has been removed from the Raspberry Pi.

On the software side, the Chromium browser and the important `x11-xserver-utils` component are usually already included by default in the current Raspberry Pi OS. However, if you are building the system on an older Raspbian version, you might still need to install the `chromium-browser` and `x11-xserver-utils` packages:

```
$ sudo apt update && ↵
sudo apt full-upgrade
$ sudo apt install chromium-browser ↵
x11-xserver-utils
```

The first line updates the system before you start the installation.

Dispatching Email

For the system to be able to send email on its own, you don't necessarily have to set up a complex mail server (or more precisely, MTA, Mail Transfer Agent) such as Sendmail or Postfix – that would



Figure 1: The complete hardware (here with an older webcam and separate USB sound card).

be over the top. It is easier to integrate an existing email account with a web mailer (e.g., AOL Mail, Gmail, Outlook.com, Yahoo! Mail). The email client's task is handled by the `msmtp` [4] command-line program. You can import all the necessary components with:

```
$ sudo apt install rdate msmtp ↵
msmtp-mta mailutils
```

For the configuration you then have to create three text files. In the following example, I will be using the services of the German portal Web.de; however, you can use any other email provider that operates an SMTP server.

The server addresses, ports, and access data are usually explained by the services in the depths of their documentation. To set up the email tool for your web mailer, open the text editor with

```
sudo nano /etc/msmtprc
```

copy the content from Listing 1, and paste it into this file.

Of course, you do need to adjust the email address and password to match the service you use. Save the change by

pressing `Ctrl + O` and `Enter` and return to the command line by pressing `Ctrl + X`. Remember this procedure: In the following instructions you will need to edit and save more text files in Nano. Note that the password (here, `<SecretPassword >`) is open and unencrypted in this file. To prevent every user on the system from gaining access to the credentials, use the

```
sudo chmod 600 /etc/msmtprc
```

command to change the file permissions.

In the next step, assign the sender, in this case the root user, the email account

Listing 1: /etc/msmtprc

```
# default values
defaults
auth on
tls on
tls_trust_file /etc/ssl/certs/ca-certificates.crt

# web.de
account <john.doe>@web.de
host smtp.web.de
port 587
from <john.doe>@web.de
user <john.doe>@web.de
password <SecretPassword>

account default: <john.doe>@web.de
aliases /etc/aliases
```

just configured as the sender. To do this, create the `/etc/aliases` file in Nano the same way as before and enter the customized contents:

```
root: <john.doe@web.de>
default: <john.doe@web.de>
```

Finally, to make sure that the system uses `msmtp` as the email program, the line

```
set sendmail="/usr/bin/msmtp -t"
```

in the `/etc/mail.rc` file defines it as the alternative to `sendmail`.

Listing 2: TV Remote Control by CEC

```
$ sudo apt install cec-utils
$ echo 'scan' | cec-client -s -d 1
opening a connection to the CEC adapter...
requesting CEC bus information ...
CEC bus information
=====
device #0: TV
address:      0.0.0.0
active source: no
vendor:      LG
osd string:   TV
CEC version: 1.3a
power status: unknown
language:    eng
[...]
```

TV Remote Control

To avoid your grandparents having to fiddle with the remote control, you now want the system to turn on the TV and switch to the correct HDMI input automatically – and switch back to the normal cable TV tuner after hanging up. For this, the Raspberry Pi and TV communicate with each other by CEC on the HDMI cable. You can install the necessary software from the `cec-utils` package. Then scan for CEC-capable devices (Listing 2).

Typically, the TV advertises itself as *device #0* and the Raspberry Pi as *device #1*. If only the Raspberry Pi appears in the list (*vendor: "Pulse Eight"*), CEC is probably not enabled on the TV.

Depending on the manufacturer, the name for the CEC technology differs: LG refers to it as *SimpLink*, Sony uses the term *BRAVIA Sync*, and Samsung says *Anynet +*. An overview of common manu-

facturer designations can be found on Wikipedia [5].

If it still doesn't work, you need to take a closer look at the HDMI cable. If it is not a high-speed cable, the communication channel to the TV is usually missing.

Autostart All Services

Processes and programs can be started automatically in different ways on the Raspberry Pi. In this example, I will be using a desktop file that starts a number of processes in parallel in the background. The advantage of this procedure is that the LXDE interface starts up completely in the background, and you end up in the graphical user interface after exiting the web browser by pressing the `Ctrl + F4` hotkey.

For this example, enter

```
sudo nano /etc/xdg/autostart/
usr_autostart.desktop
```

to create the autostart file and add the following content:

```
[Desktop Entry]
Type=Application
Name=usr_autostart.sh
Comment=user defined autostart script
NoDisplay=false
Exec=/bin/bash
/home/pi/usr_autostart.sh
```

Listing 3: `usr_autostart.sh`

```
01 #!/bin/bash
02 ### Turn off the mouse pointer after 5 seconds of inactivity.
03 unclutter -idle 5
04 ### Turn off the screen saver and power saving functions.
05 xset -dpms
06 xset s off
07 xset s noblank
08 ### Turn on the TV and switch to HDMI.
09 echo 'on 0' | cec-client -s -d 1
10 echo 'as' | cec-client -s -d 1
11 ### Update date and time from Fritzbox for email.
12 sudo rdate -4nu -s <192.168.178.1>
13 ### Send email with link that Jitsi has started.
14 sudo echo 'https://meet.jit.si/<MeetID>#config.prejoinPageEnabled=false&config.disableAP=true&config.noisyMicDetection=false&config.video.height.ideal=240&config.video.width.ideal=360' | mail -s 'Jitsi Meeting ID is online' <the.grandson@example.com>
15 ### Start Jitsi meeting in Chromium.
16 chromium-browser --noerrdialogs --disable-crash-reporter --kiosk https://meet.jit.si/<MeetID>#config.prejoinPageEnabled=false&config.disableAP=true&config.noisyMicDetection=false&config.video.height.ideal=240&config.video.width.ideal=360
17 exit
```



Figure 2: The push button is installed centrally on the top of the housing.

The system reads the `usr_autostart.desktop` file when booting the graphical interface and then executes the command entered in the `Exec` line. In this case, the system loads the script `/home/pi/usr_autostart.sh`, which has the startup commands.

Again you have to create this file manually by typing

```
nano /home/pi/usr_autostart.sh
```

and adding the content shown in Listing 3. To allow the system to execute the file, adjust the permissions after saving with

```
chmod +x /home/pi/usr_autostart.sh
```

The `usr_autostart.sh` file outsources four tasks:

- Turning off the mouse pointer and the screen saver (lines 2-7).
- Turning on the connected TV and switching to the HDMI input (lines 8-10).
- Sending a notification to a specified email address (lines 11-14).
- Starting the Jitsi meeting in the Chromium browser (line 16).

The `xset` command from the `x11-xserver-utils` package, which can be used to turn off the screen saver, among other things, is part of the default installation. To turn off the mouse pointer, install the `unclutter` package with:

```
sudo apt install unclutter
```

The TV device is controlled by the previously installed `cec-client`. With a pipe (`|`), you can forward the command output with an `echo` to the client. For example,

```
echo 'on 0'
```

wakes up the device with device number 0 from standby, and

```
echo 'as'
echo 'is'
```

switches the HDMI input of the Raspberry Pi to active and inactive.

You can set the `<MeetID>` in lines 14 and 16 freely. However, when choosing the ID, make sure that strangers do not hijack the video call. A random alphanumeric string such as that output by the command

```
xxd -l16 -ps /dev/urandom
```

would be a good choice. Additionally, you have to adjust the IP address of the wireless router in line 12; the same applies to the email address of the recipient of the notification in line 14.

The call to Jitsi in the web browser in line 16 passes a set of parameters:

- `config.prejoinPageEnabled=false` suppresses a prompt for which username to use when joining the meeting. Without a mouse and keyboard, this would be impossible to answer.
- `config.disableAP=true` disables audio processing, mainly to reduce the load on the Raspberry Pi CPU.

- `config.noisyMicDetection=false` prevents a warning message from appearing as soon as the microphone signal becomes noisy.

- `config.video.height.ideal=240 ... =360` reduces the video resolution to save the Raspberry Pi's processor.

You will often read on Internet forums that `exit` at the end of a script that runs all the way through is not good style, but when I tried a script without `exit`, Chromium failed to start; at the end of the day, the instruction does not do any harm.

Keystroke Shutdown

Now you have established a connection, but a routine is still missing that shuts down the system properly. The grandparents should not have to adjust anything on the Raspberry Pi or the TV.

For this purpose, I drilled a hole in the case for a switch that sits between the processor and the USB ports above the board; some space is available there for the connection cables. After I installed a suitable push button in the housing, I connected it with one cable each to GPIO pins 5 and 9 (GND) to close a circuit when the button is pressed (Figure 2). Connected to pin 5, the button also wakes the Raspberry Pi from sleep mode if the small-board computer is connected to a power supply (Figure 3). This function is integrated at the factory and requires no further configuration [6].

If the Raspberry Pi is running, the button will initiate a shutdown process that switches the HDMI input of the TV back to inactive, which, in turn, usually causes the TV to switch to the last selected channel. Furthermore, you want the Raspberry Pi to shut down the operating system gracefully, which includes exiting Chromium and thus the video conference.

To do this, the system needs to monitor GPIO pin 5 continuously, as performed by the `/home/pi/shutdown.py` script (Listing 4). This script needs to be started before the graphical user interface in the existing `/etc/rc.local` file with the shutdown entry,

```
#!/bin/sh -e
[...]
python /home/pi/shutdown.py &
exit 0
```

which should appear in the penultimate line, before the final `exit 0`, as shown. Be sure to include the `&` at the end of the Python program call; otherwise, the system will wait for the script to finish, which would cause the system to freeze.

Finally, you need to make the Python script executable by typing:

```
sudo chmod +x /home/pi/shutdown.py
```

You should also take a look at lines 16 and 17 of Listing 4. According to the

Listing 4: /home/pi/shutdown.py

```
01 # Shutdown script
02 # Waits for LOW at pin 5
03
04 import RPi.GPIO as GPIO
05 import os
06
07 GPIO.setmode(GPIO.BOARD)
08 # GPIO pin 5 as input with signal HIGH
09 GPIO.setup(5, GPIO.IN, pull_up_down=GPIO.PUD_UP)
10
11 try:
12     while True:
13         # Waits for the pin to be connected to GND
14         GPIO.wait_for_edge(5, GPIO.FALLING)
15         # Raspberry Pi signal is turned off at the TV
16         os.system("echo 'is' | cec-client -s -d 1")
17         # os.system("echo 'tx 10:9D:10:00' | cec-client -s -d 1")
18         # Proper shutdown, also of Chromium
19         os.system("shutdown -h now")
20
21 except:
22     GPIO.cleanup()
```

Listing 5: Detected Devices

```
$ aplay -l
List of hardware devices (PLAYBACK)
Card 0: b1 [bcm2835 HDMI 1], Device 0: bcm2835 HDMI 1
[bcm2835 HDMI 1].
Sub devices: 4/4
[...]
Card 1: Headphones [bcm2835 Headphones], Device 0: bcm2835
Headphones [bcm2835 Headphones].
Sub devices: 4/4
[...]
$ arecord -l
List of hardware devices (CAPTURE)
Card 2: C920 [HD Pro Webcam C920], Device 0: USB Audio [USB
Audio].
Sub devices: 1/1
Sub-device #0: subdevice #0
```

specification, the CEC call `echo 'is'` switches the HDMI signal to inactive. In combination with my Sony Bravia TV, however, this command did not do what the doctor ordered. The TV continued to display the picture of the Raspberry Pi system fed in over HDMI. Fortunately, deregistering the signal at the TV with a CEC frame injection by calling `echo 'tx 10:9D:10:00'` worked.

If you experience issues, try one of the two calls by removing the hash symbol `#` in one line and adding it to the other line. Unfortunately, every manufacturer does its own thing when it comes to CEC. Often, CEC only really works reliably with newer TV sets.

Input and Output Devices

To guarantee that the system always outputs audio through the TV and records audio from the microphone built into

the webcam, you need to enter the audio devices permanently on the system. To do this, display the input and output devices detected by Raspberry Pi OS with `aplay -l` and `arecord -l` (Listing 5).

The important parts of this output are the device names that follow the card numbers; in this example, `b1` is the HDMI output, `Headphones` the analog headphone jack on the Raspberry Pi, and `C920` the Logitech webcam used in the setup.

Now create the appropriate settings file by typing

```
sudo nano /etc/asound.conf
```

and pasting the content from Listing 6. Usually, you will only need to adjust the device name of the webcam microphone in line 9. After a restart, the correct devices should now be selected automatically.

The WebRTC test page [7] should then report no errors, and the audio and video test for WebRTC [8] should output image and sound from the webcam (Figure 4).

Picking Up a Phone Call

The receiving side does not need to go through as much trouble because I as-

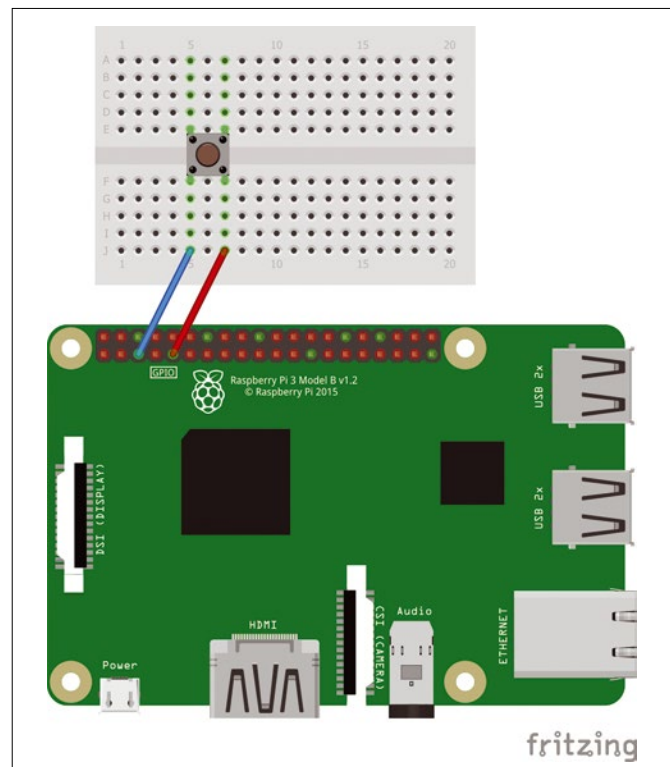


Figure 3: Schematic diagram of the circuit (created with Fritzing).

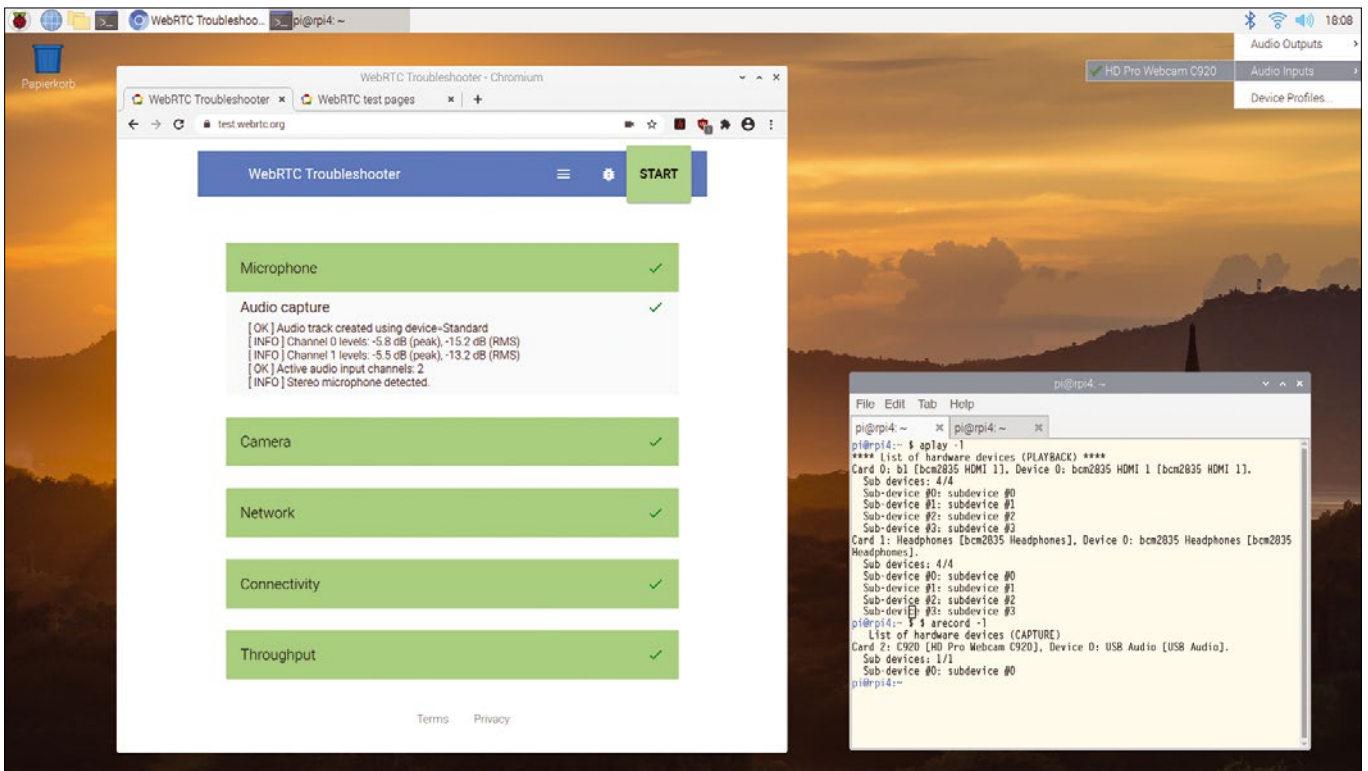


Figure 4: The WebRTC test indicates that everything is OK: To be on the safe side, use the volume slider context menu to check that the system is using the correct input and output devices.

sume that the children and grandchildren have a PC or smartphone. To receive the call from grandma and grandpa, all they have to do is open the link from the email into a web browser, and the grandparents appear on the screen.

Theory and Practice

During the first field test of the full setup, massive interference in the mi-

Listing 6: /etc/asound.conf

```
01 pcm.!default {
02     type asym
03     capture.pcm "mic
04     playback.pcm "speaker
05 }
06 pcm.mic {
07     type plug
08     slave {
09         pcm "hw:<C920>"
10     }
11 }
12 pcm.speaker {
13     type plug
14     slave {
15         pcm "hw:b1"
16     }
17 }
```

crophone signal occurred in Jitsi. In a direct test of the microphone and sound card (on the Raspberry Pi and a Windows PC), the errors could not be reproduced, so the hardware is obviously not responsible.

The explanation was too weak a power supply, which was overtaxed by supplying power to the Raspberry Pi, sound card, and webcam. A better power supply quickly provided a remedy; alternatively, the use of a USB hub with its own power supply is also an option.

It turns out that the third generation of the Raspberry Pi is overtaxed by high-definition video telephony. Transferring the picture and sound without interruption proved to be a difficult task. On the one hand, the latency caused a noticeable delay during the call. On the other, the echo and acoustic feedback effects disturbed the sound because I couldn't shield the webcam's microphone against the TV's speakers.

If you don't want to upgrade to a far more powerful Raspberry Pi 4, you can mitigate both issues by reducing the CPU load during the video call – which is why Jitsi is called with the parameters shown in the scripts. The picture

and sound quality are not perfect, but good enough.

Sources of Error

In the initial phase with the video telephony system, it quite often happened that my grandparents accidentally switched their Sony TV to standby with the remote control while the Raspberry Pi was still running. After that, setting the HDMI input to inactive did not work as desired because the TV did not know what signal source it had been displaying before the Raspberry Pi. The Raspberry Pi's HDMI input remained the last activated input source, and my grandparents needed help because they did not know how to switch to the TV function on the TV. Disaster: the TV stayed black even after it was switched back on.

The attempt to switch the signal source actively to a TV channel by CEC before shutdown did not succeed. In the worst case, the TV still showed a black screen after switching on, but at least it switched back to a TV channel after starting and shutting down video telephony. Some CEC functions are supposed to control the TV's digital tuner remotely; however, in the setup used here, the call from the

Raspberry Pi reported that the command was not recognized.

The Sony's remote control has a button for switching between analog and digital, which can also switch the signal input back to the TV. Fortunately, I could communicate this information by telephone. Happily, my grandparents only make this mistake once in a while and should improve their skills over time. To eliminate this source of error completely, the system would have to be given its own TV set. Smaller devices are available second-hand for just a few dollars.

Conclusions

The term "plug and play" seems a bit like overkill for this video telephony system, but once network access is configured, the Jitsi meeting has been given a name in the code, and the email account is set up, all you really need to do is plug the Raspberry Pi into a suitable CEC-enabled TV.

To make a call, the user then just presses a button and waits for the other party to join the Jitsi session. The first time you do this, you should go through the steps with the mouse and keyboard connected to disable one-off notifications, check the CEC response, and enable the microphone and camera. After that, grandma and grandpa should be able to manage the system.

The original source for the project from Instructables [9] is a few years old and ultimately only provided some inspiration and a starting point. The video telephony system described here configures the audio devices, controls the TV set, and automates everything with a single button.

If I had found a solution to purchase, then this project would never have taken off. But for a newcomer, the potential learning effect is massive, especially if you do not simply copy the code, but also look up what it actually

does. Most importantly, it produces results that have a tremendous utility value. ■■■

Info

- [1] Jitsi: <https://jitsi.org>
- [2] WebRTC: <https://webrtc.org>
- [3] Jitsi Meet: <https://meet.jit.si>
- [4] msmtmp: <https://marlam.de/msmtmp>
- [5] CEC: https://en.wikipedia.org/wiki/Consumer_Electronics_Control
- [6] "Raspberry Pi Raspbian Power on/off GPIO button": <http://www.barryhubbard.com/raspbian-pi/howto-raspbian-pi-raspbian-power-on-off-gpio-button/>
- [7] WebRTC test page: <https://test.webrtc.org>
- [8] Audio and video test for WebRTC: <https://webrtc.github.io/test-pages/src/audio-and-video>
- [9] "Video Calling on Raspberry Pi 3": <https://www.instructables.com/id/Video-Calling-on-Raspberry-Pi-3>

Shop the Shop

shop.linuxnewmedia.com

Missed an issue?

You're in luck.

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

shop.linuxnewmedia.com

GET IT NOW!
SAVE TIME ON DELIVERY WITH OUR ALTERNATIVE PDF EDITIONS



Linux is freedom. Freedom is freedom and Microsoft is Microsoft. Microsoft didn't like Linux and tried to destroy it, and Unix is like Linux. Aren't they cousins? And Microsoft said, "A plague upon your house...." But wait, not exactly. Forget everything you learned or, at least, recalibrate. Microsoft once developed and maintained their own version of Unix. In fact, the Unix variant known as XENIX was, for a time, the most popular Unix edition – and the first Unix to enter the PC market. We'll tell you all about XENIX in this month's Linux Voice, and we'll even show you how to try out a later version of the classic XENIX OS in VirtualBox. Elsewhere in this month's Linux Voice, we show you how to call up your favorite song lyrics in the terminal window, and we dive down into ART, an innovative RAW converter and photo image tool.



Image © Olexandr Moroz, 123RF.com

LINUXVOICE ▶

Doghouse – Weather Forecast	72
<i>Jon "maddog" Hall</i>	
A recent rocket launch has maddog thinking about high-performance computing and accurate weather forecasts.	
ART – Another RawTherapee	73
<i>Karsten Günther</i>	
This alternative RAW converter has the potential to simplify photo editing.	
Remembering XENIX	80
<i>John Knight</i>	
We look at XENIX, Microsoft's lost Unix distro, and show how you can boot up XENIX in a virtual machine.	
FOSSPicks	86
<i>Graham Morrison</i>	
This month Graham looks at MScSim, Ticker, vizex, and more!	
Tutorial – Lyrics-in-terminal	92
<i>Christoph Langner</i>	
Follow the lyrics while you listen to your favorite songs.	



CC-BY-SA WWW.PEPPERTOP.COM



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

MADDOG'S DOGHOUSE

Watching the Mars rover, maddog is delighted to observe that the small helicopter it carries named Ingenuity has many off-the-shelf components and runs under Linux using free and open source software. BY JON “MADDOG” HALL

Open Source Ingenuity

In the middle of the night a friend sent me a message about NASA designing a heavier-than-air drone named Ingenuity to fly on the planet Mars. Even though I had finished my article for this month I knew I had to scrap it and write about Ingenuity, built mostly with off-the-shelf components and Free and Open Source Software (FOSS) which has several real-world (and even Mars-world) ramifications.

The Mars rovers are machines that “have to work.” The cost of designing them, launching them, and managing them is tremendous. Not working after all of that expense and time is something that needs to be avoided at all costs.

Plus the operating environment on Mars is so unlike anything on Earth that extra care has to be taken with every component. The air is very thin, only 1/100 of the density of the Earth. The gravity is only 63 percent of the Earth, and the sunlight (needed to charge the solar panels) is much weaker due to the greater distance from the Sun. Not the least of the issues are dust storms and shadows of overhanging rocks that could block the light to the solar panels and keep them from charging.

Operating temperatures of the computer components need to be from -40 Celsius to +40 Celsius, and the electronics have to be shielded from the radiation that we on Earth are protected from by our atmosphere.

One-way communications take 11 minutes, so feedback from any command given is 22 minutes. Even though the Mars rover moves slowly, you still want to program it to move to where you want it to be rather than give “driving” commands.

All of this has to be designed and tested very carefully and under conditions duplicated as closely as possible on Earth. Therefore the design and test cycles are very long.

One of the previous rovers, Opportunity, was launched on July 7, 2003, and landed on January 25, 2004. It continued to work until June 10, 2018. Of course the design, manufacturing, and testing had to be done many years before its launch.

The processor chosen for Opportunity was a RAD6000, a PowerPC RISC processor that was made for very difficult environments and ran at various frequencies of 2.5, 5, 10, and 20MHz. It also used about 40 Field Programmable Gate Arrays (FPGAs) to do the heavy duty processing for things like image recognition and image compression, since the FPGAs could be turned on, do the processing, and be turned off to save power.

The newest Mars rover, Perseverance, was built along the lines of an earlier, successful rover named Curiosity (which is still active), both of which run VxWorks, which NASA has been working with for a long time.

Perseverance landed on February 18, 2021. However, this newest Mars rover held a surprise strapped to its “belly,” the small helicopter named Ingenuity.

Ingenuity is considered by NASA to be a “technology demonstration.” Even if the “technology demonstration” fails completely, the main aim of the mission, Perseverance, is unaffected.

Ingenuity was “icing on the cake” (or as a friend of mine put it, “lattice crust on the Pi”) because the Ingenuity was built using a lot of off-the-shelf parts that used free and open source software. This dramatically shortened the amount of time it took to design, assemble, and test the system, and also reduced the cost of the final technology.

The main processor board for the Ingenuity CPU could easily be used in a cell phone or a platform like a Raspberry Pi 3. It uses a Qualcomm 801 Snapdragon processor, very low power, but with a GPU and Digital Signal Processors (DSPs). As such, it has more processing power than the processors used on the much larger rover. Many of the other parts for the helicopter were ordered from SparkFun electronics.

It runs an Open-Source Flight Software Framework called “F” (F Prime) which runs on a Raspberry Pi 3 under Linux. You could (with some work and study) conceivably build your own “Ingenuity” and develop it along with NASA.

This is not the first time that NASA has created open source software. NASA was doing it even before the term “open source” was coined through a NASA program called COSMIC, which released many software projects to the public.

One of my favorite programs at the time (1985) was CLIPS [2], a language that is used for writing embedded expert systems. That software has now been spun out of NASA, but it is still being maintained over 25 years later. In the age of AI, sometimes it is interesting to look at the much easier and more flexible concept of “expert systems” to do our work. I may write more about my relationship with CLIPS sometime soon. ■■■

Info

[1] F Prime: <https://github.com/nasa/fprime>

[2] CLIPS: <http://www.clipsrules.net/>

A new fork of RawTherapee offers tools for photo editing

Putting It Together

RAW converters are the first tool to use in editing photos, and new solutions and programs have the potential to make users' lives easier. How does ART stack up?

BY KARSTEN GÜNTHER

ART, which stands for Another RawTherapee, is a RAW converter for photo editing. This young project combines some tools and concepts found in two well-established programs, RawTherapee and darktable, and attempts to offer additional functions, while also providing a user friendly, streamlined program. This article takes a look at the current state of this project and how it's progressing toward those goals.

A Surprising Fork

Three major RAW converters are available for Linux so far. Darktable is the most comprehensive and complex of these programs. RawTherapee is a bit smaller, but scores points with users for its "good-natured" behavior and ease of handling. A third option that appeals to many users is LightZone, a Java application that was formerly proprietary but is now open source.

Darktable offers new releases at short, regular intervals that often extend or completely replace its current collection of 60 or more tools, but RawTherapee's development is proceeding at a slower pace. To date, it includes a number of excellent exposure tools, but it still may not have all the features you're looking for. For example, RawTherapee has nothing like a cloning or stamping tool, which even the far older LightZone has. Similarly, darktable has offered masks for years, but RawTherapee has not. They are, however, under development.

With that in mind, it may be surprising at first that RawTherapee is the basis for a fork, ART. However, if you look at what the ART developer, Alberto Griggio, is doing, the reason for choosing RawTherapee as a foundation quickly becomes apparent: The developer is rebuilding or extending the existing tools (Figure 1).

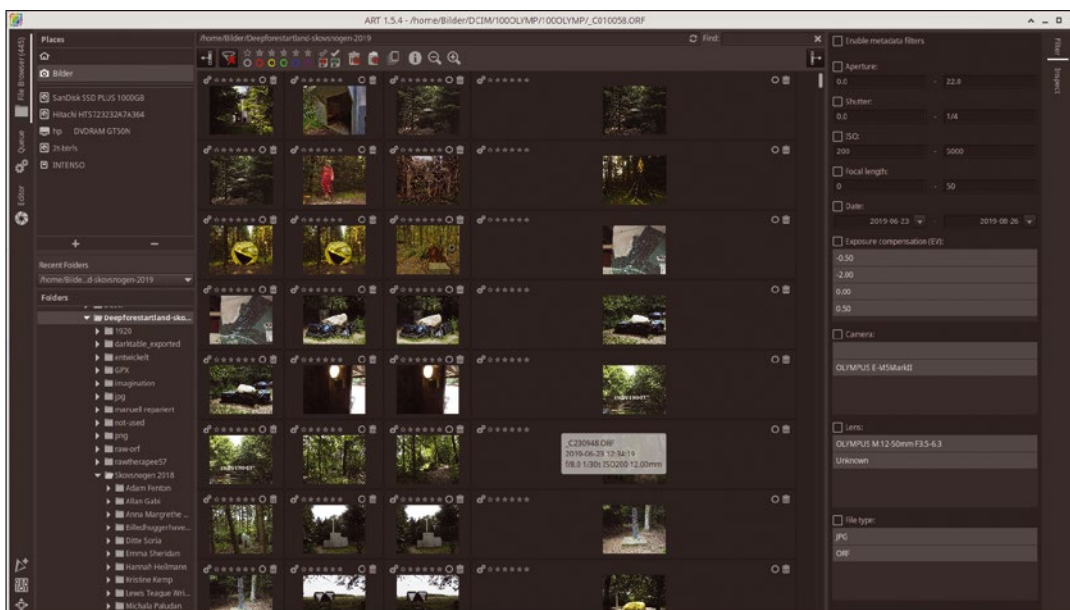


Figure 1: At first glance, ART looks like RawTherapee.

Table 1: Tools in ART 1.5.4	
Tool	Function
Exposure	
<i>Exposure</i>	Preliminary exposure settings
<i>Tone Equalizer</i>	Fine exposure settings
<i>Tone Curves</i>	Gradation curves
<i>Dynamic Range Compression</i>	Dynamic compression
<i>Log Tone Mapping</i>	Filmic-style exposure
Details	
<i>Spot Removal</i>	Cloning/healing tool
<i>Sharpening</i>	Image sharpening methods
<i>Noise Reduction</i>	Noise reduction
<i>Impulse Noise Reduction</i>	Noise reduction
<i>Defringe</i>	Remove color fringes
Color	
<i>White Balance</i>	White balance
<i>Saturation & Vibrance</i>	Saturation
<i>Channel Mixer</i>	Channel mixer
<i>Color Equalizer</i>	Color adjustments
<i>RGB Curves</i>	RGB curves
<i>L*a*b* Adjustment</i>	L*A*B* adjustments
<i>Color Management</i>	Color profiles
Local Editing (new group)	
<i>Color Correction</i>	Local color adjustments
<i>Smoothing</i>	Local blur
<i>Local Contrast</i>	Local contrast
<i>Texture Boost</i>	Boost textures
Special Effects (new group)	
<i>Black-and-White</i>	Monochrome effects
<i>Film Simulation</i>	Simulate film exposure
<i>Soft Light</i>	“Soft light” (layer mode)
<i>Vignette Filter</i>	Edge shading
<i>Graduated Filter</i>	Gray graduated filter
<i>Haze Removal</i>	Remove haze
<i>Film Grain</i>	Simulate film grain
<i>Simulate Film Negative</i>	Simulate film negative
Transform	
<i>Crop</i>	Crop
<i>Resize</i>	Scale
<i>Post-Resize Sharpening</i>	Sharpen after scaling
<i>Lens/Geometry</i>	Lens and perspective correction
RAW	
<i>Sensor with Bayer Matrix</i>	De-mosaicing tool
<i>Raw Gain/White Point</i>	Set black and white point (RAW)
<i>Preprocessing</i>	Filter hot/dead pixels
<i>Dark Frame</i>	Use dark-frame subtraction
<i>Apply Flat Field</i>	Correct effects caused by the lens-camera combination
Metadata	
<i>Exif</i>	View and edit Exif data and maker notes
<i>IPTC</i>	View and edit IPTC data

For example, additional controls for the exposure tool settings allow for more fine-tuning. In addition, ART adopts some other helpful tools, such as automatic perspective correction from darktable, and also adds some brand new tools, such as *Texture Boost*. Although ART v1.5.4 (the latest version at the time of writing) so far only offers a relatively small collection of functional tools, the toolbox already seems to be abundantly filled (see Table 1).

Documentation and Interface

Unfortunately, the ART documentation [1] can be confusing. In some cases, there are references to other sources – tools inherited from RawTherapee are documented by RawPedia [2]. Pixls.us [3] is currently the best place to find more information. In some places you will find detailed tooltips within the program (Figure 2).

The file manager mode behaves almost identically to RawTherapee when selecting images. On the left, you select the directories; on the right, you filter their content via *Filter* or load single images via *Inspect*.

ART supports several very practical approaches to displaying thumbnails (Figure 3). In addition to the embedded image, which sometimes turns out far too dark, you can quickly create preview images with a linear, film-like exposure. This goes well beyond what RawTherapee 5.8 provides and is a massive help when searching for images. Two other variants of the preview (with and without focus points) are only found in the editor mode in typical RawTherapee style and not also in the file manager, as is the case in darktable.

The compatibility between ART and RawTherapee is good so far. Although ART uses its own sidecar files with the *.arp* extension, instead of RawTherapee’s *.pp3*, they can be imported reciprocally. The respective program then ignores the tags for which it does not provide tools but takes all others into account.

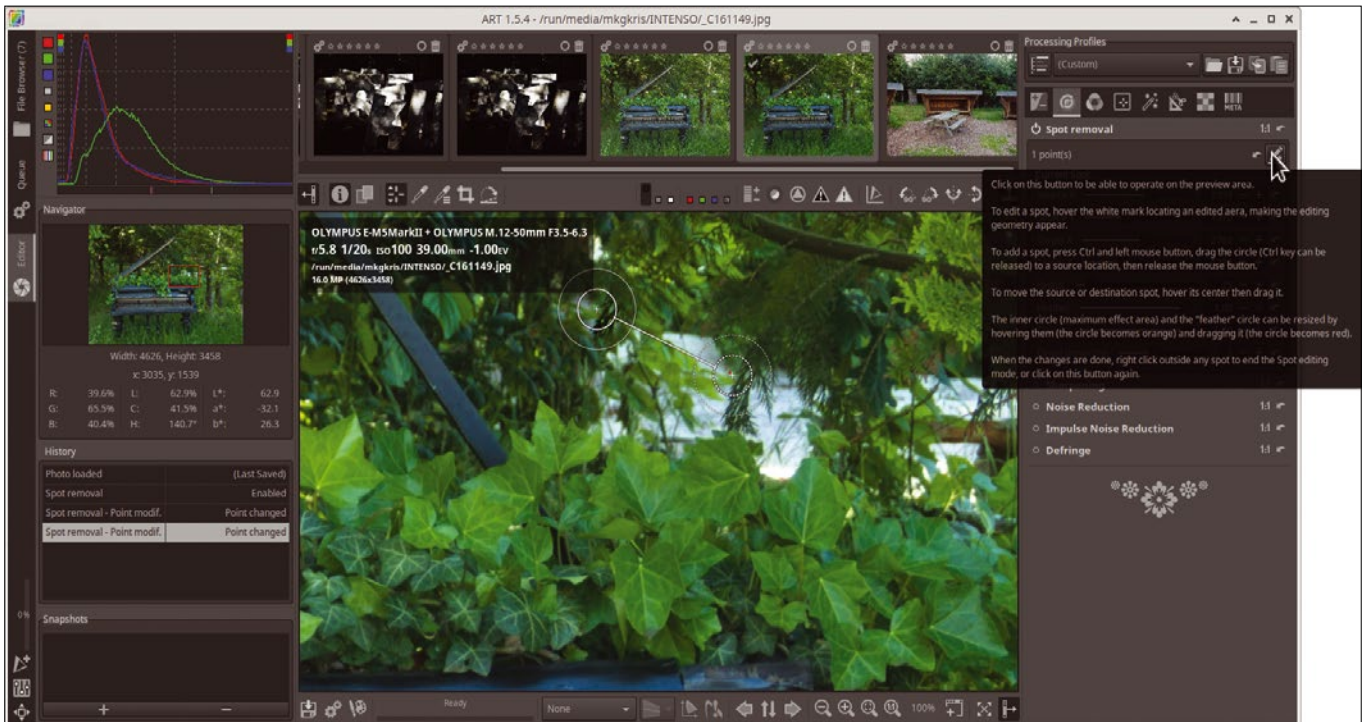


Figure 2: Detailed help texts are available for some (but not all) tools within the program.

Exposure Tools

RawTherapee's basic exposure tool is named *Exposure* and is rightly considered to be a good tool that is easy to use. Its extensive functions come with a correspondingly large number of controls, but these become intuitive quite quickly. Despite this, the ART developer decided to split the tool into three components, which then appear as independent tools in the *Exposure* section (Figure 4).

For many users, splitting the tool up probably does offer some advantages, since three steps can now be performed separately. However, some features were lost in the process. For example, ART lacks *Highlight compression*, which can give you good results when the sky is burned out. ART adopted the other settings, *Highlight reconstruction* and *Exposure compensation* (equivalent to white point), and they work like they do in RawTherapee. The fact that the controls for brightness, contrast, and saturation are missing will probably only bother a small number of users.

In the *Tone Equalizer* section, ART even provides more settings than RawTherapee. With five sliders, different tonal values can be adjusted to the photographer's liking fairly quickly. The five tonal ranges mark different colors that are reflected in the tonal map (Figure 5). The tool displays them if you check the box next to *Show tonal map*.

The third section, *Tone Curves*, now contains a dedicated saturation control in addition to the two tone curves. This is important because the



Figure 3: ART offers extensive options for previewing problematic images. In this example, *Inspect* highlights pixels where clipping occurs.

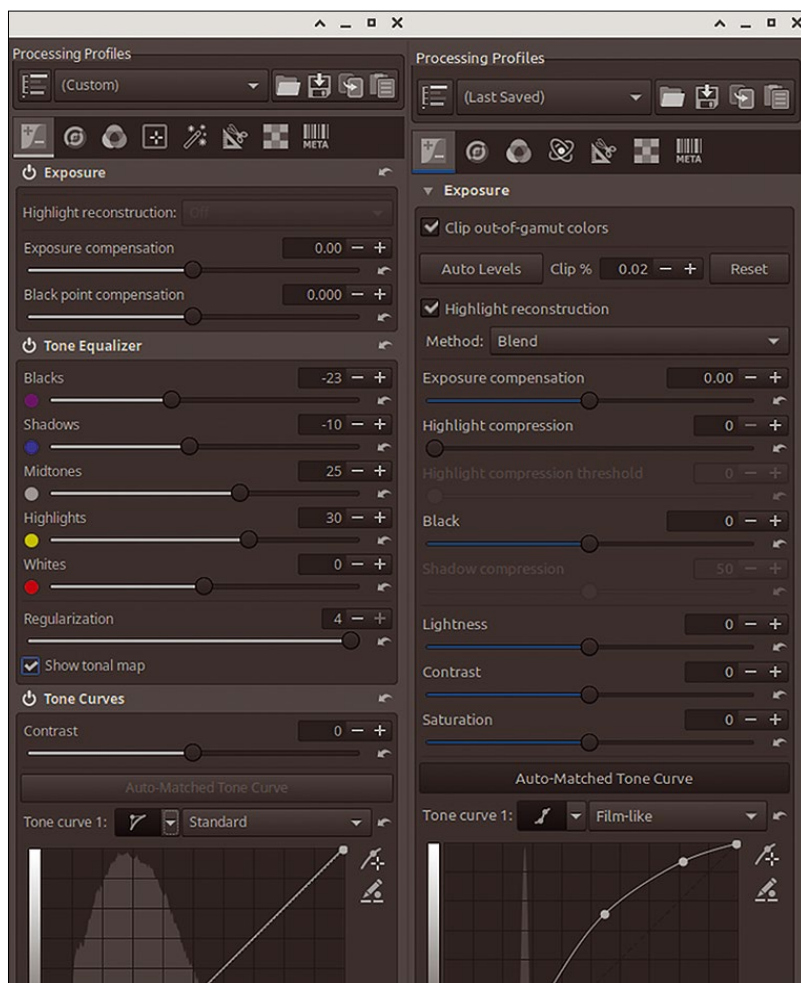


Figure 4: The ART Exposure tool (left) and RawTherapee counterpart (right).

saturation subjectively increases as the brightness decreases.

RawTherapee’s dual tone curves also exist in ART: the first is often used to adjust luminance via the *Luminance*, *Weighted Standard*, or *Saturation and Overlay* modes. The second is used to control the saturation (mode *Filmic*, *Perceptive*, or *Colorful*) as well as for fine control of the contrasts. Another option is to roughly set the exposure compensation with *Tone curve 1* and fine-tune it with *Tone curve 2*.

Be that as it may, exposure is always implicitly linked to saturation – dark colors appear to be more saturated. To influence this, the developer added a darktable-style saturation equalizer to the software. Like in a histogram, the dark tonal values are on the left and the light ones on the right. You can use the curve to reduce or increase the saturation of the colors (Figure 6).

In many places, ART’s practice-oriented development is evident. For example, the *Tone Curves* tool starts with the special *Auto-Matched Tone Curve* option, which presets tonal values based on the embedded preview images (i.e., it takes into account the camera manufacturer’s specifications).

In RawTherapee, you have to manually load the profile of the latest image you processed each time. This step is eliminated with ART, which makes editing many images taken under similar conditions immensely faster and easier. The tools keep the settings, once made; you may

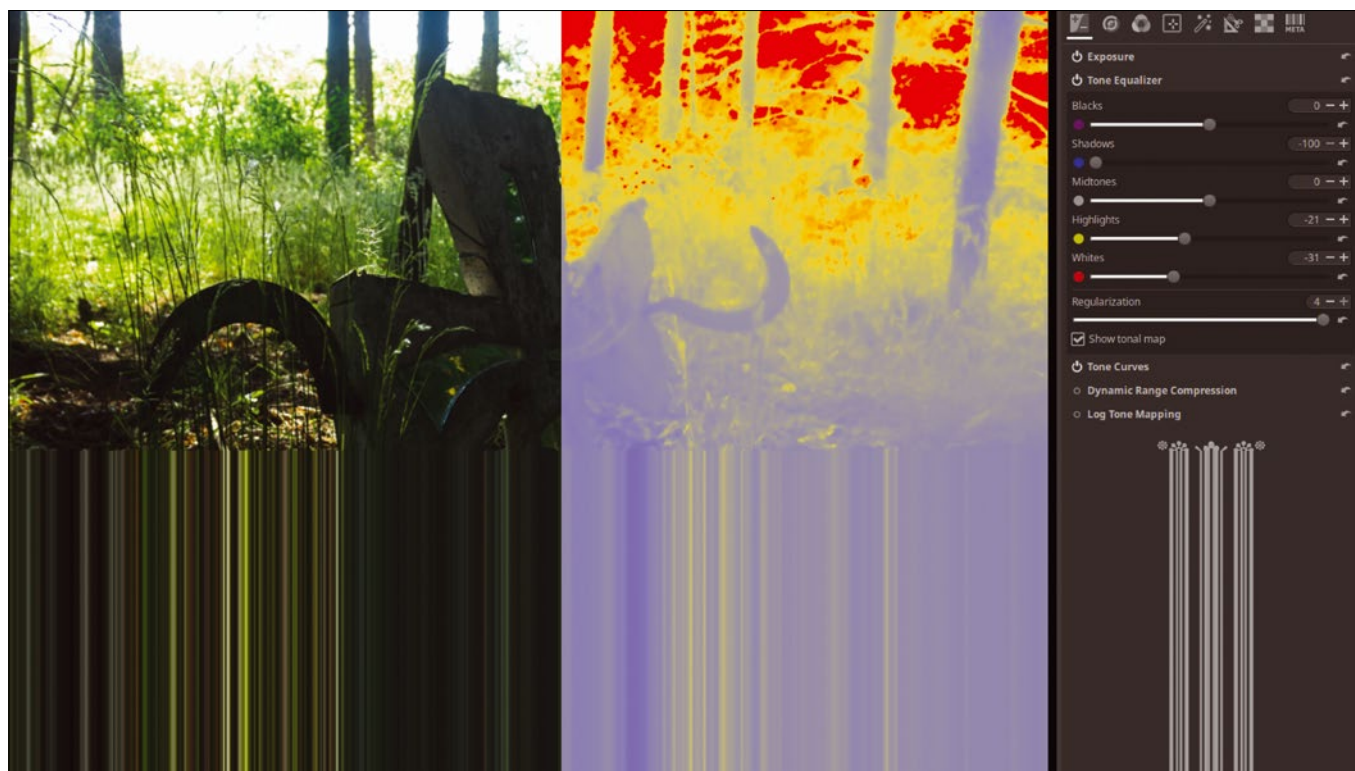


Figure 5: The tonal map shows the tonal range to which individual image sections or their pixels belong. On the left is the original image; on the right, the tonal map has been activated.

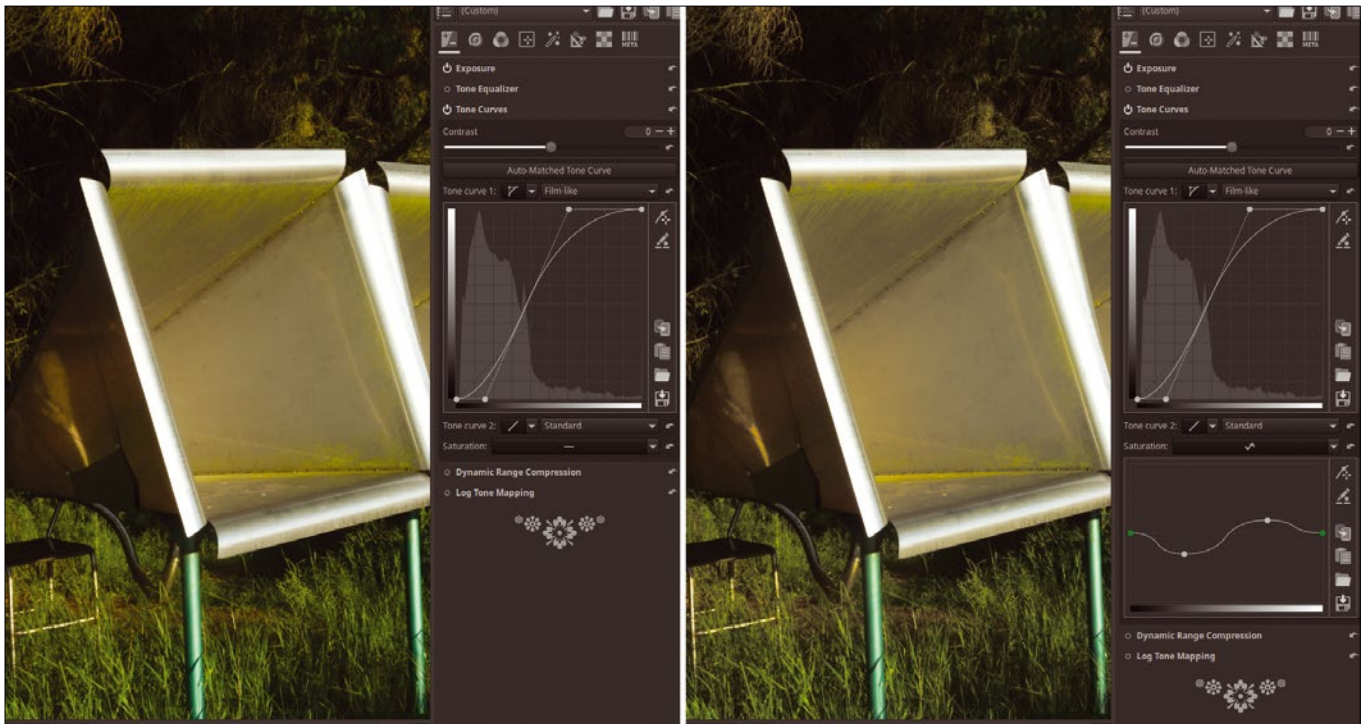


Figure 6: A highlight: brightness-independent saturation control. The left image shows normal saturation; desaturated shadows and saturated highlights are visible on the right.

have to deactivate them manually. All ART tools share this behavior.

ART lacks the practical *Shadows/Highlights* tool from RawTherapee. It supports fast and effective correction at the histogram edges, which can also be done with the other tools, just often not as easily. More of an issue is the fact that the *Tone value correction* tool is missing; it handles local tone mapping and is practically irreplaceable by other tools.

To make up for this, there is an additional tool named *Log Tone Mapping*. It is a stripped-down

variant of darktable's *Filmic* module, but without its S-shaped tone curve and limited to luminance corrections [4].

Other Useful Features

The perspective correction tool, adopted from darktable, is another practical feature. While RawTherapee only allows for simple, manually operated corrections, darktable offers a largely automatic tool. This also performs well in ART (Figure 7). In addition to the manual settings already familiar from RawTherapee, it has automatic

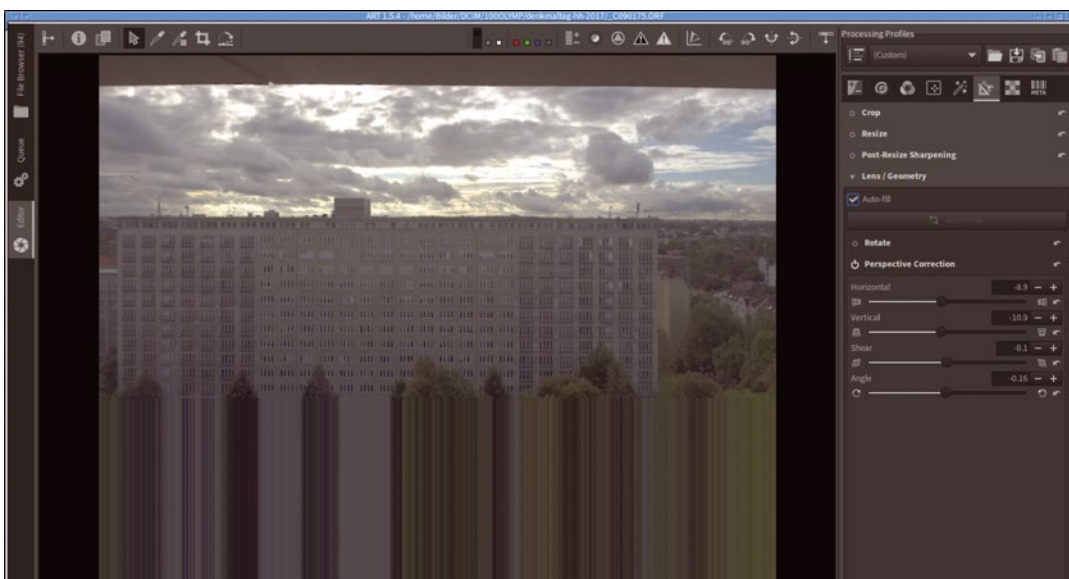


Figure 7: ART adopts automatic perspective corrections from darktable. Automatic corrections are made possible by the three buttons below the sliders.



Figure 8: ART supports a number of effects, for example, a gray gradient filter that allows targeted darkening.

modes that capably perform corrections that are horizontal, vertical, or both.

Another tool worthy of special attention is *Flat Field*. It allows very extensive automatic corrections related to the camera and lens. RawPedia [2] explains the tool in detail.

Some new tools that are not yet available in RawTherapee or darktable have also found their way into ART. Two new tool groups have been added: The *Local Editing* group (functions with local effect) includes the *Color Correction*,

Smoothing, *Local Contrast*, and *Texture Boost* tools. The *Special Effects* group provides monochrome effects, film simulations, gray gradient filters, and more (Figure 8).

In particular, the *Texture Boost* tool, which is a variant of the *Increase Local Contrast* tool and boosts small contours [5], works very well. Both tools work well in combination in ART.

In the *Details* tool group, *Spot Removal* is the outstanding feature. It is used to remove small parts of the image up to a maximum diameter of 200 pixels. In terms of the effect, it is like to Gimp's *Heal* (i.e., cloning with simultaneous brightness correction). To create a correction point, you just click on the image while holding down Ctrl. Initially, two circles appear (Figure 9) with a small cross marking their centers. Move them with the mouse to specify the source for the material to be inserted. You can then define the size of the insertion via the smaller, inner circle. The outer circle determines how soft the transition to the unchanged material is.

On the right side of the panel, you can see the parameters used in the process. You will not normally need to pay any attention to them, because all the actions can be performed directly with the mouse in the image window. You can set some parameters in the panel, for example, the size of the insert, its *Opacity*, and the *Detail protection*.

However, this tool has two disadvantages compared to darktable's variant. Darktable has even replaced it with a wavelet-based version that allows far more granular corrections at different scales

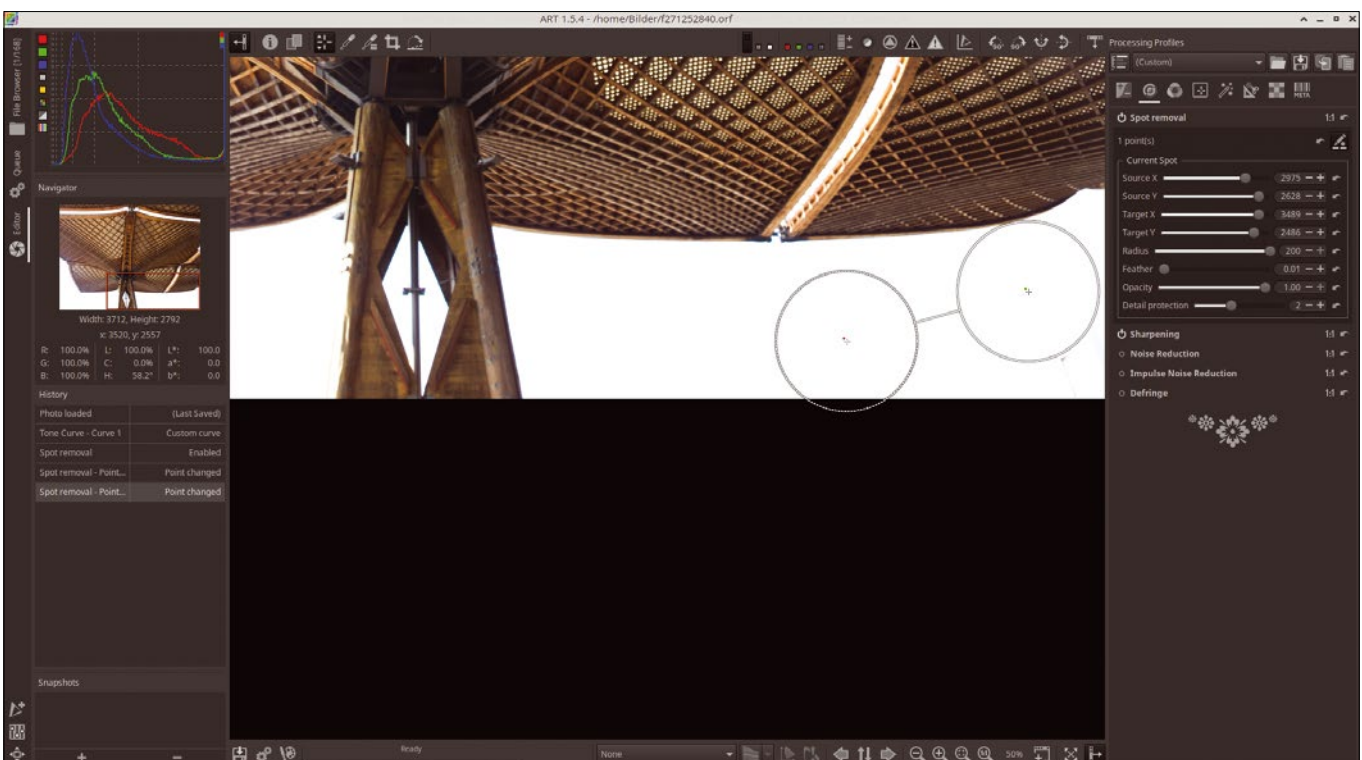


Figure 9: Quick, easy, and often quite effective: the *Spot Removal* tool.

of detail. However, the tool is less suitable for beginners due to its extreme complexity. What is more of an issue is the fact that ART only provides circles for the tool instead of freely definable shapes for the inserts. This often makes complicated structures necessary, which then cause quite a bit of additional work.

ART has seen another exciting development in the form of *User Commands*. These are plugins in the form of plain text files that can be developed from a shell script [6].

Masks

ART touts the ability to make local adjustments via masks as one of its highlights. However for now this ability is pretty limited. In the 1.5.4 version, masks can only be used with the tools from the *Local Editing* group – not very helpful, since masks often prove to be extremely useful, especially for exposure. Also, using the masks can be very awkward, because they are obviously based on the RawTherapee mask code. The latter always links masks to layers, or to be more precise to special filter layers.

The developer needs to think outside the box here. In darktable, and in LightZone, masks are extremely powerful and easy to use. The mask creation model that exists in LightZone also extends to all tools except the RAW tool.

What happens in detail in ART? All layers act independently there and each can be used with a different mask type. However, several levels act cumulatively for combined effects. ART distinguishes be-

tween four mask types. Parametric masks are defined on the basis of image properties (the parameters). The user can draw manual masks with a brush. Geometric masks, a variant of manual masks, let you define the mask using a few support points. Finally, DeltaE masks are a specialty in ART: They use similar colors to define the mask (Figure 10).

Right now, using masks is not fun with either RawTherapee or ART. However, this will probably change in the near future. If you are currently managing without masks, you would do well to keep that up for the time being. A wiki page summarizes the information available for masks [7].

Conclusions

The idea of adding tools from darktable to RawTherapee and decluttering the major RAW converters has been discussed many times. In the past, the idea mostly fell through because it seems very difficult to align the different program structures. Trying to do this with a practice-based approach is definitely a good idea and has real potential. However, the results show that the process is not easy and involves a massive amount of work.

As of this version, ART is usable, but still has a long way to go to reach its desired goals. If you are familiar with darktable or RawTherapee and are looking for perfection, you will certainly achieve your targets faster with the established programs. Having said this, the version of ART is already suitable for experimentation and for simpler tasks, and the latest available version at press time, 1.8.2, was both faster and more useful. Whether or not it will establish itself and continue to build on its strengths is something only the future will tell. ■■■



Figure 10: You can see masks, which are normally invisible, after enabling *Show Mask*.

Info

- [1] ART: <https://bitbucket.org/agriggio/art/wiki/Home>
- [2] RawPedia: <https://rawpedia.rawtherapee.com/>
- [3] Pixls.us: <https://discuss.pixls.us/>
- [4] Log Tone Mapping: <https://discuss.pixls.us/t/ive-finally-tried-art-and-it-is-amazing/20482/14?u=agriggio>
- [5] How local contrast works: <https://www.cambridgeincolour.com/tutorials/local-contrast-enhancement.htm>
- [6] ART user commands: <https://bitbucket.org/agriggio/art/wiki/Usercommands>
- [7] ART masks: <https://bitbucket.org/agriggio/art/wiki/Localediting>



Exploring Microsoft's forgotten Unix distribution Days of XENIX

It might seem surprising today, but before Linux arrived on the scene, Microsoft was a leading player in the Unix world. We look at XENIX, Microsoft's lost Unix distro, and show how you can boot up XENIX in a virtual machine.

BY JOHN KNIGHT

As strange as it may seem, there was a period when Microsoft was advertising Unix as being “the microcomputer operating system of the future,” and the company was even a big player in the Unix business.

Home computing at the time was dominated by 8-bit microcomputers, and Microsoft was planning for the next stage of computing, providing the OS for 16-bit microprocessors. After analyzing all available choices, Microsoft was expecting Unix to be the operating system of choice once home computing became sufficiently powerful.

Development and Launch

Seeking to make its own Unix adaptation, Microsoft bought a license from AT&T for Unix Version 7 in 1979. In August 1980, Microsoft announced their product would be available for 16-bit microcomputers, highlighting the fact that it would be a multiuser, multitasking system.

Microsoft wasn't able to use the name Unix, meaning they had to come up with something new. They settled on XENIX. The press release stressed all of Microsoft's existing “system software” would be ported to the system and XENIX would be compatible with “all existing software written for the Unix OS.”



Figure 1: The IBM 286 AT brought better performance, narrowing the gap with the minicomputers that ruled the server space. © MBlairMartin, CC BY-SA 4.0 [1]

XENIX, like its Version 7 base, started life on the PDP-11 server platform, but in 1980, the Santa Cruz Operation (SCO) was entrusted with the task of porting the OS to other architectures. It was ported to the Zilog Z8000 in 1981, though this platform was not very successful. A Motorola 68000 port soon followed, allowing for more machines, including the Apple Lisa 2.

Rather than selling XENIX directly to customers, Microsoft licensed the XENIX name and code to original equipment manufacturers (OEMs), who in turn sold XENIX to customers. Microsoft bought a controlling stake in SCO, who was tasked with both porting and reselling XENIX under their own name, SCO XENIX.

In 1983, XENIX was ported to the Intel 8086/8088. By now, Microsoft had moved their product away from its Unix Version 7 roots by including enhancements from BSD 4.2, such as vi and the curses library, and adding features like multiple virtual consoles and support for Micnet local area networking.

After its port to the x86 platform, XENIX became the most widely used version of Unix due to the platform's popularity and inexpensive hardware.

The 286 Era

Performance wise, a XENIX machine was now at the very high end of the home computer market but still below the performance of the minicomputers that ruled servers.

However, in 1984 there was significant buzz around IBM's new 286 AT standard (Figure 1), as well as the (incorrect) belief that DOS would soon become obsolete.

While staying backwards compatible with the 8086 and 8088 XT, 286 machines allowed for 16MB of RAM over the XT's maximum 640KB, and they also allowed for larger hard drives – critically important when creating a decent server.

In August 1984, Microsoft released XENIX 286. This system was capable of dual-booting with MS-DOS, included new features from BSD 4.1 and Unix System III, and was capable of supporting

eight dumb terminal connections. (Besides XENIX and DOS, Microsoft at one point developed an in-between product called XEDOS that was never released. See box, “XEDOS: One Product Too Many.”)

The chief strength behind XENIX was that it was a robust multiuser and multitasking environment. This gave XENIX the edge in very specific markets, whose clients found the OS indispensable.

XENIX became the go-to system for early ISPs, who often used it for Internet gateways and mail systems. Before long, ISPs were ditching enormous PDP computers for a simple PC with a large hard drive.

IBM’s \$5,000 AT 286 made for the cheapest Unix workstation around, which became an immediate hit in university computing labs worldwide.

Furthermore, because XENIX allowed inexpensive dumb terminals to connect to a central machine, it was perfect for any setting where users needed to share the same centralized resource.

XENIX became popular with retailers, fast-food outlets, and for scheduling systems used in hotels and restaurants. Internally, Microsoft used XENIX for handling email, and it was used by everyone in the company, right up to Bill Gates.

By the late 1980s, Microsoft had become the biggest Unix company, with XENIX having the largest number of installations of any Unix variant.

The 386 Era

Despite the heady success of XENIX on the 286, as computing shifted towards the 32-bit 386, the Unix market was becoming more competitive, and Microsoft grew nervous about its Unix offering.

Unix parent company AT&T had teamed up with Digital Research Inc. (DRI) for the latest commercial release of Unix System V, and DRI’s compilers were to become the agreed System V standard.

Meanwhile, IBM had unveiled its own Unix variant, PC/IX, which was developed independently of Microsoft and XENIX, effectively snubbing the two companies’ collaboration thus far. Microsoft didn’t want to compete with Unix parent company AT&T or deal with the added competition of corporate giant IBM.

Microsoft decided to essentially pull the plug on XENIX and entered into a partnership instead with IBM to develop OS/2 – an advanced desktop that appeared to be the future of operating systems.

In 1987, Microsoft transferred the ownership of XENIX to SCO, while retaining a minor stake in their company.

Meanwhile SCO busied themselves with the new 386 edition of XENIX, which they updated to conform to the latest System V standard. That same year, SCO shipped Microsoft XENIX System V/386, the first 32-bit operating system for x86 machines.

This new release added support for TCP/IP, SCSI, and a menu-driven business shell. Backwards

XEDOS – One Product Too Many

Microsoft’s strategy in the early 1980s was more than just DOS and XENIX. There was a third product that didn’t see the light of day. Mostly.

Information on XEDOS is extremely hard to find, but Microsoft had a multi-tiered plan for their customers: DOS for basic users, XEDOS for intermediate users, and XENIX for people with serious computing needs.

Byte magazine’s January 1982 editorial [2] discussed Microsoft’s offerings, with MS-DOS as Microsoft’s “single-user, single-tasking operating system.” Customers who wanted multiuser and multitasking support would buy XENIX, Microsoft’s premium product.

However, the magazine went on to say: “In the middle will be XEDOS, a new operating system written in the C language for the 68000, Z-8000, 8086, and LSI-11 processors. XEDOS will contain XENIX-like features and will be essentially a single-user version of XENIX.”

Although XEDOS itself was never released, its features were integrated with MS-DOS 2.0, which had a hierarchical filesystem, basic piping, and I/O redirection.

There were also hidden commands and functions in DOS 2.0 that were undocumented and discontinued from DOS 3.0 onwards. For instance, there was a SWITCHAR control that used Unix switches and forward slashes, and an AVAILDEV function that allowed devices to be addressed as /dev files, as under Unix.

compatibility was maintained with Microsoft XENIX 286, for which a large software base already existed.

End of Days

As the 1980s drew to a close, Microsoft began to slowly rid themselves of Unix – plus the weight of AT&T’s heavy royalty fees – and had essentially left the business by 1989.

Power users hadn’t migrated to XENIX as expected; its sophistication proved unnecessary for most customers. The majority of PC owners stayed with DOS, enhanced with environments like Norton Commander and early versions of Windows.

DOS and Windows provided an enormous software ecosystem that was sufficient for most people, whereas XENIX was complex, expensive, and had software limited to very specific markets.

Although there was talk of replacing DOS with XENIX, IBM’s OS/2 seemed a more profitable alliance, though this partnership didn’t last long.

In late 1989, Microsoft began developing their own NT architecture, based partly on work from OS/2. NT was seen as a chance to start afresh, incorporating what they saw as the best parts of Unix and the best parts of Windows.

The last XENIX server at Microsoft was removed in late 1996 – XENIX boxes at that point had been reduced to simple Internet and email gateways before Microsoft moved entirely to NT-based machines and toolsets.

The XENIX Legacy

Even though Microsoft's time as a Unix company was relatively short, it had a lasting effect on Unix and its derivatives. XENIX was the first successful multiuser system for the PC. In the era before Linux, if you wanted Unix on a PC, you ran XENIX. (BSD was ported to the Intel architecture in the 386 era, but that wasn't until 1991 – around the time Linux was created.) XENIX created the PC Unix scene that gave birth to Linux in the early 1990s.

Innovations like multiple virtual consoles accessed through key combinations and dual-booting between operating systems are features we now take for granted that started with XENIX.

While many in the 1990s and 2000s may have laughed at the idea of Microsoft promoting Unix, Microsoft now has their own Linux-based Azure services and has ported their Edge browser to Linux. They even ran a campaign saying "Microsoft Loves Linux" (not a claim anyone took seriously).

Regardless of their intentions, when it comes to Unix, in many ways Microsoft has come full circle.

Trying XENIX Yourself

While adventurous users may want to try XENIX on an old PC, for this tutorial, we're just going to settle for a virtual machine!

We won't lie, getting XENIX to work as intended has been difficult: Not every virtual machine worked with XENIX, and a lot of disk images didn't work. XENIX is from the bad old days of Unix, and there will be a lot of floppy swapping. Ubuntu this isn't.

In the end we settled for SCO XENIX 386 2.3.4 running on VirtualBox, which should provide the most functionality for the most people.

If you've never used a virtual machine before, you will likely have to enable virtualization in your BIOS. If it's not enabled, VirtualBox will probably only give you the option of creating 32-bit machines and then crash when they're activated.

If you've never activated virtualization in your BIOS, it will probably be under the *Advanced* section, named something like "Virtualization Technology" or "Virtualization Extensions." See your motherboard's manual for more information.

VirtualBox may already be available in your distribution's repositories. If not, head to the VirtualBox's website [3]. The *Downloads* section has installation instructions and packages are provided for many distributions.

Once installed, VirtualBox should be in your system menu or can be launched with the command:

```
$ virtualbox
```

We found our copy of SCO XENIX at Archive.org [4]. If you carefully look through all the download links, there are packages ranging from the base OS, to Microsoft BASIC, and even a games disk.

The most important file is `Microsoft XENIX 386 2.3.4.zip`. Download that file and extract it somewhere convenient, as you will use this folder frequently.

Getting Started: VirtualBox

Open VirtualBox and press the *New* button to create a virtual machine. In the new window, give the project a name and choose its location. Under *Type* choose *Other*, and under *Version* choose *Other/Unknown*. Click *Next*.

Under *Memory size* choose 16MB and click *Next*. In the hard disk section, choose *Create a virtual hard disk now* and click *Create*.

Under *Hard disk file type*, leave it as the default VDI VirtualBox format and click *Next*. For the *Storage on physical hard disk* section choose *Fixed size* and click *Next*.

We've seen different recommendations for your virtual hard disk' size, but 100MB should suffice. Choose that in the next window and click *Create*.

With the New Machine widget finished, you should be back at the main screen. Right-click on your new virtual machine and choose *Settings*. Open the *Storage tab*, where you will remove the CD-ROM attachment and add a floppy drive.

Right-click on the CD icon marked *Empty* and choose *Remove Attachment*. Choose *Remove* in the warning window that appears.

At the bottom of the *Storage Devices* section click on the + icon to add a new storage controller. Choose *182078 (Floppy)*. A floppy drive will now be added to your storage devices.

Click the floppy icon with the + sign, which will open a menu for adding disk images. Click the *Add* button and choose the `N1.IMG` file from the folder where you extracted XENIX and then click *Choose* in the *Floppy Disk Selector* menu.

Back in the Settings menu, click *Ok* and you will return to the main window, ready to start XENIX.

Installing XENIX

Click *Start* and your virtual machine will appear in a new window. After VirtualBox's startup screen, you will be greeted by a bare-bones XENIX boot

screen, with the word *Boot* and a flashing cursor. Advanced users can enter boot switches here, but for now, just press Enter.

After showing basic system information, XENIX will prompt you about System Maintenance Mode. Press Enter and you will be taken to a keyboard selection screen.

Choose your keyboard type, after which you will be taken to the hard disk initialization screen. For your virtual machine, choose option 1. Press Y when asked about overwriting your hard disk. (Don't worry, it's only a virtual disk!)

For most of the next few screens, we'll make use of the default parameters by just entering Q.

Enter Q for the Hard Disk Drive 0 Configuration screen. At the partitioning screen, press 2 to choose the option *Use Entire Disk for XENIX* and press Enter to continue. Now back at the partitioning screen, enter Q.

At the next screen, for covering bad tracks and Scan Disk, enter Q, then press Enter to use the default value for bad tracks.

At the prompt for swap-space allocation, the default allocation should suffice, so press Enter again.

You will be asked if you want a separate /u file-system; choose N.

When asked if you want to make any manual adjustments, enter N.

After your hard disk preparation you will be asked for the serial number and activation key. These are available in the *Readme.txt* file in your XENIX folder.

Finally, you will be prompted to remove the floppy disk before resetting the machine.

Press the Right Ctrl button to regain control of your mouse (you will do this numerous times during this process). At the bottom of the window is a floppy icon. Right-click it and select *Remove disk from virtual drive*. Press any key to reset.

Second Boot

When the machine resets, once again you will be presented with a bare XENIX prompt. Press Enter, and this time the system will run through a new series of checks, and ask you to insert disk B1.

Right-click on the floppy drive and select *Choose a disk file* (Figure 2). Choose *B1.IMG* from the XENIX disk folder and click *Open*.

Press Enter and you will be asked to assign a root password. After doing so there will be a blinking cursor and nothing happening – give the machine a few moments, XENIX is configuring itself.

Next you will be taken through some prompts for setting your time zone. It will ask you to enter an abbreviation, but you can just enter Q to skip it.

The base image of XENIX is now installed. If you would like to stop and explore XENIX, you can enter 1 at the next prompt, or 2 to install more packages.

If you aren't already prompted by the installer to restart, first remove any floppy image that may

still be attached to your virtual disk drive, and then you can reboot with the command:

```
# reboot
```

To shutdown, enter:

```
# shutdown
```

Subsequent Boots

When you reset the system, you will be asked either for a root password for maintenance mode or Ctrl+D to start normally. Given that we haven't added any users, enter the root password.

Once you enter root mode, there will be a message that says `TERM = (ansi)`. Just press Enter and you will be brought back to the terminal prompt.

Assuming you want to add more software, you can always re-enter the installer program we used earlier by entering:

```
# custom
```

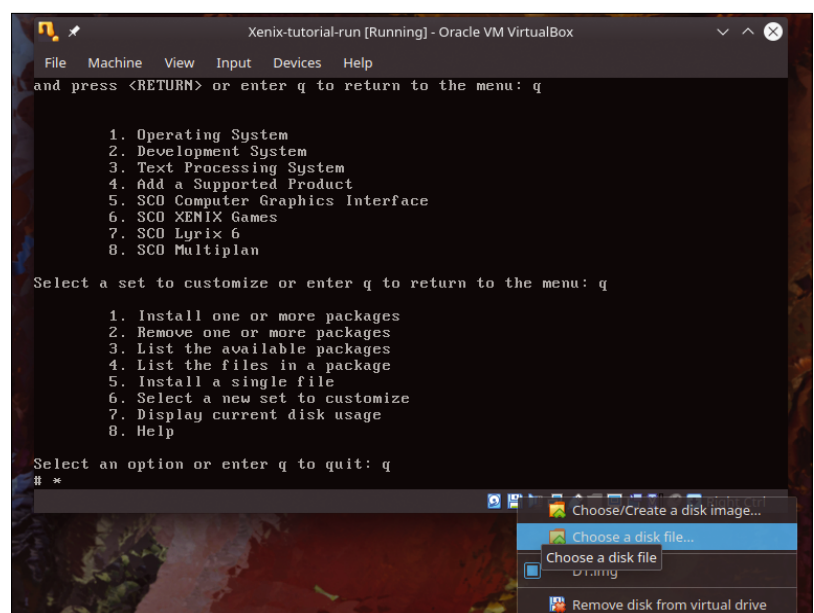
If you have more disk images, you can install applications by following the numbered menu prompts. Under a fresh installation, enter 4 for *Add a Supported Product*.

You will be asked to *Insert distribution volume 1*, meaning insert the first disk of any product recognized and supported by `custom`.

If `custom` recognizes the disk and it works correctly, you will be prompted with a new menu. If you simply enter 1 to install one or more packages, it will usually provide a program listing automatically.

You may be prompted to restart the system after installing new software. Make sure to remove the disk from the virtual drive and press a key.

Figure 2: Choose a disk file is selected. Remember that right Ctrl button for regaining mouse control – you will be changing floppy images a lot!



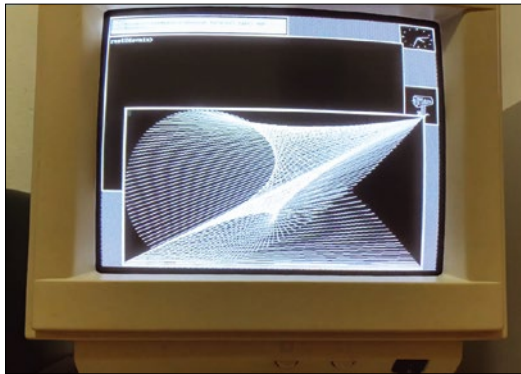


Figure 3: We know it's possible to install a graphical server with access to the right disk images, thanks here to YouTube *Dev Null* [5].

Unfortunately very few disk images that we found online beyond the base OS worked, with most giving some kind of read error. They appear to be working fine for people writing images to actual floppy disks, but didn't work with any virtual machine software we tried (Figure 3).

We were particularly hoping to experiment with any X servers and Microsoft Word. If any users out there can repack the images with virtual machines in mind, it would be greatly appreciated. Fortunately we were able to get a games disk working – see the box “So What About Games?”

Figure 4: Accessing `/usr/games`. XENIX does come with a visual shell, `vsh`, but it's probably more intuitive just to use the command line.

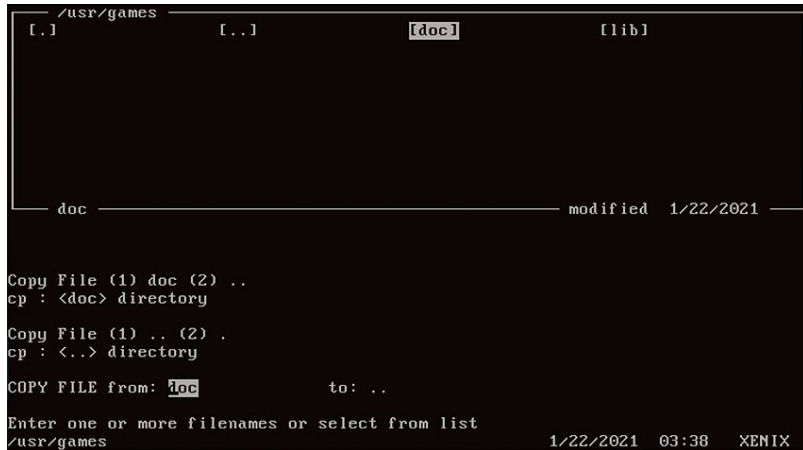


Figure 5: Hack was particularly popular among XENIX users, giving a dungeon crawler experience with ASCII graphics.



For further information on how to use XENIX, full scans of the official product manuals are available on [Archive.org](https://archive.org). ■■■

So What About Games?

Thankfully XENIX users weren't all business, and there are at least a few games available for the system. Although the IMG file from [Archive.org](https://archive.org) wouldn't work, we found a working disk image from YouTube user *MentionedBefore* [6], who provides a link below his XENIX 2.3.1 VirtualBox tutorial.

The disk comes with Worms (not the famous DOS game!), Rogue, Hack, and Trek, plus fortune and mathrec. (And there is a terminal-based version of Tetris somewhere out there!) Once installed, the executables for the games/amusements are found under `/usr/games` (Figure 4).

To start a game (we'll use `hack` for the example, see Figure 5), you can either change into the directory and run the game from there like so:

```
$ cd /usr/games
$ ./hack

Or you can run games by entering the full path:

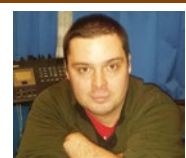
$ /usr/games/hack
```

Info

- [1] Attribution-ShareAlike 4.0 International: <https://creativecommons.org/licenses/by-sa/4.0/>
- [2] “Of IBM, Operating Systems, and Rosetta Stones” by Chris Morgan, *Byte*, vol. 7 no.1, January 1982, pp. 6-10, <https://archive.org/details/byte-magazine-1982-01/page/n9/mode/2up>
- [3] VirtualBox: <https://www.virtualbox.org>
- [4] XENIX 386 ports: <https://archive.org/details/Xenix386Ports>
- [5] Dev Null: <https://www.youtube.com/watch?v=atvTV9xU3Cw>
- [6] Games disk image: https://www.youtube.com/watch?v=n_pq297tcCM

The Author

When he's not punishing his feet double-kick drumming, **John Knight** can be found somewhere near a Commodore 64.



A Webzine for High-Performance Computing Specialists



If you work with high-performance clusters, or if you're ready to expand your skill set with how-to articles, news, and technical reports on HPC technology.

ADMIN
Network & Security

admin-magazine.com/HPC

FOSSPicks

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



Graham is currently considering putting his new collection of Raspberry Pi Picos to use as controllers in a hydraulic flight simulator rig for the ultimate in realism. **BY GRAHAM MORRISON**

Flight simulator

MScSim

The clue to the seriousness of this flight simulator is in its name, “MScSim.” The developer is also responsible for an earlier project called “BScAero,” and we can only assume these projects represent milestones in the developer’s ongoing education from “Bachelor of Science” to “Master of Science.” This is backed up by a mathematically intense document that accompanies MScSim entitled “Flight Dynamics Model for the Real-Time Flight

Simulation” (itself released under a Creative Commons Public Domain license). The paper is difficult to understand without some prior insight into how flying works, but it does presumably illustrate how accurate the flight model in the simulator is. If you want to learn how to fly, this might just be the application.

MScSim isn’t a game, but rather an accurate simulator of the flight dynamics model described in the paper, using real wind tunnel data

for aircraft types including various helicopters, the humble Cessna 172, and the not so humble F-16 and F-35A jet fighters. But the simulator is also approachable and usable by anyone, thanks to its accessible application window and beautiful graphics. By default, for example, the simulator places you in a helicopter in the middle of an airport complex. Also, despite the many detailed panels that can be opened, it is still possible to simply try your hand at flying without the study. Just ramp up on the throttle and see how far you can go. Controls default to using your keyboard, but you can easily switch to a physical controller from the *Preferences* menu where you can freely assign functions to keys and buttons.

For the proper simulator experience, you need to spend some time opening the windows you’re going to find most useful, because there are panels for everything. The amount of data they expose is intimidating, from a set of buttons to control the simulation state to the angle, pitch, and velocity values for how your craft is careering through the air. There are also the traditional flight control widgets (available as a separate open source project), feedback for controller input, and a top-down map of the

airport. Each aircraft shares a standard heads-up display and the same autopilot, which works much like hardware in a real aircraft.

What’s most surprising is that the view outside the Qt-panel festooned cockpit is far better than you might expect from an academic training tool. The package includes a significant chunk of photorealistic scenery taken from aerial imagery of Oahu Island, Hawaii, as well as low resolution scenery of the entire world! It’s not Microsoft Flight Simulator, but neither is it a 150GB download. The entire package is less than 1GB, and the island view still looks amazing as you hover or fly across its mountains and coves. MScSim is obviously not a game, and it is going to be of most use to people learning about flight mechanics, instruments, and control as part of their flight training, but it’s also fun. The other great open source flight simulator project, FlightGear, is more accessible, but there’s a different challenge in trying to take off and land within such a strict flight model.

Additionally, MScSim is obviously far more accessible, both in terms of cost and environmental damage, than flying in real life.

Project Website

<https://github.com/marek-cel/msscim>



1. Save your flight: Rerun and re-analyze a flight for further study. **2. Heads-up display (HUD):** The nine aircraft in MScSim share the same standard HUD. **3. Autopilot:** Learn how to navigate, take off, and land. **4. Data everywhere:** The flight model produces very realistic results that can be studied in the data output. **5. Map:** See where you’re going before you get there. **6. Instruments:** These lovingly crafted instruments are also available as Qt widgets outside the main project. **7. Simulation control:** Stop, start, and pause the simulator. **8. 3D view:** See realistic scenery when you explore the island of Oahu, or get a less-detailed look at the entire planet, all complete with clouds and changing weather systems.

Stock watcher

Ticker

Who would have thought stocks and cryptocurrency would become so exciting for the average Linux user? Rocketing prices and predictable crashes, billionaire influencers announcing their buy-ins, and system-crashing crowdfunding campaigns have been making history, and it can often be difficult to keep up with what's happening. The answer, of course, is to keep a careful eye on those values, but not – we recommend – via a website whose furniture has been designed to tempt you into buying this crazy stock. Which is why you need to install Ticker. Ticker will track all the stocks you care about from the comfort of your command line without adding any superfluous distractions

from the prices themselves and how they're moving. It's also beautifully designed and implemented, with a set of command-line options that are perfectly tuned to the command-line "do one thing well" ethos.

To start the utility, for example, you execute the `ticker` command with the `-w` argument and a comma-delimited list of ticker symbols for the stocks you wish to watch, such as `GME, AAPL, TSLA`. These instantly appear as live stock prices pulled from Yahoo Finance, and you can also add non-real-time data from all the major global exchanges, as well as for most cryptocurrencies. There are further options to show the open price, previous close price, and that day's price range, as well as global totals for the day on the



If you don't want to keep being distracted by stock price alerts, install Ticker and set its refresh interval to 30 minutes.

stocks you've chosen to show. Cleverly, all of this can be put into a configuration file, which can include your personal costs for the gain or loss ranges if you happen to hold stocks or want to play with virtual investments. You can also sort the range and run the request through a proxy to sidestep firewalls. Ticker provides a budding investment voyeur with everything you need to play without the temptation of buying into the game.

Project Website

<https://github.com/achannarasappa/ticker>

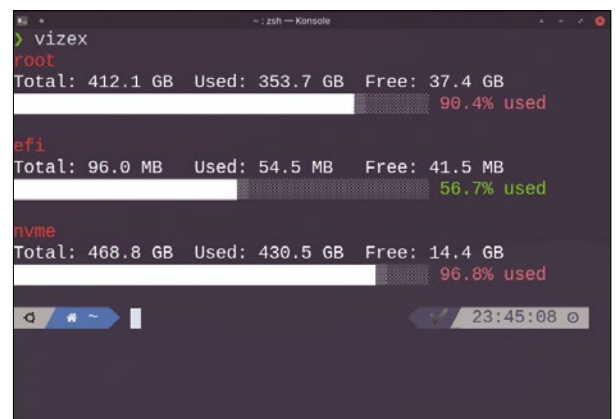
Storage status

vizex

Keeping with the theme of doing one command-line job well is vizex, a tool that simply tells you how much storage space is being used for every partition on your device. It's surprisingly difficult to find a tool that keeps it that simple. The closest we've come to finding a user-friendly equivalent from the standard terminal is the `du` disk usage command with the `-hs` arguments. This provides human readable output (h), such as gigabytes rather than a huge number of bytes, and a summary (s) rather than listing every file and folder on your system – although that can be useful with `grep` if you want individual output. Vizex does away with all of this superfluous nonsense and tells you

exactly what you need to know: How much space am I using, and how much have I got left?

What's great about this little tool is that it uses simple graphics to convey this information with a filled-in bar graph for whatever partitions you have mounted. Each partition is clearly labeled in red and annotated numerically with human-readable values for the capacities involved. Finally, there's either a green, amber, or red percentage value for how much space is left. The color reflects a perceived safety range – with green meaning there's plenty of space and red meaning you'll need to free some space soon. All of this can also be configured with optional arguments for header color, style, text color, and graph color. This can help vizex



Vizex can output storage values for all the mounted partitions on your device or find the capacity for a specific path.

integrate perfectly with your selected command-line color scheme, and you can also choose to exclude partitions (useful for snaps), display extra details such as the filesystem, and even show battery information. This makes vizex a perfect quick access tool when using high-consumption services like virtual machines on a laptop.

Project Website

<https://github.com/bexxmodd/vizex>

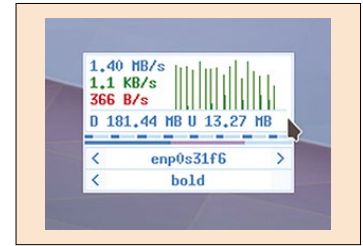
System monitor

Gambal

There are typically three kinds of Linux desktop users. The first group includes those that like to run a single application in full-screen mode, taking up the entire display. The second group contains the growing number of people who use a tiling window manager. These still use every pixel of screen real estate, but they allow for more than one application to run at once by shuffling their windows around automatically. The third group prefers to see the desktop background occasionally, often running one or more applications as floating windows they can resize and relocate at will. All of these groups, but particularly the first two, will find it difficult to accommodate small windows, precluding the use of

many of the best floating system monitors. Except this one.

Gambal is a tiny and unobtrusive floating window that constantly displays an overview of network, CPU, and memory usage. It does this without causing any other windows to resize or move. Thanks to its variable opacity, it can easily sit atop a full-screen application or tiling window manager without obscuring the information beneath it. Most of the space in its tiny window is taken up by the histogram of incoming and outgoing network data, which is also shown as numbers on the left. Beneath this, there are upload and download totals, a meter per core to show CPU usage, and a flat bar to show used memory, swap memory, and free memory. When you roll your cursor over the window,



Gambal takes very few system resources and very little screen space, making it an excellent monitoring utility.

two further options appear. One lets you select which network device to monitor while the other switches between bold and normal font rendering. Apart from using the mouse scroll wheel to change the opacity value of the window, there are no other options. You can't even change the window size. But that's also one of the best things about this little tool, because it's one less distraction to worry about.

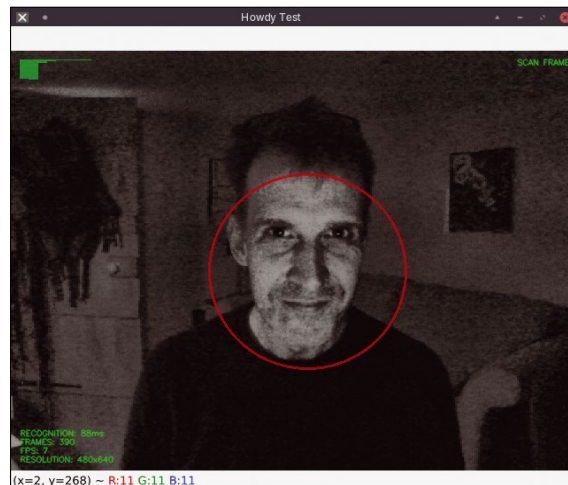
Project Website

<https://github.com/ashtum/gambal>

Face recognition

Howdy

We've been using passwords for a long time, and they're obviously a vitally important way to control who has access to your devices. But they're not the only way of authenticating access, and mobile devices in particular have been trailblazing new and more convenient ways to marshal system security. One of the best of these is using the front-facing camera for facial recognition, enabling access and payments without any user interaction, and this is what Howdy is – a facial recognition authentication system that can be introduced into your security layer. But first, a caveat: This is experimental software and cannot be relied upon in the same way you rely on your password or SSH key. To be useful, Howdy integrates with Linux



Interactions with Howdy are mostly transparent, but there is an excellent test mode that will show you the camera view and processing overhead.

Pluggable Authentication Modules (PAM) to replace your password for tasks like login and sudo, effectively granting root access to your system to anyone who looks like you. Or holds a photo of you up to the camera.

Fortunately, the complexity of integrating Howdy into your system is handled by the installer,

and there are packages for many of the most common Linux distributions. After which you need to edit a configuration file to at least set the path to the digital video device (webcam) you wish to use. With that done, Howdy needs to be run from the command line to first add a face model for a user. This can be done with the `howdy add` command, and you'll need to briefly look at your webcam while your face, or whatever the camera is looking at, is algorithmized (our word). And that's all there is to it. Wherever you attempt to perform an authentication action, your camera will briefly flicker into life while Howdy attempts to recognize you. Success means no further interaction is necessary, while failure will simply drop you back at whatever authentication method you used before. It already works remarkably well. While it might not save time typing a password, it definitely helps you to keep your concentration without having to context shift when remembering a password.

Project Website

<https://github.com/Boltgolt/howdy>

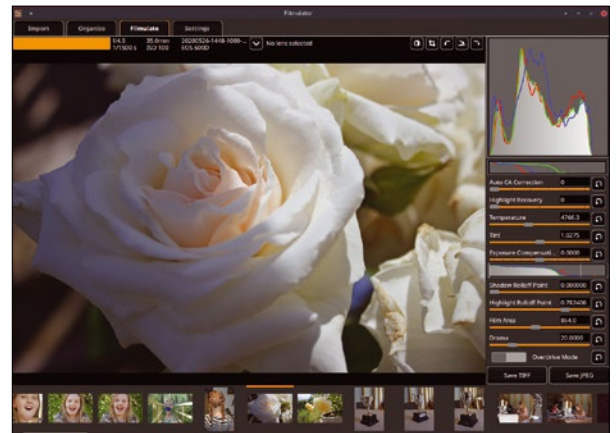
Photo processing

Filmulator

It doesn't seem so long ago that if you were into photography, you needed to process the RAW image files from your camera using a different operating system, or perhaps Wine. But we now have a glut of command-line and desktop options, including digiKam and the all-conquering Darktable and RawTherapee. The last two in particular are more than capable of professional results, and there are many professional photographers who have successfully forgone Adobe's Creative Cloud subscription with its access to Lightroom in preference for either of these. But we also have Filmulator, a RAW photo-processing application we last looked at several years ago, and it has grown to fill a significant niche – photo processing without requiring

encyclopedic knowledge of hundreds of parameters.

Filmulator aims for simplicity rather than maximum control. After you've imported your photo collection, you typically have access to around 12 parameters for fine-tuning your images. This is a huge contrast to the many modules you can access in Darktable, for instance, each with their own sets of parameters and presets. Like Darktable, though, the view is dominated by the image preview and the settings pane on the right, complete with a histogram showing the exposure levels for red, green, and blue hues. From this point you can only change vital image components such as exposure compensation, highlights, and temperature. Filmulator will also carefully adjust parameters



If you enjoy processing RAW images files, but not the hundreds of parameters that typically need to be tweaked, try Filmulator.

in the background as it attempts to simulate film development. In particular, the dynamic range is reduced a little to bring out more details, bright regions are oversaturated, and their surroundings are made a little darker to help local contrast. It can feel like the best kind of HDR, but only from a single source. The results are excellent and much easier to produce than with other applications. It's an ideal application to recommend to someone who doesn't want to read a book on theory to get good results.

Project Website

<https://filmulator.org/>

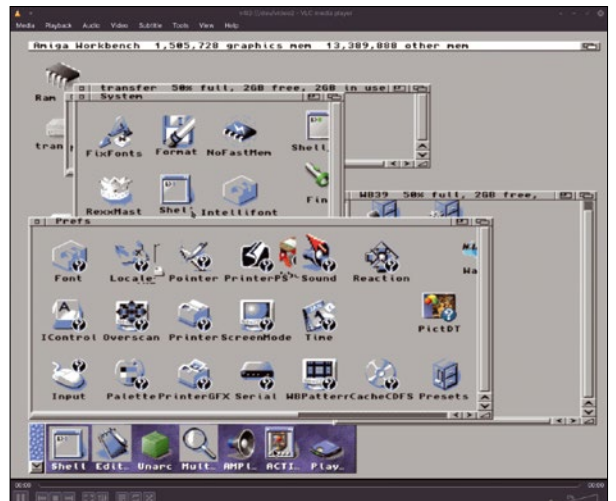
Remote control

Barrier

Many years ago, when computers were big and heavy and their screens were bigger and heavier, you might have had more than one machine, and if you did, you really wanted to avoid having more than one screen. The best solution for this was a KVM switch – not to be confused with the now more commonly used acronym for kernel virtual machines. A KVM switch connected a single keyboard, a single video device, and a single mouse to one or more machines, with a physical or software-controlled switch that was used to flip between controlling any of the connected machines. It saved space and time. But there was another clever solution that needed no clunky hardware, and that was a piece of

software called Synergy. With Synergy, one machine would act as a server, and other machines would act as its clients. The server was physically connected to the keyboard and mouse, and Synergy would handle the protocol that would send keyboard and mouse input to numerous connected clients. The client software would reinterpret these controls on the local machine, with video handled with something like VNC or a separate monitor input.

The Synergy project became Symless, which is now a proprietary product offering supported versions with the same functionality. But open source packages of the older version are still available, and Synergy 1.x has been forked into a project called Barrier that continues to be developed and aims to solve many of the issues that were still affecting the older project. While Barrier does break compatibility with the older version, it does exactly what the



You can't see it, but this is an Amiga desktop (running Workbench) being controlled by a mouse and keyboard on a Linux machine and viewed via an HDMI-to-USB adapter.

old Synergy did, letting clients be controlled by the keyboard and mouse on a server, and it works across operating systems. Other operating systems include Windows and macOS and even unofficially the ancient Amiga, so it's ideal for controlling other machines from your Linux or OpenBSD box, or even vice versa, and it works perfectly.

Project Website

<https://github.com/debauchee/barrier>

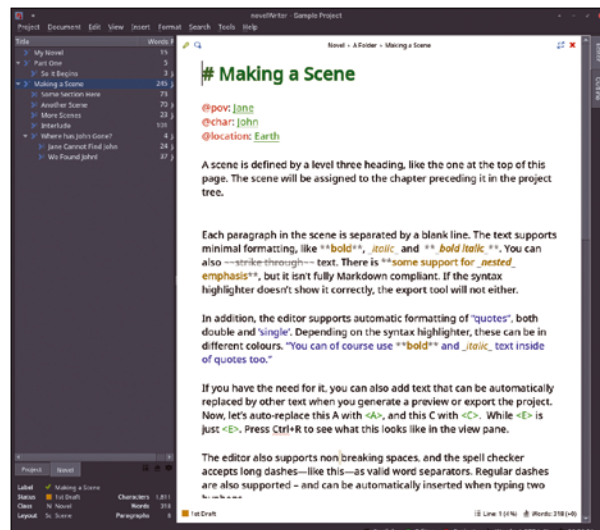
Author IDE

novelWriter

It's true that many people want to write a novel. And that many people have written novels, and that many more novels are started but never finished. None of this is a bad thing. Even published novels seldom make their authors any real money. For most people, the whole enterprise of novel writing is best described as a labor of love. And there's nothing wrong with that either. Planning and writing a novel is a fun and a valuable experience that can vastly improve your writing ability and deepen your appreciation of the written word. One thing that can help hugely with motivation and with the writing process itself is to use an application that can assist you with the more mundane aspects of novel writing – managing large numbers of notes on character and location descriptions, story ideas, narrative arcs, and the plethora of other junk that follows a brain dump.

Alongside Emacs modes and Vim plugins, there are a handful of applications that try to help you manage the more arduous aspects of writing a novel so you can focus on the fun stuff.

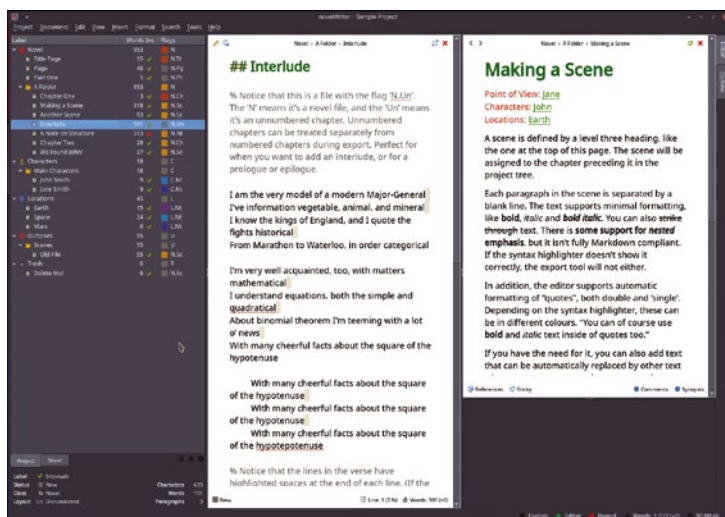
Scrivener is the best known, but as we mentioned awhile back, the Linux build has not been updated in some time and the application has become proprietary. Which is why novelWriter has become such an important project. It's an open source, cross-platform “integrated development environment” for writers that can make the difference between whether writing becomes part of your daily process or ends up being just another failed commitment. At its core, as with Scrivener and even the macOS darling Ulysses, novelWriter is a glorified file manager. Instead of files and folders, you have documents written in Markdown split into chapters, characters, locations, and notes. These are shown in a project overview pane on the left, which can be tabbed between this *Project* view and a *Novel* view. The *Novel* view is an expandable structural overview of headings within your book, ordered across Markdown documents using a global heading and subheading syntax. Level one headings signify a book title, level two the start of a new chapter, and level three the beginning



At the heart of the application is an excellent text editor that even includes a distraction-free editing mode.

of a scene. Level four headings can be used internally to separate sections. This is a brilliant feature, because it naturally maps out your writing and assists with organization. If you feel something is in the wrong place, you simply edit the heading Markdown, and you can even do this using the integrated Markdown editor.

Like many Markdown editors, an optional pane on the right can be opened to show a preview of the formatted text or another document entirely, helpfully colored to show headings and links to which characters and locations are involved. You can even enable a distraction-free mode so you can focus on your writing, or the structure of your documentation, and there are flexible color schemes to match your preferred working environment. This meta information is entered into the documentation using @ symbols to link to and reference tags in other documents that might expand on a character or location, regardless of whether you use them in the final text. You can also add synopsis: to write an overview that appears in the *Novel* view and % for un-rendered comments. It's a clean and simple system that works well, especially in the early stages of writing, and the entire project can be rendered as a fully functional set of HTML files, which can easily be edited into something like an ePub. The entire application works brilliantly and can genuinely help with both the organization of a novel and the motivation required to get you writing.



novelWriter is a great tool for organizing your thoughts, locations, and character descriptions while also writing your story.

Project Website

<https://github.com/vkbo/novelWriter>

Real-time strategy

Augustus

As more and more games are released, and our backlogs take longer and longer to be played, it's become clear that classic games will remain classics, regardless of their era. These are games that can't be improved even with the advanced hardware many of us have at our disposal. Examples of these classic titles include M.U.L.E., Syndicate, Monkey Island, Day of the Tentacle, Frontier: Elite II, System Shock 2, and Caesar III. The last title is an example of isometric city building and resource management that's yet to be bested in the 20 years since its release. The game itself tracks the heyday of the Roman Empire starting in 340 BC and ending four or five hundred years later, depending on your skill. You start

off creating housing plots and by providing utilities for your citizens. There are all types of products, educational establishments, religions, and health services, tied into your road and commerce networks, taxes, and wages. Success means building a metropolis that will stand the test of time. Or at least until the hordes of Goths ravage the empire.

The trouble with games like these, of course, is playing them on modern hardware. Fortunately, there are two brilliant projects that will help you play Caesar III on modern hardware, and even your phone. You will still need access to the original data files, but fortunately the game is still available on GOG and the executable will extract simply using Wine. The first modern interpretation is



There aren't many real-time strategy games that will let you recreate the Forum and grow wine grapes!

Julius, which is a faithful recreation of the original game that will even load and save your original games. But the second is Augustus, an incompatible fork of Julius with long-wanted features such as roadblocks, zoom controls, warehouse storage, and a global labor pool. We've already wasted hours playing it, proving that these old classics are just as addictive today as they were a generation ago, and you can now play them anywhere.

Project Website

<https://github.com/Keriew/augustus>

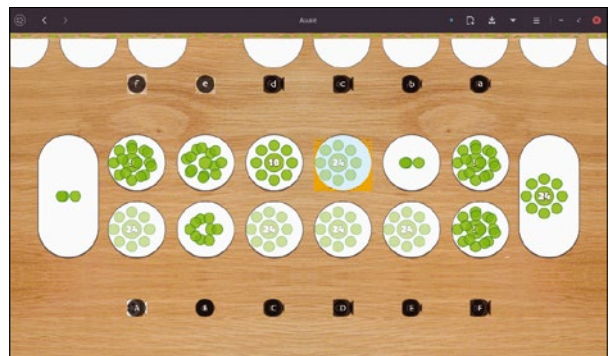
Mancala game

Aualé

Oware is a wonderfully deep and satisfying board game that contains more strategy than its simplistic layout and rules might imply, and it is a member of the much larger group of mancala – also called count-and-capture – games. The game itself is based around a horizontal board with two parallel rows of six indented pockets and longer vertical indented stores (called the mancala) to the left and right of these rows. Each row is for one of the two players, and into each of the pockets are placed seven small seeds. The victor is the player who captures the most seeds from these pockets and places them in their store, and the end game is triggered when 24 or more seeds have been captured.

Aualé is a graphical interface for playing Oware, which has three different phases of gameplay, with the first being collecting and redistributing the seeds from one of the pockets. This is all handled automatically by the games engine after you select a location.

The next phase is where you need to capture the seeds by redistributing seeds into a pocket with either two or three seeds. When this happens, you capture the entire contents of that pocket and place those seeds into your store. It's a little like checkers with fewer squares and more pieces, and while it only takes a moment to learn, there's plenty of subtle strategy to grapple with. Aualé helps with this by offering four computer playing levels, from



Aualé is a graphical interface for playing Oware, a member of the large mancala family of games.

easy to expert, and you can play as either side or with a human opponent. There's also an undo and redo option, as well as the ability to save a game and come back to it later. The documentation is excellent, with a guide on how to play that includes animations and even a strategy and tactics page that can help you improve your skill.

Project Website

<https://auale.joansala.com/>

Automatically view song lyrics as you listen

Waxing Lyrical

Whether you listen to music on Spotify or a classic audio player like Rhythmbox, Lollypop, or Audacious, a tool named Lyrics-in-terminal will let you read the lyrics for the track you are currently playing. **BY CHRISTOPH LANGNER**

In today's very challenging times, there are many artists who write expressive, interesting, and relevant lyrics. Whether it's hip-hop, pop, or folk music, you'll find many artists in recent years who definitely have something to say.

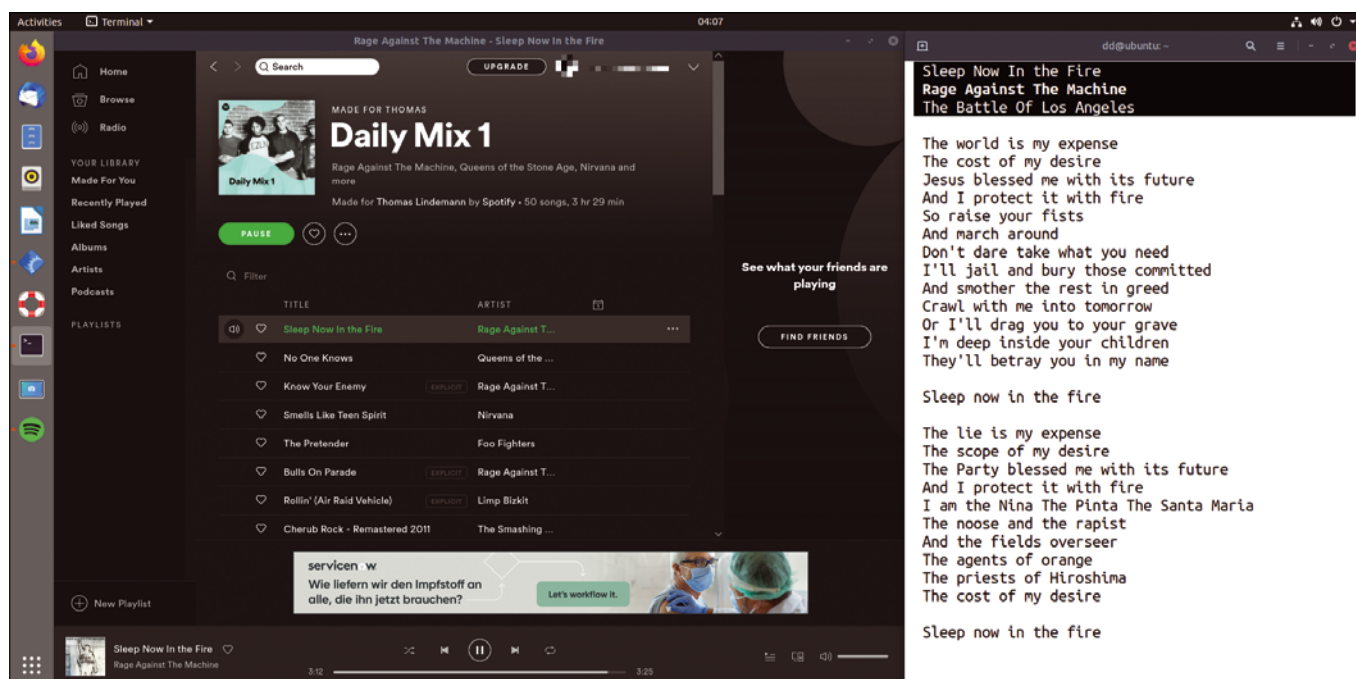
However, it is often not easy for the listener to hear exactly what an artist is saying or singing, or to follow along quickly enough – consider for example, Eminem's double-time rap. Regardless of your taste in music, for many songs you might wish you had a simple option for displaying the lyrics, so you can follow along without having to search for the lyrics on the web time and time again. Some players do offer the ability to display the lyrics for the track currently playing, but even better, the Lyrics-in-terminal [1] program can retrofit this feature for virtually any player.

Installation

Currently you will not find the program in the package sources of the common distributions. Users of Arch Linux and its derivatives have the easiest approach to installing the application: Just download the package *lyrics-in-terminal* from the Arch User Repository (AUR) onto your system. If you are working with a different distribution, use the pip Python package manager instead, and then run the commands from Listing 1 to install the program on Debian or Ubuntu.

The package manager installs the program into `~/local/bin/`. However, the shell only adds this directory to the `PATH` environment variable if the directory already exists at login time. For the program's `lyrics` command to work, you may need to log out and log back in again. If the system still does not find the command, check the `~/profile` in your home directory and append the lines from

Figure 1: By default, Lyrics-in-terminal attaches itself to the Spotify client playback.



Listing 1: Installation

```
$ sudo apt install python3-pip
$ pip install lyrics-in-terminal
```

Listing 2: ~/.profile

```
[...]
# set PATH so it includes user's
# private bin if it exists
if [ -d "$HOME/.local/bin" ]; then
    PATH="$HOME/.local/bin:$PATH"
fi
```

Listing 2. After logging out and back on again, the shell command should work.

Lyrics in the Terminal

By default, Lyrics-in-terminal automatically connects to Spotify. You just need to launch the program in a terminal window using the `lyrics` command to play any song with the official Spotify client (Figure 1). A short while later, the song lyrics will appear in the terminal window; you do not need a commercial premium account. In the header, you will also see the name of the song, the artist, and below it the album title. If the length of the lyrics exceeds the space in the terminal, you can scroll through the contents with the arrow keys.

In our test, Lyrics-in-terminal worked better than the feature included in Spotify's Android client (the PC variant of the client does not offer this option as of yet). The display in the Android app only provides results for really well-known artists, while Lyrics-in-terminal had no problem identifying salsa from the Caribbean, chansons from France, fado from Portugal, or even

Listing 3: Lyrics-in-terminal

```
$ lyrics
$ lyrics rhythmbox
$ lyrics vlc
$ lyrics Lollypop
$ lyrics org.gnome.Music
```

Deutschrap (German hip hop). The program only showed a *lyrics not found* message for instrumentals or acoustic techno tracks – no surprises there.

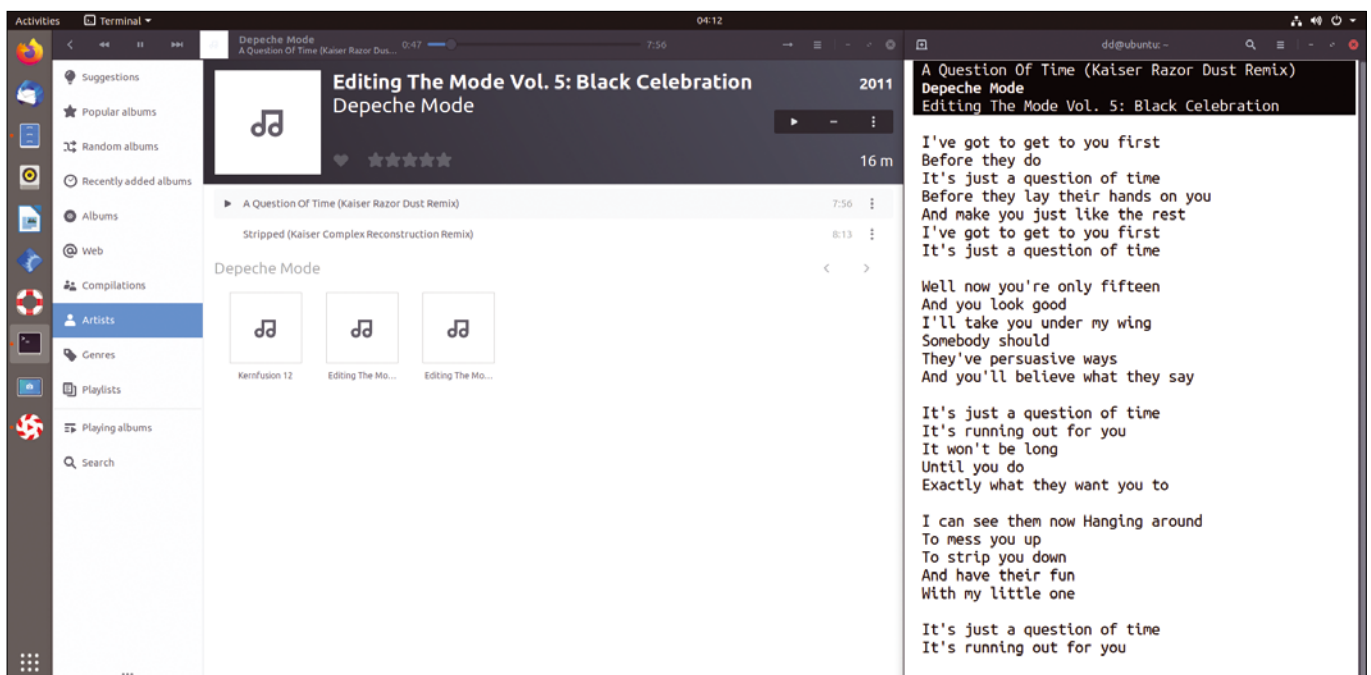
Under the hood, Lyrics-in-terminal does not use its own database, but accesses the freely available data from AZLyrics [2]. This project has been collecting song lyrics for years and legitimately licensing them from record labels. There is no danger that the texts will disappear off the web overnight. If lyrics are missing or incorrect, the community can add the data via the service's portal.

Not Only Spotify

Now, not every user uses Spotify to play music. No problem: Lyrics-in-terminal does not tap the data about the song currently playing from the player, but via the native D-Bus and the Media Player Remote Interfacing Specification (MPRIS) [3] – a feature that most media players on Linux support. For example, the desktop environments use the technology to control the player's playback via widgets or to display information about the current song.

You can specify which media players Lyrics-in-terminal should now listen out for via an option attached to the call. The `lyrics` command

Figure 2: The automatic lyrics search feature works with any audio player that can be controlled via D-Bus/MPRIS.



links with Spotify as described, while `lyrics` `rhythmbox` links with the Gnome desktop's Rhythmbox media player. The connection works with any media player that can be controlled via D-Bus/MPRIS – that is, with just about any music player on Linux. We tested VLC, Audacious, Lollypop (Figure 2), and the still quite young Gnome Music player.

The only problem here is the correct spelling: You have to write the options `rhythmbox` or `vlc` in lowercase. `Lollypop`, on the other hand, must start with an uppercase letter, and for Gnome Music you must specify the D-Bus name `org.gnome.Music`. Otherwise, the program thinks that the desired player would not run (Listing 3).

If linking to the desired player does not work right away, you need to check whether you first have to enable support for MPRIS in the player itself. Then experiment with the different spellings if necessary.

As a tip, to avoid the need to specify the audio player as an option, it is a good idea to create an alias for the `lyrics` command. To do this, edit the shell's configuration, which is usually the `~/.bashrc` file, and add an entry such as:

```
alias lyrics='lyrics Lollypop'
```

After restarting the terminal program, the requirement to specify your preferred player is then dropped.

You can determine the D-Bus name quite easily using a D-Bus debugger like D-Feet [4]. This program is included in the official package sources of most distributions. After launching the application, first switch to the *Session Bus* in the header of the window and then enter the name of the player in the search field in the left sidebar. The name then appears as a search result; on the right in the display area, the name can then also be copied and pasted to the clipboard (Figure 3).

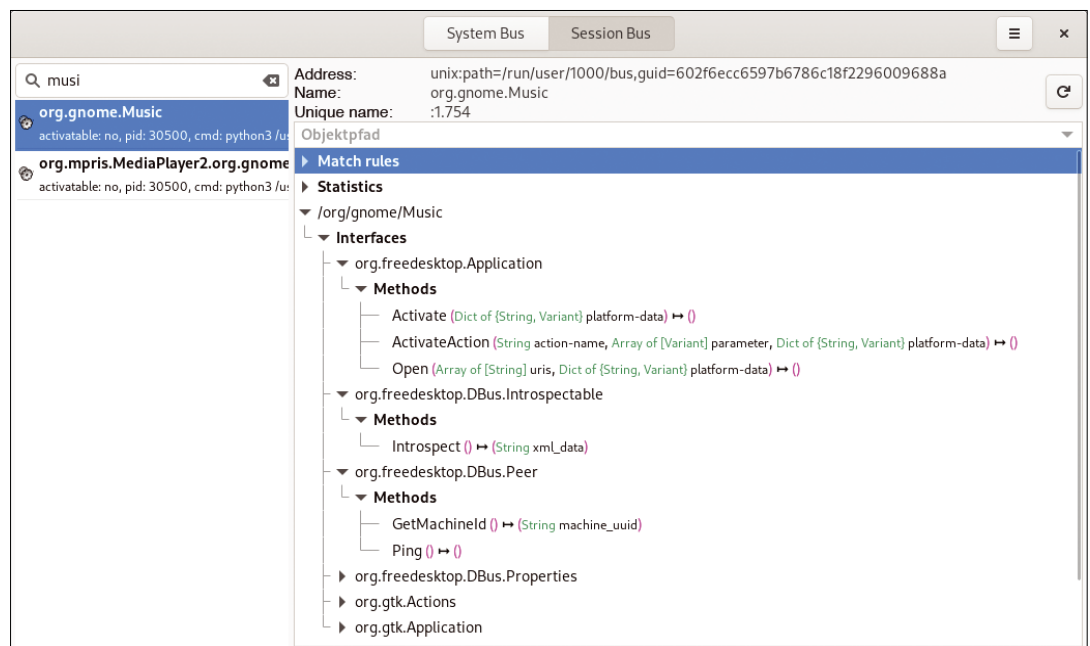
Conclusions

Lyrics-in-terminal is one of those small but good-to-have terminal helpers that extend their usefulness beyond the terminal window. The program reliably and automatically downloads the right lyrics off the web, no matter what player you use and what music you like to listen to. You are bound to find some artists whose work is not covered by AZLyrics, but the number of tracks I tried that successfully passed the test was impressive. ■■■

Info

- [1] Lyrics-in-terminal project page: <https://github.com/Jugran/lyrics-in-terminal>
- [2] AZLyrics: <https://www.azlyrics.com>
- [3] MPRIS Specification: <https://specifications.freedesktop.org/mpris-spec/latest/>
- [4] D-Feet: <https://wiki.gnome.org/Apps/DFeet>

Figure 3: If necessary, you can determine the D-Bus name of a player with a D-Bus debugger such as D-Feet.



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.



#245/April 2021

Choose a Shell

You're never stuck with the same old command shell – unless you want to be. This month we review some of the leading alternatives.

On the DVD: Arch Linux 2021.02.01 and MX Linux mx-19.03



#244/March 2021

Stream Processing

The explosion of real-time data from sensors and monitoring devices is fueling new interest in alternative programming techniques. This month we waded into stream processing.

On the DVD: FreeBSD 12.2 and GhostBSD 20.11.28



#243/February 2021

iNet

With Linux, more innovation is always on the way. This month we take a look at the iNet wireless daemon, a new wireless client that is poised to replace the venerable WPA Supplicant.

On the DVD: Linux Mint 20 and Kali Linux 2020.4



#242/January 2021

3D Printing

The weird, wonderful, futuristic world of 3D printing is waiting for you right now if you're willing to invest a little time and energy. This month we help you get started with practical 3D printing in Linux.

On the DVD: Ubuntu 20.10 "Groovy Gorilla" and Fedora 33 Workstation



#241/December 2020

Secure Your System

Security often means sophisticated tools like firewalls and intrusion detection systems, but you can also do a lot with some common-sense configuration. This month we study some simple steps for securing your Linux.

On the DVD: KDE neon 5.20.0 and elementary OS 5.2



#240/November 2020

It's Alive

Build a whole operating system? Well maybe not a Linux, but if you're interested, you can implement the basic features of an experimental OS using resources available online. We can't show you the whole process, but we'll help you get organized and take your first steps.

On the DVD: Linux Magazine 20th Anniversary Archive DVD

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.

SUSECON Digital 2021

Date: May 18-20, 2021

Location: Virtual Event

Website: <https://www.susecon.com/>

Lead your company into a digital transformation by applying reliable, secure open source solutions from technical experts, ecosystem partners, and your peers from around the world. SUSECON Digital 2021 brings you the best of enterprise Linux, Kubernetes management, and edge solutions wherever you are.

LISA21

Date: June 1-3, 2021

Location: Virtual Event

Website: <https://bit.ly/LISA-21>

LISA is the premier conference for operations professionals, where sysadmins, systems engineers, IT operations professionals, SRE practitioners, developers, IT managers, and academic researchers share real-world knowledge about designing, building, securing, and maintaining the critical systems of our interconnected world.

Events

NSDI '21	April 12-14	Virtual Event	https://www.usenix.org/conference/nsdi21
DrupalCon North America 2021	April 12-16	Virtual Event	https://events.drupal.org/drupalcon2021
DevOpsCon London Hybrid Edition	April 20-23	London, UK and Online	https://devopscon.io/london/
KubeCon + CloudNativeCon	May 4-7	Virtual Event	https://bit.ly/kube-cloudnativecon
Linux Storage Filesystem & MM Summit	May 12-14	Palm Springs, California	https://events.linuxfoundation.org/lfsfmm/
Linux App Summit 2021	May 13-15	Virtual Event	https://linuxappsummit.org/
SUSECON Digital 2021	May 18-20	Virtual Event	https://www.susecon.com/
LISA21	June 1-3	Anaheim, California	https://www.usenix.org/conference/lisa21
ODSC Europe	June 8-10	Virtual Conference	https://odsc.com/europe/
SYSTOR 2021 Hybrid	June 14-16	Haifa, Israel	https://www.systor.org/2021/venue.html
stackconf online 2021	June 15-16	Virtual Event	https://stackconf.eu/
openSUSE Virtual Conference 2021	June 18-20	Virtual Event	https://events.opensuse.org/
ISC High Performance 2021 Digital	June 24-July 2	Virtual Event	https://www.isc-hpc.com/
USENIX ATC '21	July 14-16	Santa Clara, California	https://www.usenix.org/conference/atc21
Embedded Linux Conference North America	August 3-6	Vancouver, British Columbia	https://events.linuxfoundation.org/
Open Source Summit North America	August 3-6	Vancouver, British Columbia	https://events.linuxfoundation.org/
USENIX Security '21	August 11-13	Vancouver, British Columbia	https://www.usenix.org/conferences

CALL FOR PAPERS

We are always looking for good articles on Linux and the tools of the Linux environment. Although we will consider any topic, the following themes are of special interest:

- System administration
- Useful tips and tools
- Security, both news and techniques
- Product reviews, especially from real-world experience
- Community news and projects

If you have an idea, send a proposal with an outline, an estimate of the length, a description of your background, and contact information to edit@linux-magazine.com.



The technical level of the article should be consistent with what you normally read in *Linux Magazine*. Remember that *Linux Magazine* is read in many countries, and your article may be translated into one of our sister publications. Therefore, it is best to avoid using slang and idioms that might not be understood by all readers.

Be careful when referring to dates or events in the future. Many weeks could pass between your manuscript submission and the final copy reaching the reader's hands. When submitting proposals or manuscripts, please use a subject line in your email message that helps us identify your message as an article proposal. Screenshots and other supporting materials are always welcome.

Additional information is available at:

http://www.linux-magazine.com/contact/write_for_us.

Contact Info

Editor in Chief

Joe Casad, jcasad@linux-magazine.com

Copy Editors

Amy Pettie, Megan Phelps

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

© Photo by Jason Chen on Unsplash.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
Zack Brown	12
Bruce Byfield	6, 26, 44
Joe Casad	3
Mark Crutch	71
Rainer Grimm	56
Karsten Günther	73
Jon "maddog" Hall	72
John Knight	80
Charly Kühnast	48
Christoph Langner	16, 92
Vincent Mealing	71
Paul Menzel	20
Pete Metcalfe	60
Graham Morrison	86
Manfred Puckhaber	64
Mike Schilli	50
Tim Schürmann	38
Jack Wallen	8

Issue 247 / June 2021

Attack of the Quantum Computers

The encryption techniques that seem safe today won't protect your secrets in the quantum era. Next month we examine the challenge of security in the age of quantum computing.

Approximate

UK / Europe	May 08
USA / Canada	June 04
Australia	July 05

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 © ELEN, fotofra.com



Powerful business workhorse

TUXEDO Aura 15



AMD Ryzen 7 4700U
8 Cores | 15 Watt



USB-C 3.2 Gen2
DisplayPort & PD



2 cm | 1,65 kg
slim & light



4G / LTE
Cellular Modem



100%
Linux

5

Year
Warranty



Lifetime
Support



Built in
Germany



German
Privacy



Local
Support

TUXEDO
COMPUTERS

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

HETZNER



CLOUD SERVER

STARTING AT
\$ 2.97

All prices exclude VAT and are subject to the terms and conditions of Hetzner Online GmbH. Prices are subject to change. All rights reserved by the respective manufacturers.

A little money gets you lots of Cloud

with high performance AMD EPYC™ Processors

- ✓ Load Balancer
- ✓ Networks
- ✓ Firewalls
- ✓ Volumes
- ✓ Floating IPs
- ✓ API with CLI tool
- ✓ Official Go & Python libraries
- ✓ and many more

cloud.hetzner.com