FREE DVD

Double-Sided DVD INSIDE!

USB PASSWORD MANAGER and lots of other timely and useful tools

SAFER BOOT Making Secure Boot work with Linux

LINUX PRO MAGAZINE

JANUARY 2018

SAFER BOOT

Keeping control of the startup process

UEFI Tricks
Add a custom app that runs from the firmware

5 Automated Backup Tools

Pi FM Radio
Build a stairway to maker heaven

DDoS Attack
Avoiding a denial of service nightmare

Stacer
Clean up your Linux system

LINUXVOICE

- Marketing Free Software
- Phipps: FOSS Communities Need Equal Rights
- Digitize Your LPs with Audacity

FOSSPicks
- qutebrowser 1.0
- Storyboarder
- CoreFreq
- Fragment synthesizer

Tutorials
- ffmpeg

Issue 206
Jan 2018
US$ 15.99
CAN$ 17.99

WWW.LINUXPROMAGAZINE.COM

HETZNER ONLINE

RYZE 'N' SHINE

More parallelization. More power.

Dedicated Root Server AX60-SSD

AMD Ryzen 7 1700X
Octa-Core "Summit Ridge" (Zen)
64 GB DDR4 RAM
2 x 500 GB SATA 6 Gb/s SSD
100 GB Backup Space
30 TB traffic inclusive*
No minimum contract
Setup Fee $131

monthly $ 65

The ideal solution for highly-parallelizable processes.

Our new Dedicated Root Server AX60-SSD houses the latest generation AMD processor, the octa-core AMD Ryzen 7 1700X, which is based on Zen architecture. Its performance is truly impressive, especially when used for purposes such as virtualization, encryption, and data compression, which require several processor cores.

www.hetzner.de/us

* There are no charges for overage. We will permanently restrict the connection speed if more than 30 TB/month are used. Optionally, the limit can be permanently cancelled by committing to pay $1.30 per additional TB used.

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.

TOMORROW'S NEWS

Dear Reader,

When I started working for this magazine, way back in 2004, Linux was really picking up momentum. That was back in those years when every year was supposed to be "the year of the Linux desktop," and the epic Linux vs. Windows battle was revving to a full burn.

One of the best examples of the Linux juggernaut was the city of Munich, which declared it would be transitioning all its computers to Linux in 2004. We at Linux New Media were particularly excited about this move because, at the time, the world headquarters of our small company was located in Munich, and our German colleagues were part of that groundswell of Linux support that launched the transition.

Over the years, the City of Munich worked hard on Linux integration, even maintaining their own Linux distribution (LiMux, an Ubuntu derivative) and building their own LibreOffice extension set (WollMux). The move to Linux was officially completed in 2013, but the solution still never really settled in.

Now, 13 years after the announcement that Munich was switching to Linux, they have decided they are switching back to Windows. The move has been brewing for some time, with city officials signaling it might happen for a couple years.

What happened with Linux in Munich? It depends on who you ask. In a blog post earlier this year, Björn Schießle of the Free Software Foundation Europe wrote, "In 2014, Dieter Reiter was elected new mayor of Munich. He had referred to himself as 'Microsoft fan' even before he took office. He prides himself with having played a major part in the decision to move the Microsoft Germany headquarters to downtown Munich. He started to question the LiMux strategy as soon as his term started, and asked Accenture, a Microsoft partner in the same building as Microsoft, to analyze Munich's IT infrastructure."

According to Accenture's study (and to other observers), the problems with Linux in Munich were mainly organizational – not technical. Another issue identified by Linux critics was the amount of city-government-related application software that simply wouldn't run on Linux, which meant that the city found itself in the position of doing its own software development. (This, again, is not really a problem with Linux, but with the habits and priorities of the software development industry.)

It probably would have helped if more cities had joined Munich to spread the brainshare for development and implementation knowledge. But then, the battle is never really over as long as a well-funded opponent has a stake in turning the tables. Environmental groups and historical preservation groups have encountered the same scenario: Non-profit communities can turn out lots of people with big hearts to wave placards, but a huge corporation can unleash an army of lobbyists and marketing professionals who keep chipping away at public opinion when everyone else has gone back to their day jobs.

Software has no shape or solid edges. The value of a software product is simply what whoever owns it says they will charge you for it. If you try to buy a Windows desktop system as a single user, you get one price. If you buy a large number of licenses, you get a different price, depending on how much Microsoft wants to close the deal. In the case of Munich, Microsoft *really really really* didn't want to see this major European city become a test case for Linux adoption.

Ultimately, though, one city flipping back to Windows isn't such a big victory for Microsoft – or a big defeat for Linux. Is a city government really the best place for Linux to make its mark? Linux is secure and efficient, and it will save you money, but you have to tune in to it – you have to make a real effort to make it work in a world where Windows is still very much the default for non-technical users.

Linux has all kinds of success stories in the corporate sector, where energy and innovation translate more fluidly into profit and career advancement. Companies like Red Hat, SUSE, Canonical, and IBM make billions of dollars installing Linux systems (including Linux desktop systems) in Fortune 500 companies around the world. On the server side, Linux is stronger than ever, and even Microsoft has to acknowledge the importance of Linux to the booming cloud and container business.

But really, regardless of how Munich or any other city chooses to spend their IT budget, the truth is, all the things we were fighting about back in 2004 were settled long ago. Desktop operating systems aren't as important as we used to think they were. Microsoft doesn't think Linux is a cancer anymore (in fact, they say they *love* Linux), and everyone in IT is aware of the power and importance of Free Software.

Like many in the Linux community, I have a feeling the people of Munich are going to spend *a lot* more money in the long run by climbing back into the proprietary software time machine, but hey, that's tomorrow's news. ■■■

Joe

Joe Casad, Editor in Chief

Info

[1] LiMux: *https://en.wikipedia.org/wiki/LiMux*

[2] WollMux: *https://www.wollmux.net/wiki/Main_Page*

[3] "What Happened in Munich": *https://fsfe.org/news/2017/news-20170301-01.en.html*

LINUX MAGAZINE

WHAT'S INSIDE

This month we look at the Linux boot process and explore some tools and tricks for better and safer startup, including the Shim first-stage bootloader and the Trusted Platform Module (TPM) chip. Other highlights include:

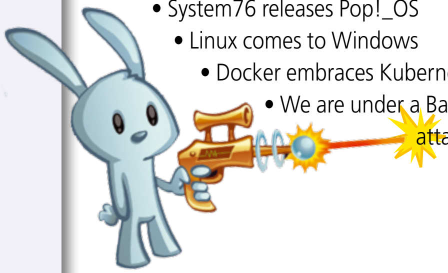
- **DDoS Defense** – Some providers offer special services aimed at thwarting denial-of-service attacks. (page 46).
- **MakerSpace** – Check out our articles on FM radio and Volumio audio playback on the Raspberry Pi. (page 58).

Elsewhere up ahead, LinuxVoice looks at audio and video with Audacity and the FFmpeg command-line video editor, and Graham puts the focus on small stuff with minimal browsers and minimal text editors in this month's FOSSPicks.

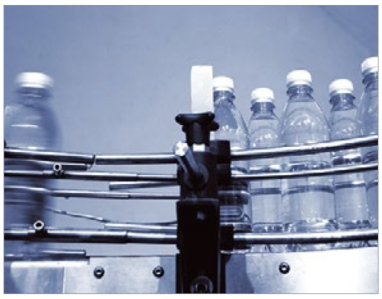
SERVICE

COVER STORIES

NEWS



REVIEWS



On the DVD

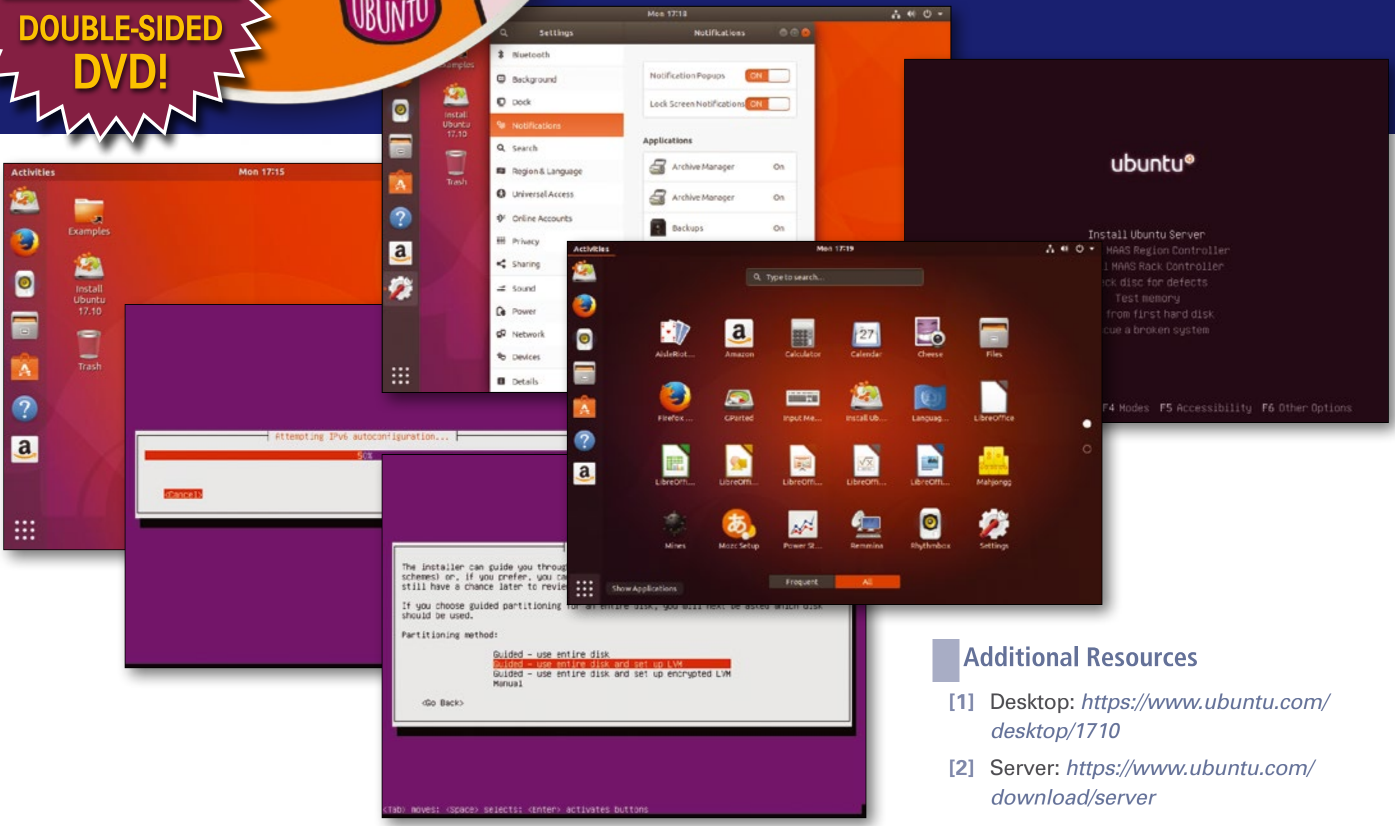
Ubuntu 17.10 Server and Desktop (64-bit)

Ubuntu has come full circle and begins again at the top of the alphabet with Artful Aardvark. Version 17.10 is the last release before the 18.04 long-term support (LTS) version; as usual for semiannual releases, you get nine months of security and maintenance updates.

This release features Linux kernel 4.13, which includes new KVM features and other enhancements. Instead of a partition, swap is now a file that scales to your needs.

The Live Desktop version now comes with Gnome Shell and GDM as the default display manager. Wayland is the default display server, but the older display server is still available by choosing *Ubuntu on Xorg* from the cog on the login screen. Gnome Shell features a new settings application and a movable dock.

Note that "USB printers do not get set up automatically and IPP-over-USB does not work at all. Please set up your USB printer using *Devices | Printers*" in Gnome Settings. For driverless printing, connect your printer over Ethernet or WiFi or not upgrade to 17.10 until the problem is fixed [3].



```
Attempting IPv6 autoconfiguration...
50%
<Cancel>
```

```
Install Ubuntu Server
 MAAS Region Controller
l MAAS Rack Controller
ck disc for defects
 Test memory
 from first hard disk
cue a broken system

F4 Modes  F5 Accessibility  F6 Other Options
```

```
The installer can guide you throug
schemes) or, if you prefer, you ca
still have a chance later to revie

If you choose guided partitioning
should be used.

Partitioning method:

              Guided - use entire disk
              Guided - use entire disk and set up LVM
              Guided - use entire disk and set up encrypted LVM
              Manual

    <Go Back>

<Tab> moves; <Space> selects; <Enter> activates buttons
```

Additional Resources

[1] Desktop: *https://www.ubuntu.com/desktop/1710*

[2] Server: *https://www.ubuntu.com/download/server*

[3] Release notes: *https://wiki.ubuntu.com/ArtfulAardvark/ReleaseNotes*

Defective discs will be replaced. Please send an email to subs@linux-magazine.com.

NEWS

Updates on technologies, trends, and tools

THIS MONTH'S NEWS

Samsung to Bring Linux to the Galaxy Phone

The same year Canonical decide to pull out of the consumer space, Samsung is bringing a pure desktop Linux experience to PCs. Unlike Apple, Google, or Microsoft, Samsung doesn't have any tightly integrated offering for professionals who need a desktop to get work done. Samsung came out with DeX, an accessory for Samsung Galaxy phones that connects with a monitor and offers a desktop-like interface. It's an experience similar to Ubuntu Dock or Motorola Atrix Webtop.

However, the desktop experience was subpar compared with Mac OS or Windows. Samsung is now looking at desktop Linux for DeX. "Installed as an app, Linux on Galaxy gives smartphones the capability to run multiple operating systems, enabling developers to work with their preferred Linux-based distributions on their mobile devices. Whenever they need to use a function that is not available on the smartphone OS, users can simply switch to the app and run any program they need to in a Linux OS environment," Samsung said in a press release.

Samsung is quite ambitious about the project; the company is also luring developers, a market that already has a strong hold on desktop Linux. "Now developers can code using their mobile on-the-go and seamlessly continue the task on a larger display with Samsung DeX," said the company.

While it's currently in the trial phase, Samsung plans to bring DeX to larger displays. If it does gain mindshare, Samsung might even consider desktop Linux-powered laptops.

One advantage Samsung has over traditional desktop Linux distributions is that Samsung owns the entire hardware chain, from touchscreen to storage. It will be relatively easier for Samsung to offer a fully polished desktop Linux experience compared with a community-based distro, where developers either rely on reverse engineering or are at the mercy of hardware vendors to offer drivers.

Desktop Linux users may finally see the year of Linux. "Linux on Galaxy is made even more powerful because it is DeX-enabled, giving developers the ability to create content on a large screen, powered only by their mobile devices. This presents a significant step forward for software developers, who can now set up a fully functional development environment with all the advantages of a desktop setting that is accessible anytime, anywhere. Samsung Linux on Galaxy is still a work in progress," said Samsung.

If you are interested in an early notification of availability, please sign up: *https://seap.samsung.com/linux-on-galaxy*.

System76 Releases Pop!_OS

System76, one of the few hardware vendors that sell systems preloaded with Linux, has released the final version of Pop!_OS, their own Ubuntu-based distribution.

System76 CEO and founder Carl Richell told us in an interview that the OS is the result of customer feedback. What makes Pop!_OS different from many other Linux distributions is that System76 sells Linux hardware, so they do have a very trusted channel of customer feedback.

System76 caters to professionals who use desktop Linux for their workloads. In a press release, System76 said that the OS is geared toward users in STEM, computer scientists, makers, and developers. During the release of 3D-rendering software as open source, Pixar developers were spotted using System76 machines running Red Hat Enterprise Linux.

Pop!_OS is seen as a System76 response to Canonical's withdrawal from the consumer space. Richell said that their current focus is on offering a very stable and minimalistic experience around Ubuntu and Gnome to cater to its customers.

Contrary to Linux Mint, Pop!_OS will be based on the latest Ubuntu instead of the long-term support (LTS) version. Richell said that they have been working with Ubuntu for more than 12 years, and they have all the needed expertise to keep up with Ubuntu. The good news is, now that Unity is discontinued, Ubuntu will focus on a rock solid base that can be used in enterprise setups. Because Gnome is already mature, it's more predictable than Unity, making the job of System76 developers easy to tack to slow-moving stable targets.

The primary focus of System76 will be to offer a very polished experience on the machines, including hardware support and optimization. System76 is also working on setting up their own manufacturing unit, where they will build desktops and laptops in-house.

Even though Pop!_OS is designed for System76 machines, it's freely available for anyone to use.

You can download it from GitHub: *https://github.com/pop-os*.

Linux Comes to Windows

Microsoft has announced that Windows Subsystem for Linux (WSL, also known as Bash on Windows) is now out of beta. With Windows 10 Fall Creator update, every Windows user will be able to use the feature. However, WSL is not enabled by default. Users have to enable it from the Settings *turn Windows features on or off* feature.

Microsoft will offer supported Linux distributions from the Windows Store, so there is no need to install them manually. Some of the supported distros include openSUSE, SUSE Linux Enterprise, and Ubuntu. Fedora is expected to arrive soon. Microsoft will offer official support for these distributions in partnership with the respective distribution.

Customers can now run multiple Linux distributions, which means they can use commands, utilities, and tools specific to different distributions.

Image © Sergei Popov, 123RF.com

Although WSL is still in the works, it now supports USB mounts that gives developers access to USB devices from Linux.

Microsoft is also bringing WSL to Windows Server and Azure Cloud. "Using WSL, Windows Server administrators, DevOps engineers, developers, etc., will be able to run their favorite Linux tools, apps, and scripts, alongside their favorite Windows admin tools. This will make it easier than ever before to automate, control, manage, and deploy an ever broader portfolio of technologies and tools atop Windows Server," wrote Microsoft Program Manager Rich Turner in a company blog.

WSL is intended for developers who need native Linux tools to run and manage their Linux systems on Azure and other clouds; officially, it's not intended for desktop users.

Docker Embraces Kubernetes

At DockerCon Europe, Solomon Hykes, the founder of Docker, announced support for Kubernetes as an orchestration platform alongside Swarm, it's own orchestration tool.

"The addition of Kubernetes as an option alongside Swarm gives our users and customers the ability to make an orchestration choice with the added security, management, and end-to-end Docker experience that they've come to expect from Docker since the very beginning. We look forward to working with the Kubernetes community to help users, partners, and customers achieve the full benefits of the containerization revolution," said Hykes.

The company said that through its integration with Docker EE, Kubernetes will be available across certified infrastructure platforms, including multiple Linux distributions (SLES, CentOS, RHEL, Ubuntu, Oracle Linux) and Windows, as well as all cloud platforms, including AWS and Azure.

Developers running Docker on Mac and Windows will be able to use features like multistage builds and application composition (Docker Compose) in container development and have them run consistently from development all the way to production. Developers have the flexibility to write their applications in Docker and can choose their orchestrator without requiring any additional modification.

In an interview with *Linux Pro*, Hykes said that Docker will continue to engage with the Kubernetes community as a good citizen.

We Are Under Bad Rabbit Attack

A new variant of the NotPetya worm, dubbed Bad Rabbit, is wreaking havoc on Windows systems across the globe. The attack initially targeted Russian and Ukrainian corporate networks, but it has now spread across the globe inflecting Turkey, Bulgaria, Japan, Germany, Poland, South Korea and even the United States.

US-CERT, a US agency responsible for mitigating cyber threats, has released an alert, "US-CERT has received multiple reports of ransomware infections, known as Bad Rabbit, in many countries around the world. A suspected variant of Petya, Bad Rabbit is ransomware – malicious software that infects a computer and restricts user access to the infected machine until a ransom is paid to unlock it. US-CERT discourages individuals and organizations from paying the ransom, as this does not guarantee that access will be restored. Using unpatched and unsupported software may increase the risk of proliferation of cyber security threats, such as ransomware."

Researchers at Kaspersky Lab said that the ransomware dropper was distributed with the help of drive-by attacks. "While the target is visiting a legitimate website, a malware dropper is being downloaded from the threat actor's infrastructure. No exploits were used, so the victim would have to manually execute the malware dropper, which pretends to be an Adobe Flash installer," said Orkhan Mamedov, Fedor Sinitsyn, and Anton Ivanov of Kaspersky Lab.

Kaspersky Lab suggests disabling WMI on Windows systems to stop Bad Rabbit from digging burrows in your networks.

Zack's Kernel News

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

I3C Support

Boris Brezillon posted some patches to implement a portion of the I3C core infrastructure. This is a wholesale upgrade of the I2C protocol for communications through serial ports. A lot of sensor devices use serial communications, because it's a simple two-wire interface. However, as that simplicity brings a proliferation of sensor devices, it becomes more important to manage the increased bandwidth and interrupt needs they create. I3C is designed to do that.

Boris' approach would transparently handle I2C backward compatibility for minimal user pain, but he also made certain compromises that would make using his APIs more difficult, and he left a fair chunk of the I3C API unimplemented for now, although he intends to fill it out in the future.

One implementation compromise was to require user code to run in a non-atomic context (i.e., only when the current process can be interrupted by something else). That's a slight annoyance, because it requires user code to be aware of its current state when calling the I3C API. However, Boris indicated he'd be fine with changing that. He'd mainly done it as a shortcut.

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.**

Among the missing API calls, Boris left out hot plugging support, which might be a deal-breaker for some users. Of course, the missing APIs would all be added later as the I3C code was fleshed out.

Wolfram Sang took a look at the code, but had no serious objections and generally approved the patches.

Arnd Bergmann asked why Boris had created a whole new subsystem for I3C, instead of simply extending the existing I2C code to support I3C as well. Boris replied:

I3C and I2C are really different. I'm not talking about the physical layer here, but the way the bus has to be handled by the software layer. Actually, I thin[k] the I3C bus is philosophically closer to auto-discoverable buses like USB than I2C or SPI.

Indeed, all I3C devices can be discovered and do not need to be described at the board level (using DT, board files, ACPI, or whatever). Also, some I3C devices are hot pluggable, and most importantly, all I3C devices describe themselves during the discovery procedure (called DAA in the I3C world).

There is some kind of "device class" concept. In the I3C world it's called DCR (Device Characteristic Register), but it plays the same role: It's a set of generic interfaces devices have to comply with when they declare themselves as being compatible with a DCR ID (like accelerometer, gyroscope, or whatever). [...]

Devices also expose a 48-bit Provisional ID which is made of sub-fields. Two of them are particularly interesting: the manufacturer ID and the part ID, which are comparable to the vendor and product ID in the USB world.

These three [pieces of] information (DCR, ManufacturerID, and PartID) can be used to match drivers instead of the compatible string or driver-name used for I2C devices.

So, as you can imagine, dealing with an I3C bus is really different from dealing with an I2C bus.

Boris added, "Of course, I can move all the code in `drivers/i2c/`, but that won't change the fact that I3C and I2C buses are completely different with little to share between them."

Arnd didn't want to let the idea go quite yet, though. He agreed that there were reasons to oppose extending I2C to cover I3C as well, but he felt there were also reasons to go through with it, even if it involved creating an ugly mess behind the scenes. He said, "there is value in representing the physical bus hierarchy in the software model, and if I2C and I3C devices can be attached to the same host bus, a good abstraction should show them under the same parent. This is true for both the kernel representation (in `sysfs` and the data structures) as well as the device tree binding (assuming we will need to represent I3C devices at all). The two don't have to use the same model, but it's easier if they do."

He also added, "We have discussed whether I2C and SPI should be merged into a single `bus_type` in the past, as a lot of devices can be attached to either of them. If it's common enough for I3C devices to support an I2C fallback mode, having a common `bus_type` might noticeably simplify device drivers by only requiring a single `i2c_driver` structure. Simplifying many drivers a little bit can in turn offset the added complexity in the subsystem."

Boris didn't really see the benefit of these ideas and expressed surprise that I2C and SPI might ever be merged. He asked Arnd for an explanation, and Arnd replied, "well, we never changed it, so at least the work required to merge the two was considered too much to justify any advantages." But he described the rationale as simplifying kernel build-time config options. He said:

The main problem with having one driver that can operate on different bus types (I2C plus either SPI or I3C) is the handling for the various combinations in configurations (e.g., I2C = m, SPI = y).

The easy case is having a `module_init` *function that registers two device drivers, but that requires having a Kconfig dependency on both subsystems, and you can't use the* `module_i2c_driver()` *helper.*

The second way is to have a number of `#ifdef` *and complex Kconfig dependencies for the driver to only register the* `device_driver` *objects for the buses that are enabled. This is also doable, but everyone gets the logic wrong the first time.*

What we end up doing to work around this for other drivers is to have the base driver in one library module, and separate modules for the bus-specific portions, which can then use `module_i2c_driver` *again. There are many instances for combined I2C/SPI drivers in the kernel, and it works fine, but it adds a fair bit of overhead compared to having one driver that would, e.g., use* `regmap` *to abstract the differences in the* `probe()` *function and otherwise keeps everything in one place.*

This made sense to Boris, and he could see the value of having a single subsystem, but he still hated the ugliness of the potential merged implementation. He asked, "Can't we solve this problem with a `module_i3c_i2c_driver()` macro that would hide all this complexity from I2C/I3C drivers?"

Wolfram, who had earlier approved Boris's patches, felt that merging I2C with I3C was possibly significantly different from merging I2C with SPI. In the latter case, there was a decent likelihood that the system might only have one or the other, in which case supporting both would cover the most possibilities. But he doubted there was any hardware currently implementing both I2C and I3C. And since the I3C code was backward compatible with I2C, that seemed to obviate the need for a merger – the user could simply plug the I2C device into the port and use it.

Boris replied that although he didn't know of any devices implementing both protocols at the moment, "the spec clearly describe[s] what legacy/static addresses are for and one of their use case[s] is to connect an I3C device on an I2C bus and let it act as an I2C device," which Wolfram agreed made it more likely that a device would one day implement both protocols.

At some point, Boris took a whack at merging I2C and I3C, but he found it difficult to conceptualize a workable design. He also added, "It's kind of hard to design something when you don't have real devices. I guess I can mimic I2C for now and make it evolve based on users' needs."

Elsewhere, Greg Kroah-Hartman asked Boris to split the documentation out from the patch and submit it separately, to make the code slightly easier to go through, but he also did go through the code and offered several technical criticisms, which Boris then said he'd address.

Greg also deduced from a missing data type that Boris had never actually tested removing an I3C device once it had been installed. Boris replied, "You got me, never tried to remove a device." He asked some technical questions about how best to handle that case.

At this point, the discussion veered off into technical locking issues, as Greg and Boris worked on resolving the need to remove devices from a running system.

There was no ultimate resolution to any of the questions raised in the discussion, but it seems clear that Boris's first attempt will probably be reworked to fit Greg and Arnd's objections. It also seems like the task might be bigger than Boris had first anticipated, and he may not have the physical equipment he needs. (He remarked at one point, "all my tests have been done with dummy/fake I3C slaves emulated with a slave IP.") So, although it seems like I3C will definitely be getting into the kernel sooner rather than later, there are still some significant hurdles to overcome.

Fixing mmap()

Dan Williams had a thorny conundrum. The `mmap()` system call didn't validate unknown flags, which meant that any modern new `mmap` behaviors that didn't work on older systems couldn't have a graceful failover. Dan wanted to implement a new system call, `mmap3()`, that would validate all flags.

There were a couple of problems. For one thing, Christoph Hellwig pointed out that "Adding new syscalls is extremely painful; it will take forever to trickle this through all architectures (especially with the various 32-bit architectures having all kinds of different granularities for the offset) and then the various C libraries, never mind applications."

Christoph suggested just using an existing `__MAP_VALID` hack as a workaround.

Dan replied, "I agree with the mess and delays it causes for other archs and libc, but at the same time this is for new applications and libraries that know to look for the new flag, so they need to do the extra work to check for the new syscall."

Regarding the possibility of using `__MAP_VALID`, he also pointed out that "any new `mmap` flag with this hack must be documented to only work with `MAP_SHARED` and that `MAP_PRIVATE` is silently ignored."

However, he said he wasn't totally opposed to doing things this way if it turned out not to be too onerous.

Christoph went digging around in the `mmap` code and thought he found a way to avoid the problem with ignoring `MAP_PRIVATE`. In which case, he felt it certainly beat the hell out of adding a new system call.

However, Kirill A. Shutemov looked at Christoph's discovery and felt that certain architectures, in particular PA-RISC, wouldn't be able to support the fix Christoph had in mind. Christoph rejoined, "I'd be happy to say that we should not care about PA-RISC for persistent memory. We'll just have to find a way to exclude PA-RISC without making life too ugly." Kirill hated this idea, though, saying that system call interfaces should be universal and not have different behaviors depending on which machine they were run on.

The debate continued, although it seemed to be getting further afield from the original issue. Eventually Dan brought the conversation back around to the real question, saying:

The problem here is that to support new the `mmap` *flags the arch needs to find a flag that is guaranteed to fail on older kernels. Defining* `MAP_DIRECT` *to* `0x8` *on PA-RISC doesn't work because it will simply be ignored on older PA-RISC kernels.*

However, it's already the case that several archs have their own `sys_mmap` *entry points. Those archs that can't follow the common scheme (only PA-RISC it seems) will need to add a new* `mmap` *syscall. I* *think that's a reasonable tradeoff to allow every other architecture to add this support with their existing* `mmap` *syscall paths.*

Helge Deller objected to this plan, saying, "I don't want other architectures to suffer just because of PA-RISC. But adding a new syscall just for usage on PA-RISC won't work either, because nobody will add code to call it then."

Helge proposed breaking the ABI for PA-RISC in this case. This, he said, would allow a proper fix without having to do any major contortions. He added that there were not a lot of PA-RISC users, anyway, and that most of them updated their kernels regularly, because of all the recent fixes that had gone into the code.

However, Dan replied, "The whole point is to avoid an ABI regression and the chance for false positive results. We're immediately stuck if some application was expecting `0x8` to be ignored, or conversely, an application that absolutely needs to rely on `MAP_SYNC/MAP_DIRECT` semantics assumes the wrong result on a PA-RISC kernel where they are ignored."

The debate petered out at that point, and it's not yet clear what solution they'll ultimately use for `mmap()`. I found it interesting to watch how each proposed solution ran into its own stumbling blocks. New system calls take time to trickle down. The proposed workaround would only work for certain cases – or if it could be fixed to work for all cases, there was still a single holdout architecture that wouldn't go along with it. Choosing to break ABI compatibility for that one case was a nonstarter because the whole point was not to do that in the first place.

Of course, many parts of the kernel would love the opportunity to break ABI compatibility, but this seems to be one of the most sacrosanct elements of the kernel. Almost the only thing capable of trumping ABI compatibility is a security hole. Security trumps all. But one of these days, it would be wonderful to have a special ABI-breaking kernel release, where every part of the kernel is free to break the ABI for just that one time. Oh the feeding frenzy! Oh the carnage! And afterward … oh the regret! Oh the recrimination! It would be glorious.

Tracing RAM Usage in OOM Conditions

Yang Shi was annoyed when his kernel ran out of memory, and the out-of-memory (OOM) killer couldn't find a process to kill to stave off the kernel panic. If there's no process to kill, how could the system be out of memory? It turned out that the system had put all its memory into unreclaimable slabs. Yang posted a patch to add a new `-U` option to the `slabinfo` program to cause that program to output only data about unreclaimable slabs. This, he said, would at least allow the user to troubleshoot the problem and possibly find a way to deal with it properly.

Michal Hocko had no objection to the patch going into the kernel, but he did note that Yang's code might produce a metric ton of output, on top of the already highly verbose OOM killer report. As such, he suggested leaving the feature disabled by default.

Yang disagreed, saying the output would only be produced in the event of catastrophic failure and not as a part of normal operations. He added that the code could easily add a file to the proc filesystem (`procfs`), to control the amount of output, even under that circumstance.

Michal said he didn't care that much about this particular issue, because "most OOM reports I have seen were simply user space pinned memory." He was fine leaving the output more verbose rather than less.

They kept talking, and eventually they seemed to reach a compromise when Yang suggested, "Maybe we can set a unreclaimable slab/total mem ratio. For example, when unreclaimable slab size >=50% total memory size, then we print out slab stats in OOM? And, the ratio might be adjustable in `/proc`."

This made sense to Michal, and the thread came to an end. ■■■

Customizing the UEFI boot process

Smart Startup

The traditional BIOS dates back decades and has not been able to keep up with the rapid development of PCs and laptops. Its powerful successor UEFI takes over its tasks and provides more features, more convenience, and better security. *By Maik Brüggemann and Ralf Spenneberg*

Since 1981, PCs have booted with the help of the Basic Input/Output System (BIOS). Over the years, different manufacturers have continued to expand the BIOS firmware system, but even after all this time, the BIOS system is difficult to adjust and still does not support 64-bit operation.

Back in the late 1992, Intel launched a new initiative to replace the venerable BIOS system with a 64-bit alternative. This Extensible Firmware Interface (EFI) initiative became the Unified EFI (UEFI) Forum in 2005. In addition to Intel, AMD, Microsoft, HP, and other prominent vendors contributed to the UEFI project.

UEFI describes the interface between a computer's firmware and operating system. Like BIOS, the UEFI implementation initializes the hardware components to prepare the launch of the operating system. However, UEFI natively supports 64-bit architectures, enables graphical user interfaces, and can protect against malicious software by only allowing signed operating systems (a feature that has come to be known as Secure Boot).

If you are using a Linux system that is preconfigured with support for UEFI and Secure Boot, you don't have to think about it very much unless you need to customize or troubleshoot the boot process. But if you want to dig deeper, you will find that UEFI is quite versatile and powerful. This article takes a closer look at the UEFI boot process and shows how you can use UEFI to add a custom application that runs independently of the operating system.

Boot Phases

UEFI firmware is modular and extensible. When it wakes up, the firmware runs through various stages in a strict chronological order (shown in Figure 1). The first phase is known as the Security (SEC) phase. However, the name is unfortunate because very few security-relevant actions occur in this first phase – for instance, the UEFI firmware does not use this phase to check signatures. Essentially, the SEC phase initializes the CPU with hardware-specific code. The CPU loads the code directly from flash memory. RAM is not yet available at this time. Instead, the UEFI firmware uses the temporary memory of the CPU cache as RAM. The SEC phase transfers phase size and location of this temporary storage, as well as optional information about the CPU, to the Pre-EFI Initialization (PEI) phase.

The PEI phase initializes the main memory, as well as hardware components that are required for the next phases. PEI Modules (PEIMs) provide APIs (PEIM-to-PEIM interface, PPI). The PEIM dispatcher is responsible for loading and running the PEIMs. It investigates the PEIMs' dependencies, loads them into the now available main memory, and runs them in a specific order.

In addition to initializing basic hardware components, the UEFI firmware also reads a structured collection of codes and data, known as the firmware volume location. This data is typically stored in flash memory and includes device drivers for the next phase. After the dispatcher has run most of the necessary PEIMs, it finally looks for the Driver Execution Environment (DXE) initial program load (IPL) PEIM. This last PEIM then transfers the execution to the following DXE phase.

In the DXE phase, most of the necessary system initialization then follows. Like the PEI phase, the DXE phase also has a dispatcher, which loads DXE drivers from the firmware volume location and runs them in the desired order. The DXE drivers initialize all required hardware components and register or use protocols.

Protocols provide the text output to the console or access to the PCI devices. If a protocol is only available before the operating system launches, it is known as a boot service. A protocol that you can access them while the operating system is running is called a run-time service.

After the dispatcher has loaded all drivers, it continues with the Boot Device Selection (BDS) phase. The BDS phase runs the UEFI Boot Manager. NVRAM variables tell the Boot Manager which UEFI applications must be launched. The operating system can influence these variables through a run-time service and determine the UEFI applications that must run. The hardware manufacturers frequently provide some UEFI applications, such as diagnostics or recovery applications. You can add additional applications yourself and integrate them in the boot process.

The OS bootloader is a special application. The system typically loads it and launches it from the EFI system partition (`/boot/efi`). By launching the OS bootloader, the boot process transitions to the Transient System Load (TSL) phase. If the system uses Secure Boot, its signature would be tested in this phase. Unsigned code will not run. The TSL phase ends on calling the `ExitBootServices()` function. This call thoroughly cleans up the memory and only leaves the specially marked interfaces in the memory that need to be available at operating system run time.

After calling `ExitBootServices()`, the system is in the Run Time (RT) phase; the operating system is running and can in turn access the UEFI run-time services. After the RT phase, the system enters the After Life (AL) phase with the termination of the operating system. This is intended for an orderly shutdown. However, it is hardly used.

UEFI Run-Time Services and Variables

UEFI run-time services are UEFI functions that can be used at operating system run time. For this purpose, a UEFI driver registers as a run-time service and creates an entry in the run-time services table. During the boot process, the operating system is given this table. The table gives the operating system access to the real-time clock; the OS can trigger a platform reset, access variables in the NVRAM, or initiate a firmware update.

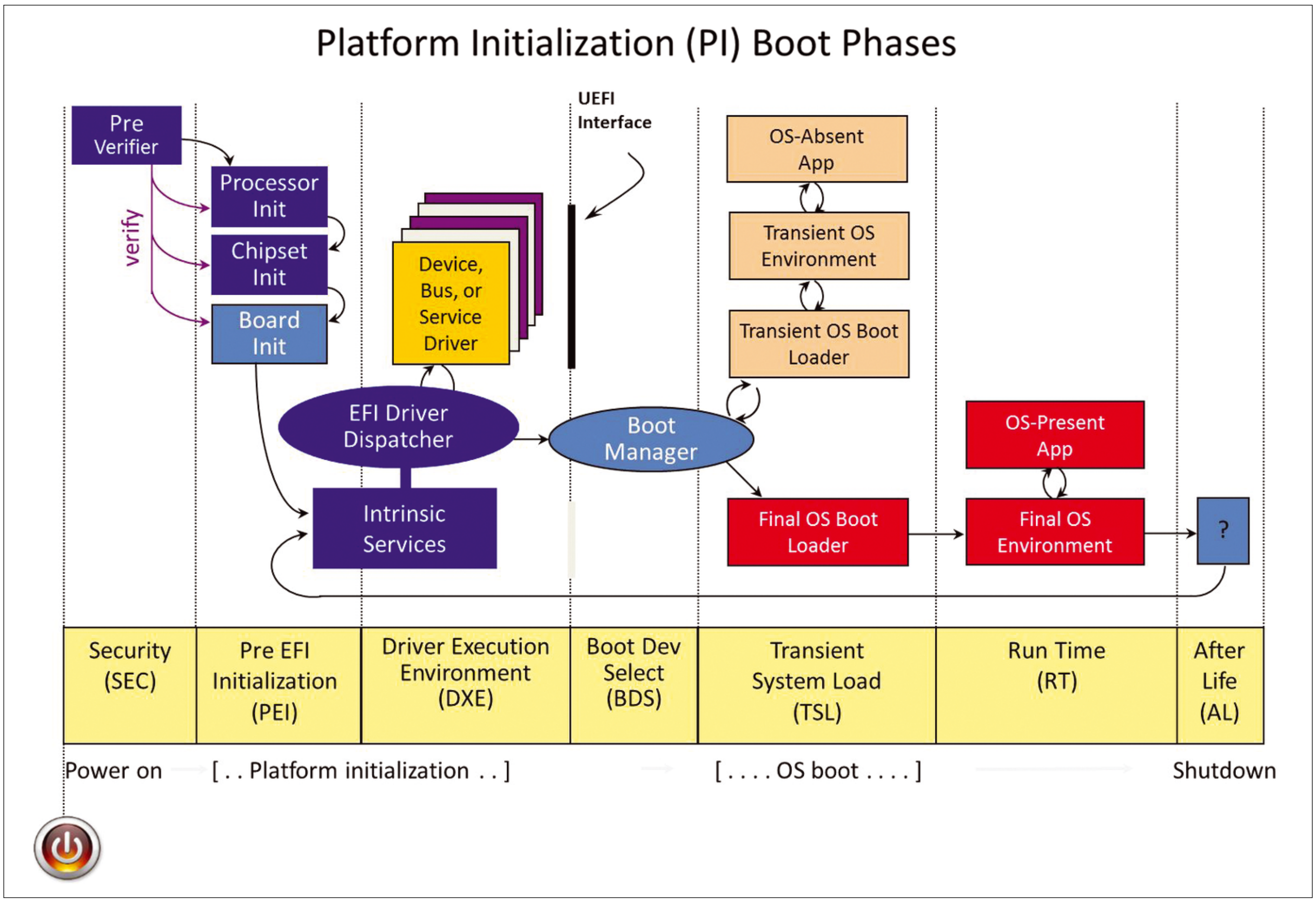


Figure 1: The UEFI boot process runs through a series of phases.

While most run-time services are only accessible to kernel drives on a Linux system, access to the NVRAM variables and the firmware update function are exported to userspace by the Linux kernel. The Linux kernel makes UEFI variables accessible in userspace via the UEFI variable filesystem (`efivarfs`). Linux distributions launched by UEFI integrate the filesystem under `/sys/firmware/efi/efivars`.

You can list all available variables via:

```
ls /sys/firmware/efi/efivars
```

You can read UEFI variables like normal files. The following command outputs the contents of a variable:

```
cat /sys/firmware/efi/efivars/variable_name
```

on the console. However, the stored information is available in a machine-readable format; the output of the `cat` command is often fairly cryptic.

You can write or delete these variables as you would a normal file. But deleting a variable can be fraught with danger. Previously, a `rm-rf /` would result in not just all the files on the hard drive being deleted but all UEFI variables too. However, some buggy UEFI implementations expected certain variables. If these variables were deleted, the computer no longer launched and you had to send the motherboard to be repaired.

Current Linux kernels therefore protect most of the variables with the immutable flag:

```
lsattr PKDefault-8be4df61-93ca-11d2-aa0d-00e098032b8c
----i-------------- PKDefault-8be4df61-93ca-11d2-aa0d-00e098032b8
```

Thus, accidental deletion or modification is ruled out. In practice, you cannot directly access the variable via commands such as `cat`. Instead, you need to use special programs such as `efibootmgr` that let you edit the group of boot variables.

The BDS phase is controlled by the boot variables. A variable starting with `Boot000` represents an entry in the UEFI Boot Manager. The value of the variable specifies the location of a UEFI application that is bootable during the BDS phase. The `BootOrder-GUID` variable determines the order in which the applications must be launched. The `efibootmgr` program manages these variables at the command line. The command in Listing 1 displays the current configuration of the BDS phase.

Listing 1 shows two entries. The entry known as `debian` is right at the front in the boot order and is therefore run first. The entry points to the EFI system partition (`/boot/efi`), on which

Listing 1: BDS Phase Configuration

```
# efibootmgr --verbose
BootCurrent: 0003
Timeout: 1 seconds
BootOrder: 0003.0002
Boot0002 Hard Drive BBS(HD,,0x0)
Boot0003* debian HD(1, GPT,
   cf2d93bb-3cd3-4903-9b7ccbfc697f7aae,  0x800,
   0x1dc800)/File(\EFI\debian\grubx64.efi)..BO
```

GRUB is stored. The system first runs the UEFI implementation of GRUB in the BDS phase. GRUB, in turn, launches the Linux kernel. You can add new boot variables or change the boot order with the `efibootmgr` command.

While the boot variables are freely adjustable, write access to other variables is restricted. They include, among others, the `PK`, `KEK`, `DB`, and `DBX` variables. These variables store the public keys for signature verification within the scope of Secure Boot.

If an attacker can change these variables, they can also bypass or switch off Secure Boot. Therefore, only signed data can be stored in these variables. The UEFI implementation verifies the signature of the transferred public key and only allows the write operation if the signature is trustworthy.

Update Capsule

Another interesting UEFI run-time service bears the name Update Capsule. The operating system can use this service to transfer data blocks to the UEFI firmware. The operating system stores the data in memory and shares the location with firmware via the Update Capsule service. A system reset is then triggered. The computer restarts. UEFI now accesses the provisioned data block.

The service is used primarily to run updates of the UEFI firmware. The advantage of this procedure is that the firmware can handle the actual update process. You don't need any proprietary tools, which are usually only available on Windows.

The standardized UEFI interface makes firmware updates independent of the operating system. For example, Microsoft uses this process to provision surface tablets with new firmware via Windows Update. Apple also uses this process for certain MacBooks. At present, Linux users can only use the update function in conjunction with a few systems. In addition to Microsoft and Apple, Dell is the only big manufacturer that offers firmware updates in the Update Capsule format.

Like the UEFI variables, the Linux kernel exports the update interface to the filesystem. The `/sys/firmware/efi/esrt` directory contains the EFI System Resource Table (ESRT). The kernel generates an input for each device that can be updated via the Update Capsule interface. The `fwupdate` tool lists these devices. The tool is installed using:

```
sudo apt-get install fwupdate
```

The command generates a list of all Update Capsule-capable devices in the system:

```
fwupdate --list
```

If there is an Update Capsule-compatible firmware for the devices listed, it can be installed as follows:

```
fwupdate --apply=guid-of-hardware firmware.cap
```

Unfortunately, as previously mentioned, only a few manufacturers support this functionality to date. However, it shows what advantages the standardized UEFI interface has over proprietary BIOS implementations.

Custom Code

The EFI Developer Kit II (EDK II) is used for developing UEFI applications. It is the current reference implementation of the UEFI specification and is used by many manufacturers as the basis for their own firmware. To avoid developers having to constantly reboot their computers to test their own applications, it makes sense to work in Qemu. EDK II includes everything you need to generate complete UEFI firmware and relaunch under Qemu.

First, you need to install some dependencies. On Ubuntu 4.16, use the following command:

```
sudo apt-get install build-essential uuid-dev ⮐
iasl git gcc-5 nasm
```

The easiest way to pick up the last stable release of EDK II is via GitHub:

```
mkdir ~/src
cd ~/src
git clone https://github.com/tianocore/edk2.git vUDK2017
```

After downloading, first compile the base tools:

```
cd ~/src/vUDK2017
make -C BaseTools
```

If everything worked, the EDK is ready for use. The source code is organized in packages. The *MdeModulePkg* and *MdePkg* packages are important because they implement the actual UEFI Specification. The code is largely architecture independent. To generate firmware capable of running on a specific platform, the firmware requires platform-specific code. The EDK provides this code for Qemu. The *Ovmf* package internally builds on the *Mde* packages and adds the Qemu specific parts.

In order for the *Ovmf* package to create an EDK, you need to edit the `~/src/vUDK2017/Conf/target.txt` file. Specify the *Ovmf* package as the `ACTIVE_PLATFORM`:

```
ACTIVE_PLATFORM = OvmfPkg/OvmfPkgX64.dsc
```

In addition, you need to adapt the architecture and the compiler tool chain you will be using. On an x64 Ubuntu 04.16 system, use the following values:

```
TARGET_ARCH= X64
TOOL_CHAIN_TAG= GCC5
```

Once the target platform is set, launch the first build process:

```
source edksetup.sh
build
```

The build takes a few minutes and stores the complete firmware in the `~/src/vUDK2017/Build/OvmfX64/DEBUG_GCC5/FV` directory.

To test the UEFI firmware using Qemu, copy the required firmware files into a new directory (Listing 2). Using the following command, you can launch a virtual machine and run the firmware:

Listing 2: Copying Firmware Files

```
mkdir ~/efibin
cd ~/efibin
cp ~/src/vUDK2017/Build/OvmfX64/DEBUG_GCC5/FV/OVMF_CODE.fd .
cp ~/src/vUDK2017/Build/OvmfX64/DEBUG_GCC5/FV/OVMF_VARS.fd .
```

```
Qemu-system-x86_64 -drive if=pflash,format=raw,⮐
file=OVMF_CODE.fd -drive if=pflash,format=raw,file=OVMF_VARS.fd
```

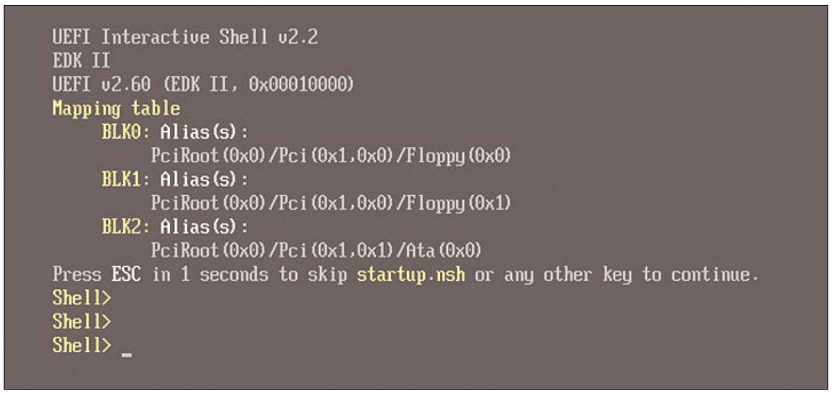
If Qemu displays a logo with the words `TianoCore` for a short time and then launches the UEFI shell (Figure 2), the whole process was successful.

For the first attempts with your own UEFI applications, you can use the *Hello World* application from the `MdeModule` as a template. The correct file can be found in `MdeModulePkg/Application/HelloWorld/HelloWorld.c`. This file, which is shown in Listing 3, was expanded and now implements a simple variant of the Hangman game. The user has to guess the word `penguin` and thus quit the application.

The entry point of the application is the `UefiMain` function (Line 8). `UefiMain` has two transfer arguments. `ImageHandle` is a handle for the process itself and not interesting for the *Hello World* application. However, the second argument `SystemTable` is a central data structure of the UEFI environment. `SystemTable` now provides the application with access to protocols that have been previously registered in the DXE phase.

In Line 21, the program uses the Simple Text Output Protocol via `SystemTable`. The protocol has a function known as `ClearScreen`. As the name suggests, this call clears the output console. This is followed by some output implemented by the print function. Under the hood, the print function also uses the Simple Text Output protocol. Line 32 relies on the `WaitForEvent` service. With its help, the program waits for the occurrence of certain events, in this case for a key press (`WaitForKey`). After the event occurs, the `ReadKeyStroke` function returns the pressed key.

The following `for` loop checks whether the character matches one of the expected characters. The entire process is repeated by the `do-while` loop until the user has guessed all the letters correctly. The program then outputs the word you are looking for and then waits for a subsequent key press before it quits.



```
UEFI Interactive Shell v2.2
EDK II
UEFI v2.60 (EDK II, 0x00010000)
Mapping table
      BLK0: Alias(s):
          PciRoot(0x0)/Pci(0x1,0x0)/Floppy(0x0)
      BLK1: Alias(s):
          PciRoot(0x0)/Pci(0x1,0x0)/Floppy(0x1)
      BLK2: Alias(s):
          PciRoot(0x0)/Pci(0x1,0x1)/Ata(0x0)
Press ESC in 1 seconds to skip startup.nsh or any other key to continue.
Shell>
Shell>
Shell> _
```

Figure 2: The UEFI shell is launched.

Listing 3: Hello World!

```
#include <Uefi.h>
#include <Library/PcdLib.h>
#include <Library/UefiLib.h>
#include <Library/UefiApplicationEntryPoint.h>

EFI_STATUS
EFIAPI
UefiMain (
 IN EFI_HANDLEImageHandle,
 IN EFI_SYSTEM_TABLE *SystemTable
 )
{
        UINTN Index;
        EFI_INPUT_KEY Key;
        CHAR16 *result = L"pinguin";
        CHAR16 *state = L"_______";
        UINTN Attempt = 0;

        do {
                Attempt++;
                SystemTable->ConOut->ClearScreen(SystemTable->ConOut);

                Print(L"Welcome to Hangman!\n");
                Print(L"Attempt: %d\n\n\n", Attempt);

                // print found characters
                for(UINTN i=0; i<StrLen(state); i++) {
                        Print(L"%c ", state[i]);
                }

                // read characters from keyboard
                SystemTable->BootServices->WaitForEvent(1,
                  &SystemTable->ConIn->WaitForKey, &Index);
                SystemTable->ConIn->ReadKeyStroke(SystemTable->ConIn, &Key);

                // check if pressed key is in result
                for(UINTN i=0; i<StrLen(result); i++) {
                        if (Key.UnicodeChar == result[i]) {
                                state[i] = result[i];
                        }
                }
        } while(StrCmp(result, state) != 0); // repeat until all characters are
          found

        // print final result
        SystemTable->ConOut->SetCursorPosition(SystemTable->ConOut, 0 , 4);
        for(UINTN i=0; i<StrLen(state); i++) {
                        Print(L"%c ", state[i]);
        }
        // wait for key
        Print(L"\n\nCongratulations!\nPress any key to continue...\n");
        SystemTable->BootServices->WaitForEvent (1,
          &SystemTable->ConIn->WaitForKey,
        &Index);

        return EFI_SUCCESS;
}
```

You have to extend the *Ovmf* package in order for the Hangman program to be included in the next build. In the `~/src/vUDK2017/OvmfPkg/OvmfPkgX64.dsc` file, add the `[Components]` section and a reference to the *Hello World* application:

```
[Components]
MdeModulePkg/Application/↩
HelloWorld/HelloWorld.inf
```

The `HelloWorld.inf` file describes the application and instructs the EDK to automatically generate makefiles. In the base directory of the EDK, launch the build process for the *Hello World* application:

```
source edksetup.sh
build
```

The build stored the generated UEFI binary `HelloWorld.efi` in the `~/src/Build/OvmfX64/DEBUG_GCC5/X64` directory. To test the application, copy it to the `efibin` directory and relaunch Qemu:

```
cd ~/efibin
cp ~/src/Build/OvmfX64/↩
DEBUG_GCC5/X64/HelloWorld.efi .
```

When launching, extend the command to launch the Qemu system by the `hda` parameter. This parameter integrates the `efibin` directory into the virtual machine as a FAT-formatted hard drive. The full command is:

```
Qemu-system-x86_64 -drive ↩
if=pflash,format=raw,↩
file=OVMF_CODE.fd -drive ↩
if=pflash,format=raw,↩
file=OVMF_VARS.fd -hda fat:~/efibin
```

UEFI directly supports the FAT filesystem and therefore has access to the *Hello World* application. After launching the UEFI Shell, enter *Hello World* to launch the program (Figure 3).

After completing the application development, you can run the binary file generated in the UEFI environment of the motherboard. For this purpose, simply copy them to the EFI system partition of your computer:

```
sudo cp ~/efibin/HelloWorld.efi ↩
/boot/efi/
```

Similar to the Qemu environment, you can manually launch the application in the UEFI implementation of the motherboard using the UEFI Shell. After resetting the computer, you then have to press one of the function keys to interrupt the normal boot process and instead launch the UEFI Shell. Which key has to be pressed depends on the UEFI implementation of the motherboard. Once you are in the UEFI Shell, you can simply call the application by typing *HelloWorld*.

Alternatively, you could set up an automatic launch of the application. You must configure the UEFI Boot Manager via the UEFI variables. The easiest way to do this is with the `efibootmgr` program. The program generates new boot variables in the `efivar` filesystem and thus an entry in the UEFI Boot Manager.

However, before this step, you should ensure that a recovery CD is ready, with which you can repair a potential misconfiguration. To generate a new entry, transfer the `create` argument to the `Efibootmgr`. A label gives the entry an intuitive name; the `loader` parameter displays the path to the UEFI application:

```
efibootmgr --create --label ⮒
"Hangman" --loader HelloWorld.efi
```

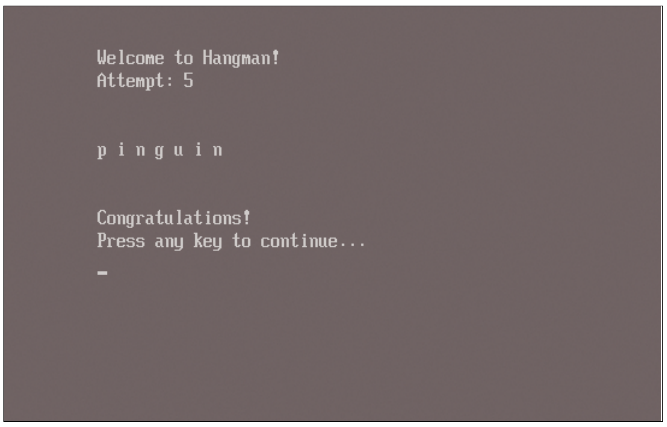
If successful, the system responds with:

```
BootCurrent: 0000
Timeout: 2 seconds
BootOrder: 0000.0001
Boot0000* debian
Boot0001* Hangman
```

Currently, the boot sequence still defines that the entry with the `debian` label will start. The `efibootmgr --bootnext 0001` command changes this order for the next boot to the entry with the number 0001. If you now reboot the computer, the UEFI firmware runs the *Hello World* application once only.

Conclusion

This example basically shows how to develop and run UEFI programs. You are not restricted to simple text input and output. UEFI allows you to introduce a graphical user interface, access a TCP/IP network, or access a USB stick. The versatile UEFI environment lets you run even complex applications independently of the operating system. ■■■



```
Welcome to Hangman!
Attempt: 5


p i n g u i n


Congratulations!
Press any key to continue...

_
```

Figure 3: The Hangman word-guessing game with UEFI control.

Linux control over Secure Boot

Better Boots

The Shim bootloader lets Linux users regain some control over the Secure Boot process. *By Eva-Katharina Kunst and Jürgen Quade*

The UEFI Secure Boot feature ensures that only software with a valid digital signature launches on a computer. UEFI searches for a bootloader on the SSD or hard disk, verifies the digital signature from one of the certificates stored with UEFI, and, if the digital signature is valid, loads and activates the code.

The bootloader searches for the operating system, verifies the digital signature, and launches the operating system. Once the operating system is launched, it only loads kernel modules and drivers that have a valid digital signature.

Authors

Eva-Katharina Kunst has been a Linux fan since the beginnings of open source. **Jürgen Quade** is a professor at Niederrhein University in Krefeld, Germany. The fourth edition of their co-authored book *Linux-Treiber entwickeln* [Developing Linux Drivers] was published late in 2015.

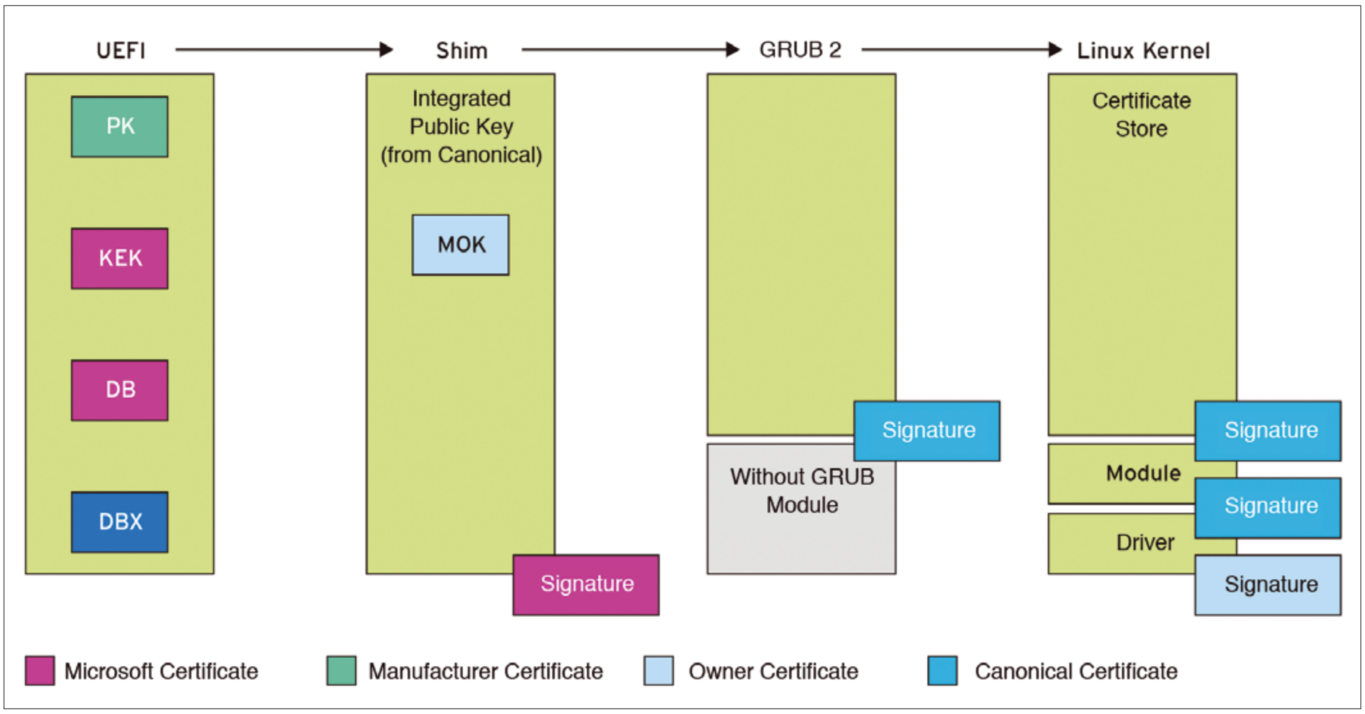


Figure 1: The certificate store used in a Secure Boot-protected computer. DBX is the forbidden signatures database.

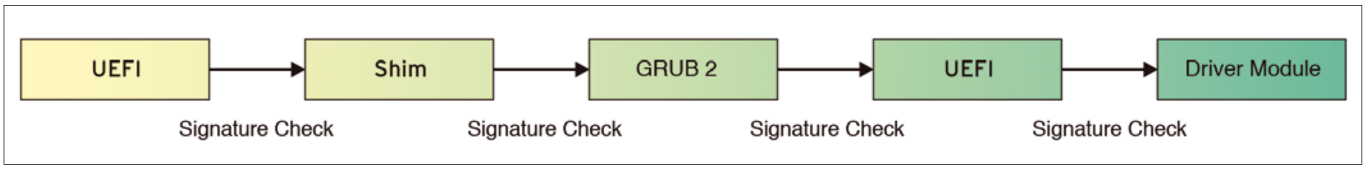


Figure 2: The Ubuntu Secure Boot process runs through a series of stations.

The idea is that, if all components only load code from trustworthy sources, it is much more difficult for malware authors hiding away in the grubby corners of the Internet to smuggle their software into the boot process.

One problem with UEFI Secure Boot for Linux developers and users is the control that Microsoft maintains over the system. Microsoft's market power means that every hardware manufacturer burns its own certificate as a Platform Key (PK), and then the Microsoft certificate is securely deposited into the Key Exchange Key (KEK) database and (authorized) database (DB) key on the motherboard (Figure 1). Therefore, x86 PCs and laptops initially only boot software with a signature by the grace of Microsoft.

Shim: An Alternative Approach

The thought of the Linux kernel needing a digital signature from Microsoft was too much for many Linux users, so Matthew Garrett created a program called the Shim bootloader, an open source alternative that integrates its own certificates. Ubuntu, Red Hat, SUSE, and Debian generate their own versions of Shim that include certificates issued by their companies.

Verisign/Symantec digitally signs the bootloader in Microsoft's stead so that the UEFI firmware will load Shim. Once Shim is loaded, it operates independently of the Microsoft verification chain. Shim has built-in certificate management that lets the owner of the computer store certificates called machine owner keys (MOKs).

Recovering Autonomy

Shim lets large distributors such as Ubuntu, SUSE, and Red Hat win back control of hardware. Using the Canonical certificate stored in Shim, for instance, Ubuntu distros sign the GRUB 2 bootloader. The firmware boots Shim, Shim boots GRUB 2, and GRUB 2 boots the operating system (Figure 2).

The user doesn't notice Secure Boot at first. For example, if you install Ubuntu on a computer with Secure Boot enabled, the installation routine places the signed Shim bootloader and GRUB 2 on the SSD or hard disk and installs the digitally signed kernel, along with verifiable modules and drivers. If Secure Boot is not enabled, the operating system installer copies the various components onto the computer without a digital signature.

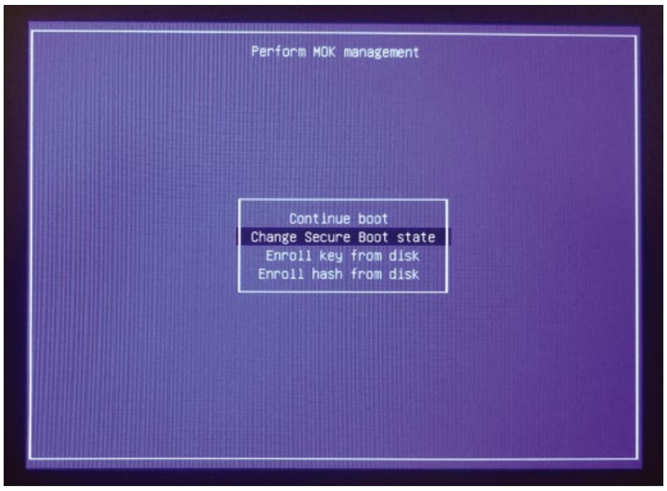
Switching Off

If you try to install VirtualBox on a Secure Boot Linux machine, the host computer might object and refuse to load the necessary kernel module because it has no valid digital signature. This behavior occurs in all third-party packages that provide their own modules or drivers. With physical access to the computer, you can inelegantly deactivate the verification of digital signatures by the Linux kernel with Shim by typing the command:

```
sudo mokutil --disable-validation
```

The `mokutil` tool requires you to enter a one-time password. After that, `mokutil` does not deactivate the check itself, but it sets up the Shim bootloader so that it asks for the password at the next reboot and performs the desired configuration after the correct password is input. After a reboot, Shim expects you to enter the one-time password within a short time frame.

In Shim, select *Change Secure Boot state* in the selection box (Figure 3). After entering the previously defined one-



Perform MOK management

Continue boot
Change Secure Boot state
Enroll key from disk
Enroll hash from disk

Figure 3: Quick and insecure: In Shim, the signature check can be disabled.

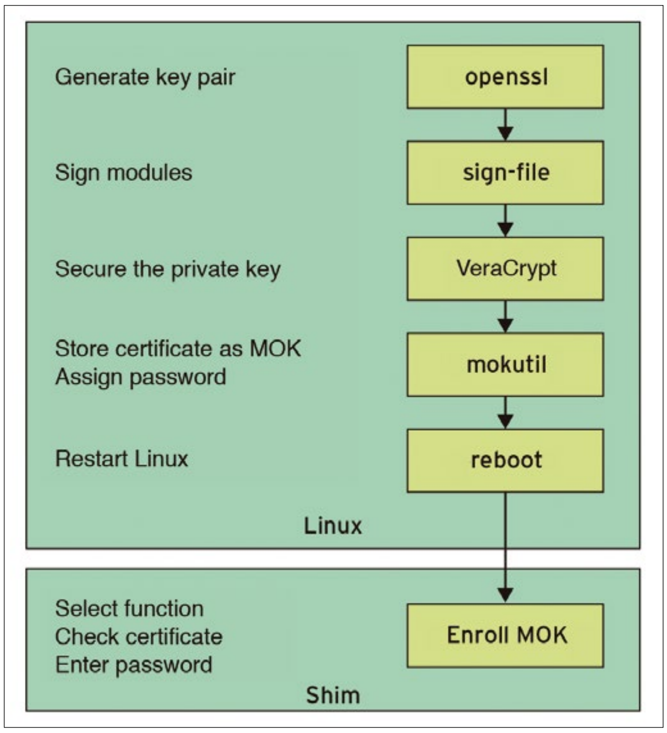


Figure 4: If you have physical access to the computer, you can integrate your own certificates. The not-particularly-intuitive individual steps toward this objective are summarized here.

time password, the compulsory verification of digital signatures is deactivated. However, deactivating verification means that you lose the protection offered by Secure Boot. (You can also switch off Secure Boot directly in the UEFI setup.)

Creating Your Own Certificate

A better way to add some flexibility to your Secure Boot configuration is to store your own certificates in the kernel and sign the modules yourself. The problem is that the new key must be signed with a key whose signature already exists in the certificate store, and only the key of the distributor – Canonical – is in the certificate store. Because the private key is (one hopes) known only to Canonical, you can't make modifications. To solve this problem, you must generate the kernel yourself and store the certificate used in the Shim bootloader as a MOK.

Figure 4 shows the procedure. First, you have to generate the new key pair. Second, the private key signs your or third-party modules. The third step is to secure the private key, and the fourth step is to give Shim the public key as the MOK. When you reboot, the Shim certificate manager pops up; use the certificate manager to validate and import the certificate. After the next startup, Linux can use the signed modules.

To generate the required key pair at the command line, use the following long command:

Keys, Passwords, and Certificates

Even the experts sometimes use the terms key, password, and certificate incorrectly. A key is a byte sequence for encrypting data. If your encryption algorithm uses block encryption, the key has a length of one block. In the case of character-by-character encryption, the ideal key is as long as the character string you are encrypting; otherwise, the key is used repeatedly.

Password or Key. A password is a character string that produces a key. For example, if you generate an infinitely long key using a pseudo-random number generator, you can use the initialization value of the random number generator as a password.

In symmetric encryption, a single key is used both to encrypt and to decrypt. The key is a shared secret – a secret known to all encrypters and decrypters. It is often necessary to transport the key to the receiver, which creates the risk that a third party will read the key.

Asymmetric encryption, on the other hand, relies on a mathematically coherent key pair. The private key is used for decryption and digital signing. A third party requires the public key for encrypting and for verifying a digital signature.

Using symmetric encryption, you can digitally sign data, and you can encrypt specifically for individual recipients. The public key is worthy of its name because it does not have to be kept locked up and can be transported safely.

Key with a Fob. It makes sense to add information about the owner to the public key and sign it all digitally so that it is tamper-proof. This digitally signed combination of metadata (name, company, address) plus the public key is known as a certificate (Figure 5).

Whether the public key comes alone or as a certificate, the owner needs to protect it because of its importance. It is quite common to encrypt it symmetrically and secure it with a password.

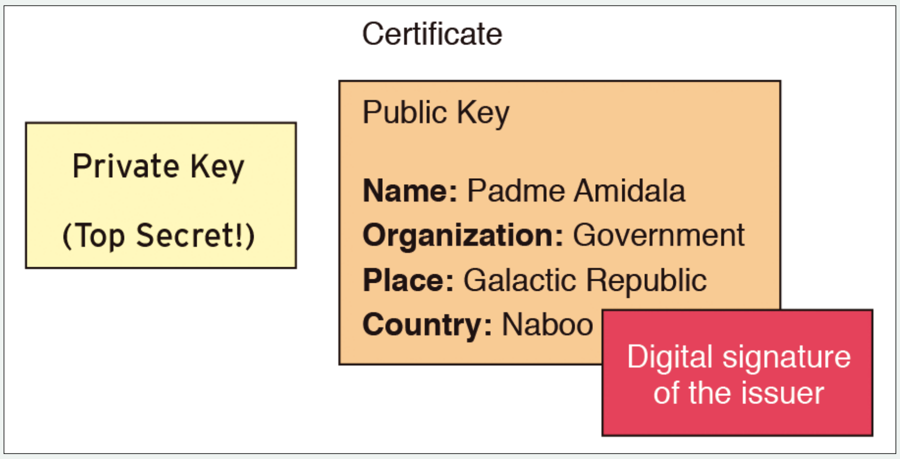


Figure 5: A certificate comprises a public key, metadata about the owner of the private key, and a digital signature.

```
openssl req -new -x509 -newkey rsa:2048 ⮐
            -keyout ownerkey.priv -outform DER ⮐
            -out ownerkey.der -nodes -days 36500 ⮐
            -subj "/CN=Machine Owner Key of My Company/
```

Now in the current directory are the private key (`ownerkey.priv`) and the certificate with the public key (i.e., the owner certificate `ownerkey.der`). (See also the "Keys, Passwords, and Certificates" box.)

You need to keep the private key safe, because anyone who can read it can use it for digital signing. It makes sense to move the private key onto an external drive, such as a USB stick. Additionally, we recommend that you encrypt the USB stick using an encryption tool such as VeraCrypt [1].

Self-Signing Modules

Unsigned modules must be digitally signed, for which you can use the `sign-file` command of the kernel build system. The command is in the kernel sources, and you can call it as follows:

```
sudo /lib/modules/$(uname -r)/build/scripts/sign-file ⮐
  sha256 ./ownerkey.priv ./ownerkey.der <Name_of_the_Module>
```

Using the script from Listing 1, you can sign all VirtualBox modules. The `$(modinfo -n vboxdrv)` loop returns the path to the VirtualBox module (e.g., to `vboxdrv.ko`). Alternatively, if you want to self-sign your kernel module, specify the path, including file name of the module.

Listing 1: Signing the VirtualBox Module

```
#!/bin/bash
for drivername in vboxdrv vboxnetflt vboxnetadp vboxpci
do
  sudo /lib/modules/$(uname -r)/build/scripts/sign-file
       sha256 ./ownerkey.priv ./ownerkey.der $(modinfo -n
       $drivername)
done
```

A simple command hands over the sample signature, which you do not need to protect against unauthorized access, to Shim's certificate manager:

```
sudo mokutil --import ownerkey.der
```

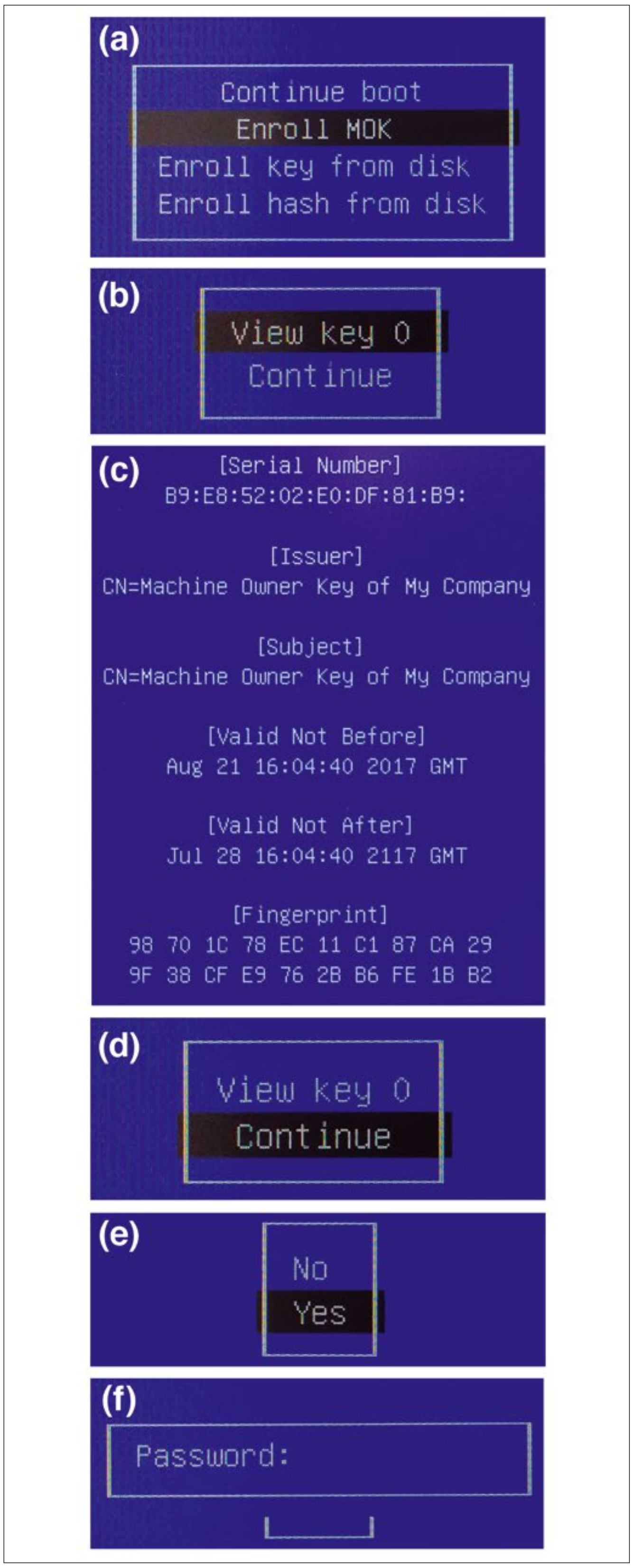
As usual, after a reboot, you have to enter a one-time password within a short time window to enable certificate management.

To activate the recently created certificate (Figure 6) in Shim, choose *Enroll MOK*. *View Key* checks the key, and the two steps after choosing the *Continue* menu item finally activate it.

Back on Linux again, the command

```
mokutil --list-enrolled
```

outputs a list of registered certificates, including self-generated certificates. The command to load the VirtualBox kernel module



```
[Serial Number]
B9:E8:52:02:E0:DF:81:B9:

[Issuer]
CN=Machine Owner Key of My Company

[Subject]
CN=Machine Owner Key of My Company

[Valid Not Before]
Aug 21 16:04:40 2017 GMT

[Valid Not After]
Jul 28 16:04:40 2117 GMT

[Fingerprint]
98 70 1C 78 EC 11 C1 87 CA 29
9F 38 CF E9 76 2B B6 FE 1B B2
```

Figure 6: (a) Shim lets you enable a self-generated certificate. The issuer of the digital certificate can tell at a glance (b) that the certificate (c) really is their own. (d) Choosing *Continue* starts activating the certificate for the boot system, (e) proceeds with a functional but fairly unnecessary security prompt, and (f) ends with the entry of the previously stipulated one-time password.

```
sudo modprobe vboxdrv
```

now runs successfully. However, be careful: With each update of the kernel, you have to digitally sign the new modules by again accessing the (one hopes) well-secured private key.

Incidentally, the command

```
sudo mokutil --delete ownerkey.der
```

tells the Shim certificate manager to remove a certificate from certificate management during the next reboot – as always under the protection of a one-time password. The menu item in Shim is *Delete MOK*.

More Secure Is Not 100% Secure

Installing the Shim UEFI bootloader and, with all the effort it involves, signing the kernel and its modules certainly makes the computer more secure. It is far more difficult for script kiddies and hackers to install and launch malicious software on the computer. However, you have no reason to celebrate when you look at, and into, the various certificate stores on the computer: As the manufacturer of the computer, Microsoft, Verisign, Symantec, and Canonical now have branches into the "secure" machine (Table 1).

First, it would be naive to believe that all employees in all companies are immune to the offer of doing someone a little favor in exchange for an impressive number of bitcoins. Second, the consequences of a letter from a national security organization, subject to confidentiality clauses and penalties, could force companies to make concessions in terms of data protection. In the end, only one Public Key Infrastructure (PKI) is secure, and that's the one that you operate yourself.

If you store your own key in UEFI, blacklist the existing keys, compile and self-sign the bootloader, compile and self-sign the kernel, and then do the same with all the modules and drivers, you can put considerably more trust in your system. If you have even higher security demands, the next step is to secure the individual applications running on your system. ■■■

Table 1: Who Controls What

Component	Sample Signature	Signatory
UEFI (PK, KEK, DB)	Manufacturer, Microsoft (2x)	–
Shim	Distributor (e.g., Ubuntu)	Microsoft (Verisign/ Symantec)
`grubx86.efi`	Distributor (e.g., Ubuntu)	Distributor
Linux Kernel	Distributor (e.g., Ubuntu)	Distributor

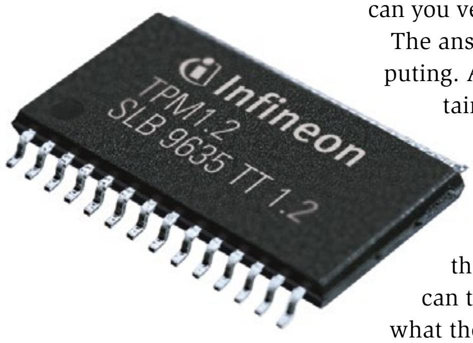
Info

[1] VeraCrypt (encrypts background memory): *https://veracrypt.codeplex.com*

Security with the Trusted Platform Module

Trusted Computing

The Trusted Platform Module on your computer's motherboard could lead to better security for your Linux system. *By Matthew Garrett*



The security of any operating system (OS) layer depends on the security of every layer below it. If the CPU can't be trusted to execute code correctly, there's no way to run secure software on that CPU. If the bootloader has been tampered with, you cannot trust the kernel that the bootloader boots. Secure Boot allows the firmware to validate a bootloader before executing it, but if the firmware itself has been backdoored, you have no way to verify that Secure Boot functioned correctly.

This problem seems insurmountable: You can only trust the OS to verify that the firmware is untampered with if the firmware itself has not been tampered with. How can you verify the state of a system without having to trust it first?

The answer lies in a set of technologies collectively referred to as Trusted Computing. A consortium of companies called the Trusted Computing Group [1] maintains the specifications related to Trusted Computing. At the heart of the Trusted Computing environment is a small hardware component called a Trusted Platform Module (TPM). The TPM is a chip connected by bus to the system motherboard, and sometimes it can be retrofitted as a module (Figure 1). TPMs are not fast or powerful – almost anything that can be done on a TPM can be done much faster on the CPU. Neither can the TPM see what's happening on the rest of the system; TPMs know only what the rest of the computer chooses to tell them.

However, it's how TPMs make use of this information that makes them so powerful. TPMs have a collection of registers called Platform Configuration Registers (PCRs). (See the box titled "Writing to the TPM.") When the system is reset, these registers are set to `0`. During the boot process, the system will generate cryptographic hashes of the boot components and pass these hashes to the TPM. Any modifications to the boot components will change the values of these hashes, and consequently change the values recorded by the TPM. This process is referred to as "measurement," and a boot that occurs in this way is a Measured Boot.

Writing to the TPM

When new values are given to the TPM, they are not written directly to the PCRs. Instead, the PCR will be set to a value determined by the combination of the PCR's current value and the new value given to it. If the component to be measured is the string *Hello world*, and the PCR's current value is *c72bf7f5a487b1e75819b5b1d1644ded23c10967*, the new value would be:

```
SHA1(c72bf7f5a487b1e75819b5b1d1644ded23c10967 || SHA1("Hello world"))
```

In other words, the `SHA1` (a cryptographic hash) of *Hello world* is appended to the current value of the PCR, and the PCR is then set to the `SHA1` of that concatenated value. This means that the order of measurements matters; measuring *Hello moon* and then measuring *Goodbye sun* will result in a different PCR value than measuring *Goodbye sun* and then *Hello moon*.

Different components are measured into different PCRs (Table 1). Each component is measured before it starts executing and, in turn, measures the next component before executing it, starting with the CPU measuring the firmware. If the firmware has been tampered with, PCR0 will be different. Even if the firmware then lies about the measurements of all later components (by pretending the values are correct even though they've been tampered with), the value of PCR0 will still be different. By looking at the PCR values and comparing them with expected values, you can tell whether any part of the boot process has been compromised.

But how do you look at those values? If the OS has been compromised, you can't trust the OS to give you the actual values stored in the TPM. You need more help from the TPM itself. This brings me to the next piece of functionality that TPMs provide – the ability to generate and store encryption keys.

When a TPM is initialized, it generates a key called the Storage Root Key (SRK). This key never leaves the TPM, and the OS has no way to access it. When an application asks the TPM to generate a key, the TPM does so and gives the public and private halves back to the application. However, before handing back the private half, the TPM encrypts it with the SRK. The private key can only be used by passing it back to the TPM, which will then decrypt it and store it in the TPM's internal RAM. Any material that's been encrypted with the public half of the key can only be decrypted by the TPM that can read the corresponding private key: If you have a TPM key pair and someone steals your hard drive, the thief will have no way of using those keys because they can only be decrypted by the TPM that generated them.

When an application asks for something to be encrypted with this key, it can specify a set of PCR values to be associated with the encrypted material. When it comes to decryption time, if those PCR values do not match, the TPM will refuse to decrypt the material. If you use the TPM to encrypt your disk encryption key, your machine will refuse to boot if any of the boot process is modified [2]. If you're using UEFI instead of BIOS, use the `verifier_tpm_module` branch of GRUB [3] instead of TrustedGRUB2.

When configured this way, a system that's been tampered with will simply refuse to boot, although it doesn't prevent an attacker with physical access from obtaining your passwords. They can simply remove your hard drive and replace it with one that boots normally even if the boot process has been modified. When you try to log in, it simply saves your password and sends it to the attacker. One approach for avoiding this is `tpmtotp` [4], which uses the time-based one-time password (TOTP) protocol used by websites for two-factor authentication (2FA). A secret is encrypted on the TPM and can only be decrypted if the PCR values match. The secret is then displayed in the form of a QR code that can be enrolled into a regular 2FA application. On boot, the system decrypts the secret and uses that in combination with the current time of day to display a six-digit value. The user compares this value to the value their phone is displaying. If they match, the user knows that the system is trustworthy and can enter the password safely. (See also the box titled "RAM Option.")

Of course, even legitimate updates of the boot components will change the values of the PCRs. Updating GRUB will

Table 1: PCRs

PCR	Measurement
PCR0	System firmware
PCR1	System firmware configuration
PCR2	Plugin card firmware
PCR3	Plugin card firmware configuration
PCR4	Partition table and bootloader
PCR5	Bootloader configuration
PCR6	Suspend and resume events
PCR7	Secure Boot configuration

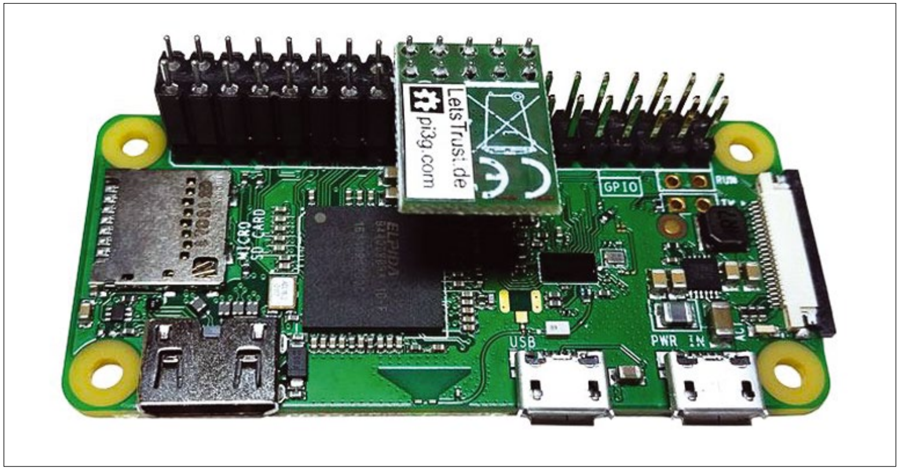


Figure 1: LetsTrust provides an attachable TPM 2.0 module for the Raspberry Pi.

RAM Option

An alternative approach is to use the TPM's small quantity of non-volatile RAM. You can configure the system so that the non-volatile RAM is only readable if the PCR values match – if they don't match, then attempting to read will fail. This approach allows a secret to be stored directly without having to worry about encryption. The downside to this technique is that the amount of `nvram` available is quite limited.

change the value of PCR4, for instance, which introduces a great deal of fragility into the system: Regular security updates may render a system unbootable. One way around this is to use the TPM in conjunction with Secure Boot. In this mode, only PCR7 is used. The firmware records the fact that Secure Boot is enabled into PCR7 and then measures each Secure Boot key that is used during the boot process. Modifying the bootloader or kernel would require the attacker to disable Secure Boot or to add new signing keys, so it would change the value of PCR7. Legitimate updates will be signed with the same key, so it won't change the value. This gives users much greater confidence that they can use TPMs safely without having to worry about their system breaking unexpectedly.

TPM encryption keys are not restricted to the boot process. It's possible to generate SSH keys that are tied to the TPM [5], giving enhanced protection. If someone manages to steal your private SSH key, they will still be unable to log in to your system unless they have access to your TPM to decrypt it.

TPMs and DRM

When Trusted Computing first appeared in the mid-2000s, people were greatly concerned that TPMs could be used to restrict access to websites or services if the user wasn't running the right version of Windows. The biggest concern was remote attestation. Each TPM has a key called an endorsement key (EK). A remote site can ask the OS to perform remote attestation, at which point the OS asks the TPM for the current PCR values and encrypts them with the EK. The remote site decrypts them, looks at the PCR values, and decides to grant access depending on the values provided.

Two big problems prevent this from being a real concern. First, for this to be viable, the remote site has to know that the EK is really from the TPM. More modern TPMs include an EK certificate that provides a chain of trust from the TPM to the TPM manufacturer, which means that a remote site can verify that the PCR values came from a TPM; however, they have no way of knowing which TPM unless the user has already registered this association in some way – which ties to the second problem: Nothing prevents a user from adding a second TPM to a system, programming the PCRs with "good" values, and then performing remote attestation with the second TPM.

Because of these problems (and other privacy concerns), remote attestation has never been used outside specific corporate or special case deployments, and this is unlikely to change in the future. Enabling your TPM does not put you at risk of having your freedoms infringed.

Using Your TPM

Using a TPM requires a few requirements. First, the TPM must be enabled; second, you must have an appropriate bootloader; and third, you must have the necessary user-space tools. The TPM can be enabled in your system firmware menu – consult your system documentation to find out how. For bootloaders, right now you'll need to use a modified version of GRUB 2 [6]. Finally, you'll need to install TrouSerS (the TPM daemon and library) and the *tpm-tools* package. You need to take ownership of your TPM by running:

```
sudo tpm_takeownership -z
```

If you get an error saying "The TPM target command has been disabled," this means that the TPM has already been enabled. If you didn't do this, you can reset the TPM by running

```
echo 14 >/sys/class/tpm/tpm0/ppi/request
```

as root and then rebooting. The firmware will then ask whether you want to clear the TPM. Follow the instructions given and try taking ownership again.

TPM 1.2 and TPM 2

Recent systems frequently ship with TPMs that conform to version 2 of the specification. Version 2 is an improvement in several ways, including support for more modern and secure hash algorithms. Unfortunately, support for TPM 2 devices in Linux is still maturing, and most of the software described in this article is only compatible with the older version, TPM 1.2.

Some systems ship with TPM 1.2 on the motherboard, but also implement TPM 2 in the form of an emulated TPM running on the Management Engine integrated into the CPU. In that case, check the system settings for a reference to "Intel PTT," "Intel Platform Trust Technology," or "Firmware TPM" and disable it.

Some vendors have taken an alternative approach and ship a hardware TPM on the motherboard that can be flashed between versions 1.2 and 2.0. Dell does this on the XPS 13. Check with your system vendor to find out whether your TPM can be downgraded to 1.2.

Conclusion

Making good use of a TPM under Linux still involves a lot of manual work, but the results can be extremely worthwhile. Plenty of work is going on to integrate this functionality into distributions so that users can benefit from these security features without having to be experts. With luck, everyone will be using TPMs in the near future. ■■■

Info

[1] Trusted Computing Group: *https://trustedcomputinggroup.org/*
[2] Using TPM for encryption: *https://github.com/fox-it/linux-luks-tpm-boot*
[3] GRUB: *https://github.com/mjg59/grub*
[4] tpmtotp: *https://github.com/mjg59/tpmtotp*
[5] Using the TPM chip to secure SSH keys: *https://github.com/ThomasHabets/simple-tpm-pk11*
[6] GRUB 2: *https://github.com/mjg59/grub/tree/verifier_tpm_module*

Automate data backup at the command line

AUTOMATIC BACKUP

Backing up data is an unpopular task that many users – and even some administrators – consider a chore, prompting us to take a look at some command-line automatic backup programs. *By Erik Bärwaldt*

Linux users have access to numerous backup tools. Administrators who like working with SSH appreciate that servers of any size and design can be backed up with command-line programs. However, the differences in terms of features are quite considerable (see Table 1 for an overview). Not every program is suitable for every application scenario. In this article, I investigate which tools work for which environments.

Table 1: Command-Line Backup Tools

	Attic	bup	Duplicity	rdiff-backup	rsnapshot
Local backup	Yes	Yes	Yes	Yes	Yes
Backup via SSH	Yes	Yes	Yes	Yes	Yes
Verification	Yes	Yes	Yes	Yes	Yes (logfile)
Encryption	Yes	Yes	Yes	No	No
Cloud services	No	No	Yes (Amazon, Rackspace)	No	No
Include/exclude directory	Yes	Yes	Yes	Yes	Yes
Time-controlled	Yes*	Yes*	Yes*	Yes*	Yes*
Front ends available	No	Yes	Yes	No	Yes
Incremental backups	Yes	Yes	Yes	Yes	No
Differential backups	No	No	No	No	No
Manual full backup	Yes	Yes	Yes	Yes	Yes
FUSE-mount possible	Yes	Yes	No	Yes	No

*Backups scheduled with the cron daemon.

Server vs. Desktop

Home users often store large volumes of data on their computers, similar in volume to those found on servers in small businesses. High-definition video collections, as well as audio files with lossless compression and photo folders, are real memory hogs. New data is often added, but once stored, the data hardly ever changes.

On the other hand, you will also often find small files (such as correspondence, tables, presentations, and databases) on server systems. These data collections are constantly changing through modifications, such as newly created records or added documents. Accordingly, backup strategies must take existing data resources into account to guarantee rapid reconstruction in the event of data loss.

Differential vs. Incremental

Administrators distinguish three backup strategies: full backup, differential backup, and incremental backup. The full backup, a copy of the existing data, is always the first backup in any plan – subsequent backups follow as differential or incremental backups. Whereas differential backups always save changes since the last full backup, incremental backups only save modifications relative to the last backup of any kind.

The differential backup procedure requires more space for individual backups, but in an emergency, only the full backup and the last backup will help you recover the entire database. Although the incremental method uses less disk space, all incremental backups need to be re-installed in the correct sequence during the restore starting from the full backup. If a small backup is missed, the database is no longer reconstructible.

Before selecting a command-line backup software, I recommend initially performing a careful analysis of your data collection and data growth, so you do not accidentally select a program that is unsuitable for your specific IT environment.

Differential backup strategies are more likely to be used for databases that have relatively few large files and moderate regular modifications, whereas incremental backups are better suited to typical office environments. Regardless of which you choose, you should always run at least one full backup per week.

Desktop users who want to back up their own databases without root privileges do not have a huge choice of backup software for the command line, which obviously requires some knowledge of the command syntax. For users, it is important to be able to run a backup as smoothly and reliably as possible. The end user will only use the backup software – and actually perform the backup – if it is quick and easy to use.

Ideally, the same software can be used for mixed environments with both a backup server and additional desktop backups by users. Using the same software saves you the hassle of having to know the syntax of two programs – and thus avoids any associated errors.

Attic

The Attic backup program, which is written in Python, can be found in the repositories of some Linux distributions, such as Mageia, openSUSE, ROSA, or Slackware Linux; it can be installed conveniently using the respective package managers. The project page also provides the source code for download. Detailed documentation is also available [1].

Attic requires Python v3.2 or greater and openSSL in a version greater than 1.0.0. Because the software also lets you mount a backup set in user space, the *llfuse* package from the Python treasure trove has to be installed to provide this function.

After a successful installation, you first have to initialize a backup repository. This is achieved with the command:

```
attic init /<Repository-Path>/⮐
  <Repository-Name>.attic
```

Several directories can then be backed up in an archive (which should be specially created) in this repository. Attic does not enable encryption by default. The names for these archives can be freely selected. The following command backs up the directories:

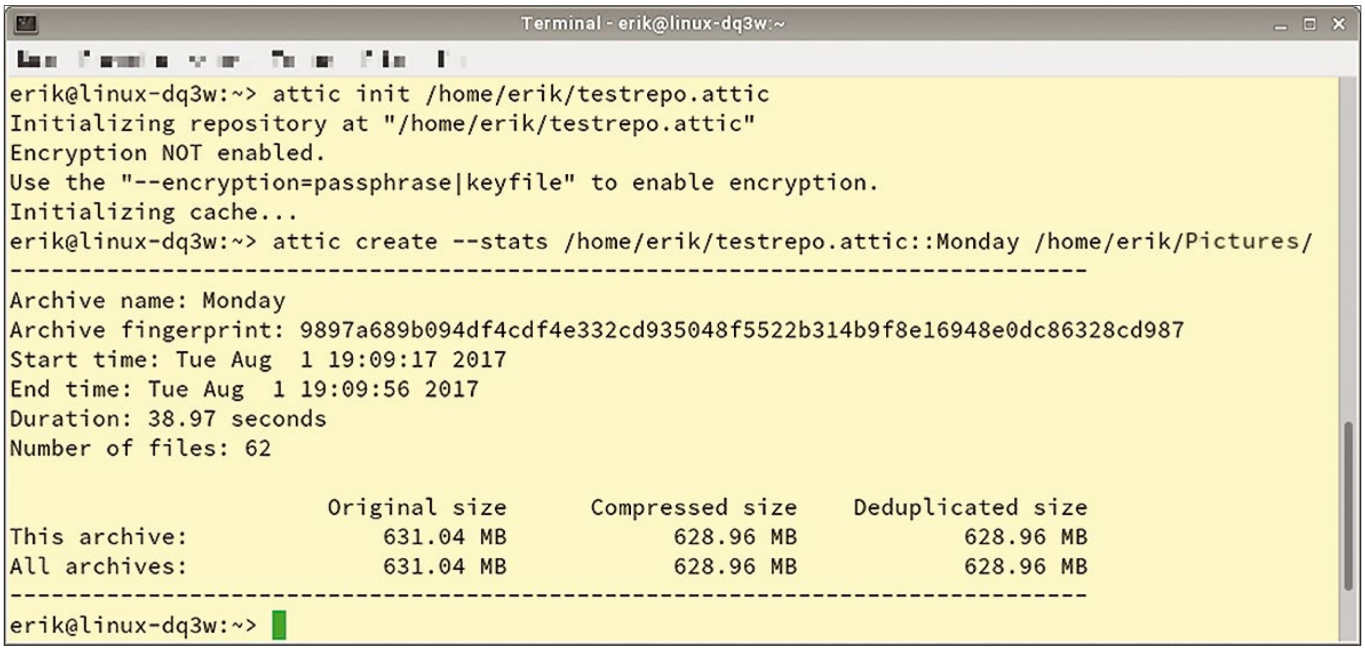
```
attic create /<Repository-Path>/⮐
  <Repository-Name>.attic::⮐
  <Archive-Name>/⮐
  <Source-Directory-1>/
  <[...]>
  <Source-Directory-n>
```

If the data must be encrypted, then the

```
--encryption=passphrase|keyfile
```

parameter command must be added.

I recommend using the weekday as the archive name for regular backups of



```
erik@linux-dq3w:~> attic init /home/erik/testrepo.attic
Initializing repository at "/home/erik/testrepo.attic"
Encryption NOT enabled.
Use the "--encryption=passphrase|keyfile" to enable encryption.
Initializing cache...
erik@linux-dq3w:~> attic create --stats /home/erik/testrepo.attic::Monday /home/erik/Pictures/
-----------------------------------------------------------------------------
Archive name: Monday
Archive fingerprint: 9897a689b094df4cdf4e332cd935048f5522b314b9f8e16948e0dc86328cd987
Start time: Tue Aug  1 19:09:17 2017
End time: Tue Aug  1 19:09:56 2017
Duration: 38.97 seconds
Number of files: 62

                       Original size      Compressed size    Deduplicated size
This archive:              631.04 MB            628.96 MB            628.96 MB
All archives:              631.04 MB            628.96 MB            628.96 MB
-----------------------------------------------------------------------------
erik@linux-dq3w:~>
```

Figure 1: Attic provides clear-cut data for the latest backup.

the same directories, which quickly gives you the correct sequence of backups during a restore. The first backup in a repository can take a long time to complete for large data volumes, but subsequent backups will be far quicker because Attic saves them incrementally (i.e., only modified or newly added data is included in the backup).

If you want to monitor the backup run, you can display the most important data for the backed up archive using the `--stats` parameter. Attic not only lists the directories and the required time for the backup run, but also the number of files backed up and the volume of data. It shows both the original and the compressed backed up data volumes, so you can keep track of data compression efficiency (Figure 1).

In contrast to many other backup tools, Attic provides a convenient approach to listing archive content. For this purpose, enter the following command at the prompt:

```
attic list -v /<Repository-Path>/⮐
  <Repository-Name>.attic::<Archive-Name>
```

The software then lists all the content, including file size, owner, and file permissions. Subdirectories are automatically included, and it shows the absolute paths.

Verifying the Archive

In the same easy way, you can check data integrity with the Attic backup program. The command

```
attic check /<Repository-Path>/⮐
  <Repository-Name>.attic
```

checks the repository and all its archives. The status is then output as a short message (Figure 2). If inconsistencies appear, you can perform repairs by repeating the command with the added `--repair` parameter.

Restore

To restore an archive, use the `extract` option by entering

```
attic extract /<Repository-Path>/⮐
  <Repository-Name>.attic::<Archive-Name>
```

which starts the recovery of the entire archive content to the original storage path. The software lists the individual files. The target path can be changed by an additional path specification, and you can exclude parts of the archive from the restore action.

Of course, Attic can also store backups on a remote server and retrieve them during a restore. It expects the same syntax as for a local backup; you address the server as follows:

```
<username>@<servername>:⮐
  <Repository-Name>.attic
```

However, enabling encryption is recommended when creating repositories on server systems. Attic uses 256-bit AES for encryption and HMAC-SHA256 for verification. Attic encrypts the data before storing it in the archive.

Automated

The software supports backup runs that are controlled and automated by cron jobs. To prevent the number of existing repositories from getting out of hand in the long run, Attic provides the `prune` parameter to let you define the storage life of older repositories. This option determines a maximum number of archives to be kept in the repository. You can define whether these are archives created hourly, daily, weekly, or monthly. However, they first must have been generated with the `date` parameter.

bup

The bup (short for backup) program has been in development and maintained for seven years, and it has established itself in small IT environments owing to its speed and efficiency [2]. The software is available from the repositories for immediate installation on many major Linux distributions.

Bup already differs from other backup tools in that it uses the Git packfile format instead of traditional TAR or ZIP archives. The software is very flexible in terms of handling: For example, bup archives can be mounted as filesystems in user space. Even backing up entire ISO images, which often contain several gigabytes of data, is no problem for the software. However, using the Git packfile format does require setting up a repository first.

For an overview of the software's numerous command parameters, call the program at the prompt without any options. Bup then lists the most important parameters in short form. You can also find detailed documentation on the project site [3]. For local backups on the desktop, the program has Gtk3 and Qt-based graphical front ends, which makes the backup process easier for inexperienced users. These are available from the bup homepage.

Usage

To create a backup, you first need to initialize the backup directory using the command sequence:

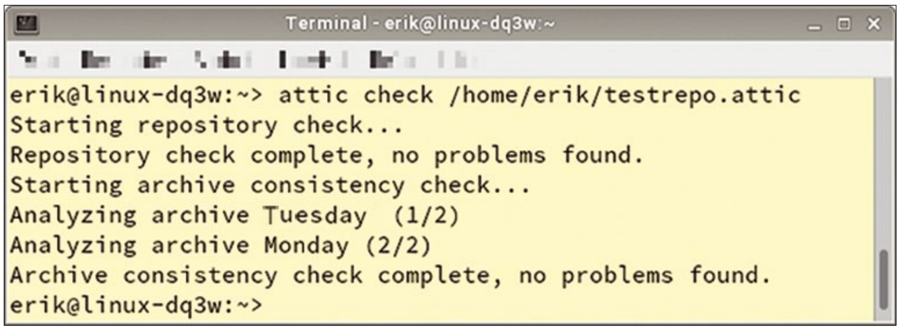
```
BUP_DIR=/<Backup-Set> bup init
```

Next, index the directory to be backed up with the command:

```
bup index -ux /<Directory>
```

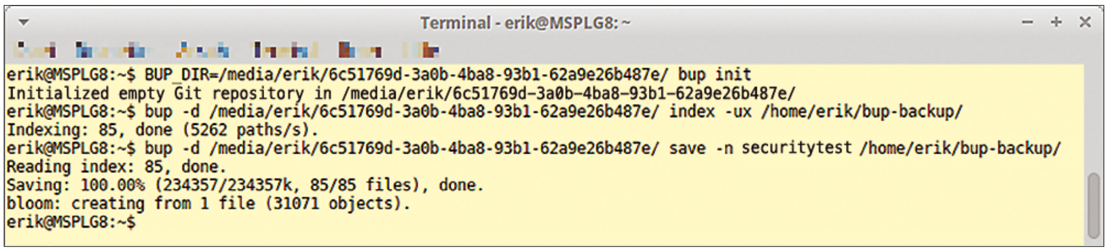
If no corresponding directory variable is defined, the `-d` parameter must also be specified with the path for the backup set. Otherwise, bup saves the backup set in the hidden `.bup` subfolder in your home directory. After indexing, perform the backup with:

```
bup save -n <Set-Name> ⮐
  /<Original-Directory>
```



```
erik@linux-dq3w:~> attic check /home/erik/testrepo.attic
Starting repository check...
Repository check complete, no problems found.
Starting archive consistency check...
Analyzing archive Tuesday  (1/2)
Analyzing archive Monday (2/2)
Archive consistency check complete, no problems found.
erik@linux-dq3w:~>
```

Figure 2: At a glance, you can see whether all the data was stored correctly.



```
erik@MSPLG8:~$ BUP_DIR=/media/erik/6c51769d-3a0b-4ba8-93b1-62a9e26b487e/ bup init
Initialized empty Git repository in /media/erik/6c51769d-3a0b-4ba8-93b1-62a9e26b487e/
erik@MSPLG8:~$ bup -d /media/erik/6c51769d-3a0b-4ba8-93b1-62a9e26b487e/ index -ux /home/erik/bup-backup/
Indexing: 85, done (5262 paths/s).
erik@MSPLG8:~$ bup -d /media/erik/6c51769d-3a0b-4ba8-93b1-62a9e26b487e/ save -n securitytest /home/erik/bup-backup/
Reading index: 85, done.
Saving: 100.00% (234357/234357k, 85/85 files), done.
bloom: creating from 1 file (31071 objects).
erik@MSPLG8:~$
```

Figure 3: With bup, data is backed up in a few steps.

In the event of missing directory variables, the path to the backup set must also be specified with a prefixed `-d` parameter. The set name can be optionally defined and is used to identify the correct set for recovery if several backups exist on the same target media.

During the first backup run, bup creates a full backup, which can take some time depending on the original data volume. The runs that follow only create incremental backups, which is much faster.

Keep in mind that before an additional backup run you have to re-index after each change to the original content for the software to detect changes.

If you want to store the backup on a remote server, in the simplest of cases, you would enter the following command sequence:

```
bup save -r <Username>@<Server-IP> ↩
  <Directory-Name> -n <Set-Name> ↩
  /<Original-Directory>
```

In this case, a full backup is run for the first pass, and the subsequent backups are incremental (Figure 3).

Bup also supports backing up a remote machine to a local machine. To do this, enter the `bup on` command followed by the server address and the target directory on the local system. Also, an indexing run must be performed beforehand.

To verify existing backup sets, use the `bup ls` command to list the individual files. The columns give you a clear overview (Figure 4).

Reverse Gear

Restoring backups is just as easy for users: After specifying the backup set path, use the `restore` parameter instead of `save`, followed by the set name and the path specification. Another option here is to restore individual subdirectories from the set.

Extras

Bup provides some little extras to check the integrity of backup sets. One of the most important goodies in practice is probably the option to mount a backup in user space like a conventional drive. The *python-fuse* package must be installed, and a mount directory must be created. The set is then integrated into the existing system by using:

```
bup -d <Path_to_Backupset> ↩
  fuse <Target-Directory>
```

You can then proceed to use the target directory and its content like any conventional drive.

Duplicity

Duplicity [4], which is available as a binary package for almost all common Linux distributions, is very flexible in terms of supported storage locations. Not only can you save backups locally, but also on FTP or SSH servers in the intranet. Additionally, Duplicity supports Windows shares and WebDAV storage.

If such central storage options are not available, Duplicity can store the backups in the cloud. For this option, it supports Amazon's S3 cloud and Rackspace cloud solutions. One of the unique selling points of Duplicity is encryption: All files can be encrypted with GnuPG (GPG) and stored safely in a cloud, without curious outsiders being able to view them.

The software takes a very professional approach to creating backups: The first pass is a full backup, which it bundles into a TAR archive. The program then uses incremental backups, which saves not only storage space, but also time – in particular for larger databases. Signatures ensure data integrity, even in insecure environments.

That said, Duplicity does require a fair amount of user training because of its very complicated parameters [5]. Alternatively, graphical front ends (e.g., Déjà Dup [6]) that operate more intuitively are recommended, especially in smaller environments or for desktop backups, even though they fall well short of reproducing Duplicity's full functionality. The software does not support differential backups.

Duplicity is suitable for backing up individual folders, but not for creating complete system images. (Programs like Clonezilla [7] are better suited to this purpose.) In doing so, the program stores the backed-up data in volumes. For local backups, you also need to specify the absolute path, so the data is stored in the correct target folder.

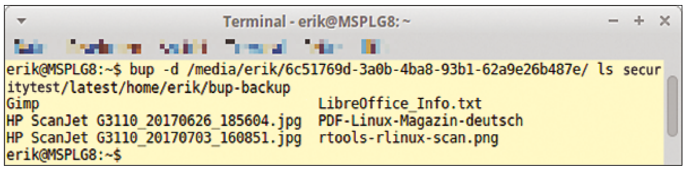
Key Service

To let Duplicity play to its strengths (encrypted backups), you first have to generate a GPG key or already be in possession of one. Entering `gpg --gen-key` at the prompt generates a new key in just a few steps, and you can also define the key strength (Figure 5).

The passphrase you specify when generating a key also provides increased security: If the key is lost, third parties cannot simply decrypt the archive because the passphrase is requested when restoring the database. Particularly paranoid users can also digitally sign their archives so that the data integrity of the backup volumes can be checked at a later time.

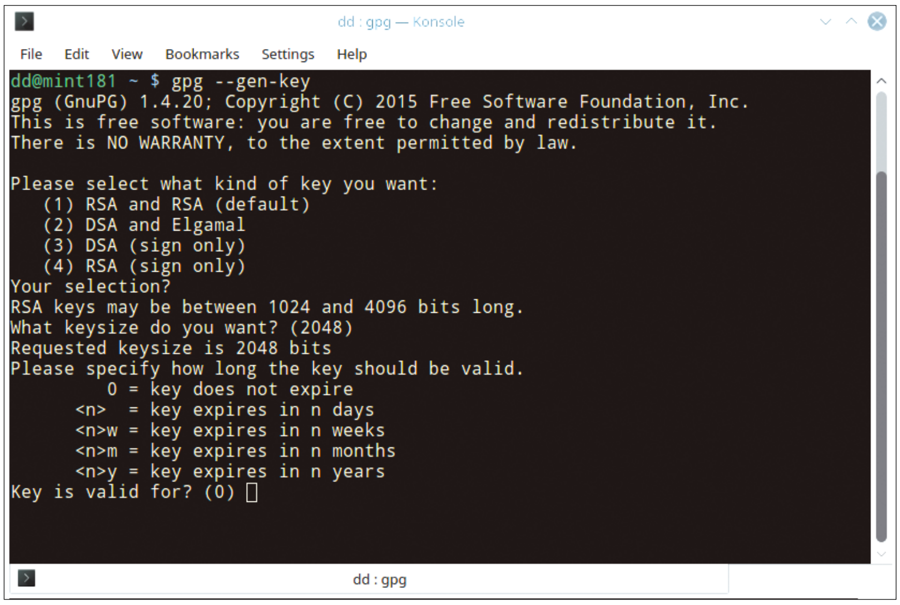
In the simplest case, specify the following command for a local backup:

```
duplicity /<Source-Directory> ↩
  file://<Target-Directory>
```



```
erik@MSPLG8:~$ bup -d /media/erik/6c51769d-3a0b-4ba8-93b1-62a9e26b487e/ ls secur
itytest/latest/home/erik/bup-backup
Gimp                                LibreOffice_Info.txt
HP ScanJet G3110_20170626_185604.jpg  PDF-Linux-Magazin-deutsch
HP ScanJet G3110_20170703_160851.jpg  rtools-rlinux-scan.png
erik@MSPLG8:~$
```

Figure 4: A simple display of the backed up data makes it easier for admins to check the backup.



```
dd@mint181 ~ $ gpg --gen-key
gpg (GnuPG) 1.4.20; Copyright (C) 2015 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Please select what kind of key you want:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
Your selection?
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
Requested keysize is 2048 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0)
```

Figure 5: A key can be created in just a few minutes.

After prompting you for the passphrase twice, the software creates the backup in the target directory.

To save data on an FTP server, call the software using the command:

```
duplicity /<Source-Directory> ⮐
  ftp://<backupuser>@<Host.Name>/⮐
  <Target-Directory>
```

The passphrase is also requested here before saving; you can work around the prompt by prefixing the sequence `FTP_PASSWORD=<Password>` to the backup command. In both cases, Duplicity automatically checks to see whether a full backup already exists in the target path. If this is not the case, it automatically creates the full backup during the first pass. With the second pass, it creates an incremental backup (Figure 6).

Manual Trigger

An increasing number of incremental backups accumulates over time, especially if you automate the backup runs. Because these need to be restored in sequence from the last full backup, it is a very good idea to create a new full backup regularly to minimize the number of incrementally backed up archives. Duplicity launches into a full manual backup when started with the `full` parameter. The `incremental` parameter manually triggers an incremental backup.

If specific directories are to be excluded from the backup, you can specify these using the `--exclude` parameter. Several directories that you do not want backed up can be added in a space-separated list. Excluding individual directories is useful, especially for subfolders that contain, for example, temporary, cache, or logfiles.

However, caution is required when backing up the root directory: The `/proc` directory must be excluded; otherwise, Duplicity will freeze. Conversely, the `--include` parameter can include additional folders in the backup.

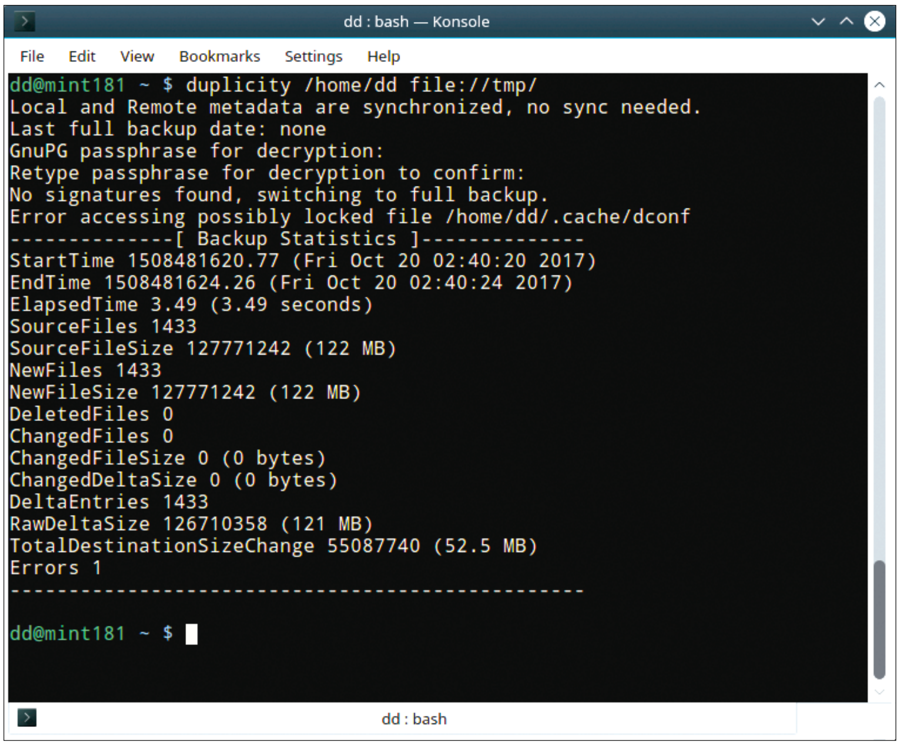
Restore

Data recovery is just as easy: Duplicity is activated without any additional parameters; only the source and target paths need to be swapped. Alternatively, you can also specify a different target path. The archive is only unpacked after entering the passphrase, so unauthorized persons cannot access the data without knowing the password.

Using the `verify` and `--compare-data` parameters, you can test the integrity of your backups in a simple way. This test is possible even if the volumes are in a cloud. Particularly for documentation purposes, it is recommended to use the additional `-v<x>` parameter (verbosity level), and to replace the *x* placeholder with one of the numbers 4, 8, or 9 to obtain meaningful information. The data can then be saved in the specified file using the `--log-file <file name>` parameter.

rdiff-backup

Rdiff-backup [8] is another popular tool for backing up at the command line. The program is maintained in the repositories of virtually all major Linux distributions. The software is extremely easy to use: To produce a local backup of a directory or directory tree at the prompt, type:



```
dd@mint181 ~ $ duplicity /home/dd file:///tmp/
Local and Remote metadata are synchronized, no sync needed.
Last full backup date: none
GnuPG passphrase for decryption:
Retype passphrase for decryption to confirm:
No signatures found, switching to full backup.
Error accessing possibly locked file /home/dd/.cache/dconf
--------------[ Backup Statistics ]--------------
StartTime 1508481620.77 (Fri Oct 20 02:40:20 2017)
EndTime 1508481624.26 (Fri Oct 20 02:40:24 2017)
ElapsedTime 3.49 (3.49 seconds)
SourceFiles 1433
SourceFileSize 127771242 (122 MB)
NewFiles 1433
NewFileSize 127771242 (122 MB)
DeletedFiles 0
ChangedFiles 0
ChangedFileSize 0 (0 bytes)
ChangedDeltaSize 0 (0 bytes)
DeltaEntries 1433
RawDeltaSize 126710358 (121 MB)
TotalDestinationSizeChange 55087740 (52.5 MB)
Errors 1
-------------------------------------------------

dd@mint181 ~ $
```

Figure 6: Duplicity also provides information about the backup.

```
rdiff-backup <Source-Directory> ⮐
             <Target-Directory>
```

This creates a full backup that the administrator can recover without special tools just as easily as using a conventional folder hierarchy, using only onboard Linux tools.

Subsequently backups made by rdiff-backup contain deleted or older versions of the changed files. The latest versions of the changed files all end up in the full backup, just like new files, so you always have the current status of the backup. Regular backups only save older versions of the database. The advantage of this special form of incremental backup, known as reverse delta, is that you do not have to fight your way through several incremental backups to restore a complete database.

Servers

Using rdiff-backup, you can also back up servers in the intranet. Note that you do need to install the application on the server. To run a backup over SSH, at the remote client prompt enter:

```
rdiff-backup <Username>@<Server-IP::⮐
  <Source-Directory> <Backup-Directory>
```

This command backs up the source directory from the server on the local client. If the backup also includes files to which only root has access, you will need to log in with the appropriate privileges.

Displays

Rdiff-backup, like all other tools being reviewed here, has a collection of parameters for managing backups. To view the content of a backup, enter

```
rdiff-backup -l <Backup-Name>
```

at the prompt.

Because rdiff-backup does not output status messages during the backup runs, it also makes sense to enter the `-v6` parameter to switch to the highest verbosity level. The software then outputs all the processed files. Additionally, after launching the backup, various status messages appear about enabled and disabled parameters (Figure 7).

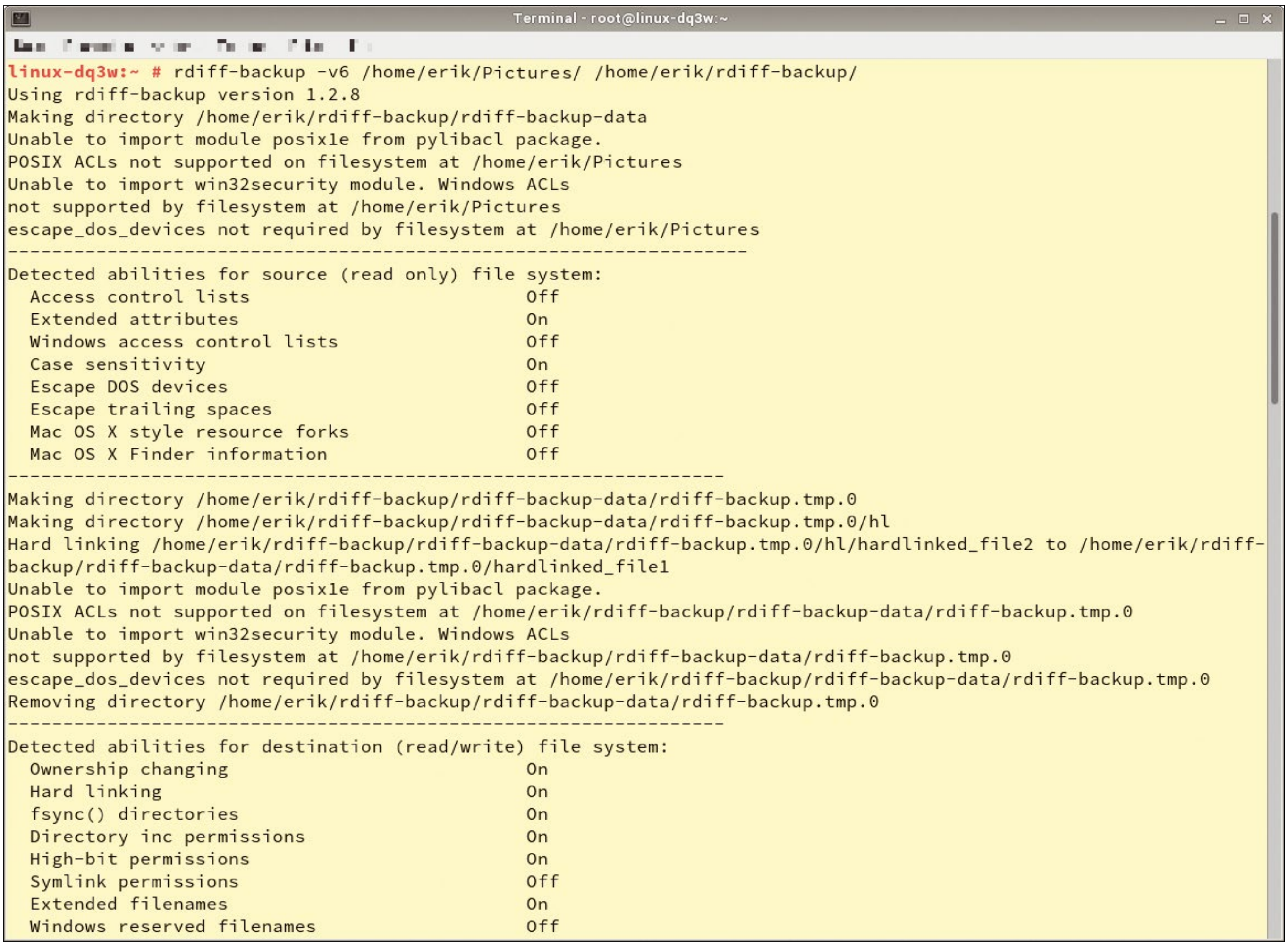
The `--compare` and `--list-increments` parameters provide information about the modified files and the different backup points. Using the command

```
rdiff-backup-statistics ⮐
  <Backup Directory>
```

you can also retrieve statistical data for the backup directory (Figure 8).

Exceptions

As with other backup programs, you can also exclude files or directories from the backup. This can be beneficial, for example, if the backup path includes directories for temporary files that you don't want to save. In such cases, use the `--exclude` parameter to exclude individual files.



```
linux-dq3w:~ # rdiff-backup -v6 /home/erik/Pictures/ /home/erik/rdiff-backup/
Using rdiff-backup version 1.2.8
Making directory /home/erik/rdiff-backup/rdiff-backup-data
Unable to import module posix1e from pylibacl package.
POSIX ACLs not supported on filesystem at /home/erik/Pictures
Unable to import win32security module. Windows ACLs
not supported by filesystem at /home/erik/Pictures
escape_dos_devices not required by filesystem at /home/erik/Pictures
-------------------------------------------------------------------
Detected abilities for source (read only) file system:
  Access control lists                       Off
  Extended attributes                        On
  Windows access control lists               Off
  Case sensitivity                           On
  Escape DOS devices                         Off
  Escape trailing spaces                     Off
  Mac OS X style resource forks              Off
  Mac OS X Finder information                Off
-----------------------------------------------------------------
Making directory /home/erik/rdiff-backup/rdiff-backup-data/rdiff-backup.tmp.0
Making directory /home/erik/rdiff-backup/rdiff-backup-data/rdiff-backup.tmp.0/hl
Hard linking /home/erik/rdiff-backup/rdiff-backup-data/rdiff-backup.tmp.0/hl/hardlinked_file2 to /home/erik/rdiff-
backup/rdiff-backup-data/rdiff-backup.tmp.0/hardlinked_file1
Unable to import module posix1e from pylibacl package.
POSIX ACLs not supported on filesystem at /home/erik/rdiff-backup/rdiff-backup-data/rdiff-backup.tmp.0
Unable to import win32security module. Windows ACLs
not supported by filesystem at /home/erik/rdiff-backup/rdiff-backup-data/rdiff-backup.tmp.0
escape_dos_devices not required by filesystem at /home/erik/rdiff-backup/rdiff-backup-data/rdiff-backup.tmp.0
Removing directory /home/erik/rdiff-backup/rdiff-backup-data/rdiff-backup.tmp.0
-----------------------------------------------------------------
Detected abilities for destination (read/write) file system:
  Ownership changing                         On
  Hard linking                               On
  fsync() directories                        On
  Directory inc permissions                  On
  High-bit permissions                       On
  Symlink permissions                        Off
  Extended filenames                         On
  Windows reserved filenames                 Off
```

Figure 7: rdiff-backup gives you far more information than other backup programs – if you tell it to do so.

If you need to exclude complete directories from the backup run, you can use the `--exclude-filelist` parameter; you need to specify a file containing the paths of the files to be excluded. The file has to be created manually.

Restore

To restore backed-up files from the archives, you don't need to invoke rdiff-backup again: If you need to restore your data collection from the current archive, you can simply copy the data from the backup directory. The Linux `cp` command helps you do this; you need to specify the archive option. The easiest form is:

```
cp -a /<Backup-Directory>/<File-Name> ⮐
  /<Restore-Directory>/<File-Name>
```

To access old backup files, the software offers two options: You can either access the incremental files from the `rdiff-backup` program, or you can use `rdiff-backup-fs`, which must be installed separately as an external package, to mount the backup archive as a conventional filesystem.

The *rdiff-backup-fs* [9] package (the name can differ among distros) is maintained in some software repositories of major Linux distributions, but it can also be picked up from the project website. For direct access, use the `-r` parameter (short for "restore as-of") followed by a date. For example, you need the following command sequence to restore a 10-day-old backup from a server to a local directory on a client:

```
rdiff-backup -r 10D ⮐
  <Server-Name>::/<Source-Directory>/⮐
  <Restore-Directory>
```

The detailed documentation [10] lists various application scenarios that makes the somewhat unusual nomenclature understandable.

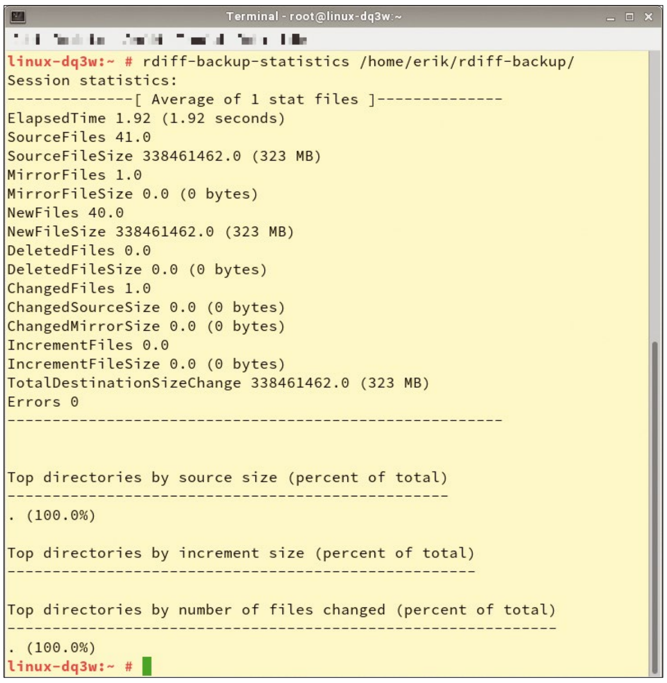
rsnapshot

As the name suggests, rsnapshot is a tool for creating complete snapshots of a filesystem [11]. It can create both local snapshots and use SSH to create snapshots of remote systems. The Rsync tool creates backups in which hard links replace unchanged files. Rsnapshot actually only writes modified data in the backup. The cron daemon regularly initiates the backup. Manual backup runs are not intended. The software takes all its information from a configuration file.

Because rsnapshot works with hard links, the backups always have to end up on the same filesystem. Otherwise, the tool would have to create a space-consuming full backup. Therefore, rsnapshot is more suited to servers on small networks or to local workstations than to extensive storage setups. However, the tool, which is written in Perl, only stores a configurable number of backups, so the storage space remains manageable, even with short backup intervals.

Configuration

After the installation, you need to configure the program, which is available from the repositories of practically all major Linux distributions. To do this, call the `/etc/rsnapshot.conf` file, which



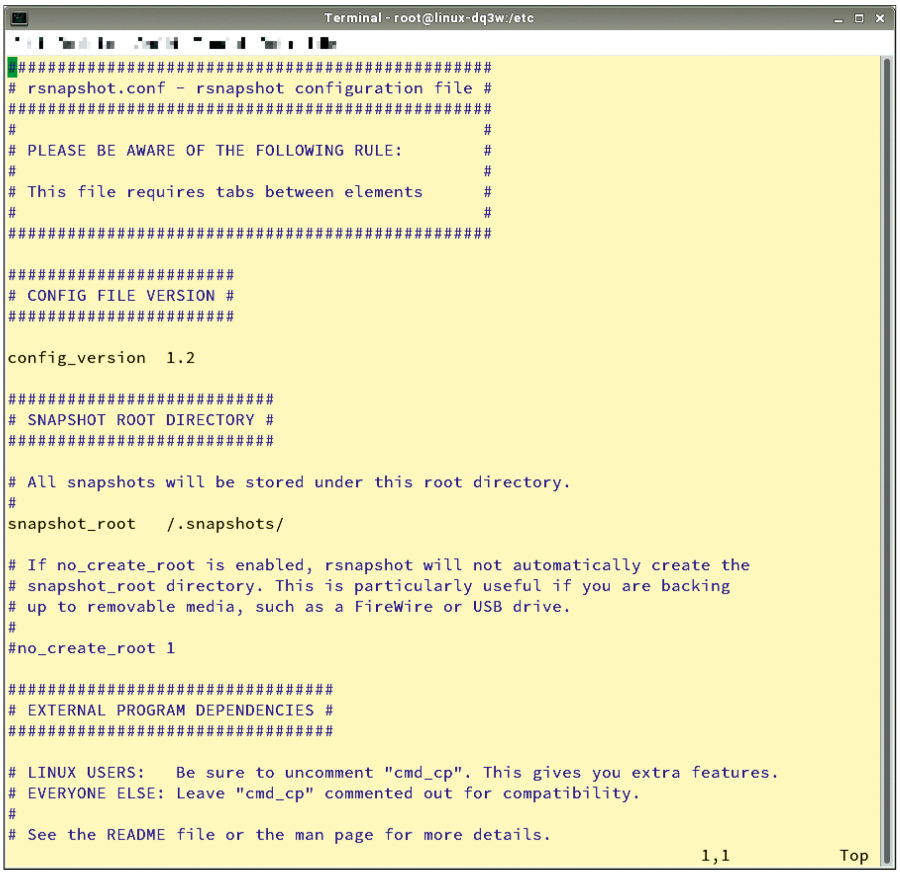
```
Terminal - root@linux-dq3w:~
linux-dq3w:~ # rdiff-backup-statistics /home/erik/rdiff-backup/
Session statistics:
--------------[ Average of 1 stat files ]--------------
ElapsedTime 1.92 (1.92 seconds)
SourceFiles 41.0
SourceFileSize 338461462.0 (323 MB)
MirrorFiles 1.0
MirrorFileSize 0.0 (0 bytes)
NewFiles 40.0
NewFileSize 338461462.0 (323 MB)
DeletedFiles 0.0
DeletedFileSize 0.0 (0 bytes)
ChangedFiles 1.0
ChangedSourceSize 0.0 (0 bytes)
ChangedMirrorSize 0.0 (0 bytes)
IncrementFiles 0.0
IncrementFileSize 0.0 (0 bytes)
TotalDestinationSizeChange 338461462.0 (323 MB)
Errors 0
-------------------------------------------------------


Top directories by source size (percent of total)
-------------------------------------------------
. (100.0%)

Top directories by increment size (percent of total)
----------------------------------------------------

Top directories by number of files changed (percent of total)
-------------------------------------------------------------
. (100.0%)
linux-dq3w:~ #
```

Figure 8: One of the various statistical data displays makes it easier for you to check the backups.



```
Terminal - root@linux-dq3w:/etc
#################################################
# rsnapshot.conf - rsnapshot configuration file #
#################################################
#                                               #
# PLEASE BE AWARE OF THE FOLLOWING RULE:        #
#                                               #
# This file requires tabs between elements      #
#                                               #
#################################################

#######################
# CONFIG FILE VERSION #
#######################

config_version  1.2

###########################
# SNAPSHOT ROOT DIRECTORY #
###########################

# All snapshots will be stored under this root directory.
#
snapshot_root   /.snapshots/

# If no_create_root is enabled, rsnapshot will not automatically create the
# snapshot_root directory. This is particularly useful if you are backing
# up to removable media, such as a FireWire or USB drive.
#
#no_create_root 1

#################################
# EXTERNAL PROGRAM DEPENDENCIES #
#################################

# LINUX USERS:   Be sure to uncomment "cmd_cp". This gives you extra features.
# EVERYONE ELSE: Leave "cmd_cp" commented out for compatibility.
#
# See the README file or the man page for more details.
                                                              1,1           Top
```

Figure 9: rsnapshot is set up with a configuration file.

can be edited easily with any standard text editor.

Because the numerous options are easily accessible, thanks to detailed comments, the configuration does not pose unsolvable problems, even for inexperienced users. Note that the different options need to be separated by tabs. Also, path specifications must always end with a slash.

The configuration file not only lets you specify the source and target directories and list included or excluded files, it also contains a schedule for a backup plan. Moreover, settings can be defined to back up a remote server via SSH or handle an LVM network. Settings for the snapshot retention period (Figure 9) are also important.

The configuration file is quite extensive, so you can validate it after a modification. To do this, enter the `rsnapshot configtest` command at the prompt with administrative privileges. The *Syntax OK* output signals a consistent configuration. A cron or anacron job then launches the software automatically. A predefined sample configuration is available in the `/etc/cron.d/rsnapshot` file on some distributions, and its data can be adapted to suit your individual needs. A FAQ [12] provides more information.

Rotation Keeps Data Under Control

The rotation principle helps you keep the number of backups manageable in line with your needs. You can use the configuration file to define intervals at which backups are created and specify how many backups must be retained for each interval. The `retain` parameter in the `/etc/rsnapshot.conf` file uses time parameters such as `daily` or `hourly` followed by a number to configure both.

For example, `daily retain 5` means that a daily backup launches, and the last five daily backups should be kept. In further lines, additional intervals can be defined in the same way, each with its own number of backups to be retained, making rsnapshot very flexible in terms of use.

After completing the configuration, launch a first test run by entering the following command:

```
rsnapshot -v <Interval>
```

Use the `Interval` parameter to specify the defined interval in the configuration file (i.e., `hourly` or `daily`). You do not have to specify source and target directories, because rsnapshot takes this information from the configuration file.

Because rsnapshot does not store its backups in archives or its own formats, the data is directly accessible and can be easily copied back for recovery. After a successful test run, you can set up the required cron jobs.

Conclusions

The backup solutions explored here are all reliable and stable. They are suitable for backing up both local systems and servers. However, if you want to outsource your backup archives into the cloud, most programs will not be appropriate because of their lack of encryption support. In a public cloud, you will not want to store your backups without encryption.

Additionally, some candidates have poor to simply catastrophic documentation. In part, the existing description does not list function parameters or sample applications, and the man pages are terse 10-liners. Consequently, developers should not expect their programs to be widely accepted by users whose time and patience are limited.

These backup solutions, most of which are developed for Unix-style operating systems, also exhibit some design weaknesses, in that they may not always work in heterogeneous environments. For example, hard links cannot be used on all platforms or with all filesystems. For a tool like rsnapshot, this factor limits the applications to Linux and related systems. ■■■

Info

[1] Attic: *https://attic-backup.org/installation.html#installation*

[2] bup on GitHub: *https://github.com/bup/bup*

[3] bup documentation: *https://bup.github.io/man.html*

[4] Duplicity: *http://duplicity.nongnu.org*

[5] Duplicity documentation: *http://duplicity.nongnu.org/duplicity.1.html*

[6] Déjà Dup: *https://launchpad.net/deja-dup*

[7] Clonezilla: *http://clonezilla.org*

[8] rdiff-backup: *http://www.nongnu.org/rdiff-backup/*

[9] rdiff-backup-fs: *https://github.com/rbrito/rdiff-backup-fs*

[10] rdiff-backup documentation: *http://www.nongnu.org/rdiff-backup/examples.html*

[11] rsnapshot: *http://rsnapshot.org*

[12] rsnapshot FAQ: *http://rsnapshot.org/faq.html*

Janitorial Services

Stacer is a handy graphical tool for cleaning up your Linux system. *By Ferdinand Thommes*

Classic command-line utilities are considered the go-to tools for system administration, but some powerful graphical tools also are available for monitoring and optimizing a Linux system. One of those tools is Stacer, which lives on GitHub [1]. Sourceforge [2] also offers sources for compiling, as well as a DEB packages for 32- and 64-bit systems and an AppImage for 64-bit machines. In this article, I take a close look at the Stacer AppImage version. (See the box titled "AppImage" for more on the AppImage format.)

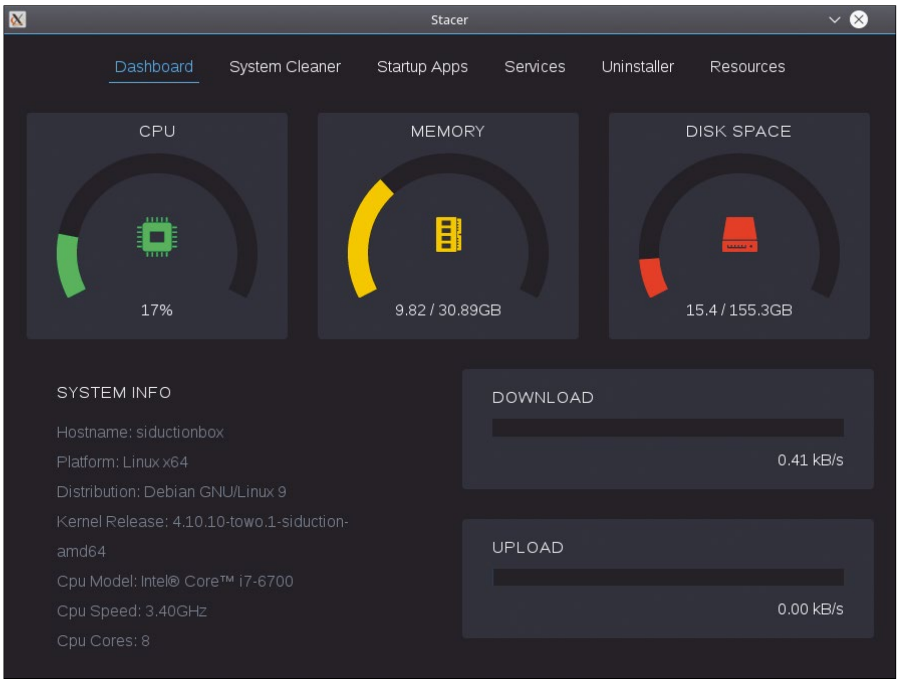
What Is Stacer?

Stacer was designed for Ubuntu but works with any distribution, with a couple of restrictions. The application, created by GitHub developers, is based on the Electron framework [4] and can be used for building cross-platform apps on the basis of JavaScript, HTML, and CSS. Skype for Linux and the Atom editor, Franz messenger, and Darktable image processing tool are some of the better known representatives of the framework.

AppImage

If AppImage [3] does not mean anything to you, you're not alone. Other cross-distribution packaging alternatives, such as Snap and Flatpak, have received much more attention. AppImage, which has been under development since 2004 (initially known as Klik and later as PortableLinuxApps), is relatively unknown. The apps packaged in AppImage format run without installation.

After downloading the AppImage, I distributed it to various virtual machines running Ubuntu, Linux Mint, Mageia, Manjaro, Apricity OS, and openSUSE and tried a static installation with Siduction (Debian Unstable). Stacer ran on all the distributions tested. On systems with KDE Plasma desktop, however, you need



SYSTEM INFO
Hostname: siductionbox
Platform: Linux x64
Distribution: Debian GNU/Linux 9
Kernel Release: 4.10.10-towo.1-siduction-amd64
Cpu Model: Intel® Core™ i7-6700
Cpu Speed: 3.40GHz
Cpu Cores: 8

Figure 1: Among other things, the Stacer start screen displays animated CPU, Memory, and Disk Space gauges.

to launch Stacer as root because of a kdesu authorization error.

Before the first start, you need to make the AppImage executable, which you can do as a user working in the directory where the package is located; then, run Stacer from the same directory:

```
$ chmod a+x Stacer-1.0.6-x86_64.AppImage
$ ./Stacer-1.0.6-x86_64.AppImage
```

A message tells you that the setup routine is adding the application to the menu and putting an icon on the desktop, which is the only way in which the application changes your computer. Alternatively, you can launch Stacer, like all AppImage applications, by double-clicking on the executable file.

Graphically Managed

Stacer welcomes you with a modern window featuring six tabs: *Dashboard*, *System Cleaner*, *Startup Apps*, *Services*, *Uninstaller*, and *Resources* (Figure 1). The window is static, which means you can neither increase nor decrease its size.

The program always starts with the Dashboard, which only provides information and does not allow any interaction. The Dashboard gives you a animated view of CPU, Memory, Disk Space, and network interface utilization, as well as information about the installed processor and operating system.

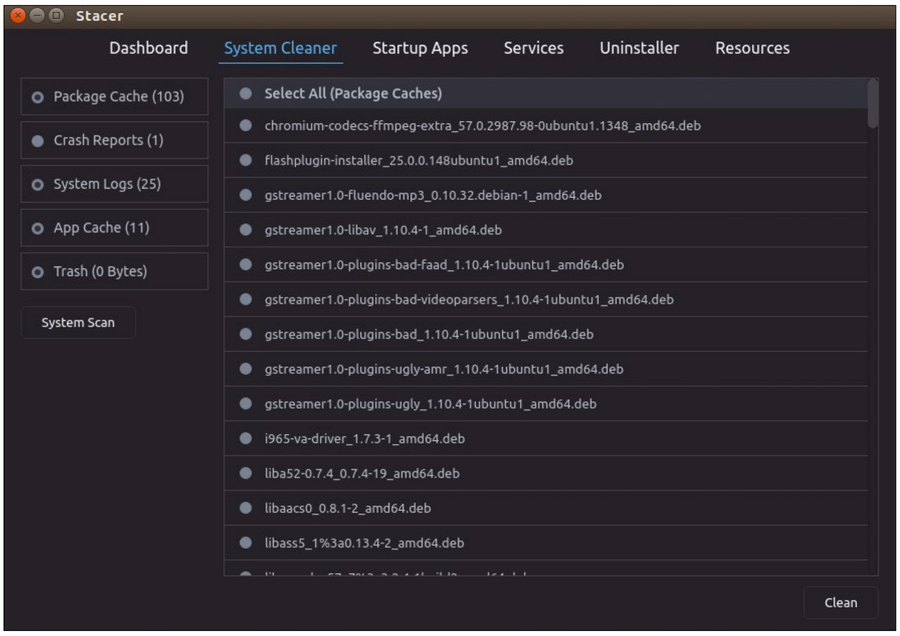
System Cleaner

The *System Cleaner* tab (Figure 2) helps you ditch the ballast: This is where you can remove unnecessary log or cache files and empty the trash can on your system. In the initial state, Stacer does not provide any data for trash disposal; you first need to enable the desired categories and then launch a system scan.

Caution is advisable in the *App Cache* tab: Deleting here could slow down application launch, and you should proceed with caution when it comes to the logs and keep at least the current X.org log and the Apt and Dpkg logfiles. Numbered logs are always older and can typically be disposed of without any worries.

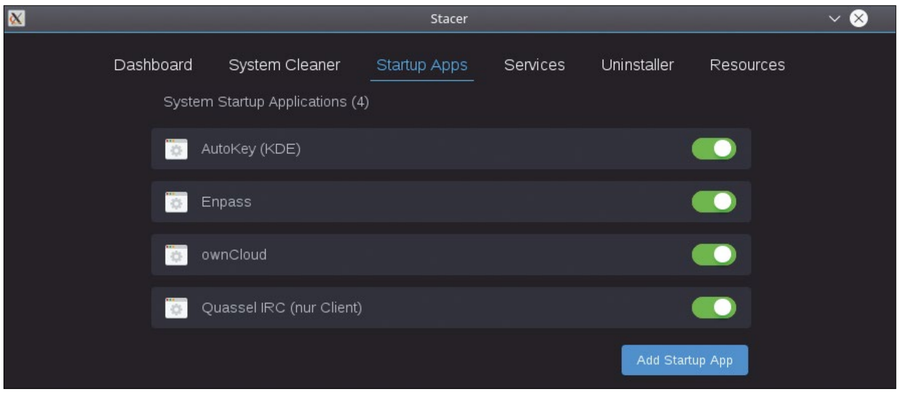
Apps and Services

In the *Startup Apps* tab, you can view the applications the system launches at boot time (Figure 3) and set up new startup apps. This is especially handy if



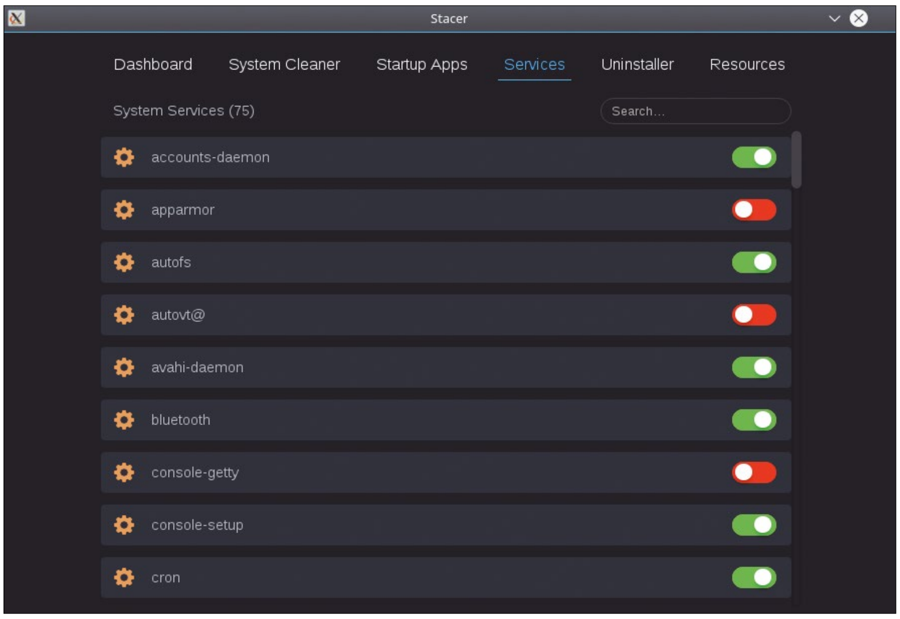
Select All (Package Caches)
chromium-codecs-ffmpeg-extra_57.0.2987.98-0ubuntu1.1348_amd64.deb
flashplugin-installer_25.0.0.148ubuntu1_amd64.deb
gstreamer1.0-fluendo-mp3_0.10.32.debian-1_amd64.deb
gstreamer1.0-libav_1.10.4-1_amd64.deb
gstreamer1.0-plugins-bad-faad_1.10.4-1ubuntu1_amd64.deb
gstreamer1.0-plugins-bad-videoparsers_1.10.4-1ubuntu1_amd64.deb
gstreamer1.0-plugins-bad_1.10.4-1ubuntu1_amd64.deb
gstreamer1.0-plugins-ugly-amr_1.10.4-1ubuntu1_amd64.deb
gstreamer1.0-plugins-ugly_1.10.4-1ubuntu1_amd64.deb
i965-va-driver_1.7.3-1_amd64.deb
liba52-0.7.4_0.7.4-19_amd64.deb
libaacs0_0.8.1-2_amd64.deb
libass5_1%3a0.13.4-2_amd64.deb

Figure 2: In the System Cleaner section, you can get rid of temporary files, caches, and old logfiles.



System Startup Applications (4)
AutoKey (KDE)
Enpass
ownCloud
Quassel IRC (nur Client)

Figure 3: In the Startup Apps section, you can add and remove applications to be launched at boot time.



System Services (75)
accounts-daemon
apparmor
autofs
autovt@
avahi-daemon
bluetooth
console-getty
console-setup
cron

Figure 4: Consider carefully starting and stopping services to avoid endangering the running system.

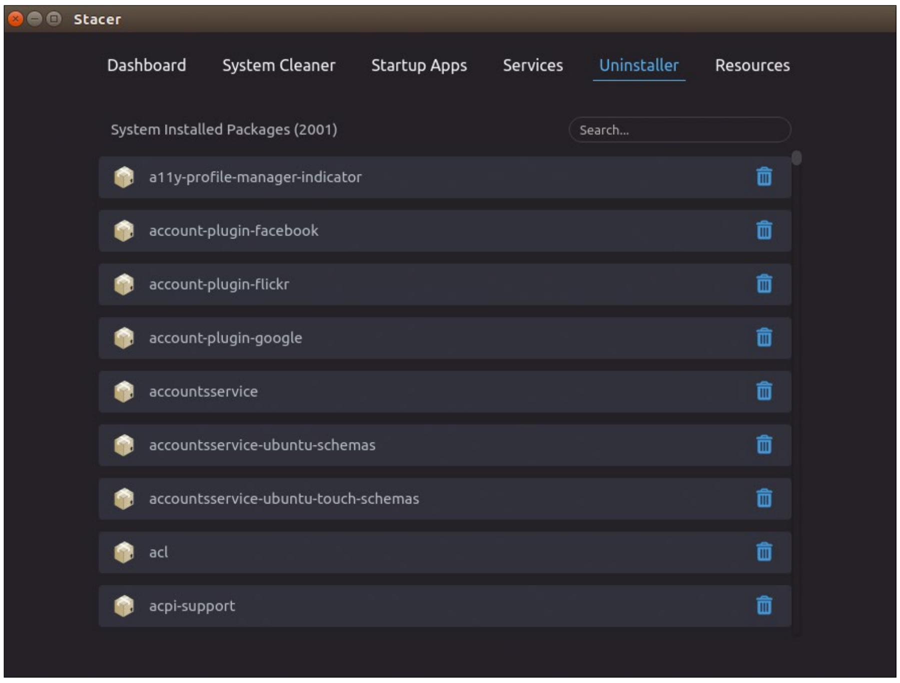
you work with different distributions: You do not always need to think about where you need to set up applications that run at boot time on the respective systems, and you can also tell Stacer to lock an application for the next start as a test, without having to plumb the depths of the Control Panel.

Starting and stopping system services is just as easy in the *Services* tab (Figure 4). A search function facilitates finding a particular service. A word of caution: If you shut down the wrong service here, you can look forward to a reboot.

Out!

Like the first tab, the last two tabs are purely informative: As the name suggests, the penultimate tab, *Uninstaller*, lets you remove packages (Figure 5). You will find many of the applications installed on the system here, and you can point and click to uninstall and remove. Stacer does not list basic packages, to keep users from pulling the rug out from under their own feet.

The uninstaller works perfectly with Ubuntu and Apricity OS, but not with any other distribution tested in the lab. Rummaging around the bug reports on GitHub revealed an announcement stating that Stacer can only handle this function on Ubuntu and Arch Linux (on which Apricity OS is based). Failure to uninstall here is not really tragic, because it makes more sense to delete packages with your distribution's package manager anyway.


Stacer

Dashboard | System Cleaner | Startup Apps | Services | Uninstaller | Resources

System Installed Packages (2001)

Search...

a11y-profile-manager-indicator
account-plugin-facebook
account-plugin-flickr
account-plugin-google
accountsservice
accountsservice-ubuntu-schemas
accountsservice-ubuntu-touch-schemas
acl
acpi-support

Figure 5: The Uninstaller section only works on Ubuntu and Arch Linux distros and their derivatives.

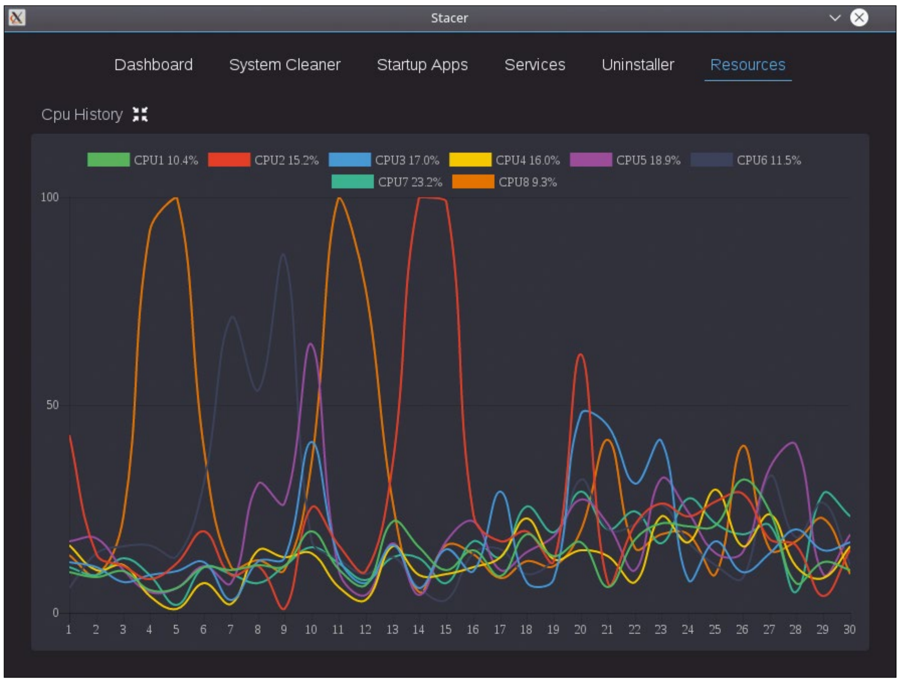


Figure 6: In the Resources section, you can open selected objects (e.g., the CPU) and view animations of all the cores individually.

Colorful Plots

The *Resources* tab displays the last 30 seconds of CPU, RAM, and network activity (Figure 6). If you have four, eight, or more cores, Stacer shows them individually in contrasting colors. To view each plot separately, press the *Cpu History* button, for example.

Conclusions

All tasks handled by Stacer can be run at the command line, with standard graphical tools, or even with the use of competitor products such as BleachBit [5], but not with the same graphical appeal. Stacer looked best on Apricity OS [6] because it perfectly matches the modern appearance of this Arch Linux-based distribution.

Delivery as an AppImage lets you distribute Stacer to multiple distributions in a single package; additionally, you save yourself the installation overhead. However, the AppImage weighs in at more than 50MB, and the executable file is around 75MB after unpacking; the complete package tips the scales at 130MB. Keep in mind when using Stacer that you could shoot yourself in the foot with poorly considered actions. ■■■

Info

[1] Stacer: *https://github.com/oguzhaninan/Stacer*

[2] Download: *https://sourceforge.net/projects/stacer/files/v1.0.6/*

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

[4] Electron: *https://electron.atom.io*

[5] BleachBit: *https://www.bleachbit.org*

[6] Apricity: *http://www.apricityos.com*

Teach Neural Networks to identify sequences of values

First Things First

2, 5, 7, 10, 12 – and what number comes next? Mike Schilli tests whether intelligence tests devised by psychologists can be cracked with modern AI Networks. *By Mike Schilli*

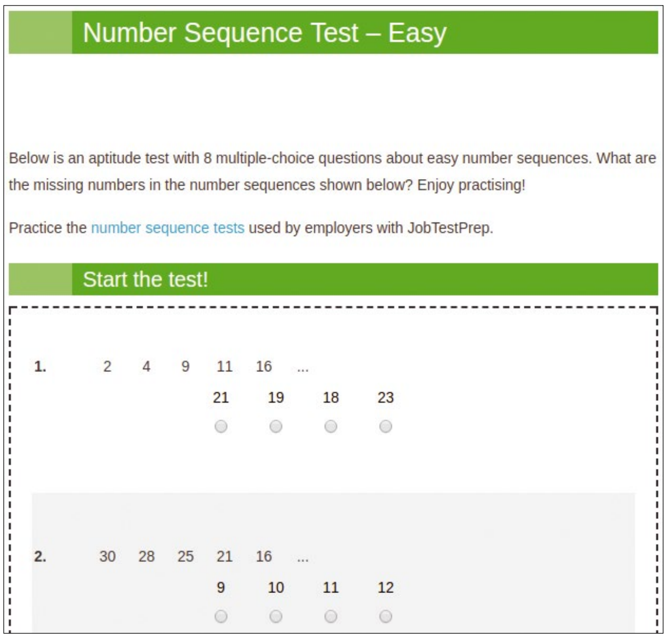
Neural networks do great things when it comes to detecting patterns in noisy input data and assigning unambiguous results to them. If a dozen people with different handwriting enter the letters A or B in a form, a trained network can identify with almost 100 percent certainty what they wrote. Or consider pattern recognition systems for identifying the license plates of passing vehicles: Aren't these technical miracles? They extract the digits from a camera feed so that the Department of Transportation knows exactly who is going where.

Once a neural network is done learning, it always assigns the same result to the same input data, but when it comes to tasks that need to determine the next value in time-discrete value sequences, neural networks often fail to deliver perfect results, especially if the input signal is subject to variations of unknown periodicity.

In a neural network, the learning algorithm adjusts internal weights based on the training data. However, once these weights are determined, they won't change anymore at run time and thus cannot account for temporal changes in the input data, because the machine doesn't remember any previous state. Recurrent neural networks (RNNs) maintain internal connections back to the input, and thus a result can influence the next input vector, but this does not help a simple network identify temporal patterns that extend over several cycles.

At the Psychologist's

A somewhat entertaining example of predicting sequences are intelligence tests (Figure 1) performed by psychologists, where the candidate is asked to determine the number that comes next in a numeric sequence. Any school kid can tell that 2, 4, 6 is followed by 8, but what comes next after the sequence 2, 5, 7, 10, 12? Figure 2 shows two learning steps and a test step for a Long Short-Term Memory (LSTM) network that I want to teach which number comes after 12. In the first learning step, in the first row of the matrix, it learns that the combination `2, 5, 7` is always


Number Sequence Test – Easy

Below is an aptitude test with 8 multiple-choice questions about easy number sequences. What are the missing numbers in the number sequences shown below? Enjoy practising!

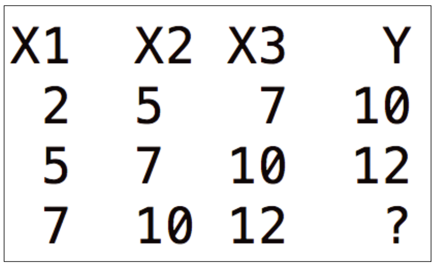
Practice the number sequence tests used by employers with JobTestPrep.

Start the test!

1. 2 4 9 11 16 ...
21 ○ 19 ○ 18 ○ 23 ○

2. 30 28 25 21 16 ...
9 ○ 10 ○ 11 ○ 12 ○

Figure 1: An intelligence test asking the candidate to complete a sequence of numbers [1].



X1	X2	X3	Y
2	5	7	10
5	7	10	12
7	10	12	?

Figure 2: Input and output values for training the LSTM network.

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 go to *mschilli@perlmeister.com* he will gladly answer any questions.

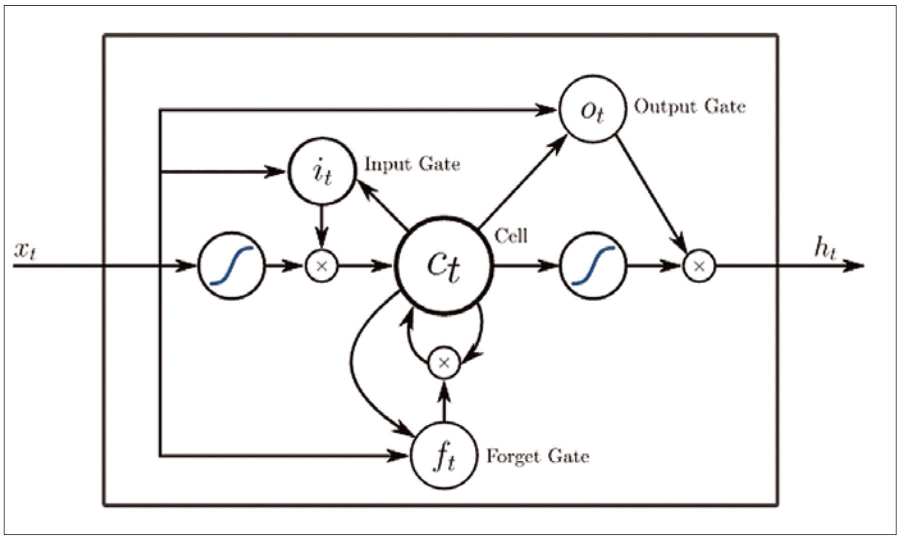


Figure 3: Structure of the LSTM cell. Source: Wikipedia

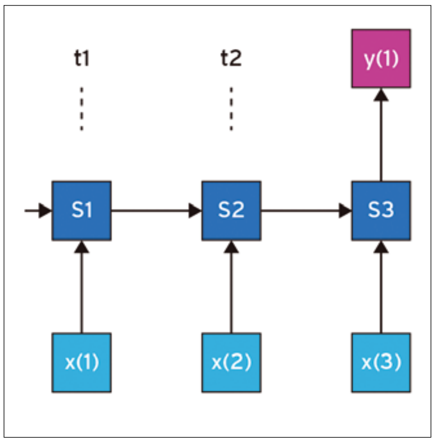


Figure 4: Temporally consecutive input values initially only change the current internal status and produce output every three time steps.

followed by *10*. The second row assigns a result of *12* to the subsequence *5*, *7*, *10*, therefore examining a window shifted by one step. The LSTM network uses this training data to adjust the parameters of its internal cells (Figure 3).

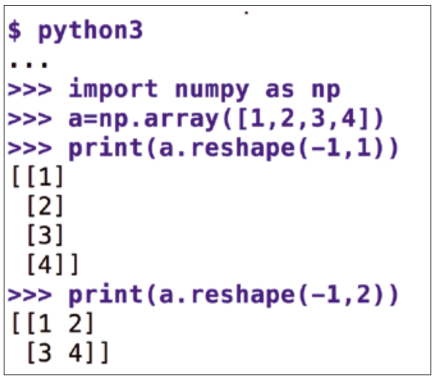
Unlike the neural network, not every input value produces an output value; instead, the LSTM keeps track of the current state in a hidden memory cell (Figure 4). It is only after receiving the third snippet of input and evaluating the carried over memory state that an output value (*y*(1)) is produced.

Reshaping Matrixes

To implement the LSTM network, Listing 1 [2] uses the Python `keras` library [3].

Listing 1: iq

```
01 #!/usr/bin/python3
02 import numpy as np
03 from sklearn.preprocessing \
04         import StandardScaler
05 from keras.models import Sequential
06 from keras.layers import Dense, Activation
07 from keras.layers import LSTM
08 import os
09
10 os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'
11
12 def window(npa, n=2):
13     for start in range(npa.size-n+1):
14         yield npa[start:start+n:1]
15
16 input_size=3
17
18 seq= np.array(
19         [2,5,7,10,12]).astype('float64')
20
21 print("learn input: " + str(seq))
22
23 scaler = StandardScaler()
24 seq = seq.reshape(-1,1)
25 seq = scaler.fit_transform(seq)
26 seq = seq.reshape(-1)
27
28 X=np.array([])
29 y=np.array([])
30
31 for chunk in window(seq, n=input_size+1):
32     X=np.append(X, chunk[:-1])
33     y=np.append(y, chunk[-1])
34
35 X=X.reshape((-1,input_size,1))
36 y=y.reshape((-1,1))
37
38 model = Sequential()
39 model.add(LSTM(
40           5,input_shape=(input_size,1)))
41 model.add(Dense(1))
42 model.add(Activation("linear"))
43 model.compile(loss="mean_squared_error",
44               optimizer="rmsprop")
45 model.fit(X,y, epochs=500, verbose=0)
46
47 print("\nresults:")
48 for input in X:
49     input=input.reshape(1,input_size,1)
50     pred=model.predict(input)
51     print(scaler.inverse_transform(
52           input.reshape(-1,1)))
53     print(scaler.inverse_transform(
54           pred.reshape(-1,1)))
55
56 test = seq[-input_size::1]
57 print(scaler.inverse_transform(
58       test.reshape(-1,1)))
59 test=test.reshape(1,input_size,1)
60 y1=model.predict(test)
61 print(scaler.inverse_transform(
62       y1.reshape(-1,1)))
```



```
$ python3
...
>>> import numpy as np
>>> a=np.array([1,2,3,4])
>>> print(a.reshape(-1,1))
[[1]
 [2]
 [3]
 [4]]
>>> print(a.reshape(-1,2))
[[1 2]
 [3 4]]
```

Figure 5: **A NumPy array assumes various dimensions with `reshape()`.**

Because many of its functions expect data in the form of matrixes of varying shapes and sizes, it makes sense to run a quick tutorial of the `reshape()` function exported by the NumPy array library first. A one-dimensional NumPy array (i.e., a vector) is converted by `reshape()`, as shown in Figure 5, to matrixes of predefined dimensions.

The first parameter passed to `reshape()` is the number of elements in the first dimension, followed by the number in the second, and so on. Because the number of elements is implicitly determined by the number of remaining elements after defining deeper dimensions, the former is often stated as `-1`. Then, the library fills the matrix with what's left over.

Called with just one parameter (`reshape(-1)`), the method converts a nested array structure back into a one-dimensional vector.

Counting Games

Called with an array like `[3,4,5,6,7]` (Figure 6), the script in Listing 1 relatively accurately produces the next sequence number (7.84 instead of 8). Listing 1 breaks down the series of numbers with the `window` function defined in line 12 into sliding windows of four (`[3,4,5,6]`, `[4,5,6,7]`); it stores the first three elements in the input vector `X` and the last element in the result vector `y`.

To prevent the LSTM network's internal weight adjustments from going haywire, the `StandardScaler` from the `sklearn` library normalizes the original input values to small positive and negative floating-point numbers around zero for both the input and the result vector, the latter containing the anticipated correct results required for supervised learning.

The `fit_transform()` method then applies the scaling procedure and standardizes the data. Later, before it comes to printing the results, `inverse_transform()` turns the tables and maps the data back to the original scale for an edifying inspection.

Lines 38 to 45 stack the individual layers of the LSTM network on top of one another. First, the LSTM core layer is added in lines 39 and 40, with five internal neurons. It is followed by the connected output layer of type `Dense` and the `Activation` function, which sets the response curve of the neurons used internally to `linear`, because this achieved the best results in testing.

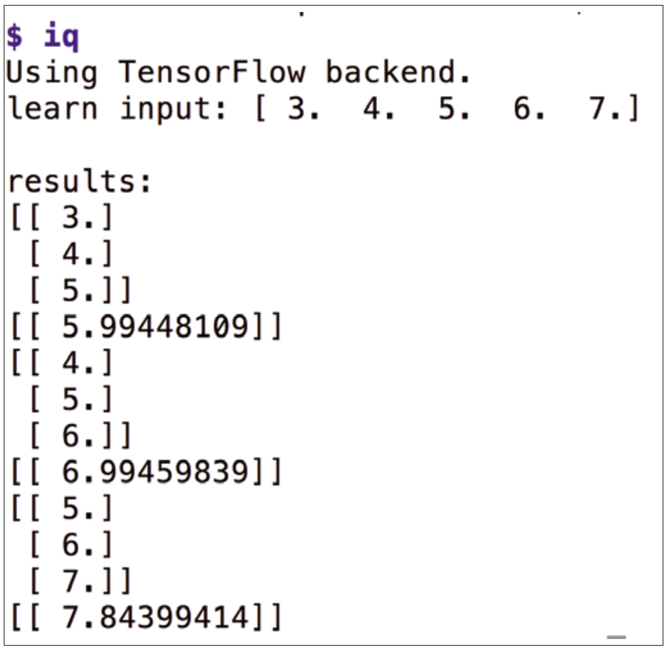
Off We Go!

The `compile()` method then readies the learning model for processing, determines `mean_squared_error` (deviations from optimal learning success are measured according to the mean square method) as the learning parameter, and sets `rmsprop`, a common method for neural networks, as the algorithm `optimizer`.

Line 45 then calls the model's `fit()` method, passes the learning data to it, and stipulates the number of learning iterations as `epoch = 500`. In my lab, a lower number affected the results negatively, but larger values of `epoch` did not achieve any greater learning success, since the system reached a steady state afterward, and the learning process visibly stagnated at a constant value for the `loss` function.

The section starting in line 48 of Listing 1 then monitors the predictions with the model in the current phase in the training, both for the training data and for previously unseen sequences that the system needs to guess without precedence in the training data.

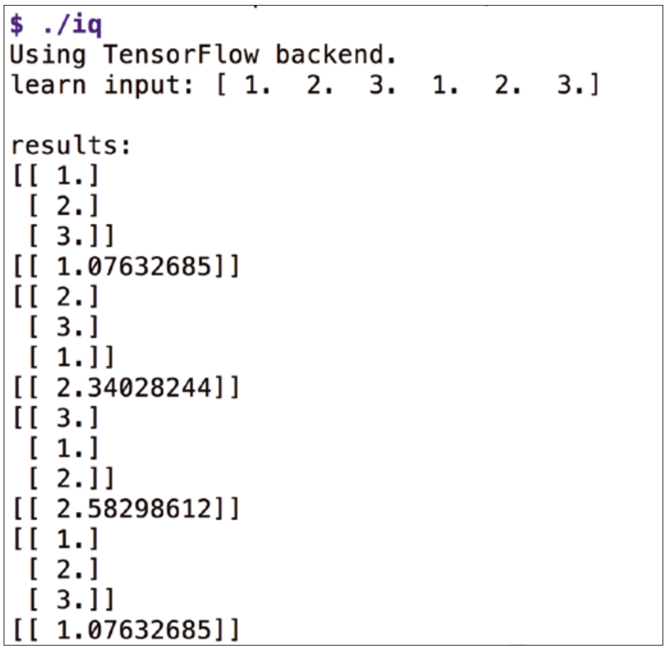
Figure 7 shows that the network does a good job, even for cyclic data (1,2,3,1,2,3), regardless of the length of the signal period, which is a major advantage over traditional neural networks, for which



```
$ iq
Using TensorFlow backend.
learn input: [ 3.   4.   5.   6.   7.]

results:
[[ 3.]
 [ 4.]
 [ 5.]]
[[ 5.99448109]]
[[ 4.]
 [ 5.]
 [ 6.]]
[[ 6.99459839]]
[[ 5.]
 [ 6.]
 [ 7.]]
[[ 7.84399414]]
```

Figure 6: **The LSTM network handles a simple sequence relatively accurately.**



```
$ ./iq
Using TensorFlow backend.
learn input: [ 1.  2.  3.  1.  2.  3.]

results:
[[ 1.]
 [ 2.]
 [ 3.]]
[[ 1.07632685]]
[[ 2.]
 [ 3.]
 [ 1.]]
[[ 2.34028244]]
[[ 3.]
 [ 1.]
 [ 2.]]
[[ 2.58298612]]
[[ 1.]
 [ 2.]
 [ 3.]]
[[ 1.07632685]]
```

Figure 7: **In a cyclic sequence, the algorithm predicts the next values more or less correctly.**

you have to state the periodicity in advance for reliable predictions.

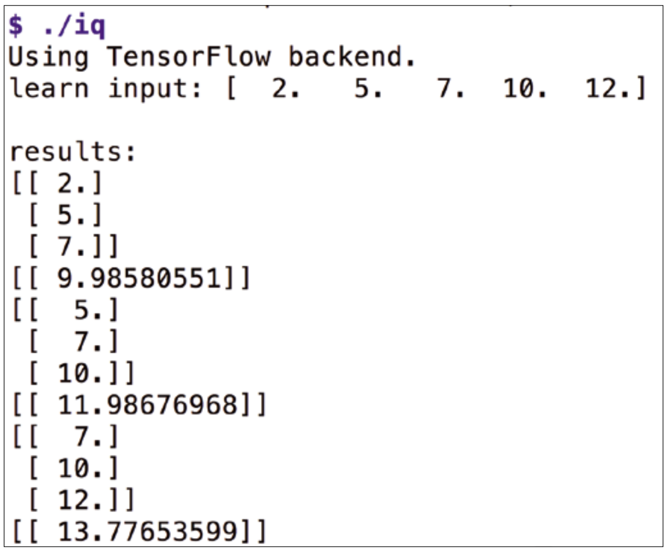
Fast as a Snail

You need to install the following libraries from the Python treasure trove to be able to install the `keras` library:

```
pip3 install --user keras pandas ➘
  tensorflow sklearn numpy
sudo apt-get install python-tk
```

The TensorFlow back end used by `keras` is not exactly speedy; it took a good 10 seconds for the program to get going and start the training on my five-year-old PC.

At the end of the day, the network did not perform particularly well in the intelligence test (Figure 8; see also the "Online Example" box). As you will have guessed, you need to alternately add 3 and 2 to the existing numbers in the `2,5,7,10,12` series. Since the jump from 10 to 12 was two units in length, a count of three thus needs to be added to predict the next number: 15 is the correct result. The network tended toward 14 most of the time, so it cannot compete with human intelligence (as of yet). ■■■



```
$ ./iq
Using TensorFlow backend.
learn input: [  2.   5.   7.  10.  12.]

results:
[[ 2.]
 [ 5.]
 [ 7.]]
[[ 9.98580551]]
[[  5.]
 [  7.]
 [ 10.]]
[[ 11.98676968]]
[[  7.]
 [ 10.]
 [ 12.]]
[[ 13.77653599]]
```

Figure 8: The LSTM network did not do well in the intelligence test, because it failed to identify the different growth rates.

Online Example

Mike Schilli demonstrates the example in a screencast (in German) at: *http://www.linux-magazin.de/Ausgaben/2017/10/plus*

Info

[1] Number sequence test: *https://www.fibonicci.com/numerical-reasoning/number-sequences-test/easy/*

[2] Listings for this article: *ftp://ftp.linux-magazine.com/pub/listings/linux-magazine.com/206/*

[3] Brownlee, Jason. *Long Short-Term Memory Networks with Python: Machine Learning Mastery*. 2017, *https://machinelearningmastery.com/lstms-with-python/*

Providers that protect against DDoS attacks

Shielded

To ward off DDoS attacks, websites and services often seek the protection of Internet giants, such as Amazon, but you have other ways to protect your connectivity. *By Konstantin Agouros*

Distributed Denial of Service (DDoS) attacks are a plague with consequences just as horrific as ransomware attacks. As a blackmail scheme, a targeted attack, or a form of virtual vandalism, a swarm of attackers floods a website with an influx of requests in an attempt to shut it down. In the past, attackers often used reflection attacks, which involve an attacker sending several packets with the IP address of the victim as the sender to servers, which then acknowledge the requests with long answers. Because of the spoofed IP address, however, these massive responses go to the victim's address.

Even companies with Internet connectivity in the 10 to 40Gbps range can be powerless against attacks with several hundreds of gigabits per second bandwidth. An Internet search with the keywords "biggest DDoS" regularly shows new and increasing values for such attacks; the peak is currently around 1Tbps. With the Internet of Things, attackers can now choose platforms that are much easier to use; for example, hacked surveillance cameras, refrigerators, and cheap routers have been responsible for recent attacks. In this article, I look at methods, providers, and the costs of protecting your connectivity.

The DDoS Family

Roughly three categories of DDoS attacks can be distinguished. Flooding, as the method described above is known, occurs when a large group of computers send many large data packets to a victim, exhausting the bandwidth or loading the infrastructure to its full capacity from the excessive number of individual packets.

In a state exhaustion attack, firewalls or load balancers maintain a state for each connection and enter these states simultaneously in a table. Keeping many connection attempts open will quickly fill this table; Linux acknowledges full tables with the message *nf_conntrack: table full, dropping packet*. If a customer tries to access your website at this stage, their request is rejected. I am aware of cases in which an attack with a bandwidth of 2Mbps was enough to take down a network behind a firewall. SYN flooding does the same thing with the kernel's TCP connection table. In the course of the three-way handshake when establishing a connection, the attacker only sends the first packet, which means an entry has been created for the connection. Once the table is full, any subsequent connection attempts receive a *Connection Refused* response.

Finally, in sophisticated attacks against vulnerabilities in applications, attackers often don't even need the first D in DDoS. As a penetration tester, I once encountered a search function in a web portal that was implemented in Java and launched a gigabyte-scale subthread for each search. A script with 10 queries per second paralyzed the entire platform.

Building Floodwalls

The firewall is regarded as a standard tool in network defense. However, because it only uses the data of a single package as a criterion for allowing or rejecting connections – not the quantity and volume of packets – the firewall is easily negotiated. Better firewalls (as in the Linux kernel) provide features for controlling the bandwidth per connection, but the CPU handles these and not the network hardware.

A separate class of devices filters like a firewall, but as a function of the quantity of packets (total and per connection). Dedicated application-specific integrated circuits (ASICs) or field-programmable gate arrays (FPGAs) define rules in the hardware, but the devices are quite expensive. If you want to intercept attacks of 150Gbps, you are looking at a six-figure sum.

Unfortunately these approaches are of little help if the defender of the network is sitting at the narrow end of the pipe. Before the device starts to filter, the pipe is already blocked. Although the measures protect the infrastructure, neither the internal network nor the company website can be reached from outside.

Basically, the defense must be implemented in two steps: First, you need to identify an attack as such; second, you need to divert the attack traffic. You can detect attacks by reference to various parameters:

- The quantity and size of packets per source and target port or IP address.
- The quantity and size of packets within a single connection; because a connection usually comprises just one request and one response packet, 20 request packages are already suspicious.
- The totals of these data.

Collecting Data

Only a few protocols and applications follow static rules. If up to 100 requests per hour is usually normal in a web store, that number might multiply quickly in the pre-Christmas period or during marketing campaigns. Regular shifts in values according to the time of day or day of week are also common. Statistics programs can discover what is normal and plot it against a timeline. If deviations from the norm then emerge, they can be classified as an attack.

Two approaches allow for collecting traffic data: (1) Installing a device inline (i.e., using two cables) that acts as a bridge between the inside and outside. The device then records all data in promiscuous mode and thus collects statistics. (2) Connecting such a device to the mirror port of a switch so that a failure does not interrupt the connection.

Another option is to use NetFlow [1], sFlow [2], or Ipfix (Internet Protocol flow information export) [3]. These protocols provide the sought-after connection data (with different levels of detail). Network components such as routers or switches send statistics to a receiver, which then carries out the statistical analysis and alerting.

Switching to Defense Mode

Defensive measures consist of blocking unwanted traffic. Usually, the admin filters out the attacked target IP address (or network block), either by creating access control lists (ACLs) on the routers or by defining a null route with the Border Gateway Protocol (BGP) [4]. This entry in the routing table rejects all packages trying to reach the attacked IP address on the router upstream. As a result, the attacked server is still offline, but the rest of the line remains free.

The BGP flowspec [5] extension, which allows admins to distribute ACLs containing target ports and protocols via the BGP protocol, is more surgical. If a Network Time Protocol (NTP) reflection attack sends many packets to UDP port 123 on the server and the upstream router only blocks it, third parties can still reach the server via ports 80 and 443 (TCP); however, few router manufacturers support this strategy.

Generally speaking, hardly any providers let a customer distribute filter rules of this kind to its routers. If the provider has a powerful DDoS appliance, it will route the customer's infected traffic through and enable matching rules so that only the desired data reaches the victim.

The defenses against flooding must be made at the thick end of the pipe (i.e., the provider side) or at the data center where your web pages are hosted. This begs the question as to the bandwidth of the data center's Internet connection. A 40Gbps connection may be completely sufficient for normal operation of a data center, but attacking it is child's play in the DDoS world. I have experienced an attack on a customer of more than 200Gbps against a single server, without any commercial interests being involved.

Data Scrubbing

Individual providers offer their customers DDoS protection for a charge. Some purchase expensive hardware that can do this and then rent it out to customers.

Cloud mitigation is most common, wherein a dedicated provider "scrubs" the traffic by running a farm of anti-DDoS appliances in a broadband-connected data center to which the victim's traffic is rerouted for cleaning and then routed back again to the target through a tunnel.

There are two options for redirecting the traffic. On the one hand, it can happen on the DNS level. If the attack is directed against *www.example.com*, you can redirect the DNS entry to an IP in the scrubbing system. However, it takes a while for the rest of the Internet to discover the change. Admins need to keep the time-to-live for DNS requests low, so name server caching does not delay the redirection.

On the other hand, routing protocols (usually BGP) can redirect traffic. Note that this does not work with individual IPs, but only with full network blocks. The advantage is that these path changes propagate more quickly on the network.

The Narrow End

Defenses against state exhaustion attacks can also be implemented on the narrow end of the line. If Netfilter is used as a firewall, the first step is to check the `net.netfilter.nf_conntrack_max` (or `/proc/sys/net/netfilter/nf_conntrack_max`) system parameter, which specifies the maximum number of connections. The default value varies between `32768` and `65536`, which an attacker can exhaust relatively quickly.

You can use `sysctl -w` to increase this value up to 2GB. However, the machine does need to have enough physical memory to store this number of entries. An entry consumes a good 300 bytes (in reality, it is somewhat more complicated; a blog post [6] describes it more accurately), meaning that more than 700GB of RAM would be needed for the 2 billion entries. According to the cited page, the kernel can handle 1.7 million entries with around 512MB set aside for connection tracking, which is quite a few orders of magnitude greater than the default value.

The kernel can use SYN cookies to defend against SYN flooding. The server sends a response but does not create an entry in the table. In SYN flooding attacks, the third packet is missing. If it does arrive, the server recognizes the situation and creates the entry. To make this possible, the kernel constructs the sequence numbers so that it recognizes them in the ACK packet.

DDoS appliances can do this, too, but also offer the ability to work as TCP proxies, which means that they first complete the handshake acting as proxies and only open the connection to the server if it actually works. Most modern firewall systems can also use this approach.

Defenses against more powerful attacks also exist at the same level. For web applications, a reverse proxy, such as an Apache with `mod_security`, can fill the breech. The module makes it possible to provide rules for, and limit the number of, requests per source IP address. However, you have to configure this by hand for each URL.

DDoS appliances allow you to set limits at the transaction level, but you should first check with the manufacturers as to which protocols they support. A10 Networks [7] even offers to enable Captchas dynamically when a threshold is exceeded and then only allow further requests to the affected source address in question once a human user has been confirmed by the Captcha.

AWS Shield

The Amazon Web Services (AWS) Shield [8] provides protection against DDoS attacks (Figure 1). The Standard protection is available to any AWS customer. The product includes detection of network flow data and automatic mitigation of DDoS attacks against SYN flooding or UDP reflection attacks. However, you do not receive information about a successful defense. If you choose the AWS Shield Advanced product, you receive the following additional features for around $3,025 per month plus charges for data transfer:

- In addition to connection data at the network level, Amazon collects and analyzes transaction logs at the application level.
- Access to advanced scrubbing capacities.
- Notification of attacks on ISO Layers 3 and 4, as well as data about the type of attack.
- Reports for ISO Layers 3, 4, and 7.
- Incident management by the Amazon DDoS response team.
- If necessary, manual mitigation.
- Manual analysis after the attack.
- Reimbursement for costs incurred by the attack associated with CloudFront, Route 53, and ELB services.

Of import is that Amazon only protects what runs on Amazon. Although it is possible to protect data traffic on your own servers using services such as CloudFront or a reverse proxy and to protect your own network connection in another way, you cannot fight off targeted attacks.

Arbor Appliance

Arbor [9] produces DDoS detection and defense systems with its own hardware. The company provides traffic scrubbing systems in data centers in the US, Europe, and Asia. The service only defends against attacks detected by the customer.

Licensing depends on the bandwidth of clean traffic. If a customer has a 1Gbps line, they pay for the 1Gb package. Additionally, Arbor offers packages for a monthly fee or for defense against 12 attacks per year. The packages each include the protection of a */24* network, including five DNS names, and are expandable. Arbor lets you link triggering of the defenses to your appliance. If an attack is detected, the Cloud defense is triggered and provides reports for the attacks.

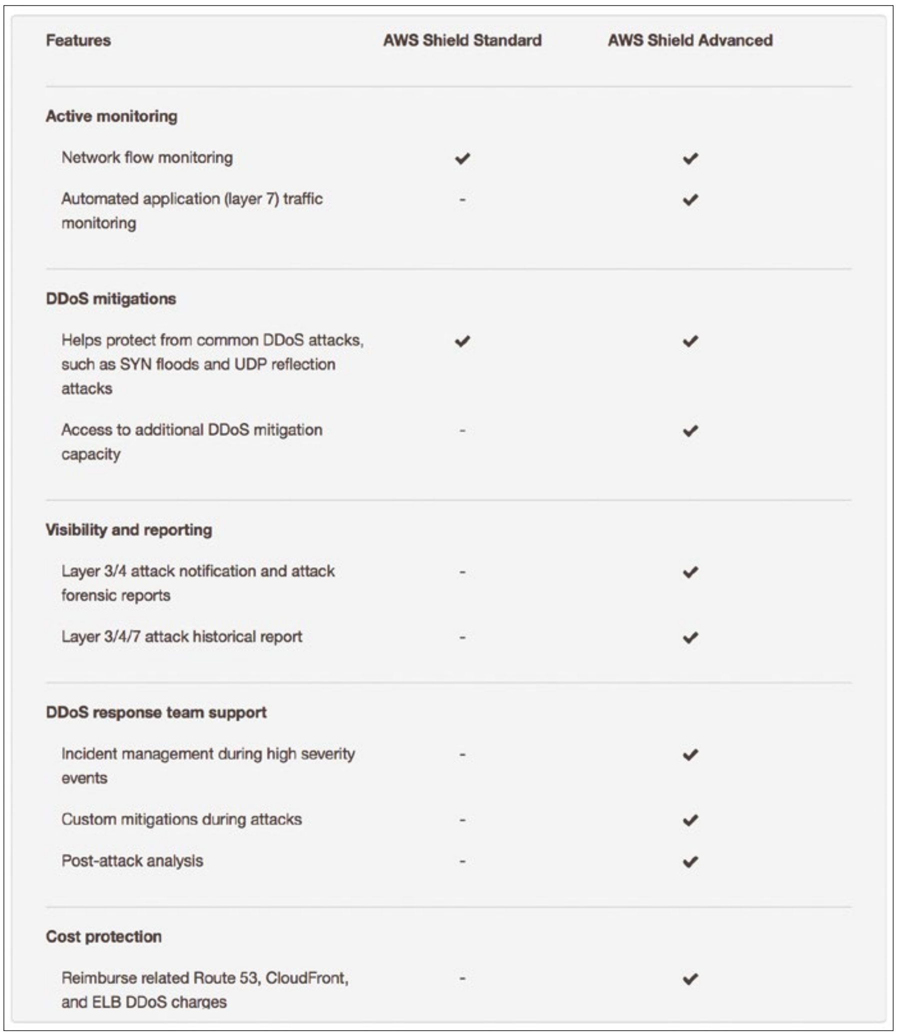
Link11

Link11 [10] also offers protection by BGP, by DNS redirection, and by connecting directly to host service provider data centers. The licensing differs in the DNS and BGP versions, but both define clean traffic as a 95th percentile of normal traffic (without attacks).

On DNS, Link11 counts how many IP addresses it protects on the original systems, with clean traffic speed levels of 25, 50, 100, 150, and 250Mbps. In the case of BGP, the size, in terms of a netmask, counts as a parameter, and the count starts with a */24* network as the first. The second is again the protected bandwidth (Link11 differentiates between symmetric and asymmetric routing); the scales are 250, 500, and 1,000Mbps.

Akamai

Akamai [11] operates one of the largest content delivery networks (CDNs)



Features	AWS Shield Standard	AWS Shield Advanced
Active monitoring		
Network flow monitoring	✓	✓
Automated application (layer 7) traffic monitoring	-	✓
DDoS mitigations		
Helps protect from common DDoS attacks, such as SYN floods and UDP reflection attacks	✓	✓
Access to additional DDoS mitigation capacity	-	✓
Visibility and reporting		
Layer 3/4 attack notification and attack forensic reports	-	✓
Layer 3/4/7 attack historical report	-	✓
DDoS response team support		
Incident management during high severity events	-	✓
Custom mitigations during attacks	-	✓
Post-attack analysis	-	✓
Cost protection		
Reimburse related Route 53, CloudFront, and ELB DDoS charges	-	✓

Figure 1: Amazon protects customers against DDoS attacks – to an extent. For more protection, you will have to dig very deeply into your pockets.

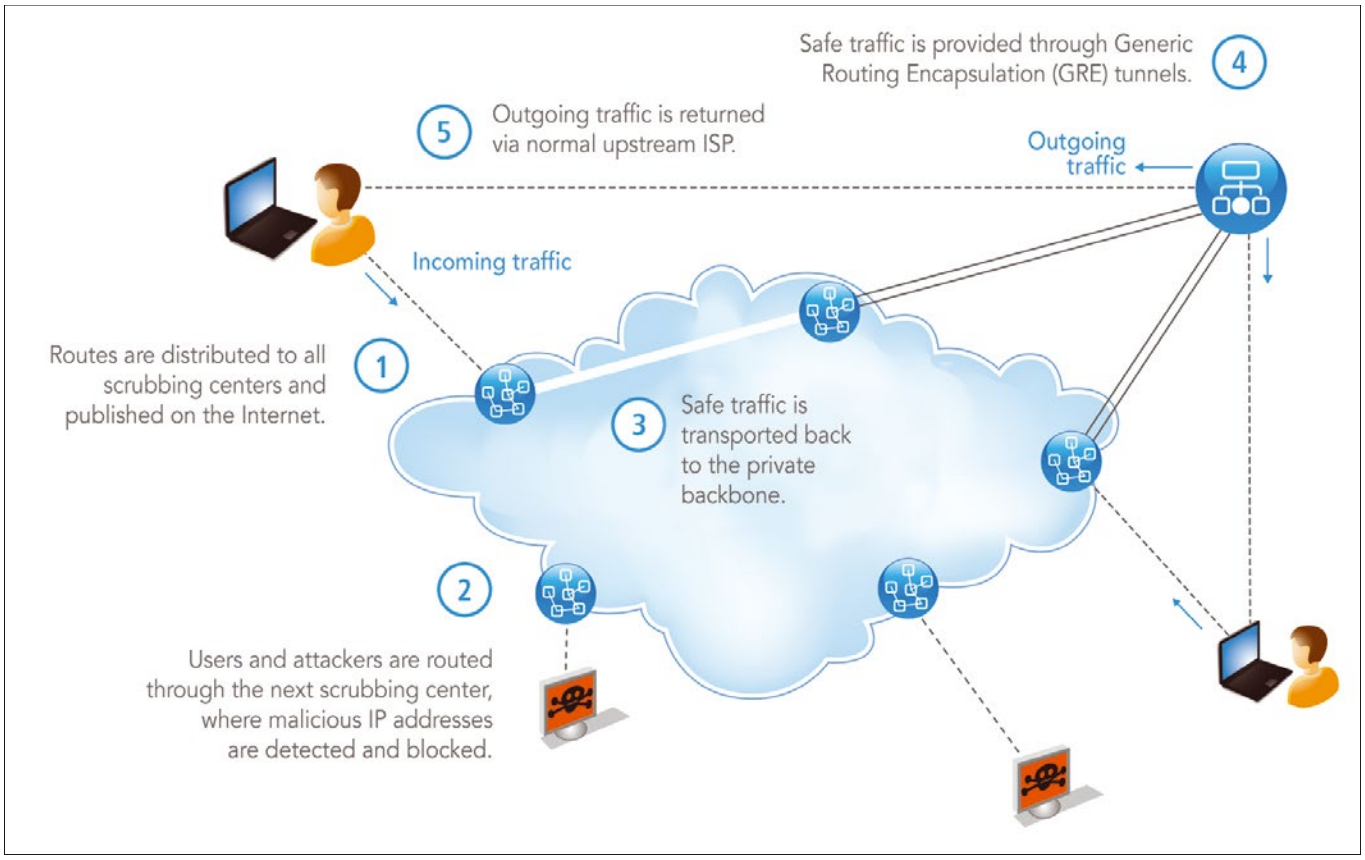

Safe traffic is provided through Generic Routing Encapsulation (GRE) tunnels.
4
5
Outgoing traffic is returned via normal upstream ISP.
Outgoing traffic
Incoming traffic
Routes are distributed to all scrubbing centers and published on the Internet.
1
3
Safe traffic is transported back to the private backbone.
2
Users and attackers are routed through the next scrubbing center, where malicious IP addresses are detected and blocked.

Figure 2: **Akamai, known for its global CDN, provides customers with protection against attacks for a fee.**

worldwide and uses its network to offer DDoS protection in the cloud. In addition to the DNS and BGP variants, Akamai sells proxy protection, which places your individual applications (e.g., ports on IP addresses) under their protection.

Like Link11, the Prolexic Connect product provides data centers with the option to slip under the wing of CDN (Figure 2). Like Link11, Akamai structures the billing model as a subscription and uses the 95th percentile of clean or normal incoming traffic to calculate a price. In the BGP variant, the number of */24* networks and the number of protected sites still play a role.

Conclusions

If you run your infrastructure with AWS, you will be grateful for the basic protection provided. Advanced protection is quite pricey, though. You should consider whether Amazon's presence is so relevant for your business that it justifies a premium of more than $3,025 per month. The option to pay per attack can quickly have an adverse effect on the purse of the victims. If you want to get away from Amazon completely, you need to look around for another provider in terms of DDoS protection.

The information in this article can probably steer you in the right direction, but it does not present a universal solution. Because the pricing structures differ considerably between providers, you should first analyze your own threat situation closely. Have you already experienced attacks or been threatened with attacks? If so, was the entire company threatened or just one department? In the latter case, even Amazon can be useful, especially if the volume of regular traffic is not too high.

Even if you do not take precautions against DDoS attacks with specific measures, you should at least have a contingency plan up your sleeve and discover your options beforehand. Link11 and Arbor offer emergency links on their websites in the event of an attack. For those at risk, there is no time to lose. ■■■

Author

Konstantin Agouros works as Head of Open Source Projects at Matrix Technology AG, where he and his team advise customers on open source and cloud topics. His new book, *Software Defined Networking: SDN-Praxis mit Controllern und OpenFlow* [Practice with Controllers and OpenFlow], is published by De Gruyter.

Info

[1] NetFlow: *https://en.wikipedia.org/wiki/Netflow*
[2] sFlow: *http://www.sflow.org*
[3] Ipfix: *https://tools.ietf.org/html/rfc7011*
[4] Border Gateway Protocol: *https://tools.ietf.org/html/rfc1772*
[5] Flowspec: *https://tools.ietf.org/html/rfc5575*
[6] Netfilter Conntrack memory usage: *https://johnleach.co.uk/words/372/netfilter-conntrack-memory-usage*
[7] A10: *https://www.a10networks.com/products/thunder-series/ddos-detection-protection-mitigation*
[8] AWS Shield: *https://aws.amazon.com/en/shield/*
[9] Arbor: *https://www.arbornetworks.com/ddos-protection-products*
[10] Link11: *https://www.link11.com/en/*
[11] Akamai: *https://www.akamai.com/us/en/resources/ddos.jsp*

Record screencasts with Peek on Gnome

In the Can

A screencast shows what happens on the desktop. Peek lets you create screencasts in the blink of an eye and export them to popular formats. *By Christoph Langner*

As the famous saying goes, a picture is worth a thousand words. But how many words can a video save you? A million, maybe? In many situations, a short screencast (i.e., a video of desktop events) gives a far better explanation of a problem or an action than wordy text with images. A wide range of tools is available for this purpose.

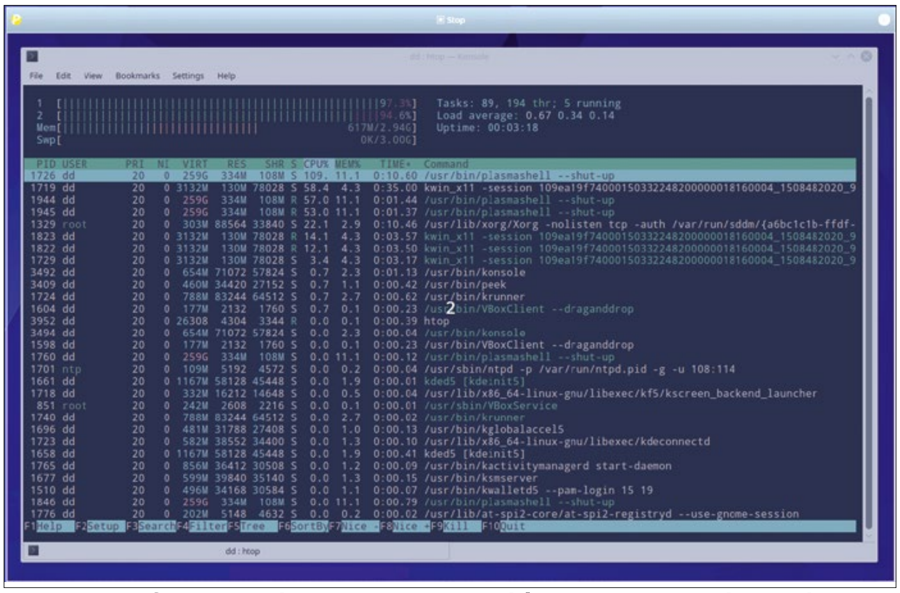
The range extends from SimpleScreenRecorder [1] to recordMyDesktop [2]. Compared with these candidates, the fairly recent Peek [3] has a very small feature set, but the program is not trying to compete with the more established applications. Originally, it simply recorded the desktop as a GIF, thus producing videos that were easy to embed into web pages. However, Peek now also supports more traditional video formats such as WebM and MP4.

Recording the Desktop

In terms of the interface, Peek is deliberately oriented on the LICEcap [4] screencast tool for Windows. The program shows a scalable transparent window that is always in the foreground on top of all your other applications. Everything inside the window frame, is grabbed as a video by the software when you click *Record*. After pressing *Stop* (Figure 1), Peek immediately saves the results on the hard disk.

The program has supported real video formats (e.g., WebM and MP4) since version 1.0.0, which was released in March 2017. To set the output format, as shown in Figure 2, click on the Peek icon in the upper left corner and choose *Preferences*.

You can open these formats – in a style typical of modern Gnome applications – from the application menu next to the Activities button in the desktop's header bar. Then adjust the additional parameters, if necessary, such as the *Delay in seconds before the recording starts* and the *Framerate*, or influence the size of the recording with *Resolution downsampling*.

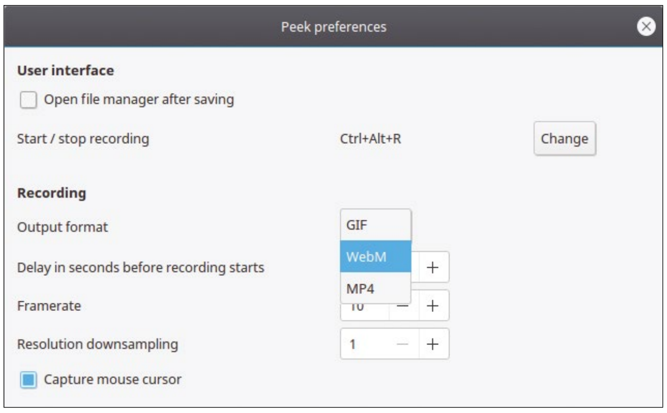


```
  1  [|||||||||||||||||||||||||||||||||||||||||||||||||||97.3%]   Tasks: 89, 194 thr; 5 running
  2  [|||||||||||||||||||||||||||||||||||||||||||||||||||94.6%]   Load average: 0.67 0.34 0.14
  Mem[|||||||||||||||||||||||||||||            617M/2.94G]   Uptime: 00:03:18
  Swp[                                          0K/3.00G]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 1726 dd         20   0  259G  334M  108M S 109. 11.1  0:10.60 /usr/bin/plasmashell --shut-up
 1719 dd         20   0 3132M  130M 78028 S 58.4  4.3  0:35.00 kwin_x11 -session 109ea19f7400015033224820000018160004_1508482020_9
 1944 dd         20   0  259G  334M  108M R 57.0 11.1  0:01.44 /usr/bin/plasmashell --shut-up
 1945 dd         20   0  259G  334M  108M R 53.0 11.1  0:01.37 /usr/bin/plasmashell --shut-up
 1329 root       20   0  303M 88564 33840 S 22.1  2.9  0:10.46 /usr/lib/xorg/Xorg -nolisten tcp -auth /var/run/sddm/{a6bc1c1b-ffdf-
 1823 dd         20   0 3132M  130M 78028 R 14.1  4.3  0:03.57 kwin_x11 -session 109ea19f7400015033224820000018160004_1508482020_9
 1822 dd         20   0 3132M  130M 78028 R 12.1  4.3  0:03.50 kwin_x11 -session 109ea19f7400015033224820000018160004_1508482020_9
 1729 dd         20   0 3132M  130M 78028 S  3.4  4.3  0:03.17 kwin_x11 -session 109ea19f7400015033224820000018160004_1508482020_9
 3492 dd         20   0  654M 71072 57824 S  0.7  2.3  0:01.13 /usr/bin/konsole
 3409 dd         20   0  460M 34420 27152 S  0.7  1.1  0:00.42 /usr/bin/peek
 1724 dd         20   0  788M 83244 64512 S  0.7  2.7  0:00.62 /usr/bin/krunner
 1604 dd         20   0  177M  2132  1760 S  0.7  0.1  0:00.23 /us bin/VBoxClient --draganddrop
 3952 dd         20   0 26308  4304  3344 R  0.0  0.1  0:00.39 htop
 3494 dd         20   0  654M 71072 57824 S  0.0  2.3  0:00.04 /usr/bin/konsole
 1598 dd         20   0  177M  2132  1760 S  0.0  0.1  0:00.23 /usr/bin/VBoxClient --draganddrop
 1760 dd         20   0  259G  334M  108M S  0.0 11.1  0:00.12 /usr/bin/plasmashell --shut-up
 1701 ntp        20   0  109M  5192  4572 S  0.0  0.2  0:00.04 /usr/sbin/ntpd -p /var/run/ntpd.pid -g -u 108:114
 1661 dd         20   0 1167M 58128 45448 S  0.0  1.9  0:00.01 kded5 [kdeinit5]
 1718 dd         20   0  332M 16212 14648 S  0.0  0.5  0:00.04 /usr/lib/x86_64-linux-gnu/libexec/kf5/kscreen_backend_launcher
  851 root       20   0  242M  2608  2216 S  0.0  0.1  0:00.01 /usr/sbin/VBoxService
 1740 dd         20   0  788M 83244 64512 S  0.0  2.7  0:00.02 /usr/bin/krunner
 1696 dd         20   0  481M 31788 27408 S  0.0  1.0  0:00.13 /usr/bin/kglobalaccel5
 1723 dd         20   0  582M 38552 34400 S  0.0  1.3  0:00.10 /usr/lib/x86_64-linux-gnu/libexec/kdeconnectd
 1658 dd         20   0 1167M 58128 45448 S  0.0  1.9  0:00.41 kded5 [kdeinit5]
 1765 dd         20   0  856M 36412 30508 S  0.0  1.2  0:00.09 /usr/bin/kactivitymanagerd start-daemon
 1677 dd         20   0  599M 39840 35140 S  0.0  1.3  0:00.15 /usr/bin/ksmserver
 1510 dd         20   0  496M 34168 30584 S  0.0  1.1  0:00.07 /usr/bin/kwalletd5 --pam-login 15 19
 1846 dd         20   0  259G  334M  108M S  0.0 11.1  0:00.79 /usr/bin/plasmashell --shut-up
 1776 dd         20   0  202M  5148  4632 S  0.0  0.2  0:00.02 /usr/lib/at-spi2-core/at-spi2-registryd --use-gnome-session
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```

Figure 1: Before recording, a programmable timer counts down the screencast, giving you time to arrange everything.

Lead Image © Roman Fedin, 123RF, 123RF.com



Peek preferences

User interface

☐ Open file manager after saving

Start / stop recording: Ctrl+Alt+R (Change)

Recording

Output format: GIF / WebM / MP4 (WebM selected)

Delay in seconds before recording starts

Framerate: 10

Resolution downsampling: 1

☑ Capture mouse cursor

Figure 2: You can select the video format in the Peek preferences dialog. In addition to GIF, Peek supports WebM and MP4.

Listing 1: Add PPA

```
sudo add-apt-repository ppa:peek-developers/stable
sudo apt update
sudo apt install peek
```

For the GIF format, especially, you should keep in mind that it was not originally designed for recording: Recording the entire desktop in full HD at 30 frames per second (FPS), will result in huge files.

Therefore, choose only the snippet that you actually want to view later. When scaling the window, the size display can help you align the frame precisely (Figure 3). Additionally, you can reduce the framerate to about 10fps, and, if necessary, use the *Resolution downsampling* option to scale the recording by an integer factor.

Installation

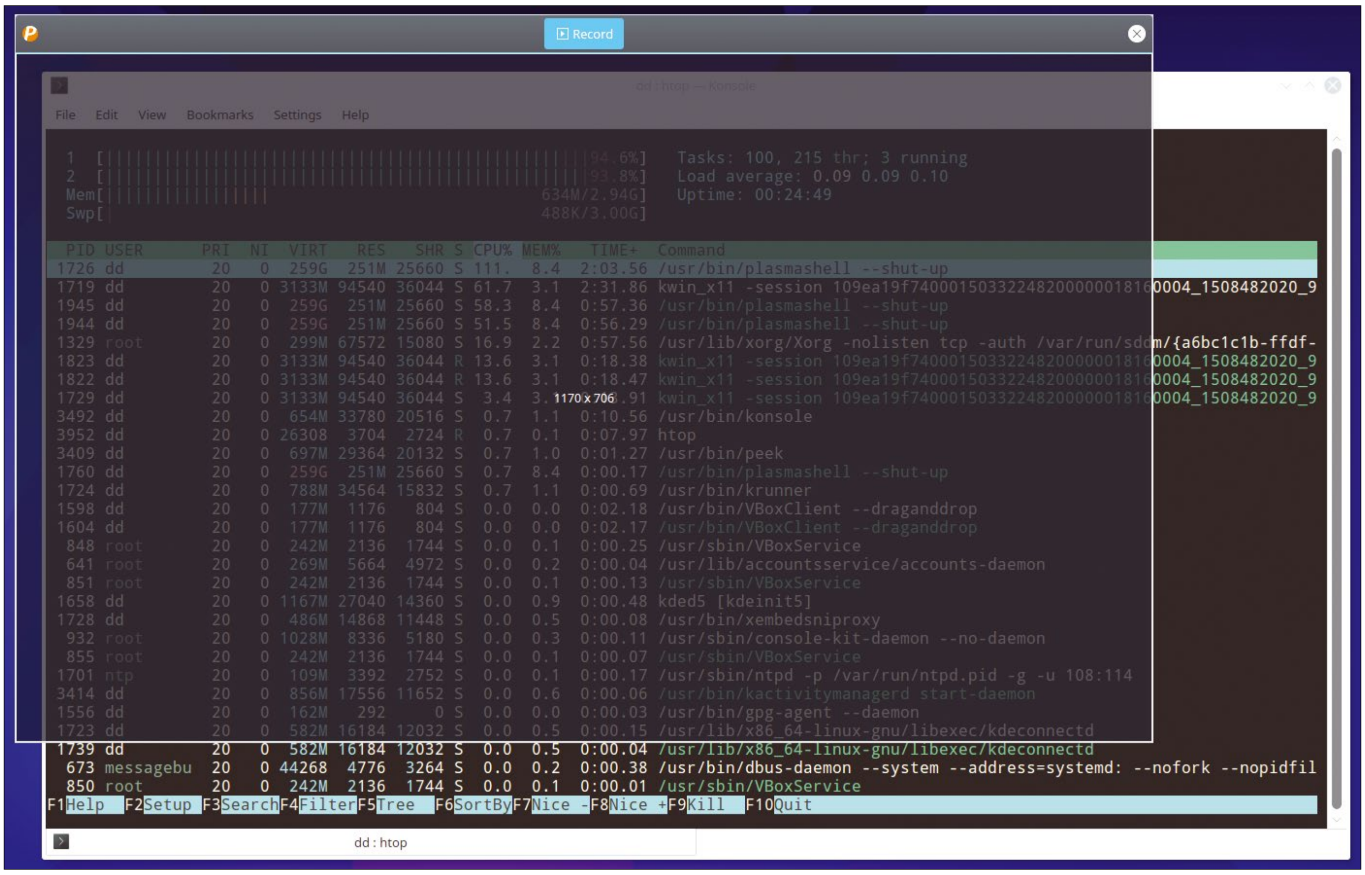
Peek's first commits date back to December 2015; since then, the developer has worked quite actively on the program. At the beginning of 2017, the work started to pick up speed: Changes to the code appeared daily on GitHub. As a fairly young program, it is not currently found in the repositories of the major distributions.

However, Ubuntu has a Personal Package Archive (PPA) repository (Listing 1). On Arch Linux, you can simply install the application from the Arch User Repository (AUR) [5]. For other distributions, such as Fedora, Debian, or Solus, the developer provides instructions for installation on the project page.

What is currently causing Peek difficulties is the change to the new display server, Wayland. For security reasons, Wayland isolates individual applications on the desktop from each other. Software is not allowed to read the content of another program's window. Thus, screenshots of the entire desktop are no longer easily achievable.

However, Peek is not the only applicatoin facing this problem; other screenshot tools, such as Shutter [6], are also affected. Additionally, Wayland no longer delivers absolute coordinates on the position of the application window, which – at least theoretically – puts it in a position to arrange windows on round or curved 3D displays.

Unless you launch Gnome under the classic X server via the display manager using Gnome on Xorg, Peek therefore needs to revert to the XWayland [7] compatibility layer, which now happens automatically when you start



```
  1  [|||||||||||||||||||||||||||||||||||||||||||||94.6%]   Tasks: 100, 215 thr; 3 running
  2  [|||||||||||||||||||||||||||||||||||||||||||||93.8%]   Load average: 0.09 0.09 0.10
  Mem[||||||||||||||||                     634M/2.94G]   Uptime: 00:24:49
  Swp[|                                     488K/3.00G]

  PID USER      PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 1726 dd         20   0  259G  251M 25660 S 111.  8.4  2:03.56 /usr/bin/plasmashell --shut-up
 1719 dd         20   0 3133M 94540 36044 S 61.7  3.1  2:31.86 kwin_x11 -session 109ea19f74000150332248200000018160004_1508482020_9
 1945 dd         20   0  259G  251M 25660 S 58.3  8.4  0:57.36 /usr/bin/plasmashell --shut-up
 1944 dd         20   0  259G  251M 25660 S 51.5  8.4  0:56.29 /usr/bin/plasmashell --shut-up
 1329 root       20   0  299M 67572 15080 S 16.9  2.2  0:57.56 /usr/lib/xorg/Xorg -nolisten tcp -auth /var/run/sddm/{a6bc1c1b-ffdf-
 1823 dd         20   0 3133M 94540 36044 R 13.6  3.1  0:18.38 kwin_x11 -session 109ea19f74000150332248200000018160004_1508482020_9
 1822 dd         20   0 3133M 94540 36044 R 13.6  3.1  0:18.47 kwin_x11 -session 109ea19f74000150332248200000018160004_1508482020_9
 1729 dd         20   0 3133M 94540 36044 S  3.4  3. 1170 x 706 .91 kwin_x11 -session 109ea19f74000150332248200000018160004_1508482020_9
 3492 dd         20   0  654M 33780 20516 S  0.7  1.1  0:10.56 /usr/bin/konsole
 3952 dd         20   0 26308  3704  2724 R  0.7  0.1  0:07.97 htop
 3409 dd         20   0  697M 29364 20132 S  0.7  1.0  0:01.27 /usr/bin/peek
 1760 dd         20   0  259G  251M 25660 S  0.7  8.4  0:00.17 /usr/bin/plasmashell --shut-up
 1724 dd         20   0  788M 34564 15832 S  0.7  1.1  0:00.69 /usr/bin/krunner
 1598 dd         20   0  177M  1176   804 S  0.0  0.0  0:02.18 /usr/bin/VBoxClient --draganddrop
 1604 dd         20   0  177M  1176   804 S  0.0  0.0  0:02.17 /usr/bin/VBoxClient --draganddrop
  848 root       20   0  242M  2136  1744 S  0.0  0.1  0:00.25 /usr/sbin/VBoxService
  641 root       20   0  269M  5664  4972 S  0.0  0.2  0:00.04 /usr/lib/accountsservice/accounts-daemon
  851 root       20   0  242M  2136  1744 S  0.0  0.1  0:00.13 /usr/sbin/VBoxService
 1658 dd         20   0 1167M 27040 14360 S  0.0  0.9  0:00.48 kded5 [kdeinit5]
 1728 dd         20   0  486M 14868 11448 S  0.0  0.5  0:00.08 /usr/bin/xembedsniproxy
  932 root       20   0 1028M  8336  5180 S  0.0  0.3  0:00.11 /usr/sbin/console-kit-daemon --no-daemon
  855 root       20   0  242M  2136  1744 S  0.0  0.1  0:00.07 /usr/sbin/VBoxService
 1701 ntp        20   0  109M  3392  2752 S  0.0  0.1  0:00.17 /usr/sbin/ntpd -p /var/run/ntpd.pid -g -u 108:114
 3414 dd         20   0  856M 17556 11652 S  0.0  0.6  0:00.06 /usr/bin/kactivitymanagerd start-daemon
 1556 dd         20   0  162M   292     0 S  0.0  0.0  0:00.03 /usr/bin/gpg-agent --daemon
 1723 dd         20   0  582M 16184 12032 S  0.0  0.5  0:00.15 /usr/lib/x86_64-linux-gnu/libexec/kdeconnectd
 1739 dd         20   0  582M 16184 12032 S  0.0  0.5  0:00.04 /usr/lib/x86_64-linux-gnu/libexec/kdeconnectd
  673 messagebu  20   0 44268  4776  3264 S  0.0  0.2  0:00.38 /usr/bin/dbus-daemon --system --address=systemd: --nofork --nopidfil
  850 root       20   0  242M  2136  1744 S  0.0  0.1  0:00.01 /usr/sbin/VBoxService
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```

Figure 3: The scalable frame with a size display helps you align the area for the recording.

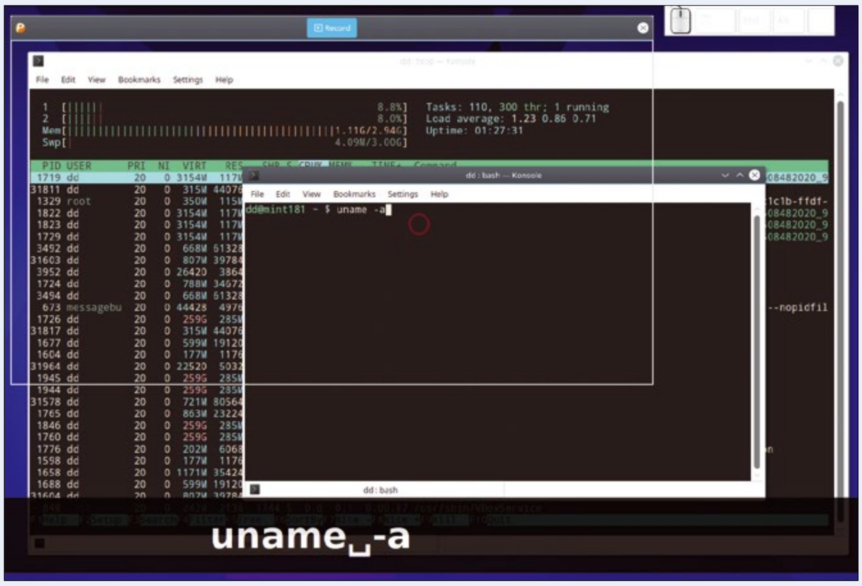
Input Visualization

If you want to demonstrate a certain step to your viewers, rather than just grabbing a video of a program, it is useful to see input and mouse clicks. Linux offers two applications to help you with this include key-mon [8] and screenkey [9]. Key-mon simply displays a small window with an iconized mouse and the Ctrl, Alt, and Shift keys. However, screenkey displays all input from the keyboard in a bar.

The applications can complement each other, if necessary. On one hand, key-mon illustrates mouse actions. Starting the program with the command

```
key-mon --visible_click
```

draws a red circle around the mouse cursor, even for clicks. On the other hand, screenkey helps you with entries such as Ctrl+C or helps improve the viewer's understanding of the commands (Figure 4).



```
  1  [|||||                                   8.8%]   Tasks: 110, 300 thr; 1 running
  2  [|||||                                   8.0%]   Load average: 1.23 0.86 0.71
  Mem[|||||||||||||||||||||||||||||||1.11G/2.94G]   Uptime: 01:27:31
  Swp[                                 4.09M/3.00G]

  PID USER      PRI  NI  VIRT   RES
 1719 dd         20   0 3154M  117M
31811 dd         20   0  315M 44076
 1329 root       20   0  350M  115M
 1822 dd         20   0 3154M  117M
 1823 dd         20   0 3154M  117M
 1729 dd         20   0 3154M  117M
 3492 dd         20   0  668M 61328
31603 dd         20   0  807M 39784
 3952 dd         20   0 26420  3864
 1724 dd         20   0  788M 34672
 3494 dd         20   0  668M 61328
  673 messagebu  20   0 44428  4976
 1726 dd         20   0  259G  285M
31817 dd         20   0  315M 44076
 1677 dd         20   0  599M 19120
 1604 dd         20   0  177M  1176
31964 dd         20   0 22520  5032
 1945 dd         20   0  259G  285M
 1944 dd         20   0  259G  285M
31578 dd         20   0  721M 80564
 1765 dd         20   0  863M 23224
 1846 dd         20   0  259G  285M
 1760 dd         20   0  259G  285M
 1776 dd         20   0  202M  6068
 1598 dd         20   0  177M  1176
 1658 dd         20   0 1171M 35424
 1688 dd         20   0  599M 19120
```

```
dd@mint181 ~ $ uname -a
```

Figure 4: key-mon and screenkey help you visualize mouse clicks and keystrokes in a video.

Wayland; however, if needed, you can call the application directly under XWayland with:

```
GDK_BACKEND=x11 peek
```

After fulfilling this condition, working with the program is very easy: Launch, align the frame, and record the screencast. For more tips on screencasts, check out the "Input Visualization" box.

Conclusions

The enhancement list, a roadmap for Peek on GitHub [10], lists a large number of innovations and ideas for the future of the tool. The most important of these has already been implemented in the form of support for a genuine video format (WebM), but other useful features, like the ability to pause the recording, a picker for selecting a screen area, or a progress bar when rendering the recording, are still on the roadmap. ■■■

Info

[[1] SimpleScreenRecorder: *http://www.maartenbaert.be/simplescreenrecorder*

[2] recordMyDesktop: *http://recordmydesktop.sourceforge.net*

[3] Peek: *https://github.com/phw/peek*

[4] LICEcap: *http://www.cockos.com/licecap*

[5] Peek in the AUR: *https://aur.archlinux.org/packages/peek*

[6] "Shutter does not work in Wayland": *https://bugs.launchpad.net/shutter/+bug/1502263*

[7] XWayland: *https://wayland.freedesktop.org/xserver.html*

[8] key-mon: *https://code.google.com/archive/p/key-mon*

[9] screenkey: *https://github.com/wavexx/screenkey*

[10] Future improvements: *https://github.com/phw/peek/issues?q=is%3Aissue+is%3Aopen+label%3Aenhancement*

The sys admin's daily grind: inxi

Info Tubbies

The name of the tool that columnist Charly Kühnast recommends this month may sound like the Teletubbies, but it is but far from infantile when it comes to functionality. In fact, inxi provides detailed and precisely controllable hardware and system usage information for the host computer. *By Charly Kühnast*

Every admin knows how to retrieve information about the system on which they are working. How many cores does the CPU have? `cat /proc/cpuinfo`! Is `eth3` a gigabit interface? `ip l sh`! But instead of many tools, you can just use one: inxi [1].

Suppose I need an overview of a machine with which I don't normally work. Then, I call inxi without any parameters and get some basic information about the hardware (CPU, clock speed, RAM, and disk size) and the system (kernel and shell processes). If I want to see a few details, the `-F` parameter provides information on the video and audio hardware, partitioning, RAID, temperatures, and fan speeds (Figure 1).

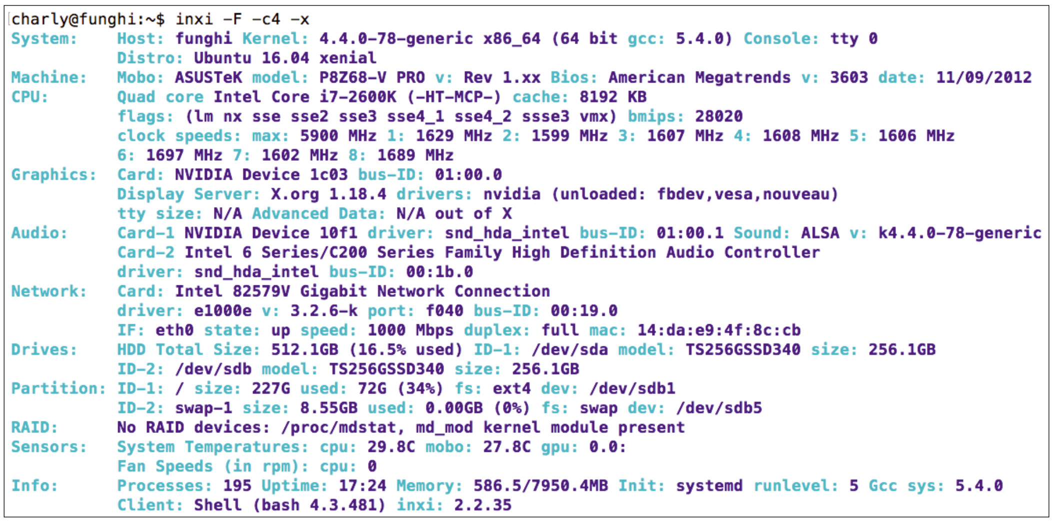
If I'm only interested in a particular component, I can target this with specific parameters, such as `-C`, `-A`, and `-G`, which stand for information on the CPU, audio, and graphics, respectively. Information on the RAM is returned after a (lowercase!) `-m`, which takes some getting used to.

Memory Details

Running with root privileges, inxi tells me more about the RAMe: Apparently four 2GB DDR modules are plugged into my test machine and clocked at 1600MHz – yes, this is a fairly ancient beast (Figure 2). The `-c4` parameter shown in Figures 1 and 2 is responsible for the color scheme. The default color scheme is not easily legible on terminals with a light background, but thanks to the plethora of options from `-c1` to `-c32`, selectable sets are available to suit your taste.

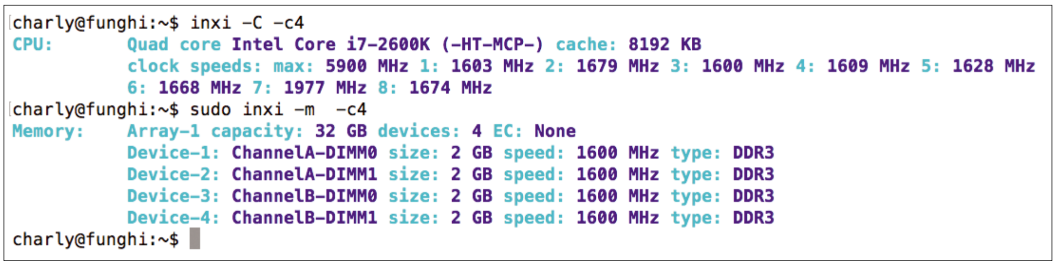
I can even talk inxi into a spot of simple process monitoring. If I want to know which five (this is the default value) processes are currently hogging the most RAM, `inxi -t m` will help me find out. If I want to see the top 10 processes, I enter `-t m10`. If I hear the CPU fan humming, on the other hand, I just need to replace the `m` with a `c` to view the processor load. You can also combine the two: `inxi -t cm 10` returns the top 10 RAM and CPU hogs.

At the end of this informative newscast on the computer, I'll take a quick look at the weather: `inxi -w Berlin, Germany` tells me what the situation looks like outside the server room. ■■■



```
charly@funghi:~$ inxi -F -c4 -x
System:    Host: funghi Kernel: 4.4.0-78-generic x86_64 (64 bit gcc: 5.4.0) Console: tty 0
           Distro: Ubuntu 16.04 xenial
Machine:   Mobo: ASUSTeK model: P8Z68-V PRO v: Rev 1.xx Bios: American Megatrends v: 3603 date: 11/09/2012
CPU:       Quad core Intel Core i7-2600K (-HT-MCP-) cache: 8192 KB
           flags: (lm nx sse sse2 sse3 sse4_1 sse4_2 ssse3 vmx) bmips: 28020
           clock speeds: max: 5900 MHz 1: 1629 MHz 2: 1599 MHz 3: 1607 MHz 4: 1608 MHz 5: 1606 MHz
           6: 1697 MHz 7: 1602 MHz 8: 1689 MHz
Graphics:  Card: NVIDIA Device 1c03 bus-ID: 01:00.0
           Display Server: X.org 1.18.4 drivers: nvidia (unloaded: fbdev,vesa,nouveau)
           tty size: N/A Advanced Data: N/A out of X
Audio:     Card-1 NVIDIA Device 10f1 driver: snd_hda_intel bus-ID: 01:00.1 Sound: ALSA v: k4.4.0-78-generic
           Card-2 Intel 6 Series/C200 Series Family High Definition Audio Controller
           driver: snd_hda_intel bus-ID: 00:1b.0
Network:   Card: Intel 82579V Gigabit Network Connection
           driver: e1000e v: 3.2.6-k port: f040 bus-ID: 00:19.0
           IF: eth0 state: up speed: 1000 Mbps duplex: full mac: 14:da:e9:4f:8c:cb
Drives:    HDD Total Size: 512.1GB (16.5% used) ID-1: /dev/sda model: TS256GSSD340 size: 256.1GB
           ID-2: /dev/sdb model: TS256GSSD340 size: 256.1GB
Partition: ID-1: / size: 227G used: 72G (34%) fs: ext4 dev: /dev/sdb1
           ID-2: swap-1 size: 8.55GB used: 0.00GB (0%) fs: swap dev: /dev/sdb5
RAID:      No RAID devices: /proc/mdstat, md_mod kernel module present
Sensors:   System Temperatures: cpu: 29.8C mobo: 27.8C gpu: 0.0:
           Fan Speeds (in rpm): cpu: 0
Info:      Processes: 195 Uptime: 17:24 Memory: 586.5/7950.4MB Init: systemd runlevel: 5 Gcc sys: 5.4.0
           Client: Shell (bash 4.3.481) inxi: 2.2.35
```

Figure 1: "Extensive" is probably the best description of what inxi bundles into its system overview here.



```
charly@funghi:~$ inxi -C -c4
CPU:       Quad core Intel Core i7-2600K (-HT-MCP-) cache: 8192 KB
           clock speeds: max: 5900 MHz 1: 1603 MHz 2: 1679 MHz 3: 1600 MHz 4: 1609 MHz 5: 1628 MHz
           6: 1668 MHz 7: 1977 MHz 8: 1674 MHz
charly@funghi:~$ sudo inxi -m  -c4
Memory:    Array-1 capacity: 32 GB devices: 4 EC: None
           Device-1: ChannelA-DIMM0 size: 2 GB speed: 1600 MHz type: DDR3
           Device-2: ChannelA-DIMM1 size: 2 GB speed: 1600 MHz type: DDR3
           Device-3: ChannelB-DIMM0 size: 2 GB speed: 1600 MHz type: DDR3
           Device-4: ChannelB-DIMM1 size: 2 GB speed: 1600 MHz type: DDR3
charly@funghi:~$ ▌
```

Figure 2: If you give inxi root privileges, you are rewarded with precise information on the computer's memory banks.

AUTHOR

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

INFO

[1] inxi: *https://github.com/smxi/inxi*

Playing and editing audio-visual from the command line

Melt

Whether you are an expert or a beginner, you can learn to edit audio and video clips at the command line with Melt.

By Bruce Byfield

When free software users think of Melt, they usually think first of GCC MELT [1], the popular extension system to the GCC compiler. However, Melt, the command-line multimedia player [2], is just as interesting in its own right, because it supports every file format you can imagine, and probably a few that you haven't. Admittedly, Melt's non-standard syntax takes a bit of learning, but as a command, it can be as simple or as complex as you choose to make it.

Melt is part of the Media Lovin' Toolkit (MLT) [3], a cross-platform multimedia framework designed for television. Two characteristics make Melt stand out: First, it has few dependencies, and, second, it works with existing multimedia libraries and applications. These characteristics are possible because of Melt's modular design and its high-level bindings for major programming languages like C++, Java, Lua, Perl, PHP, Python, Ruby, and Tcl. Additionally, Melt is thoroughly modern, making use of multicore processors and GPU processors.

Functionally, Melt is a full-featured editor that can customize both audio and video clips in detail, either for one-time playback or for permanent changes. Strictly speaking, Melt was originally a test tool for the MLT framework. However, its versatility means that, in a few small circles, Melt has become that most free software of applications: A command-line tool for purposes that are usually expected only in a desktop environment. These days, Melt is often found by itself in the repositories of major distributions.

Basic Playback

Like most commands at the prompt, Melt begins simply enough. To play a video, audio, or audiovisual clip, enter:

```
melt CLIP
```

A window opens for the video. It will be blank for an audio-only file and, of course, have no sound for a video-only file. When all clips have played, regardless of their format, you will have to close the window manually (Figure 1).

Multiple files can be entered with a space between them. Options for each file follow immediately after its name. Alternatively, you can use `-group` immediately after the basic command to apply options to all the listed files. For example,

```
melt -group in=0 out=25 CLIP1 CLIP2
```

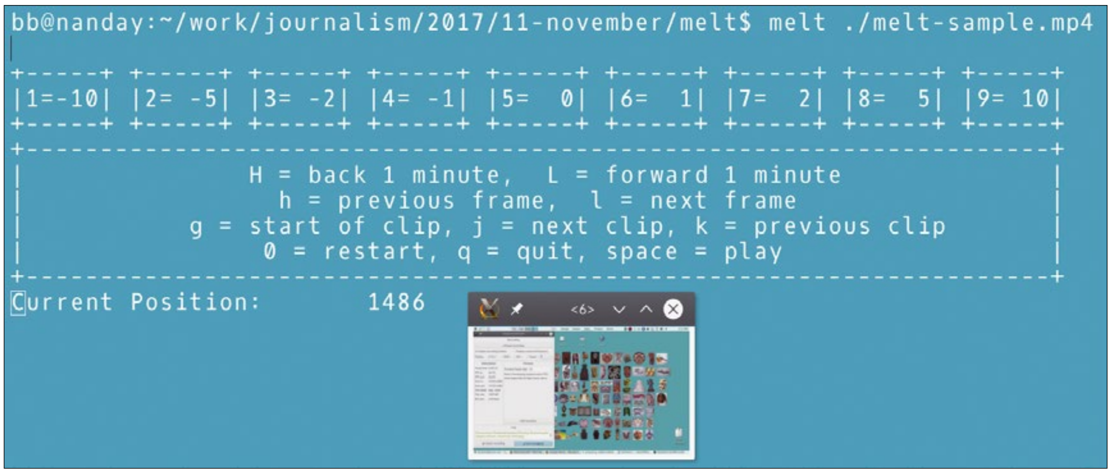
will play the first 25 frames of `CLIP1` and then the first 25 frames of `CLIP2`. Deciding exactly which frames to play, of course, requires some experimentation.

The group settings apply until the next `-group` option is entered in the command, so if you want to play the first 25 frames of `CLIP1` followed by the whole of `CLIP2`, the command would be:

```
melt -group in=0 out=25 CLIP1 ➘
     -group CLIP2
```

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. You can read more of his work at *http://brucebyfield.wordpress.com*



```
bb@nanday:~/work/journalism/2017/11-november/melt$ melt ./melt-sample.mp4

+------+ +------+ +------+ +------+ +------+ +------+ +------+ +------+ +------+
|1=-10| |2= -5| |3= -2| |4= -1| |5=  0| |6=  1| |7=  2| |8=  5| |9= 10|
+------+ +------+ +------+ +------+ +------+ +------+ +------+ +------+ +------+
+---------------------------------------------------------------------+
|              H = back 1 minute,  L = forward 1 minute                |
|                h = previous frame,  l = next frame                    |
|          g = start of clip, j = next clip, k = previous clip           |
|                0 = restart, q = quit, space = play                   |
+---------------------------------------------------------------------+
Current Position:       1486
```

Figure 1: Melt opens a window in which to run a video.

In effect, this command creates a new, empty group for the second clip. However, you could add a different range of frames to play for the second clip:

```
melt -group in=0 out=25 CLIP1 ⮐
     -group in=35 out=60 CLIP2
```

When playing an audiovisual clip, you can play the audio only by using either `-audio-track` or `-hide-video`. Conversely, `-video-track` or `-hide-audio` plays the video portion of a clip without the accompanying sound.

Besides `in=` and `out=`, you can modify playback with several other options. If you are experimenting with exactly what frames to use with `in=` and `out=`, adding `-progress` will give you a graphical representation of the file that is currently playing. You can also loop playback with `-repeat TIMES`.

Services and Filters

For simple playback, the options listed above are usually enough. However, the `melt` command also includes a wealth of advanced options for what the MLT documentation refers to as "services."

If you want to add a service to all the clips specified in the command, enter

```
-group -filter SERVICE
```

at the end of the command. For instance, to play all the clips in the same directory without color, enter:

```
melt -group in=0 out=25 CLIP1 ⮐
     -group in=35 out=60 CLIP2 ⮐
     -group -filter greyscale
```

Do not forget to include `-group`, or else the service may be affected by options for the last-named clip.

However, if you want the service to be used only by a specific clip, enter the option

```
-attach SERVICE:ARGUMENT NAME=VALUE
```

after the clip, just as you would any other option.

Melt supports hundreds of services, so you might often want to run the `-query` option first to see what services are available. To reduce the length of the output, you can run

```
melt -query SERVICE-TYPE
```

to see a group of related services (Figure 2). Even then, you will probably want to filter the command with `|less` to make it readable.

Note, too, that many services are listed by `-query` with a prefix, although all filters are entered in a `melt` command without the prefix. Additionally, the bare list may not tell you very much, so after you locate a possibly useful service, run

```
-query SERVICE = ID
```

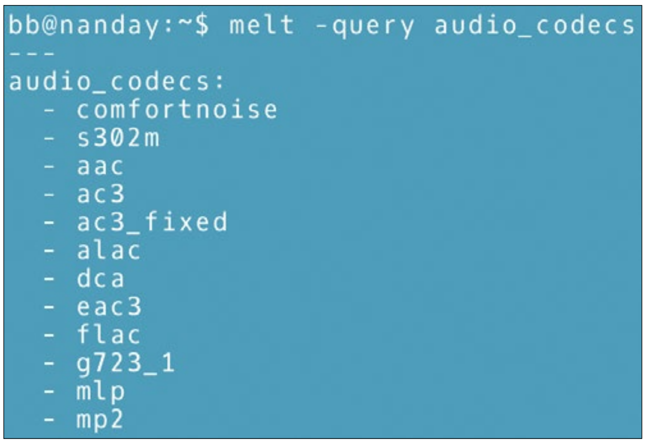
to receive more detailed information about the arguments and values that a particular service supports. If more detailed information is not available for a service, Melt simply displays the list of available services, organized by type.

The following service types can modify the `-query` option:

- `"consumers"`: The application or utility that plays clips, such as XML or JACK. If no consumer is specified, the default is SDL.
- `"filters"`: Frame modifiers that change how audio or visual are displayed. No filter affects the files – only how they are displayed. Examples include saturation, volume, and watermark.
- `"producers"`: Software playback libraries or components, or else wrappers for hardware drivers.
- `"transitions"`: How playback moves from one clip to another – for instance, `luma`, a change in brightness achieved by specifying a grayscale bitmap, or `matte`, a brief overlay of the two clips. When adding a transition to a command, use `-mix LENGTH` to set its duration, followed by `-mixer TRANSITION`.
- `"profiles"`: How a clip is processed, such as the frame resolution and scan rate. If not specified, the characteristics of the clips themselves are used. Beginners can generally ignore these services.
- `"presets"`: Playback options and formats. Again, beginners might want to ignore these services.
- `"formats"`: Audio, video, and audiovisual formats supported by Melt. Formats can be specified to force Melt to use a clip when it is having trouble identifying its format. Many listed formats are likely to be known only by experts.
- `"audio_codecs"`: Audio formats supported by Melt. Use this setting with `-query` rather than `formats` to reduce the size of the list displayed.
- `"video_codecs"`: Video formats supported by Melt. Use this setting with `-query` rather than `formats` to reduce the size of the list displayed.

Unless otherwise specified above, you can run any of these types of service, by entering the name of the type followed by the service (e.g., `-process NAME`).

Although Melt can be run with only a basic knowledge of its command structure, these lists of services reveal exactly how complex the command can be. Very likely, the average user does not use half of the available services, but if you are an expert in audio-video matters, Melt can probably accommodate your needs as well as those of a beginner.



```
bb@nanday:~$ melt -query audio_codecs
---
audio_codecs:
  - comfortnoise
  - s302m
  - aac
  - ac3
  - ac3_fixed
  - alac
  - dca
  - eac3
  - flac
  - g723_1
  - mlp
  - mp2
```

Figure 2: The start of the list of audio formats supported by Melt, as shown by the query option.

Last Words

As you can imagine, the `melt` command can be complex by the time you are finished. Any given instance can quickly become so complex that recreating its structure may be more than you are eager to attempt. You can always use Bash history, but in the long run, you can save the command structure with:

```
-serialise FILENAME.melt
```

To play back the saved command structure, run it in Melt as though it is just another supported format. You can add another clip by including `-track CLIP` to the same command.

Advanced users may want to learn about MLT's XML format [4], which uses the same components as the `melt` command to save complex playback options. Note, though, that the online MLT documentation presupposes a knowledge of audiovisual matters and might be only intermittently useful, unless you are willing to look elsewhere constantly for definitions or to experiment until you understand.

Most users, though, should find a basic knowledge of Melt more than enough to begin using it. Like all audiovisual editors, including those that run on the desktop, Melt is very much what you make it. The MLT website admits to a few gaps in functionality, such as the ability to add a watermark to every frame. However, for the most part, Melt can meet the needs of not only users who want nothing more than simple playback, but also more advanced users who want to edit playback in more elaborate ways. ■■■

Info

[1] GCC MELT: *http://gcc-melt.org/*

[2] Melt: *https://www.mltframework.org/docs/melt/*

[3] Media Lovin' Toolkit: *https://www.mltframework.org*

[4] MLT XML: *https://www.mltframework.org/docs/mltxml/*

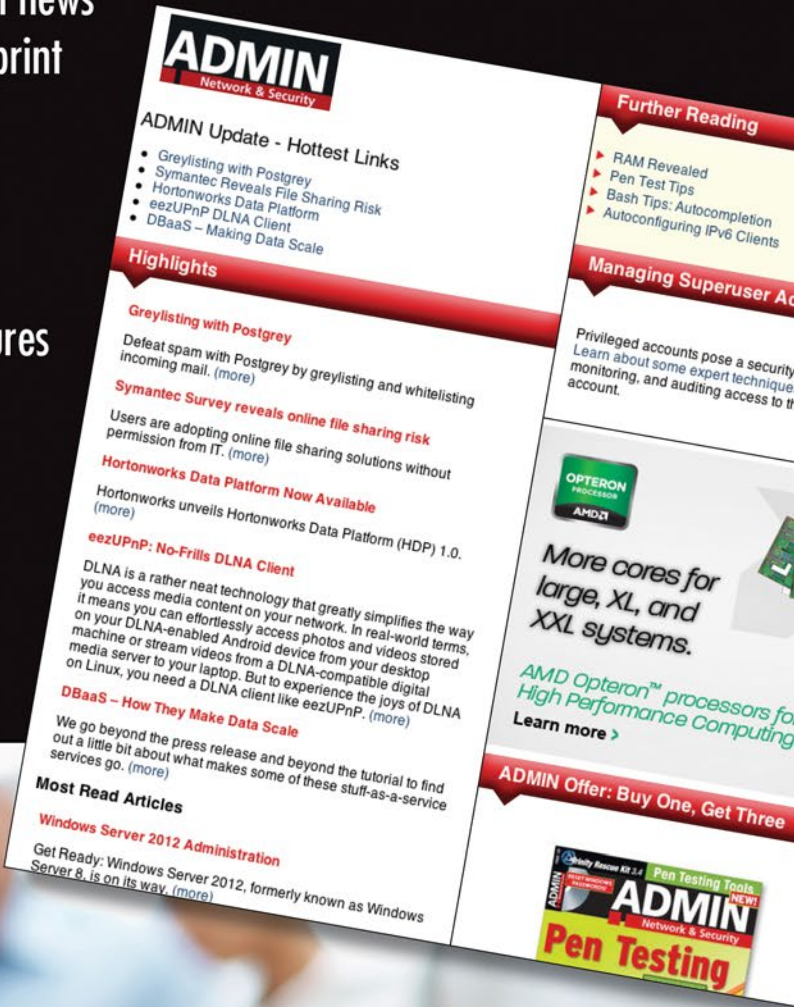
Too Swamped to Surf?

Our ADMIN Online website offers additional news and technical articles you won't see in our print magazines.

Subscribe today to our free ADMIN Update newsletter and receive:

- Helpful updates on our best online features
- Timely discounts and special bonuses available only to newsletter readers
- Deep knowledge of the new IT

ADMIN Network & Security


ADMIN Network & Security

ADMIN Update - Hottest Links

- Greylisting with Postgrey
- Symantec Reveals File Sharing Risk
- Hortonworks Data Platform
- eezUPnP DLNA Client
- DBaaS – Making Data Scale

Highlights

Greylisting with Postgrey

Defeat spam with Postgrey by greylisting and whitelisting incoming mail. (more)

Symantec Survey reveals online file sharing risk

Users are adopting online file sharing solutions without permission from IT. (more)

Hortonworks Data Platform Now Available

Hortonworks unveils Hortonworks Data Platform (HDP) 1.0. (more)

eezUPnP: No-Frills DLNA Client

DLNA is a rather neat technology that greatly simplifies the way you access media content on your network. In real-world terms, it means you can effortlessly access photos and videos stored on your DLNA-enabled Android device from your desktop machine or stream videos from a DLNA-compatible digital media server to your laptop. But to experience the joys of DLNA on Linux, you need a DLNA client like eezUPnP. (more)

DBaaS – How They Make Data Scale

We go beyond the press release and beyond the tutorial to find out a little bit about what makes some of these stuff-as-a-service services go. (more)

Most Read Articles

Windows Server 2012 Administration

Get Ready: Windows Server 2012, formerly known as Windows Server 8, is on its way. (more)

Further Reading

- RAM Revealed
- Pen Test Tips
- Bash Tips: Autocompletion
- Autoconfiguring IPv6 Clients

Managing Superuser Ac

Privileged accounts pose a security
Learn about some expert technique
monitoring, and auditing access to th
account.

OPTERON PROCESSOR AMD

More cores for large, XL, and XXL systems.

AMD Opteron™ processors fo
High Performance Computing

Learn more >

ADMIN Offer: Buy One, Get Three

Pen Testing Tools NEW! ADMIN Network & Security Pen Testing

© Yuri Arcurs, fotolia.com

www.admin-magazine.com/newsletter

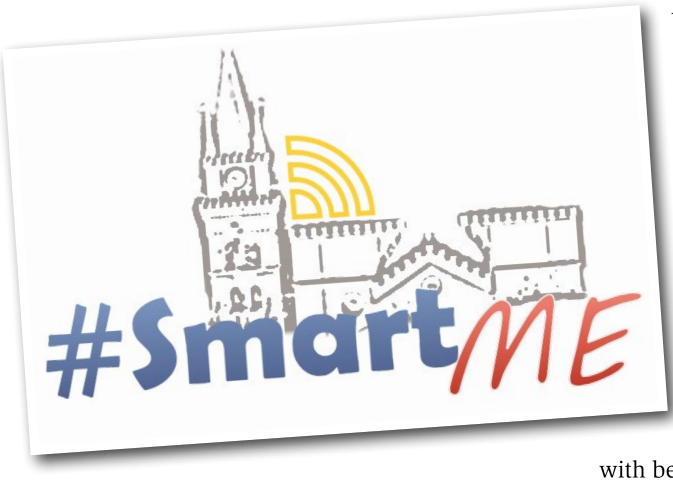
MakerSpace

Smarter with open source

Too Clever

Making the city of Messina, Italy, smarter with open source and IoT. *By Swapnil Bhartiya*

Imagine you are driving downtown in the middle of the night. Rows of red lights shine ahead. Traffic is thin, almost non-existent, and you wish you could turn all the red lights green and arrive home earlier. What if you could pull your phone out, automatically connect to the nearby lights, and turn them green?



That's precisely what a team at the University of Messina, Italy, tried to do. "It worked. It was very precise in real time, but there are so many rules, laws, and regulations to comply with before even being considered for actual usage in the streets," said Giovanni Merlino, research fellow in the Department of Engineering, University of Messina. Merlino delivered a talk about his team's work at the Boston OpenStack Summit 2017.

Messina is one of the largest cities on the island of Sicily and is home to more than 300,000 inhabitants. A city that size has its own set of civil challenges – environmental pollution, wear and tear on roads, safety and security of citizens, traffic monitoring and optimization, and potholes.

The modern way of solving these problems is by building an infrastructure that collects data through Internet of Things (IoT) devices spread across the city. That data is then analyzed to find the solutions to different problems. Take, for example, potholes. If sensors installed in vehicles or on mobile devices carried by people with the appropriate app can notify the municipality of the pothole locations, these potholes could be fixed. If the city can monitor the air, acoustic pressure, brightness, humidity, and other environmental aspects, officials could improve the quality of life.

SmartMe.IO

Traditionally, solutions to these problems start with municipalities; however, they don't usually have the resources or incentives to invest in IT to find solutions. SmartMe.IO, an academic spin-off born out of a team of researchers from the Mobile and Distributed Systems Lab (MDSLab) at the University of Messina, tries to change that by coming up with vendor-neutral, open source solutions that can help a city "on a shoestring budget."

SmartMe.IO kick-started a smart city-related crowdfunded project [1] aimed at encouraging "a conversation with the municipality of Messina in order to spur the creation of a novel virtual ecosystem based upon the paradigm of the Internet of Things."

This SmartMe.IO initiative started with some prototypes that used Arduino-based sensors to collect environmental data, such as humidity, brightness, and carbon dioxide level. However, they didn't want it to remain a research project. They wanted to expand and scale it out so that it could serve people beyond the city of Messina. The team ran a successful crowdfunding project to grow the project.

In a nutshell, the SmartMe.IO project has three core components: IoT or edge devices scattered across the city to collect different kinds of data, an Infrastructure as a Service (IaaS) platform to collect and process the data, and data processing to offer more refined services to the city and other stakeholders.

Stack4Things

The SmartMe.IO team created a fully open source framework called Stack4Things, which is a complete solution to manage the fleet of IoT devices remotely. The framework has two agents: one for the server side and one for edge devices. The server/cloud-side agent is called IoTronic [2], and the node-side (IoT edge device) agent is called Lightning-Rod.

IoTronic is an IoT resource management service for OpenStack Cloud, traditionally known for hyperscale cloud computing and massive infrastructure. SmartMe.IO wanted to build an IaaS-oriented solution for these IoT devices, an infrastructure that can be scaled on the basis of demand. At the same time, they wanted to expand the scope of OpenStack beyond the data center, toward the management of sensing and actuation resources. Initially, the software was written in Node.js, but the researchers have been porting it to Python to make it OpenStack compliant. The cloud currently runs in the University data center.

On the client side, Lightning-Rod runs on edge devices powered by single-board computers like Raspberry Pi, running a customized version of the OpenWrt/LEDE flavor of Linux developed in-house, an environment common to all the supported SmartMe.IO edge devices. These edge devices are called Arancino and are armed with a wide range of commodity sensors to collect different kinds of data. The University once hosted one of the Arduino labs. As the lab was shut down, the university absorbed the human expertise from the lab, bringing valuable resources to the SmartMe.IO team that enabled them to work on Arancino. More than 100+ IoT devices spread across the city are actively collecting valuable data.

Virtual Points

In addition to the physical boxes on the map are virtual boxes, which represent points of interest that could be either landmarks or most-clicked areas. These areas won't have any physical boxes installed.

"The virtual box would then be just flagged/labeled/colored differently to let the user know data is extrapolated/predicted, but not actually sampled. The predictive engine analyzes and correlates time series from neighboring boxes (say, four or more surrounding the location of the virtual one) and predicts expected values of homogeneous metrics (say, humidity values from four neighboring humidity sensors), or even heterogeneous ones (say, temperature values from a mix of neighboring and/or co-located humidity/brightness sensors). Use cases [include] faulty sensors (incomplete time series) [and] sparsely covered areas," said Merlino.

CKAN

The third, and the most important, part of the equation is data. The SmartMe.IO platform uses open source CKAN [3] data management software to manage the collected data, which is openly available to developers in raw format and can be downloaded for use.

The real value add-on of SmartMe.IO is extrapolating that data and offering meaningful metrics so the municipality can correlate data to understand, for example, the cause of pollution and mitigate it.

However, SmartMe.IO is not stopping at running the cloud and managing the IoT devices. They want to make these devices smarter, so they can offer more value. Now they have started to use cameras with these devices to offer smart camera services, such as help with head counts in crowded places. Some projects are already running in places like airports to assist in safety and security. However, cameras increase the risk of privacy invasion. To mitigate such concerns, SmartMe.IO is pushing machine learning to these devices so that no image is sent to or saved on the server. Data processing happens on the edge device and only statistical data is sent to the server. Once again, it's all powered by open source, with the use of Darknet on the cameras to protect privacy.

Future

The next plan for the project is to offer complete end-to-end solutions, all the way from Arancino boxes to the cloud, so interested parties don't have to worry about the infrastructure and can consume the data that they need.

The SmartMe.IO team is also working on using mobile devices like smartphones to expand their scope. To make the project financially sustainable, SmartMe.IO is planning to sell services, consulting, and integrated systems. What they are not selling is datasets; at the moment, data is meant to be open.

The SmartMe.IO platform is part of the Fiware [4] community, a European initiative to "build an open sustainable ecosystem around public, royalty-free and implementation-driven software platform standards that will ease the development of new Smart Applications in multiple sectors."

The SmartMe.IO efforts are a great example of how open source technologies enable anyone in the world to come up with innovative solutions to their problems. As the project matures, it holds so many possibilities, such as helping cities cope with pollution, infrastructure management, wear and tear of roads, water leakage, wildfire control, traffic control, security at public places, and more. ∎∎∎

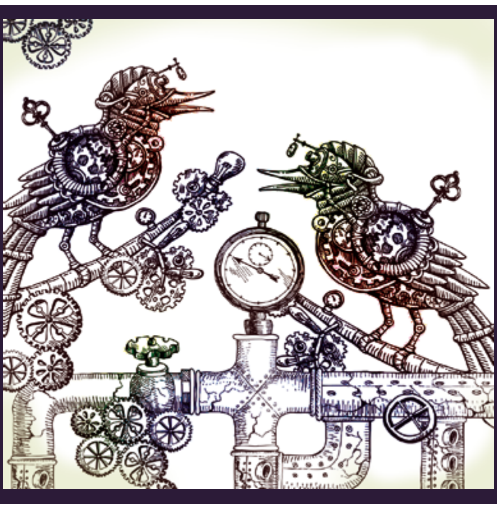
Info

[1] SmartME project: *smartme.unime.it*

[2] IoTronic: *https://github.com/openstack/iotronic*

[3] CKAN: *https://ckan.org/*

[4] Fiware: *https://www.fiware.org/devguides/hosting-your-application-on-a-fiware-cloud/*



MakerSpace

Build an FM radio using an RTL-SDR dongle

Pi FM Radio

Low-cost RTL-SDR dongles can read frequencies between 24 and 1,766MHz. We built a simple FM radio with a Raspberry Pi, a USB dongle based on the RTL2832U chipset, an LCD HAT, and some Python code. *By Pete Metcalfe*

FM radio projects that use chipsets with a low-level Inter-Integrated Circuit (I2C) interface, like the RDA5807M and TEA5767, can work, but I found that both solutions have some drawbacks. The RDA5807M chip is poorly documented, with only Arduino C libraries, and the TEA5767 chip has no volume control.

Software-defined radio (SDR) offers a higher level interface that allows access to mixers, filters, amplifiers, modulators/demodulators, and detectors on the hardware. A wide range of hardware supports SDR, and the RTL-SDR [1] USB dongles based on the RTL2832U chipsets are well-priced at $10-$15.

SDRs have a large list of supported applications; some of the cooler projects include tracking airplanes, free-to-air TV, and monitoring satellite data.

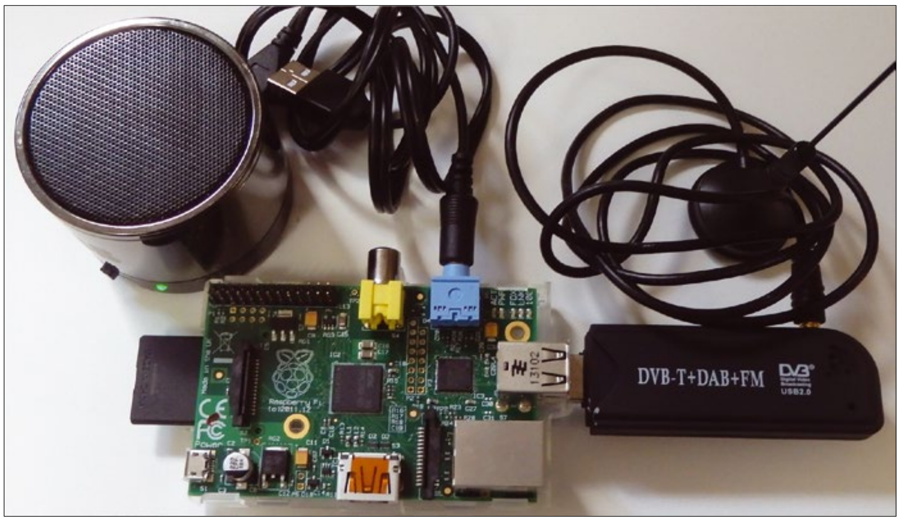


Figure 1: FM radio hardware setup.

Getting Started

To install the basic software, enter:

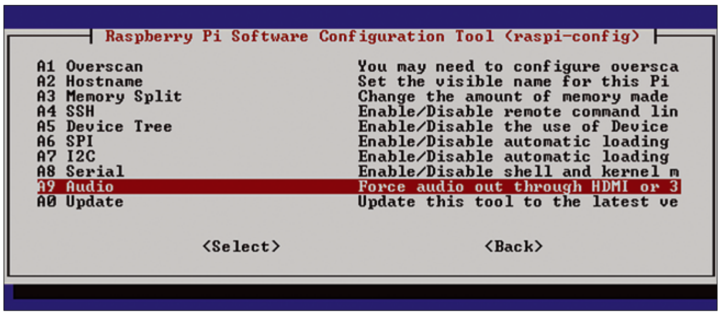
```
sudo apt-get install rtl-sdr
```

For my tests, I used a basic Raspberry Pi 1 Model B, but I also tested it on the Rasp Pi versions 2 and 3 and on an old, low-end PC running Lubuntu.

An important difference between the RTL-SDR dongle and an FM tuner module is that the FM tuner module needs the speakers to be directly connected to the tuner module. When you are using the RTL-SDR dongle, the audio is generated on the Raspberry Pi (or PC), so you will use the sound output of your computer. For my Raspberry PI setup, I used powered speakers (Figure 1), and for my laptop testing I used the internal speakers in the laptop.

The RTL-SDR dongle includes an externally connected antenna; if possible, you should try to place this antenna close to a window.

The `rtl_fm` command-line utility is an FM demodulator that is used to read and sample a selected frequency. If you are looking to do more serious applications, see the GNU Radio Project [2].



```
Raspberry Pi Software Configuration Tool (raspi-config)

A1 Overscan              You may need to configure oversca
A2 Hostname              Set the visible name for this Pi
A3 Memory Split          Change the amount of memory made
A4 SSH                   Enable/Disable remote command lin
A5 Device Tree           Enable/Disable the use of Device
A6 SPI                   Enable/Disable automatic loading
A7 I2C                   Enable/Disable automatic loading
A8 Serial                Enable/Disable shell and kernel m
A9 Audio                 Force audio out through HDMI or 3
A0 Update                Update this tool to the latest ve

              <Select>                  <Back>
```

Figure 2: `raspi-config` ***Advanced* | *Audio* option.**

A number of options can be passed to `rtl_fm`; the key ones are the frequency (`-f`), the sample rate (`-s`), and the output rate (`-r`). The output of `rtl_fm` needs to be directed to an audio player program. I used `aplay`, which is a command-line Advanced Linux Sound Architecture (ALSA) player, but other players could be used. I matched the sample rate (`-r`) for `aplay` with the `rtl_fm` output sampling rate and used the 16-bit little-endian (`S16_LE`) `aplay` sample format (`-f`).

The syntax with the options required to play an FM radio station at 107.9MHz is as follows:

```
rtl_fm -f 107.9e6 -s 200000 -r 48000 | ↩
  aplay -r 48000 -f S16_LE
```

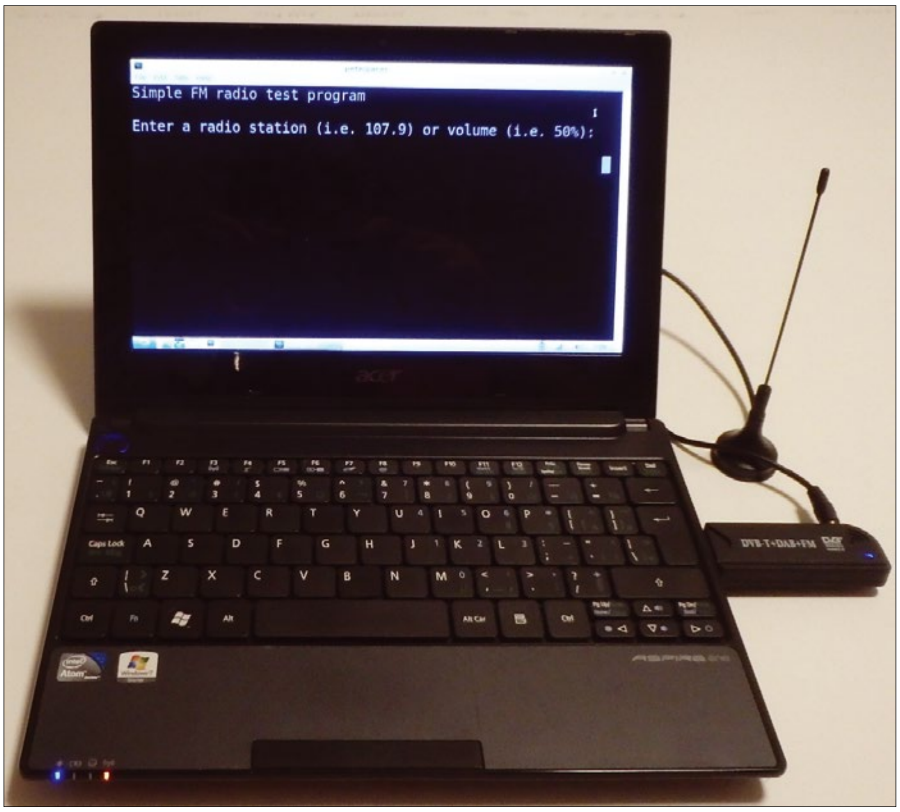
The `rtl_fm` application needs to be stopped when you want to play a new radio station. If `rtl_fm` is running in the background, the `ps` command can be used to find process IDs. The `kill` command can then be used to terminate the task. An example of finding and terminating the `rtl_fm` task is:

```
pi@raspberrypi:~ $ ps -e | grep rtl_fm
 1709 pts/0    00:00:33 rtl_fm
pi@raspberrypi:~ $ kill 1709
pi@raspberrypi:~ $ ps -e | grep rtl_fm
pi@raspberrypi:~ $
```

Adjusting the Volume

For Raspberry Pi applications, you can force the audio connections to use either the HDMI port or the phone jack on the Pi in `raspi-config` by selecting the *Advanced* menu option and then *Audio* (Figure 2).

You have a few ways to adjust the audio volume. One method is to use the `amixer` utility. To change the audio out-



```
Simple FM radio test program

Enter a radio station (i.e. 107.9) or volume (i.e. 50%):
```

Figure 3: **RTL-SDR dongle on a Lubuntu laptop.**

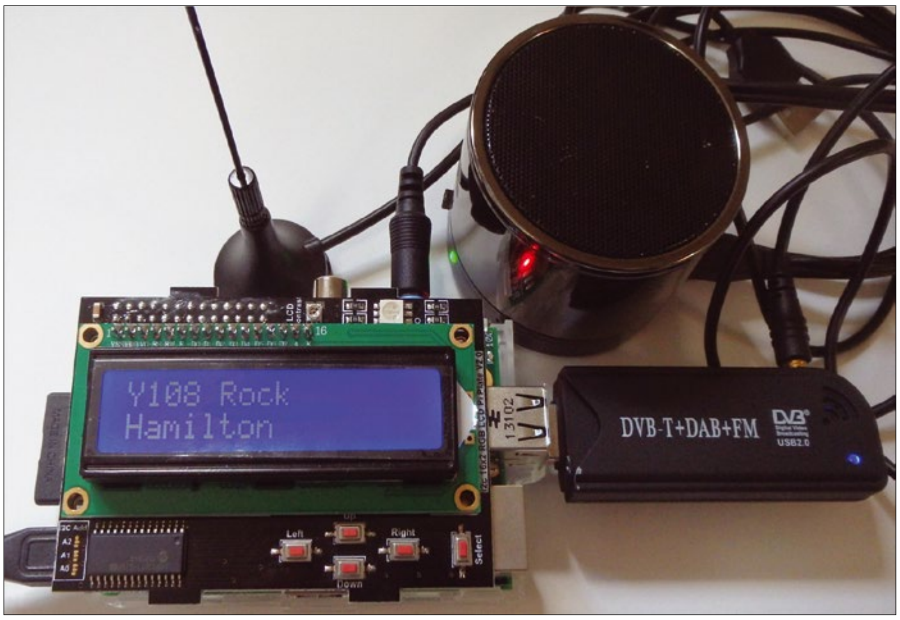


Figure 4: **LCD FM radio control.**

Listing 1: Command-Line FM Radio Utility

```
# FM_radio.py
# simple FM radio test utility using a RTL SDR dongle

import subprocess, signal, os, time

def newstation(station):
    global process, stnum
    # create a rtl_fm command line string and insert the new freq
    part1 = "rtl_fm -f "
    part2 = "e6  -s 200000 -r 48000 - | aplay -r 48000 -f S16_LE"
    cmd = part1 + station + part2

    print ('Playing station :', station)

    # kill the old fm connection if it was running
    if process != 0:
        process = int(subprocess.check_output(["pidof","rtl_fm"] ))
        print ("Process pid = ", process)
        if process != 0:
            os.kill(process,signal.SIGINT)
            time.sleep(2) # wait 2 seconds to restart rtl_fm

    # start the new fm connection
    print (cmd)
    process = subprocess.Popen(cmd, shell=True)


def setvolume(thevolume):
    # pass the new volume setting to the amixer command
    os.system('amixer sset "PCM" ' + thevolume)
    print ('volume = ' , thevolume)

# Simple FM radio test program
process = 0

print ("Simple FM radio test program\n")
while True:
    answer = raw_input(
      "Enter a radio station (i.e.
      107.9) or volume (i.e. 50%): ")

    if answer.find('%') > 0:
        setvolume(answer)
    else:
        newstation(answer)
```

put volume on the Pi (or laptop) speakers, the Pulse Code Modulation (PCM) device is addressed. An example using `amixer` to set the volume to 70 percent would be:

```
amixer sset "PCM" 70%
```

Python Test Program

For my basic testing, I created a simple Python command-line application that I could run on both my Raspberry Pi and on my old laptop running Lubuntu (Figure 3).

A few Python libraries were used, including the `subprocess` library to launch `rtl_fm` and return a process ID, the `os` library to kill a process, and the `time` library to add a delay (sleep), so the `rtl_fm` task was given enough time to shut down cleanly before restarting.

A `newstation()` function was created to stop a running `rtl_fm` FM station and then restart it with a new FM station frequency, and a `setvolume()` function was created to pass new volume settings to `amixer`.

To run the test program (Listing 1) [3], I entered the command,

```
$ python FM_radio.py
Simple FM radio test program
```

which then accepts a volume level as a percentage (e.g., 50%) or a radio station frequency (e.g., 107.9).

Raspberry Pi FM Radio

For the Pi FM radio project in this article, I tried a few different arrangements, but I found that an LCD HAT (hardware attached on top) with buttons (Figure 4) worked best. However, other options such as a PiFace digital HAT (Figure 5) or a button HAT could also be used.

The LCD Python libraries will vary according to the hardware used; however, most LCD HATs are based on the Adafruit libraries [4].

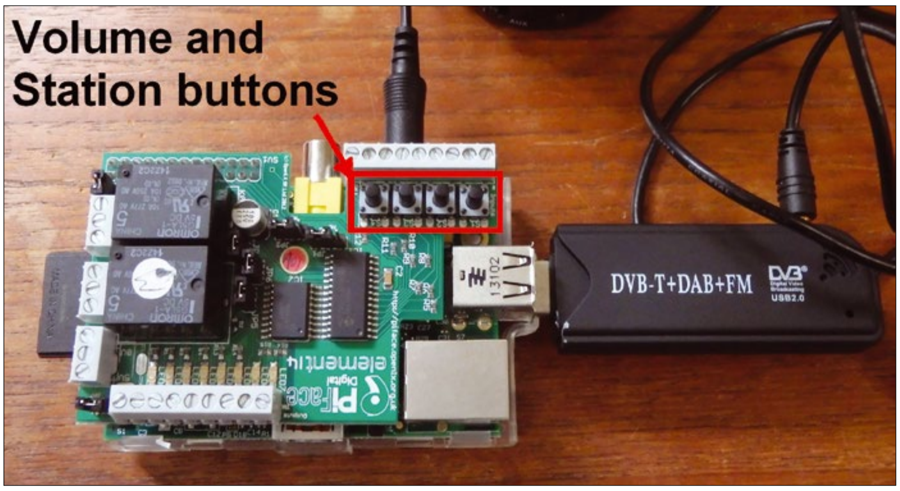


Figure 5: PiFace FM radio control.

Listing 2: Pi FM Radio with an LCD HAT

```
#!/usr/bin/python
# PI_FM_radion.py - FM Radio with an LCD Shield for controlling the stations and volume

import os, subprocess, signal
import time
import Adafruit_CharLCD as LCD

def newstation(direction):
    global stnum, stations, process

    print 'stnum=',stnum,'direction=',direction

    part1 = "rtl_fm -f "
    part2 = " -s 200000 -r 48000 | aplay -r 48000 -f S16_LE"

    if (stnum + direction < (len(stations)  )) and (stnum + direction > -1):
        stnum = stnum + direction
        print('Playing station :', stations[stnum])
        cmd = part1 + stations[stnum] + part2
        if process != 0:
            process = int(subprocess.check_output(["pidof","rtl_fm"] ))
            print "Process pid = ", process
            os.kill(process,signal.SIGINT)
        # start the new fm connection
        print cmd
        process = subprocess.Popen(cmd, shell=True)


def setvolume(voldif):
    global thevolume
    if (thevolume + voldif > 0) and (thevolume + voldif <100):
        thevolume = thevolume + voldif
        os.system('amixer sset "PCM" ' + str(thevolume) + '%')
        print 'volume = ' , thevolume


lcd = LCD.Adafruit_CharLCDPlate()

# Add your own stations and station info
stations = ['95.3e6','94.7e6','102.9e6','107.9e6']
sinfo = ['95.3 country', '94.7 light','102.9 easy','Y108 Rock\nHamilton ']
thevolume = 40  #initial volume

stnum = 1         #pick a starting station
process = 0
newstation(0)
lcd.message(sinfo[stnum])
setvolume(thevolume)


print 'Press Ctrl-C to quit.'
```

The Python code for the LCD HAT (Listing 2) is based on the simple test code with some additional logic for an LCD button interface and some predefined radio frequencies. The main code loops through, looking at each of the buttons with the `lcd.is_pressed()` function. The `LCD.UP` and `LCD.DOWN` buttons are used for volume control, and the `LCD.LEFT` and `LCD.RIGHT` buttons cycle through the predefined radio stations. The `lcd.clear()` and `lcd.message()` functions show the new radio station information on the LCD HAT. In my testing, I used a 0.25-second delay on all the button presses, although you might need to tune this value for your hardware and usage.

Summary

Creating a homemade FM radio is just one of the many interesting applications for SDR utilities and the low-cost RTL-SDR dongle [5]. In the future, I would like to add a Python Tkinter or web interface to my FM radio application. ■■■

Listing 2: Pi FM Radio with an LCD HAT (continued)

```
while True:  #look for button presses
    if lcd.is_pressed(LCD.UP):
        setvolume(5)
        time.sleep(0.25)
    if lcd.is_pressed(LCD.DOWN):
        setvolume(-5)
        time.sleep(0.25)
    if lcd.is_pressed(LCD.LEFT):
        newstation(-1)
        lcd.clear()
        lcd.message(sinfo[stnum])
        time.sleep(0.25)
    if lcd.is_pressed(LCD.RIGHT):
        newstation(1)
        lcd.clear()
        lcd.message(sinfo[stnum])
        time.sleep(0.25)
```

Info

[1] About RTL-SDR: *http://www.rtl-sdr.com/about-rtl-sdr/*

[2] GNU Radio: *https://www.gnuradio.org/*

[3] Code for this article: *ftp://ftp.linux-magazine.com/pub/listings/linux-magazine.com/206/*

[4] Adafruit libraries: *https://github.com/adafruit/Adafruit_Python_CharLCD*

[5] See more of Pete's Projects: *https://funprojects.blog*

MakerSpace

The Volumio 2.0 web-based audio player

Playback

Volumio and a Raspberry Pi can add smart functions to any stereo system. Whether playing diverse audio formats or streaming Spotify, the combination of smartphone control, a Raspberry Pi Display, and Volumio outperforms many commercial solutions.

By Christoph Langner

Old-fashioned AM and FM radio has been out for years: Nowadays, a radio must be able to receive digital audio and be web-enabled to stream music from Spotify and other providers. However, the devices that were conceived by former audio giants such as Sony and Panasonic are often fraught with quirky controls. Virtually no Internet radios come with large displays and precise touchscreens. The fun only begins when you can control the radio with a mobile phone or tablet app. In this category, network loudspeakers (e.g., by Sonos or Raumfeld) are achieving massive sales.

Legacy hi-fi systems – without a display or network connection, but with excellent sound that cost an arm and a leg just a few years ago – are still in existence in many living rooms. If you do not want to exchange your precious amplifier for the latest technology, you will find a convenient and elegant upgrade solution with the Volumio [1] distribution. The Linux-based digital audio player software is easy to set up and converts any radio or stereo system with a line-in socket into a "smart radio," with network access and a Spotify connection. In combination with a Raspberry Pi and the original Raspberry Pi Display, the solution is bound to impress with its ease of use and elegance.

Volumio 2.0

Since the end of 2016, Volumio 2 has been available as a significantly updated version that introduces various innovations and eliminates numerous errors [2]. In contrast to the first editions of the software, Volumio 2 has acquired a plugin interface, a hotspot feature, and a revised interface.

Volumio 2 for the Raspberry Pi is based on the current Raspbian Jessie. Volumio supports other single-board computers in addition to the Rasp Pi, such as the Odroid-C1/C2 and the CuBox-i. An image for classical Intel PCs has Debian underpinnings. To use the software, you store the Volumio image on an SD card, as you would a Raspbian image, and boot the Rasp Pi from it. The card should have at least 4GB of free space; if it also needs to store part or practically all of your music collection, you will have to dimension it accordingly.

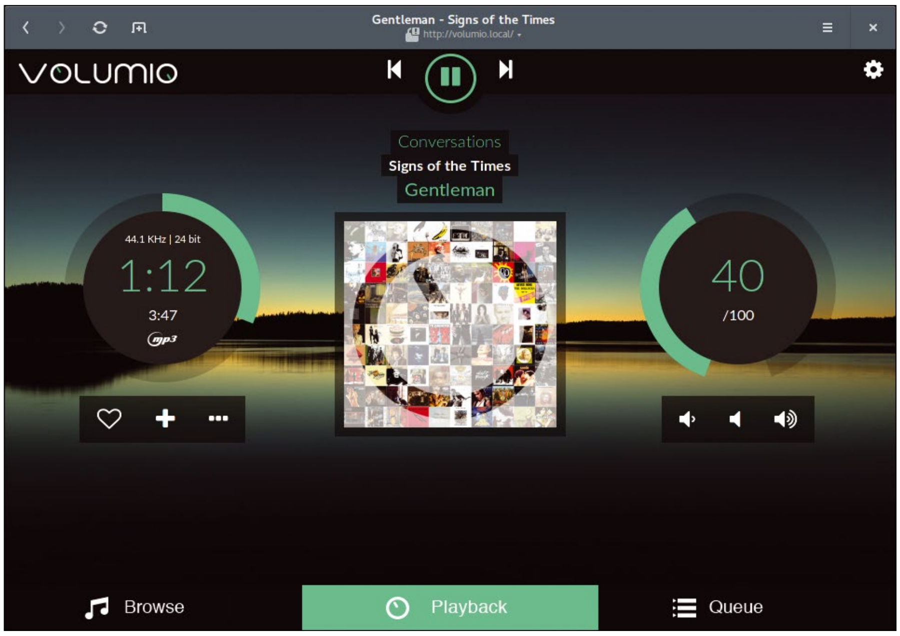


Figure 1: The Volumio web front end in a web browser.

After installation and the first launch, the web interface of the Volumio Rasp Pi interface can be accessed using any Apple computer and most Linux systems from the URL *http://volumio.local* or by typing in the IP address of the Raspberry Pi directly.

Because Volumio does not explicitly display the IP address on the screen during the boot procedure, you have to read it from the configuration interface of your WiFi router or use a network scanner such as Fing for Android [3]. Linux users can run `arp-scan`,

```
$ sudo arp-scan --localnet | ↵
  grep Raspberry
192.168.111.195 b8:27:eb:66:ab:44 ↵
  Raspberry Pi Foundation
```

which you can install from your distribution's package manager, if necessary.

If you have connected the Volumio Rasp Pi to a monitor, you will only see a login prompt immediately after installation. In the basic configuration, Volumio is aimed at users who want to control the audio player through the network. However, a graphical interface can be installed easily when setting up the software.

Command Control

Because of the full-fledged Linux underpinnings, Volumio natively provides an automated SSH server with authentication via public keys, allowing you to log in to the system from a network with

```
ssh volumio@volumio.local
```

or, alternatively, with the corresponding IP address in place of `volumio.local`. Both the password and login are *volumio*. The root account is disabled as in other Linux distributions, but you can gain administrative rights by prefacing a command with `sudo`.

Volumio offers a command-line interface: Using commands on the Volumio machine such as `volumio pause` or `volumio volume 50`, you can address the player without a web or Music Player Daemon client. Typing `volumio --help` displays an overview of all commands.

GUI

The most important functions of the web interface may well be self-explanatory (Figure 1): At top center is the button for Play/Pause; the buttons on either side let you jump to the next or previous track in the playlist. Under the buttons are the names of the current track and album. The gauge on the left is a timeline of the current track that you can use to jump to any point; the gauge to the right is for volume control.

To change the language in Volumio, open the sidebar by clicking the gear icon in the right corner. Choose *Appearance* | *Language* and select your language of choice in the Select Language drop-down. The web browser automatically rebuilds the page in the selected language after you click *Save*. You can also select or drop different background images and color schemes for the web interface in this dialog.

The easiest way to add music is on a USB storage device loaded with MP3 files. In addition to MP3, Volumio supports the AAC, ALAC, FLAC, WAV, and PLS formats. The software automatically mounts the medium; you can then access your music by clicking *Browse* in the main window and then using the *Music Library* | *USB* menu. Alternatively, you can use Samba to upload data directly to Volumio Rasp Pi memory. The URL is *smb://volumio*.

The Rasp Pi interface has poor network throughput because the network module has to share an internal USB interface with the USB ports, so transferring a large music collection can take a while to complete. If you are an impatient user, you might prefer disconnecting the Rasp Pi from the mains, removing the memory card from the device, and reading it in a card reader on your PC. Volumio searches for new tracks in the `volumio_data` drive in the `/dyn/data/INTERNAL/` folder. The partition uses the ext4 Linux filesystem.

Strengthened

Volumio plays sound through the headphone socket of the Raspberry Pi by default. The socket can be connected to the stereo system line-in with an RCA cable. Alternatively, you can send audio to the HDMI output from the Playback menu. Volumio also supports many digital-to-analog converter (DAC) HATs, or Raspberry Pi GPIO extensions. They provide far better audio quality and more audio outputs (e.g., Toslink) than the Raspberry Pi with standard tools. Audiophile fans might even want to upgrade the Rasp Pi with a tube amplifier [4].

If you are working with a Rasp Pi 3 or have connected a compatible USB WiFi dongle to an older Rasp Pi, Volumio automatically sets up a hotspot. The SSID is *Volumio* and the access password is *volumio2* (Figure 2). You can give friends and acquaintances at a party access to your music collection from their cellphones, without having to reveal your home network (Figure 3). The

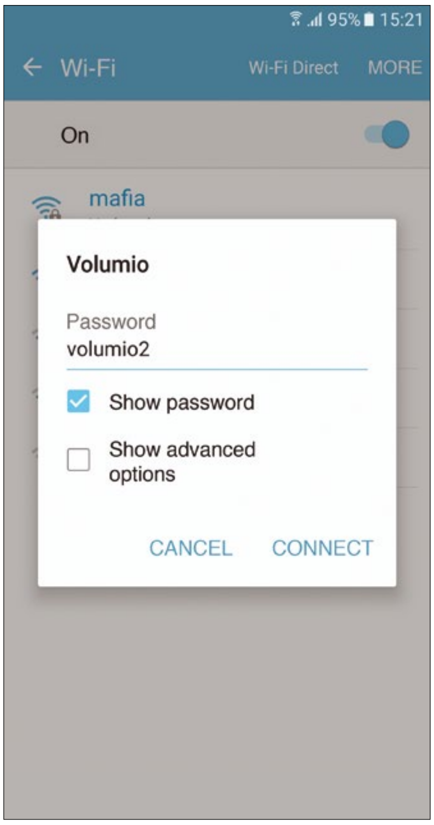


Figure 2: Guests can log in to the Volumio machine from their cellphones and be DJs at your party.

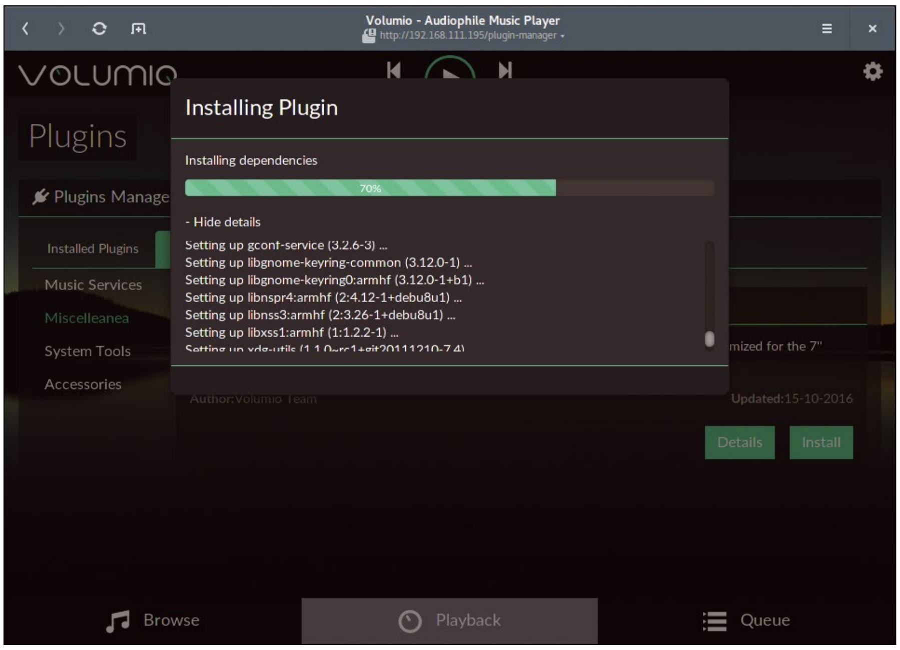
hotspot settings can be changed under *Network* | *Hotspot Settings*, if necessary. You can also completely disable WiFi access.

Display

The creators of Volumio primarily have users in mind who want to connect the Rasp Pi to their stereo and hide it behind a hi-fi system. The system can then be controlled from a smartphone or from the web browser on a PC. In combination with the official Raspberry Pi Display and a suitable housing, the stereo system can be made "smart" very easily and controlled conveniently on the 7-inch touchscreen.

In the basic setting, a monitor connected to the Volumio Rasp Pi does not display anything apart from the launch messages at boot time and the login prompt. The system does not even provide a graphical desktop environment. However, additional functions (e.g., screen output) can be installed easily through the *Plugins* entry in the sidebar.

The currently available plugins can be displayed in various categories in



Installing Plugin

Installing dependencies

70%

- Hide details

```
Setting up gconf-service (3.2.6-3) ...
Setting up libgnome-keyring-common (3.12.0-1) ...
Setting up libgnome-keyring0:armhf (3.12.0-1+b1) ...
Setting up libnspr4:armhf (2:4.12-1+debu8u1) ...
Setting up libnss3:armhf (2:3.26-1+debu8u1) ...
Setting up libxss1:armhf (1:1.2.2-1) ...
Setting up xdg-utils (1.1.0~rc1+git20111210-7.4)
```

Figure 4: Plugin management lets you supplement Volumio with additional features, like support for the Raspberry Pi Display.

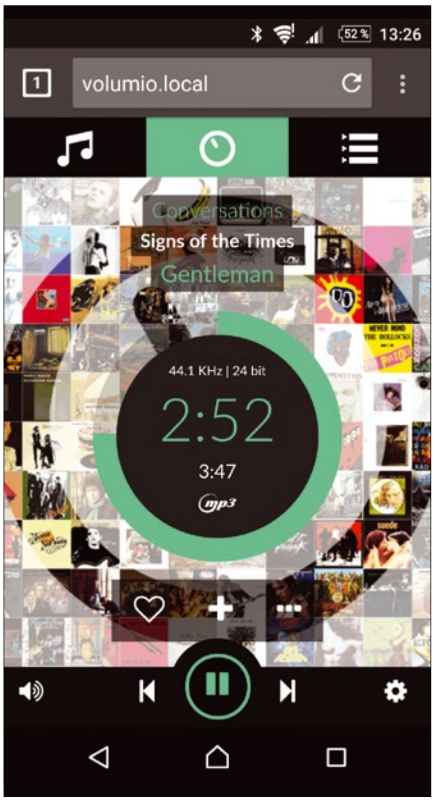


Figure 3: The web interface shrinks to suit smartphones, allowing easy control of Volumio from a cellphone.

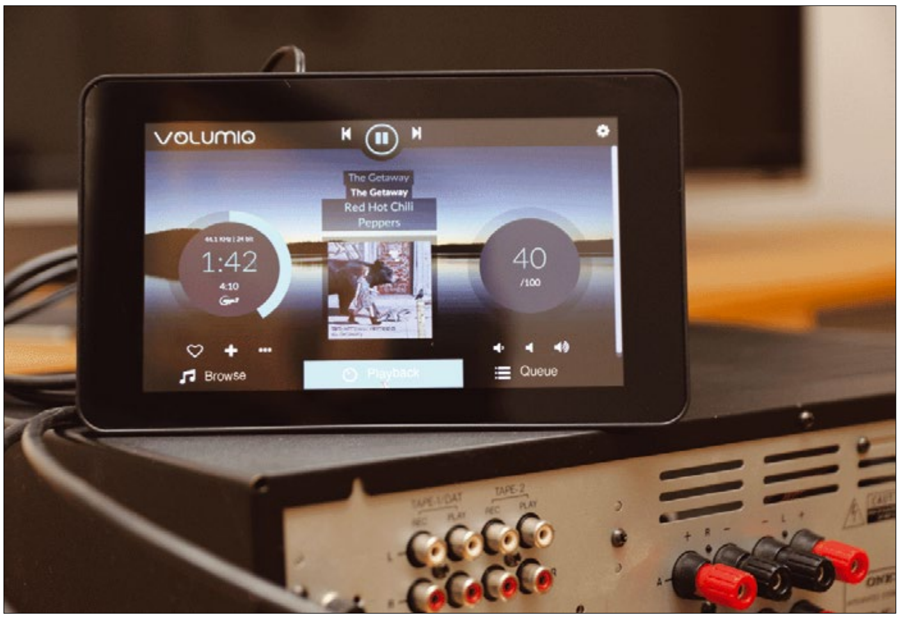


Figure 5: The Touch Display plugin is optimized for the official Raspberry Pi Display but also works with other monitors.

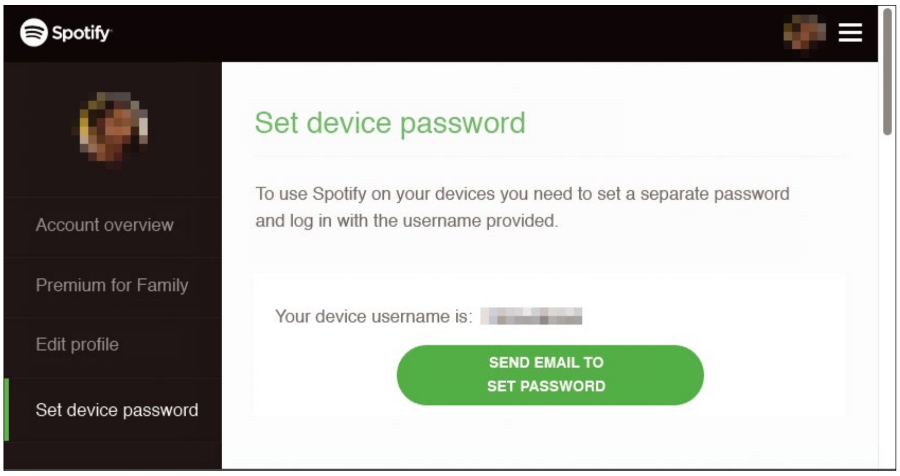
the *Search Plugins* tab. The *Touch Display Plugin* is found under *Miscelleanea* [sic], and you can install the module on the system by pressing *Install*. Behind the scenes, Volumio installs the X server and Chromium browser from the Raspbian system's package manager – the process takes several minutes (Figure 4). Make sure you disconnect any keyboards from the system before installing; otherwise, the installation routine will hang.

Finally, you still need to activate the plugin. To do so, switch to the *Installed Plugins* tab and slide the Touch Display controller to *On*. Directly after doing so, the software launches the graphical environment with the Volumio web page, which keeps screen distractions from interfering with your listening pleasure and prevents the Rasp Pi from being used or misused for other purposes (Figure 5). The plugin configures the system so that the graphical environment loads automatically, even when restarted.

Spotify

Installing the Spotify plugin from the plugin manager is just like installing the touchscreen function. By outsourcing the Spotify function to an extension, the developers can respond quickly to changes in the service and only need to update the plugin. After installation, activate the extension in the *Installed Plugins* tab. The *Settings* button then appears, from which you can enter your Spotify access data – not your "normal" Spotify data, but a special device password that you will find on the Spotify web page: Log in there and then open your *Account overview*. Lower down on the page, you will find the *Set device password* option. The subsequent dialog shows your device username. After pressing the button, Spotify emails you an individual password (Figure 6). You need to enter this data in the Spotify plugin on Volumio.

After opening your music library from the *Browse* tab on the main screen, you can access your music collection in the *Spotify* entry. Volumio installs your own playlists as well as those offered by the service, shows new releases, and filters by genre and mood. When browsing, Volumio separately lists hits from the catalog of the music service under *Spotify Artists* (Figure 7). The integration of other music services, such as Amazon Prime or Google Play Music, has been considered, but so far Volumio does not support alternative providers.



Set device password

To use Spotify on your devices you need to set a separate password and log in with the username provided.

Your device username is: [redacted]

Figure 6: You need a device password from the settings of your Spotify account to integrate Spotify with Volumio.

Substructure

Technically, Volumio is not based on software developed completely in-house, but on the Music Player Daemon (MPD) [5] music server, which has been around for years and is designed to be controlled through client programs. The project wiki explains the complete architecture [6]. MPD is found in the repositories of many Linux distributions. Volumio enhances the server by adding a web front end and optimizes the service for interaction with a small computer like the Raspberry Pi.

Volumio also can be controlled using classic MPD clients [7], which are available for all major operating systems and for iOS and Android [8] mobile devices (Figure 8). However, weaknesses still exist in daily use. Although Volumio access to legacy mounted media (i.e., USB or the network share) can be controlled nicely by the MPD client, Spotify does not appear in the list. Additionally, local media collide with Spotify streams during playback; for example, while playing a song on

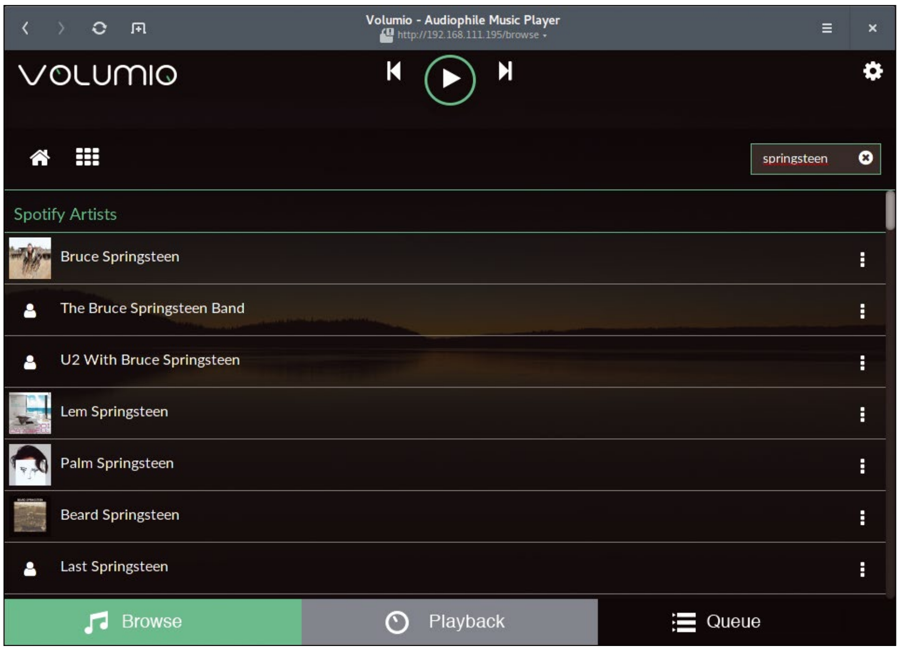

Volumio - Audiophile Music Player
http://192.168.111.195/browse
VOLUMIO
springsteen
Browse
Playback
Queue
Spotify Artists
- Bruce Springsteen
- The Bruce Springsteen Band
- U2 With Bruce Springsteen
- Lem Springsteen
- Palm Springsteen
- Beard Springsteen
- Last Springsteen

Figure 7: Volumio seamlessly integrates Spotify into the system. However, this only works in the web interface.

Spotify in the web front end, you can start playing a second song from the MPD client.

The Volumio web app (available in some countries, but not the US at this time), which Volumio itself offers in the Google Play Store, also shows some weaknesses [9]. Although, it can connect to the Volumio Rasp Pi, the display remains blank. Bad reviews from other users confirm this problem. To control Volumio from a smartphone or tablet, you will want to ignore the various apps and open the Volumio web front end in a browser.

Multiroom

Compared with commercial multiroom stereo systems like Sonos or Raumfeld, Volumio playback cannot be synchronized across several rooms. However, a multiroom system based on the Raspberry Pi is what many Rasp Pi fans desire. Volumio does not offer this feature itself, but developers of the Android app Sound@home for Volumio is committed to resolving this problem.

However, the app was having a hard time keeping pace with the speed of Volumio development. At the time I wrote this article, Sound@home was only optimized for Volumio 1.55, although the application worked with the current version of the audio sever – with limitations. Not until August 2017 was a version strictly for Volumio 2 released [10]. The new version now has an autoconfiguration mechanism for multiple rooms, YouTube plugin support, speed improvements, and a long click function for manual configuration (Figure 9).

Conclusion

Volumio has undergone many improvements since the first versions. Accompanied by a fast Rasp Pi 3, the Raspberry Pi Display, a case, and an output device, the combination creates a web radio that beats most of the commercial alternatives by miles. Volumio can be operated quickly and conveniently – whether by gesture, web browser, or smartphone app. Additionally, the audio player plays back music from local and network sources in various formats and supports the widely used Spotify streaming service.

Volumio has grown an active developer community. You can pick up the source code for the interface and the core components from the project's GitHub account [11], where you can also report bugs and submit your own suggestions for improvement. Answers to your questions about configuration and controlling Volumio are available from the forum on the Volumio community website [12]. ■■■

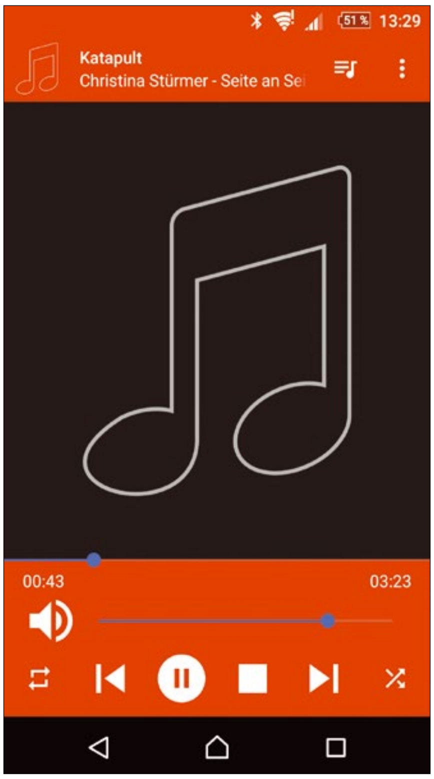


Figure 8: Volumio can also be controlled by MPD clients like M.A.L.P. for Android. However, Spotify integration is missing.

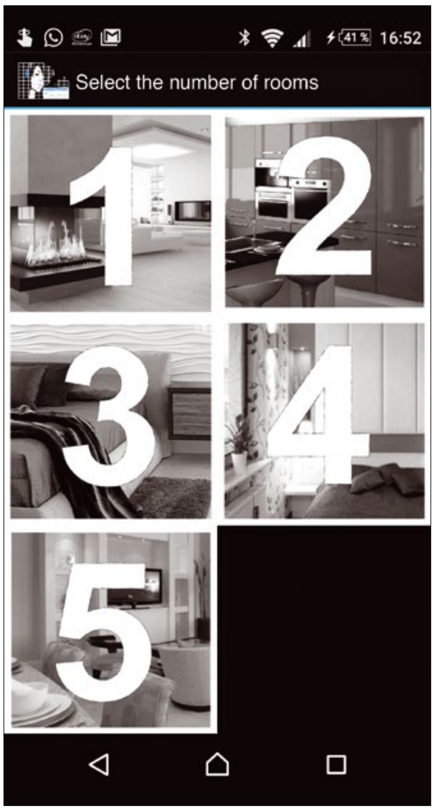


Figure 9: The Android app Sound@home for Volumio autoconfigures multiple rooms.

Info

[1] Volumio: *https://volumio.org*

[2] Volumio 2 shop: *https://volumio.org/volumio-2-stable-release/*

[3] Fing: *https://play.google.com/store/apps/details?id=com.overlook.android.fing*

[4] Tube amplifiers for the Rasp Pi: *http://www.pi2design.com/502hta.html*

[5] MPD: *https://www.musicpd.org*

[6] Volumio architecture overview: *https://github.com/volumio/Volumio2/wiki*

[7] MPD clients: *https://www.musicpd.org/clients/*

[8] M.A.L.P.: *https://play.google.com/store/apps/details?id=org.gateshipone.malp*

[9] Volumio web app: *https://play.google.com/store/apps/details?id=com.volumio.moritz.volumiowebapp_release*

[10] Sound@home for Volumio: *https://play.google.com/store/apps/details?id=com.digx.soundhome*

[11] GitHub: *https://github.com/volumio*

[12] Volumio forum: *https://volumio.org/forum*

LINUX UPDATE

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**

High Performance Computing is a big job ... ... better bring the right tools

Download a FREE ADMIN Magazine Special Edition

LINUX UPDATE
EXPLORING THE WORLD OF LINUX

- JavaScript Alternatives
- Largest Open Source Summit to Date Around the Corner
- Reddit Closing Doors to Open Source
- PhotoQt

FEATURED ARTICLES

JavaScript Alternatives
JavaScript is the stuff of which many interactive web clients is made, but it comes with a fair amount of historical ballast. The creators of four alternative scripting languages seek to ditch the ballast. (more)

Largest Open Source Summit to Date Around the Corner
LinuxCon is now Open Source Summit. (more)

Reddit Closing Doors to Open Source
The peanut gallery of the Internet is closing its open source repository. (more)

PhotoQt
The lean PhotoQt tries to join the ranks of modern image viewers, but it's still not very stable. (more)

FURTHER READING

- Tutorials – Systemd
- Building a Boot Repair Disk
- Master PDF Editor 4
- Calamares
- Red Hat to Drop Support for Btrfs

Complete Your Open Source Library with Archive DVDs
Order Your DVDs Today!

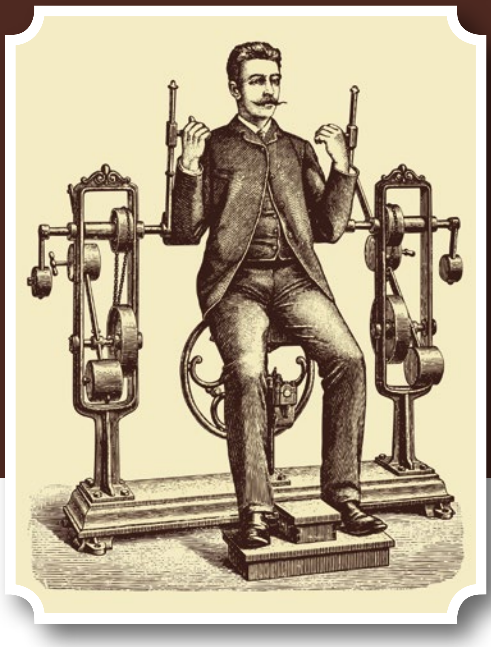
MOST READ

Disk Tests
When it comes to cheap flash memory, buyers should beware. Fake flash memory often offers only a fraction of the advertised storage capacity. With no visible calibration mark, it isn't easy to discover a counterfeit. Here's a test to weed out fake disks. (more)

Real-Time Linux OS Monitoring
Track Linux Performance at Scale

www.linuxpromagazine.com/mc/subscribe

Ft Photography, Fotolia



MakerSpace

Signet password manager

Let's Get Physical

At the intersection of free software and crowdfunding, a USB password manager offers an innovation in security.

By Bruce Byfield

Small, crowdfunded businesses creating innovative open hardware are becoming one of the technological trends of the last few years. For instance, Keyboardio [1] is shipping its first ergonomic, customizable keyboard, while Purism [2] is gaining a reputation for its high-end laptops and is currently building the security-conscious Librem 5 phone. More recently, after a successful fundraising campaign [3], a two-person startup called Nth Dimension [4] is releasing Signet, a USB device for managing passwords that brings a few new twists to security.

Neils Nesse, the founder of Nth Dimension, writes that, "I have been a user and advocate of free and open source software for my entire adult life, although I haven't made many contributions so far outside of a few bug fixes and releasing some small graphics-related libraries on GitHub [5]. I started developing Signet soon after I made a DIY hardware password manager using some instructions online. It worked okay, but the user experience had a lot of pain points, and the device had limited portability. I didn't find any other open source offline hardware password manager options that I liked, so I resolved to create my own. In the long term, I plan to produce other consumer electronic devices, particularly devices where security and privacy are desirable."

Introducing Physical Security

Signet consists of a USB device (Figure 1) and a software client for Android, GNU/Linux, OS X, or Windows (Figure 2). Like any setup designed for security, Signet is based on encryption – specifically, the AES-256 standard [6] with cipher blockchaining [7] for authentication and encryption of the database. The encryption for each database entry is encrypted as a blockchain with unique initial blocks, which eliminates the possibility that similar blocks might be used for more than one entry and makes cracking more difficult.

However, the use of an external device and the exchange of information between the device and the client allows for a number of unique security features (Figure 3). Placing the password manager on a USB thumb drive provides elements of physical security – an aspect of security that is so simple that it is often overlooked. Unless the Signet device is plugged into the system, access to the information it manages – such as logins, bookmarks, contacts, and credit card numbers – is inaccessible. That means that a system can be secured simply by

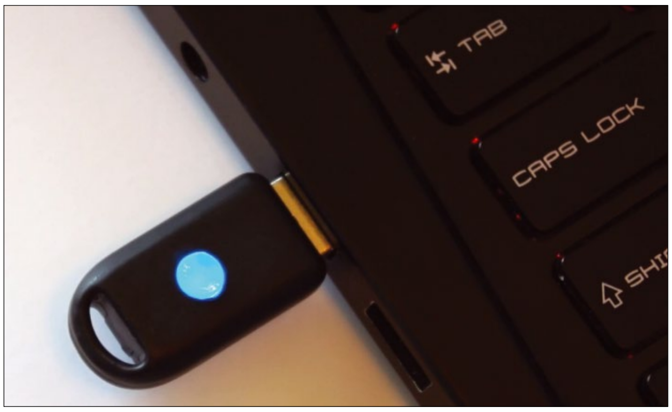


Figure 1: The first part of Signet is an external USB device, which must be present for a login to work.

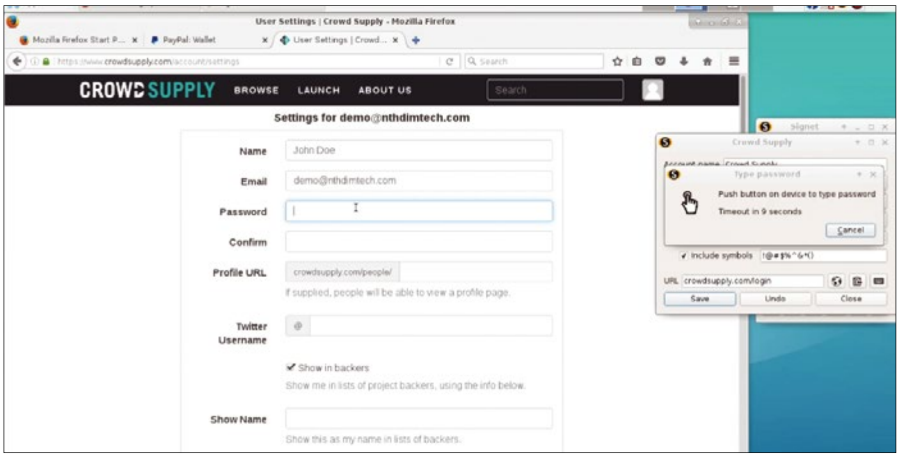
removing the device and carrying it around with you.

Even when Signet is available, the information it manages can only be accessed by pressing the device's button. When the device receives any command that is "sensitive" – that is, any command that reveals private information or is destructive – the button flashes, and the command is suppressed until either the button is pressed or the time to press the button expires and the command is rejected. This arrangement means that cracking Signet's database is of no use by itself. Moreover, Nesse says, "if there is any malicious software on the system you are using, it can only intercept data when you request it, rather than it being potentially able to get a complete copy all at once."

The main potential vulnerability occurs only if you back up Signet's database to the USB device. Even then, the database is encrypted. However, even this vulnerability can be avoided by backing up the Signet database elsewhere. Nesse recommends that other "removable media backup are probably the most secure, provided you don't use the drive you select on unsecured systems."

Moreover, Signet's hardware design choices provide additional security. Information is stored inside the microcontroller's on-chip flash memory, which, according to Nesse, can only be attacked "by desoldering the memory chip and reading out the memory contents in a separate circuit." Furthermore, the microcontroller and ARM-based chip have a memory protection mode that can be enabled to prevent the chip from using its hardware debug mode and to prevent the activation of the factory boot mode.

Additionally, any firmware updates can only be applied by unlocking the device first. In theory, the data might be cracked by a brute-force attack [8], but as Nesse points out, such an effort would be "impractical." Signet uses scrypt [9], an algorithm that is so memory-intensive that each attempt at authentication takes hundreds of milliseconds. For a legitimate user who is authenticating once, this delay hardly matters. However, since a brute-force attack

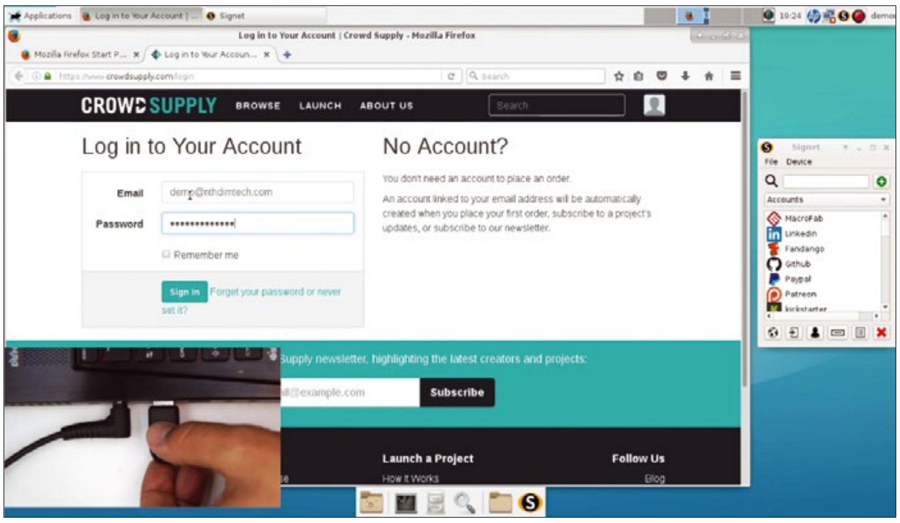


Settings for demo@nthdimtech.com

Name: John Doe
Email: demo@nthdimtech.com
Password:
Confirm:
Profile URL: crowdsupply.com/people/
If supplied, people will be able to view a profile page.
Twitter Username: @
✓ Show in backers
Show me in lists of project backers, using the info below.
Show Name:
Show this as my name in lists of backers.

Push button on device to type password
Timeout in 9 seconds
✓ Include symbols
URL: crowdsupply.com/login

Figure 2: The second part of Signet is a software client.



Log in to Your Account

Email: demo@nthdimtech.com
Password: •••••••••••••
Remember me
Sign In Forgot your password or never set it?

No Account?

You don't need an account to place an order.

An account linked to your email address will be automatically created when you place your first order, subscribe to a project's updates, or subscribe to our newsletter.

Figure 3: Both the Signet USB device and the software client are required to log in to protected data.

by definition requires multiple guesses of the password, unless an obvious password is used, the delay soon mounts up, and any attack would take too long to have much chance of success.

Still another aspect of physical security is that the device's encryption key is randomly generated by three different sources of random data: the hardware random number generator on the microcontrollers, random data from the host, and random data generated by measuring variation on two different oscillators in the microcontroller. These multiple sources not only help to ensure that the encryption key is truly random, but it also means that both the Signet device and its devices on the system must be present to access information.

To further add to the security, Signet also has restrictions that prevent two programs from accessing the USB device at the same time. Because of this restriction, communication between the device and the client software is not encrypted. "This might seem like an oversight," Nesse says, "but if the system you are using is compromised, then the communication link between the device and the application is hardly the only place where the data could be intercepted. A keylogger could capture the data as it's being typed into the GUI, or when the USB keyboard function of the device types some private data. Even if encryption was used, the client's key could be extracted from RAM at run time, or the client could be replaced with a hacked client. The only way to really counter [these possibilities] is by limiting the types of data you access on systems that you have less trust in."

Advantages over Local and Cloud Password Managers

As Nesse notes, both hard drive and cloud-based password managers are widely used. However, Signet has advantages over both.

"For a user who keeps their offline database only on their home systems and secures them well," Nesse says, "the physical security offered by Signet might not matter as much. However, many people have to use a number of computes and networks that they don't have much control over: both work and school systems, often with proprietary operating systems. This reality forces a choice between not logging in outside of the home, choosing duplicated or easy to remember passwords to make the password manager less essential, or making copies of their password database. Signet, on the other hand, takes away the incentive to accept these kinds of security risk factors."

Moreover, a password manager installed on the same system as the data it is protecting is only as secure as the system itself. The introduction of a secure external device makes an intrusion much more difficult. Signet reduces the risk even further by receiving only a set of metadata when databases are unlocked – not a complete copy of the database.

Similarly, while cloud storage or services are convenient for users who regularly work from more than one system, as Nesse notes, "the question is whether or not it makes sense to store all of your passwords and other identifying information outside of your physical control." Unless you use some additional security measures such as Least-Authority File Store (LAFS) [10], only one source needs to be cracked for an intruder to have complete access to your data – and you may not know what has happened until long after the fact, if ever. "It's difficult to determine the likelihood of this happening," Nesse says, "but every service I've looked into has a spotty track record."

True, as Nesse admits, Signet is also a single source for your data. However, he adds, "it's a physical one, and you can study or even modify how it works," since it is open source. Any attack "would still require the attacker to get a hold of your device or gain access to the backup of the device's data. Even if you are being personally targeted, it's a risky operation. Going after all or a significant number of Signet users would be even more difficult and impractical," because each Signet device would have to be cracked individually. By contrast, a cloud database is centralized, and the sheer number of users makes it a far better target for an intruder. Nor do users have anything beyond a vendor's assurances about a cloud database's security, especially if it uses proprietary software.

Next Steps

Nth Dimension has exceeded its fundraising goal of $2,000 by over 500 hundred percent. As a result, Nesse is currently taking a break from his day job to fulfill the campaign's stretch goals. These goals include command-line tools, which should be ready for the first shipment of Signet, and browser plugins, which Nesse expects to be ready by early 2018. Nesse also hopes eventually to add support for GPG encryption, which "would allow Signet to manage the encryption of media and communications, keeping sensitive private encryption keys off the host system."

Other enhancements Nesse hopes to add at an unspecified future date are a feature that indicates password strength and the ability to start the client automatically when the USB device is inserted. "Other than that, I'm pretty happy with the desktop experience," he says. "I've been using Signet personally in various forms almost a year, so when things happen that bother me, I fix them fast."

Whether Nth Dimension will be a success remains to be seen, although the number of backers for its fundraising campaign gives the new company a chance for at least modest success. However, whether or not Nth Dimension is a financial success, in Signet, the company has already proved itself a source of technological innovation – and one that wouldn't exist without the intersection of free software and crowdfunding. ■■■

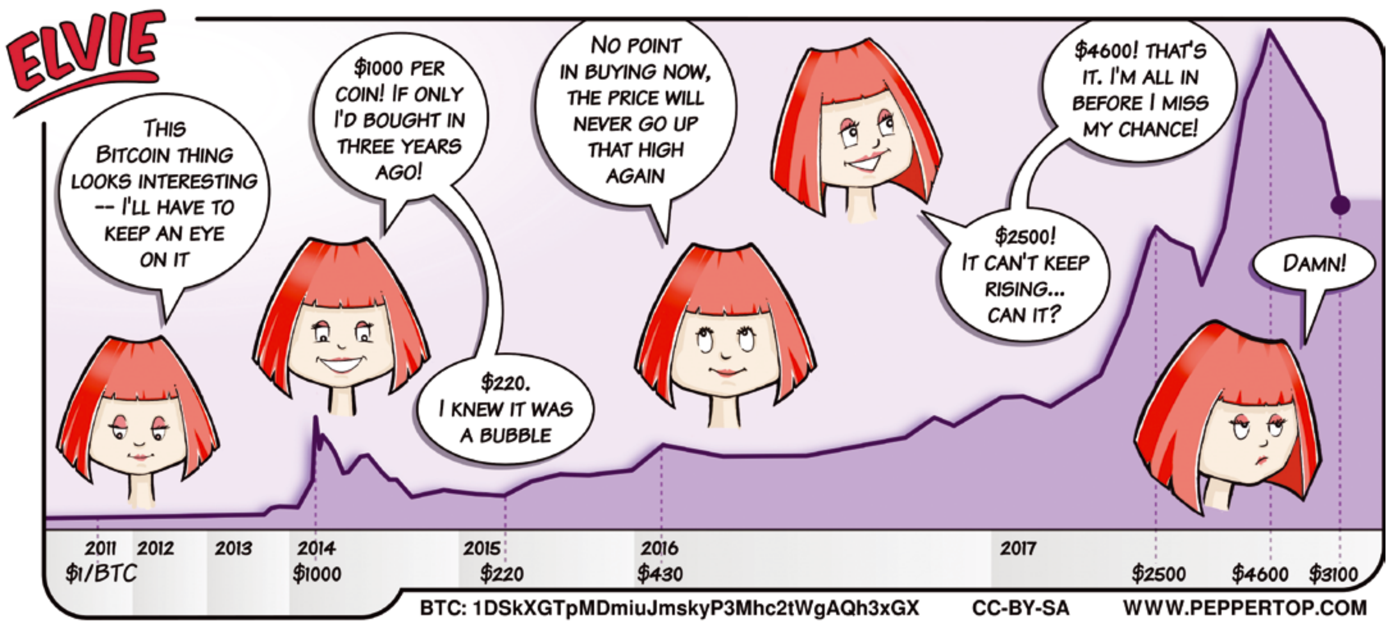
Info

[1] Keyboardio: *https://shop.keyboard.io/*
[2] Purism: *https://puri.sm/*
[3] Crowdfunding campaign: *https://www.crowdsupply.com/nth-dimension/signet*
[4] Nth Dimension: *http://nthdimtech.com/*
[5] GitHub repositories: *https://github.com/nthdimtech*
[6] AES-256: *https://en.wikipedia.org/wiki/Advanced_Encryption_Standard*
[7] Cipher block chaining: *https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#CBC*
[8] Brute-force attacks: *https://en.wikipedia.org/wiki/Brute-force_attack*
[9] LAFS: *https://en.wikipedia.org/wiki/Scrypt*
[10] scrypt: *https://tahoe-lafs.org/trac/tahoe-lafs*

Linux has a way of keeping things real. You can pay a lot for proprietary software if you listen to the people who sell it, but before you flash your debit card, be sure you can't find a free Linux tool that does the same thing. Multimedia tools are an especially lucrative market for software vendors. If you come from a Windows or Mac environment, you can pay hundreds of dollars for audio and video applications. What better reason to turn to Linux, where similar tools are free and often easier to use. This month we explore some audio and video tools for the Linux environment, including the Audacity sound editor and the convenient FFmpeg command-line video-editing utility.

Image © Olexandr Moroz, 123RF.com

LINUXVOICE


ELVIE
THIS BITCOIN THING LOOKS INTERESTING -- I'LL HAVE TO KEEP AN EYE ON IT
$1000 PER COIN! IF ONLY I'D BOUGHT IN THREE YEARS AGO!
$220. I KNEW IT WAS A BUBBLE
NO POINT IN BUYING NOW, THE PRICE WILL NEVER GO UP THAT HIGH AGAIN
$2500! IT CAN'T KEEP RISING... CAN IT?
$4600! THAT'S IT. I'M ALL IN BEFORE I MISS MY CHANCE!
DAMN!
2011 2012 2013 2014 2015 2016 2017
$1/BTC $1000 $220 $430 $2500 $4600 $3100
BTC: 1DSkXGTpMDmiuJmskyP3Mhc2tWgAQh3xGX
CC-BY-SA
WWW.PEPPERTOP.COM

NEWSANALYSIS

The Linux Voice view on what's going on in the world of Free Software.

Opinion

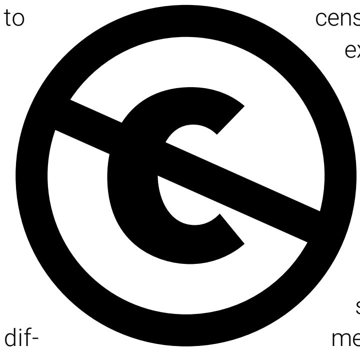
The Universal Donor

Open source communities need equal rights. BY SIMON PHIPPS

Simon Phipps is a board member of the Open Source Initiative, the Open Rights Group, and The Document Foundation (makers of LibreOffice).

A few people reacted negatively to my article on why Public Domain Software (PDS) is broadly unsuitable for inclusion in a community open source project. Most argued that because public domain gave them the rights they need where they live (mostly the USA), I should not say it was wrong to use it.

That demonstrates either parochialism or a misunderstanding of what public domain really means. It should not be used for the same reason code known to be subject to software patents should not be used – namely that only code that, to the best efforts possible, can be used by anyone, anywhere, without the need to ask permission (e.g., by buying a patent license) or to check if it's needed (e.g., is that PDS code public domain here?) to be used in an open source project. Public domain fails the test for multiple reasons: global differences in copyright terms, copyright as an unalienable moral rather than as a property right, and more.



Yes, public domain may give you the rights you need. But in an open source project, it's not enough for you to determine you personally have the rights you need. In order to function, every user and contributor of the project needs prior confidence that they can use, improve, and share the code, regardless of their location or the use to which they put it. That confidence also has to extend to their colleagues, customers, and community as well.

Some members of the Apache Software Foundation (ASF) describe this condition as "being a universal donor" of software. The ASF has rigorous rules concerning the licensing of all the software they maintain. They require contributors to confirm the originality or prove origin of their contributions and to grant to the ASF as a legal entity unrestricted copyright and patent rights. They prohibit use of licenses that might require code recipients to take further licensing actions – this even means banning some OSI-approved licenses such as the GPL.

They have recently also banned use of additional patent statements that modify rights under otherwise acceptable licenses (the recent Facebook example was the origin of this). They permit public domain code, but only after a large amount of due diligence to ensure everyone has the necessary rights. Their goal is to ensure all known obstacles to use, improvement, and sharing are removed in advance, so that their own community can innovate freely and so that unknown others can freely use Apache code.

PDS per se breaks this condition of being a universal donor. While there is a good chance that many contributors will discover they have sufficient rights, the problem is that every one of them has to make their own local determination. That breaks the community. It's not enough that you have the rights you need; in a community, everyone needs those rights, and by including public domain code, you give everyone a burden rather than a benefit. ■■■

MADDOG'S DOGHOUSE

Take some advice from Linus Torvalds and learn how to program an FPGA. BY JON "MADDOG" HALL

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®.

Field-programmable gate arrays

I was at LinuxCon in Beijing, China, this year, and Linus Torvalds was being interviewed onstage. During the interview, Linus said that if he were starting over in FOSS today, he might not create a kernel but instead, learn how to program a field-programmable gate array (FPGA). I found this interesting.

What is an FPGA? Traditional central processing units (CPUs) and graphics processing units (GPUs) are made of fixed circuits designed by computer engineers that form an "instruction set." This instruction set tells the hardware to do certain things when an instruction (designated by a program) is executed. All of this is controlled by a clock that acts like a drum keeping a beat and makes sure that everything works in harmony.

However, another way of building a computational unit is to build specialized circuitry that always processes data in a particular way – like "hard coding" the circuitry. Once the circuitry is set, there is only one thing that circuitry can do. Because the circuitry does not necessarily have to follow a "drumbeat," it can often solve that particular problem 10, 20, or even 100 times faster than a traditional CPU or GPU.

Of course, most of us do not want to buy a computer for every single problem we want to solve, so the programmer is able to change the program on the FPGA to solve different problems. One moment, the FPGA might be compressing an image; the next moment it might be encrypting (or decrypting) a stream of data. If you combine one (or more) FPGAs in a system along with a CPU and GPU, the FPGAs can offload a lot of processing from the main processing units, allowing them to do more varied tasks while the FPGA does the computation it was programmed to do.

Programming an FPGA does not follow the same procedural logic that programming a CPU or GPU normally does. Special languages and tool chains are used to build the program, and it does take some study to learn how to use them. Also, in the past, FPGAs were very expensive (and large), so not too many people learned how to program them. Now they are inexpensive and getting more inexpensive by the day. Some companies are now building FPGA functionality into their Systems on a Chip (SoCs).

FPGAs are not just about speed of computation; they can be used to reduce the cost of energy, as well. If your main CPU has to run for a long time to solve a problem, it may use a *lot* of electrical power. But if the FPGA can do the calculation quickly, then the main CPU might be able to go into "standby" mode and use less electrical power.

The Mars Rover was designed a long time ago, and although it does have regular CPUs on board, they are relatively slow and consume a lot of power. On the other hand, the Rover has a significant number of FPGAs on board that do the "heavy lifting," which allows it to reduce its overall use of electricity by turning the FPGAs off when they are not in use.

People are often confused by the term "real time." To me real time means that the computation occurs as the data comes in. If you are not doing the data processing in real time, then the data has to stop or be buffered until the processing element can "catch up." In most data processing, this causes a backlog that may never be overcome. FPGAs can be used to keep this data flowing, so you never have to stop the data flow.

FPGAs can have another useful function, particularly in embedded systems. Many GPUs have proprietary, binary-only firmware that can prevent new versions of the kernel from working unless the firmware is updated. I call these "binary blobs." I was lamenting to Professor Marcelo Zuffo of the University of Sao Paulo that many 3D functions (particularly "simple" ones) could be done by an FPGA and therefore eliminate the need for proprietary GPU binary blobs. Three months later, two graduate students had programmed an FPGA to run all of the OpenGL test suites. By using an FPGA for simple 3D and 2D graphics processing, embedded systems do not suffer when companies drop support for their GPUs (or the operating system that is using the GPUs).

One last idea on GPUs is using them for crypto-currency mining. While your main CPU is running the system, your FPGA can be taking care of the mining process.

Therefore, I recommend finding an inexpensive FPGA and learning to program it. Don't let Linus have all the fun! ■■■

How to Sell Open Source

Marketing FOSS requires some novel approaches compared with proprietary software. We share our experiences.

BY MIKE SAUNDERS

You've heard this sentence so many times: "Free and open source software like Linux doesn't need a marketing department!" To some extent, there's some truth to it: People learn about FOSS thanks to word-of-mouth and grassroots movements. Many of us discovered Linux from friends and colleagues or from reading about it in online discussions, rather than from flashy TV adverts. We keep using it because we like it, not because our brains are being toyed with. We'll keep using, promoting, and supporting Linux regardless of what happens to the companies developing and using it.

On the other hand, Linux and FOSS exist in a highly competitive (and often unpleasant) market. If you want it to grow, you need to consider the marketing strategies and approaches that the competitors are using and combat the "fear, uncertainty, and doubt" that's often spread by groups and individuals that want to cause FOSS harm. You want to present the software you love in a positive light, even when things are not going so well (e.g., with the Heartbleed security vulnerability).

Marketing open source is something everyone can do, whether it's using social media effectively, creating videos or infographics to explain FOSS, or attending events and giving talks. Over the next few pages, I'll examine some of the traditional approaches to software marketing, look at examples from Firefox and IBM, and provide some tips that you can use.

Open vs. Closed

Imagine you're a newly appointed marketing manager for a proprietary software company. What's your responsibility? Well, you'll want to examine your current product portfolio and see how those products fit into the market (e.g., who they're targeting, whether there are growth opportunities, and whether the pricing is right). You will also look at products in development and see what opportunities are ahead. You can then work on marketing materials to describe, promote, and sell your products to the right people. As you become more experienced, you can work more closely with product developers to shape the final result.

Now, imagine you have the same role, but within an open source project (or a company that sells FOSS). Some of your responsibilities will be the same, but others will differ significantly. For instance, when an open source program is being developed by a large community of volunteers around the world, you can't easily shape its development. Sure, you can do market research and find out what features people need – but then what? Post them on a mailing list and see what happens? When developers are scratching their own itches and not being paid by you, it's harder for you to push development in the direction you want.

Similarly, if you're not selling the software as a product but rather support contracts and other services around it (like many notable FOSS-friendly companies), then your approach has to change. And on top of that, you don't just want end users to try your product, you want potential contributors to get involved, as well. So your marketing job ends up stretching out into community outreach and management.

Classic Example 1: IBM

Back in the late 1990s, GNU/Linux was emerging as a decent server operating system and gaining some use in small companies and ISPs, but it still lacked major commercial backing. Various companies in the FOSS ecosystem supported it – such as Red Hat – but they were small fries at the time. Linux was still very much the domain of geeks and hackers, and it didn't have the image or money behind it to really take off.

That is, until IBM got involved. In 2000, IBM made a major announcement: The company would spend $1 billion on Linux development over the next year [1]. The reason? According to IBM chief executive Louis Gerstner at the time: The company "is convinced that Linux can do for business applications what the Internet did for networking and communications." IBM already had its own gamut of operating systems, including a flavor of Unix called AIX, but it was showing serious commitment to the open source fledgling by making the decision to back Linux.

While that $1 billion went toward improving GNU/Linux and other FOSS tools, IBM also started marketing Linux. Most notably, in 2004, the company produced a 90-second video advertisement called "Prodigy" that can still be found on YouTube today [2].

In the video, a small boy is sitting alone in a white room (see Figure 1). Over the 90 seconds, various famous people approach the boy and offer him advice in all walks of life – even Muhammad Ali makes an appearance, telling the kid: "Speak your mind. Don't back down." While all of this is going on, two unidentified observers are making notes. At the end of the video, one asks the other: "What's his name?" The response is: "His name is Linux"; then the advert ends.

The video is striking and creative, and it takes a novel approach to marketing software. IBM didn't shy away from positioning Linux as the young and relatively inexperienced contender, but it focused on its ability to learn and absorb information (i.e., through the open source community).

What's especially notable is the positioning of this advert. This wasn't some cute in-joke for geeks at a conference or shown internally at IBM meetings. No, it was shown during the 2005 Super Bowl – one of the most viewed events in the world. IBM was reaching out way further than pointy-haired IT bosses. IBM wanted everyone to know that it was serious about Linux.

Classic Example 2: Firefox

Still, IBM was focused on large Linux deployments in enterprises. If you want an example of consumer-oriented FOSS marketing, you can turn to the Mozilla Foundation's spectacular push for the mass market in 2004. Firefox 1.0 had just been released, Internet Explorer was still hugely dominant, and Firefox was seen as a way to introduce the masses to FOSS – ideally setting them on the path to explore more and maybe even try Linux in the end.

The Mozilla Foundation set up a site called Spread Firefox and sourced donations to be used for marketing. The end result was a two-page advert in *The New York Times* (see Figure 2) that asked readers if they were "fed up" with their web browser – and showed them an alternative. Finally, they added some quotes from users, a snippet of information about the software, and a download URL.

The advert obviously had an effect: Firefox tripled its market share over the following 18 months, breaking the 10 percent mark in June 2006. (It peaked at 32 percent in 2009.) Of course, other factors were involved as well, but it's a great example of how a FOSS project can reach out to a wider audience in a clear, targeted, and effective way.

Figure 1: IBM's "Prodigy" advert in 2004 introduced Linux to millions of people across the USA.

Risky Approaches

When advocating and marketing open source software, it's very tempting to make bold claims and sweeping generalizations. How many times have you heard the statement "Open source is more secure than proprietary software" (or some variant on that), for example? You know there's some truth in that statement; history has demonstrated again and again that proprietary software is often riddled with security holes that are exploited by not-so-nice people for years before they become public.

However, you have to be careful. You can say "open source is more secure," and then something like Heartbleed [3] happens (Figure 3). If you're out of the loop, Heartbleed is a whopping security vul-

Figure 2: Mozilla reached out to disgruntled web surfers with a two-page advert in *The New York Times*.

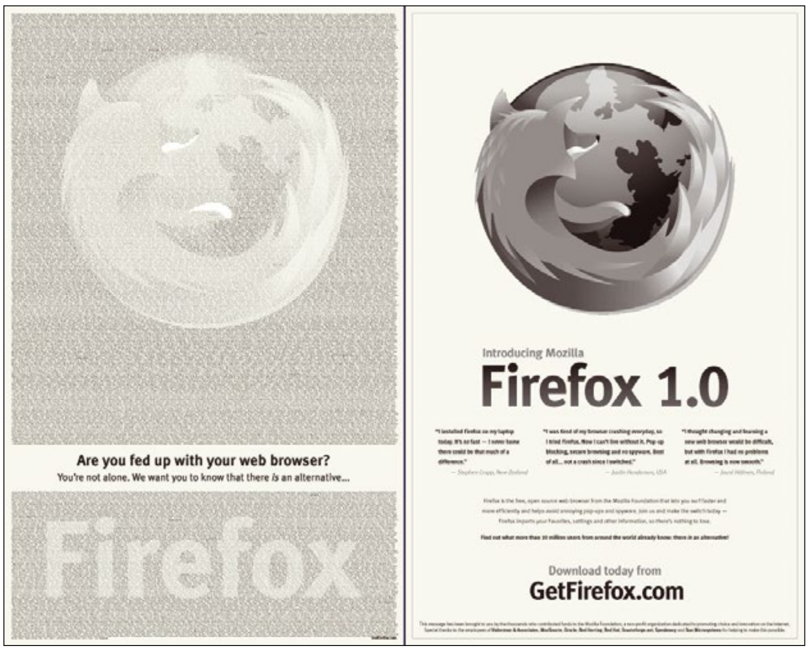

Are you fed up with your web browser?
You're not alone. We want you to know that there *is* an alternative...
Firefox
Introducing Mozilla
Firefox 1.0
Download today from
GetFirefox.com

Figure 3: Be careful when marketing FOSS using "it's more secure" arguments! Things like Heartbleed can still happen.

nerability in the OpenSSL crypto library, which is used pretty much everywhere. The bug was disclosed in 2014, and vast numbers of websites were affected. It was so bad that the OpenBSD folks forked the library to do a massive cleanup, in the form of LibreSSL.

Most importantly, though, those of us who'd trotted out the mantra "open source is more secure" ended up with egg on our faces. Yes, we still believed that statement, on the whole, but Heartbleed provided nay-sayers with a big chunk of ammunition: Open source is also insecure, open source has had vulnerabilities sitting in the code for years, and so forth.

Therefore, it's important to state that open source itself doesn't just magically make things secure; rather, it's the development process. You could even argue that OpenSSL was barely open source to begin with – yes, the code was there, but hardly anyone was working on it, and those who poked around inside tended to run away screaming. The oft-quoted line "Given enough eyeballs, all bugs are shallow" didn't really apply here.

With this example in mind, you shouldn't say "open source is more secure"; instead, clarify it somewhat: "Open source development processes generally lead to more secure software." It's not as snappy, and it leads to more questions, but it's more honest. I think the same phrasing should be applied in terms of reliability as well – Linux adherents know that GNU/Linux is pretty rock solid, but they need to remember that other users may be affected by corner case bugs. As tempting as it is to poke fun at Microsoft for "blue screens of death," Linux is not competing against Windows ME any more. Windows has plenty of problems, and I'm glad Linux can avoid them, but for many users, it's pretty reliable.

Money, Money, Money

Another area where caution is required is price. Yes, GNU/Linux is free – as in beer – to obtain, but many other factors come into play when determining how much it really costs. For instance, if you need to replace a piece of hardware for something that's Linux compatible, that bumps up the price. If you want to deploy Linux (or other open source software, such as LibreOffice) in a large company or government body, end users might need training – which also affects the end price.

For this reason, many IT purchasing people talk about total cost of ownership (TCO). How much will it actually cost to use program X, given the required hardware changes/upgrades, staff retraining, and support costs? In the short term, this doesn't always look so great for Linux and FOSS – if a large company is already using a proprietary system, perhaps getting large discounts for bulk purchases, then a switch to a FOSS solution could initially be very costly.

So you need to bear this in mind when marketing FOSS. Don't just say "Linux is free," because pretty much every large deployment will require support of some kind, whether from the likes of Red Hat, SUSE, or Canonical or from in-house IT staff. A better approach is to say: "Over the long run, Linux/FOSS can reduce the total cost of ownership, with no need for license fees, along with improved reliability and security." Again, you may have to qualify the "reliability" and "security" parts somewhat, but it's the right way to sell FOSS.

Also on the subject of support: A good marketing strategy is to highlight that, with FOSS, you often have many more choices for support. As an example, if a company has thousands of PCs running Microsoft Office and a serious bug is affecting many users, what can the company do? Call Microsoft, maybe spend some more money, and hope that it gets fixed in the next release? If that doesn't work, they're up a certain creek without the slightest hint of a paddle.

Contrast this with a company that deploys LibreOffice: If a bug is affecting workers, the company can choose from many certified developers [4] and hire one to (one hopes!) fix it. Sure, it still costs money, but the company is not reliant on a single vendor and can shop around for solutions (including local ones). It's the free market at work – so much for FOSS being anti-capitalist, as some people used to say!

What Can You Do?

Many FOSS projects are in desperate need of marketing. Maybe their website is subpar, with no proper information on what the app actually does (alarmingly, a common occurrence). Maybe the project is making great progress but is not communicating this effectively with the outside world. Perhaps the project could reach out to new users and potential contributors via social media, Reddit, or Hacker News, but nobody involved has the skills or know-how.

Of course, most FOSS projects don't have the budget to hire a full-time paid marketing person, but if you're interested in this field, dive in and offer to help. As with all things in FOSS, you don't need to be an expert in the field – any help is appreciated. Plenty of books are out there that provide an introduction to marketing, and you can

Figure 4: Mozilla's marketing community even has its own mascot.

take those basic skills and techniques and apply them to the open source projects you use and want to support.

You can also look at the materials and processes used by large and established FOSS projects. For instance, the Mozilla community has a marketing guide [5] and other materials, including web and print materials [6] (Figure 4). Fedora, meanwhile, has a marketing team that's based around a mailing list [7]; in LibreOffice, there's a small community working on presentations, press releases, and other materials (Figure 5), and it organizes through calls once a month [8].

Of course, in the process, you'll build experience that you could potentially use in a paid marketing job some day, and if you end up rich with your own private island, don't forget about the humble *Linux Magazine* writers who set you off on your new career path. ∎∎∎

Info

[1] IBM and Linux: *https://www.cnet.com/news/ibm-to-spend-1-billion-on-linux-in-2001/*

[2] IBM "Prodigy" advertisement: *https://www.youtube.com/watch?v=s7dTjpvakmA*

[3] Heartbleed: *http://heartbleed.com*

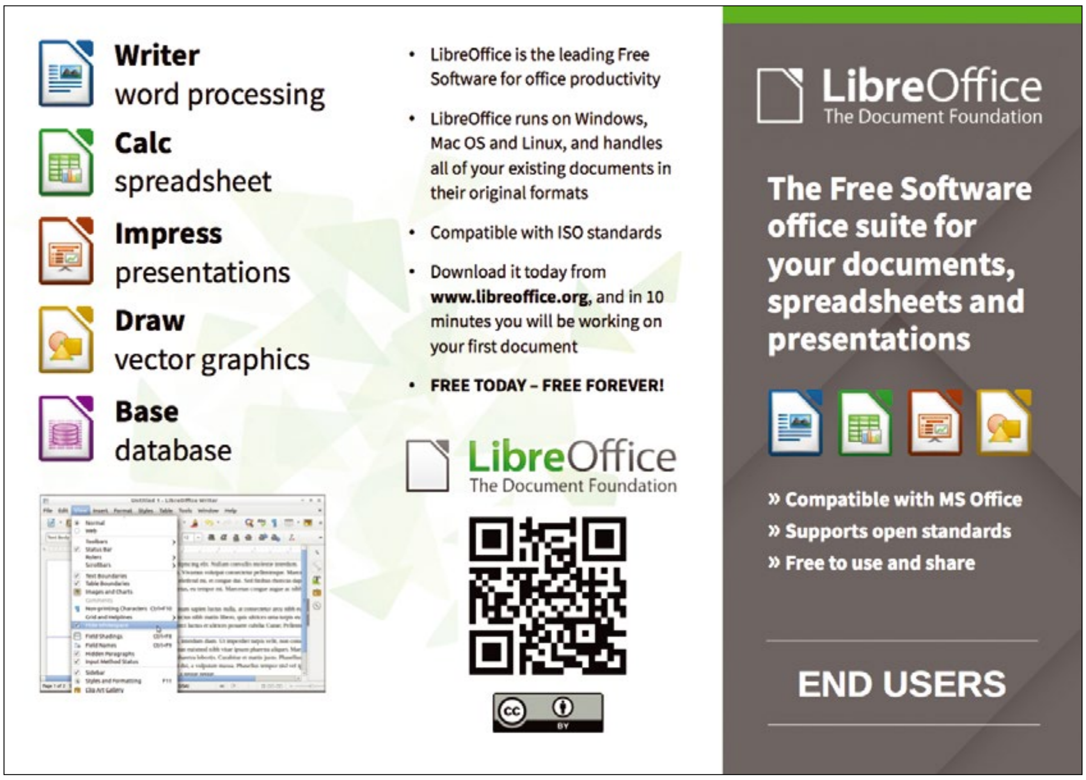
[4] Certified LibreOffice developers: *http://www.documentfoundation.org/gethelp/developers/*

[5] Mozilla marketing guide: *https://wiki.mozilla.org/MarketingGuide*

[6] Mozilla web and print materials: *https://wiki.mozilla.org/Marketing:Firefox_Materials*

[7] Fedora marketing: *https://fedoraproject.org/wiki/Marketing*

[8] LibreOffice marketing: *https://wiki.documentfoundation.org/Marketing*



Writer word processing
Calc spreadsheet
Impress presentations
Draw vector graphics
Base database

- LibreOffice is the leading Free Software for office productivity
- LibreOffice runs on Windows, Mac OS and Linux, and handles all of your existing documents in their original formats
- Compatible with ISO standards
- Download it today from **www.libreoffice.org**, and in 10 minutes you will be working on your first document
- **FREE TODAY – FREE FOREVER!**

LibreOffice
The Document Foundation

LibreOffice
The Document Foundation

The Free Software office suite for your documents, spreadsheets and presentations

» Compatible with MS Office
» Supports open standards
» Free to use and share

END USERS

Figure 5: Look at existing FOSS projects for inspiration – for example, the leaflets produced by LibreOffice's marketing team.

Needlework – Digitize your LPs and cassettes with Audacity

Armed with the Audacity sound editor, you can convert the analog content of LPs, tapes, and cassettes to the digital world. BY MARIO BLÄTTERMANN

The good old vinyl LP is currently experiencing a revival – you will even see hipsters listening to cassettes – but it is not a good idea to expose these treasures to the ravages of time: Audacity helps you archive analog music in a digital format on your hard drive.

Analog

In their day, records and cassettes were undoubtedly milestones in the development of music media. Many of us still have shelves full of these now archaic sound storage media in the attic or basement. Whereas vinyl LPs are seeing a growing community of friends flocking back to the fold, the miniature version of the reel-to-reel audiotape has shrunk virtually to irrelevance.

You might already have replaced many of your old treasures with CDs, but some old recordings mastered in professional recording studios were probably only ever available on LP. Certainly private recordings, demos, and local productions of your school band are consigned to the medium of their era. Therefore, it is important to preserve these treasures against decay.

Professional

The easiest way to convert your material to digital is to connect an analog player directly to a digital recorder. Quite a few all-in-one devices on the market that play records have a memory card slot and often even a cassette drive. The quality of the hardware components are typically entry level to middling, and the ability to influence the output is usually around zero.

These limitations often result in unacceptable quality, so it makes more sense to connect your turntable or cassette deck to the sound card of a PC or a USB adapter, which ideally often also includes a matching preamplifier (see the "Connection" box).

For recording, you also need sound editor software, such as the open source tool I use in this article, Audacity [1]. Audacity comes with both basic recording tools and functions for restoring and optimizing the quality of a recording.

Recording in Progress

After launching Audacity, first select *File | New Project* and click to save. Next, click the *Record* button in the toolbar and start playback on the cassette deck or lower the stylus onto the LP. In principle, you can now grab a cup of coffee while you wait for the recording to complete. However, you might want to take the time to watch the two green level indicators at the top right: If they deflect too far to the right and turn red, you either need to reduce the signal intensity from the amplifier or use the slider in Audacity and then restart the recording to avoid distorting the sound.

In general, you need to do this only once, unless you change your setup. To help you find a peak level, it makes sense to listen to particularly loud sections up front and then record them for test purposes. In this way, you can prevent distortion without having to discard the entire side of an LP. When you are done (Figure 1), you can click the *Zoom In* and *Zoom Out* icons in the toolbar to enlarge the recording and check certain areas in more detail.

Spring Cleaning

As analog, mechanically scanned recordings, LPs deliver noise in addition to the payload signal.

Connection

The classical record player, with a needle that follows a record's groove and converts its deflections into audio signals, outputs a signal so weak that it cannot be fed directly into the line input of a sound card. Although some turntables have a built-in preamp, you will typically need an amplifier with a phono input socket. If you don't have one, a special USB preamplifier is a passable solution – especially for laptops, which often no longer support line input. Typically, you need to enable such an external sound card in the system's sound settings or switch to an alternative input. Cassette decks do not need a preamplifier; their output can be fed directly to a line input.

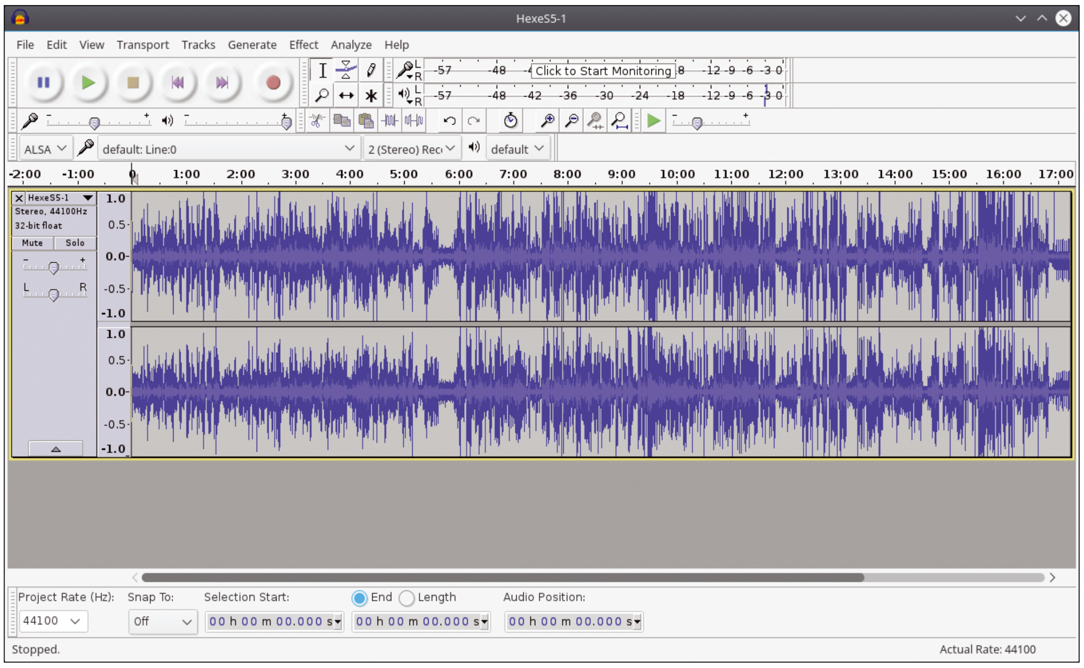


Figure 1: After recording, zoom out once again for a general overview.

Frequently played records pop, crackle, and distort the signal because of dirty grooves. Cleaning with an antistatic brush before playing helps, but other tricks can trim an old recording for a cleaner sound (see the "New for Old" box).

Scratches will be unimpressed by all of these efforts, and the best cleaning will not prevent the resulting pops. Pops are visible as short spikes in the view window; the levels climb steeply and stand out clearly in the wave plot. To correct, select the appropriate section with the mouse and zoom the view with the *Zoom In* icon in the toolbar, by selecting *View | Zoom In*, or with Ctrl+1 until the scratch is shown as a single pulse (Figure 2). This works best using *View | Zoom to Selection* to fit the highlighted area to the view and dragging the markers even further with the mouse.

Now use the *Effect | Fade Out* function to iron out the scratch virtually. Applying the effect once only will not knock the unwanted peak flat straightaway; as a rule, you need to repeat the procedure several times, but when you are done, the pop and crackle in the recording should disappear completely. Be sure to interrupt your work occasionally and save the project to avoid data loss.

Alternatively, Audacity offers a semi-automatic scratch removal tool; just select *Effect | Click Removal* (Figure 3). A little tuning of the default values usually results in clearly audible improvements, but the results can hardly compare with the manual scratch removal method described first, because Audacity lets too many scratches pass. Excessive values can even affect the sounds of the percussion section if the click removal algorithm is overly rigorous, so you should be careful when using this function.

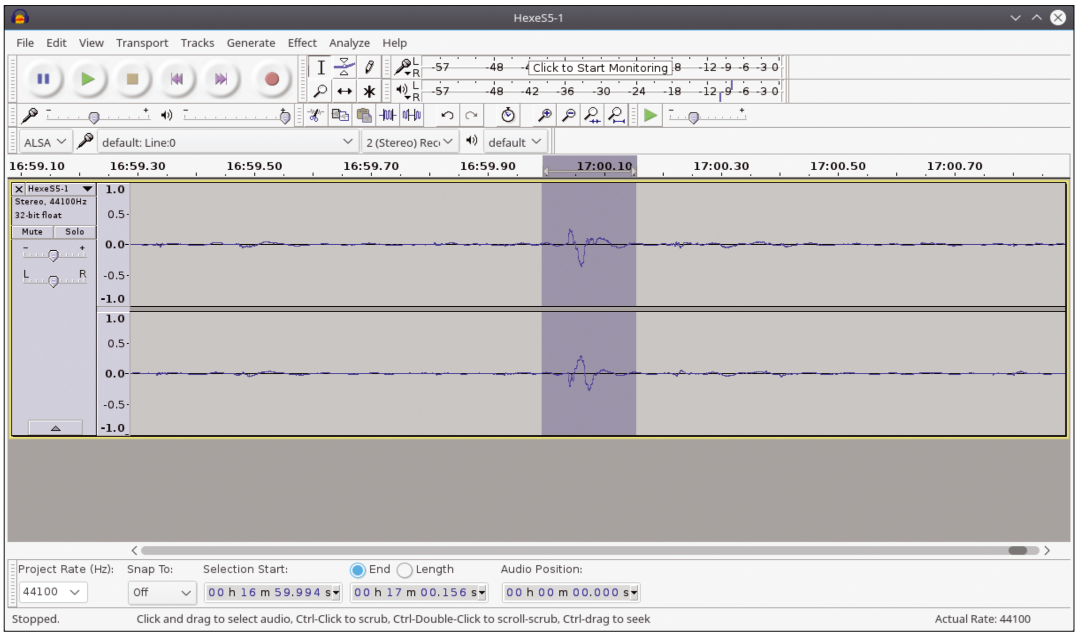
Because an LP is an analog recording, automatic routines will always find life hard. No automatic recipe can remove the typical crackle of vinyl. The *Effect | Noise Reduction* option built into Audacity will achieve moderate improvements, but don't be surprised if the program first asks for a noise profile: The spectral composition of noise is so diverse that it is difficult for Audacity to assess it correctly.

New for Old

Besides the obligatory brush, you have other ways of removing dirt. One method is to place in front of the pickup arm another arm that uses a fine brush to apply a special liquid in the region about to be played back. The typical vinyl pops can be reduced significantly in this way. The liquid evaporates without any residue; however, the LP plays back when wet and will sound more weathered than when played dry.

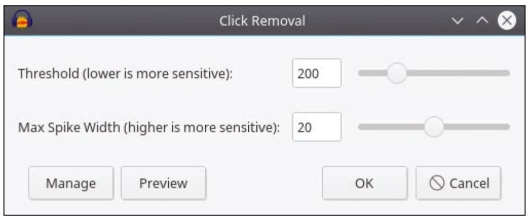
Another alternative is to use a "disk washer," in which two brushes running through distilled water with a detergent clean the record. Covers protect the label, and the record is not played when wet but is left to dry first.

For very stubborn areas in which dirt cannot be removed in the ways already described, you can use a compound found in specialty stores that coats the record and then is pulled off when dry – like a facial mask – along with dirt particles that adhere to it. The dried film is water soluble and can be reused at least a couple of times after boiling in water (although the manufacturers do not advertise this option).



Project Rate (Hz): 44100
Snap To: Off
Selection Start: 00 h 16 m 59.994 s
[✓ End] Length: 00 h 17 m 00.156 s
Audio Position: 00 h 00 m 00.000 s
Stopped.
Click and drag to select audio, Ctrl-Click to scrub, Ctrl-Double-Click to scroll-scrub, Ctrl-drag to seek
Actual Rate: 44100

Figure 2: A scratch is clearly visible with extreme zoom.



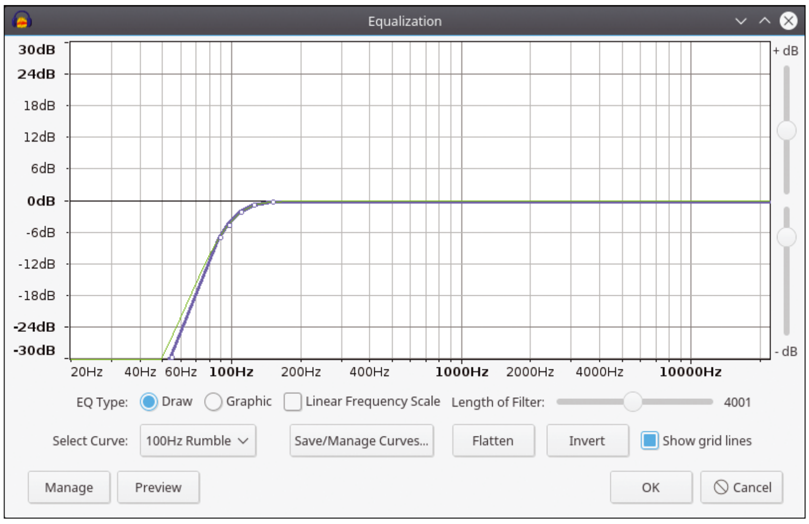
Threshold (lower is more sensitive): 200
Max Spike Width (higher is more sensitive): 20

Figure 3: Audacity can smooth out scratches in a semi-automatic process, but the effect has its limitations.

Mark a small noisy area in front of or between tracks, and click on *Get Noise Profile*. Audacity then stores the profile in the project. Next, select *Edit | Select | All* to select the entire recording on which to apply the effect.

Do not delude yourself: The record will not sound as fresh is it did when you first picked it up from the store. Crackling is far too inhomogeneous for this method; however, you will achieve far better results when processing audio tapes, because their noise is created by the low speed of the magnetized tape and has a much smoother frequency spectrum.

Figure 4: A Rumble filter is useful if the amplifier does not have a high-pass filter.



EQ Type: [✓ Draw] Graphic
Linear Frequency Scale
Length of Filter: 4001
Select Curve: 100Hz Rumble
[✓ Show grid lines]

Sound Development

In addition to noise removal, Audacity can digitally equalize the frequency response. A rumble filter, which minimizes the low-frequency noise of the turntable drive is a typical example. Under *Effect | Equalization* is a settings window (Figure 4), in which you can select from several equalizer profiles. *Rumble*, for example, is a high-pass filter that attenuates low frequencies. If the preset 100Hz swallows too much bass, you can change the curve by dragging it with the mouse to find a workable compromise.

The Recording Industry Association of America (RIAA) standardized the distortion of the frequency response caused by nonlinear movement of the stylus across an LP; many, but not all amplifiers compensate for this. To improve the audio recording, you should take a look at the *RIAA* equalization curve [2], which reduces the trebles and boosts the bass. What sounds better in the end is up to you. The standardized frequency adjustment does not necessarily represent the best choice.

Since the mid-1960s, Dolby Laboratories has developed various procedures to get rid of tape hiss that have consistently improved over the decades. Noise contains especially high frequencies, so the signal in this area is raised and lowered during playback, depending on the input level, which also reduces noise. Dolby SR (spectral recording), used in studio work, is capable of reducing the noise level by up to 25dB. The equalizer is also interesting for tape recordings. A cassette recorded with a Dolby method [3] skews the frequency response when played back with a Dolby-enabled playback device. However, Audacity does not offer a profile for this purpose. For the appropriate standards, see a discussion of Dolby noise reduction online [4].

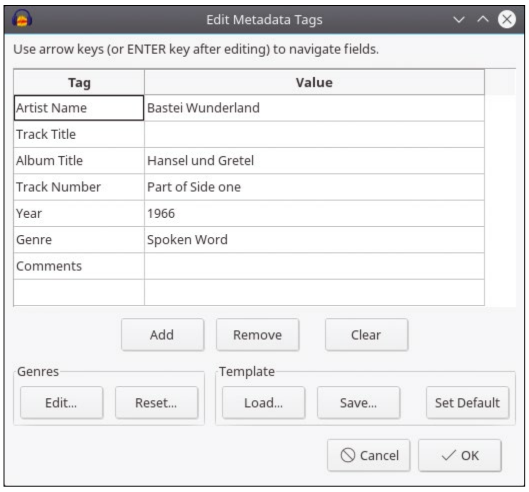
Divide and Conquer

On an LP, but not a cassette, you can see the breaks between tracks with the naked eye. However, when you digitize an LP, the breaks vanish, giving you an album of a single file. Thus, the recording needs to be split into individual tracks.

The timeline above the recording view helps you find the breaks. If the album cover lists play times, you can easily calculate where you need to insert your virtual knife. Simply click on the view and drag while holding down the mouse button to expand a section corresponding to the desired track. Drop-offs in the recording level also give you clues, but they can be misleading, especially with live albums.

In practice, the best approach has proven to be starting at the end of the recording. It is easier to listen to just a few seconds than a whole track, just to keep the fade-out at the end of the track. Once you have found the end, drag the opposite edge of the highlighted area to where you think the track starts. You can find the exact starting point by briefly listening in and moving the marker as needed.

Once the selection fits the bill, export the current title to a standard format for audio players. *File | Export Selected Audio* opens a file dialog that saves the selection as an uncompressed WAV file by default. At bottom right, you can select additional codecs, such as Ogg Vorbis and MP3. When you select these options, additional options appear, such as Bit Rate Mode and Quality. If you prefer to use an exotic encoder, choose *(external program)* instead of a listed codec, where you can specify a custom command to be executed on the command line for an external application.



Use arrow keys (or ENTER key after editing) to navigate fields.

Tag	Value
Artist Name	Bastei Wunderland
Track Title	
Album Title	Hansel und Gretel
Track Number	Part of Side one
Year	1966
Genre	Spoken Word
Comments	

Figure 5: The metadata editor is sufficient but cannot compete with specialist tools.

Rolling up the recording from the back is also a very convenient way to split live albums without losses. For the penultimate title, just drag the end marker of the last track forward with the mouse to the beginning of the new track. In this way, the start marker for the first track you stored becomes the end marker for the new track.

Secret Code

Formerly, digital players obtained information about the artist, album, and track from the file name. The related album cover was only displayed if it was provided in the folder. These days, details of the recording are saved as metadata stored directly in the audio file. When exporting to such formats, a corresponding editor pops up (Figure 5); you then can add information such as the title, artist, album, release year, and more.

By default, the dialog only lists a couple of tags; if you want more, you also need to have a good knowledge of metadata standards for the desired format. Additionally, the editor is not particularly useful if you want to edit an entire album. Tracks exported from the same project appear over and over again with the same tags as the file saved previously. In many cases, it makes sense to export the recordings without the metadata and add the metadata later with a specialist tool such as EasyTag [5].

Conclusions

With relatively little effort, you can rescue LPs and tapes as MP3 or FLAC files for a new life in the digital world. Some caveats still apply; for example, the metadata editor is not always practical. The wxGTK basis of the program is also noticeably ugly in some places. The shift from Gtk2 to Gtk3 occasionally causes display glitches.

Alternative candidates are few and far between. EKO [6] and similar simple sound editors are of limited use for this job. Only KWave [7] offers functionality similar to Audacity; otherwise, you will be hard pressed to find a better audio editor in the repositories. ■■■

Info

[1] Audacity: *http://www.audacityteam.org*

[2] RIAA equalization: *https://en.wikipedia.org/wiki/RIAA_equalization*

[3] Noise reduction systems: *https://en.wikipedia.org/wiki/Dolby_Laboratories*

[4] Dolby profiles: *http://hyperphysics.phy-astr.gsu.edu/hbase/Audio/tape5.html*

[5] EasyTag: *https://wiki.gnome.org/Apps/EasyTAG*

[6] EKO: *http://semiletov.org/eko/*

[7] Alternative KWave: *https://www.kde.org/applications/multimedia/kwave/*

FOSSPicks

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

Graham tears himself away from updating Arch Linux to search for the best new free software. BY GRAHAM MORRISON

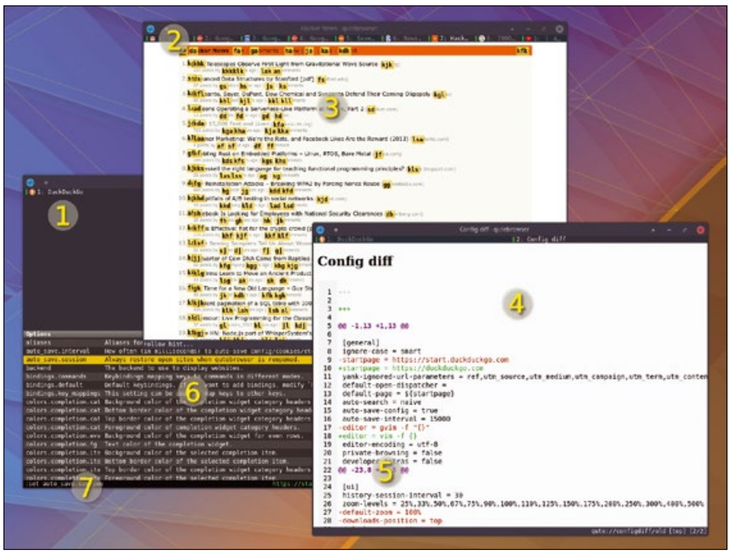
Minimal browser

qutebrowser 1.0

Since starting to use qutebrowser some time ago, it's done nothing less than change the way I think about Linux and user interface design. The minimal, shortcut-driven approach it uses for everything, based on Vim key bindings that many of us are already familiar with, saves you time and mental energy; it helps you navigate the web quickly, removes distractions, and feels amazingly intuitive. Even with only a few Vim commands committed to memory, it's enough to launch new tabs, yank URLs, open hints for links across a page, and save bookmarks. Chromium and Firefox plugins do similar things, but qutebrowser beats these with its level of integration, and also because the entire application is so lightweight. If you've yet to use it, the release of version 1.0 is the best excuse yet to brush up on your Vim skills.

After two successful crowdfunding campaigns to fund development, qutebrowser 1.0 is the rationalization of everything that's gone into development up to this point, and in particular, the re-implementation of its considerable configuration framework. Unfortunately, this means you can't migrate your own settings automatically as you update from an older version. This is important because one of qutebrowser's strengths is that almost anything can be reconfigured through the use of the `:set` command and quickly stored with the `:save` command. However, bookmarks and browsing history are retained from the old version, and a new *diff* view will show you the differences between your old pre 1.0 configuration and the new one. This makes it relatively easy to copy your old settings over to the new format, and the new configuration file is much easier to edit and create yourself than the old one.

The other major addition for the 1.0 release, besides the dozens of small fixes that help to make it much more stable, is that QtWebEngine is now the default backend rendering engine. This means sites like Facebook, GitHub, Gmail, TweetDeck, and Google Docs will work without complaining, and very few sites are now incompatible. Also included is the spell checking that comes with the QtWebEngine, although you'll need to run a Python setup script first to download your required dictionaries and run a configuration command. Afterward, spelling errors will be highlighted just as they are in Chromium, an essential feature for terrible spellers – whether you're firing off a tweet or writing an email – and something you don't realize you rely on until it's missing. Alongside these changes, there are lots of new configuration options to play with and an entirely updated history database, which can now store your entire browsing history or, optionally, a specific number of entries or none at all. And that's the brilliant thing about qutebrowser; not only is it powered by Vim shortcuts (by default) and light on system resources, but it can be reconfigured very easily to perform almost any function you need a browser to perform. The only exception is a plugins interface, which will hopefully come with the next major update.



```
---

+++

@@ -1,13 +1,13 @@

 [general]
 ignore-case = smart
-startpage = https://start.duckduckgo.com
+startpage = https://duckduckgo.com
 yank-ignored-url-parameters = ref,utm_source,utm_medium,utm_campaign,utm_term,utm_conten
 default-open-dispatcher =
 default-page = ${startpage}
 auto-search = naive
 auto-save-config = true
 auto-save-interval = 15000
-editor = gvim -f "{}"
+editor = vim -f {}
 editor-encoding = utf-8
 private-browsing = false
 developer-extras = false
@@ -23,8 [illegible] @@

 [ui]
 history-session-interval = 30
 zoom-levels = 25%,33%,50%,67%,75%,90%,100%,110%,125%,150%,175%,200%,250%,300%,400%,500%
-default-zoom = 100%
-downloads-position = top
```

```
Options
aliases                    Aliases for
auto_save.interval         How often (in milliseconds) to auto-save config/cookies/et
auto_save.session          Always restore open sites when qutebrowser is reopened.
backend                    The backend to use to display websites.
bindings.commands          Keybindings mapping keys to commands in different modes.
bindings.default           Default keybindings. If you want to add bindings, modify `
bindings.key_mappings      This setting can be used to map keys to other keys.
colors.completion.cat      Background color of the completion widget category headers
colors.completion.cat      Bottom border color of the completion widget category head
colors.completion.cat      Top border color of the completion widget category headers
colors.completion.cat      Foreground color of completion widget category headers.
colors.completion.eve      Background color of the completion widget for even rows.
colors.completion.fg       Text color of the completion widget.
colors.completion.ite      Background color of the selected completion item.
colors.completion.ite      Bottom border color of the selected completion item.
colors.completion.ite      Top border color of the completion widget category headers
colors.completion.ite      Foreground color of the selected completion item.
:set auto_save.session
```

1. DuckDuckGo. Press *o* and type a search query. The results will load immediately. Add *!g* to search Google. 2. Tabs: Tabs can be saved, moved, and now pinned. 3. Hints: These quick shortcuts live on your home row and make navigation quick, easy, and mouseless. 4. Config diff: Go to *qute://configdiff* to see your old configuration differences. 5. New config: Configuration has been overhauled for this release. 6. Command completion: Just like Vim, use help and completion to configure and use your environment. 7. Save. Automatically or manually save and restore your session.

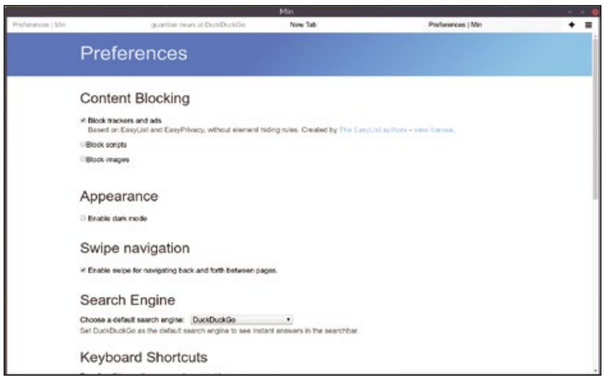
Project Website
https://www.qutebrowser.org/

Minimal browser

Min

We're hopefully entering a new era of web browsing, one where content, privacy, and navigation are the focus, rather than incessant advertising, notification popups, location requests, and videos that automatically play. Members of this vanguard are simple, distraction-less web browsers, such as qutebrowser (see opposite), the wonderful Firefox Focus on Android, and this new browser simply called Min. Min gets this right from the first launch, which sensibly asks whether you want to block ads, trackers, scripts, and even images. You can also enable a very useful dark mode, which even many full-fat browsers can't offer yet.

The browsing experience is equally honed, as you might expect, and the user interface is focused on search results. Start typing almost anywhere, and you'll get suggestions in real time, usually from the very top of the main window. Results are delivered via DuckDuckGo by default, a search engine that finally seems to be finding its own strengths. Multiple tabs can be created, but these can also be grouped into "tasks," which are a great way of managing work and personal pages, for example. Tabs can even be viewed as a list. At the other end of the scale, a distraction-free mode can be enabled to focus on a single tab and disable the creation of any more, which is useful if you don't want to be tempted by Reddit. Plus, there are plenty of keyboard shortcuts and even swipe gestures for navigation. The only slight downside to Min is that it's written atop Electron with JavaScript and CSS, but the speed and efficiency of the application gives none of this away, and the design makes it definitely worth a look.



Preferences

Content Blocking

✓ Block trackers and ads
Based on EasyList and EasyPrivacy, without element hiding rules. Created by The EasyList authors - view license.
☐ Block scripts
☐ Block images

Appearance

☐ Enable dark mode

Swipe navigation

✓ Enable swipe for navigating back and forth between pages.

Search Engine

Choose a default search engine: DuckDuckGo
Set DuckDuckGo as the default search engine to see instant answers in the searchbar.

Keyboard Shortcuts

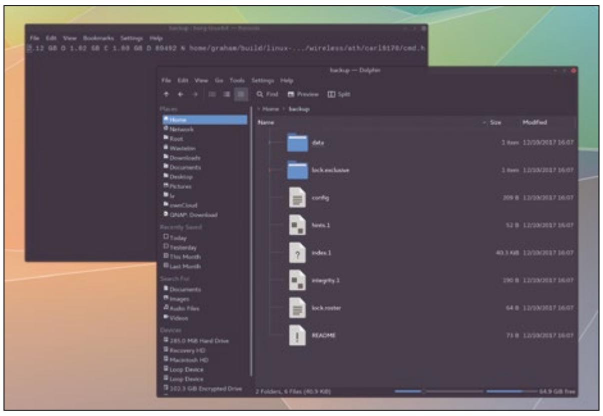
Remove the bloat from your web surfing with a powerful, quick, and minimal browser.

Project Website
https://minbrowser.github.io/min/

Backup

BorgBackup 1.1

BorgBackup is a command-line backup tool, but it has some special abilities that make it a better choice than `tar cvzf`, for example. In particular, it calls itself a "deduplicating backup program." What this means is that it aims to store only the changes in a file, rather than multiple complete iterative duplicates – much as `rsync` does when copying a directory from one location to the next. This is obviously much more efficient on storage space and bandwidth, and many other backup tools use a similar mechanism. Beneath the shell, BorgBackup does this deduplication by splitting a file into chunks, generating a hash for that chunk, and only storing a chunk if its hash hasn't been seen before. This process, combined with client-side encryption and compression, makes BorgBackup an excellent tool that could just pull you away from your "it will never happen" apathy. Unlike `rsync`, BorgBackup couldn't be easier to use. You first initialize a backup repository with the `init` command, as you would with `git`, and then `create` a backup job using a source and destination path. The backup will then proceed silently, unless you ask for `--stats` and `--progress` as additional arguments, and you have many other options to use to fine-tune both the backup and the listing and restoration of files. To create another backup, simply execute the same command, and the incremental backup will start automatically, saving only the changes into the repository. It works excellently, and this release is a major update. More than 60 contributors have helped with new features that include the ability to remove files from an existing archive, automatic compression, more encryption methods, and better logging, plus plenty of speed and security enhancements.



Although the command-line interface is a bit austere, a separate web interface can be installed and used as a graphical user interface.

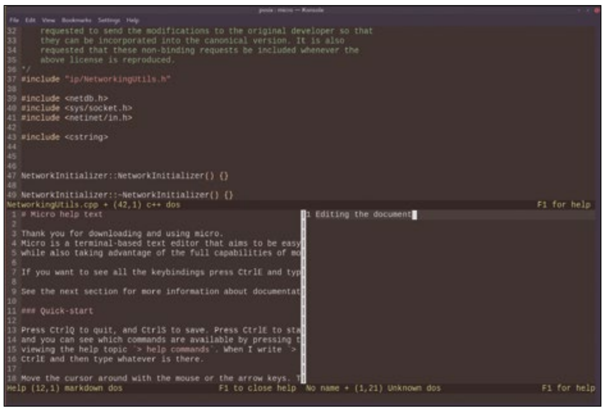
Project Website
https://www.borgbackup.org/

Minimal text editor

micro

Keeping with the minimalist theme, micro attempts to do the same for text editing as Min attempts to do with web browsing. Launching it from the command line gives you the first clue: It runs within your terminal environment, just like Vim or Nano, rather than in an application window like Gedit or Kate. It's also a static binary that can be run in place if you're happy to trust the build; this means you can keep it on a USB stick or email attachment and simply execute this single file when you need its text editing abilities. Like Vim and unlike Nano, though, you can't immediately see how it performs simple operations, such as `quit`, but this is easily discoverable by pressing the F1 key to access the built-in help. How to quit is mentioned in the first paragraph (Ctrl+Q). The help documentation is short, but comprehensive, and even includes a tutorial.

A minimal interface doesn't mean minimal functionality. Split views are easily accessed via the auto-completing command mode, and micro has support for more than 75 languages. The syntax highlighting for these languages appears instantly and happens automatically, depending on the file type you're editing. Even though typing and editing speed is difficult to measure and can often be subjective, micro feels very quick indeed. Because the key bindings can be modified easily, it can be made to feel almost like any familiar editor. LUA-written plugins can even be used to add features, such as code snippets, spelling, in-line editor configuration, and version control changes, and it's easy to write your own. Micro might not replace something like Vim or Emacs, but it's a very useful addition when you need a small editor that doesn't compromise on speed or function. The only editor that gets close to this functionality and form is Textadept.



```
    requested to send the modifications to the original developer so that
    they can be incorporated into the canonical version. It is also
    requested that these non-binding requests be included whenever the
    above license is reproduced.
*/
#include "ip/NetworkingUtils.h"

#include <netdb.h>
#include <sys/socket.h>
#include <netinet/in.h>

#include <cstring>



NetworkInitializer::NetworkInitializer() {}

NetworkInitializer::~NetworkInitializer() {}
NetworkingUtils.cpp + (42,1) c++ dos                                   F1 for help
# Micro help text

Thank you for downloading and using micro.
Micro is a terminal-based text editor that aims to be easy
while also taking advantage of the full capabilities of mo

If you want to see all the keybindings press CtrlE and typ

See the next section for more information about documentat

### Quick-start

Press CtrlQ to quit, and CtrlS to save. Press CtrlE to sta
and you can see which commands are available by pressing t
viewing the help topic `> help commands`. When I write `> 
CtrlE and then type whatever is there.

Move the cursor around with the mouse or the arrow keys. T
Help (12,1) markdown dos          F1 to close help
```

```
Editing the document
No name + (1,21) Unknown dos                                 F1 for help
```

Excellent split view and syntax highlighting functionality in a static binary smaller than 10MB.

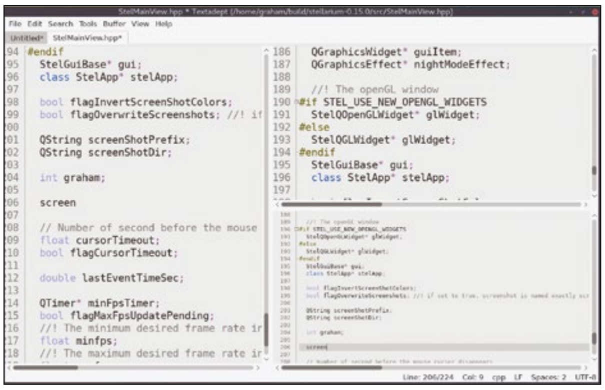
Project Website
https://micro-editor.github.io

Minimalist text editor

Textadept

Not many other software categories offer such breadth of choice as text editors. Of course, the old stalwarts are still causing trouble and discussion, but there's also a constant supply of new pretenders, each attempting a different take on entering one letter at a time. Textadept, like Min and micro, is aiming for the minimalist dollar, promising speed and distraction-free design without sacrificing essential features. Thankfully, it does have a different emphasis from Min, in an attempt to be an editor for programmers. It doesn't support quite as many languages, with around 100 languages currently supported for syntax highlighting purposes, but such large numbers don't really mean much when 90 percent of programming is done with just a handful of languages. And you can do more with those languages in Textadept than you can with Min.

Textadept is certainly mature. It's now 10 years old, with a release every two months for more than six years. It can be run either as a curses binary within the command line or as an application within its own window. The executable is around 5MB and can be run off a USB stick, with a promise to consume a mere 15MB of RAM. Into this tiny space, it's not only capable of cramming in the lighting fast syntax highlighting but, more importantly, code completion, too. For those of us without photographic memories, or perhaps getting on a little, from the time when source code was printed in yellow pages within magazines, this feature is essential. Autocompletion works with symbols within the files you're editing, as well as symbols for the language you're working with, complete with links to the API documentation. It's also almost entirely keyboard driven, completely themable, and well documented.



StelMainView.hpp * Textadept (/home/graham/build/stellarium-0.15.0/src/StelMainView.hpp)
File Edit Search Tools Buffer View Help
Untitled* StelMainView.hpp*

```
#endif
  StelGuiBase* gui;
  class StelApp* stelApp;

  bool flagInvertScreenShotColors;
  bool flagOverwriteScreenshots; //! if

  QString screenShotPrefix;
  QString screenShotDir;

  int graham;

  screen

  // Number of second before the mouse
  float cursorTimeout;
  bool flagCursorTimeout;

  double lastEventTimeSec;

  QTimer* minFpsTimer;
  bool flagMaxFpsUpdatePending;
  //! The minimum desired frame rate ir
  float minfps;
  //! The maximum desired frame rate ir
```

```
  QGraphicsWidget* guiItem;
  QGraphicsEffect* nightModeEffect;

  //! The openGL window
#if STEL_USE_NEW_OPENGL_WIDGETS
  StelQOpenGLWidget* glWidget;
#else
  StelQGLWidget* glWidget;
#endif
  StelGuiBase* gui;
  class StelApp* stelApp;
```

```
Line: 206/224  Col: 9  cpp  LF  Spaces: 2  UTF-8
```

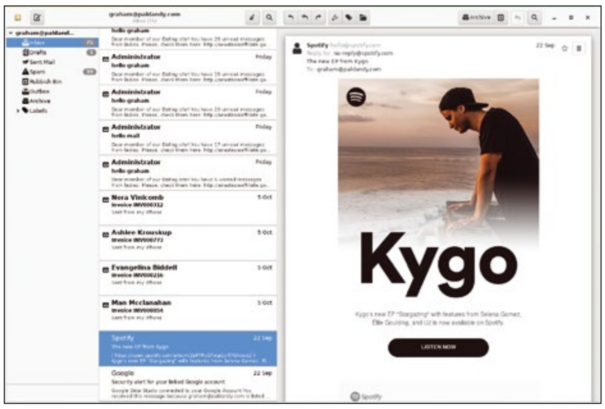
How many editors take less than 15MB and can have variable font sizes in unlimited horizontal and vertical split views?

Project Website
https://foicica.com/textadept/

Email client

Geary 0.12

Offline email clients are still important and should remain important, despite many of us using web-based email clients for our day-to-day email needs. They're essential if you want to back up your email, for example, which is something you should be doing, but they're also useful productivity tools, keeping you away from the temptation of a browser or constant updates. The problem is that the popularity of web-based email clients has come at the cost of desktop clients, with most popular Linux clients going the same way as the awesome Eudora. Geary, after spending months in what felt like suspended animation, is proving the exception with this update, released almost 18 months after version 0.11. It is now part of the Gnome project, which is exactly where a project as good as Geary deserves to be. The best thing about Geary is that it uses Vala/Gtk+, showing the best of Gnome's capabilities. Its slick design is better than the equivalent web client and will display more information, more efficiently. The transitions, for example, are wonderful, and the way the top bar is used as a toolbar and menu holder keeps everything looking clean, even on a KDE desktop with an entirely different window manager. Like Gmail, which you can integrate seamlessly, clever keywords can be used to group and search your inbox, and conversations are grouped vertically, making Geary one of the only offline Linux email applications to do this well. This new update adds lots of much needed stability and improves rich text message composition considerably. The best feature, though, is that you can now press Ctrl+? (Shift+/) to see a crib sheet of shortcuts for whichever mode you're in, helping you learn your way around the application effortlessly.


Geary eases the transition from web email by looking slick and cleverly grouping conversations.

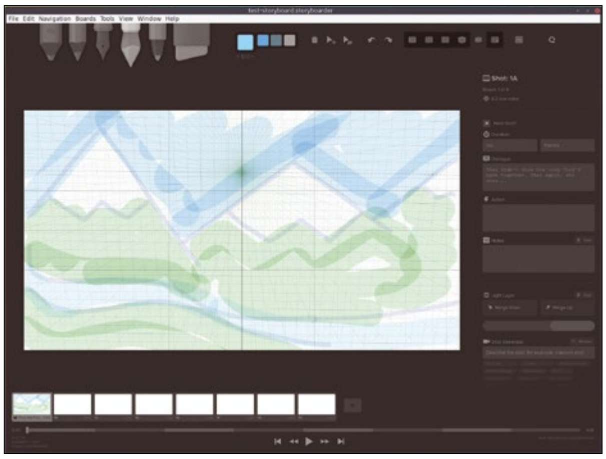
Project Website
https://github.com/GNOME/geary

Story planning

Storyboarder

First, despite this project describing itself as "open source," it's not open source enough to satisfy the Open Source Initiative definition. It's MIT with a few exceptions to stop you making money off the code. These exceptions were motivated by the developer's bad experience with someone forking the project and removing references to the company that developed the software and is perhaps why you need to enter an email address to get a download, even when the code is hosted on GitHub. Despite this, the project is still worth the attention because it's a brilliant way for children in particular to experiment with stories and storyboards, and there's very little software like it for Linux. Because the project is still in its infancy, perhaps with some proper support and discussion, the developer may be convinced to open source the project properly in future.

Storyboarding is the process of sketching out your plan for a story across different pages, creating a primitive cartoon script that a director might use to organize the shots for a film, including composition, locations, and movement. Storyboarder helps you do this with various drawing tools and pens, much as you can with Krita or Gimp, along with additional fields to describe a shot or the dialog. The application does try to keep things simple, by only offering a restricted palette and a few tools, but it's not intended to create the final output. You really just want to move through your ideas in the flow of thought, rather than create a masterpiece out of each frame. You can "play" through the pages to see how it feels and obviously edit and reorder the boards as you see fit. The application has a professional feel, even with its link to Photoshop, and is a great way of planning out a short film or animation project, especially if you're teaching about film or showing children how to plan their next project.


Plan each shot of your next YouTube masterpiece with a great piece of (almost) open source software.

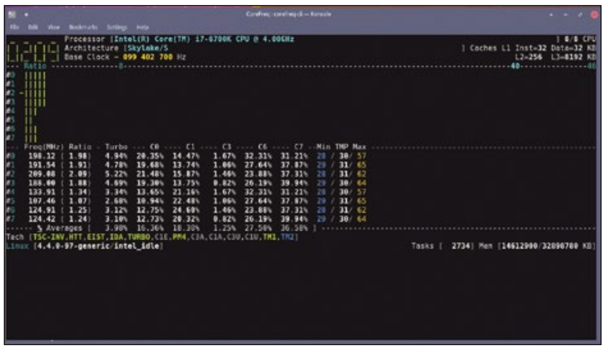
Project Website
https://github.com/wonderunit/storyboarder

CPU monitor

CoreFreq

This wouldn't be FOSSPicks if I didn't look at a performance monitoring tool or two, and this one is rather good. CoreFreq is a CPU performance monitoring tool that specifically targets relatively modern hardware. Your CPU needs to be running a 64-bit operating system and be either an Intel Atom, Core 2, Nehalem, Sandy Bridge, or better. AMD users just need a CPU from the 0Fh family (AMD K8 Hammer) and later. The reason for these requirements is that CoreFreq promises a high degree of precision and is specifically written to monitor modern CPU technologies such as SpeedStep (EIST), Turbo Boost, Hyper-Threading (HTT), and Base Clock. It can also deliver high levels of detail about the code running through your CPU, including the number of instructions per cycle or second, IPS, IPC, or CPI, C-states, thermal monitoring, and the output from various performance counters.

This level of CPU intrusion comes at a cost, and that's mainly paid via the authority required by CoreFreq to run. Not only does it need its own daemon with root credentials, it also needs its own kernel module. This is understandable considering the way the monitor works, but it's worth considering if you're running the monitor on a critical machine. With everything built, installed, and running, the default view looks much like the monitor in htop. Each core gets its own histogram, alongside a table that lists many of the statuses mentioned previously. A menu system can also be used to switch between various modes and display various parameters, and there's a primitive display manager for showing details such as CPU topology and hardware info above the monitor. Additional details can also be viewed directly with additional command-line arguments, making CoreFreq one of the most comprehensive tools for monitoring your CPU that you can install.


```
        Processor [Intel(R) Core(TM) i7-6700K CPU @ 4.00GHz                         ] 8/8 CPU
 0209   Architecture [Skylake/S                                      ] Caches L1 Inst=32 Data=32 KB
        Base Clock - 099 402 700 Hz                                               L2=256  L3=8192 KB
--- Ratio ---------------8-------------------------------------------------------40-----------48
#0  |||||
#1  |||||
#2 -|||||
#3  |||||
#4  |||
#5  ||
#6  |||
#7  |||
--- Freq(MHz) Ratio - Turbo --- C0 ---- C1 ---- C3 ---- C6 ---- C7 --Min TMP Max ----------------
#0   198.12 ( 1.98)   4.94%  20.35%  14.47%   1.67%  32.31%  31.21%  28 / 30/ 57
#1   191.54 ( 1.91)   4.78%  19.68%  13.74%   1.06%  27.64%  37.87%  29 / 31/ 65
#2   209.08 ( 2.09)   5.22%  21.48%  15.87%   1.46%  23.88%  37.31%  28 / 31/ 62
#3   188.00 ( 1.88)   4.69%  19.30%  13.75%   0.82%  26.19%  39.94%  29 / 30/ 64
#4   133.91 ( 1.34)   3.34%  13.65%  21.16%   1.67%  32.31%  31.21%  28 / 30/ 57
#5   107.46 ( 1.07)   2.68%  10.94%  22.48%   1.06%  27.64%  37.87%  29 / 31/ 65
#6   124.91 ( 1.25)   3.12%  12.75%  24.60%   1.46%  23.88%  37.31%  28 / 31/ 62
#7   124.42 ( 1.24)   3.10%  12.73%  20.32%   0.82%  26.19%  39.94%  29 / 30/ 64
------ % Averages [    3.98%  16.36%  18.30%   1.25%  27.50%  36.58% ] -------------------------
Tech [TSC-INV,HTT,EIST,IDA,TURBO,C1E,PM4,C3A,C1A,C3U,C1U,TM1,TM2]
Linux [4.4.0-97-generic/intel_idle]                          Tasks [  2734] Mem [14612900/32890700 KB]
```

CoreFreq displays an unparalleled amount of information about your CPU, as well as the way it's being used by your applications and operating system.

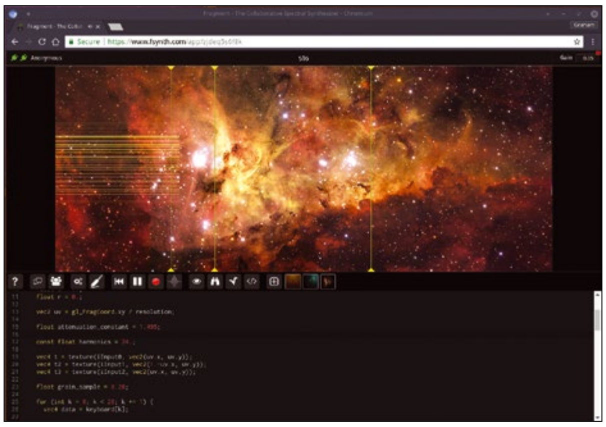
Project Website
https://github.com/cyring/CoreFreq

Granular audio synthesizer

Fragment

Fitting into the niche audio category for this month is Fragment, a "collaborative cross-platform audiovisual live coding environment with pixels based real-time image-synth approach to sound synthesis." To make sense, that sentence really needs to be broken apart and the words looked at individually, but this confusion of words is itself an accurate reflection of the complexity and capabilities of the software. Essentially, Fragment makes sounds, but those sounds are programmed rather than dialed in, and you can share the programming task in real time with collaborators, editing the text as you perform. The audio is generated from pixel data produced by your code, which is pushed through OpenGL Shading Language (GLSL). It's audio/visual because inputs can be images, videos, and sound, essentially shaders in GLSL, and it's granular because the output is generated by taking small samples of those various inputs, processing them, and outputting them again as a cloud of sounds mixed together. Fragment is as complicated as it sounds and especially difficult to get your head around with the little supporting documentation or examples provided. But Fragment "sounds" fantastic, from ethereal pads and ambient abstracted textures to metallic percussion and additive sine waves.

Fragment is slightly unusual in that, although it does run locally, it's built with various web technologies and is accessed via a browser like Chromium or Firefox. You run the code and send notes to the synth using either OSC or MIDI, although Firefox doesn't yet support MIDI. You can then edit and add to the code in real time, and the sound will change. This is the collaborative part, and you can play around with the synth online without having to install it. Load up some images, add them as inputs in the code, and play with the processing.


Use the power of your GPU to generate weird and wonderful sounds.

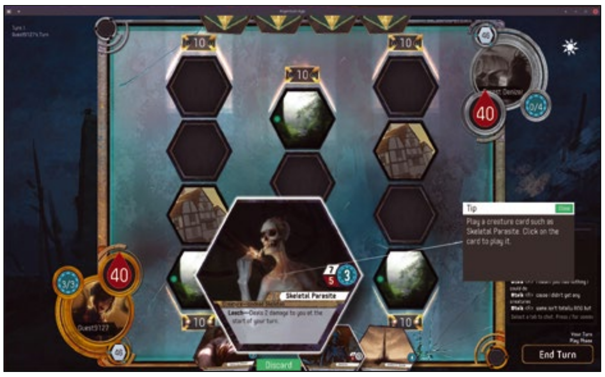
Project Website
https://www.fsynth.com/

Collectable card game

Argentum Age

Despite the incongruity, computer- and tablet-based card game growth appears to match the growth of the physical variety; yet, very few open source card games exist, usually because they depend on excellent assets, and the artists behind those assets aren't typically as well versed in the advantages of open source as developers. Argentum Age's assets are almost an exception, released under a mixture of CC BY-NC-ND, GPL, and CC0, but the assets are more open than many similar games, and content and licensing could change.

Collectable card games (CCG) are a specific genre wherein the player collects and creates their own deck of cards, each card with variable abilities and roles to play, which are then played within the board game's wider rules and context. One card type could be used to summon a specific offensive creature, for example, while another may expand the number of cards you can hold. These attributes are common to most CCGs, but it's the context – the story – and the game rules that make them specific to any one game. In Argentum Age, village size corresponds to the size of your hand, for example. It can be quite complex to get started and to understand many of the rules, but it's also deeply satisfying when you've mastered a specific deck you've tailored to your particular style. One great thing about Argentum Age is that it includes a single-player campaign mode, so you can experiment with your own cards within the confines of an unfolding story. However, you can also take your skills online and pit your deck against real people, complete with inline chat and an entire online community.


Not many CCGs have such beautiful and accomplished artwork.

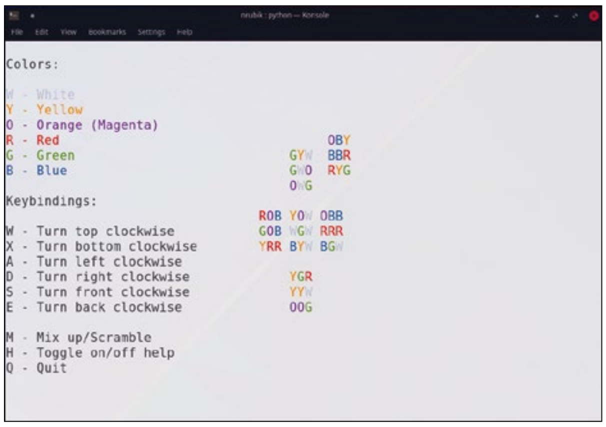
Project Website
http://argentumage.com/

CLI cube

NRubik

Rubik's Cube and its multifarious copies and variations are as popular as ever, because it's perhaps one of the best toys invented. Oddly satisfying to move, seemingly accessible to start, and yet challenging to solve. Even when you've mastered the basics of solving the cube, you quickly get obsessed with shortening your time, perhaps starting with five minutes with an aim to get from random to solved in less than 30 seconds. All solutions require good memory, problem solving, dexterity, and perseverance, and this is why perhaps the cube is still so popular. With so many distractions, focusing on solving a cube for a few minutes is a wonderful reset switch, and while the physical cube is always likely to be best, virtual recreations in software present their own challenges as you try to transpose your knowledge onto a two-dimensional (2D) projected image that is much more mentally challenging than the original.

Several good cube applications for the desktop range from 2D to OpenGL-based 3D, which can be the easiest to use, as you drag your mouse across the various layers to mimic the movement of your fingers. The problem is that you usually can't see the reverse sides very quickly. This isn't a problem with NRubik, however, because it runs from the command line, using curses to display a simple planar view of a virtual cube. All the controls are displayed onscreen and allow you to turn each row, as well as mix up the cube. These controls are all you need to create a solution. Although it's difficult even with a memorized set of sequences, the challenge is perfectly in keeping with what makes the challenge of the cube so long lived.


nrubik : python — Konsole
File Edit View Bookmarks Settings Help

```
Colors:

W - White
Y - Yellow
O - Orange (Magenta)
R - Red                                      OBY
G - Green                                GYW BBR
B - Blue                                 GWO RYG
                                         OWG
Keybindings:
                                     ROB YOW OBB
W - Turn top clockwise               GOB WGW RRR
X - Turn bottom clockwise            YRR BYW BGW
A - Turn left clockwise
D - Turn right clockwise                 YGR
S - Turn front clockwise                 YYW
E - Turn back clockwise                  OOG

M - Mix up/Scramble
H - Toggle on/off help
Q - Quit
```

Don't let the click of turning a real cube stop you from solving this immortal puzzle in the office.

Project Website
https://github.com/cheertarts/nrubik

Video Wizardry

Linux has some excellent graphical video-editing tools, but sometimes working from the command line with FFmpeg is just better.

BY PAUL BROWN

How much better? Well, it makes stuff easier to batch process, for starters. Say you have to change every instance of "Bill" in a 100-page text file to "Gary." Sure, you could use the search-and-replace feature in your text editor. That would work if you only had one file, but what would you do if you had a filesystem with hundreds of files scattered all over the place? You would never consider seriously trawling through every directory and subdirectory, opening each file in turn, and clicking through the search-and-replace process, would you? A Bash script using `find` and `sed` would be the way to go.

The same goes, believe or not, for video editing. You can do dozens, nay, scores of things with your videos, without ever having to open a graphical video-editing application. All you need is FFmpeg [1].

You've probably used FFmpeg before for converting video and audio files between formats. In its simplest form, that is what it does. The instruction

```
ffmpeg -i input.mp4 output.webm
```

converts an MP4 video file into a WebM video file.

However, FFmpeg can do much more than that. It can be used to change the frame rate, switch in and out audio and subtitle tracks, and even cut up and re-arrange sequences within a movie.

Inserting a Watermark

One of the most powerful FFmpeg features is its effects pipeline [2], or "filters," as they are known by FFmpeg users.

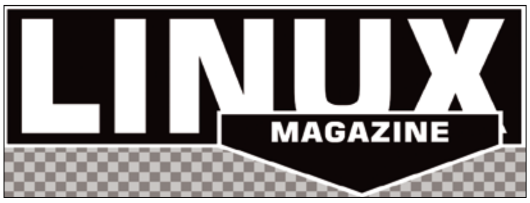


Figure 1: The logo for watermarking your video.

You can apply filters to whole audio or video streams or only to certain parts, use them to merge several streams into one in interesting ways, and do much more.

To illustrate how filters work, I'll show you how to use a logo to watermark a video. You can see the logo in Figure 1. It is a PNG with a transparent background. I'll assume the video you'll be using is a 720p (1280x720) MP4 video called `example.mp4`.

There are several ways you can carry out this task, but the FFmpeg filter page mentions the `overlay` filter, and that seems to be the most straightforward way to go.

In the instruction

```
ffmpeg -i example.mp4 -i LM_logo.png ⮐
       -filter_complex "overlay" ⮐
       -codec:a copy example_marked.mp4
```

FFmpeg takes two inputs, the `example.mp4` video file and the `LM_logo.png` file, and outputs them together – the second placed on top of the first – to `example_marked.mp4`. Figure 2 shows the result.

Of interest is the `-filter_complex` construct, which sits between the inputs and the output. Within `-filter_complex`, you can string filters together, and they will be applied one after the other to one stream, the other, or both.

Although this is a step in the right direction, the result isn't very subtle. The logo is in the wrong place. Instead of the upper-left corner, it would be better in the lower right, like most channel logos on TV.

Fortunately, most filters can take parameters, and `overlay` can too:

```
ffmpeg -i example.mp4 -i LM_logo.png ⮐
       -filter_complex "overlay=W-w-10:H-h-10" ⮐
       -codec:a copy example_marked.mp4
```

When you pass a parameter to a filter, you do so using the `<filter>=<value>` syntax. In this case, you pass to `overlay` the horizontal position and then the vertical position, separated by a colon (`:`), of the top layer (containing the logo).

FFmpeg also provides a convenient way to pass the width and height of each layer to the `overlay` filter: `W` is the width of the first input (the bottom layer), and `w` is the width of the second input (the top layer). This means that `W-w-10` will place the top overlay layer 10 pixels from the left-most edge of the bottom video layer. The same goes for `H-h-10`, but in the vertical axis (see Figure 3).

However, the logo is still way too big. You can solve this by adding a new filter and chaining it to `overlay`:

```
ffmpeg -i example.mp4 -i LM_logo.png ⮐
       -filter_complex "⮐
        [1:v] scale=150:-1 [ol], ⮐
        [0:v] [ol] overlay=W-w-10:H-h-10" ⮐
       -codec:a copy example_marked.mp4
```

Several new things are going on here. First, notice the new `scale` filter, which changes the scale of a video stream by taking a new width and height separated by a colon. If you want to make sure FFmpeg keeps the proportion correct, pass one of the parameters and then `-1` as the other. In the example above, you tell `scale` to make the stream 150 pixels wide and to scale its height proportionally.

But, what video stream are you talking about? The FFmpeg instruction above has two inputs: `example.mp4` and `LM_logo.mp4`. How do you specify which one you want to scale? Well, that is the purpose of `[1:v]`. In the `filter_complex` string, you can specify the input on which you want to operate with a number and the type of stream. Inputs start from `0`, so the first input (`example.mp4`) is `0`, and the second input (`LM_logo.png`) is `1`. The letter tells the filter on what kind of stream it should operate. A PNG image only has a visual/video component, so you tell `scale` to use `[1:v]`. It is totally possible that other types of inputs have more components. For example, the `example.mp4` input has video and audio components. To apply an effect to the audio, you would use `[0:a]`; if it has built-in subtitle tracks to which you want to apply an effect, you would use `[0:s]`, and so on.

At this stage, it is worth mentioning that FFmpeg allows you to label your streams, which is what `[ol]` is doing: You apply a scaling effect to `[1:v]`, and then (so you can refer to the scaled stream later and not the original input) you give it a name: `[ol]` (for *overlay layer* – the name can be anything you want).

As you can see in the fourth line of the code above, you use the video stream from the first input (`[0:v]`) and overlay the scaled image (`[ol]`). The comma separating the `scale` and `overlay` filters indicates that the output from the first element of the string is piped to the second element, which means you could have used

```
[0:v] overlay=W-w-10:H-h-10
```

instead of

```
[0:v] [ol] overlay=W-w-10:H-h-10
```

and the result would have been the same.

However, as your `filter_complex` string becomes more complex, you will discover that labeling streams is not only a helpful memory aid, but also essential to achieving the result you desire.

The end result is that `LM_logo.png` is shrunk down to a reasonable size and then overlaid in the bottom right-hand corner of the frame, as shown in Figure 4.

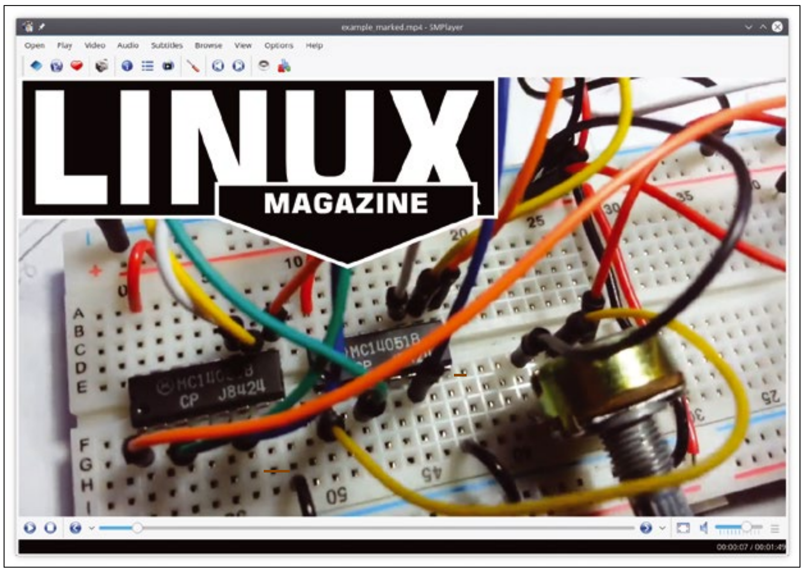

example_marked.mp4 - SMPlayer
Open Play Video Audio Subtitles Browse View Options Help
LINUX MAGAZINE
00:00:07 / 00:01:49

Figure 2: The movie marked with a gigantic watermark.

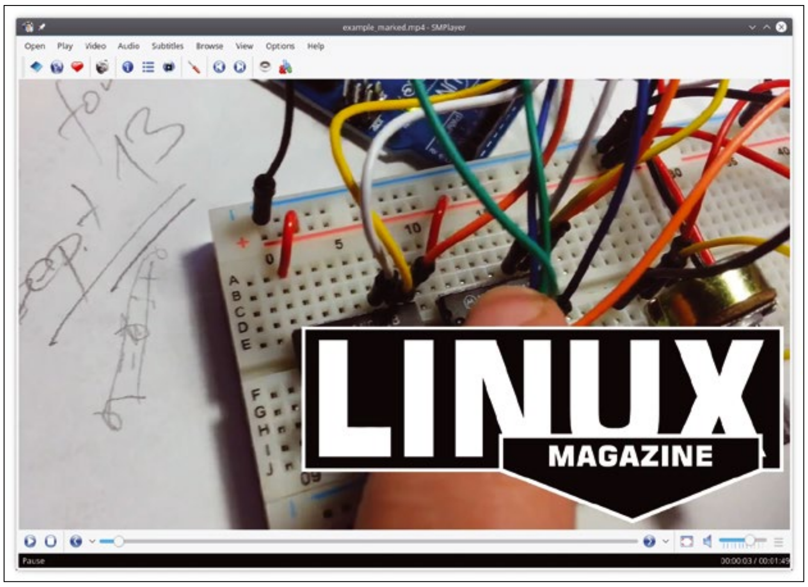

example_marked.mp4 - SMPlayer
Open Play Video Audio Subtitles Browse View Options Help
LINUX MAGAZINE
Pause
00:00:03 / 00:01:49

Figure 3: You can place your logo by passing the *x* and *y* position as parameters to `overlay`.

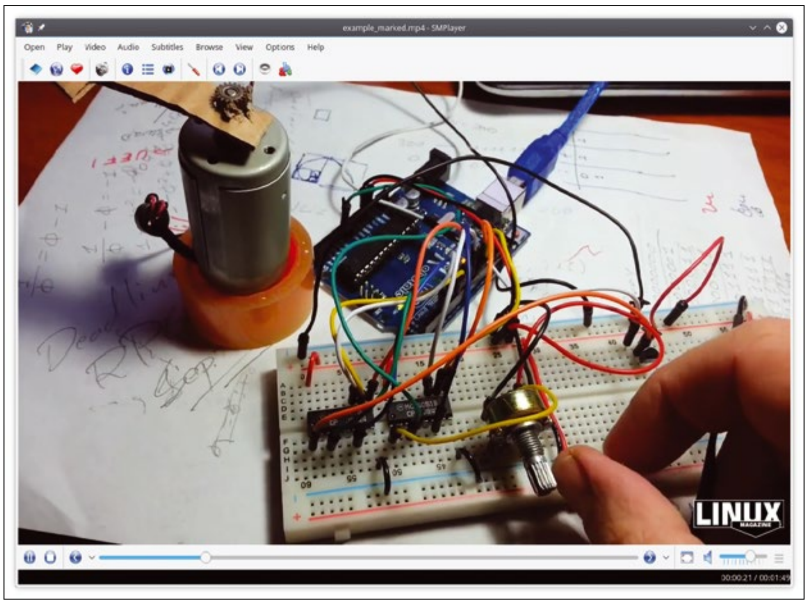

example_marked.mp4 - SMPlayer
Open Play Video Audio Subtitles Browse View Options Help
LINUX MAGAZINE
00:00:21 / 00:01:45

Figure 4: The logo is now placed and scaled thanks to the `overlay` and `scale` filters.

Blending In

Overlaying is fine if you are okay with an opaque logo obscuring part of your video, but if you want something that will not keep your viewers from seeing all the action, a translucent logo is the way to go.

To do this, you have to shift gears and use the `blend` filter, instead of `overlay`, which lets you apply different kinds of blending algorithms to your layers. You can use addition, subtraction, XOR, and so on, which makes `blend` much more versatile than `overlay`.

The caveat, though, is that `blend` requires the streams you are merging to be the same resolution and have the same storage aspect ratio (SAR), which is the dimension of the pixels that make up a frame expressed as a ratio. In modern digital formats, pixels are usually perfectly square; that is, they have a ratio of 1:1. If you are not sure of your clips' SARs, you can use FFmpeg's `ffprobe` command to find out what they are:

```
ffprobe example.mp4
```

The output returns a resolution of 1280x720 pixels and a SAR of 1:1, as expected.

The command

```
ffprobe LM_logo.png
```

returns a resolution of 800x314 pixels and a SAR of 2515:2515, which means you have to increase the resolution of the top layer to 1280x720 pixels and, although 2515:2515 is the same as 1:1, FFmpeg doesn't know that, so you also will have to correct the SAR.

If you simply scale up your logo and change its SAR with `setsar=sar=1`, as in Listing 1 (one command broken into multiple lines), it works, but you get what you see in Figure 5.

Even if what you see in Figure 5 is not what you want, take a look at the code, because it introduces several new and interesting features. First, notice you have three blocks of filter chains on lines 3, 4, and 5 separated by semicolons. A semicolon between blocks of filters indicates that each block is not related to the next, because line 3 applies filters to the logo (input 2), line 4 applies filters to the video (input 1), and line 5 is where the result of both filtered streams are merged.

More specifically, line 3 scales the logo (input 2) up to 1280x720 pixels and changes its SAR to `1`. Line 4 makes sure the video's SAR is correct by setting it to `1`, as well, and converts each frame to the RGBA color space, so they can be melded with the logo without causing any strange color effects. Finally, on line 5, you use the `blend` filter on the output from lines 3 and 4 and convert the blended layers to a video-friendly color space.

The `blend` filter can take more than one parameter. In fact, it can take more than a dozen [3], so instead of just placing the parameters in order one after another and separating them by colons, as was done with `scale`, you will want to refer to the name of the parameter explicitly to avoid becoming confused. You do this by pairing off each parameter name with its value, as shown in Figure 6.

Listing 1: Translucent Logo

```
01 ffmpeg -i example.mp4 -i LM_logo.png
02   -filter_complex "
03     [1:v] scale=1280:720, setsar=sar=1 [lo];
04     [0:v] setsar=sar=1, format=rgba [bg];
05     [bg][lo] blend=all_mode=addition:all_opacity=0.5, format=yuva422p10le
06     "
07   -codec:a copy example_marked.mp4
```

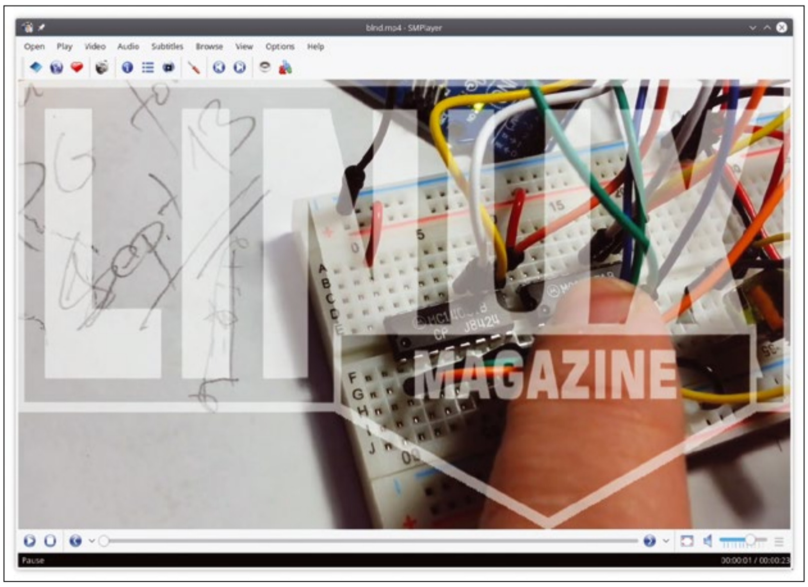


Figure 5: A translucent logo blended into the video.

In this case, you are passing an option that specifies how you want to merge each pixel using the `blend` parameter `all_mode` and telling it what the opacity of each pixel has to be (0.5 is 50% opaque) with the `blend` parameter `all_opacity`.

Notice how you are now confidently using custom labels to describe each filtered input by using `[bg]` for the background video and `[lo]` for the logo overlay.

As mentioned earlier, this is not exactly the desired outcome. Again, you want the logo to be down at the bottom right-hand corner tucked away inconspicuously, not splattered all over the video obscuring the action.

Fortunately, FFmpeg provides yet another filter that helps with this problem: `pad` [4] allows you to resize the frame around the image, filling in the new space with a color or alpha transparency. Listing 2 shows how this works.

All the changes happen on line 3, where you resize the logo to make it smaller with the `scale` filter you used before. Then you create a "padding" around it, to make the frame 1280x720 pixels. As with `overlay`, you can decide where to place the original image within the padded frame, either by using numbers or playing with the built-in variables: `ow` (the padded width of the frame), `iw` (the original width of the image), `oh` (the padded height of the frame), and `ih` (the original height of the image). As with the `overlay` example, `ow-iw-10:oh-ih-10` places the logo 10 pixels to the left of the frame's rightmost edge and 10 pixels up from the frame's bottom-most edge.

Finally, to make sure nothing funny happens to the colors when you merge the logo layer with the video layer, you convert the color space of the padded frame to `rgba` with the `format` filter.

Figure 7 shows the outcome of running this instruction, which is exactly what you want.

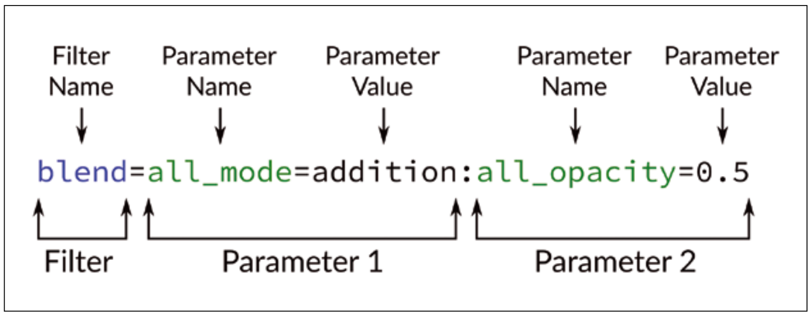


Figure 6: Passing several parameters with their names to a filter.

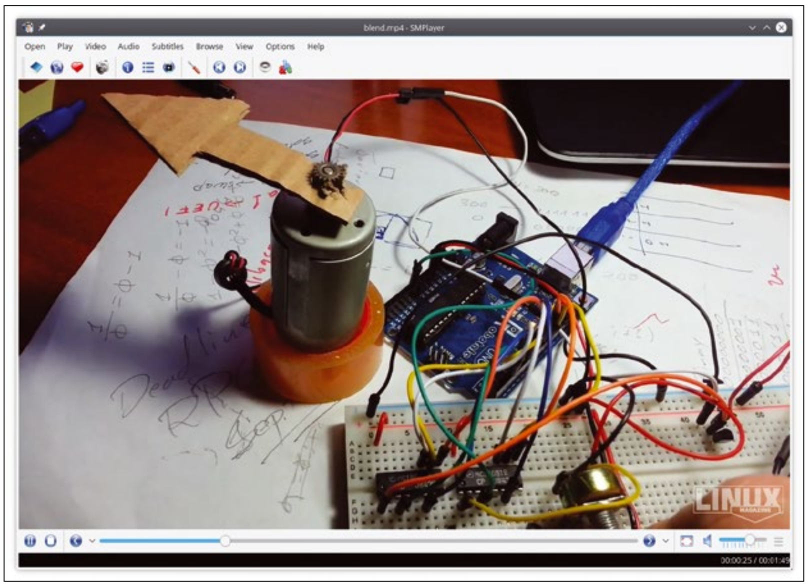


Figure 7: An overlaid translucent logo gives a touch of class to your videos.

Conclusion

All of these command-line manipulations might seem like overkill, and there is no denying FFmpeg's steep learning curve, to say the least, but that is in part because video editing is a complex art.

However, the payback is immense. By handing off trivial and repetitive tasks to FFmpeg, you can avoid having to run power-hungry graphical applications and wasting time manually placing and filtering and then rendering your clips.

FFmpeg is also very mature at this stage and is probably much stabler than most graphical video editors, which means you will avoid the frustration of crashing apps. Because it is a command-line tool, it allows you to batch process scores of videos in one go, with no need for human supervision. It also allows you to ship this kind of cycle-consuming work off to a headless server, freeing up your workstation for more important things, like playing games or browsing the web.

Regardless of how you look at it, using FFmpeg for automated video editing is win-win-win. ■■■

Listing 2: Positioned and Scaled

```
01 ffmpeg -i example.mp4 -i LM_logo.png
02        -filter_complex "
03         [1:v] scale=150:-1, pad=1280:720:ow-iw-10:oh-ih-10, setsar=sar=1, format=rgba [lo];
04         [0:v] setsar=sar=1, format=rgba [bg];
05         [bg][lo] blend=all_mode=addition:all_opacity=0.5, format=yuva422p10le
06         "
07        -codec:a copy example_marked.mp4
```

Info

[1] FFmpeg: *http://ffmpeg.org/*

[2] List of FFmpeg effects: *https://ffmpeg.org/ffmpeg-filters.html*

[3] The many parameters of `blend`: *https://ffmpeg.org/ffmpeg-filters.html#blend_002c-tblend*

[4] The `pad` filter: *https://ffmpeg.org/ffmpeg-filters.html#pad-1*

FEATURED EVENTS

Users, developers, and vendors meet at Linux events around the world. We at *Linux Magazine* are proud to sponsor the Featured Events shown here.

For other events near you, check our extensive events calendar online at *http://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*.

Black Hat Europe

Date: December 4–7, 2017

Location: London, England

Website: *https://www.blackhat.com/*

Tune in to the very latest in research, development, and trends in information security over four days – two days of deep technical hands-on Trainings (skill building for both offensive and defensive hackers) and two days of the research and vulnerability disclosures in Briefings (latest information on security risks and trends).

Embedded Linux Conference

Date: March 12–14, 2018

Location: Portland, Oregon

Website: *http://events.linuxfoundation.org/events/embedded-linux-conference*

The Embedded Linux Conference is a vendor-neutral technical conference for companies and developers using Linux in embedded products. ELC has a large collection of sessions dedicated exclusively to embedded Linux and embedded Linux developers.

Open Networking Summit

Date: March 26–29, 2018

Location: Los Angeles, California

Website: *http://events.linuxfoundation.org/events/open-networking-summit-north-america*

Join business and technical leaders across enterprise, cloud, and service providers to share information, highlight innovation, and discuss the future of open networking and orchestration.

Events

Black Hat Europe	December 4–5	London, England	https://www.blackhat.com/
LinuxLab 2017	December 6–7	Florence, Italy	https://2017.linux-lab.it/
KubeCon and CloudNativeCon North America	December 6–8	Austin, Texas	http://events.linuxfoundation.org/events/kubecon-and-cloudnativecon-north-america
AGL Showcase at CES 2018	January 9–12, 2018	Las Vegas, Nevada	http://events.linuxfoundation.org/events/agl-showcase-at-ces-2018
Enigma 2018	January 16–18, 2018	Santa Clara, California	https://www.usenix.org/conference/enigma2018
FAST '18	February 12–15, 2018	Oakland, California	https://www.usenix.org/conference/fast18
Open Source Leadership Summit	March 6–8, 2018	Sonoma Valley, California	http://events.linuxfoundation.org/events/open-source-leadership-summit
Embedded Linux Conference	March 12–14, 2018	Portland, Oregon	http://events.linuxfoundation.org/events/embedded-linux-conference
OpenIoT Summit	March 12–14, 2018	Portland, Oregon	http://events.linuxfoundation.org/events/openiot-summit
NSDI '18	April 9–11, 2018	Renton, Washington	https://www.usenix.org/conference/nsdi18
heise Security Tour 2018	April 10, 12, 18, 24, and 26, 2018	Numerous European cities	https://www.heise-events.de/securitytour
Open Networking Summit	March 26–29, 2018	Los Angeles, California	http://events.linuxfoundation.org/events/open-networking-summit-north-america
2018 HPC for Wall Street	April 16, 2018	New York, New York	http://www.flaggmgmt.com/linux/
Cloud Foundry Summit North America	April 18–20, 2018	Boston, Massachusetts	https://www.cloudfoundry.org/event/nasummit2018/

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.

NOW PRINTED ON recycled paper from 100% post-consumer waste; no chlorine bleach is used in the production process.

Authors

Konstantin Agouros	46
Erik Bärwaldt	30
Swapnil Bhartiya	8, 58
Mario Blättermann	82
Paul Brown	92
Zack Brown	11
Bruce Byfield	54, 70
Joe Casad	3
Mark Crutch	75
Matthew Garrett	26
Jon "maddog" Hall	77
Charly Kühnast	53
Eva-Katharina Kunst	20
Christoph Langner	50, 64
Vincent Mealing	75
Pete Metcalfe	60
Graham Morrison	86
Simon Phipps	76
Jürgen Quade	20
Mike Saunders	78
Mike Schilli	42
Ralf Spenneberg	14
Ferdinand Thommes	38

Contact Info

Editor in Chief
Joe Casad, jcasad@linux-magazine.com
Managing Editor
Rita L Sooby, rsooby@linux-magazine.com
Localization & Translation
Ian Travis
News Editor
Swapnil Bhartiya
Copy Editor
Amy Pettle
Layout
Dena Friesen, Lori White
Cover Design
Dena Friesen
Cover Image
© Maksym Yemelyanov and Chainat, 123RF.com
Advertising – North America
Ann Jesse, ajesse@linuxnewmedia.com
phone +1 785 841 8834
Advertising – Europe
Brian Osborn, bosborn@linuxnewmedia.com
phone +49 89 99 34 11 48
Publisher
Brian Osborn, bosborn@linuxnewmedia.com
Marketing Communications
Gwen Clark, gclark@linuxnewmedia.com
Linux New Media USA, LLC
616 Kentucky St.
Lawrence, KS 66044 USA

Customer Service / Subscription
For USA and Canada:
Email: cs@linuxpromagazine.com
Phone: 1-866-247-2802
(Toll Free from the US and Canada)
Fax: 1-785-856-3084

For all other countries:
Email: subs@linux-magazine.com
Phone: +49 89 99 34 11 67

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.

Printed in Nuremberg, Germany by hofmann infocom GmbH on recycled paper from 100% post-consumer waste; no chlorine bleach is used in the production process.

Distributed by Seymour Distribution Ltd, United Kingdom

LINUX PRO MAGAZINE (ISSN 1752-9050) is published monthly by Linux New Media USA, LLC, 616 Kentucky St., Lawrence, KS, 66044, USA. Periodicals Postage paid at Lawrence, KS and additional mailing offices. Ride-Along Enclosed. POSTMASTER: Please send address changes to Linux Pro Magazine, 616 Kentucky St., Lawrence, KS 66044, USA.

Published monthly in Europe as Linux Magazine (ISSN 1471-5678) by: Sparkhaus Media GmbH, Zieblandstr. 1, 80799 Munich, Germany.

Issue 207 / February 2018

Compilers

Approximate

UK / Europe	Jan 13
USA / Canada	Feb 09
Australia	Mar 12

On Sale Date

Software development as we know it today wouldn't exist without the amazing tool that is the real interface between the human and computer. Next month we look at compilers.

Image © pratyaksa, 123RF.com